ESI ISCN

Practical Improvement of Software Processes and Products

ESI-ISCN '95 : Measurement and Training Based Process Improvement
September 11-12, Vienna, Austria

ALCATEL

CMS - BRITISH STEEL PLC

BRAMEUR SZTAKI SIEMENS

OBJECTIF TECHNOLOGIE

UNIVERSITY OF MARYLAND

ETNOTEAM PROCESS INC

FESTO

European Software Process Improvement Training Initiative

Approaches

ami provides an overall improvement framework based on the quantitative approach "Assess-Analyse-Metricate-Improve". In ISCN' 95 ami will evaluate how the improvement paradigm works using different assessment methodologies.

Bootstrap is a method for assessing and quantitatively evaluating process quality attributes. In ISCN' 95 a project combining ISO and Bootstrap to an auditing method will be presented.

EQA The European Quality Award provides a guideline and a questionnaire for self-assessment within an organisation and to compare these results internationally. In ISCN' 95 Alcatel will present it's experience with Total Quality Management and the use of EQA as a tool to achieve this.

Metkit is a set of integrated training courses that demonstrate how to implement process improvement programmes and how to measure processes and their products.

Scope has developed procedures and a guide for the evaluation of software product quality. In ISCN' 95 Scope will present a statistical analysis of the operational profile of systems installed in field.

SPICE develops an ISO standard for software process improvement and capability evaluation including a baseline practices guide, a process assessment guide, and a capability determination guide.

TickIT procedures ensure that ISO 9001 audits of software development organisations are performed by IT-competent auditors working for TickIT accredited certification bodies. Guidance material is published in the "TickIT Guide"

Active Integration

Seminar participants are entitled to submit with their registration one page of questions, issues, problem statements or experiences for inclusion in the workshop.

Organisations

ESI The European Software Institute is a major industry initiative dedicated to supporting its member and European industry in improving their customer's competitiveness by promoting and disseminating best practices.

ISCN The main goal of the International Software Consulting Network is to set up a resource pool of experts involved in well known process improvement projects and initiatives such as ami, Bootstrap, Metkit, Scope, Spice, TickIT, and to establish teams of experts from these projects.

The ISCN Co-ordination Office has established formal procedures for new membership, team selection project co-ordination and network maintenance. Each expert's details are stored in an expert skill database based on a standard expert skill profile. Each customer describes his requirements based on a standard customer requirement report. The database supports the mapping of customer requirement data onto expert skill data.

ISCN guarantees that the experts they promote are highly skilled, well qualified, experienced consultants. Details of ISCN membership is available from the ISCN Co-ordination office.

Who Should Attend and Why?

The ISCN conference series provides continuous information about the evolution of different European strategies, projects, and the experience of users with regard to the implementation of process improvement practices.

Participants will see the process improvement field from three different aspects: Strategic European Programmes, Methodologies, and Industrial Experienc The ultimate goal is that all participants learn how the methodologies can be implemented in their own organisation and what the benefit in measurable terms will be.

The workshop will enable participants to discuss with the experts and the industrial users how to utilise the methodologies to achieve the optimum benefits.

Es wird ersucht, diese für eine Person gültige Einladung
beim Eintritt in das Rathaus unbedingt vorzuweisen.

Please note: This invitation is valid for one person only
and presentation of card is necessary for admittance to City Hall.

Cette invitation valable pour une personne sera à présenter
à l'entrée de l'Hôtel de Ville.

Der Bürgermeister der Bundeshauptstadt Wien

beehrt sich, für

Montag, den 11. September 1995, um 20 Uhr

anlässlich der

ESI-ISCN '95 Konferenz

zu einem

Cocktailempfang

im Wappensaal des Wiener Rathauses

einzuladen

Zugang: 1, Lichtenfelsgasse 2, Feststiege II

Conference Chair : Takis Katsoulakos

Chairman of the First Day´s Proceedings: Günter Koch

08:30	Registration	
09:10	Official Opening	
	Improvement Strategies and Programmes	
09:30	ESI: European Strategies for Software Process and Product Improvement	Hans Jürgen **Kugler** ESI
10:00	ESSI: Objectives, Strategy and Implementation of Software Best Practice Actions	Mechthild **Rohen** EC DG III
10:30	ESPITI: A European-wide Training Initiative	Kevin **Eakin** MARI Northern Ireland
11:00	**Coffe Break**	
11:15	BOOTSTRAP Institute: European Overview of Quality Improvement	Adriana **Bicego** ETNOTeam, Italy
11:45	Leonardo - The Networking Programme	John **Stewart** Hibernia UETP, Ireland
12:15	**Lunch**	
	Projects and Methodologies	
13:45	AINSI - An Inductive Method for Software Process Improvement	Lionel **Briand** CRIM, Canada
14:15	The Evolution of a Quantitative Process Analysis - the BOOTSTRAP - Approach	Richard **Messnarz** TU Graz, Austria
14:45	SPICE: The Software Process Assessment Model	Pasi **Kujava** University of Oulu
15:15	ami: A Taylorable Framework for Software Process Improvement	Christophe **Debou** Alcatel, Belgium
15:45	**Coffee Break**	
16:00	TickIt: Providing Confidence in ISO 9001 Certification	Alec **Dorling** ESI
16:30	Process Improvement - How to Measure It	Mike **Kelley** Brameur UK
17:00	Scope: Improving Software Quality Through QuantitativeEvaluation of Products and Processes	Gualtiero **Bazzana** Onion Srl, Italy
17:30	Object-Oriented Software Measures	Horst **Zuse** University of Berlin
18:00	Soap-box: The PERFECT Approach to Continous Improvement	Christer **Mattsson** Q-Labs, Sweden
20:00	**Reception with Lord Mayor of Vienna**	

Conference Chair : Takis Katsoulakos

Chairman of the First Day´s Proceedings: Günter Koch

Organising Committee

Co-ordinating Board	**Richard Messnarz**	**(IST, ISCN)**
	Debbie Penney	**(ISCN)**
Support	**Andreas Bolin**	**(IST, ISCN)**
	Tina Glaser	**(IST)**
	Werner Kerschenbauer	**(IST, ISCN)**
	Michael Loidl	**(ISCN)**
	Markus Pöschl	**(ISCN)**

Programme Committee

Chair: Richard Messnarz (Austria)

Micheal Mac an Airchinnigh	**(Ireland)**
Gualtiero Bazzana	**(Italy)**
Adriana Bicego	**(Italy)**
Martin Brett	**(Germany)**
Christophe Debou	**(Belgium)**
Alec Dorling	**(Spain)**
Hans Jürgen Kugler	**(Spain)**
Pasi Kuvaja	**(Finland)**
Jean Martin Simon	**(France)**
Christer Mattsson	**(Sweden)**
Colin Tully	**(UK)**
Günter Waldner	**(Austria)**

Main Organisers

ESI	**European Software Institute, Spain**
I.S.C.N.	**International Software Consulting Network, Ireland**
IST	**Institute for Software Technology, Austria**

Supporting Organisations

Alcatel
***ami* User Group**
APAC GmbH, Austria
APS Leonardo, Austria
Brameur, UK
CCC Software Professionals, Finland
Centre de Recherche Public Henri Tudor, Luxembourg
Colin Tully Ass., UK
Center for Software Engineering, Regional ESPITI Organiser, Ireland
Etnoteam, Italy
Hibernia Learning Partnership, Ireland
INESC, Portugal
Intersoft, Greece
Leansoft Oy, Finland
Mari Northern Ireland
Network for Women in Technology in Europe (WITEC)
Österreichischer Verband der Wirtschaftsingenieure, Austria
Q-Set Ltd., Ireland
Research Center Seibersdorf, Regional ESPITI Organiser, Austria
SERC, Netherlands
University of Iceland
University of Linz, Regional ESPITI Organiser, Austria
University of Maryland, USA
Unix User Group Austria

ESI

ESI: European Strategies for Software Process and Product Improvement

Hans-Jürgen Kugler

Director of Process Improvement

European Software Institute
Bilbao, Spain

Abstract

This paper describes the mission and work of the European Software Institute (ESI) against the background of a recent analysis of the state of practice in the European software industry.

Opportunities of working with ESI are identified.

1. BACKGROUND

The Bangemann report [EC 94] identifies three major 'network structures' as being of crucial importance to the development of European industry:

- transport,
- energy, and

- information.

The creation and operation of each of these networks within effective and economically feasible boundaries relies on software; the so-called information highways even are by their very nature "made of software." Since software is increasingly becoming an enabling factor there are many strategic initiatives to improve the competitiveness of European industry with respect to software. The European Strategic Programme for Research and Development in Information Technologies (ESPRIT), the European Systems and Software Initiative (ESSI), the Advanced Communications Technologies and Services programme (ACTS), the Telematics Applications programme etc. are names for just a few of the transnational initiatives implemented by the European Commission. The European Software Institute (ESI) is an industry-driven initative in just this field. This paper concerns itself with ESI as one such initiatives, and its relation to other activities in the software arena.

2. THE EUROPEAN SOFTWARE INSTITUTE

The European Software Institute (ESI) is a membership-based non-profit organisation founded as an industry joint venture with support from the European Commission and the Basque Government. Its mission is to help improve the competitiveness of its members through software. This includes both improved software production (software producers) and improved deployment (software users) — ESI's members come from both segments of the European software industry.

Improvements leading to increased competitiveness deal with the three major variables in product or service creation and delivery:

- time to market,
- productivity, and
- quality.

The main working assumption of ESI is that improving the software process leads to sustainable improvements in these competitive factors concerning software production and use. Consequently ESI focuses on supporting its members in improving their software processes. Through a projected large membership and through the interaction of its members with the wider industry this will facilitate an overall improvement in the state of software practice in the software industry in Europe.

However, ESI relies on co-operation with other, similar partners and initiatives to create synergy: ESI's vision is to be the primary node in a network of collaborating European partners in the software process field.

ESI — and its partners — work through the provision and dissemination of good software management practice, through the licencing of products and services supporting process improvement, and by brokering partnerships for software process improvement.

3. BENEFITS OF PROCESS IMPROVEMENT

There is already significant evidence to underpin ESI's belief in the benefit of improving software processes. Some interesting quantitative facts concerning the return of such investments have been brought together in publications. A report published by the Software Engineering Institute (SEI) [Herbsleb 94] lists factors of return on investment of between close to 4 and close to 9, not even taking into account all of the benefits generated (especially the time to market benefit was found difficult to quantify.)

A significant amount of this has to do with the increased quality of intermediate deliverables and correspondingly earlier defect removal as an organisation matures in its software process capabilities. For the purposes of this paper the fivel level model of the SEI Capability Maturity Model (CMM) is used as a reference model for indicating organisational process capability improvement Figure 1 below relates the frequency of defect discovery and the relative cost of error to the capability level of the software producing organisation.

level	Rqts.	Design	Impl.	Unit test	System test	Post Delvry	Rel. Defect Cost
mang. 4	5%	10%	50%	20%	10%	<5%	1
def'd. 3	3%	5%	7%	25%	50%	10%	2.6
repeat. 2	0%	0%	5%	15%	50%	30%	3.5

Figure 1: Frequency of discovery and relative cost of defects

From a business perspective it is important to related these potential improvements (savings) to the overall cost introduced by an improvement programme. A widely publicised example is that of Raytheon. Figure 2 shows the summary of the Raytheon experience as presented in Scientific American in September 1994.

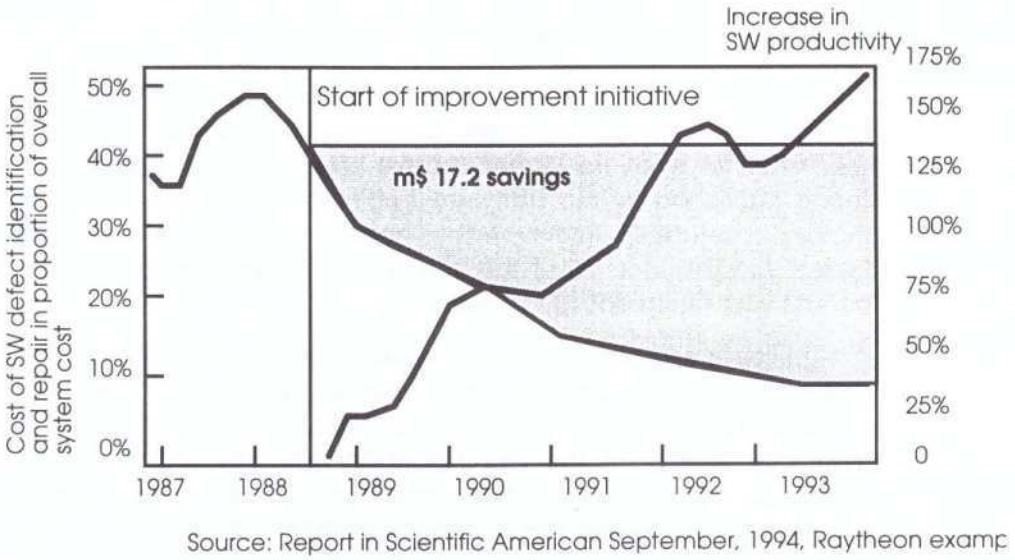


Figure 2: Raytheon's process improvement savings

The relationship between the quality of the process and an overall productivity index is shown in figure 3.

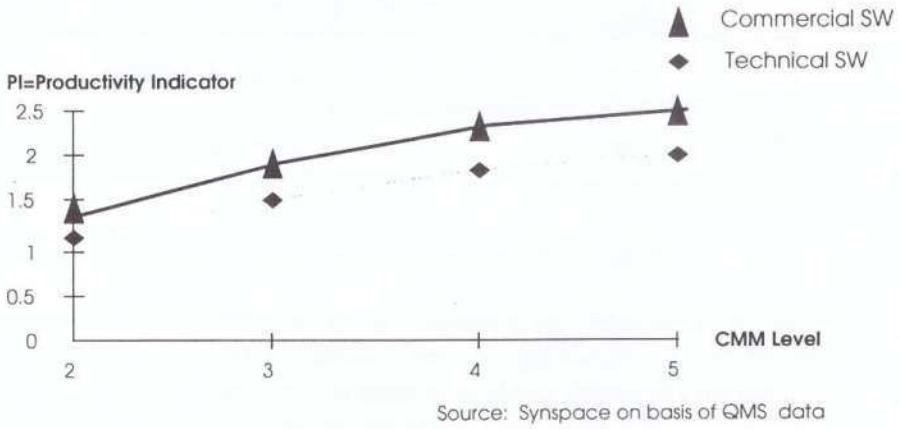


Figure 3: Process capability and productivity

The SEI report on benefits of software improvement benefits also reports shorter time to market in some cases through better control of the process. This is of particular interest since time to market is a crucial ingredient for the profit margin of a software intensive business (cf. figure 4.)

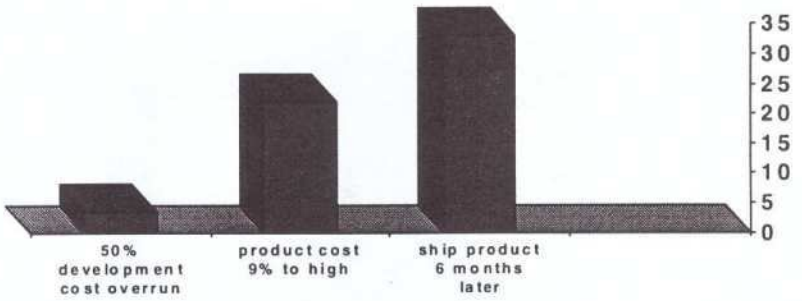


Figure 4: Lateness in delivery is the most influential factor in profit reduction

4. STATE OF SOFTWARE PRACTICE IN EUROPE

With this background information let us take a look at the state of software practice in European industry, specifically to analyse how far the knowledge about the benefits of software process improvement is know and, even more importantly, to what degree such practices are commonplace.

In trying to examine this let us simplify the analysis and, for the time being, reduce the complex software process improvement issue to the notions of software quality and productivity. What can one say about the European software industry view of the interrelationship of these notions and the impact on the organisations competitiveness?

In late 1994 and early 1995 ESI conducted a survey of software practices by addressing questionnaires to IT managers across Europe [Tully 95]. The survey was conducted to be representative for Europe and was correspondingly weighted by the relative importance of IT in the various countries and the various industry segments.

4.1 Industry priorities

One of purposes of the survey was to find out what organisations perceived as their main strategic objective for IT and to contrast this with the understanding possessed by the IT part of the organisation.

The result concerning the organisation is not surprising: **91%** name **customer satisfaction** as **the primary objective** for the organisation.

The view of the IT managers is more diverse (cf. figure 5), here customer satisfaction ranks at par with time to market, and not much higher than productivity and quality. Obviously these factors are interrelated and further elements of the survey examined what relationship these have in the mind of the IT managers.

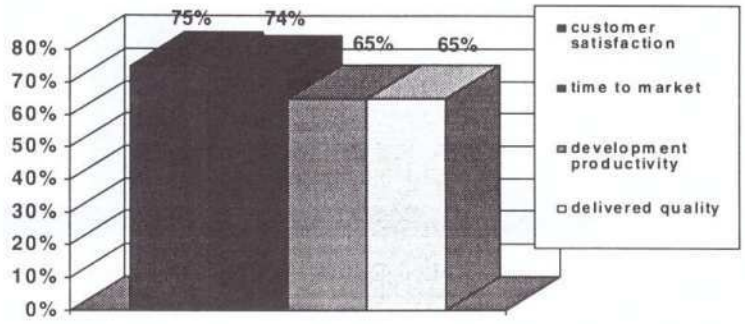


Figure 5: Priorities as seen by IT manager

With customer satisfaction being rated so highly it is, however, surprising that only 37% use a defined and formalised process for determining customer satisfaction. The remaining **63%** rely solely in **informal feedback**!

4.2 Quality and productivity

To analyse the perceived relationship between quality and productivity the survey examined what effect certain improvements in the software process would, in the opinion of the IT managers, have on quality and productivity, respectively. Figure 6 shows the response for a number of quality improvement actions, and their perceived effect on quality and on productivity. Obviously **IT managers do not believe that quality improvements significantly contribute to increasing productivity**.

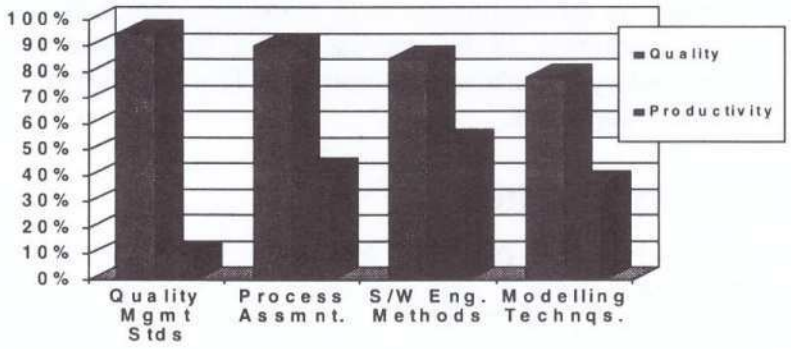


Figure 6: Improving quality and the impact on productivity

When concerning measures that might be taken specifically to improve the productivity during software development we get a somewhat different picture. These, generally more technique and tool oriented investments, are also perceived to have a greater impact on quality that the other way around. It is worth noting that there still is a lot of belief in achieving productivity and quality improvements through case tools — more so than for other approaches (cf. figure 7.)

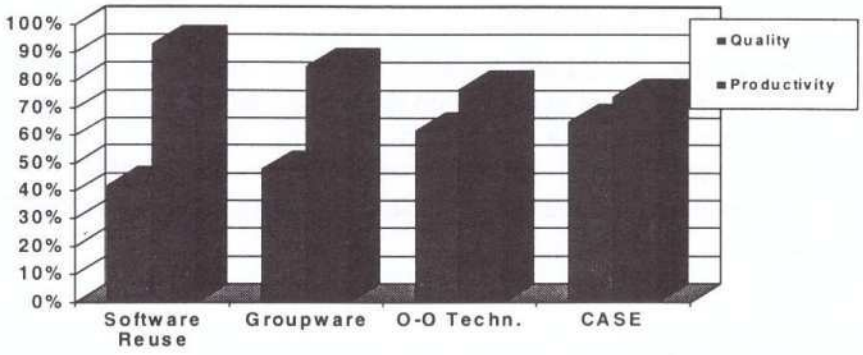


Figure 7: Productivity improvement and the impact on quality

The situation gets somewhat more interesting when examining what IT managers perceive to be the main barriers to achieve improvement in productivity. The responses shown in figure 8 indicate again a very strong belief in tools as the solution (and therefore their lack is a barrier), but more importantly, **75% believe quality improvement programmes to be "counter-productive".**

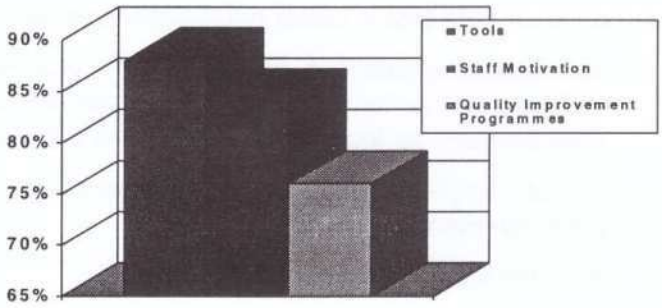


Figure 8: Barriers to improving productivity

When we examine the situation concerning quality process improvement programmes deployed in organisations the answer is quite revealing (cf. figure 9): 47% have no such programme at all, and only 33% follow an independently recognised formal approach.

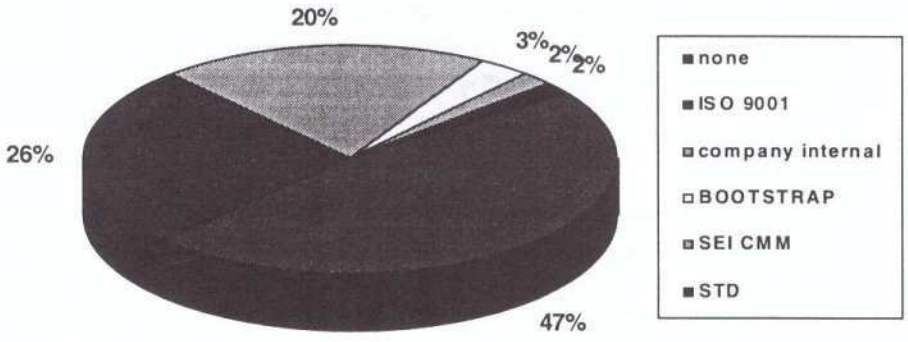


Figure 8: Improvement programmes employed

4.3 Survey conclusion

The survey produced a result which is not quite unexpected in its quality. However the level of confusion is far more significant than expected. In a nutshell one can conclude that the majority of IT managers perceive quality improvements to hinder performance. While the relevance of some organisational elements (e.g. staff) is recognised, tools are primarily seen as the answer.

5. ESI'S GOALS

The results of this survey have had some impact on ESI's direction of work. This section deals with ESI's aims, and the derived actions will be described in the following section.

One of the key goals of ESI is

1. *to analyse the software segment in European industry.*

More detail, both in breadth and depth needs to be known about the European software industry — both the supplier and the user segment.

Based on such data ESI is better positioned to achieve its second key goal

2. *to create awareness in European industry about software as a competitive factor, and in the software community about process improvement as the road to better quality and productivity.*

The third and complementary goal is to support those now interested in software process improvement by aiming

3. *to identify improvement opportunities*

The benefit to the supplier industries are market increase and improved customer maturity. Users will gain guidance to a more mature software supply industry — both development and procurement of software are targets for improvement.

6. ESI'S PROGRAMME OF WORK

This section gives an overview of some of the technical activities ESI performs to achieve its goals. Other aspects of ESI's work and the services offered to its members are covered elsewhere in this conference.

6.1 EXPRESS

The ESI EXPRESS (excellence in software practice in Europe: survey and self-diagnostic) project aims at software producing units (SPUs). The aim of the survey, supported by a questionnaire built on a model derived from SPICE and the European Quality Award, is not only to get a picture of the current state of software practice in large segments of European industry, but also to build awareness of the benefits of software process assessment and improvement.

The questionnaire establishes a profile of the SPU, and then addresses enabling factors of a software excellence strategy — such as leadership, SPU policy, people management, resources, and the software processes themselves. A section on the results of the SPU's software excellence strategy examines issues such as end-user satisfaction, SPU staff satisfaction, development cost, and financial performance.

The results of this survey will be fed back to the participating companies. Repetition of the survey at regular intervals will lead to an analysis of trends in the industry over a period of time.

The survey will later be supported by a low cost software process self-diagnostic tool and service, which, however, concentrates more on the software processes of the organisation. The tool, Bootcheck, is build in the SPICE framework and allows SPUs to self-assess their software capabilities and use these as a basis for improvement. A second aim is to produce another set of aggregated data for ESI to enable comparisons to be made.

These two actions provide a wide, but less deep, understanding of the European software industry through the survey, and a more detailed understanding of the state of practice in software processes through Bootcheck. This will be complemented by even more detailed analysis gained through SPICE assessments (see below.)

6.2 SPICE

More detail on SPICE (software process improvement and capability determination) is given elsewhere in this conference. The role of ESI in SPICE is that of the international and European trials co-ordinator. Now that the SPICE defining documents have been accepted for the standardisation process the trials are an essential instrument to gain experience with the standard and to lay the foundation for future improvement of the standard.

Besides participating in the trials ESI is planning enhancements of SPICE to cover processes concerning data centres and those of organisational learning.

6.3 Practice and experience repository

The practice and experience repository is a (logical) information base which augments the general state and trends of industry information mentioned above when talking about express. Essentially this "repository" has three major components:

- information about the software industry as a whole — which we plan to augment through benchmarking information,
- a comprehensive collection of case studies, and
- a "technology shelf".

Case Studies

This activity aims at building a comprehensive repository of — user — experiences with software process improvement approaches, specifically to answer the following questions:

- what were the reasons and expectations when starting the programme,
- what was the approach chosen,
- what were the results achieved,
- what was the effort needed,
- what was the return achieved, and
- what are the key lessons learnt?

These success and failure "stories" describe the approaches and the context in which they were performed. There already is a lot of material out there that can be used to provide meaningful guidance and help for other organisations considering similar approaches.

Sources of such information are events such as this, and special dissemination efforts such as project VASIE, which aims at creating an electronic library of ESSI application experiment experience report at ESI and several mirror sites across Europe.

Essentially everyone is invited to share their experience with others. By making such reports available to the editing team at ESI you can ensure that others can learn from your experience, too.

Cross-relating information and adding additional guidance will lead to a European experience repository accessible on-line. This will stimulate the collection of measurement and benchmarking information on a wider scale.

Technology Shelf

The idea of a "technology shelf" for software process improvement is to gather information from service and product suppliers about their products and services. This information will be categorised with respect to the aspect of software process improvement it addresses, the type of service or product it constitutes, where it is available , and in what industry segments it has been used to what effect.

This information will be further structured to provide an on-line introductory guide to what is available on the market in software process improvement. In a further phase, once a sufficient volume of information has been collected, relations between the technology shelf and the case studies will be established.

You can contribute !

Everyone is invited to share information about their experience to help us publicise the evidence that process improvement is beneficial. Success stories, measurement and statistical data, and also service and product offerings are

welcome. For this ESI can be contacted electronically at **info@esi.es** or **www.esi.es**.

6.4 Strategic alliances

ESI is a membership organisation, i.e. it forms a network. Additionally it acts with local and regional associate partners across Europe to reach its members. At an international level ESI forma part of a network, too. ESI recently finalised a collaborative research and development agreement with the Software Engineering Institute at Carnegie-Mellon University, USA, which includes cross-licencing of each others technologies.

Other international partners of ESI include the Software Technology Centre, Canada, the Software Quality Institute, Australia, INSEAD, France, and the ISERN network.

7. CONCLUSION

Software process improvement is the key to improved competitiveness through software in Europe. European industry as a whole needs to adopt the technologies available and share experience gained to produce a more mature state of industrial software practice. ESI and its members and partners act as facilitators in this process.

8. REFERENCES

[Herbsleb 94] Herbsleb, J. et al., *Benefits of CMM-Based Software Process Improvement: Initial Results*, CMU/SEI-94-TR-13, Software Engineering Institute, August 1994

[Tully 95] Tully, C. (Ed.), *The ESI White Book on Software Practices in Europe*, ESI, August 1995

[EC 94] European Commission, *Growth, competitiveness, employment — The challenges and the ways forward into the 21st century*, White Paper, Brussels, 1994

ESSI - THE EUROPEAN SYSTEMS AND SOFTWARE INITIATIVE: OBJECTIVES, STRATEGY AND IMPLEMENTATION OF SOFTWARE BEST PRACTICE ACTIONS

Mechthild ROHEN

European Commission
Directorate III - Industry

ABSTRACT

Software Best Practice is a group of actions being supported by the European Commission[1] to promote improvements in the software development process. A pilot phase of Software Best Practice was initiated under the auspices of the *European Systems and Software Initiative (ESSI)*[2]. The results so far have been very encouraging and further actions[3] are being undertaken in the Specific Programme for Information Technology within the Fourth Framework Programme. The first call for proposals was issued in March 95, with a closing date in June 95.

The goal of Software Best Practice is to provide increased efficiency, higher quality and greater economy for European organisations involved in software development, through the take-up of well founded and established - but insufficiently deployed - methods and technologies. A focused strategy for achieving Best Practice, outlined in section 2, is adopted which aims to cover the different dimensions of Software Best Practice, as given in section 3. To achieve the goals of the initiative, five lines of complementary action, as given in section 4, are available, which provide support for software developing organisations at different stages of maturity: Stand-Alone Assessments, Process Improvement Experiments, User/Experience Networks, Training Actions and Dissemination Actions. A first analysis of the response to the current call has been undertaken and the results are reported in the section 5. It is worth noting, that all five lines of action are based on the experience gained in the pilot phase of ESSI. Feedback received so far from industry has been particularly valuable and a summary of the key observations is given in the concluding section 6.

---

1 Specific Programme for Research and Technological Development in the field of Information Technologies (IT) under the Fourth Framework Programme (1994-1998).

2 as accompanying measure to the European Commission's ESPRIT programme, in 1993.

3 as specific accompanying measures in the 1994 Work Programme of the Information Technologies Programme ([2]).

1. INTRODUCTION

Enterprises in all developed sectors of the economy - and not just the Information Technology (IT) sector - are becoming increasingly dependent on the relevance and quality of software based systems. Such systems support management, production, and service functions in diverse organisations. Furthermore, the products and services now offered by the non-IT sectors, e.g., the automotive industry or the consumer electronics industry, increasingly contain a component of sophisticated software. For example, televisions require in excess of half a Mbyte of software code to provide the wide variety of functions we have come to expect from a domestic appliance. Similarly, the planning and execution of a cutting pattern in the garment industry is accomplished under software control, as are many safety-critical functions in the control of, e.g., aeroplanes, elevators, trains, and electricity generating plants.

Today, approximately 70% of all software developed in Europe is developed in the non-IT sectors of the economy. This makes software a horizontal technological topic of considerable significance. More and more organisations from all industries across Europe are finding that software development is a key part of their process and that the efficient and effective production of high-quality, reliable, re-usable and maintainable software - resulting from a complex and expensive software development process - is increasingly important for them, if they are to maintain and enhance their competitiveness and profitability. This area of management concern is the main focus of the Software Best Practice actions.

2. OBJECTIVES, STRATEGY AND TARGET AUDIENCE OF THE INITIATIVE

Objectives

The goal of the Software Best Practice actions is to promote improvements in the software development process in industry, through the take-up of well-founded and established — but insufficiently deployed — methods and technologies, so as to achieve greater efficiency, higher quality and greater economy. The actions are aimed at the application of mature, proven and appropriate methods and technologies in the software development process.

The aim of the initiative is to ensure that European software developers in both user and vendor organisations continue to have the world class skills, the associated technology, and the improved practices necessary to build the increasingly complex and varied systems demanded by the market-place.

The full impact of the initiative for Europe will be achieved through a multiplier effect, with the dissemination of results across national borders and across industrial sectors to stimulate adoption of process improvement at a European level.

Strategy

To achieve the above objectives it is intended to implement actions that will:

- Raise awareness of the importance of the software development process to the competitiveness of all European industry.
- Demonstrate what can be done to improve software development practices through experimentation.
- Create communities of interest in Europe working to a common goal of improving software development practices.
- Raise the skill levels of software development professionals in Europe.

Target Audience

Any organisation in any industrial sector which regards generation of software to be part of its operation may benefit from the adoption of Software Best Practice. Such a user organisation is often not necessarily classified as being in the software industry, but may well be an engineering or commercial organisation in which software has emerged as a significant component of its operation. Indeed as the majority of software is produced by organisations in the non-IT sector and by small and medium sized enterprises (SMEs), it is these two groups who are likely to benefit the most from this initiative.

In addition to the user organisations participating directly in the initiative, software vendors and service providers also stand to benefit, as demand for their methodologies, tools and services is stimulated and valuable feedback is given on the strengths and weaknesses of their offerings.

3. DIMENSIONS OF SOFTWARE BEST PRACTICE

Software Best Practice activities focus on continuous improvement of the software development process. Software process improvement starts with addressing the organisational issues. Experiences in the past have shown that before any investments are made in true technology upgrades (through products like tools and infrastructural computer support) some critical process issues need to be addressed and solved. They concern how software is actually developed: the methodology and methods, and, especially, the organisation of the process of development and maintenance of software.

Organisational issues are more important than methods and improving methods is, in turn, more important than introducing the techniques and tools to support them.

Finding the right organisational framework, the right process model, the right methodology, the right supporting methods and techniques and the right mix of skills for a development team is a difficult matter and a long-term goal of any process improvement activity. Nevertheless, it is a fundamental requirement for the establishment of a well defined and controlled software development process.

Process improvement and implementation concerns people and needs to take into account all people related aspects (human factors). These are orthogonal to the normal technology and methodology driven approaches and are crucial to the success of adopting best practice. The people aspects cover all the different groups which have an input to the software development process including Senior Management, Middle Management and Software

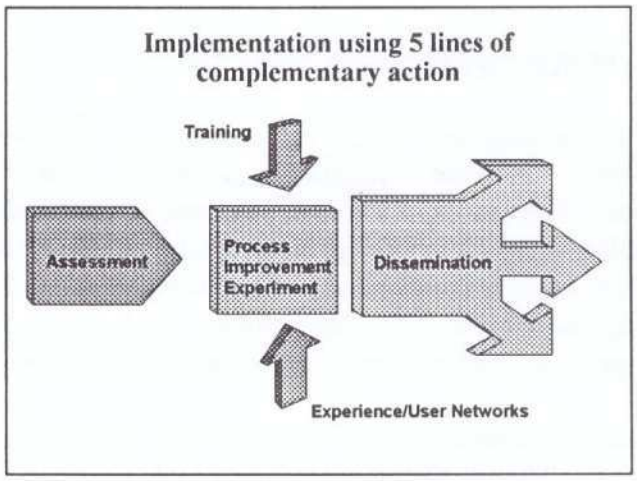
Engineers. Successful management of change includes staff motivation, skilling and promotion of the positive contributions that staff can make.

In order to ensure an appropriate environment for the successful adherence to a total quality approach it is imperative that Senior Management are fully aware of all the issues. Their commitment and involvement are crucial to the successful implementation of the improvement process and it might be necessary to raise their awareness regarding this issue.

It is important to identify clear milestones that will enable the software developer to measure progress along the road of software improvement. Certification through schemes such as ISO 9000, while not an end in itself, can play a valuable role in marking and recognising this progress.

4. LINES OF ACTION

The goals of Software Best Practice as outlined in the previous sections can be achieved through a number of focused activities. Support for organisations at different stages of maturity[4] will be provided through five specific and complementary lines of action, identified as specific accompanying measures in the 1994 Work Programme of the Information Technologies Programme ([2]), in Domain 1 - Software Technologies, Subdomain 'Software Best Practice', Tasks 1.27 to 1.31, inclusive:



1. Stand Alone Assessments (Task 1.27)
2. Process Improvement Experiments - PIEs (Task 1.28)
3. Dissemination Actions (Task 1.29)
4. Experience/User Networks (Task 1.30)
5. Training Actions (Task 1.31).

The following subsections give a brief overview of the five lines of action. More detailed information on the lines of action is provided in the ESSI Information Package ([1]). It sets

[4] 'maturity' is used in this document to indicate the extent to which an organisation is producing high quality software, regularly, efficiently and economically, according to recognised standards.

out the nature and focus of each activity and includes for each of the first four action lines detailed guidelines on how to prepare a proposal.

4.1 STAND-ALONE ASSESSMENTS

The main objective of the Stand Alone Assessments action line is to raise the awareness of user organisations to the possibilities for improvement in their software development process, as well as give the momentum for initiating the improvement process.

Assessments are targeted particularly at organisations at the lower levels of software development maturity (e.g. those which are starting to improve their development process). It is expected that assessments will stimulate the pursuit of quality through continuous improvement of the software development process.

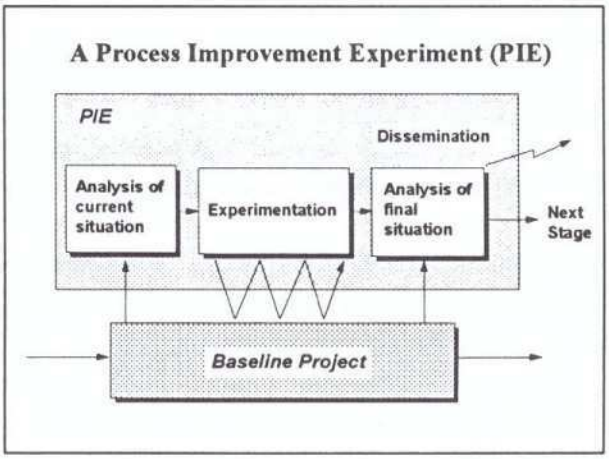
For carrying out the assessment, an underlying methodology is needed. This methodology should recognise that software development is governed by the *processes* which an organisation uses to capitalise upon the potential talent of its employees (*people*) in order to produce competitive, top quality, software systems and services (*products*).

No single standard methodology is advocated; however, the adopted approach should be a recognised assessment methodology, such as BOOTSTRAP, TickIT, etc.

4.2 PROCESS IMPROVEMENT EXPERIMENTS (PIES)

Process Improvement Experiments (PIEs) will form the bulk of the Software Best Practice actions. Their aim is to demonstrate the benefits of software process improvement through user experimentation. The results will be disseminated both internally within the user organisations to improve software production and externally to the wider community to stimulate adoption of process improvement at a European level.

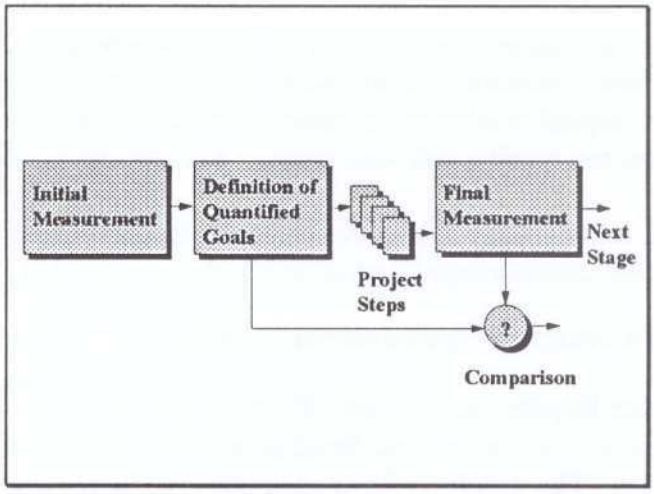
The emphasis is on continuous improvement through small, stepped actions. During a PIE, a user organisation undertakes a controlled, limited experiment in process improvement, based on an underlying *baseline* project. The baseline project is a typical software project undertaken by the user organisation as part of its normal business and the results of the experiment should therefore be replicable.



Support is available for the direct costs associated with carrying out the experiment, whereas the cost of the baseline project - including any software development - remains the responsibility of the user organisation.

Proposers are expected to have already considered their strengths and weaknesses, and have at least an idea of the general actions required. They will also need to demonstrate that they are aware of quality issues and are considering the people aspects of their actions.

With regard to the measurement of results, proposers are expected to define measurable goals for the experiment and undertake a final comparison of the situation before and after the changes to assess whether the goals have been achieved.



Adequate means to measure the results of the experiment (e.g. metrics) should be defined and introduced. It is suggested that an evaluation of performance and a cost/benefit analysis should be considered as necessary on-going activities.

Dissemination of the results of the experiment, from a software engineering and business point of view, to the wider community, will be an essential aspect of a PIE and will be undertaken with the support of the Software Best Practice Dissemination Actions (see section 4.3).

4.3 DISSEMINATION ACTIONS

Dissemination Actions are aimed at promoting the dissemination of information concerning Software Best Practice and the effectiveness of software process improvement, in general, and the results of the Process Improvement Experiments, in particular.

The objective will be to give users across Europe, with common sets of interests, both knowledge of what improvements can be achieved and contact information on where to find help in achieving the improvements for themselves.

Although the exact nature of the Dissemination Actions will vary, they are likely to correspond to one of two types:

1. Dissemination Actions which focus on valuable and generally-useful software engineering material which is representative of a class of processes, methodologies, assessment methods, tools, etc. Such material should not be exclusively represented by the 'products' of a single agency, company, or institute. Particular attention is paid to innovative mechanisms for dissemination and valuable but under exploited information.

2. Dissemination Actions with the specific objective of ensuring that organisations on a community-wide basis are aware of the experiences of the users which are involved in the Process Improvement Experiments.

4.4 EXPERIENCE/USER NETWORKS

The encouragement of a European software supply industry cannot be achieved simply by funding the RTD activities (Research and Technological Development) of the suppliers. There is also a need for the user industries in the forefront of development to co-ordinate their requirements and to send some clear messages to the suppliers.

There will be opportunity for networks of users, with a common interest, to pursue a specific problem affecting the development or use of software. Experience/User Networks will mobilise groups of users at a European level and provide them with the critical mass necessary to influence their suppliers and the future of the software industry through the formulation of clear requirements.

Support will be available to cover the necessary costs of organising and implementing the Experience/User Network.

As an example, a group of European banks involved with software development may wish to form an Experience/User Network to consider the issue of software pricing and licensing. Their objective might be to encourage software suppliers to provide them with the tools necessary to develop their software, in a way which is economical, flexible and befits the changing banking community. They would produce a final report for public dissemination, aimed at the software tool providers, specifying their requirements and conclusions for software pricing and licensing.

4.5 TRAINING ACTIONS

Training actions are broad in scope and cover training, education, and skilling for all groups of people who are involved in the software development process. Of particular importance is the need to address the universal problem of raising the profile of process improvement with senior management. Senior managers need to be shown measurable benefits of process improvement activities. The aim is to raise the awareness of management so as to ensure an appropriate environment for process improvement and the successful adherence to a quality approach.

It is worth noting, that a first training action was started in the ESSI pilot phase in late 1994, under the title of ESPITI (European Software Process Improvement Training Initiative). ESPITI aims to increase European software producers' awareness of the benefits

of ISO 9000 and software process improvement by providing training opportunities across Europe.

5. FIRST ANALYSIS OF CURRENT ESSI CALL

A call for proposals for the first four action lines, as given in the previous section, was issued on 15 March 95 (closing date 15 June 95). A preliminary analysis of the response to the call was undertaken, and the first key observations, with an emphasis on the Process Improvement Experiments, are provided in the following:

Some 464 proposals were received for those four action lines, which were subject of the call. 41 were for Stand Alone Assessments, 387 for Process Improvement Experiments (PIEs), 22 for Dissemination Actions, and 14 for User/Experience Networks.

Of importance is the widespread interest shown by organisations across all industrial sectors, from agriculture to retail trade. In total proposals were received from thirty five different industrial sectors. Overall, more than half of all proposals came from organisations outside the specialist IT software producing sector thus demonstrating in a compelling manner the importance that organisations in all fields of activity now attach to software and its efficient and effective development.

Of equal importance is the fact that, the response from small and medium sized enterprises (SMEs) was particularly strong and even above the high level in the pilot call. SMEs form the largest category of applicant. This again confirmed, as already in the pilot phase, that Software Best Practice is a key issue for SMEs in all sectors of the developed economy and that the programme is very well appreciated by and attractive for them.

With regard to the communities of interest (from the systems point of view) as indicated in the Process Improvement Experiment proposals, there was an equally strong response across the three communities of interest: business systems[5], technical systems[6] and embedded systems[7]. The technical and embedded systems interests together exceed by a small margin those concerned with business systems.

A very first rough analysis of the technical content and main focus of the PIE proposals resulted in the following top six categories: Object Oriented (OO) methods and tools, CASE tools, configuration management and version control, client/server architecture,

---

5 include e.g. all traditional MIS and EDP systems, found in banks, insurances, industry and public administration; for instance sales and marketing support systems, personnel management and administration systems, financial information systems, reservation systems, office automation systems, etc.

6 would e.g. include the development of systems aimed at engineering professionals, which may be used in a business or control environment; for instance computer aided design/manufacturing systems, computer aided software engineering tools, geographic information systems, simulation systems, document image processing systems, etc.

7 would e.g. include all software applications which are embedded within systems, which the user may or may not be aware of their existence; for instance in industrial automation and process control systems, signal processing systems, telecommunications systems, consumer products, etc.

software component re-use, project management methods and tools. Many proposals referred explicitly to more than one category and most of the OO referrals were also interested in re-use. Most of the proposals mentioned standard items such as various aspects of quality management and metrics.

The evaluation results indicate a significant increase in the overall quality of the proposals with respect to the results obtained in the pilot call. The current situation is indicative of a good level of comprehension regarding the ESSI programme and its focus on software process improvement, particularly true for the Process Improvement Experiments.

It is clear from the first analysis of the response that there is a widespread recognition of the need to improve software development practice and that software development capabilities are of increasing importance to the business strategies of a wide variety of enterprises distributed across the spectrum of all developed sectors of economy in Europe.

Further analysis of the response needs to be undertaken, particularly taking into account the results of the analysis of the questionnaires (i.e. the 'Software Best Practice Questionnaire' which accompanied the Information Package), which had to be filled by all proposers of Stand Alone Assessments and Process Improvement Experiments[8]. The analysis of the questionnaires will provide a snapshot of the situation and the results will be analysed at the European level to provide an overall picture for Europe on the maturity of industry with regard to software development. In addition, the results will provide a reference for future surveys so that the evolution of European industry can be monitored.

6. THE ESSI PILOT PHASE - FEEDBACK AND KEY OBSERVATIONS SO FAR

It is worth noting that all five lines of action (i.e. Tasks 1.27 - 1.31 in the Work Programme) are based on the experience gained in the ESSI pilot phase.

Detailed overall results of the pilot phase projects are not available at present, as only the first ones (out of about 100 projects) are in their completion phase.

However, feedback so far received from industry on ESSI has been particularly valuable and a summary of their key observations is given in the following:

- The successful uptake of software engineering products - techniques, tools, methodologies - often necessitates considerable changes in the organisational process as well as the skills of the software professionals.

- For this process-innovation to succeed, it is necessary to take a realistic view of the challenge: organisations should introduce process improvements in a step-wise manner.

- A balance between the different categories and disciplines of the technology is needed in addition to improvements at all levels of the organisation.

---

8 Proposers were informed that the information in the questionnaire would be treated in the strictest of confidence, the contents of the questionnaire will be stored and processed anonymously.

- Furthermore, it is necessary to base all actions on the use of mature, proven and appropriate techniques and processes.

- Best Practice activities need a focused orientation towards continuous and incremental improvement of the whole development process in software producing organisations, rather than just promoting the introduction of new technologies and tools.

- Metrication is essential. Senior managers will only be convinced of the benefits of adopting Best Practice if they can be shown evidence that it will produce an improvement in the key activities of the business to which they can directly relate, e.g. profitability, customer service, etc. Improvements in the process must be measurable, giving a basis for assessing the level of maturity/practices and for comparability.

- Software quality is an increasingly important topic. It cannot be added as an afterthought, but has to be built in from the very beginning of the software development process.

- Process improvement should be driven by business needs, not by technology.

- In the definition of what 'best practice' means in the context of software engineering and software process improvement the role of standards is seen to be vital in the long term. Organisations undertaking continuous software process improvement will have to base their activities on well established and accepted standards, however, at this stage of evolving standards, they will need to follow closely (and contribute to) the evolution of ongoing activities in this area (e.g. the ISO-SPICE initiative). The stronger the communities of interest are, which are building up, the more they may contribute to the creation and establishment of standards.

Perhaps one of the strongest features of the ESSI pilot phase is the manner in which the benefits obtained by individual organisations are multiplied across the Community. This is accomplished both by the dissemination actions and by the creation of communities of interest. It is intended that this aspect will be developed further in future ESSI calls.

Finally, it is worth mentioning that information relating to individual pilot phase projects is publicly available: Summaries of the pilot phase projects are provided in [3]. Individual results of ongoing projects are reported continuously in the ESSIGRAM ([4]), a regular ESSI newsletter edited by one of the pilot phase Dissemination Actions, which also includes the calendar of ongoing ESSI Dissemination Events (e.g. Workshops, Conferences, etc.). Within the near future the Final Reports which are interesting for the wider audience will be published on the World Wide Web (Server at ESI - European Software Institute, Bilbao).

GLOSSARY

CASE	Computer Aided Software Engineering
DBMS	Database Management System
EDP	Electronic Data Processing

ESI	European Software Institute
ESSI	European Systems and Software Initiative
ESPITI	European Software Process Improvement Training Initiative
ISO-SPICE	International Standards Organisation-Software Process Improvement and Capability dEtermination
IT	Information Technology
MIS	Management Information System
OO	Object Oriented
PIE	Process Improvement Experiment
RTD	Research and Technological Development
SME	small and medium sized enterprise

REFERENCES

[1] RTD in Information Technologies - Information Package
Part II E Software Best Practice (ESSI), Version: March 95

[2] European Commission, Directorate III - Industry
RTD in Information Technologies - Work Programme

[3] European Commission, Directorate III - Industry
IT R&D Programme: Software Systems and Software Best Practice
Summaries of ESSI projects (pilot phase)
Report EUR 15375 EN/8, November 1994

[4] ESSIGRAM: Newsletter of the European Systems and Software Initiative
Edited by ESSI Dissemination Action AENEID (Ovum Ltd., London, UK)

Acknowledgements

Much of this paper reflects information given in the ESSI Information Package ([1]). The opinions in this paper are the author's and in no way reflect official position of the European Commission.

ESPITI - A European-wide Training Initiative

Kevin Eakin
Project Manager - ESPITI Partner
MARI (Northern Ireland) Limited.

Abstract.

'Growth, Competitiveness, Employment - the Challenges and Ways Forward into the 21st. Century' is the name of a strategic CEC White Paper, which concentrates on the issues around keeping European companies competitive in the years to come.
Software will pay a pivotal role in the next century - the age of the Information Society.
While academics, managers and entrepreneurs may have a great vision of the potential of that age - will software developers -be able to deliver the goods, and meet the challenge of global competition?

ESPITI stands for 'European Software Process Training Initiative', and results from the White Paper.
It is an initiative, primarily using training, with a major goal - to improve the processes involved in producing software.

ESPITI is, in effect, the training element of the broader ESSI Programme - the 'European Systems and Software Initiative', which is, in turn, a component of the ESPRIT initiative.

It has been allocated 8.5 MECU. It runs until May 1996 and involves the fifteen Members of the CEC, Iceland and Norway.

Recognising the increasing role of software within organisations, the key message of ESPITI is the need to implement an approach of continuous improvement of the software development process, hence, awareness and key training is given in the areas of implementation of Software Process Assessment and Improvement , registration to ISO9001, and supporting subjects.

ESPITI spreads its messages through structured programmes of awareness, specialised training and public relations, and has set up experience networks (or, common interest groups).

An early initiative in the project was the conduct of an industry survey in all regions. This looked at subjects such as Quality Culture, Software process Assessment and Improvement, ISO 9001, and the Way Ahead. It attempted to find out attitudes and perceptions, and to measure levels of awareness, involvement and implementation.

It is hoped that ESPITI will bring lasting benefits. Organisations should have a clear view of the key issues and many will undertake programmes of improvement. Experience Networks will continue and provide useful forums. Regions will have a clearer view (from the survey) of their state of practice and governments may well decide to continue funding in areas where such investment will bring the best return. Organisations will be more profitable.

ESPITI - A European-wide Training Initiative

Kevin Eakin
Project Manager - ESPITI Partner
MARI (Northern Ireland) Limited.

Introduction

Information Technology is an enabling technology, already pervasive within organisations, and increasing in importance. It is the engine of most organisations -

- it is central to management and administration
- it provides information, both within and, increasingly, from external sources
- it is frequently the whole product or an embedded part of the product
- it provides the communication infrastructure.

The controlling mechanism of this engine is software.

As the Information Society rapidly expands, we will require more software and become more dependent upon its reliability. Already over 70% of all European software developed in non-IT sectors. In the future there will be a significant increase in the use of software by individuals in their homes, for leisure pursuits, etc. In this Society information will be more important than ever before, and we will require the technologies to move it around faster, and process it more accurately.

Orderly progress into the Information Society requires the development of software of high reliability and quality, produced very productively. In today's increasingly competitive marketplace this is a strategic subject for management to address, a key issue likely to bear directly on the profitably, perhaps survival, of the enterprise.

Global Transformation of Economic and Social Life

The Information Society is being constructed at a time of great change.

- Geographical boundaries are blurring.
- Business sector boundaries are blurring.
- Businesses are re-structuring.
- Changes are now occurring on a global scale.

Information technologies are playing a major role in this transformation process and the rate at which it can take place. The business world is changing fast, and this evolution will be the pattern for the future.

'Growth, Competitiveness, Employment - The Challenge and Ways Forward into the 21st. Century.'

The White Paper, 'Growth, Competitiveness, Employment' was produced by the CEC in June 1993. This paper addressed a whole range of issues aimed at focusing minds on what Europe needs to do to meet the challenge of global competition from Japan, the Americas and the Tiger Economies in the 21st century. The paper commented that "a new 'Information Society' is emerging, in which management, quality and speed of information are key factors to competitiveness..."

In this environment of rapid change and intense competition, it is not sufficient for Academics, Managers and Entrepreneurs to have a vision of the future, and expect their staff to cope with the changes. People need to be trained in new ways of working, at the right level, to react to changing times and technologies.

In particular, Europe needs to upgrade skills within organisations , in the area of continually improving the software development processes, in commercial organisations. The Community needs to 'improve the quality of training...increasing exchanges of experience and information on good practices and developing joint projects.'

One of the outcomes of the White Paper was the ESPITI project.

ESPITI is, in effect, the training element of the broader ESSI Programme - the 'European Systems and Software Initiative', which is, in turn, a component of the ESPRIT initiative.

What are the Goals of ESPITI?

The Goals set for ESPITI are to:

- Improve the software development process, through a programme of training
- Identify the true needs of Industry in each Member Country.
- Increase the level of awareness and benefits of ISO9001.
- Facilitate ISO 9001 Registration, through training.
- Promote the use of Software Process Assessment and Improvement techniques.
- Encourage information exchange between organisations and across country boundaries.

It was seen as important that small and medium sized enterprises were involved and that the non-IT sectors were encouraged to participate in the programme.

How is ESPITI organised?

The primary organisation is at Regional level. Within each country, the activities are organised by a 'Regional Organisation' - an existing commercial organisation who have been contracted to carry out a specific range of activities in support of the ESPITI goals. There are fourteen Regions spanning 17 countries.
However, there is built-in flexibility to allow for local variation in the interests of tuning messages and deploying resources where most needed. This customisation of the approach has been determined following the conduct of a survey of local business - one of the first activities of the project.

The fourteen Regions are divided into two sets - each set is supported by a 'Partner'. The two partners are MARI (Northern Ireland) Limited and Forschungszentrum Karlsruhe (Germany). They provide project management support and a range of resources:

- templates for various awareness and training events.
- templates for an information database.
- brochures, newsletters, briefings.
- a resources file of relevant materials.
- seminar sessions available on Powerpoint.
- Harmonisation of the Initiative throughout Europe by the Partners.
- An electronic Information Centre for on-line communication between Partners and regional Organisations.

What has ESPITI been doing?

Recognising the increasing role of software within organisations, the key message of ESPITI is the need to implement an approach of continuous improvement of the software development process. A key role of the initiative is to be a starting point for organisations who hear that message and ask 'what should I do next?'.

Regional Organisations have been busy implementing the ESPITI programme across Europe using five broad programmes of activity:

- **Industry Surveys**
- **Awareness Seminars**
- **Public Relations Activities**
- **Key Training Programme**
- **Experience Networks.**

Industry Surveys.

In order to match the ESPITI training event calendar to the true needs of Industry, a user survey questionnaire was designed by the Partners containing a core set of mandatory questions. Each Regional Organisation constructed their own questionnaire which included the mandatory core, and conducted a survey in their Region. The results have been used to fine-tune each Regional Organisation's training programme and the core elements of each survey are currently being merged to produce a picture of software development on a Pan-European basis.

Although the results of the Pan-European Analysis will not be available until the end of September 1995, the following has been noted about SME's at the Regional Organisation level of analysis:

- Awareness of SPI is low - e.g. less than 1 in 7 organisations have heard of SPI in the UK .

- The major problem areas in software development are perceived to be:
    - Requirements Specification
    - Lack Of Standards and Structured Approach
    - Project Management
    - Testing
    - Estimation

- Certification to ISO9000 is not perceived to significantly address the problems associated with the development of software.

- Human attributes such as experience , technical skills and management capability are perceived to have a stronger effect on performance than methods, processes and tools.

- Training budgets within SME's are currently low - e.g. 4 days per person per year in the UK.

- Greater Awareness of the following areas is required:
    - The Benefits Of Quality Systems
    - Software Process Improvement
    - Software Process Assessment

- The biggest demands for training courses currently lie in the following areas:
    - Requirements Specification
    - Structured Analysis And Design
    - Project Management
    - Structured Testing
    - Estimation

Awareness Seminars.

Another activity is the organisation of a large number of Awareness Seminars.
These seminars have a number of objectives:

- to explain the complete ESPITI programme
- to review the current state of software development and its shortfalls
- to present the challenges of anticipated future competition, and the need for action
- to present the need for the practice of 'continuous improvement' to the software development processes
- to explain ISO 9001 Registration and Software Process Improvement as frameworks for implementing programmes of change
- to provide a preliminary forum for discussion and advice

A key ESPITI message is the need to face the challenge of the competition by the implementation of the policy and practice of continuous improvement to the software processes. In response to this, delegates are given a briefing on a generic model for Software Process Improvement, with an explanation of how the method operates.
For many delegates this would have been their first real exposure to these concepts.
Sessions in Awareness Events include the reasons for seeking ISO9001 Registration, how an implementation project would proceed, and what specific steps to take to ensure that improvements to quality and productivity were actually obtained, including the incorporation of 'continuous improvement' practices.

To supplement the awareness programme, brochures and newsletters were distributed, with information about ESPITI , SPI, ISO9001, etc.

Many more people will now know something of significance about ISO9001 and SPI, and will have heard about Bootstrap, CMM, SPICE, etc. More importantly, they will have been advised to address the necessity for improvement in their organisations with some urgency and determination.

Public Relations

Regional Organisations carried out a structured programme of publicity, taking key 'ESPITI messages' to selected areas. The messages were concerned with

- providing information about the ESPITI programme,
- raising awareness of the growing importance of software to the economy, and,
- creating interest in implementing software process improvement programmes.

Some key personalities have been addressed. Political Figures (including senior civil servants) were lobbied. Shortly, they will be asked to secure the momentum of improvement, by adopting their own follow-up Regional Programmes. Leaders in Industry have been briefed, particularly on the challenges ahead, and the means of meeting them.

The Briefings were delivered personally or to small meetings. In some cases, suitable conferences were visited, small stands erected, brochures distributed, and delegates lobbied and informed.

The Awareness programme has been extended to a very wide audience through a regular programme of press releases to the more 'serious' newspapers and appropriate journals in each Region.

Key Training Programme

Following the planting of the seeds in the 'Awareness' phase, the ESPITI programme has sought to be a catalyst for promoting suitable specialised training, in the area of improvement to software development processes.

Key areas of training have addressed:

- Establishing a foundation for SPI through the implementation of a quality system.
- Identifying the current state of your process through software process assessment.
- Providing the means for process improvement through SPI tools and techniques.

All training events sponsored by ESPITI have been subsidised to encourage participation.

Experience Networks.

To support the various implementation activities which it is hoped that organisations will initiate, 'Experience Networks' have been set up, some Regionally based and some reaching across national boundaries. In some cases, existing common-interest groups have been supported and sponsored, when they addressed areas of interest.

They are meant to provide a valuable forum for discussion, dissemination of knowledge, exchange of experiences, supplier evaluation, etc.

It is hoped that these networks will continue to develop on a permanent basis, and include smaller and less experienced organisations.

Events Staged:

Between 500 and 600 awareness and training events will be carried out in the ESPITI project. In the first 8 months of the project , the following numbers of events have taken place :

Awareness Events	:	99
Training Events	:	91
Working Group Meetings	:	50

Awareness Events have attracted audiences of between 30 and 150 people.
Training events have attracted audiences of between 4 and 15 people.

Continuous Improvement - the Key to Success.

The essential messages from the ESPITI programme are twofold:

- Each organisation , to face the anticipated environment of ongoing competition, needs to adopt practices of 'continuous improvement' to its software development processes
- Staff need to be trained in the relevant skills to implement effective improvement programmes to address increases in quality and productivity.

It is hoped that ESPITI will bring lasting benefits. Organisations should have a clear view of the key issues and many will undertake programmes of improvement. Experience Networks will continue and provide useful forums. Regions will have a clearer view (from the survey) of their state of practice and governments may well decide to continue funding in areas where such investment will bring the best return. Organisations will be more profitable.

The ESPITI initiative is a key strategic programme for future prosperity of both organisations and the Community.

References:

Commission Of The European Communities. 1993. "Growth Competitiveness, Employment The Challenges And Ways Forward Into The 21st Century - White Paper". ISBN 9282670007

BOOTSTRAP INSTITUTE: EUROPEAN OVERVIEW OF QUALITY IMPROVEMENT

Adriana Bicego
Etnoteam S.p.A., Milan, ITALY,
and
Pasi Kuvaja
University of Oulu, Oulu, FINLAND

Abstract *- The paper presents the experience performed in Europe to first develop and then use the BOOTSTRAP software process assessment methodology. The origin of the methodology are summarised and the BOOTSTRAP Institute, now managing the evolution of the methodology, is presented, with a synthetic description of its organisation and activities. An overview of the methodology is then included, describing the BOOTSTRAP main characteristics and choices . Figures on the assessments performed in Europe are presented and commented. Comments on benefits reported from assessment-based software process improvement conclude the paper.*

1. Introduction

The BOOTSTRAP methodology was originally developed by a European consortium partially funded by the European Commission within the ESPRIT program. The goal of the project, that was considered a pre-ESSI (European Systems and Software Initiative) pilot - was to fertilise the ground for good software engineering practices in Europe and to analyse the awareness of the European software industry in this area. The American experiences inspired the project to develop a kernel of a process assessment methodology based on the maturity model developed by the Software Engineering Institute (SEI) at Carnegie Mellon University [1] and [8]. As in Europe the ISO (International Standards Organisation) 9000 standards ([2], [3] and [4]) were highly considered by the software industry, these standards were also taken as a starting point of the methodology. Also features of the life cycle model of the European Space Agency (ESA) ([5]) were applied. The newly developed approach was piloted at Bosch GmbH laboratories in Germany and it soon showed to be very helpful in establishing the basis for software process improvement. Furthermore, the goal to provide a preliminary evaluation of the European awareness in software engineering established the starting point for a data base of (anonymous) assessment data.

After the completion of the ESPRIT project, some of the participating partners, who were already using the BOOTSTRAP approach within their commercial services, decided to exploit the results of the ESPRIT project by establishing an international organisation to professionally carry on the methodology. This organisation is the BOOTSTRAP Institute.

2. The BOOTSTRAP Institute

The BOOTSTRAP Institute was founded in March 1994 as a European Economic Interest Group (EEIG), that is a non-profit organisation legally established under the European law.

The Objectives of the Institute are as follows: (from the BOOTSTRAP Institute's statute):

"1. The joint continuous improvement of the BOOTSTRAP methodology for the assessment and improvement of quality of software processes, also taking into account

the relevant ISO standard and other international initiatives in the field; this includes the proper packaging of the methodology and related training material;

2. The promotion of the BOOTSTRAP methodology;
3. The licensing of the BOOTSTRAP methodology to third Parties;
4. The proper handling of the database of results of assessment carried on by BOOTSTRAP licensees
5. The accreditation of BOOTSTRAP Assessors;
6. The certification of assessed organisations."

The above objectives are carried out by the members of the Institute. In addition to the Founding Members (mainly some of the organisations previously participating in the Esprit project or inheriting the position of a previous Esprit Partner) there are two levels of membership:

- the Full members, including mainly profit organisations, deeply involved in the life of the Institute, both from the point of view of the managing responsibilities as well as for carrying on the operational activities;
- the Corporate members, including organisations interested in following the technical evolution of the BOOTSTRAP methodology (these are typically Universities).

Additionally those organisations willing to use the BOOTSTRAP methodology without a deeper involvement in the BOOTSTRAP Institute can become licensees. BOOTSTRAP licences allow to perform assessments internally to the licensee organisation as well as to establish a market service based on BOOTSTRAP assessment.

3. Overview of the methodology

The BOOTSTRAP methodology was developed to facilitate software process assessment based improvement [6] with the following characteristics:

- emphasis on main existing de facto standards, like CMM, but focusing on standards widely accepted in Europe as ISO 9000 series and ESA;
- providing guidelines to goal-oriented process improvement;
- providing a methodology suitable also for Small-to-Medium Enterprises (SMEs), that are the majority in the European software industry;
- providing database support for improvement and for monitoring the evolution of the European software industry maturity in software development and management.

3.1 Features from CMM, ISO 9001 and ESA

The CMM provides a model of good software engineering practices that an organisation can instantiate through a sequence of evolutionary stages called maturity levels: by applying the model an organisation implements the continuous improvement approach recommended within Total Quality Management. In Europe the ISO 9001 certification has been widely applied also in software organisations, to demonstrate their ability to produce software according to industrial quality standards and as a means to select suppliers in large bids. The BOOTSTRAP methodology provides a software process assessment approach which integrates CMM and ISO 9000 ([2], [3]). A classical software life cycle model that was standardised by the European Space agency was also adopted as a reference model for the development life cycle [5]. Based on these choices, the BOOTSTRAP methodology ([7], [9] and [14]) provides:

- support to software process improvement based on good software engineering practices recommended by the widely accepted background standards;

- a methodology to guide establishing an effective quality system, also in preparation to ISO 9001 certification by supporting definition of improvement priorities, which can be used to plan the quality system implementation;
- a long-term improvement perspective beyond the certification; an organisation which implements ISO 9001 requirements equals approximately BOOTSTRAP maturity level 3, from which the organisation can proceed with a continuous improvement approach, which the BOOTSTRAP methodology guides to implement.

3.2 Scoring, rating and assessment result

Realistic process improvement targets can be set up only based on the current situation: an organisation should not plan to achieve level 5 if it is at level 1. Additionally, process improvement always requires investments: to make it effective and grasp the benefits of performed investments, improvement should be based on a true and fair representation of the current situation. To achieve this objective the following requirements should be fulfilled:

- to assure that assessment results are reliable and repeatable;
- to provide both synthetic as well as analytical assessment results to support management decisional process as well as detailed analysis for improvement planning.

To fulfil the first requirement, the BOOTSTRAP methodology adopts a scoring and rating mechanism, which allows to minimise differences between evaluations performed by different assessors. Scoring is based on four values (represented by adjectives as absent, weak, fair, extensive) for each question included in the questionnaire that guides the execution of the assessment. This four value scale was adopted because it allows to better reflect the reality than a two value scale based only on "yes" and "no" answers. Additionally the questions that cannot be applied to the assessed organisation may be left out from the evaluation as non applicable questions. Finally to reduce differences in evaluation due to individual experiences, all the assessors are trained in official BOOTSTRAP Institute classes and are accredited according to the BOOTSTRAP Institute assessor qualification procedure.

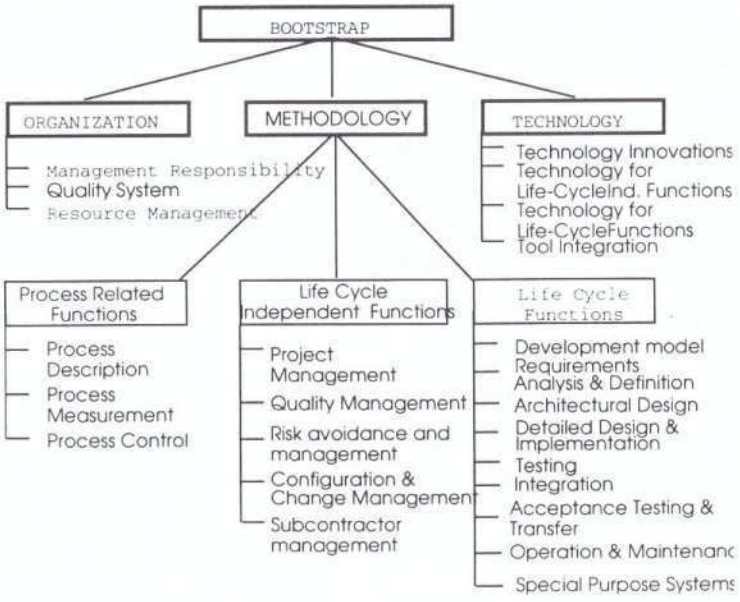


Fig. 1 The BOOTSTRAP process model

To fulfil the second requirement the BOOTSTRAP methodology produces capability profiles that express the strengths and weaknesses of the assessed organisation at various levels of details, according to the tree structure of the underpinned process model (see Fig. 1). The results may be presented for each level of the tree structure (see Fig. 2) using the five-level maturity scale including quartile precision inside each maturity level. This makes it possible to better understand the current status of each assessed organisation and the differences between organisations at the same maturity level.

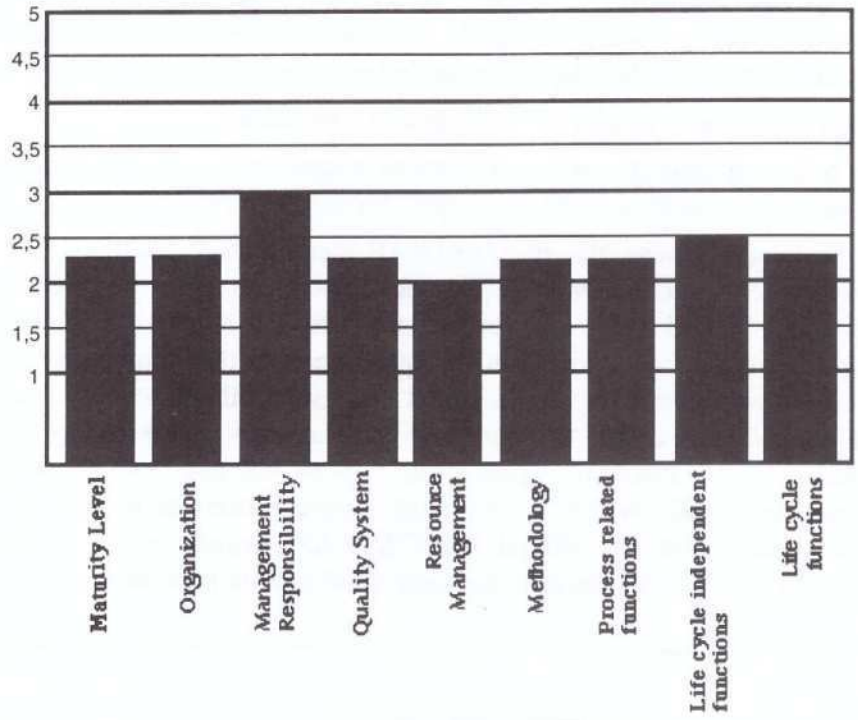


Fig. 2 Example of a BOOTSTRAP profile

Additionally Bootstrap provides separate capability profiles for the so-called Software Producing Unit (SPU)[1] and its project, in order to provide a more complete picture of the assessed organisation. Separate profiles are produced according to the Bootstrap assessment schema, which requires the assessment to be performed at two levels:

- the SPU's level, where the overall organisation is evaluated, including the quality system, its responsibilities and procedures;
- the project level, where the practices adopted by each individual project are evaluated.

SPU's and projects' results are not merged to provide a synthetic evaluation of the assessed organisation: on the opposite separate profiles for the SPU and each assessed project are provided. The reason behind this choice is that the SPU's and projects' profiles are often different from each other: sometimes the SPU's profile shows processes at a higher maturity

[1] An SPU is an organisational unit which develops and manage software applying a common set of rules: it can be a department within a company (for instance the department which develops one family of software product) or even an entire company itself (for instance in the case of a small software house)

level, sometimes on the opposite some of the projects show to apply processes more mature than the correspondent procedures at the SPU level.

Comparing the SPU's and the projects' profile allow to understand such issues as :

- to what extent the quality system, or at least the existing procedures, are applied;
- whether there are mature practices at project level that can be adopted at the level of the overall SPU;
- where strengths and weaknesses are.

3.3 Improvement approach

Effective process improvement should meet the specific organisation's needs. Therefore, the BOOTSTRAP methodology uses its own process model as a support to identify improvement goals and define improvement actions, but it also analyses the specific organisation's characteristics and needs to establish improvement priorities. Fig. 3 synthesises the elements of the BOOTSTRAP approach to software process improvement.

Fig. 3 BOOTSTRAP approach to software process improvement

One of the objectives during a BOOTSTRAP assessment is to collect information on topics as the current business goals, the organisation's planned strategy, the culture and attitudes and possible previous experiences in process improvement. Such information are used to:

- identify improvement priorities, in order to assure that the identified improvement targets can help to meet the organisation's needs and to achieve the organisation's business goals;
- identify possible risks if the improvement will not be undertaken; this provides a rationale for the suggested improvement actions; and
- evaluate possible risks of failure for the suggested improvement actions: this information allows to identify mitigation strategies to support successfully carrying out improvement.

Finally the database provides an input to improvement planning, by showing the maturity of European software industries: as process improvement should provide a competitive advantage, a comparison between the assessed organisation and other organisations in the same industrial sector provides a major input for establishing improvement targets.

4. The BOOTSTRAP database

The assessment data from all BOOTSTRAP assessments are automatically collected into the central database. The BOOTSTRAP database has the following important role in the BOOTSTRAP assessment:

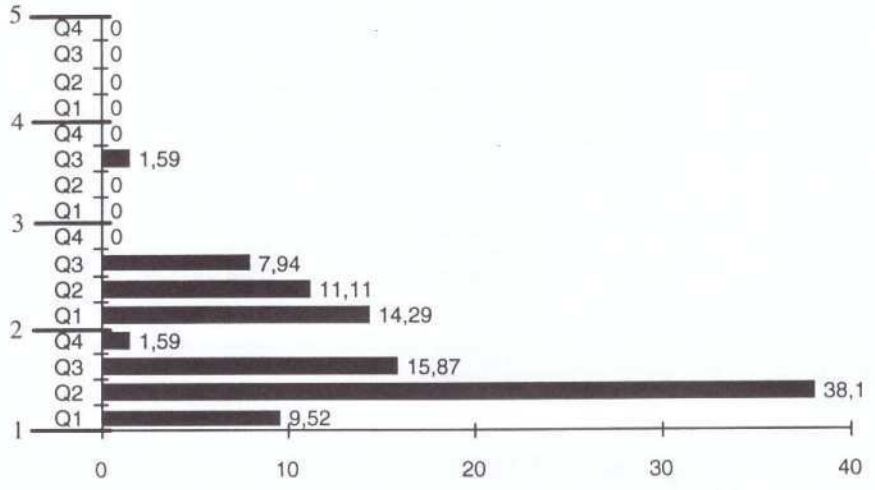
- to collect data on assessment results performed within Europe, in order to provide a picture of the maturity of the European software industries; and
- to position each assessed organisation towards other organisations in the same field.

This information gives better understanding on how the current capability of the assessed organisation should be taken into account in improvement planning.

Assessment confidentiality is guaranteed because all assessment data in the BOOTSTRAP database are anonymous.

Data in the database are grouped into two classes, the SPU's and the projects' data, according to the Bootstrap assessment schema.

Currently (July 1995) the BOOTSTRAP data base contains assessment data from 63 SPUs' and 176 projects representing various European countries (Austria, Belgium, Finland, Germany, Great Britain, Ireland, Italy, Spain and Switzerland). Graphs in figure 4 show the percentage distribution of the assessment results according to the BOOTSTRAP results presentation format which uses quartiles inside the maturity levels.



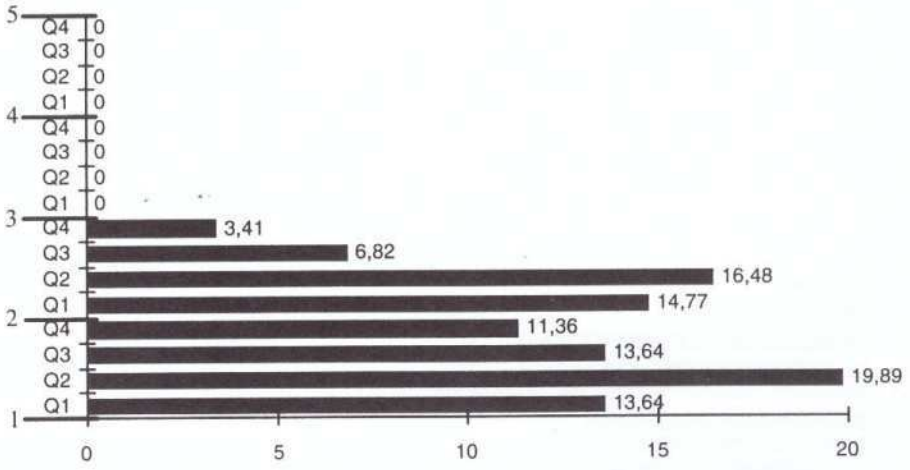
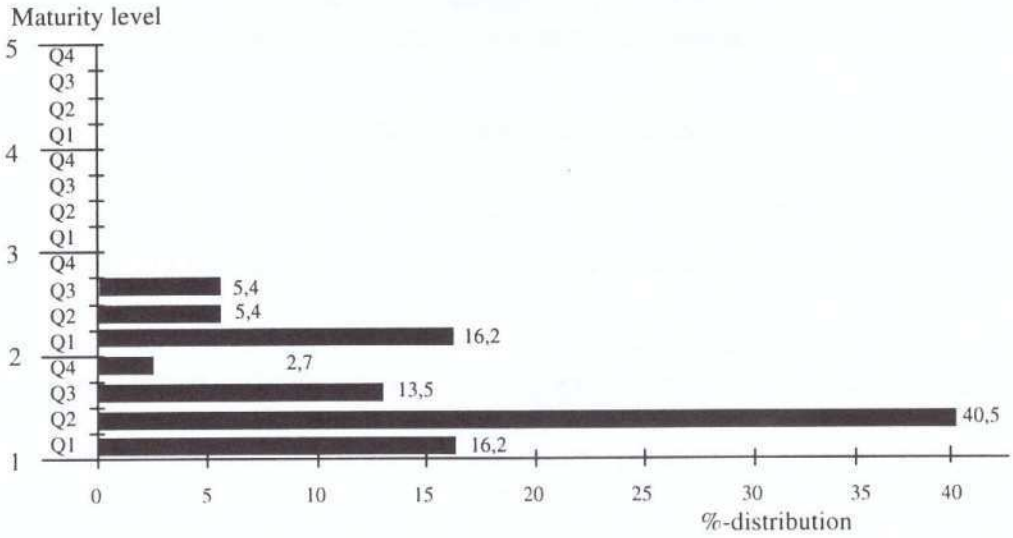


Fig. 4 BOOTSTRAP database at May 1995: SPUs' and projects' maturity distribution

In figure 5 the distribution of the assessment results a year ago (April 1994) is presented using the same format. At that time the BOOTSTRAP data base contained assessment data from 37 SPUs and 90 projects.



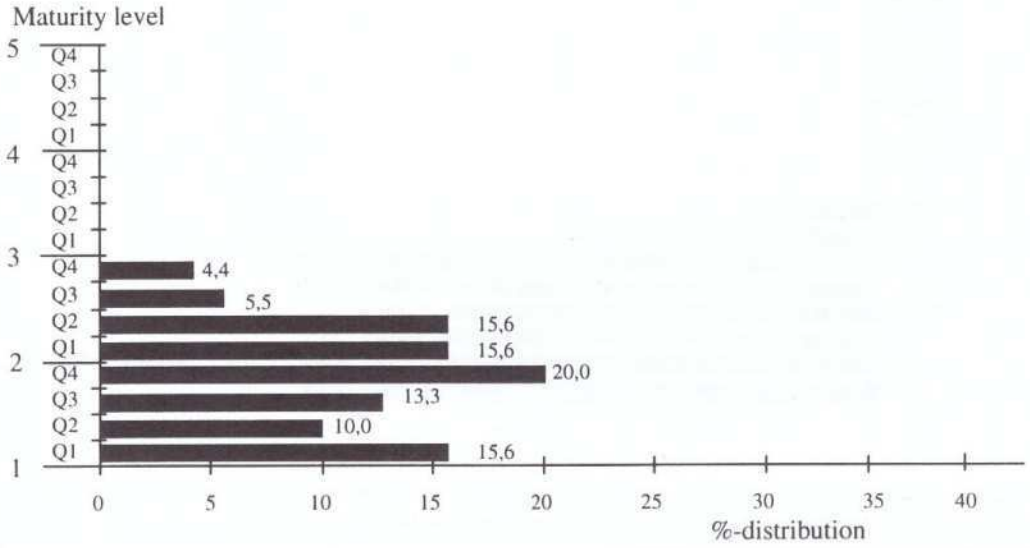


Fig. 5 BOOTSTRAP database at April 1994: SPUs' and projects' maturity distribution

Adding the percentage distribution per maturity level in the above graphs, we obtain the following figures:

- looking at the 1994 data, 73% of the assessed SPUs are at level 1 and 27% are at level 2; 58.9% of the assessed projects are at level 1 and 41.1% are at level 2;
- looking at the 1995 data, 33.34% of the assessed SPUs are at level 1 and 65.08 are at level 2; 58.52% of the assessed projects are at level 1 and 41.48% are at level 2;

Comparison between the two data sets points out the following:

- Most SPUs are currently still at level 1 but the percentage of SPU's at level 2 has increased from 1994 to 1995; additionally one SPU has been rated onto level 3;
- the percentage distribution of maturity at the project level is almost unchanged from 1994 to 1995 data.

On the other hand, comparison between SPUs' and projects' data at the same time shows that the percentage of projects at level 2 is higher than the correspondent percentage at SPU level. It is not easy to comment these data, also because the amount of assessments performed does not yet provide a sample statistically meaningful. It might be guessed that the focus on an organisation wide quality system currently being spread in the software industry has probably increased in the last few years the number of SPUs with established procedures.

Additionally is encouraging to know that quite many projects perform at level 2.

5. Reported benefits from assessment-based process improvement

Software process assessment has become a standard way to start process improvement. Yet, even if the approach is quite reasonable, very few real data are available confirming its actual benefits. One reason may be that the real effort is to carry out the improvement actions resulting from assessment and not the assessment itself. Improvement activities require a quite long time to be completed (at a minimum eighteen months) and only very few organisations have already undertaken a complete improvement cycle.

A technical report [10] has been recently published by the SEI presenting software process improvement results based on a survey of thirteen organisations. Anyway, it is still a small number of experiences.

The BOOTSTRAP results include successful experiences either in the context of a pure process improvement as well as in the context of establishment of a quality system to be certified according to ISO 9001.

A very interesting BOOTSTRAP experience has been reported ([11]), where the assessment results were used together with software metrics to monitor process improvement at Italtel, a large telecommunication company in Italy. An intensive process improvement programme carried on over 30 months allowed to move from assessed level 2.5, both at SPU and project level, to assessed level 3.5 for SPU and 3 for project. In addition to the capability quantitative evaluation resulting from the assessment, a set of metrics strongly related with the SPU's goals were defined and applied. These metrics are related to timeliness, reliability, degree of documentation of the existing software systems and resources. They are used to monitor the results achieved through the improvement initiatives.

The encouraging result of this experience is that the improvement actions which resulted from the first assessment allowed both to achieve a higher maturity level and capability profile and, in the same time, to achieve better values of the selected indicators, showing that a real improvement in the SPU's and Project's performance have been obtained.

In several cases the BOOTSTRAP methodology has been used to analyse the adopted software engineering practices in organisations willing to prepare and submit to ISO 9001 certification. The experience has shown that the improvement plan based on BOOTSTRAP assessment results were helpful to effectively establish the quality system and successfully undergo the certification process.

6. Conclusion and further activities

The development and application of the Bootstrap methodology have provided a good opportunity to understand the state-of-the-art and the needs in software process improvement. This experience is now being exploited through the following activities:

- contribution to the SPICE (Software Process Improvement and Capability dEtermination) project to develop a standard for software process assessment to be published as an ISO IEC JTC1/SC7 technical report [12] and [13];
- Training and dissemination activities, both on the BOOTSTRAP methodology and on process improvement approaches, performed within various events including the ongoing European ESPITI initiative.

The BOOTSTRAP Institute is also actively engaged to further develop the BOOTSTRAP methodology, where the main goal is now to achieve compliance with the emerging ISO standard ([12] and [13]).

References

[1] Humphrey, W.S. and Sweet, W.L., "A Method for Assessing the Software Engineering Capability of Contractors", SEI Technical Report SEI-87-TR-23, September 1987.

[2] ISO 9001. Quality Systems. Model for Quality Assurance in Design/Development, Production, Installation and Servicing. International Organisation for Standardisation, Geneva (1989)._

[3] ISO 9000-3. Quality management and quality assurance standards. International Standard. Part 3: Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software. ISO (1991).

[4] ISO 9004-4. Quality management and quality system elements. International Standard. Part 4: Guidelines for quality improvement. ISO (1993).

[5] ESA Software Engineering Standards ESA PSS-05-0. Issue 2. ESA Board for Software Standardisation and Control, European Space Agency, Paris (February 1991).

[6] Kuvaja, P., and Bicego, A., BOOTSTRAP: Europe's assessment method, IEEE Software, Vol. 10, Nr. 3 (May 1993), pp. 93-95.

[7] Kuvaja, P., Similä, J., Krzanik, L., Bicego, A., Koch, G., and Saukkonen, S., Software Process Assessment and Improvement. The BOOTSTRAP Approach. Blackwell Business, Oxford, UK, and Cambridge, MA 1994.

[8] Humphrey, W. S., Managing the software process. Addison-Wesley Publishing Company Inc., Reading, Mass. (1989).

[9] Kuvaja P., BOOTSTRAP: A Software Process Assessment and Improvement Methodology, in Paolo Nesi: (Ed.): Objective Software Quality, Springer-Verlag, Germany, 1995, pp. 31 - 48.

[10] Herbsleb J., Carleton A., Rozum J., Siegel J. and Zubrov D., "Benefits of CMM-based Software Process Improvement: Initial Results", SEI, August 1994, CMU/SEI-94-TR-13

[11] Damele G., Bazzana G. and Maiocchi M.,Quantifying the benefits of software process improvement in Italtel Linea UT exchange in Proceedings of ISS 95 - XV International Switching Symposium - Berlin, April 23-28, 1995

[12] Paulk, M.C., Konrad M. D., An overview of ISO's SPICE project, American programmer, February, 1994, pp. 16 - 20.

[13] Coletta, A., The SPICE project: an International Standard for Software Process Assessment, Improvement and Capability Determination,in Paolo Nesi: (Ed.): Objective Software Quality, Springer-Verlag, Germany, 1995, pp. 49 - 63.

[14] Haase, V., Messnarz, R., Koch, G., Kugler, H. J., Decrinis, P., Bootstrap Fin Tuning Process Assessment, IEEE Software, July 1994.

LEONARDO - THE NETWORKING PROGRAMME

ISCN CONFERENCE
SEPTEMBER 1995

John Stewart
Hibernia Learning Partnership

On 6 December 1994 the Council of Ministers of the E.U. adopted the Leonardo da Vinci programme. The Leonardo programme is the European Union's main policy instrument for supporting training in Europe between 1995 - 1999. In this paper I would like to outline the main features of the programme and present the project development opportunities available within it. Thirdly, I will give you some practical examples of projects and how they have been developed and finally I would like to outline other development opportunities available to small companies under some of the other EC initiatives such as the 4th Framework Programme, European Structural Funds and the SME facility.

LEONARDO

The Leonardo programme will run to the year 2000. Its objective is to prepare European industry for the 21st century by improving the quality of human resource training systems and training products available to the industry. With a budget of 600 Million ECU and its focus on training, it is an avenue small businesses are well advised to explore. Indeed, SME involvement is one of the stated aims of the programme.

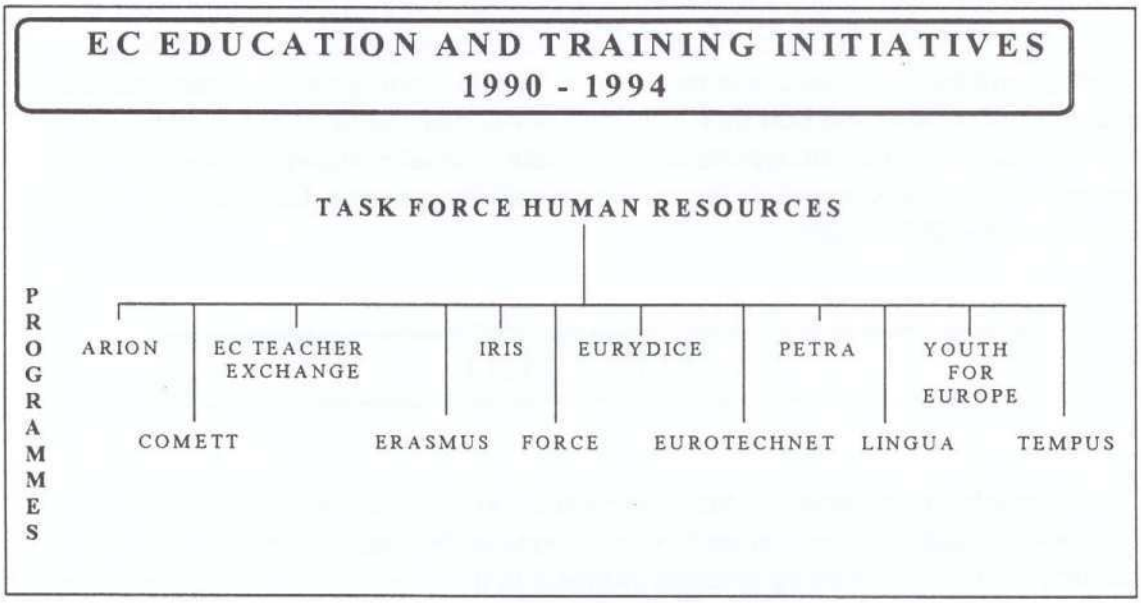
Leonardo will fund the design and development of training networks, training materials and programmes as well as the delivery of short intensive courses that rely on the combined expertise of universities and industry. Up to 75 % of the cost of these projects can be funded. A network of support agencies is in place from the previous programmes such as the Hibernia Learning Partnership which will facilitate small companies involvement in this programme through information and advise on project development and applications, as well as making the necessary links with higher education and European partners.

Leonardo will also provide grant aid for students, graduates and young workers to work in companies for up to 12 months as well as funding international exchanges of staff between companies and to third level institutions to upgrade their expertise.

BACKGROUND

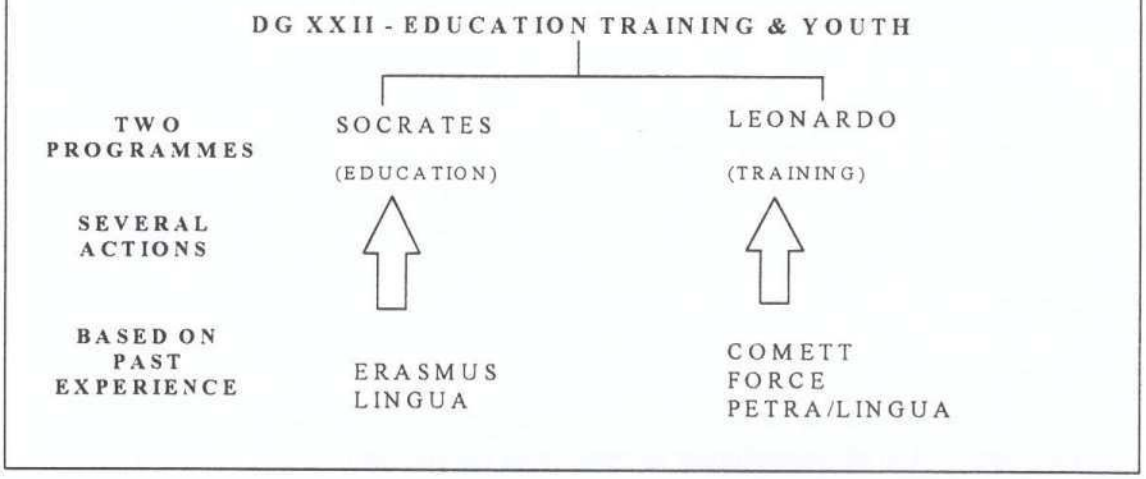
Prior to 1995, the EC Task Force for Human Resource, Education and Youth operated a range of programmes focusing on strategic issues.

The COMETT programme sought to advance and disseminate technological expertise,
The FORCE programme focused on vocational training,
EUROTECNET on innovation in training,
ERASMUS on higher education initiatives,
LINGUA on languages and
PETRA on initial vocational training and education.



Each programme had its own deadlines, programme and administration. The EC has now sought to simplify and rationalise these structures. The Maastricht Treaty was a watershed in EU's education and training policies, as Article 127 gave EU a legal responsibility to implement a vocational training policy for Europe. The result is the creation of DG XXII and the bringing together of all these previous strands together under two main programmes - Socrates for education and Leonardo for training.

EC EDUCATION AND TRAINING INITIATIVES 1995 - 1999



What will Leonardo support?

The programme has 4 Strands

LEONARDO DA VINCI

THE MEASURES

STRAND I	SUPPORT THE QUALITY OF MEMBER STATE SYSTEMS & ARRANGEMENTS
STRAND II	SUPPORT FOR IMPROVEMENT OF TRAINING MATERIALS AND DELIVERY
STRAND III	(a) LANGUAGE SKILLS (b) TRAINING KNOWLEDGE SUPPORT
STRAND IV	SUPPORT MEASURES

OPPORTUNITIES FOR HUMAN RESOURCE DEVELOPMENT

There are 5 particular opportunities for training projects targeted at industry mainly under Strand II & III of Leonardo.

1. PILOT PROJECTS

The design and development of training materials and courses. These must be implemented by a pan-European partnership and focus on improving technology and its application to industry and enhancing its competitiveness. These pilot projects can be funded up to 100,000 ECU per annum although 70,000 ECU is a more typical level of support.

2. SHORT COURSES

The design and delivery of short intensive training courses to transfer technology, enhance application of technologies in companies particularly SMEs - funding for these courses will be smaller than the pilot projects.

3. NETWORK

Leonardo will support regional or sectoral networks to develop collaboration on a pan-European basis between industry, education and social partners.

It must be said that SMEs have had particular difficulties in accessing EU project programmes. These networks are the ideal channel for companies with similar interests to come together and improve the application of technology and develop their human resource through joint training initiatives. The ISCN would seem to me to be quite a good example of how these network projects work. Funding will be up to 100,000 ECU but again typical funding may be half this amount.

4. HUMAN MOBILITY

Leonardo will fund travel and susbsistence costs for exchanges of staff throughout Europe, so SMEs can send their training manager or production manager to other companies or universities to develop their expertise and build European contacts.

Leonardo will also grant aid the placement of students graduates to work in companies for periods of 3 - 12 months. It is very easy to access such grant aided students (all you need to do is to outline the job profile and candidate profile you propose and send it to Hibernia Learning Partnership or any other student placement facilitator such as APS.

5. MULTIPLIER EFFECT PROJECTS

Funding for multiplier effect projects is available through Strand III. They should seek to disseminate innovative training expertise or methodologies throughout Europe, especially skills and experiences which have been developed through other E.U. initiatives such as 4th Framework programme.

EXAMPLES OF PROJECT DEVELOPMENT

In order to demonstrate how these programmes can work, I would like to use some practical project examples that I have developed as manager of Hibernia Learning Partnership.

Hibernia Learning Partnership was set up in 1987 as a University Enterprise Training Partnership under the COMETT programme. We are a private limited company but our aim is to improve other companies. The objective of the partnership is to facilitate regional development through the enhancement of human resources. Our role is to act as a pro-active catalyst in bringing together expertise from both university and industry sectors to collaborate on human resource initiatives. As such we focus on Training Needs Analysis, the development of university/industry collaboration and networks and the design of training programmes and Technology Transfer initiatives to address the needs identified.

The Hibernia partnership uses 12 member colleges as platforms for regional development to design and deliver training and to disseminate European expertise to industry. Hibernia has invested 750,000 ECU in the design and delivery of some 40 intensive training courses. These have typically been of 2 - 3 days duration and have focused on sectors of primary importance to our region, including the agro-food sector, aquaculture, advanced manufacturing, the environment and software technologies. For example in 1994, Hibernia grant-aided the 1st annual International Software Consultancy Network Conference in Dublin.

This opportunity remains available under the Leonardo provision for short courses.

Hibernia has fostered the development of 10 major training initiatives through the COMETT and FORCE programmes which the objective of advancing technological development and disseminate the expertise through pan-European networks.

Biomerit is a biotechnology network involving 30 partners in 7 countries. It developed from a short training course under the COMETT programme and between 1990-1995 received 500,000 ECU to utilize the expertise within the network to train trainers and industry in the latest Biotechnology applications. Focusing on leading edge technology in a crucial sector, this network has been very successful in advancing technology and ensuring the competitiveness of European industry. This type of opportunity is also available under Leonardo

Aquaculture is another good example of the benefits of Hibernia's regional development strategy. Through our various student placement schemes and short training courses, we built up a network of SMEs and established an aquaculture network. Aquaculture is an industry which by its very nature will be always located in the most peripheral parts of Europe. The network includes 280 companies and universities from 20 countries and has been the platform for human resource initiatives which have secured 1 million ECU since 1992 under the various EU programmes.

The development of such networks on a regional or sectoral basis is open under Leonardo.

Other initiatives of this type include bringing together European partnerships of experts to design and develop training programmes and will give you an example of how to use the pilot projects opportunities:

Training Programme	*Total cost*	*EU support*	*EU Programme*
Airport Management Training Programme	300,000	120,000 ECU	FORCE
Programmeable Logic Controllers	300,000	150,000 ECU	FORCE
Instrumentation	600,000	250,000 ECU	COMETT
HPLC (Chromatography)	600,000	250,000 ECU	COMETT
Training Needs Analysis of the Construction Industry	250,000	115,000 ECU	FORCE
I.T. for SMEs	250,000	120,000 ECU	COMETT

As well as developing training partnerships and projects, Hibernia has been actively pursuing Technology Transfer through human mobility.

Since 1989, Hibernia has grant aided 600 student placements in industry with 1.5 million ECU while Hibernia's staff mobility initiatives have facilitated 16 secondments or visits between Ireland and Europe. Both these mobility initiatives remain available under Leonardo and we will be happy to put you in touch with the appropriate agency.

DEVELOPMENT OPPORTUNITIES FOR SMEs

While the European Union provides significant funds to support the development of small businesses, the myriad of different schemes and administrations and even the language in which they are framed has deterred many companies from availing of these initiatives. In order to simplify access to these programmes small companies should first concentrate on the 4 main areas of direct relevance. These are 1. Human Resources, which I have covered under Leonardo 2. ESF Initiatives, 3. Research and Technological Development and 4. Subsidized Loans.

The EU's Fourth Framework Programme (1994 - 1998) has a budget of £ 1,250 million ECU and is aimed at advancing research and technological expertise in Europe. The framework programme is an umbrella for 20 specific programmes ranging from Biotechnology to Information Technologies to Nuclear Fission Safety! Each of the 20 programmes has its own particular schedule and deadlines, so keeping track of all of them is difficult, particularly for the uninitiated.

There are support services however. The EU has produced a single information package for the "technological stimulation" measures for small and medium companies. The SME infopack covers the exploratory awards and the cooperative research grants common to most of the programmes. There are also a range of national research and technology development grants in each country, many supported by the EU. Perhaps the most productive avenue for companies with research and technological development ambitions is through the university system.

The Universities are major players in the Framework programmes and cooperate with large and small companies across the full range of RTD programmes. As the higher education institutions are placing an increasing emphasis on interaction with small companies, they are the ideal platform for SMEs to launch research collaboration. The universities are very familiar with the workings of these EU programmes and have the best access to European networks, as well as the experience and facilities to undertake the research. They are also alert to other funding opportunities available such as the national schemes mentioned above.

One of the most relevant and accessible Framework programmes for SMEs is the CRAFT programme. CRAFT encourages links between SMEs and universities or research institutes on joint research initiatives. As CRAFT will fund up to 50 % of the costs of the RTD project, it is an excellent opportunity for small companies to avail of the research expertise and facilities in the higher education institutions. The research funding covers a range of industrial and material technologies as well as standards measurements and testing issues such as quality and product developments.

The application deadline for CRAFT is open up to June 1996.

EUROPEAN STRUCTURAL FUND PROGRAMMES

While there are at least 13 separate ESF initiatives, there are three of particular relevance to small businesses.

The aim of the **ADAPT** programme (1994 - 1999) is to make industry more competitive and, as the name suggests, to facilitate adaptation to industrial change. ADAPT has a budget of over 1.4 Billion ECU and the programme will provide up to 75 % of the costs of projects. The deadline for each country varies so those seeking further information should contact their relevant National Authorities or the Offices of the EC in each member state.

The programme will fund specific training initiatives for companies as well as supporting them to identify and adapt to industrial change. On a broader scale, the programme will support the provision of shared services at regional or other levels, train trainers and encourage transnational exchanges between companies.

The **SME Initiative** (1994 - 1999) has a budget of 1 million ECU and was designed to assist small companies to adjust to the single market and become more competitive at an international level. The initiative will fund training, the improvement of production systems and organizational methods, access to new markets and international networking and cooperation. The definition of an SME in this instance is a business which employs less than 250 people, has an annual turnover of less than 20 million ECU and which has not more than 25 % of their capital held by larger companies.

INTERREG II aims to develop cross border cooperation and assist areas to overcome their relatively isolated position It only applies to certain regions, mainly land border regions. Many of the eligible measures could benefit companies in these regions. For example Interreg II will support training and employment initiatives, the establishment of trade and professional groups and the development of support for small and medium sized businesses. Overall Interreg II has a budget of 2,400 Million ECUwith objective I regions assured 1,800 Million ECU.

1000 MILLION ECU IN SUBSIDIZED LOANS

Most small companies planning to increase capital investment coupled with increased staff levels can avail of the EU's subsidized loan scheme entitled "The SME Facility". The objective of the scheme is to support the creation of employment in growing companies by subsidizing capital investment in the industrial, agro-industrial, tourism and service sectors (excluding retail).

Currently, a financial package of 1 million ECU is available in subsidized loans. The fund will offer loans of up to 30,000 ECU per job created with a maximum interest rate subsidy of 3,000 ECU attached. The subsidy is paid in one installment for new jobs in place for at least 6 months. While larger loans are available, the above subsidy limit continues to apply.

The fund is administered by the European Investment Bank and national commercial banking systems coordiante the loan facility in their countries. They can provide you with the application forms and other information. Although there are a number of eligibility conditions, the majority of small and medium sized companies can avail of this facility and priority will be given to smaller companies.

PROGRAMME	FOCUS	BUDGET
1. Leonardo	Human Resource Development	600 million ECU
2. ADAPT	Training, Competitiveness & Employment	1,400 million ECU
3. SMEs	Competitiveness & Adjustment to single market	1,000 million ECU
4. INTERREG II	assist border regions	2,400 million ECU
5. Fourth Framework Programme RTC	Research & Technological Development	1,250 million ECU
6. The SME Facility	subsidized loans	1,000 million ECU

USEFUL ADRESSES

(I) ***Hibernia Learning Partnership***
Mr. John Stewart
12, Upper Fitzwilliam Street
Dublin 2
Ireland
Tel.: +353/1/6621502
Fax: +353/1/6621510

(II) ***APS (Ausbildungspartnerschaft Hochschule-Wirtschaft Südösterreich)***
Mr. Bernhard Posch
Schlogelgasse 9/III
8010 Graz
Austria
Tel.: +43/316/8737195
Fax: +43/316/816340

(III) ***Leonardo Bureau D'Assistance Technique***
9, Avenue d'Astronomie
1030 Brussels
Belgium
Tel.: +32/2/2270100
Fax: +32/2/2270101

(IV) ***Directorate General XXII (DG XXII)***
7, Rue Beliard
1040 Brussels
Belgium

(V) ***4th Framework Programme is operated by DG XII***
I/38, Rue de la Loi 200
1049 Brussels
Belgium
Fax: +32/2/2958220

A help desk has also been established to facilitate interested parties in locating sources of information on research activities

RTD Help Desk
DG XIII Section D/2
JMO B4/077
2920 Luxembourg
Luxembourg
Tel.: +352/43/0133161
Fax: +352/43/0132084

Details on the ***SME*** infopack and the ***CRAFT*** programe as well as other research programmes can be sourced through these two agencies.

(VI) The ***ADAPT*** programme is operated by Directorate General for Employment, Industrial Relations and Social Affairs.

DG V
Section B.4
Fax: +32/2/2969770

(VII) The ***SME Initiative*** and ***INTERREG II*** is operated by Directorate General for for Regional Policies

DG XVI
Section A
Fax: +32/2/2962568

(VIII) The ***SME Facility*** is administered by the European Investment Bank as well as through normal National Banking Systems.

AINSI
An Inductive Method for Software Process Improvement: Concrete Steps and Guidelines*

Lionel Briand, Khaled El Emam and Walcélio L. Melo

Abstract

Top-down approaches to process improvement based on generic "best practice" models (e.g., CMM, TRILLIUM, BOOTSTRAP, SPICE) have become popular. Despite the idiosyncrasies of each of these approaches, they share some common characteristics: all of them are based on numerous assumptions about what are best practices, and about the business goals of organizations and the problems they face. Other organizations, like the Software Engineering Laboratory of the NASA Goddard Space Flight Center, HP and CRIM in Canada, have adopted the Quality Improvement Paradigm (QIP). The QIP stipulates a more bottom-up and inductive approach to process improvement. The focus of this paradigm is to first understand what processes exist in the organization and to determine what causes the most significant problems. Based on this, opportunities for improvement are devised, and empirical studies are conducted to evaluate potential solutions. In this paper, we present a method, named AINSI

(An INductive Software process Improvment method), which defines general but concrete steps and guidelines for putting in place the QIP. This method is the result of the collective experiences of the authors and integrates many lessons learned from process improvement efforts in different environments. It also integrates many complementary techniques such as qualitative analysis, methods for data collection (e.g., the Goal/Question/Metric paradigm), and quantitative evaluation.

1. Introduction

Top-down process improvement approaches (e.g., the Software Engineering Institute's Capability Maturity Model for software (Paulk *et al.* 1993), TRILLIUM (Coallier 1995), BOOTSTRAP (Haase *et al.*, 1994), SPICE (Drouin 1995)) provide a high-level model of what ought to be the process of a software development organization. Such models are based on the consensus of a designated working group about how software should be developed or maintained. They are very useful in the sense that they provide general guidelines to people who do not know where to start improving, and in which order. Also, these models are useful in contract award situations where alternative bidders may be comparatively evaluated.

A general assumption of these models is that the more an organization's processes match the stipulations of the model, the greater its effectiveness on some criteria (e.g., product quality, productivity, predictability etc.). In general, there is a dearth of evidence supporting this assumption. For instance, in the context of CMM-based assessments, Hersh (1993) states *„despite our own firm belief in process improvement and our intuitive expectation that substantial returns will result from moving up the SEI scale - we still can't prove it"* and Fenton (1993) notes that evaluating the validity of the SEI's process maturity scheme is a key

---

* Briand and El Emam are with the Centre de Recherche Informatique de Montreal (CRIM), 1801 McGill College Av., Suite 800, Montréal (Quebec), H3A 2N4, Canada;. Melo is with the University of Maryland, Institute for Advanced Computer Studies, College Park, MD, 20742. E-mails: { lbriand | kelemam}@crim.ca, melo@cs.umd.edu.

contemporary research issue. There is now some *initial* evidence supporting a positive relationship between the CMM's maturity levels and some subjective criteria of including quality and productivity (Goldenson and Herbsleb 1995). In addition there is some *initial* evidence based on two BOOTSTRAP case studies showing that when weaknesses identified by an assessment are addressed, improvements in quality and productivity result (Haase *et al.*, 1994). Similar studies are being conducted as part of the SPICE project (El Emam and Goldenson 1995). However, despite such commendable efforts, the overall evidence remains weak and is contradicted by other works (El Emam and Madhavji 1995c).

Furthermore, in practice, it is not so obvious that these models will address the causes of problems in all organizations. For example, an analysis of findings from assessments based on the CMM (for which the appropriate data was available) identified that more than half the organizations had problems in areas not covered by the CMM (Kitson and Masters 1993).

Alternatively, one could adopt a more inductive approach to process improvement where the focus of the first step would be to understand what exists in an organization and determine what causes significant problems. Then, solutions could be devised and evaluated in pilot studies or even controlled experiments (Basili, 1992). Only after a solution is found to be effective and efficient, then it should be integrated into the existing process or the process may be modified. Such an approach is inspired by the Quality Improvement Paradigm (QIP) recommended by several researchers and practitioners (Basili, 1992). The QIP is a specialization of the scientific method for software engineering research.

The question now is to determine how such an approach can realistically be implemented in different software organizations. One well known and typical example is the Software Engineering Laboratory (SEL) at the NASA Goddard Space Flight Center, where the understand, assess, change paradigm has been in place for the last 20 years. Experience has shown that the cost of such a continuous software process improvement mechanism can be kept under a reasonable percentage of the development cost. This paper is in large part based on the SEL experience and a few other process improvement experiences in which the authors were involved (Briand *et al.*, 1994; 1995; El Emam and Madhavji 1995a; 1995b).

This paper is structured according to the phases of a process improvement activity as we see it. Although these phases are presented sequentially, in most cases they are overlapping. We try not to focus exclusively on technical issues but to provide a complete overview supported by key references.

2. Set up a process improvement team

The first and most critical activity at start-up is getting the commitment of management to software process improvement. One of the major reasons why process improvement programs fail is that management did not provide the adequate resources to make it happen. It is the case that many organizations whose main business is not software development may not consider software to be important. Therefore they are reluctant to invest in software process improvement. Efforts within such an organizational context are commonly an uphill battle with management to make resources available for improvement

It is widely recognized that a process improvement team separate from the developers must be set up at the start of an improvement effort (Fowler and Rifkin 1990; Software Engineering Laboratory 1995). The main reason for this is that the priority of the developers is to produce systems, and therefore process improvement activities will not receive as much

attention as they deserve or need. A process improvement team has process improvement as its priority.

The SEI calls this process improvement team a Software Engineering Process Group. This group would be staffed, ideally full-time, by experienced technical and credible personnel from the organization, and may also be staffed by researchers and external consultants. Similarly, the SEL calls such a group the "analysts" who have a mix of backgrounds in empirical methods, data analysis, and software development and maintenance. Some are experienced researchers and scientists, others are senior managers or software engineers. It is important that the individuals in this group have strong technical knowledge and are respected in the organization. One sign of a lack of management commitment to process improvement is when they assign those people that they "don't know what else to do with" to a process improvement team.

Whatever the label, the process improvement team is responsible for initiating and carrying out many of the process improvement activities, such as setting up training, designing evaluation studies, defining the goals of measurement, studying developed projects to feedback to the organization any lessons learned, following the execution of actual activities in order to find deviations and opportunities for improvement, etc.

The mandate of the process improvement team must be well defined and must be tied to important business goals. In some cases, the process improvement team is formed with the mandate to help the organization become ISO 9001 certified within, say, one year. This may considered to be tied to business goals because it will help the organization get contracts or because it is demanded by a major customer (especially in Europe). Similarly, some organizations consider achieving Level 3 on the CMM to be important for business because it would help them get contracts (especially for the defense community in the US). Progress is then measured by the proportion of Level 3 practices that have been implemented or the extent of compliance to ISO 9001 clauses. Success is measured by having all necessary practices implemented and ISO 9001 clauses adequately satisfied. This, using the terminology in (Schaffer and Thomson 1992), is termed an activity-driven approach to improvement.

Strict adherence to the activity-driven approach has a number of disadvantages. The organization does not need to actively measure important variables like product quality, productivity, and meeting budget and schedule targets, because these are not needed to evaluate progress and success. If they do, this data is not used to evaluate progress and success. Therefore, the organization may continue to put practices in place without knowing the effects on product quality, productivity, and predictability. Consequently, the organization would not know if all the activities that have been implemented really provided any quality, productivity, or predictability benefits.

Putting practices in place on the faith that they are somehow "good" is not prudent. As noted in (Basili, 1992), many of our intuitive ideas about software development may turn out to be incorrect once put to empirical test. In general, many practices that are in current use in software engineering have not been adequately empirically tested, and so we do not know if they really work, and under what conditions they do work. If the organization expects practice X to improve, say product quality, then the process improvement team's mandate would include improving product quality. The process improvement team may subsequently decide that practice X is a worthwhile candidate, and evaluate it internally within the

organization to determine if it really improves product quality and by how much. It is therefore important not to confuse or be ignorant of the difference between cause and effect. Process improvement efforts ought to be driven by the desired effect, i.e., results driven.

The process improvement team should start by creating an awareness in the organization about the existence of the team, its mandate, objectives, and approach. The purpose of this is to enlist the support and cooperation of the organization's members. This is necessary in preparation for the subsequent activities of the improvement effort.

3. Model the existing process

Process modeling may serve many purposes and is believed to have many benefits (Finkelstein *et al.* 1994). In the context of process improvement, descriptive process models are useful for understanding the way things currently work in the organization and for communicating this understanding. Below we discuss four important issues in process modeling: (a) how to collect information for process models, (b) what information to collect, (c) the formalisms to use for modeling, and (d) the appropriate level of detail of process models.

There are a number of methods for eliciting process modeling information. One method based on experiences at Contel is given in (McGowan and Bohner 1993). This has four main phases: observe the organization and interview staff, determine context, purpose, and viewpoint of model, construct process model, and verify the process model. Madhavji *et al.* (1994) have generalized from their experiences and presented a method for eliciting information suitable for the construction of process models. Some of this method's main steps that are most relevant here are: understanding the organizational environment, defining the objectives of the modeling exercise, planning the elicitation strategy, developing the process model, and validating the process model. Similarly, Briand *et al.* (1994; 1995) have described their acquisition process for collecting information suitable for the construction of models. The steps in the acquisition process include: determining the hierarchical structure in the organization, determining the roles covered by the positions in the hierarchy, and identifying the various dependencies within the reporting hierarchy. These three methods should serve as reasonable starting points for a process modeling effort. Furthermore, a discussion of issues to consider in a real world process modeling exercise is given in (Henry and Blasewitz 1992). In particular, the authors of that article suggest tasking the developer groups to build models describing their own processes.

Some of the basic questions that need to be answered in a process model are: What are the tasks in the process? What are the dependencies between these tasks? When do the tasks start and end? Who are the actors that perform these tasks? What are the interdependencies between the actors? A conceptual framework defining information that could be collected has been presented in (Armitage and Kellner 1994). This framework has three entity classes (activities, agents, and artifacts) and two aspects (relationships within and among entity classes, and behavior of the entity classes and the relationships). Another article presents a set of process modeling attributes which has perspectives (such as process steps, roles, resources, and constraints) and properties for each of the perspectives (Madhavji *et al.* 1994). Based on contemporary knowledge, these articles provide a comprehensive view of process information that can be collected.

Sources of information that we have found useful for answering these questions and for collecting relevant information are: interviews with members of the organization, inspection

of process documentation if any exists, inspection of project management documentation (such as project plans), and direct observation of development staff (for example "shadowing" and attendance of meetings). One issue that ought to be considered, especially in a large and diverse organization and when the modeling effort has a wide scope, is differing terminologies used within the organization. A common terminology (that is acceptable to all developers in the organization being modeled) should be found. Suggestions for achieving this include: debating the issue until an acceptable compromise has been found, giving the components of the model (e.g., the documents produced or the activities) identifying numbers, and using terminology from public standards (e.g., military, IEEE or ISO standards).

Many formalisms for process modeling have been developed. Few of these formalisms have actually been used in practice. Some that have been used and that have been found to be useful are Statemate (Kellner and Hansen 1989), structured English, ETVX (Radice et al. 1985), the Actor-Dependency modeling approach (Briand et al. 1995; Yu and Mylopolous 1994), the commonly used Data Flow Diagrams (Frailey 1991) and the SADT notation (McGowan and Bohner 1993). The latter four support a single perspective, while the former integrates multiple perspectives (these are state-transition, data flow, and structure).

The models developed at this point will be useful for the analyses to be conducted later. It should be remembered, however, that the developers must be able to understand the formalism that is used. Therefore, a formalism that is most similar to those that are commonly used in software development, for example data flow diagrams or control flow diagrams, is likely to be the most easily understandable and rapidly accepted, even if it is considered less „powerful" than some other formalism. Another issue that needs to be considered is the ease of change of the process models. As the process improvement effort progresses, it will be necessary to change the process models. The formalism that is chosen ought to be easy to change. This may be facilitated by the use of an automated tool that supports the chosen formalism, e.g. TEMPO (Belkhatir and Melo 1994).

It is difficult to decide how deep we must go into the software process definition and modeling. In general, we would discourage the development of very detailed models, mainly because we have not found this excessive to be useful and their construction consumes considerable resources. It must be remembered that the objective is not to automate software processes, it is only to understand and communicate.

The process models that are developed will be used in subsequent steps of the improvement effort. However, the models also help the developers understand their process and how it fits into the overall organization. To be useful for developers, the models must make sense to them. The best sign of this is when the developers make copies of (parts of the) models and hang them in their offices; then they are finding them useful.

4. Conduct qualitative analysis

The goal at this stage is to identify the causes and inner mechanisms that lead to costly or risky problems related to the quality of the delivered products or the efficiency of the development process. The analysis described in Section 2 is providing the context in which the analyst can inquire about the problems. A well defined process model will help differentiate realistic working hypotheses about problems from non-relevant issues. There are three main alternative techniques for performing qualitative analysis in the current context: structured interviews as performed in knowledge acquisition, causal analysis of problems,

and well-designed questionnaires. In fact, these techniques are usually complementary since they allow cross-checking of information.

When performing causal analysis, the analyst usually starts from a problem report coming from the testing phases or operation of a particular, but representative, software system. The analysis that will lead him/her to determine what happened will be based on multiple sources of information: the process documentation, the product documentation, the project (release or new development) documentation, (structured) interviews (Vogel, 1988) with the people involved in the process and, if possible, the originator of the problem report and the owner(s) of the product part where the problem was identified.

Of course, it is difficult to provide a precise procedure to perform such an analysis and the experience and talent of the analyst will be an important criterion for success. However, several useful guidelines exist in the literature (Collofello and Gosalia 1993; Chillarege et al. 1992; Nakajo and Kume 1991) and, for example, the authors of this paper were involved in defining various methodologies focusing on maintenance (Briand et al. 1994; 1995) and the requirements engineering process (El Emam and Madhavji 1995a; 1995b). Eventually, in a given organization, typical causal chains leading to problems can be devised by senior project managers and validated by causal analyses such as the ones discussed in this paper. This would help the originator of an error or a problem report to provide an explanation for what happened in a consistent terminology. This would, as a consequence, facilitate any future analysis to a great extent.

Performing causal analysis for a fault consists of determining: the fault type or category, the phase where the fault was found, the verification process that uncovered the fault, the phase where the fault was created, the cause(s) of the fault in terms of organizational or process flaw (when relevant), its severity, and the solutions that might prevent the fault from occurring in the future. In a given organization, typical fault occurrence patterns should be identified by associating types of faults to likely process and human causes. Based on our experience, an empirical investigation of a few representative projects ought to be sufficient to identify such patterns. In ideal cases, causal analysis should be performed shortly after errors have been detected and changes have been implemented and tested. However, very often, in order to speed up the analysis, the analyst has to study completed and carefully selected developments or releases. Also, not only faults should be investigated but delays or budget overruns could provide interesting insights.

Examples of fault classification schemes are provided in the references above. Such schemes need to be developed based on the specific experience of each organization so that they make sense for the organization's software engineers and technical managers. However, at the beginning, external experience reports can be a very useful source of ideas. The references provided above should be suitable in that context. Because causal analysis can be expensive and time intensive (it requires talented and experienced people), it should focus on costly changes or high-severity faults that could have led to significant problems in operations.

For example, Nakajo and Kume (1991) describe the following procedure. They link program faults with system failures. Then, they attempt to determine what type of human error is involved (e.g., communication error within the development team). Last, the human error is explained in terms of work system flaws (i.e., inherent characteristic of methods, workers and environments affecting human error occurrence and deviation.) Patterns of fault occurrences were identified by looking at the statistical associations between program fault types, human

error categories and work system flaw categories. For example, interface faults (mismatching between software components) was found to be closely associated with communication problems between various component's development teams. It is important to note that taxonomies of work system flaws are inherently specific to the organization under study and, as a consequence, reusing causal analysis techniques is not always straightforward.

A general procedure to perform causal analysis would follow the following outline (Briand et al 1994; Collofello and Gosalia 1993):

1. Select representative system development(s) or release(s).
2. Obtain all problem reports.
3. Classify program faults according to pre-defined severity and type taxonomies.
4. Determine the causes of high-severity faults in terms of human errors and then process flaws.
5. Develop recommendations for process changes and validate them with the participants of the causal analysis (one should attempt to reach a maximum consensus on issues).
6. Refine causal analysis guidelines and taxonomies to help improve future analyses.

Performing such procedures assumes the existence of a well-defined process and organization model. Without a clear context for reflection, these steps are difficult, if not impossible, to perform, since consistency checks between the description of the process and organization, the documentation available, and the stories gathered during interviews are necessary. Such verification may lead to refine the perceptions the analyst has of a problem report but may also lead to the refinement of the defined process and organization models.

5. Define and document an action plan

Once the process in place and its main weaknesses and strength are well understood, an action plan must be defined. In (Fowler and Rifkin 1990), the action plan is defined as: *"A formal, written response to the (process) assessment, and the 'map' for improvement."* The action plan is extremely important for several reasons. First, it is required in order to get a suitable budget for the next phases (evaluation of solutions, changes to the process). Second, it is crucial to convey the right information to management and developers about the importance and difficulty of what is going to be achieved. This is particularly important since two common sources of failures for improvement programs are, among others, the lack of adequate funding/resources and senior management commitment.

In practice, it is very common to see action plans which do not specify all the activities involved (e.g., non-technical activities such as training), the responsibilities and prerogatives of all participants, the risks taken, and corresponding contingency plans. Process improvement in software is always exploratory and risky. An action plan should be carefully thought out and not overlooked. Once promises have been made to senior management and once participants have expectations, the software process improvement team has to make sure not to disappoint them, or the consequences might be disastrous for the future of process improvement in the organization. The participants should be made aware of all uncertainties and new challenges to come.

Another important point is that interdependencies between action plan tasks should be carefully studied. The impact of underestimating the efforts of certain tasks can create significant damages to the pertinence of the remainder of the action plan. It is very frequent, for example, to see action plans based on a very superficial understanding of the processes in

place, or inspections introduced without adequate data collection to evaluate their impact or refine them.

Actions plans contain, at least, three levels of decomposition. Each step of improvement describes an increment from a lower to a higher state of practice, e.g., substitute informal peer reviews by inspections. Within each step, sequential and overlapping phases of improvement actions are described, e.g., for inspections, phases such as preparation of participants and material, implementation, and evaluation on pilot projects are commonplace. Within each phase, a set of (usually unordered) activities should be described to better convey the size of the phase and better estimate its required effort, e.g., for the preparation phase, design the inspection checklists and error log forms. Each document to be written, tool to be developed, training course to be given, etc. should be listed in the description of the activities.

A possible high-level outline of what an action plan could look like is as follows:

1. Corporate improvement objectives and motivations.
2. The various groups and participants of the process improvement action plan, their responsibilities and prerogatives (high level at this stage).
2. Improvement steps, their connections to corporate objectives, their entry and exit criteria.
3. The phases leading incrementally to the exit criterion of each improvement step, the risks and uncertainty associated with each phase and the corresponding contingency plans, the managers in charge of monitoring and organizing the execution of phases.
4. The set of activities involved in each phase of each step (normal procedures or contingency plans), the participants in each activity and their responsibilities (outputs, quality, schedule).
5. The effort and cost for each step and phase, with uncertainty intervals.
6. The expected benefits, based on external experiences and whatever information exists about the organization's productivity and quality.

6. Set up a measurement program

At first, a measurement program is generally designed to (1) provide a quantitative baseline of comparison for future process changes, (2) better understand the issues that may or not have been identified in the qualitative analyses preceding measurement (e.g., the high-cost of certain activities may make them priority improvement targets), and (3) make the process of decision making less risky by providing accurate and on time quantitative management information about the software products and the processes used to produce these products. There is no possible improvement or even technology assessment if one does not know the current quality and productivity baselines of his/her organization in terms, for example, of error density and KLOC's per man-month. Such a common sense principle is very frequently forgotten under the pressure for quick improvement and immediate tangible results!

In addition, it has been shown in many occasions that a measurement program is more likely to succeed if its definition is driven by corporate and project goals, e.g., reduce development costs by reducing requirements' definition instability (Bassman *et al.* 1994; Basili and Rombach 1988). The data collected must closely reflect in their definition, format, and collection procedures, the local constraints, practices, and needs of the organization.

Typically, a measurement program collects a mix of product and process data about pre- and post-release defects, the effort breakdown per activity and phase, the complexity and size of

the systems under development or maintenance, etc. In the details, however, the definition of what a phase or an activity means, the classification taxonomies of defects, and the complexity measures assessed as relevant, can significantly vary from one environment to another.

Once a measurement program has been defined, participants must be trained in the data collection procedures and their feedback must be considered in order to pre-validate and refine the data collection program. It is important to recognize that the commitment of participants is one of the most important success criteria. Also, a great deal of useful information cannot be collected under the form of quantitative data. In some circumstances, questionnaires may be needed. A substantial literature exists in the social sciences in order to help collecting qualitative data (e.g., qualitative scoring scale) .

In general, data collection can be used for many different purposes. Post-mortem analysis of projects is one, e.g., build a project cost estimation model. Understanding project dynamics and evolution across phases is another one, e.g., the distribution of cost across phases. Measurement can be used for the sake of characterization (e.g., what are the most frequent origins of errors), evaluation (e.g., are my inspection techniques effective?), prediction (e.g., cost estimation model), and, of course, improvement (e.g., what are the most frequent causes of high-severity errors).

The usual steps of implementation of a measurement program can be summarized as follows :

1. Define corporate and measurement goals.
2. Derive models (evaluation, prediction) and measures that appear suitable in the specific context of measurement.
3. Define data collection procedures to collect valid and accurate data.
4. Train participants to data collection procedures.
5. Test data collection and validation procedures on pilot projects, simplify and refine them as needed.
6. Expand the use of measurement to the whole organization.
7. Analyze collected data to assess the usefulness of the data and the accuracy of the models.
8. Refine models, measures, and data collection procedures, return to step 4.

Some usual sources of problems when implementing a measurement program are:

- A lack of thorough technical reflection about the defined measures and models, their implicit assumptions and implications, their properties, and how valid they are in the specific context of measurement.
- Data collection is not goal driven.
- The goals of measurement are not clearly specified and communicated to participants.
- Too many goals are addressed at once.
- Conflict between high-level (corporate) goals and lower-level (practitioners) goals. If only top-down (manager) goals are defined, developers may feel unmotivated in participating in the data collection. On the other hand, if only bottom-up (developer) goals are set, managers may not be interested in spending resources on a measurement program. Compromises should, thus, be found.
- Data collection procedures are not well-designed and disruptive.
- Developers are asked to collect too much process data.

- The people in charge of measurement do not have the right training, education and/or experience.
- There is no real management commitment, participants are not clearly convinced and/or feel threatened by measurement.

For more details, see (Bassman *et al.* 1994; Briand, Morasca and Basili, 1994; Hall and Fenton 1994; Grady 1992).

7 Perform a pilot project

Once problems are identified, options for improvement ought to be defined and evaluated. Example options would be the introduction of new practices, like code inspections, changing of programming language (e.g., Waligora *et al.* 1995), the introduction of new tools, like tools for enforcing coding standards, etc. In all cases, however, the identified problems should guide the identification and selection of possible options.

Ideally, a pilot project would be initiated to evaluate the new practice(s) and/or tool(s) (henceforth referred to by the generic term *technology*) that have been selected. Evaluation in the context of the host organization is important. This is because for many new, or even old, technologies in software engineering, there is little empirical evidence supporting their claimed benefits. A good discussion of this is given in (Fenton *et al.*, 1994) for formal methods, and a good example in (Melo *et al.*, 1995; Basili *et al.*, 1995) for Object-Oriented methods. For instance, most OO methods currently available have never been empirically validated. In addition, even when there are a handful of empirical studies evaluating a technology, and showing positive results, one would not be prudent to conclude that the benefits are achievable in his/her organization. Pilot projects would also have other benefits, such as providing feedback for *tailoring the technology* to the organization, and *generating buy-in* within the organization for the new technology.

The scope of the pilot project has to be defined. This is the number of projects and personnel that will take part in it. In general, a small part of the overall organization (in terms of projects or personnel) would be selected. In small organizations, it may sometimes be necessary to include all personnel in the pilot project.

A decision has to be made regarding the evaluation strategy. Kitchenham et al. (1994) have identified three strategies that may be followed: (a) formal experiments, (b) case studies and (c) surveys. In the same the authors express a strong preference for case studies in conducting evaluations in industrial settings. However, another evaluation strategy may be followed, namely quasi-experiments (Cook and Campbell 1979).

In quasi-experimental designs, it is recognized that there are many variables that are difficult to control (as one would attempt to do in a formal experiment). In particular, subjects are not assigned to treatment and control groups randomly, resulting in nonequivalent groups. To illustrate this point, we take a simple example. Assume we were to evaluate a technology by letting a group of people use it (the treatment group), and comparing their performance to another group that does not use the technology (the control group). In a formal experiment one would assign subjects to the two groups randomly (e.g., using random number tables).

For a pilot project in an industrial setting, this approach may not work because of the setting and because there are other objectives to the pilot study than just evaluation (e.g., create buy-in). For instance, there may be specific individuals that you want to be in the treatment group.

Also, there may be a diffusion of treatment from the treatment to the control groups if they are in close physical proximity and/or if they interact extensively on a regular basis; in such a case we may select the two groups to minimize interaction. If there is diffusion of treatment, then the control group will no longer serve the no-cause-baseline function, and the two groups will be more similar than planned. Another effect that has to be considered is that of withholding treatment. Given that in some cases subjects in the pilot study may have volunteered (by themselves or by their managers), it may not be prudent to withhold the treatment (i.e., the technology use). In the case of random assignment, the subjects will not know until after volunteering whether they will be in the treatment or control groups. This may create resentment, which is not a good thing if you want to create buy-in within the organization.

For the above reasons, quasi-experiments are suggested as an alternative for formal experiments. It should be noted however that if the unit of analysis is larger than the individual programmer or small team, the expense of any such experimental design may be too large to justify.

A simpler strategy is to use case studies. With case studies one compares the outcome of the pilot with an existing baseline. It would not be useful to conduct a case study where no baseline exists. A good set of examples of case studies are the experiences of the SEL with ADA and an Object-Oriented analysis/design method (Waligora *et al.* 1995).

The selection of projects and/or personnel for pilot studies must balance three concerns (Leonard-Barton and Kraus 1985; Veryard 1987): (a) the risk of failure, (b) achieving the benefits that are expected (i.e., the value of the technology), and (c) their representativeness and credibility to the rest of the organization.

If, for example, a low risk project is selected in order for the improvement effort to survive politically, it may not provide benefits or be representative. If the personnel selected are the most capable in the organization, then it would not be clear from the outcomes whether the benefits were due to the people or the technology. The same applies if the worst capable personnel were selected. The motivation of subjects or project managers taking part in the pilot should also be examined. If such personnel are not too eager, then the pilot project may fail. Conversely, if such personnel have too much enthusiasm, this may be indicative of another agenda that may place the project at risk. Some general recommendations on selection of personnel are (Kitchenham et al. 1994):

- do not allow technology "champions" to run evaluation exercises
- use normal procedures to staff pilot projects; do not select people who are particularly enthusiastic or particularly cynical about the technology

The pilot project should be sufficiently small and simple (otherwise it may take too long and may overtax the as yet novice personnel). It must also be large and complex enough to be credible to the remainder of the organization and to provide realistic feedback for tailoring the technology.

It is critical that the participants in the pilot project receive adequate training in the new technology. Training is NOT an overview or briefing session. A training course will properly prepare the participants to use the technology to solve realistic problems. For the training course(s) all the necessary material for conducting the pilot must be prepared. For example, for training on inspections, all the data gathering forms and inspections procedures must be

available. Furthermore, the pilot project should commence shortly after the training so that participants do not lose interest and knowledge with the passing of time.

8. Change the process and the organization

Subsequent to the pilot project and the review of its results, the technology would be transferred to the remainder of the organization. Lessons learned from the pilot should be used to tailor the technology and its training materials to the organization. Also, the data collection procedures may have to be modified to take account of lessons learned in the pilot project.

To transfer the technology to the remainder of the organization, a number of issues require particular attention. First, the reward structure must be changed to reflect the goals of the technology and/or to be congruent with the technology. For example, if the current organizational culture rewards productivity, then the introduction of structured analysis and design methods, which slow down the early phases of a project, are unlikely to be adopted by the organization, or if adopted are unlikely to be used effectively. Second, there must be a number of consultants expert in the technology available to solve problems and, if necessary, to provide coaching to members of the organization.

Given that the introduction of the technology will involve changes to the process and possibly to the organizational structure, the process and organizational models that were developed earlier have to be modified to reflect the changes. This ensures that the models are current.

Depending on the resources available and the organization's capacity to change and to support change, the implementation of the technology may have to be incremental. This means that only a part of the organization changes at a time rather than the whole organization.

It is also necessary to continue the collection of data on an on-going basis. This will allow for continuous monitoring and evaluation of the technology that was inserted in the organization.

9. Conclusions

We have presented in this paper a bottom-up approach for the practical improvement of software processes and products (AINSI: An INductive Software process Improvement method). It may be seen as an instantiation of the more general Quality Improvement Paradigm (Basili, 1992). AINSI differs from top-down approaches which are intended to provide an ideal framework for process improvement, e.g., SEI CMM, because it relies on a more detailed interpretation of available and interdependent sources of information on the software organization. Therefore, we attempt to identify issues with very little preconceived ideas about how the process should look like. In order to facilitate its application in different software development and maintenance organizations, we have provided a set of concrete steps and guidelines. Actually, we have tested and constantly refined it based on the experience we have been acquiring by conducting qualitative and quantitative studies on public and private organizations. In the remainder of this section, we present some lessons learned based on these studies.

First, we think one should not confine oneself to a single learning strategy (e.g., only controlled experiments). On the contrary, it is important to remember that there are many different ways to learn. We should use our common sense to decide which strategy is most appropriate given the decision we want to make and the resources we have available. For

example, controlled experiments allow a better level of control of the factors under study; and they can be more easily replicated in different environments. However, this is because it may be difficult or impossible to apply such an approach that social scientists have developed techniques such as quasi-experimentation (Cook and Campbell, 1979). Since it takes time for a technology to mature in an organization, case studies are in general also necessary in order to tailor it to the organization based on the feedback from developers and managers [Kitchenham *et al.,* 1995]. For example, at the SEL, only after the execution of some case studies did it become evident that Cleanroom produced benefits in quality and productivity (Basili and Green, 1994) only if a few modifications to the original method were implemented .

Second, qualitative studies should be seen as complementary to quantitative studies. In general, developers in an organization have many useful insights into existing problems and ways to address them; they are a valuable source of information in process improvement. Knowledge acquisition methods like structured interviews (Vogel, 1988) would help interpret quantitative data and provide ideas for solving problems.

In closing, one problem we have witnessed is the attempt by some organizations to try to solve their problems too rapidly. This is manifested in improvement programs with dozens of working groups and technologies being introduced concurrently. In practice, many organizations do not have the capacity to change very rapidly without considerable disruption to ongoing projects. Developers cannot continue to produce software at the previous rate, and change all their practices at the same time. There is no magic.

References

J. Armitage and M. Kellner (1994). „A conceptual schema for process definitions and models". In *Proc. of the 3rd Int'l Conf. on the S/W Process*, pp.153-165.

V. Basili (1992). „The experimental paradigm in software engineering". In D. Rombach, V. Basili and R. Selby (editors), *Experimental Software Engineering Issues: Critical Assessment and Future Directions*. LNCS, Vol 706. Spring-Verlag.

V. Basili and S. Green (1994). „Software process evolution at the SEL". In *IEEE Software*, July 1994.

V. Basili and H. D. Rombach (1988). „The TAME project: towards improvement-oriented software environments". In *IEEE Transactions on Software Engineering*, 14(6):758-773, June.

V. Basili; L. Briand; W. Melo (1995). „An Validation of Object-Oriented Design Metrics". University of Maryland, Dep. of Computer Science, College Park, MD 20742. CS-TR-3443.

M. J. Bassman; F. McGarry; R. Pajerski (1994). „Software Measurement Guidebook." Software Engineering Series, SEL-94-002.

N. Belkhatir and W. L. Melo (1994)." Evolving softwre processes by tailoring the behavior of software objects". In *Proc. of the IEEE Int'l Conf. on S/W Maintenance*, Victoria, Canada.

L. Briand; V. Basili; Y.M. Kim; D. R. Squier (1994). "A change analysis process to characterize software maintenance projects". Proc. of the *Int'l Conf. on S/W Maintenance*. Victoria, Canada.

L. Briand; W. L Melo; C. Seaman; V. Basili (1995). "Characterizing and assessing a large-scale software maintenance organization". In *Proc. of the 17th Int'l Conf. on S/W Eng.*, Seattle, WA.

L. Briand, S. Morasca, V. Basili (1994). *"Property-Based Software Engineering Measurement"*, CS-TR-3368, University of Maryland, College Park, MD, 20742, November 1994.

R. Chillarege, I. Bhandhari, J. Chaar, M. Halliday, D. Moebus, B. Ray, and M.Y. Wong. „Orthogonal defect classification - a concept for in-process measurement". *IEEE Transactions on Software Engineering*, 18(11):943-956, November.

F. Coallier (1995). „TRILLIUM: A model for the assessment of telecom product development and support capability". In *Software Process Newsletter*, No. 2, pp. 3-8, Winter.

J. Collofello and B. Gosalia (1993). „An application of causal analysis to the software production process". In *Software Practice and Experience*, 23(10):1095-1105, October.

T. Cook and D. Campbell (1979). *Quasi-experimentation: Design and Analysis Issues for Field Settings*, Rand McNally, Chicago.

J-N Drouin (1995). „The SPICE project: An overview". In *Software Process Newsletter*, No. 2, pp. 8-9, Winter.

K. El Emam and N. H. Madhavji (1995a). „A field study of requirements engineeirng practices in information systems development". In *Proc. of the 2nd IEEE Int'l Syposium on Requirements Engineering*, pages 68-80.

K. El Emam and N. H. Madhavji. (1995b). „Measuring the success of requirements engineering processes". In *Proc. of the 2nd IEEE Int'l Syposium on Requirements Engineering*, pages 204-211.

K. El Emam and N. H. Madhavji (1995c). „The reliability of measuring organizational maturity". In *Software Process Improvement and Practice Journal*, 1(1).

K. El Emam and D. R. Goldenson (1995). „SPICE: An empiricist's perspective". In *Proceedings of the Second IEEE International Software Engineering Standards Syposium*, August.

N. Fenton (1993). „Objectives and context of measurement/experimentation". In D. Rombach, V. Basili and R. Selby (editors), *Experimental Software Engineering Issues: Critical Assessment and Future Directions*. LNCS, Vol 706. Spring-Verlag.

N. Fenton; S.L. Pfleeger, R.L Glass (1994). „Science and substance: A challenge to software engineers". IEEE Software, July, pp. 86-95.

A. Finkelstein; J. Kramer; B. Nuseibeh, eds. (1994). *Software Process Modeling and Technology.* Research Studies Press (distributed by Wiley & Sons).

P. Fowler and S. Rifkin (1990). „Software Engineering process Group Guide". Technical Report CMU/SEI-90-TR-24, Software Engineering Institute.

D. Frailey (1991). „Defining a corporate-wide software process". In *Proc. of the 1st Int'l Conf. on the Software Process*, pp. 113-121.

D. Goldenson and J. Herbsleb (1995). „What happens after the appraisal? A survey of process improvement efforts". Paper presented at the *1995 SEPG Conf.*.

R. B. Grady (1992). *Practical Software Metrics for Project Management and Process Improvement*, Prentice-Hall, 1992.

V. Haase, R. Messnarz, G. Koch, H. Kugler, and P. Decrinis (1994). „Bootstrap: Fine tuning processs assessment". In *IEEE Software*, pp. 25-35, July.

T. Hall and N. Fenton (1994). „Implementing software metrics – the critical success factors". In *Software Quality Journal*, 3(4):195–208.

J. Henry and B. Blasewitz (1992). „Process definition: theory and reality". In *IEEE Software*, pp.105, November.

A. Hersh (1993). „Where's the return on process improvement?". In *IEEE Software*, page 12, July.

M. Kellner and G. Hansen (1989). „Software process modeling: a case study". In *Proc. of the 22nd Annual Hawaii Int'l Conf. on System Sciences*, vol. II, pp. 175-188.

B. Kitchenham, S. Linkman, and D. Law (1994). „Critical review of quantitative assessment". In *Software Engineering Journal*, 9(2):43-53, March.

B. Kitchenham; L. Pickard; S.L. Pfleeger (1995). „Case studies for method and tool evaluation". IEEE Software, 12(4):52–62.

D. Kitson and S. Masters (1993). „An analysis of SEI software process assessment results: 1987-1991". In *Proc. of the 15th Int'l Conf. on S/W Engineering*, pp. 68-77.

D. Leonard-Barton and W. Kraus (1985). „Implementing new technology". In *Harvard Business Review*, pp. 102-110, November-December.

N. Madhavji, D. Hoeltje, W.K. Hong, and T. Bruckhaus (1994). „Elicit: a method for eliciting process models". In *Proc. of the 3rd Int'l Conf. on the S/W Process*, pp.111-122.

W. Melo; L. Briand; V. Basili (1995). „Measuring the Impact of Software Reuse on Productivity and Quality in Object-Oriented Systems". University of Maryland, Dep. of Computer Science, College Park, MD, 20742. CS-TR-3395. [To appear in the Communications of the ACM]

C. McGowan and S. Bohner (1993). „Model based process assessments". In *Proc. of the Int'l Conf. on S/W Engineering*, pp. 202-211.

T. Nakajo and H. Kume (1991). „A case history analysis of software error cause-effect relationship". In *IEEE Transactions on Software Engineering*, 17(8), August.

M. Paulk, B. Curtis, M-B Chrissis, and C. Weber (1993). „Capability Maturity Model, version 1.1". In *IEEE Software*, pp. 18-27, July.

R. Radice, N. Roth, A. O'Hara, Jr., W. Ciarfella (1985). „A programming process architecture". In *IBM Systems Journal*, 24(2):79-90.

R. Schaffer and H. Thomson (1992). „Successful change programs begin with results". In *Harvard Business Review*, pp. 80-89, January-February.

Software Engineering Laboratory (1995). „*Software Process Improvement Guidebook*". Software Engineering Laboratory Series, April, SEL-95-002.

R. Veryard (1987). „Implementing a methodology". In *Information and Software Technology*, 29(9): 46-474, November.

C. Vogel (1988). „Génie cognitif", Masson, Paris.

S. Waligora, J. Bailey, and M. Stark (1995). „*Impact of ADA and Object-Oriented Design in the Flight Dynamics Division at Goddard Space Flight Center*". Software Engineering Laboratories Series, March, SEL-95-001.

E. Yu and J. Mylopolous (1994). „Understanding 'why' in software process modeling, analysis, and design". In *Proc. of the 16th Int'l Conf. on Software Engineering*, Italy.

The Evolution of a Quantitative Process Analysis - the BOOTSTRAP - Approach

Richard Messnarz
University of Technology
Graz, Austria

Johannes Scherzer
APAC GmbH
Vienna, Austria

Abstract/Introduction. The aim of the BOOTSTRAP project was to develop a method for software process assessment ([6], [7], [13]), quantitative measurement, and improvement [3]. BOOTSTRAP enhanced and refined the SEI method ([9], [10], [18], [19]) for software process assessment and adapted it to the needs of the European software industry including non-defense sectors such as banking, insurance, administration, etc. in order to be applicable to all kinds of Software Producing Units [4]. Within the BOOTSTRAP project an evaluation technique was developed which calculates capability profiles of software processes [7], [8]. These capability profiles serve as a quantitative basis for making decisions about process improvements. This paper presents some experience with the impact of process efficiency on product quality. In addition a comparison of the BOOTSTRAP evaluation and profiling approach with the SPICE ([21], [22], [23]) guidelines for calculating adequacy profiles of processes is made.

The paper is also going to discuss some results of a joint project with an ISO auditing organisation dealing with the question of how BOOTSTRAP could be adapted to serve as an ISO auditing methodology and technology [16],[17].

The goal of this paper is to discuss 8 years experience with measurement approaches used by process assessment and auditing methodologies and to analyse how the BOOTSTRAP approach satisfies the SPICE measurement guidelines and how the SPICE evaluation approach could be refined in the future.

1. The Impact of Process Efficiency on Product Quality Goals

This section illustrates how product quality goals can be met by improving process and cost efficiency. Fig. 1 shows a sample selection of product quality attributes [2], [4].

Fig. 1: Sample Selection of Product Quality Attributes

The product quality attribute *reliability*, for instance, can be measured by detection time (in which phase the problem is found) and by the defect rate during maintenance (comparison of size with the number of problems found). The correction of problems becomes more expensive the later the problem is found and solved.

Based on a figure presented and discussed at the ISCN'94 conference in Dublin the ratio between the correction of a design problem and a problem found in maintenance phase was about 1 to 6 - for a problem found in design it was about 2000 German Mark and for a problem found in maintenance it was above 10000 German Mark. Other international measures have been published stating that the maintenance costs of large systems are approximately 3-5 times the original development cost [4].
Therefore to achieve a more cost efficient process the management will have to introduce effective *quality management* (reviews and controlling) and *testing*. The main goal on the process side is to shift the problem distribution as shown in Fig. 2 and to obtain a more cost efficient process (problems are found earlier and corrected at lower costs).

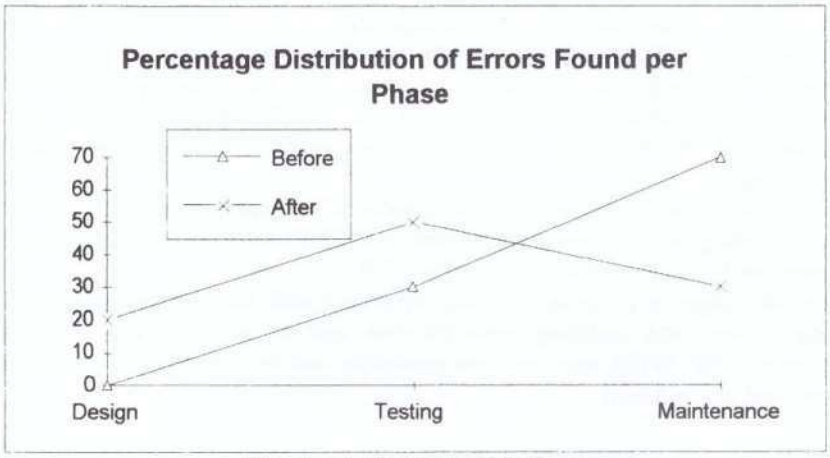


Fig. 2: Distribution of Problems

Fig. 2 shows a typical example (based on practical experiences) of an organisation which starts off between maturity level 1 and 2 and improves the attributes *quality management* and *testing* to reach maturity level 2.
We may therefore conclude that the effective use of quality management and testing methods leads to an improvement of cost efficiency on the process side, whereas on the product side the requirement of higher reliability will be met [12], [20].

The following example concentrates on the product quality attribute *maintainability* which is characterised by the sub-attributes *changeability* (mean effort for a change), *understandability* (mean effort to fix a problem), *modularity* (clear decomposition of a system into modules and clear description of interfaces), and *traceability* (version control).
In an organisation which started to develop software with a small team of hardware engineers. They directly implemented code and there was no problem as long as the system remained small. In the course of 6 years development the system became more and more complex, the maintenance effort increased considerably, and finally 70% of the personnel was stuck in maintenance. This was the main reason why the company ordered a BOOTSTRAP assessment. The BOOTSTRAP assessors found that there was no life cycle at all, the design was completely missing, and for project planning a technology was in place which no one used [7].
Fig. 3 shows the effect of the introduction of a professional CASE design methodology on process efficiency. Due to the clear decomposition of the system into modules the establishment of PERT plans is facilitated which allows to have parallel development of modules with clear integration guidelines - "a straight forward development. This

way an introduction of a design methodology can lead to higher productivity: according to a study published by Lempp 1988 [15] it is 13% less development effort, and according to a study performed in one of the BOOTSTRAP improvement projects productivity was increased by about 50% (which means about 30% less development effort). Yet, there is a significant influence on the mean effort for carrying out a maintenance activity because due to the clear description of modules and interfaces the time to fix problems (especially functional errors) decreases (e.g. by 67% according to Lempp).

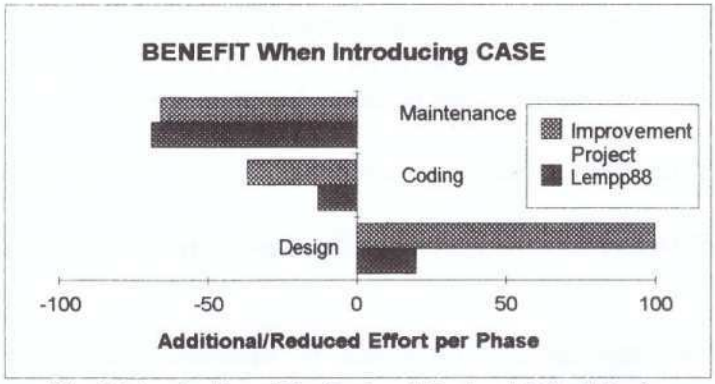


Fig. 3 Introduction of Professional Design Methodologies

Thus, if the maturity level for requirements analysis and design is improved from 1 to 2 (chaotic process on level 1, efficient methodology in place on level 2) the cost benefits shown in Fig. 3 will be obtained. Again it is visible that through cost efficiency on the process side you directly achieve product quality goals such as *changeability*, *understandability*, and *modularity*.
However, maturity level 2 means that the organisation has so far not standardised best practices across all projects (level 3 would be the defined stage). This usually leads to a situation that some projects use very efficient work flows, methods, and technologies but others do not. You will obtain the cost improvement shown in Figs. 2 and 3 when going from level 1 to 2, whereas on the way from level 2 to 3 you can multiply the benefits (e.g. achieve the cost benefits in all projects).

2. How to Measure Process Maturity

Section 1 points out that firstly the maturity of process attributes should be measured (because from this we can directly derive information about product quality) and secondly that this type of measurement needs a sufficient way of granularity. If, for instance, there is only one maturity measure for the entire process, how should we map design maturity onto maintenance efficiency, or testing maturity onto cost efficiency as is shown in Fig. 2 ? Therefore a maturity measure for each single attribute of a process is required [7].

2.1 Introduction to BOOTSTRAP's Evaluation Principles

2.1.1 SEI as the Starting Point

The SEI algorithm was strictly sequential and highly dependent on single questions. Only if the majority of key questions on level i are answered by YES (e.g. 11 of 12 key questions on maturity level 2) and if the total satisfaction percentage is above 80%

is level i satisfied and scores on the next highest level i+1 are taken into account. This evaluation approach has three statistical problems:

(1) The evaluation is highly dependent on single questions. If you answer ten instead of eleven key questions on maturity level 2 by YES one question decides that you fall back to maturity level 1.

(2) The evaluation approach does not take into account innovations. If either the key question rule or the total satisfaction rule fails, for instance, on maturity level 2 all scores on the next highest level will be disregarded. This does not award SPUs who plan innovation in certain organisational parts, therefore project management might be defined whereas quality management is still a chaotic process.

(3) The algorithm can only be applied to entire processes, SPUs or projects.

A major goal of BOOTSTRAP ([3], [7], [8]) was to develop an evaluation algorithm which

(1) is based on statistical considerations to minimise the dependence on single questions,

(2) takes into account innovations, and

(3) calculates a maturity level for each process attribute.

2.1.2 CMM´s Evaluation Approach

CMM (Capability Maturity Model) defines different key process areas for each maturity level. Each CMM key process area (key process area = attribute) is valid only within one maturity level. Thus it is not possible to map a single CMM attribute onto a maturity level scale from 1 to 5. This means that the whole improvement approach is maturity level and not attribute based because CMM´s structure supports the calculation of only one maturity measure for the entire process. The CMM approach evaluates each key process area by NS (Not Satisfied), PS (Partially Satisfied), and FS (Fully Satisfied). To satisfy a certain level all key process areas must be evaluated by FS.
The BOOTSTRAP methodology evaluates each attribute separately on a five point maturity scale. For each attribute the BOOTSTRAP assessment team checks a number of activities that have to be performed on levels 2 to 5. Thus BOOTSTRAP can concentrate on single attributes, evaluate each attribute on a five point maturity level scale, identify weaknesses and strengths, and establish attribute based improvement plans.

2.1.3 Step Based Maturity Level Scale

Each question represents a step. Each question is answered on a linguistic scale: complete (1 step), fair (0.66 steps), basic (0.33 steps), absent (0 steps). Three questions answered, for instance, by absent, basic, and complete represent 0 + 0.33 + 1 = 1.33 steps. Each question is assigned to a maturity level, and based on this

assignment a number of steps per maturity level is calculated. The total number of steps represents all steps achieved on all levels.

This very simple calculation rule is based on the following statistical considerations. Steps imply that there are different distances between the maturity levels. The distance d2 between maturity level 1 and 2 is defined by the number of steps (applicable questions) on level 2. The distance d3 between maturity level 2 and 3 is defined by the number of steps (applicable questions) on level 3. etc. This is very different from a scale where only satisfaction percentages per level are calculated leading to equal distances between the levels. In the SEI as well as in the BOOTSTRAP the number of questions per level differs considerably. In the BOOTSTRAP questionnaire version 2.22 there are, for instance, 28 questions on level 2, 54 on level 3, 19 on level 4, and only 5 on level 5. Fig. 4 shows a sample calculation done with the step based scale and with the percentage based scale.
Steps also imply that satisfaction percentages are weighted by statistical significance. If you calculate pc2 (average satisfaction percentage on level 2) and d2 (number of applicable questions on level 2) and multiply pc2*d2 you obtain the number of the steps achieved on level 2 (the simple adding procedure described at the beginning of this section 2.1.3). This means that steps are obtained by multiplying the satisfaction percentage with the number of questions on which the calculation is based. This means that 50% on level 4 (d4 = 19 in Fig. 4) represent a distance of 19*0.5 on the step scale, whereas 50% on level 3 (d3 = 54) represent a distance of 54*0.5 on the step scale which is about three times more significant.

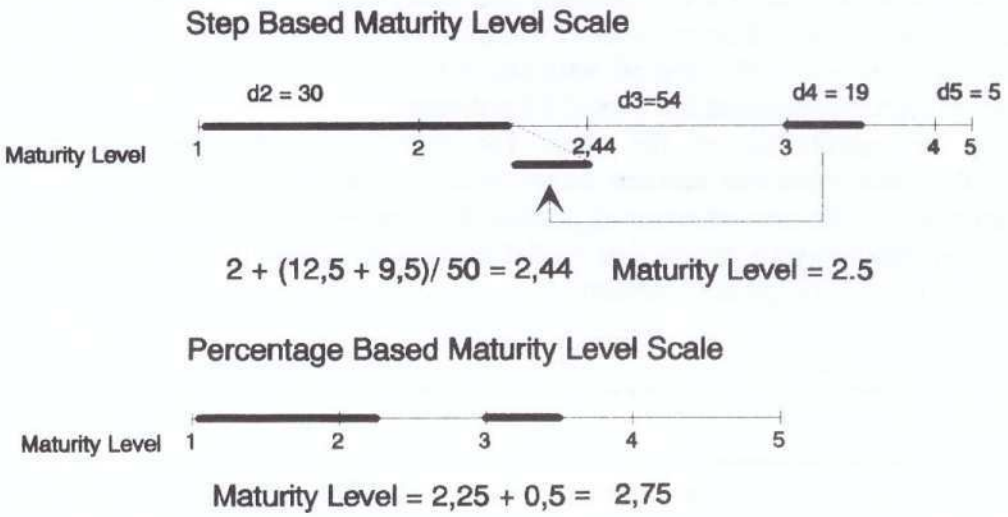


Fig. 4: Maturity Level Scale Based on Steps

Fig. 4 presents an example where level 2 is fully satisfied, level 3 is satisfied by 25% and level 4 by 50%. Using the percentage based scale a maturity level of 2,25 + 0,5 = 2,75 is obtained. If the statistical significance is taken into account 50% on level 4 represent 9,5 steps on the scale, which means a maturity level of 2,25 + (9,5 / 54) = 2,44 ≅ 2,5. As long as the questions are not equally distributed (the same number of questions per level for each attribute) a step based maturity scale is statistically more reliable.

2.1.4 How to Refine the Number of Total Steps ?

Key Attributes Instead of Key Questions. When calculating the maturity level of the entire SPU / project the BOOTSTRAP methodology uses an approach that avoids the one path of improvement presented in the CMM. In CMM each key process area must be evaluated by FS (Fully Satisfied) to fulfil a certain level. BOOTSTRAP does not use single questions like SEI'87 or all attributes like the CMM approach but selected clusters of questions (*key attributes*). Quality management, for example, is a key attribute consisting of 10 questions which have to be satisfied by a certain threshold percentage to fulfil a certain level. A threshold level T_{max} is calculated by the lowest maturity level and by which one of the key attributes is satisfied by less than 50%. For the SPU/PRJ level only the steps achieved on levels 2 to T_{max} are taken into account. The idea behind this rule is to ensure that the maturity level of the organisational unit depends on the lowest key attribute levels in the profile [7], [8].

Innovation Rule. The SEI'87 as well as the CMM evaluation approach are strictly sequential. The SEI'87 evaluation approach presumes that only when level i is satisfied by a minimum of about 80 percent and if the majority of key questions on level i are answered by yes the scores on level I+1 are taken into account. The same applies for the CMM approach where all key process areas must be evaluated by FS. This does not award SPUs and projects which plan and stagger innovation over a period of time. If an SPU is below level 2, why shouldn't we take into account scores on level 3 when looking at a particular attribute. For example, suppose an organisation already has an efficient design methodology and standardised way of creating design documents, but has no efficient project management method. With the SEI/CMM method it would rate a level 1 overall, with the BOOTSTRAP method it would rate a level 1 for project management but a level 3 for design.
Thus for the calculation of the steps for each individual process attribute BOOTSTRAP also takes into account scores which the SPU or project gained on the next highest level. This way of counting applies to all individual attributes of an SPU / project. It is mandatory to ensure that no flat profiles but characteristic profiles with high and low values [7], [8] are obtained.

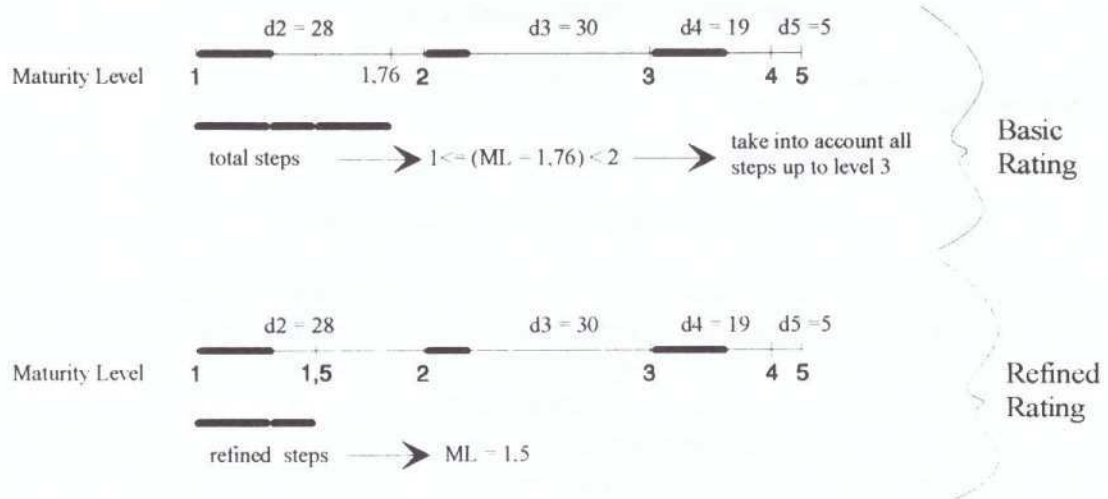


Fig. 5 : Using the Innovation Rule

In Fig. 5 the steps on levels 2 and 3 are only partially satisfied, whereas on level 4 50% of the practices are fulfilled.

This way of counting ensures

(i) that innovations on level 3 - the next highest level - are taken into account, and

(ii) avoids that extreme cases such as the satisfaction of level 4 are taken into consideration (too many basic practices are missing).

Satisfaction Rule. If more than 80% of the steps are satisfied on a maturity level i , level i is fully satisfied. This rule was adopted from SEI. It is based on the consideration that questions/attributes are dependent on each other and thus with the coverage x percent aspects of the remaining 100-x percent are automatically covered. However, it is technically more sound to use exponential distribution functions for the simulation of this situation. A new version of this algorithm has recently been implemented in a prototype and is currently being tested [7], [8].

3. Comparison of BOOTSTRAP with SPICE

3.1 Comparison of BOOTSTRAP´s Architecture With SPICE

The following analysis is based on the Baseline Practices Guide [21], the Software Process Assessment Guide [22], and the Model for Process Measurement [23] of the SPICE project. I will briefly compare the approaches and illustrate that BOOTSTRAP´s evaluation technique is largely compliant with the SPICE requirements. In addition I will make some suggestions as to how to refine the SPICE evaluation approach.

The Baseline Practices Guide describes two different architectures: a process based architecture, and a capability level based architecture. The process based architecture defines *process categories* (customer supplier process category, engineering process category , project process category, etc.). Each process category consists of 5 to 7 *processes*, each of which contains a number of *base practices*. This is similar to the BOOTSTRAP process architecture, in which the combined attributes (organisation, life cycle functions, etc.) represent the process categories, the elementary attributes (project management, quality management, requirements analysis, etc.) represent the processes, and the questions are related to the base practices. Fig. 6 shows BOOTSTRAP´s process architecture of questionnaire version 2.3, which is not SPICE compliant. However, it shows some specific relations to the process based architecture in SPICE. The *project process category* corresponds to BOOTSTRAP´s life cycle independent functions, the *engineering category* with the life cycle functions, the *support category* with the process related functions, and the *organisation category* with organisation. BOOTSTRAP lacks most processes described in the *customer supplier category* such as provide customer service, assess customer satisfaction, etc..

SPICE´s capability level based architecture defines *capability levels*, *common features*, and *generic practices*. The BPG (Baseline Practices Guide) defines capability levels from 0 to 5 (CMM has levels 2 to 5): Not Performed (0), Performed Informally (1), Planned and Tracked (2), Well Defined (3), Quantitatively Controlled (4), Continuously Improved (5). The capability based architecture is similar to SEI´s CMM architecture. The common features are similar to the key process areas in CMM. Like in CMM where a key process area is only valid within one maturity level, a common

feature in SPICE is valid only within one capability level. And the key practices of CMM correspond to the generic practices in the BPG.

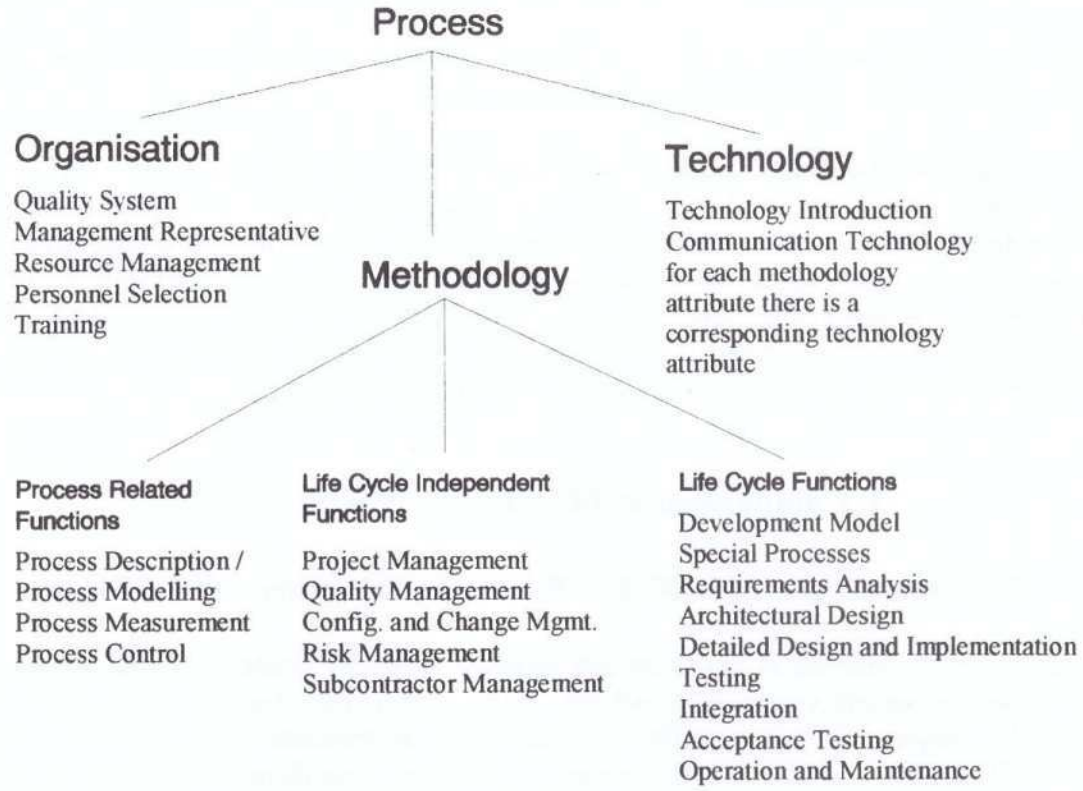


Fig. 6: BOOTSTRAP's Process Quality Attribute Hierarchy [8], [13]

The BPG contains a guideline for assigning processes to common features and generic practices so that a capability level is assigned to each process. This way the BPG is a useful guideline to enable engineers to set up a comprehensive checklist (covering all base practices) and for assigning capability levels to processes. This is similar to CMM's overall architecture because each process is only valid within one capability level (assigned to one common feature) so that again the processes cannot be evaluated on a maturity scale from 0 to 5 and only the satisfaction of common features is evaluated based on process adequacy ratings - which is a detailed representation of satisfaction percentages (see 3.2.1).

In SPICE as well as in BOOTSTRAP each question is answered on a 4 point linguistic scale: F (Fully Adequate), L (Largely Adequate), P (Partially Adequate), and N (Not Adequate). In addition each question can be answered by N/A (Not Applicable). In SPICE like in BOOTSTRAP a profile - in BOOTSTRAP a maturity profile, and in SPICE a process adequacy profile - is calculated. Fig. 7 shows typical BOOTSTRAP maturity profiles for all attributes of an SPU.

Process maturity profiles like the one in Fig. 7 can easily be interpreted. A maturity level ([3], [10], [18], [19]) of 2 means that an effective method is in place, level 3 means that the most effective methods are documented and standardised across all projects, level 4 means that the efficiency of these methods, the productivity of major process steps and the quality of the products are quantitatively measured, and level 5 means that the quantitative feedback is analysed and action plans to improve the process are established. All quarters (e.g. 1.75, 2.25, 3.5) can be interpreted in linguistic terms (e.g. The method employed is weak at 1.25, basic at 1.5, significant at

1.75, and effectively employed at 2; The method is effectively used but weakly documented and standardised at 2.25, basically documented and standardised at 2.5, etc.) [7].

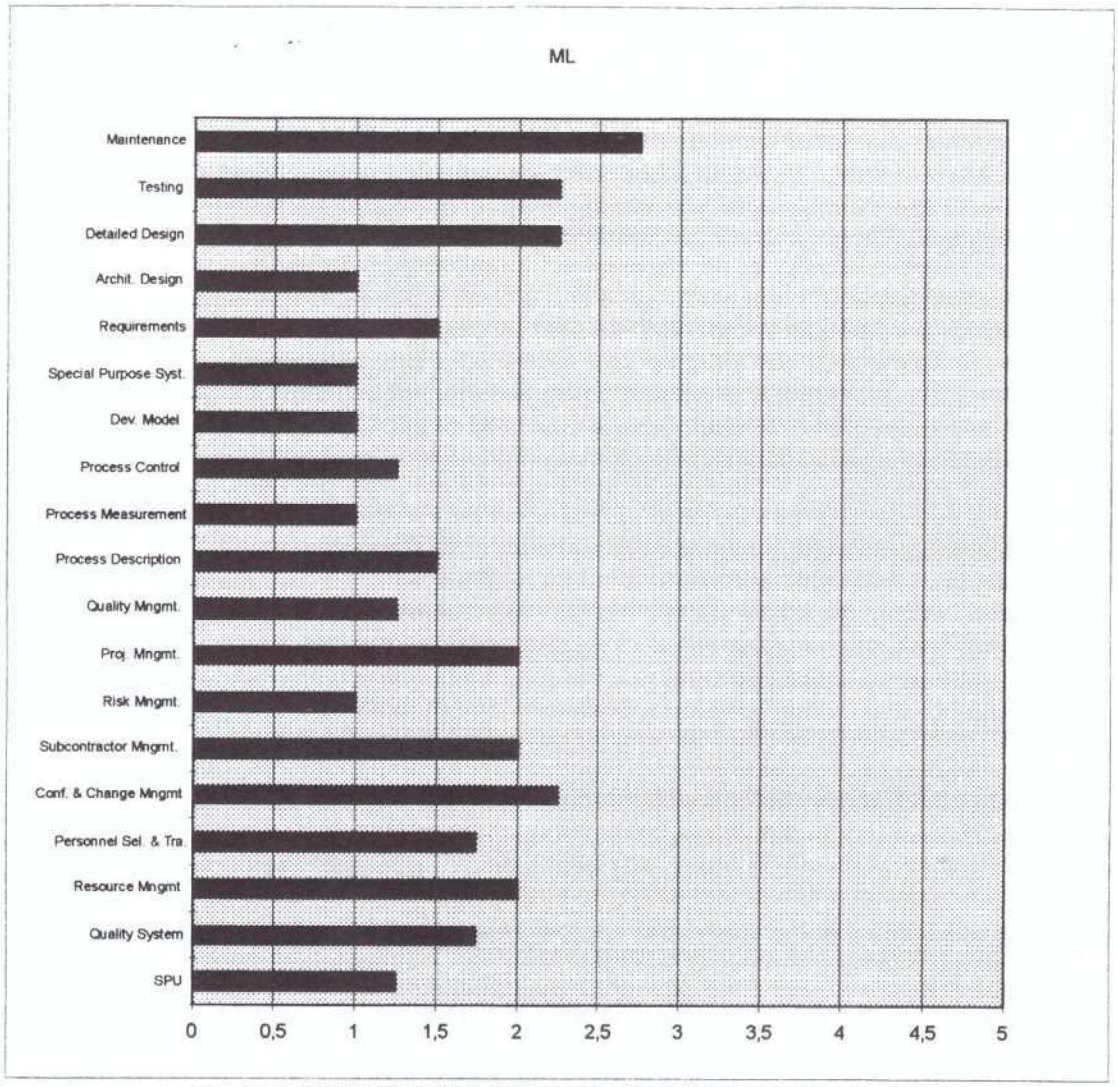


Fig. 7 : BOOTSTRAP Maturity Profile of a Real Case Organisation

3.2 The SPICE Compliant BOOTSTRAP Evaluation

3.2.1 The SPICE Scoring and Rating

The Model for Process Measurement as part of the Process Assessment Guide describes the evaluation procedure of SPICE. SPICE calculates an adequacy rate for each process, common feature, and per capability level. An adequacy rate is described by a three-dimensional vector (%F,%L,%P) for example (30,40,10), which means that in 30% of the cases Fully Adequate was selected, 40% Largely Adequate was selected, 10% Partially Adequate was selected, and 20% {100 - (30+40+10) } Not Adequate. In this way SPICE calculates an adequacy vector for each process :

Adequacy Vector of Process =

$\{ (\%F,\%L,\%P)_1, (\%F,\%L,\%P)_2, (\%F,\%L,\%P)_3, (\%F,\%L,\%P)_4, (\%F,\%L,\%P)_5 \}$

with $(\%F,\%L,\%P)_i$ representing the adequacy rate for capability level i for i =1.. 5

3.2.2 BOOTSTRAP's Attribute Evaluation Vector

Section 2.1.3 pointed out that steps can be calculated by simply multiplying the average satisfaction percentage by the number of questions on which the satisfaction calculation is based. Therefore a vector

(pc2, d2, pc3, d3, pc4, d4, pc5, d5)

for each attribute / process is needed to be able to calculate the number of steps. In addition this vector must include a minimum level (mil) and a maximum level (mal). The attribute process measurement, for instance, only contains level 3 to 5 questions but no level 2 questions. Thus the minimum level for process measurement is 2. Requirements management, for instance, only contains level 2 questions so that the minimum is 1 and the maximum 2. The calculated maturity levels can only be interpreted within the ranges of mil and mal. A maturity level of 2 for resource management means a strength because the maximum level is achieved. A maturity level of 2 for quality management, however, leaves a high improvement potential because the maximum to be achieved is 5.

Attribute Evaluation Vector =

{ (pc2, d2, pc3, d3, pc4, d4, pc5, d5), mil, mal, Maturity Level)

3.2.3 Relationships Between BOOTSTRAP's and SPICE's Evaluation Approach

Fig. 8 shows the relation between the SPICE adequacy vector and BOOTSTRAP's attribute evaluation vector.

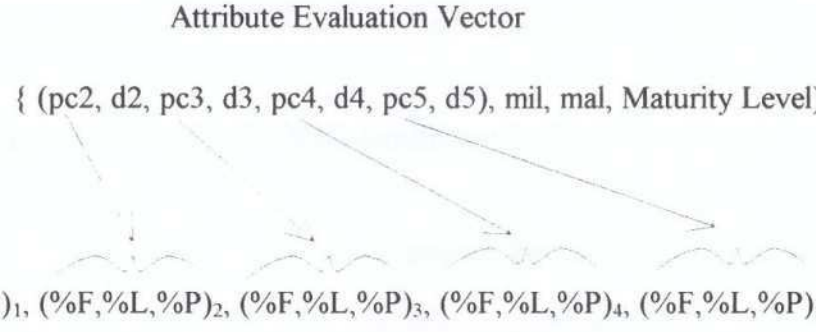


Fig. 8: Relations Between the SPICE and the BOOTSTRAP Evaluation Vector

The SPICE adequacy vector refines the satisfaction percentage values in the BOOTSTRAP vector - e.g. pc2 is refined by the adequacy rate $(\%F,\%L,\%P)_2$. However, the SPICE vector does not take into account the statistical significance of the adequacy ratings as it is done in the BOOTSTRAP vector through the values d2,

d3, d4, d5. The adequacy rate (100,0,0) could be the answer to one question or the answer to 20 questions, the adequacy vectors would not tell you. Moreover the five adequacy ratings are not combined into one capability level as it is done by the BOOTSTRAP maturity level algorithm. In addition the SPICE architecture is very much the same as the CMM architecture - in CMM each key process area was evaluated by NS, PS, or FS - because SPICE assigns processes to common features and evaluates the adequacy rating of each common feature so that again the mapping of processes onto a scale from 0 to 5 is missing. And finally SPICE does not define ranges (like mil and mal) within which the values are to be interpreted.

3. Use of BOOTSTRAP as an ISO Auditing Technique

In October 1994 one of the BOOTSTRAP partners began to co-operate with a consulting company that performs ISO audits for the TÜV in southern Germany [16], [17]. The project started off by comparing the TÜV checklist with the BOOTSTRAP questionnaire version 2.22. The work was based on a comparison of BOOTSTRAP and ISO published in the ISCN'94 proceedings 18]. The goal of this project was to evaluate if the calculation of maturity profiles can be employed as an ISO auditing technique.
Questions in the TÜV questionnaire are answered on a three point scale with Fully Adequate, Partially Adequate, and Not Adequate. For each ISO 9001 attribute there is a set of checkpoints like:

4.8 Product Identification and Traceability

1 Existence of procedures and responsibilities for planning and performing the identification of product components.

2 Are all phases of the product life cycle taken into account in the configuration planning including maintenance.

3 Existence of an effective system for configuration management comprising
version management, change documentation
configuration identification
identification of all related product components
identification of all related documents

For each of these questions the ISO auditors evaluate the quality system documentation as well as the implementation of the practices described in the quality manual.
BOOTSTRAP questions were assigned to ISO attributes, BOOTSTRAP questions which TÜV does not cover, and TÜV checkpoints which BOOTSTRAP does not cover were identified. Based on these findings a combined questionnaire was developed which is based on the 20 ISO 9001 attributes and on BOOTSTRAP's architecture (each checkpoint was assigned to a maturity level). There are three types of questions:

type = *o* ... original BOOTSTRAP question
type = *a* ... adapted BOOTSTRAP question
type = *i* ... ISO question added (was missing in BOOTSTRAP)

Type *a* was used if one and the same question was applicable to different ISO attributes. Typical questions in the combined questionnaire are:

Attribute:	4.3 Contract Review
Number:	9001
Level:	2
Text:	Are joint reviews performed with the customer to ensure that all requirements are agreed and signed by all partners involved
Answers:	
Fully Adequate/Largely Adequate/Partially Adequate/Not Adequate/Not Applicable	
Comments:	*type* = *i* | 4.3&p1&3,5 and 4.3&p2&2,3,6 : check of offer, order, contract; if contract changes occur

Attribute:	4.10.1 Receiving Insp. & Test
Number:	1007
Level:	2
Text:	Existence of a formal procedure for selecting and purchasing externally developed HW/SW.
Fully Adequate/Largely Adequate/Partially Adequate/Not Adequate/Not Applicable	
Comments:	*type* = *a* | 4.10.1&1 : responsibility, selecting and testing of product, (quality requirements)

The comments describe the type of the question and contain a reference to checkpoints in the TÜV questionnaire. When there was no relation to any ISO 9001 checkpoint we marked the BOOTSTRAP question with "(+)". We finally obtained an SPU questionnaire comprising 190 questions, 127 being of type o, 34 of type a, and 29 of type i.
Based on the assignment of maturity levels to questions the BOOTSTRAP maturity level algorithm was used to evaluate all type o questions and to calculate maturity values for BOOTSTRAP attributes (see Fig. 6) as well as to evaluate all questions applicable to ISO 9001 audits (not marked as plus) and to calculate maturity values for the 20 ISO 9001 attributes. This way the same questionnaire could be used and either BOOTSTRAP profiles (Fig. 7) or ISO certification profiles (Fig. 9) could be printed.
It is visible that there is a relationship between Figs. 7 and 9. The attributes *4.4.9 design changes* and *4.5.3 document changes* have the same maturity as the BOOTSTRAP attribute *configuration and change management*. In addition *maintenance* and *4.19 servicing* have the same value, and there is a strong relation between testing and *4.10 testing*. Both profiles are based on an ISO audit of a middle sized Austrian software company for which the ISO profile was calculated. The profiling technique in Fig. 9 proved to be beneficial to ISO audits.

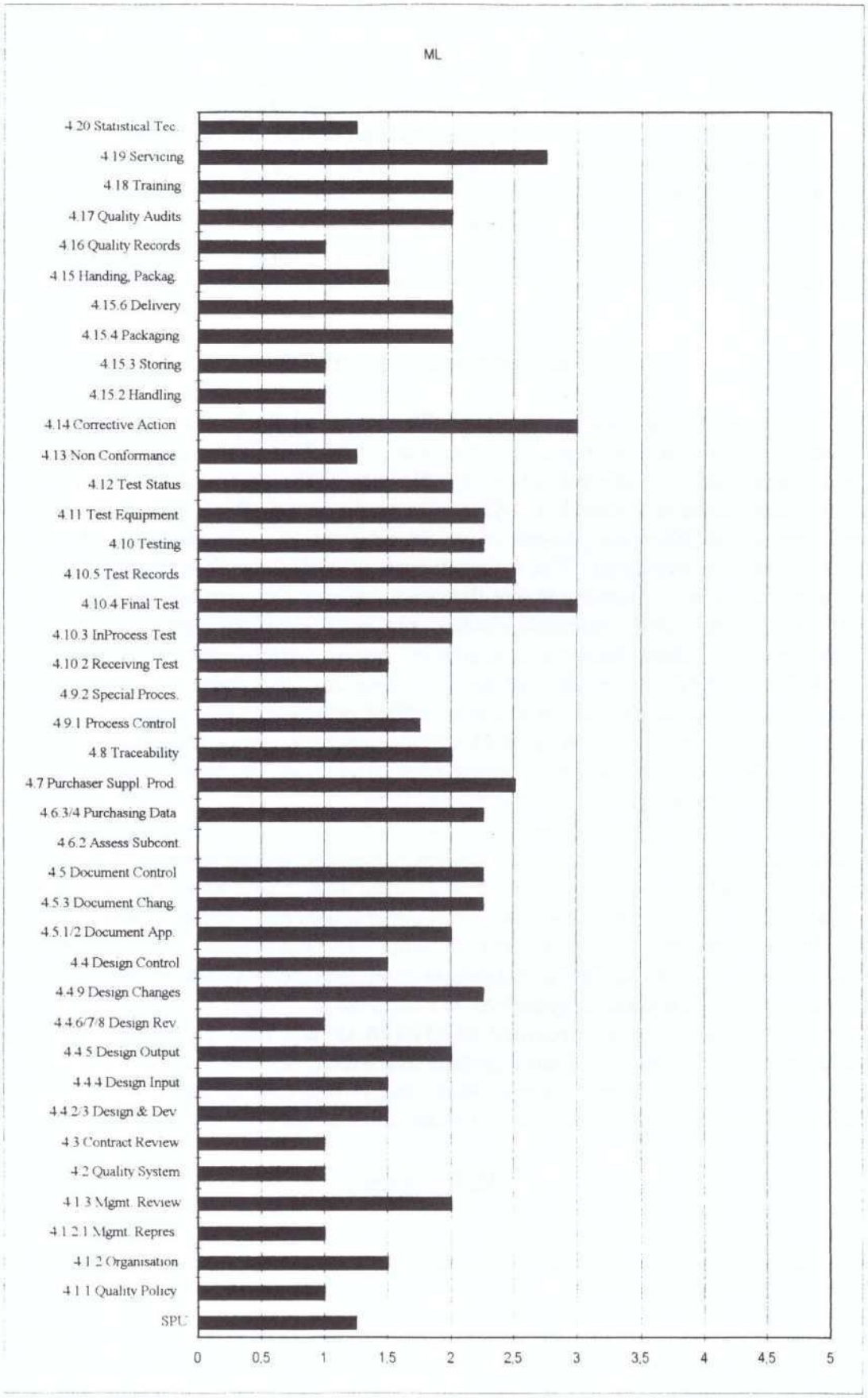


Fig. 9: ISO Audit Profile for the Organisation in Fig. 7

However, the BOOTSTRAP assessment process differs from the ISO audit process. BOOTSTRAP selects an SPU and two or three projects and speaks with the managers involved. An ISO audit first evaluates the quality manual and secondly checks if all people in the organisation follow the defined procedures. They do not restrict the audit to only one SPU and a selected set of projects but evaluate the organisation across all its SPUs. Thus the work of an ISO auditor cannot be restricted to a discussion game in one assessment room but he/she has to look at different SPUs, and projects moving to different rooms. This mobility requires a different set of tools to be used. An ISO audit tool therefore should be usable on a pocket organiser making the auditors independent of certain locations. This kind of tool was developed in our joint co-operation.

5. Conclusion and Future Aspects

It is not enough to only have a maturity profile measuring attributes of an organisation. If project management is improved from level 1 to 2, it is necessary to know the specific goals and how this will affect the efficiency of the organisation [5]? You can, for instance, measure a value like |*effort estimated - actual effort*|, and if this measure decreases in the follow-up projects using the new project management methodology the efficiency has improved. The main criticism of quality profiles (maturity profiles, score profiles, etc.) in industry is that they only measure the process on the top level but do not base their recommendations on more detailed metrics such as the measurement of effort, #defects, productivity, etc. In future it will be very important for BOOTSTRAP to calculate maturity profiles and also take into account more detailed metrics to be able to verify the benefit of process improvement in measurable terms. A quality profile is useful in identifying where to start process improvement. Effort, defect, and productivity metrics help to analyse the benefit of the process improvement activities that have been carried out.

In the last two years we developed a prototype QUES (Quantitative Quality Evaluation System) which represents a database system storing general SPU data, the calculated capability profiles, and product specific data. This system makes it possible to statistically analyse the relations between general data and capability profiles as well as between capability profiles and product data. The QUES forms the basis for future enhancement of the BOOTSTRAP database to an evaluation system enabling relations between process and product quality to be calculated.

As far as product data is concerned BOOTSTRAP will have to develop a product questionnaire to be filled in at each process assessment. It took us over three years to collect general and process maturity data, and it will take a further three years to collect all the product data necessary to make reliable quantitative evaluations.

References

[1] Biro M., Kugler H.J., Messnarz R., et. al., BOOTSTRAP and ISCN - A Current Look at a European Software Quality Network, in: *The Challenge of Networking, Proceedings of the CON'93 Conference*, Oldenbourg, 1993

[2] Boegh J., SCOPE - A Guide for Software Product Quality Evaluation, in: *Proceedings of the ISCN'94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd., Dublin, Ireland, 1994

[3] Cachia R.M., M. Maiocchi, *Middle Management Briefing*, Deliverable 10 and 20, BOOTSTRAP ESPRIT 5441, Commission of European Communities, 1992

[4] Chroust G., Modelle der Software Entwicklung, Oldenbourg, Munich, 1992

[5] Debou C., ami - A New Paradigm for Software Process Improvement, in: *Proceedings of the ISCN'94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd. , Dublin, Ireland, 1994

[6] Galimberti R., BOOTSTRAP Institute - Services and First Results, in: *Proceedings of the ISCN'94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd., Dublin, Ireland, 1994

[7] Haase V., Messnarz R., Koch G., Kugler H.J., Decrinis P., BOOTSTRAP: Fine Tuning Process Assessment, *IEEE Software*, pp. 25-35, July 1994

[8] Haase V., Messnarz R., Cachia R.M., Software Process Improvement by Measurement, in (ed.) Mittermeir R., *Shifting Paradigms in Software Engineering*, pp. 32 - 41, Springer Verlag, Wien, New York, Sept. 1992

[9] Humphrey W.S., Sweet W.L., *A Method for Assessing the Software Engineering Capability of Contractors*, Software Engineering Institute, CMU, Technical Report CMU/SEI-87-TR-23, 1987

[10] Humphrey W.S., *Managing the Software Process*, (ed.) Software Engineering Institute (USA), Addison-Wesley Publishing Company, New York, Wokingham, Amsterdam, Bonn, Madrid, Tokyo, 1989

[11] Jack R., Applying ISO 9001 to Small Scale Software Development Projects, in: *Proceedings of the Software Engineering Standards Symposium 1993*, pp. 11-18, IEEE Computer Society Press, Los Alamitos, USA 1993

[12] Kelly M., METKIT - Approaches, Experiences and Results, in: *Proceedings of the ISCN'94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd., Dublin, Ireland, 1994

[13] Kuvaja P., Simila J., Krzanik L., Bicego A., Koch G., Saukkonen S., Software Process Assessment and Improvement, The BOOTSTRAP Approach, Blackwell Business, Oxford, UK, 1994

[14] Koch G., The ESI Approach - European Strategies for Software Process and Product Improvement, in: *Proceedings of the ISCN'94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd., Dublin, Ireland, 1994

[15] Lempp P. Lauber R., What Productivity Increases to Expect from a Case Environment, Results of a User Survey, Chikofsky E.J. (ed), Computer Added Software Engineering, pp. 105-119, IEEE Computer Society Press Technology Series, 1989

[16] Messnarz R., Kugler H.J., BOOTSTRAP and ISO 9000: From the Software Process to Software Quality, in: *Proceedings of the APSEC'94 Conference*, Comput. Soc. Press of the IEEE, Tokyo, Japan 1994

[17] Messnarz R., Kugler H.J., BOOTSTRAP and ISO 9000 - A Quantitative Approach to Objective Quality Management, in: *Proceedings of the ISCN'94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd., Dublin, Ireland, 1994

[18] Paulk M.C., Curtis B., Chrissis M.B., *Capability Maturity Model for Software*, (ed.) Software Engineering Institute (USA), Technical Report CMU/SEI-91-TR-24, Carnegie Mellon University, Pittsburgh 1991

[19] Paulk M.C., Weber C.V., Garcia S.M., Chrissis M.B., Bush M., *Key Practices of the Capability Maturity Model*, Version 1.1, Technical Report CMU/SEI-93-TR-25, Software Engineering Institute, Pittsburgh 1993

[20] PYRAMID Consortium, ESPRIT Project 5425, *Quantitative Management: Get a Grip on Software!*, December 1991

[21] SPICE, *Baseline Practices Guide*, ISO/IEC JTC1/SC7/WG10, Version 1.00, Sept. 1994

[22] SPICE, *Process Assessment Guide*, ISO/IEC JTC1/SC7/WG10, Version 1.00, June 1994

[23] SPICE, *Model for Process Measurement*, Process Assessment Guide, Version 1.02, March 1994

SPICE: THE SOFTWARE PROCESS ASSESSMENT MODEL

Pasi Kuvaja
University of Oulu, Oulu, FINLAND
Adriana Bicego
Etnoteam S.p.A., Milan, ITALY
and
Alec Dorling
European Software Institute, Bilbao, SPAIN

Abstract: *SPICE (Software Process Improvement and Capability dEtermination) is an international collaborative project to develop a Standard for Software Process Assessment under the auspices of the International Committee on Software Engineering standard, ISO/IEC JTCI/SC7/WGlO. This paper provides an overview of the project and its results. More specifically are presented the SPICE process model (the model for process management) and the assessment framework, that form the basis of software process assessment and improvement. The SPICE process model contains processes comprising process categories and relating to capability levels that define common features to process performance specific to each capability level. The assessment framework is used during the process assessment activities to evaluate the implemented practices and generate a capability profile for each process assessed.*

Keywords: software process assessment, capability determination, software process, software engineering standard

1. Introduction

Process assessment and improvement methodologies came into the highlight of discussion throughout the software industry after the Software Engineering Institute (SEI) published method for assessing the software engineering capability of contractors in 1987 ([1]). After that an ESPRIT project BOOTSTRAP developed a European process assessment and improvement methodology, where the original SEI assessment method was applied as one of the main background approaches ([2]). In Canada a CMM-based ([3], [4] assessment methodology called Trillium ([5]) was developed for telecom-specific software process. Additionally, some in-house assessment approaches as Healthcheck ([6]) and some national approaches as STD ([7]) were created. At the same time, especially in Europe, industrial demand for ISO-9000 based software quality system certification pushed companies to improve their processes to pass certification.

As a continuum of these approaches the ISO/IEC[1] JTC1/SC7[2] established a working group[3] to develop a set of standards on SoftwareProcess assessment that attempts to harmonise above mentioned approaches. The decision was made based on a report ([8]) of a preliminary study that indicated that:

---

[1] International Organisation for Standardisation/Electrotechnical Commission

[2] Joint Technical Committee 1/Subcommittee 7 (Software Engineering)

[3] Working Group 10 (Software Process Assessment)

- there was international consensus on the need and requirements for a standard for process assessment;
- there was international consensus on the need for a rapid route to develop and trial the standard in order to provide usable output in an acceptable time scale and to ensure that the standard will fully meet the user needs; and
- there was international commitment to provide resources for a project team co-ordinated through four technical development centres based in Europe, North America (Canada and USA) and Pacific Rim.

To speed up the work, WG10 decided to carry out the development stage of the new standard through an international project called SPICE (Software Process Improvement and Capability dEtermination) and established a *modus operandi* to define the relationship between SPICE and WGlO ([9]). The SPICE project was kicked-off at the beginning of 1993 in Dublin and is currently supported by organisations from 20 different countries ([10]).

This paper presents the purpose of the process assessment standard, its overall architecture and the fundamental process model. The emphasis is on the SPICE process model that defines at the high level, the goals and base activities that are essential to good software engineering. The model categorises processes into five process categories that contain in total 35 processes.

2. The purpose and requirements of the process assessment

Software process assessment has its origin in the TQM movement, and derives from the basic premise that the quality of manufactured products is largely determined by the quality of the processes which produce them ([11]). According to Humphrey ([12]) the first step of software process improvement is to understand the current status of the process. *Software process assessment* provides means to this, and is understood in the SPICE project ([8]) as:

> *"The disciplined examination of the processes used by an organisation against a set of criteria to determine the capability of those processes to perform within quality, cost and schedule goals. The aim is to characterise current practice, identifying strengths and weaknesses and the ability of the process to control or avoid significant causes of poor quality, cost and schedule performance."*

The purpose of the SPICE project is to develop a basis for a Process Assessment Standard that allows examination of software development processes of an organisational unit in order to:

- characterise current practices of the processes by identifying their strengths, weaknesses and risks,
- determine the extend to which the current practices are effective in achieving process goals, and
- determine the extend to which current practices conform to a set of baseline practices used as a reference framework in the assessment.

The Process Assessment Standard should be applicable for use both for *software process improvement* and *software process capability determination* as the following figure (Figure 1) indicates. In the process improvement an assessment is used typically by a software supplier organisation for determining current status of its processes in order to improve them or to evaluate their suitability for a particular set of requirements. In capability determination process assessment is typically used by a software acquirer for determining the suitability of its potential supplier's processes for a particular contract or specific requirements.

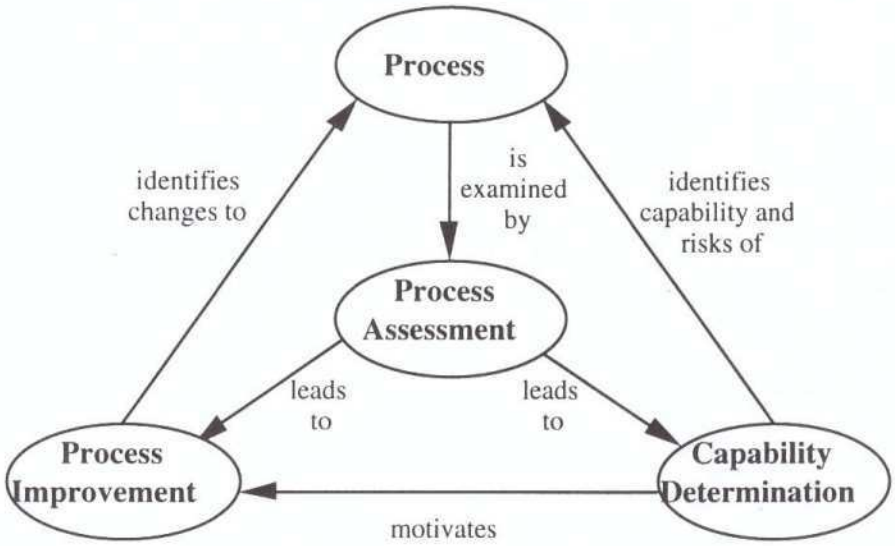


Figure 1. Software Process Assessment

Within a process improvement context, process assessment provides the means of characterising the current practice within an organisational unit[4] in terms of the capability of the selected processes. Analysis of the results in the light of the organisation's business needs identifies strengths, weaknesses and risks inherent in the processes. This, in turn, leads to the ability to determine whether the processes are effective in achieving their goals, and to identify significant causes of poor quality, or over runs in time or cost. These provide the drivers for prioritising improvements to processes.

Process capability determination is concerned with analysing the proposed capability of selected processes against a target process capability profile in order to identify the risks involved in undertaking a project using the selected processes. The proposed capability may be based on the results of relevant previous process assessments, or may be based on an assessment carried out for the purpose of establishing the proposed capability.

3. Overview of the SPICE products

The results of the SPICE project (comprising the process assessment standard) provide requirements and guidance on how to conduct a software process assessment using a model of processes and practices essential to good software engineering. They also provide guidance on software process improvement, software process capability determination, assessors training and qualification and use of assessment instruments. The SPICE documents were developed in seven product teams whose work items had functional dependencies as illustrated in the following figure (Figure 2).

Finally, when the SPICE products were consolidated to have common structure and format, the results of the Introductory Guide and Process Assessment Guide teams were each divided into two separate documents, and the whole document suite was renamed. Therefore, the final output from the SPICE project contains nine parts as described in Appendix 1. The core documents that form the final Process Assessment Standard are parts 1, 2 and 3 (see Appendix 1). The remaining parts are predominantly recommended guidelines on how to perform an assessment and how to use the assessment results.

---

4 That part of an organization that is the subject of an assessment. An organizational unit is typically part of a larger organization, for example a specific project or set of (related) projects, a unit within an organization focused on a specific lifecycle phase (or phases such as acquisition, development, maintenance or support), or a part of an organization responsible for all aspects of a particular product or product set.

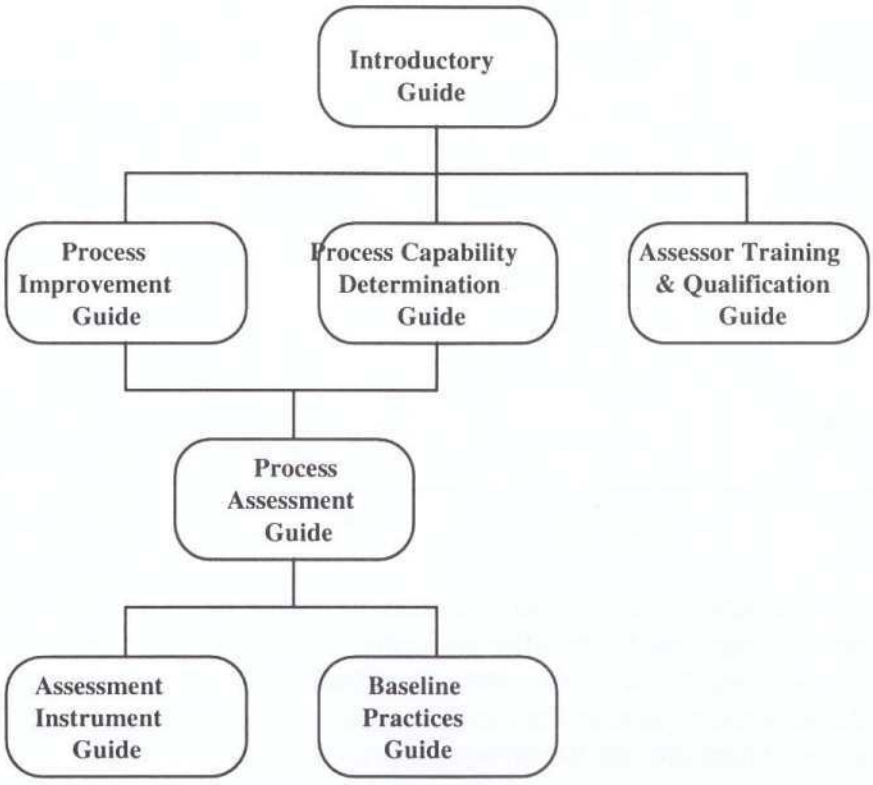


Figure 2. The SPICE products and their functional relationships

4. Process assessment context

Process assessment is generally performed either during a process improvement initiative, or as a part of a capability determination exercise. As such it is invoked by and returns results to either the improvement or the capability determination ([11]). In either case there should be an input to the process assessment defining the purpose (why the assessment is being carried out), scope (what processes should be assessed) and what constraints, if any, will apply. The assessment input also defines the responsibility for carrying out the assessment and gives definitions for any processes within the scope of the assessment that are variants of the processes defined in the SPICE process model (Model for process management). The context under which a SPICE conformant process assessment takes place is illustrated in the following figure (Figure 3).

Process assessment is undertaken to understand an organisational unit's current processes. It is carried out by assessing selected processes against the process model. The model consists of a set of process-specific practices on one hand and a set of generic practices on the other hand. The generic practices apply across all processes. The practices are grouped into process capability levels 1 - 5 that define how well the process is implemented, performed and managed. The assessment output is a vector (or process profile) of generic practice adequacy ratings and process capability level ratings for each process instance assessed.

The assessment output is returned to the calling process (either process improvement or capability determination) and consists of the scores assigned to the processes assessed (the process profile), and a record of the context in which these ratings were awarded. The sophistication and complexity required of a process is dependent upon its context. For instance the planning required for a five person project team is much less than for a fifty person team. This context influences how a qualified assessor judges a practice when assessing its adequacy and influences the degree of comparability between process profiles.

The context is also important because, among other things, it is used to determine the comparability of different assessment results.

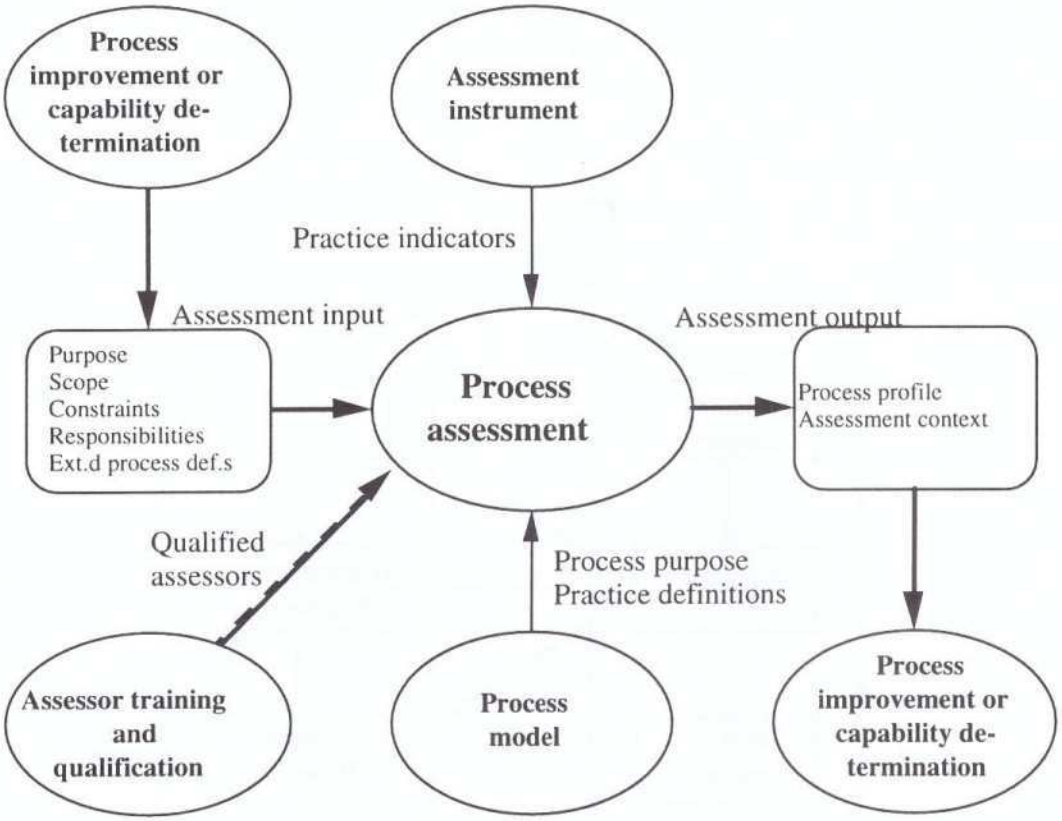


Figure 3. Assessment context

An assessment may be conducted as a self-assessment, an assisted self-assessment, a self-assessments with external verification, or an independent assessment. A team or an individual approach can be used to perform the assessment. SPICE recommends a team approach to carry out process assessment with at least one qualified assessor, who has the competence to apply the assessment framework.

The process assessment framework sets the assessment steps, defines the scoring and rating principles and offers means to present the assessment results. It is also defined to assess specific process instances according to the scope of the assessment. In the results of the assessment each process instance is characterised by a set of five process capability level ratings, each of which is an aggregation of the practice adequacy ratings that belong to that level. Practice adequacy is a rating of the extent to which a practice meets its purpose as defined in the process model. Hence the practice adequacy ratings are the foundation for the rating system. From the ratings of process instances, a number of derived or average ratings may be derived (according to the assessment approach used) that may provide better insight into the capability of a process within an organisational unit as a whole.

5. The SPICE process model

The SPICE process model defines software development processes and practices that may be implemented to establish and improve an organisation's software acquisition, development, maintenance, operation and support capabilities. Practices in this model are organised using an architecture that will help software personnel understand how to continuously improve

their software processes. The architecture has been designed to facilitate the assessment of an organisation's software processes and make judgements and recommendations regarding improvements to them.

The architecture of the SPICE process model contains two hierarchies (see Figure 4) formed from different viewpoints. The hierarchy on the left is based on grouping by type of activity while the hierarchy on the right is based on grouping by type of implementation or institutionalisation activity.

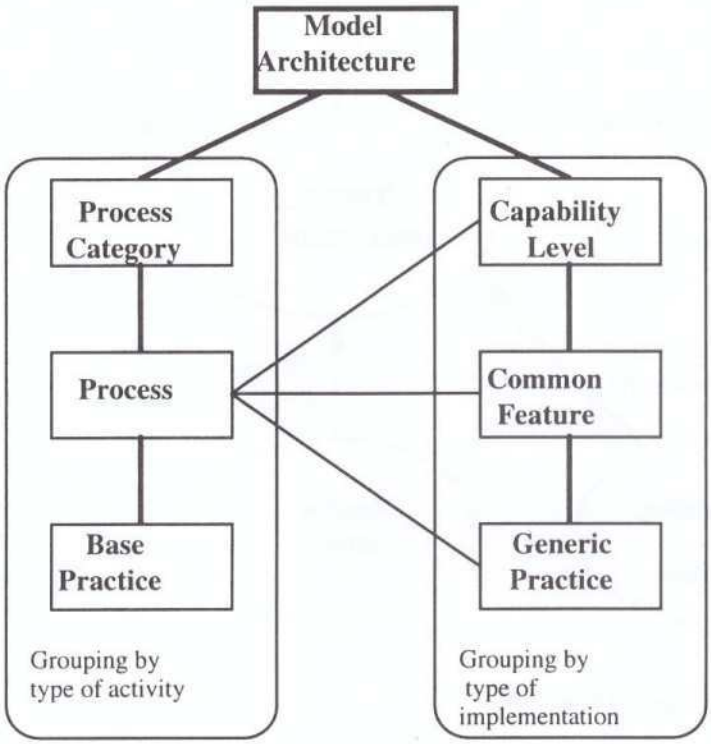


Figure 4. SPICE process architecture

The left branch of the process model is composed into process categories, processes, and base practices. Each process in the model is described in terms of base practices, which are its unique software engineering or management activities. Processes in turn are grouped into five process categories. These components of the hierarchy characterise performance of a process in the following way:

- **Process Category** Process category is a set of processes addressing the same general area of activity. For example the Engineering process category consists of the processes like: requirements definition, analysis and design, implementation, testing, and maintenance of system and software.

  The process categories covered in the process model address five general areas of activity that are: customer-supplier, engineering, project, support, and organization (see in details in Appendix 2).

- **Process** A process is a set of activities that achieves a purpose. Each process has a purpose and consists of a set of practices that address that purpose.

  For example, the Engineering process *Develop Software Design* contains practices like: develop software architectural design, design interfaces at top level, develop detailed design and establish

traceability. These practices together fulfil the purpose of the process that is stated as follows: *"The purpose of Develop Software Design is to establish a software design that effectively accommodates the software requirements; at the top-level this identifies the major software components and refines these into lower level software units which can be coded, compiled, and tested."* The purpose statement indicates the reason for performing the *Develop Software Design* process.

- **Base Practice** A base practice is a software engineering or management activity that addresses the purpose of a particular process. Consistently performing the base practices associated with a process will help in consistently achieving its purpose.

  Thus a process consists of a set of *base practices that may be p*erformed ad hoc, unpredictably, inconsistently, based on poor planning, and/or result in poor quality products, but whose work products are at least marginally usable in achieving the purpose of the process. Implementing only the base practices of a process may be of minimal value and represents only the first step in building process capability.

The processes as defined in the SPICE process model are not processes in the sense of being complete process models or descriptions. These definitions contain descriptions of essential practices, but they do not determine how to perform the process (see the summary list of process categories and their processes presented in Appendix 2). From this viewpoint the SPICE process model is rather descriptive than prescriptive in nature.

The right-branch of the SPICE process model defines evolving capability of the implemented or institutionalised processes in terms of capability levels, common features, and generic practices. The decomposition was derived based on grouping by type of implementation or institutionalised activity. The result of the composition is in details as follows:

- **Capability level** A capability level is a set of common features (sets of activities) that work together to provide a major enhancement in the capability to perform a process.

  There are six capability levels in the SPICE process model. Each of them provides a major enhancement in capability to that provided by its predecessors in the performance of a process. Together, the levels constitute a route map for improving a specific process in a logical fashion. The capability levels are a rational way of progressing through enhancements to a process. This may not, however, apply to all software environments, because organisation's own business goals may require changes that are different to capability level expectations. It should also be noted that there are dependencies between processes, and that the successful satisfaction of a capability level within a process may require support from another process.

- **Common Feature** A common feature is a set of practices that address an aspect of process implementation or institutionalisation.

  As an example, the Planned-and-Tracked level contains common features of planning, performing disciplined, verifying, and tracking performance. The tracking performance common feature consists of practices that track process status using measurement and take corrective action as appropriate.

- **Generic Practice** A generic practice is an implementation or institutionalisation practice that enhances the capability to perform any process.

> The generic practices apply to any process as a whole and can be aggregated by common features and by capability levels to describe the capability of a process. The generic practices describe the implementation or institutionalisation of a process, by determining activities necessary to manage a process and improve its capability to perform.

The capability levels are characterised by the common features as described above. The common features are comprised of the generic practices that form the basis of process ratings. The summary of the six capability levels with their common features and generic practices are shown in details in Appendix 3.

By their nature, there is more than one way to group aspects into common features and common features into capability levels. The ordering of the common features in this model stems from the observation that some implementation and institutionalisation aspects benefit from the presence of others. This is especially true if institutionalisation aspects are well established. For example, the provision of a well-defined, usable process for an entire organization to tailor and use should follow from some experience in managing the performance of that process at the level of individual projects. An example of this is that prior to institutionalising a specific estimation process for an entire organization, the organization first attempts to use the estimation process on a project. Some aspects of process implementation and institutionalisation should be considered together (not one ordered before the other) since they work together toward enhancing capability.

Common features and capability levels are important in performing an assessment and improving an organisation's process capability. In the case of an assessment where an organization has some, but not all common features implemented at a particular capability level for a particular process, the organization usually is operating at the lowest completed capability level for that process. For example, at capability level 2, if the tracking performance common feature is lacking, it will be difficult to track project performance. If a common feature is in place, but not all its preceding ones (i.e.-those at lower capability levels), the organization may not reap the full benefit of having implemented that common feature. An assessment team should take this into account in assessing an organisation's individual processes.

The components of the whole SPICE process architecture form together a process model, where the processes are composed of the base practices, and the implementation of the processes in practice is characterised by the capability levels. The SPICE principle is that the processes are not characterised or do not belong just on certain capability levels, but each of them is characterised by ratings of all capability levels.

In the case of improvement, organising the practices into capability levels provides an organization with an improvement route map should it desire to enhance its capability for a specific process. For these reasons, the generic practices are grouped into common features which are ordered by capability levels.

In either case, an assessment should determine the capability levels for all processes within the assessment scope. Processes can, and probably will, exist at different levels of capability. The organization will then be able to utilise this process-specific information to focus the improvements of its processes. The priority and sequence of the improvement of the organisation's software processes should take into account their business goals.

6. Scoring and rating principles

When a SPICE conformant assessment is performed, an organisation's implemented processes are compared with the SPICE process model. The practices in the process model form the

criteria against which the assessment is performed. At least one process instance of each process defined in the assessment scope will be included. The assessment is supported with an *assessment instrument* that is a data gathering tool (or set of tools) for the evaluation scores of the practices. An assessment instrument aids the assessor by providing a consistent set of discriminators to help judge how well the practices have been implemented in the organisational unit's processes. An assessment instrument provides a mechanism to record the scores and contextual information from an assessment. Storage and retrieval capabilities provide the ability to maintain the results and supporting information for post-assessment analysis and improvement. Sophisticated assessment instruments may help the assessor to process the data and generate the results, and thereby improving the efficiency and effectiveness of the assessment.

The process instances are evaluated by evaluating them against practice definitions and giving them scores using existence and adequacy ratings. The existence scores apply only with the base practices while the adequacy ratings may be used both with the base practice and the generic practice evaluations.

Base practice existence is scored using the base practice existence rating scale containing a binary scale of non-existent (N) or existent (Y) scores. When the base practice is scored as non-existent it is either not implemented or does not produce any identifiable work products. That means that there is no evidence of its existence. When the base practice is evaluated to be existent, there is evidence that it is implemented and that it produces identifiable products.

A process is considered to be at capability level 0 (Initial) when none of its base practices is existent. If the process exists, then the assessment team may try to understand whether all its base practices are adequately performed (at capability level 1) using the following four-point scoring scale:

- **Not adequate:** The base practice is either not implemented or does not to any degree contribute to satisfying the process purpose;
- **Partially adequate:** The implemented base practice does little to contribute to satisfying the process purpose;
- **Largely adequate:** The implemented base practice largely contributes to satisfying the process purpose;
- **Fully adequate:** The implemented base practice fully contributes to satisfying the process purpose.

From level 2 to 5 the assessors will judge whether the process is managed and institutionalised in terms of being planned, tracked, defined, measured and continuously improved. The assessor's scores at each these levels is based on the *generic practice adequacy rating* defined as *a judgement, within the process context, of the extent to which an implemented generic practice satisfies its purpose.* The scoring scale used for the adequacy rating of the generic practices has also four discrete values as follows:

- **Not adequate:** The generic practice is either not implemented or does not to any degree satisfy its purpose;
- **Partially adequate:** The implemented generic practice does little to contribute to satisfy its purpose;
- **Largely adequate:** The implemented generic practice largely satisfies its purpose;
- **Fully adequate:** The implemented generic practice fully satisfies its purpose.

Using this scale, the assessment team examines all process occurrences, within the scope of the assessment, and scores adequacy for each generic practice in each capability level. The result for each capability level (except level 0), is derived using equal weight for each score

and applying a percentage scale. The resulted capability level rating of each process is defined as a matrix: CL n = [%N, %P, %L, %F],

where CLn = capability level (n =1,..5)[5],

%N = percentage of generic practices judged as not adequate,

%P = percentage of generic practices judged as partially adequate,

%L = percentage of generic practices judged as largely adequate,

%F = percentage of generic practices judged as fully adequate, and where

%N + %P + %L + %F = 100 %.

A typical output from a SPICE assessment for a process X, may therefore be:

CL1: [0%, 0%, 40%, 60%]

CL2: [30%, 35%, 25%, 10%]

CL3: [80%, 20%, 0%, 0%]

CL4: [0%, 0%, 0%, 0%]

CL5: [0%, 0%, 0%, 0%].

The result may also be presented graphically in form of a process profile.

7. Conclusions

The SPICE project produced its first baseline set of documents (version 1) listed in Appendix 1 in June 1995. These documents will be trialed in two phases. In the first phase the SPICE process model (Part 2) and assessment guides (Parts 3 and 4) will be trialed in practice. The feedback and the findings discovered during the trial phase may affect changes into the documents. In the second trial phase the assessment instrument guide (Part 5), process improvement guide (Part 7) and capability determination guide (Part 8) will be integrated into the parts already trialed and corrected. At the same time the focus of trialing will be enlarged into the usability of the assessment results. The two years trial period is expected to help in solving some technical questions that still are open to ensure the usability of the evolving standard in the practice.

The CMM approach for process assessment and the ISO 9000 approach for quality system certification have to date formed the reference points for the software development community. Therefore, nearly all of the commercial and internal assessment and capability determination approaches have been developed based on these approaches. Now, the results of the SPICE project form a complete framework for developing a software process assessment, improvement, and/or capability determination approach. New versions of the existing commercial assessment, improvement and capability determination methodologies will be SPICE compliant, especially, the methodologies that have provided the foundation for the SPICE work, because they may now in turn gain from the new expertise resulting the SPICE effort. Additionally, new internal software process assessment, improvement and capability determination approaches will be developed and already are under development based on the SPICE framework.

As software process improvement is now in the focus of software engineering[6], new commercial (like Drive-SPI, [13]) and internal software process improvement methodologies

---

5 The special case is level 1 where there is only one geeneric practice. Therefore the initila score and the rating are equal on that level.

6 For example the new European ESSI (European Systems and Software Intiative) with Best Practice Program, and ESPITI (European Software Process Improvement Training Initiative) funded by the Commission of European Countries.

will be developed starting with the results of SPICE. Linkages are also being defined to use SPICE in conjunction with extended ISO 9000 certification services, and in procurement with Euromethod guidance.

Acknowledgements

This paper is mainly based on the recent documents produced in the SPICE project that are still unpublished. It would not be possible to write this paper without to use the contribution of the members of the project: in particular the product team members, reviewers, product managers and the project management.

References

[1] Humphrey, W.S. and Sweet, W.L., "A Method for Assessing the Software Engineering Capability of Contractors", SEI Technical Report SEI-87-TR-23, Software Engineering Institute, Pittsburgh, September 1987.

[2] Kuvaja, P., Similä, J., Krzanik, L., Bicego, A., Koch, G., and Saukkonen, S., Software Process Assessment and Improvement. The BOOTSTRAP Approach. Blackwell Business, Oxford, UK, and Cambridge, MA 1994.

[3] Paulk, M., et al. Capability Maturity Model for Software, Version 1.1, CMU/SEI-93-TR-24, February, 1993.

[4] Paulk, M., et al. Key Practices of the Capability Maturity Model, Version 1.1, CMU/SEI-93-TR-25, February, 1993.

[5] Bell Canada Trillium - Telecom software product development capability assessment model, Draft 2.1, July 1992, Canada.

[6] British Telecom, Healthcheck.

[7] Compita Ltd, Software technology diagnostics, V2.5.

[8] ISO/IEC JTC1/SC7/WG7/SG1, *The Need and Requirements for a Software Process Assessment Standard, Study* Report N944R, Issue 2.0, 11 June 1992.

[9] ISO/IEC JTC1/SC7/WG10 N002R, *Modus operandi,* Ver. 1.01, March 1995.

[10] Dorling, A., "SPICE: Software Process Improvement and Capability dEtermination", Software Quality Journal, Volume 2,1993, pp. 209 - 224.

[11] Deming, W.E., Out of the Crisis, Cambridge, MA: MIT Center for Advanced Engineering, Study, 1982.

[12] Humphrey, W. S., Managing the software process. Addison-Wesley Publishing Company Inc., Reading, Mass. (1989).

[13] Maupetit, C., Kuvaja, P., Palo, J., Belli, M., and Isokaanta, M., The DriveSPI project - a risk-driven approach for software process improvement, to be published in the Proceedings of 8th International Conference on Software Engineering and its Applications, to be held November 15 - 17 1995 in Paris, France.

ami: a Taylorable Framework for Software Process Improvement

C. Debou, N. Fuchs, M. Haux,
Alcatel Austria AG
Scheydgasse 41
A-1210 Vienna

At the first ISCN seminar, **ami** (application of metrics in industry) was introduced as a new paradigm for software process improvement. It was one of the first attempts to reengineer successful technologies e.g. the G/Q/M [BaR88] into a general framework. Measurement was stressed as being an essential component of software process improvement for understanding, monitoring the changes and evaluating the return on investment.
This paper describes how major software process improvement approaches like CMM, Bootstrap or even ISO 9001 fit into the **ami** framework. The objective is to demonstrate on the one hand the ability of ami activities to be taylored to those specific methods and on the other hand the necessity of having an overall strategy (cycle) for software process improvement e.g. ami. Furthermore, newly defined software process improvement cycles (SEI IDEAL, SPICE) are described and also compared to **ami**.

1. INTRODUCTION

When the software process issue was first raised, the emphasis had been on how to assess your process [HuS87]. Then W. Humphrey wrote his famous book "Managing The Software Process" [Hum89] which served as basis for the development of the Capability Maturity Model [CMM93a, CMM93b]. The CMM and other derived methods (Bootstrap, Trillium) are a real step forward for performing software process improvement, describing software engineering best practices in an evolutionary way but are still not sufficient. The overall vision (all necessary detailed steps towards an appropriate software process improvement programme) was missing. We now finally reach that phase ("re-engineering phase"). In software re-engineering, all components/modules are studied to derive the design. For SPI, components like assessment procedure, CMM, ... were existing but not integrated within an improvement cycle. **ami** (application of metrics in Industry) was one of the first attempts to combine together successful methods in one improvement cycle.
The **ami** approach has been extensively promoted during the past years in major conferences and journals [DLP92, DPF92, DFS93, RoW94]. At the first ISCN conference who took place in Dublin in May 1994, an in depth presentation of **ami** as a framework for software process improvement [Deb94] was performed and illustrated by case studies from the industry. In October 1994, during the first **ami** user group in Basel [AMI94], one of the main concerns was how **ami** relates to other methods like SEI CMM, Bootstrap [MHK94], SPICE, ISO 9001. Is **ami** working totally separately from major initiatives? Users always worry about the compatibility and adaptability aspects of the methodology.

The purpose of this paper is twofold:

- To describe how those existing methods or paradigm fit into the **ami** framework namely SEI CMM, Bootstrap, ISO 9001/TickIT. This will be illustrated through examples.
- To demonstrate how **ami** is influencing current initiatives ensuring its future compatibility e.g. SPICE, SEI IDEAL.

After a brief introduction of **ami** as an improvement framework, the previously listed methods will be mapped into the **ami** cycle. The same framework will be applied for each method: A brief introduction and a description of the 4 generic **ami** activities (assess, analyze, metricate, improve) instantiated with one of the methods.

2. THE AMI APPROACH: A QUANTITATIVE APPROACH TO PROCESS IMPROVEMENT

2.1 Origin.

ami (application of metrics in industry) was a two-year project which started in December 1990 under sponsorship of DG XIII of the Commission of the European Communities through the ESPRIT programme promoting the use of measurement in software development. The goal of the project was to develop a practical approach and to validate it on a variety of projects all over Europe [PFS92, DLS93]. The approach is described in the **ami** handbook [AMI92]. The **ami** paradigm (assess, analyze, metricate, improve) is similar to the Shewart cycle (plan, do, check, act) for process improvement. **ami** has taken this cycle, based on common sense principles, and developed it for software measurement mainly in the context of software process improvement. The **ami** method is an stepwise, iterative, incremental, goal-oriented procedure coupling together a model-based process assessment technique with a quantitative approach to software development issues from the viewpoint of the process, product and resources.
The twelve steps are grouped by three in activities which are highlighted next (see schematic description in figure 1) .

2.2. Assess

The first step deals with the assessment of the software development environment for defining primary goals for metrication. Weaknesses and critical parts of the software development process are first pointed out. From those findings and business objectives as defined by the management, primary goals are defined (step 2). A hierarchy of top level goals is being considered depending on the maturity of the development process. Before setting up "change"goals (e.g. to improve productivity while maintaining quality), one should envisaged "understanding" goals e.g. support of project management with process and product metrics. The assumption behind improvement goals is that the process is well defined. Then a validation of goals against the assessment conclusions, the timescale and the budget is performed to avoid too ambitious goals are set up (step 3).

2.3 Analyse

The aim of the second activity is to build a so-called goal-tree which is the translation of the primary goal into sub-goals and metrics (step 4). Primary goals are broken down into more manageable sub-goals until directly quantifiable goals are reached. The process is based on the Goal/Question/Metric paradigm [BaR88]. After having verified the consistency of the tree (step 5), metrics are derived from these bottom goals with the help of questions (step 6). The results are a documented goal tree and the associated set of metrics.

2.4 Metricate

The metricate activity encompasses three steps:

1. Writing the measurement plan which is the reference document for collection and analysis of data and for ease of tracing of these tasks (step 7)
2. Collecting the data (step 8)
3 Verifying the data (step 9)

2.5 Improve

The exploitation of measures has to be performed in reference with the goals defined in the analyse activity. The improvement activity starts with an appropriate presentation of the measurement data (step 10). Graphics (histograms, pie charts, scatterplot, ...) should present the data so that both trend and outliers can be detected. Then, by relating data to goals (steps

11 and 12), it has to be determined whether the goals are fulfilled, how quickly they are fulfilled or why they did fail. Further corrective and/or improving actions are based on this determination.

The first loop is achieved. Now, it is worth quantifying the first benefits of the measurement programme. The iterative aspects of the approach permits not only -by a reassessment- the refinement of goals and consequently the metrics set, but also the improvement of the metrication process.

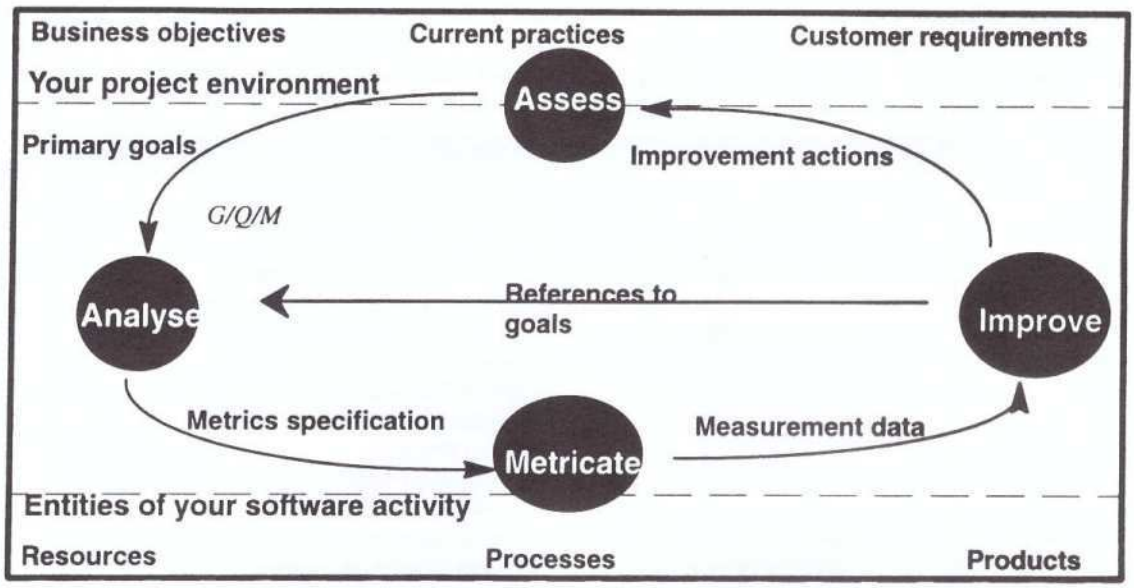


Figure 1. ami improvement cycle

3. AMI AND SEI CMM

3.1. Introduction

The comprehensive Capability Maturity Model, CMM, with its current version released in 1993 [CMM93a,b], is extensively used in industry for

- identifying strengths and weaknesses of a software development organization, i.e. benchmarking to industry's best practices
- supporting process improvement initiatives
- helping software procurers evaluate the capability of contractors

In the following sections we will take a look at possibilities to use ami as a quantitative framework for CMM-based process improvement programs.

3.2. Assess

A primary business goal of the top management is to control and eventually reduce software development costs. It has been shown by experience that one way to achieve this is to use the SEI Software Process Assessment (SPA) method for investigating strengths and weaknesses and creating change momentum and then use the SEI CMM as a roadmap for improvement actions.

SEI SPA is a method for assessing the organization's software development process. It is an intensive 5-day investigation lead by a team of 6 to 8 individuals. During the week this assessment team gets input from 30 to 40 of the organization's employees, and out of this input establishes the organization's maturity level and formulates about 8 findings, i.e. main

problem areas which the organization is facing. The consequences of these problems in the organization are considered as well.

The already mentioned CMM, see table 1, contains a number of key process areas (KPA) for level 2 to 5. Each key process area has its specific goals and a set of key practices for reaching these goals. The key practices are categorized in commitment, abilities, activities, measurement & analysis, and verification - the so called common features.

The final report of the SPA contains the already mentioned findings and consequences as well as suggestions concerning what to do to improve the process. The findings and suggestions are often mapped to, but are not limited to, the key process areas of the CMM. On the basis of the final report concrete planning of improvement actions is done, resulting in an action plan with detailed descriptions of improvement tasks with responsibilities, effort estimates, etc.

Level	Key Process Areas
5 - OPTIMIZING	Defect Prevention
	Technology Change Management
	Process Change Management
4 - MANAGED	Software Quality Management
	Quantitative Process Management
3 - DEFINED	Peer Reviews
	Intergroup Coordination
	Software Product Engineering
	Integrated Software Management
	Training Program
	Organization Process Definition
	Organization Process Focus
2 - REPEATABLE	Software Configuration Management
	Software Quality Assurance
	Software Subcontract Management
	Software Project Tracking and Oversight
	Software Project Planning
	Requirements Management
1 - INITIAL	

Table 1. Capability Maturity Model Overview

One major principle of the CMM is that the sequence in which changes are performed is important. Thus the action plan will concentrate on the key process areas one level higher than the organization's own maturity level, e.g. a level one organization may have one action domain concerning requirements management (RM).

The top management must now allocate enough resources, i.e. people for working groups, to implement the action plan contents within 12-18 months. Main focal point for the software process improvements is the newly established SEPG, software engineering process group. Its tasks will be to coordinate, control and promote the process improvement program [FoR90].

The primary management goal is at this stage to successfully implement the action plan, while keeping track of costs. The organization was assessed to be level 1- thus the top goal is *G1- Reach CMM level 2.*

3.3. Analyse

As 2nd level goals, the findings from the final report has benn chosen. The corresponding consequences of the deficiencies might provide ideas on what metrics may be used (see figure 2 where we show the subtree in full depth). For example one major consequence of the lacks in requirements management was high rework effort caused by non-controlled requirements changes and weak requirements traceability.

3rd level goals are the domains (high level definition of improvement actions) of the action plan. Thus the ami goals tree was used to show the mapping between findings and these domains, see figure 1.
In the 4th level there are two paths, one for each of the management's main concerns. The left one covers implementation/ institutionalization of the incomplete key practices. For the

domains of the action plan not representing key process areas of the CMM a similar structure is possible. The right path goal is to monitor the costs and benefits of the new process element, in this case requirements management.

The metrics for implementation/ institutionalization measure to what extent the key practices of a certain common feature of requirements management has been covered, separated in definition and usage, as appropriate.

The measures of costs and benefits are on one hand the cost of requirements management activities and on the other hand, relating to the already discussed consequence with high rework, rework effort related to lacks in requirements management.

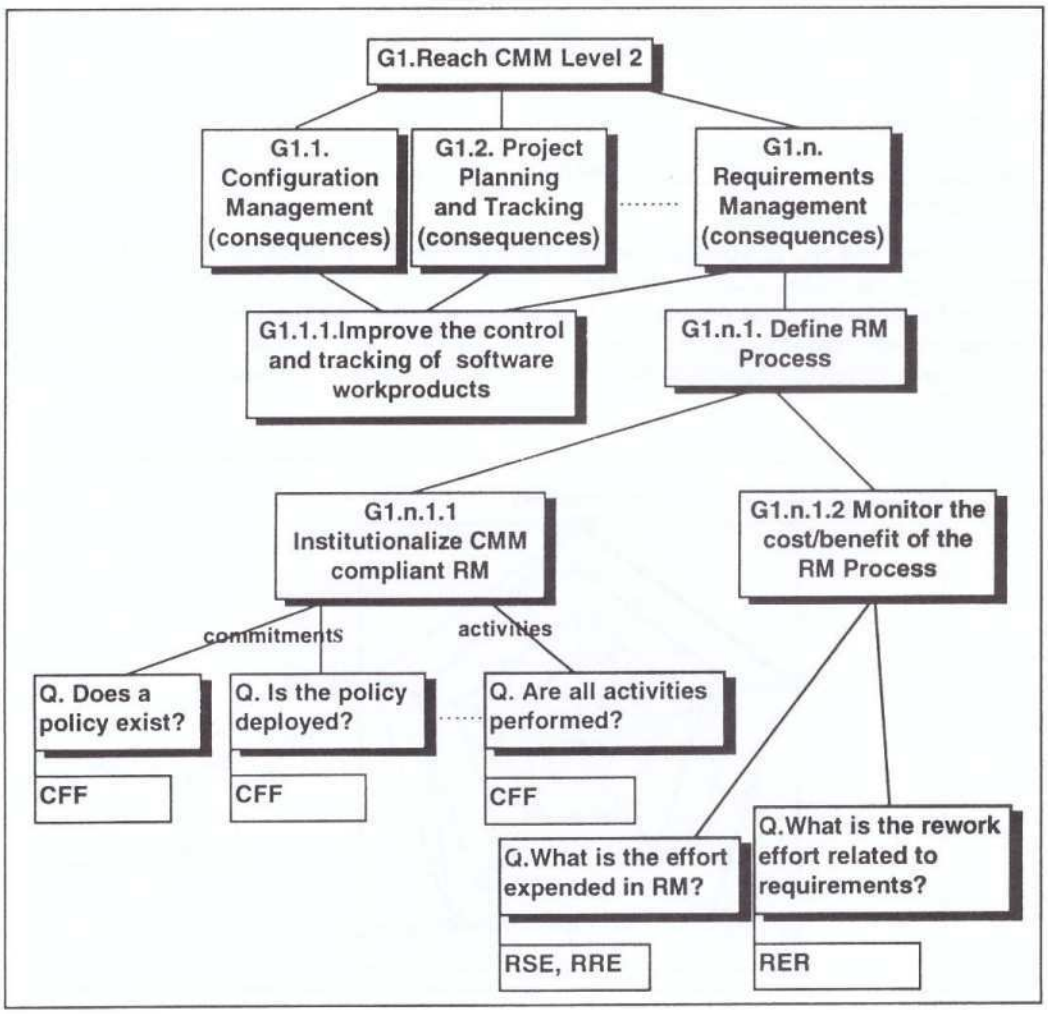


Figure 2. Goal tree

3.4. Metricate

As shown in figure 2, some metrics that we will collect for the action domain requirements management are

- Common Feature Fulfillment, *CFF*, indicates to which degree the key practices of a specific common feature is fulfilled.

- Requirements Specification Effort, *RSE*, is a measure of the total effort spent in documenting new and changed requirements.
- Requirements Review Effort, *RRE*, is a measure of the effort spent in preparing and conducting reviews of new and changed requirements.
- Rework Effort related to Requirements, *RER*, is a measure of the rework effort caused by lacks in requirements management. To be able to get this information, defects found in reviews and test should be assigned to the phase in which they were introduced.

The necessary information is gathered through forms or, in the case of CFF, by the SEPG in periodic meetings with the working groups. For capturing information on CFF, the Interim Profile method of SEI might be useful [WNH94].

3.5. Improve

The data will be presented by the SEPG periodically in reviews with working groups and management.

Management will like to see that the primary goal, reaching CMM level 2, is accomplished in time. Using the radar chart (Kiviat) in 3, the SEPG can make progress within each action domain towards this goal visible. For management information, an additional radar chart with compound information, i.e. for each action domain one degree (like NS=Not Satisfied, PS=Partially Satisfied, FS=Fully Satisfied), could be useful. This would be similar to the SEI Interim Profile.

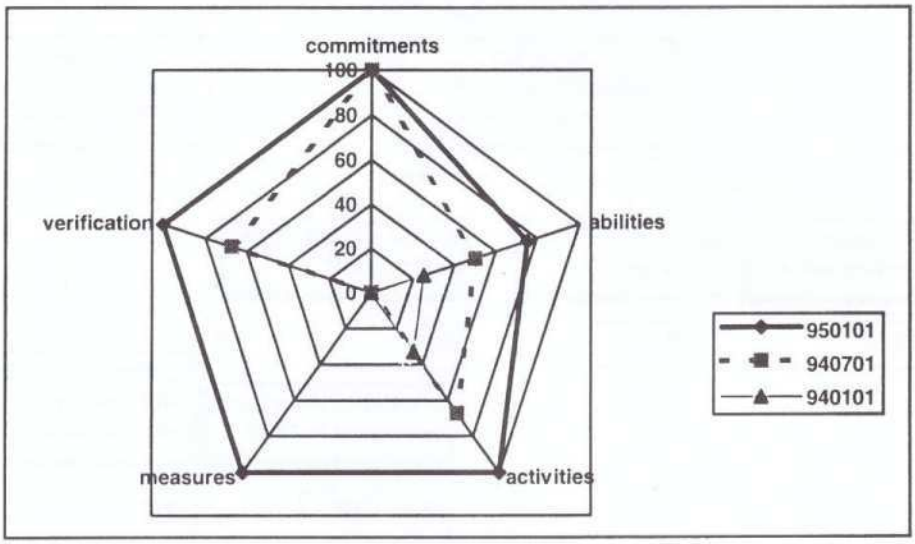


Figure 3. Monitoring the implementation/institutionalization (mapping to goal G1.n.1.1)

Also, the cost increase will be monitored and compared with the reduction in rework effort (remember - high rework effort was recognized to be a major consequence of lacks in the RM process in this organization). Any anomalies in the relation between these cost deltas can be easily recognized and causes analyzed to make corrections of the improvement program in time.

These measured benefits of the improved requirement management process could also be used for return on investment calculation. Of course, to do this we need to measure the effort put into definition of new procedures, training, etc.

After 12 to 18 months a full SPA-style re-assessment will be made, to validate the improvements made and set pace to continue improvements towards level 3.

4. AMI AND BOOTSTRAP

4.1 Introduction

As **ami**, the Bootstrap project [HMK94] was performed within the frame of the ESPRIT programme. Its goal was to develop a method for software-process assessment, quantitative measurement and improvement. Bootstrap enhanced and refined the SEI's process assessment method and adapted it to the needs of the European software industry i.e. including ISO 9000-3 (guidelines for software quality assurance) attributes and the ESA's PSS05 software engineering standards.

4.2 Assess

The Bootstrap method covers the assessment activity. A maturity level is calculated for each attribute of the process quality model by answering a questionnaire. This model takes into account three main dimensions in the process: organization, technology and methodology issues, methodology being broken down in life-cycle-independent functions, life cycle functions and process related functions. Bootstrap provides a detailed profiling technique as well as a standard format of an action plan. But there is no goal-oriented approach to map the maturity profile onto improvement activities and goals. The idea would be to analyse the profile and to derive main goals (mainly based on the improvement of weak quality atrributes). Then a goal tree is derived which nodes contains goals and necessary actions (Action plan). Metrics are associated to each goal in order to measure whether the goals are achieved.

The proposed approach is illustrated with a real case study [HMK94] derived from a Bootstrap article in IEEE Software of july 1994. The assessed organization is developing a control system to visualise industrial processes. The assessed project is currently in the maintenance phase, and the project is continued with development of a larger system in which the assessed system is integrated. An improvement project was started and runs in parallel with the development introducing software engineering practices based on the findings of the Bootstrap assessment. The bootstrap profile showed big weaknesses in project management and design.

Design: 12 Person days were needed to correct an error during maintenance because there was no graphical or textual representation of the module structure, the links among modules and the product interfaces

Project management: No detailed workflow charts and cost and effort based on sound historical data were available despite the fact that a project-management tool has been purchased. But since no training had been performed, it is never been used properly.

Consequently, the top level goals and related actions are:

G1: To reduce effort for maintenance
A1: Introducing a case tool based on structured design

G2: To produce reliable estimates
A2: To implement a project management methodology based on the existing tool.

4.3 Analyse

From the analysis of the top level goals and related actions, a framework for action plan and measurement is built (see figure 4).

The goal tree merges main improvement actions and related goals. The leafs define the necessary metrics to ensure the achievement of the goals.

With regards to G1, a redesign of software will be performed. The benefits will be evaluated on the next maintenance activities. With regards to G2, the existing project management methodology shall be applied and supported with the definition of a historical and tracking database for estimation purposes.

4.4 Metricate

For supporting and evaluating the benefits of the redesign, the following metrics were collected:

- Interface complexity metrics to select critical parts of the software
- Cost of redesign and reimplementation for the forthcoming cost/benefits analysis
- Cost of maintenance activities after redesign
- Number of defects as an indicator of quality for the redesigned product.

For project management, typical project progress metrics from completed projects were collected for feeding the historical database (effort, size, time) and facilitating the estimation for new projects. The same data were collected for project tracking purposes

4.5 Improve

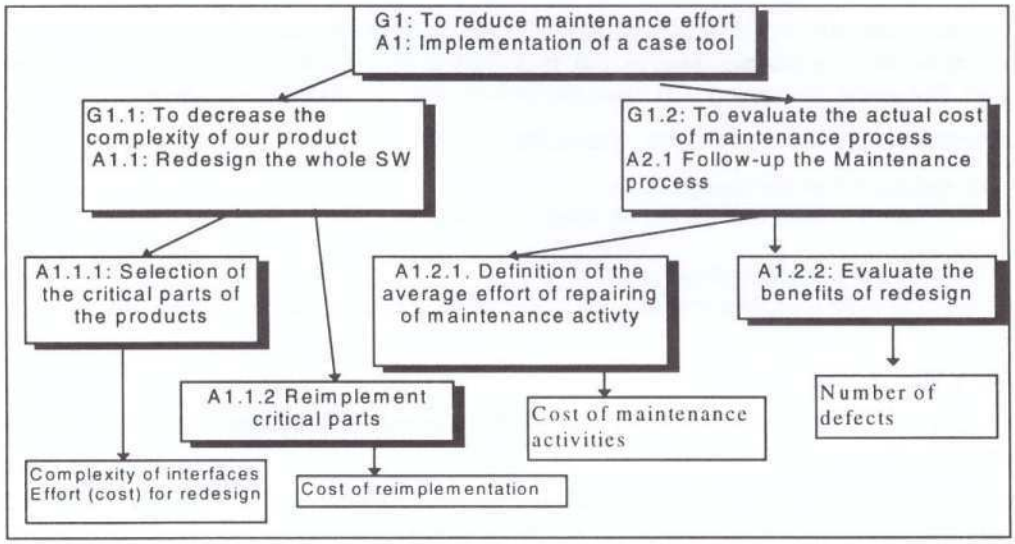
The data analysis revealed the following facts:

G1.1.1
The redesign took only 4 person-months. The redesign was not really important but the effect of design on the maintenance seemed to be important.
After re-design of parts of the system and the use of professional design techniques in a further project, a reduced meaningful effort per maintenance activity (about 4 person days) was achieved, which was about one third of the previous effort. Moreover the redesign effort was amortized in about 6 months of maintenance activities. The quality of the product significantly increases.

G1.2.1 :
The implementation time was reduced by half with the same amount of effort. As a matter of fact, the graphical and textual design facilitated the decomposition of the product into different units that could be developed in parallel.

G2.1
The main benefits were the clear breakdown of a project into workpackages together with a more reliable resource estimation. In the requirements phase resources were estimated within 30% and after analysis and design within 10%.

Repeating the Bootstrap assessment will check whether the weak process quality attributes detected at the first reassessment have been improved.



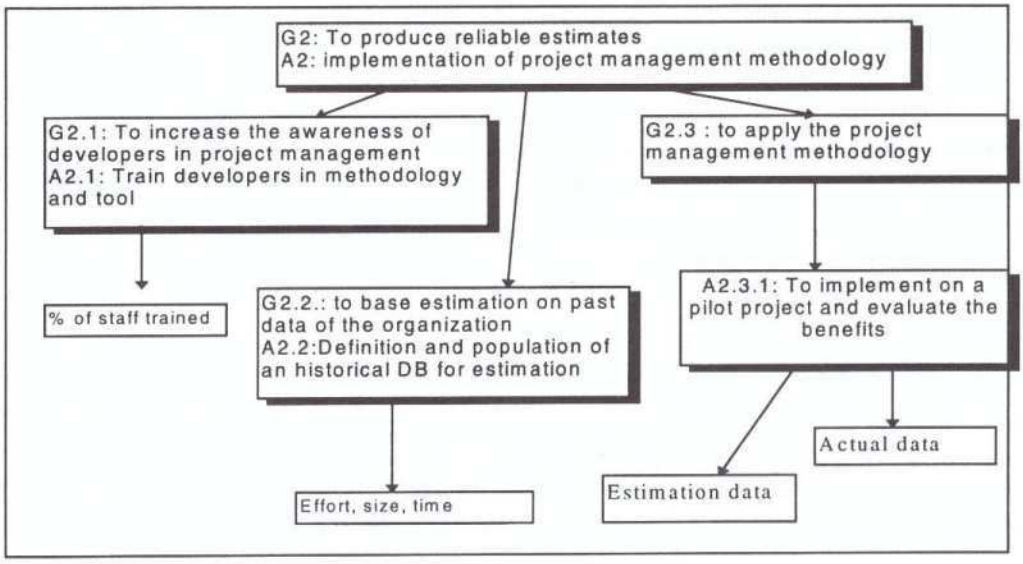


Figure 4. Goal trees

5 AMI AND ISO 9000-3

5.1 Introduction

In the last years ISO 9000 series became more and more THE quality standard. Never before a standard has got such a broad acceptance all over the world.

As ISO 9000 is independent from the industry segment, IT industry has some problems with the application. 1991 ISO 9000-3 Guidelines for the application of ISO 9001 to the development, supply and maintenance of software attacked this problem but was only a guideline. TickIT is an initiative to take ISO 9000-3 as a baseline and uses it for certification purposes. This is not a standard but a UK based certification scheme towards the ISO 9001 application in IT.

To demonstrate a quality management system according to ISO 9000-3 (or ISO 9001) internal audits have to be performed regularly. The nonconformities are handled through so-called corrective actions. Corrective actions aim at either improving a specific process or improving the way a process is followed.

In the following we show how ami can be apply to use this mechanism of internal audits for starting a process improvement cycle. The tracking of corrective actions is required by ISO 9000-3, nevertheless there are no guidelines how to do it. We will see how ami can be used for tracking and visualisation of the improvements.

5.2 Assess

As described in section 2 the first phase of the ami method deals with the assessment of the software development environment. In our example the assessment required by ami can be done through an internal audit. ISO 9000-3 requires a checklist. On hand of this checklist as well the quality system as written on paper as well as how far daily life differs from the written procedures is assessed. The weaknesses are described in corrective actions describing the nonconformities. For the application of ami the next step is to define goals as a baseline for the analysis phase. In our example TickIT certification is used as top-level goal.

5.3 Analyse

In the analysis phase the results of the internal audit are analysed in order to define improvement actions. As ami requires the result of this phase is a goal-tree. How our example is decomposed can be seen in figure 5. The goals are broken down until quantifiable goals are

reached. In our case all the bottom goals are either metrics concerning the definition or concerning the deployment.

5.4 Metricate

The Measurement plan for these very simple metrics has to be defined here. In our example we have two different kinds of metrics: level of definition and percentage of institutionalisation.

For the definition metrics (DF definition factor) we use three categories

- \- redefinition necessary
- 0 minor weaknesses
- \+ o.k.

For the institutionalisation metrics (IF) we use five categories:

- 0% little effective usage
- 25% applied to about one-quarter of the potential
- 50% applied to about half the potential
- 75% applied to about three-quarters of the potential
- 100% applied to full potential in all relevant areas and activities.

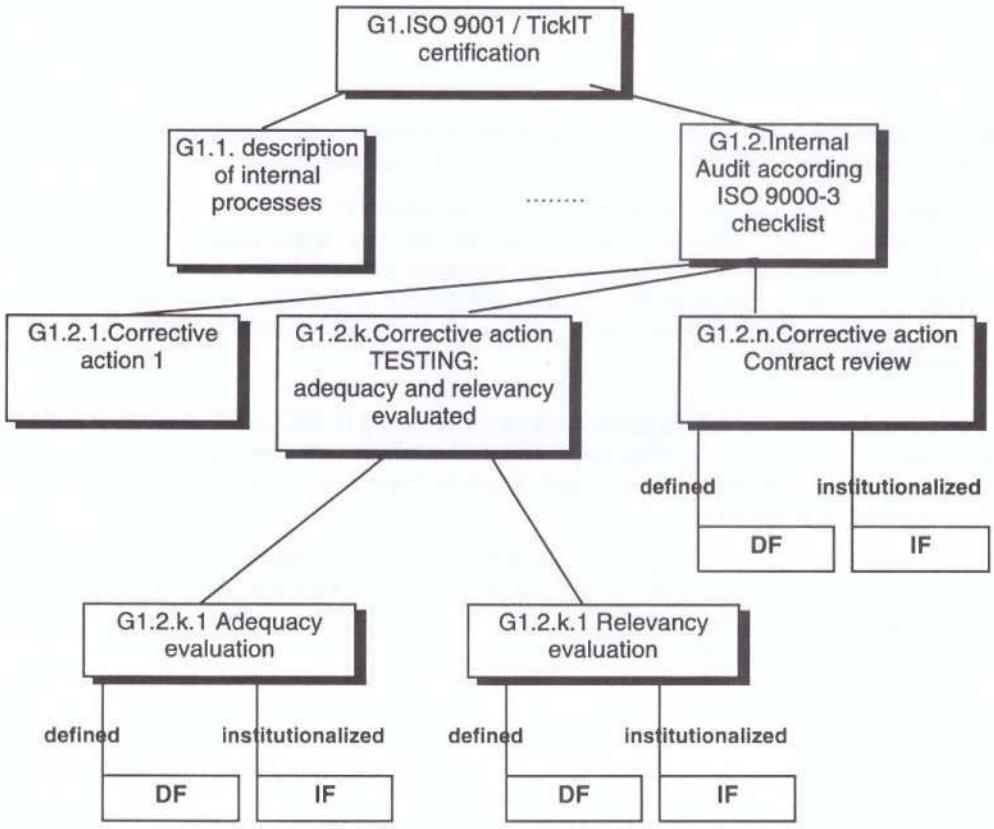


Figure 5. ISO 9001 / TickIT goal tree

5.5 Improve

Through follow-up audits an assessment of the actual status of the progress can be made. The following diagram can be used for monitoring. After closing all corrective actions the next higher level goal is reached.

	Definition			Institutionalization				
Element	-	0	+	0	25	50	75	100
Quality System Framework								
Management responsibility		X				X		
Quality system	X						X	
Internal quality system audits		X					X	
Corrective actions			X					X
Life Cycle Activities								
Contract review			X					X
Purchaser's requirements specification		X						X
Development planning			X					X
Quality planning	X					X		
Design & implementation			X				X	
Testing & Validation		X					X	
Acceptance		X					X	
Replication, delivery & installation	X				X			
Maintenance			X				X	
Support Activities								
Configuration management	X						X	
Document control			X			X		
Quality records			X		X			
Measurement			X			X		
Rules, practices & conventions			X					X

6. AMI AND SPICE

6.1 Introduction

The ISO Software process assessment initiative has grown out of an initial UK MOD effort call ImproveIT. ISO/JTC1/SC7/WG10 develops an international standard for software process assessment.

The objective of SPICE (Software Process Improvement and Capability Determination) is to provide a common approach and framework for assessment and improvement. Initial field trials of SPICE have started in January 1995.

We will concentrate on the so-called Process improvement guide [PIG94] which describes steps to perform within a complete improvement cycle (based on the SPICE assessment technology). **ami** has advocated and successfully applied the measurement techniques in a process improvement context. [DeS93, DLS93, DKR94]. SPICE makes use of the **ami** concepts to design its improvement cycle. The figure 6 show a mapping of the PIG improvement steps onto the **ami** activities.
In parallel, a mapping is made with IDEAL, a framework for software process improvement as recently being defined by the SEI [RaP94]. SPICE has generalized IDEAL.

6.12 Assess

A process improvement programme is triggered by the organization's needs, for instance, to fill in the market expectations or from a customer request in terms of maturity level. Initiating process improvement consists in defining an overall plan like any types of project, including responsibilities, resources, timeframe (**Initiating** phase of IDEAL).

How to conduct a SPICE assessment is described in the Process Assessment Guide. [PAG94]. The output mainly consists in conformance measurements which describe to what extend an organization's software process conforms to a specification of industry base practices contained in the Baseline Practices guide [BPG94] as well as additionnal comments (example of good practice, suggestions, etc.) consider as useful for the analysis.

Conformance measurement is undirectly connected with the organization's needs and may be therefore complemented with other metrics, the effectiveness measures (resulting from the previous cycle - see next analyse activity). It corresponds to the **diagnosing** phase of IDEAL.

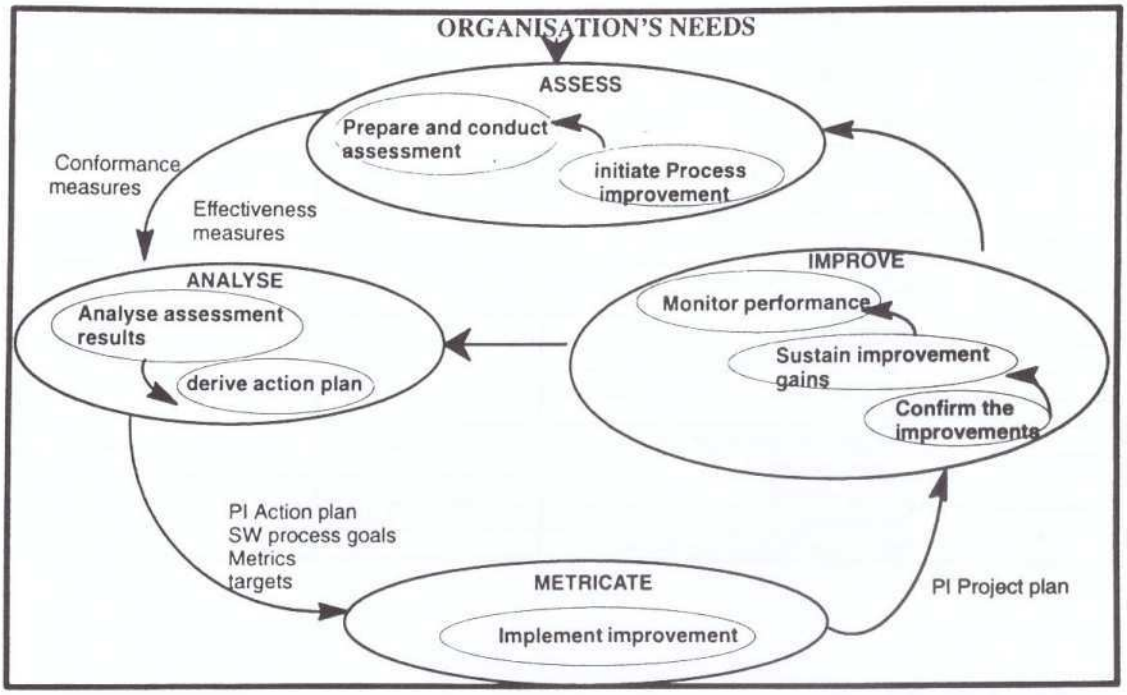


Figure 6. SPICE process improvement cycle

6.3 Analyse

The analysis (**Establishing** phase of IDEAL) is split into three steps:

- identify areas for improvement: based on the assessment results as well as effectiveness measures. They help in identifying areas in the software process which do not adequatly meet the organization's needs.
- Set qualitative software process goals for each selected area for improvement and derive suitable metrics to measure achievement of these goals.
- Set appropriate improvement targets values for these metrics (either for process effectiveness or target capability profiles) which meet the organization's needs.

An action plan is derived describing in detail process improvement actions to meet the process goals and quantified targets.
The process measurement framework is illustrated figure 7 and compare with **ami**'s measurement structure.

6.4 Metricate

A detailed action plan is developed and implemented (**Acting** phase of IDEAL)

6.5 Improve

When the action plan is completed, the organization monitor whether the current measurement of process effectiveness confirm the achievement of improvement targets and goals. Risks and cost and benefits should be also evaluated.

As soon as the improvement is confirmed, the improvement gains need to be sustained, i.e. to monitor institutionalisation of the improved process and to give encouragement when necessary. Then, the performance of the organization's software process should be continuously monitored using the effectiveness and conformance measures previously defined. New assessment can take place when a long term goal is about to be achieved or when the organization needs changes (e.g. higher capability). This is called the **leveraging** phase in SEI IDEAL

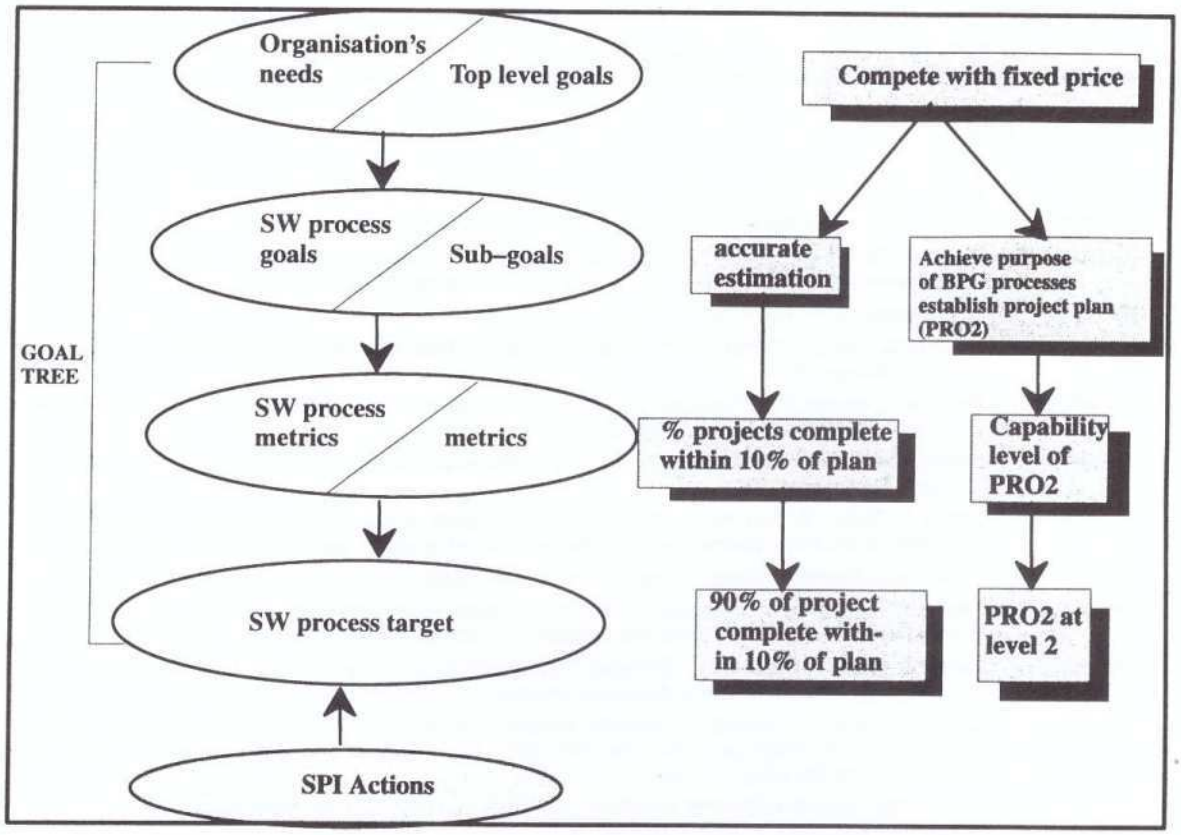


Figure 7: SPICE process measurement framework

8. CONCLUSION

An overview on existing software process assessment and improvement approaches and how they fit in the **ami** framework was provided. They all converge to the same improvement cycle as initiated by **ami** and taking its roots in Deming's work [Dem86]. They are all based on a few agreed basic principles such as continuous and iterative process, business needs driven, essential role of measurement to support process improvement.

The added value of **ami** relates to the effective implementation of measurement whatever the context is, i.e. software process improvement program, ISO 9001 certification etc.. Even though measurement is being identified as a crucial management tool, the same mistakes are still made, lack of goals, no vertical commitments (at all levels of management) within the organization and consequently misunderstandings between practictioners and management on what the objectives are. Those points may have caused that two third of all software process improvement initiative in the US have failed (this gossip is being confirmed at the last 1995 SEPG conference in Boston by B. Curtis and M. Paulk).

What is new about **ami**? The **ami** user group has been officially started beginning of 1995. An ami method update workshop took place in May 1995. A version 2 of the ami Handbook

should be available by the end of this year, including more guidelines towards applying ami in different context i.e. SPI, procurement, ... Large companies known as being at the leading edge in their field are applying **ami** as part of their software process improvement programme e.g. HP, Nokia, Schlumberger. A mailing list has been also set up[1].

Finally, to answer the question raised in the introduction, **ami** partners are involved in major initiatives such as SPICE and are collaborating with the European Software Institute. Hence, the **ami** method and the related tools will evolve in accordance with those standards.

Acknowledgements: The authors would like to thank all partners involved in the **ami** project who permit this work to be carried out. Special thanks to A. Combelles from Objectif Technology and one of the contributors to the PIG and R. Messnarz from ISCN as well as the reviewers of the European Software Institute for their valuable comments.

References

[AMI92] Esprit II 5494 ami: ami Handbook, Published version, March 1992.

[AMI94] Proceedings of the first ami user group meeting, Basel, October 1994.

[BaR88] Basili, V., Rombach, H.: The TAME Project: Towards Improvement-Oriented Software Environments, IEEE Transactions on Software Engineering, Vol 14, Number 6, June 1988.

[BPG94] SPICE consortium: The Base Practices Guide, Draft

[CMM93a] Paulk M C, Curtis B, Chrissis M B: Capability Maturity Model for Software, version 1.1, CMU/SEI-93-TR-24, February 1993.

[CMM93b] Paulk M C, Weber C V, Garcia S M, Chrissis M : Key practices of the Capability Maturity Model, version 1.1, CMU/SEI-93-TR-25, February 1993.

[Deb94] Debou C: ami a new paradigm for software process improvement, In: Proceedings of the first ISCN Seminar, Dublin, May1994

[DeS93] Debou C, Stainer S.: Improving the maintenance process: a quantitative approach, In: Proceedings of the 6th international conference on software engineering and its application, Paris, Nov 1993.

[DFS93] C.Debou, N. Fuchs, H. Saria, Selling believable technology, IEEE Software, Nov 1993.

[DKR94] Debou C, Kuntzmann-Combelles, Rowe A.: A quantitative approach to software process, In: Proceedings of the 2nd international symposium on software metrics, Oct 1994, London.

[DLP92] Debou C, Lipták J, Pescoller L: Managing Software Process by Applying ami. In: Proceedings of the MSP-92 IFAC - annual review in Automatic programming, volume 16, Graz, May 1992.

[DLS93] Debou C, Lipták J, Shippers H.: Decision making for software process improvement: a quantitative approach, In: Proceedings of the 2nd internation conference on "achieving quality in software" ACQUIS 93, Venice (Italy), pp 363-377, Oct 1993.

Also In: The Journal of Systems and Software, 1994; 26:43-52, Elsevier Science Inc.

[DPF92] Debou, C., Pescoller, L. and Fuchs, N.: Software Measurements on telecom systems - Success stories ? : Proceedings of the 3rd European conference on software quality, Madrid, November 1992.

[Dem86] Demin W.E., Out of the Crisis, MIT Press

[FoR90] Fowler, P. and Rifkin, S., *Software Engineering Process Group Guide*, CMU/SEI-90-TR-24, September.

[HMK94] Haasse V, Messnarz R, Koch G., Kugler H, Decrinis P: Bootstrap: Fine-Tuning Process Assessment, In: IEEE Software pp25-35, July 1994

[Hum89] Humphrey, W.: Managing the Software Process, Addison-Wesley, Reading, Mass., 1989.

[HuS87] Humphrey W.S., Sweet W.L.: A Method for Assessing the Software Engineering Capability of Contractors, Software Engineering Institute, Sept 1987

[MKH94] Messnarz R, H-J. Kugler, Haase V.: BOOTSTRAP and ISO 9000: A Quantitative Approach to Objective Quality Management, In: Proceedings of the first ISCN Seminar, Dublin, May1994

[PAG94] SPICE Consortium, The Process Assessment Guide, Draft

[PeR94] Perterson B, Radice R.: IDEAL: an integrared approach to software process improvement (SPI), SEI Symposium, Pittsburgh, August 1994.

[PFS92] I. Perez, P. Ferrer, A. Fernandez: Application of Metrics in Industry : Proceedings of the 3rd European conference on software quality, Madrid, November 1992.

[PIG94] SPICE consortium: The process improvement guide, issue 0.05, Draft, October 1994

[RoW94] Rowe A., Whitty R.: ami: a Quantitative Approach to Software Management, In: Software Quality Journal 2, 291-296, 1994

[WNH94] Whitney, R., Nawrocky, E., Hayes, W., Siegel, J. (1994) Interim Profile: Development and Trial of a Method to Rapidly Measure Software Engineering maturity Status, CMU/SEI-94-TR-4, March.

[1] send subscription to ami-request@aut.alcatel.at

TickIT: Providing Confidence in ISO9001 Certification

Alec Dorling
European Software Institute
Parque Tecnologico de Zamudio
Building 204
E-48016 Bilbao
Vizcaya
Spain

In July 1990 the UK Department of Trade launched the TickIT initiative which provides an accredited certification scheme for quality management systems specially designed for the needs of the software industry. TickIT is a means of providing confidence in ISO9001 certification. This paper traces the events that lead up to the initiative, outlines the features of the scheme which provide that confidence, discusses some of the outstanding issues concerning TickIT and looks forward to the proposed international arrangement for IT quality system certification.

1. Introduction

TickIT is a quality initiative aimed at software and the IT industry. It was originated by UK industry and sponsored by the UK Department of Trade and Industry (DTI). It is now administered by DISC, that part of the British Standards Institution (BSI) responsible for standards relevant to Information Systems.

At the heart of the initiative is an accredited certification scheme for Quality Management Systems (QMS) specially designed to meet the needs of the software industry. TickIT certification is available to suppliers of software-based systems and to in-house developers who meet the requirements of the international standard ISO9001 and its European equivalent EN29001.

Certification is conducted by independent third party certification bodies using technically qualified TickIT auditors who have undergone specialist training and who are subject to rigorous professional vetting. To ensure the certification bodies are fully competent they are accredited by the National Accreditation Council for Certification Bodies (NACCB) on behalf of the Secretary of State for Trade and Industry.

2. The TickIT Initiative

In 1979, the BSI published its BS5750 series of generic quality system standards deriving their content from existing military standards. In 1987, the International Standards Organisation (ISO) published its ISO9000 series of standards for quality systems and quality assurance which were re-adopted by the BSI as BS5750:1987.

[Note: The ISO9000 series of standards have undergone a further revision in 1994. The International, European and British standards now have a similar numbering, BS5750 having been dropped. ISO9001, EN29001 and BS EN ISO9001 are all identical.]

These were not the only quality standards around. NATO for instance had developed AQAP-1 and in 1984 published AQAP-13 to supplement and support the requirements of AQAP-1 to the design and development of software. The nuclear power industry also had its own quality assurance standard BS5882.

In order to establish the relevance of all these standards to the production of software, the UK DTI in 1987 commissioned two complementary studies:

- Quality Management Standards for Software (Logica)
- Software Quality Standards: The Costs and Benefits (Price Waterhouse)

The studies undertook extensive research into the respective subjects and included a broad consultative process with users, suppliers, in-house developers and purchasers. The reports whilst evaluating different problems, arrived at similar conclusions and made a number of recommendations to the DTI which included:

- all quality Management System (QMS) standards in common use were generically very similar and that the best harmonisation route was through ISO9001
- action was required to improve market confidence in third party certification of QMSs, and there was an urgent need to establish an accredited certification body or bodies for the software sector
- before progress could be made guidance material was required to assist QMS implementors in relating generic ISO9001 requirements to specific software QMS procedures and to facilitate common interpretation by auditors
- work was required to improve professional practice amongst software QMS auditors.

These principal recommendations were accepted by the DTI and culminated in the launch of TickIT in July 1990. TickIT advocated a fresh approach to software QMS certification with the IT professionals, professional bodies and the main players coming together to underpin its success and credibility.

For many the TickIT certification is considered a higher grade of certificate than previously seen, since those within the scheme have to conform to uniform accreditation arrangements beyond those required for normal accreditation and certification.

Underpinning TickIT is an overall infrastructure consisting of three main elements. The three main elements of the infrastructure are:

- TickIT Uniform Accreditation Criteria
- TickIT Guide
- TickIT Auditor Registration and Training

3. TickIT Uniform Accreditation Criteria

In accrediting certification bodies to provide TickIT certification a number of topics have been embodied into uniform accreditation arrangements to ensure a common approach is applied. These arrangements provide confidence to both purchasers and users and meet the

wider expectations of the software community. The TickIT uniform accreditation arrangements include:

- the use of the TickIT name and logo
- three year certification cycle
- monitoring the applicants management review activities
- mutual recognition of TickIT certification
- use of TickIT guide
- use of TickIT auditors

The use of the TickIT logo infers accreditation. The name and logo are protected and can only be used to indicate accredited certification.

Certification under TickIT arrangements follows a three year cycle comprising initial assessment and surveillance visits followed by re-assessment and re-certification every three years. This was deemed to be necessary due to the changing nature of the industry.

As part of the surveillance visits the certification body must at least annually carry out a complete evaluation of the applicants management review activities which is a key activity in the operation of any quality management system.

Under the arrangement each certification body must give mutual recognition to any TickIT certification held.

All certification bodies offering TickIT certification must use the TickIT guide as its reference document excepting those requirements that relate directly to the standard itself.

Auditors carrying out TickIT certification and surveillance must meet the auditor performance standards set out in the TickIT guide.

There are currently seven accredited TickIT certification bodies:

- BMT Quality Assessors Ltd
- BSI Quality Assurance Ltd
- Bureau Veritas Quality International Ltd
- Det Norske Veritas Quality Assurance
- Electricity Association Quality Assurance
- Lloyd's Register Quality Assurance
- SGS Yarsley Quality Assured Firms

In the UK, certified companies are listed in the DTI Register of Assessed Companies. The DISC TickIT office also maintains a list of TickIT certificated companies. Certification bodies maintain and publish their own lists.

It is important to note that certifications are carried out against scopes which may vary considerable. It is always important therefore for a potential purchaser to look closely at the defined scope of certification.

The general scope of the TickIT scheme includes:

- software development
- software replication
- computer operations
- system integration
- peripheral services
- software development subcontracting

but does not include:

- software stockholding (warehousing)
- software sales (over the counter, mail-order).

4. The TickIT Guide

The TickIT Guide to Software Quality Management System Construction and Certification using EN29001 was produced by IT professionals. It is a comprehensive document which includes the authoritative international standardisation guide ISO9000-3 Guidelines for the Application of ISO9001 to the Development, Supply and Maintenance of Software which explains what ISO9001 means in the context of information systems supply.

In producing the guide, use of existing guidance material was made where possible, editing and updating it for TickIT purposes. There are therefore many sources from which the original material was taken.

The TickIT Guide comprises five sections covering:

- Introduction
- Application of ISO9001 to software (i.e. ISO9000-3)
- Purchasers guide
- Suppliers guide
- Auditors guide

Each section is structured in a different way to make it easy for appropriate guidance to be made available to all the parties involved in quality management work.

Section 2 of the TickIT Guide is a reprint of ISO9000-3 'Guide for the Application of ISO9001 to the Development, Supply and Maintenance of Software'. This presents the authoritative guidance at the ISO level. Unfortunately in the updating of the planned revised TickIT Guide the reprint of ISO9000-3 will be absent due to internal wrangling within BSI and DISC over publication revenues. ISO9000-3 has also not yet been revised at the ISO level to be in line with ISO9001:1994 and is therefore inconsistent with the rest of the proposed revised TickIT Guide.

Section 3, the Purchaser's guide, sets out an explanation of the purchasers' expectations of a supplier's QMS assessed and certified to ISO9001. Most of the material originated from earlier work undertaken by the National Computing Centre (NCC) on behalf of the DTI through the STARTS initiative.

Section 4, the Supplier's guide, provides guidance to suppliers and in-house developers implementing QMSs for compliance to ISO9001. It is structured in a way that is easy to understand from a supplier's point of view defining relevant quality system and quality control elements that should be in place. Each element describes an activity definition, objectives and criteria, outputs, standards and procedures and control mechanisms. Most of this material originated again from the STARTS initiative and guidance material provided by the Computing Services Association (CSA) now renamed the CSSA in a working paper.

Section 5, the Auditor's guide, is a reprint of the European IT Quality System Auditor's Guide produced by the ITQS, the European Agreement Group on IT Quality System Certification. It is structured according to the clauses of ISO9001 and provides a common basis for enabling auditor consistency and depth of enquiry when auditing against the requirements of ISO9001.

Additionally, in an Appendix to the Guide, the professional attributes/performance standards for software quality management auditors are described. These were developed to be aligned to the British Computer Society's (BCS) Industry Structured Model (ISM) which defines a set of personal performance standards designed to cover the skills of all staff within the broad spectrum of information technology.

The current TickIT Guide (version 2) has been under revision for some time to incorporate changes necessary to align with the new revision of ISO9001 in 1994 and to encapsulate auditor experience. It is due to be published in the 4th quarter of 1995.

5. TickIT Auditor Registration and Training

The TickIT scheme has emphasized the confidence placed in the qualification of software auditors and the strict rules it applies during the interview process which is required for auditors wishing to register as TickIT auditors.

The auditor performance standards defined in the TickIT Guide are use to screen applicants wishing to practice and register as software quality auditors. TickIT auditor registration is recorded in International Register of Certified Auditors IRCA) which is administered and maintained by the Institute of Quality Assurance (IQA) in the UK. A joint panel of IQA and BCS scrutinize applications and carry out the interview process.

There are three levels of software auditor defined:

- Lead TickIT auditor
- Senior TickIT auditor
- Provisional TickIT auditor

All applicants for TickIT auditor registration must be experienced computer professionals. This experience must include appropriate training in informatics, software systems development and formal training in audit procedures.

Specifically auditors must have attended a 5 day TickIT auditors course and have passed the examination set. Training courses must meet the syllabus defined by the registration authority and training organizations must themselves be accredited to provide training according to defined procedures.

Before registration all candidates are required to conform formally their willingness to observe and be bound by a strict code of conduct.

6. TickIT Products

TickIT is more than a certification scheme, it is an overall initiative with supporting products and marketing aimed at stimulating developers to think about what quality is and how it may be achieved. TickIT and quality management systems are promoted in the context of Total Quality Management (TQM).

The overall initiative is supported by the DISC TickIT Office from where all products can be purchased and further information sought.

TickIT products consist of:

- TickIT brochure (free)
- TickIT Guide
- TickIT Auditor's Training Course material
- TickIT video (Just the TickIT)
- TickIT case studies
- TickIT conferences and seminars
- TickIT annual awards
- TickIT international news (subscription based)

The contact address of the TickIT Office is:

DISC TickIT Office
389 Chiswick High Road
London W4 4AL
England

Telephone +44 171 602 8536
Facsimile +44 171 602 8912

7. TickIT to Success

TickIT has been voted a resounding success by the UK software industry as reported by a survey of the Computing Software and Services Association (CSSA) in February 1994 which listed the benefits gained as follows in order of importance:

- assessment is now undertaken by people who understand software

- customers have increased confidence in the quality of our products
- the number of second party audits has been cut dramatically
- software developers can relate to a common set of guidance contained in the TickIT guide
- UK software quality is improving, those who have implemented TickIT relate stories of improvements generated in their organizations

Improvements have all been made without any significant increase in the cost of certification. Now organizations are reaching their three year re-assessment period, companies are now starting to ask the question 'what about life after TickIT'. These organizations are starting to take a great interest in process assessment through the SPICE project as a means to continually improve their processes matched to there business needs.

8. TickIT Issues

While TickIT has been a success in the UK, it has faced some problems which are now being addressed both by the TickIT scheme locally and in a wider context through the international community in the developing international arrangement for IT quality system certification. These problems have included:

- UK certification bodies only being able to be accredited for TickIT
- the TickIT logo refering to EN29001
- confusion over the use of a separate TickIT logo
- lack of accountability of the TickIT scheme
- coverage of TickIT Guide
- internationalization of TickIT

Accreditation of TickIT certification bodies is carried out by NACCB which until very recently was only able to accredit UK based certification bodies. This provided a situation where foreign certification bodies whilst having all the necessary credentials could not be TickIT accredited. This was seen as a barrier to free trade.

The TickIT logo was designed with EN29001 as the reference standard due to political reasons as the European Community had at the time a project underway called ITQS which is an agreement group of certification bodies in Europe. TickIT wanted to be seen as harmonious with its European counterparts. The logo has now changed to reference ISO9001.

There has been confusion in the market place over the use of a separate logo for the IT sector. If this were to promulgate to other sectors certificates might have as many as ten logos.

Until late 1994 there had been a lack of accountability of the TickIT scheme. This has now been addresses and an infrastructure is now in place to manage all parts of the scheme.

The TickIT guidance was produced mainly for the software development community. It has lacked guidance on such issues as system integration, software product support and facilities management. The TickIT guide is currently under revision and a new version published later this year.

The lack of access to the scheme by foreign certification bodies and auditors has meant that the scheme has been seen as a UK scheme and not a world-wide scheme. Nevertheless it has been adopted in principle and modeled in several other countries. In Europe however there has always been conflict, although friendly cooperation between TickIT and ITQS. This has resulted in moves towards the internationalization of a software sector quality system certification scheme.

9. International Arrangement for IT Quality System Certification

As goods and services are purchased in global markets there is a clear need to recognize the credentials of both certificated organizations and certification bodies across national boundaries. The success of the TickIT scheme and the influence of ITQS in the main, together with support from the United States through the ASQC Software Division, has led to a proposal for an International Arrangement for IT Quality System Certification (Arrangement).

It is proposed that such an Arrangement be driven by the demands of its customers and users. The first international forum will be held in the Hague in September 1995 to progress such an Arrangement forward.

The essential features of an international Arrangement are perceived to include:

- an international Steering Group to ensure that the certification process meets the requirements of buyers and suppliers worldwide
- participating bodies accredited to the requirements of standards for certification body conduct
- consistent application of ISO9001/2 referring to best practice standards and guidelines such as ISO9000-3
- use of a common logo to indicate adherence to the arrangement
- scope of Arrangement to be processes delivering products or services whose provision depends on the application of software skills
- maximization of user choice in the selection of a certification body, by ensuring equivalence of certificates issued by participating certification bodies
- harmonization of audit and certification practices
- a register of suppliers certificated under the rules of the schemes
- a feedback mechanism for continuous improvement which will monitor how effectively the Arrangement is meeting industry needs
- rigorous requirements and procedures for auditor qualification to ensure adequate professional knowledge, training and experience of auditors

Whether such an Arrangement can deliver these features within an international context remains to be seen. Within this context, the European Software Institute (ESI), with international support, aims to move forward to develop the criteria for software auditor qualification and a common training syllabus for auditor training to ensure that the industry can have confidence in the professional knowledge, training and experience of auditors.

10. Conclusions

TickIT as a UK initiative has had tremendous success in promoting the benefits of quality management certification to UK companies and in providing confidence in the ISO9001 certification process as applied to the software industry.

The interest in TickIT has spread to all four corners of the world. There are now over 800 TickIT certificated companies (approximately 20% of these outside the UK) with over 30,000 TickIT Guides distributed to 20,000 contacts in 50 countries.

TickIT however has not been without problems, which have mainly arisen out of its visibility and success. One of the outstanding problems is raising and gaining acceptance of the principles of TickIT within an international context. This is being achieved through the proposed International Arrangement for IT Quality System Certification.

References

1. TickIT Guide, Guide to Software Quality Management System Construction and Certification Using EN29001, Issue 2.0, DISC TickIT Office, 1992.

2. Application of the TickIT Scheme, A Concise Definition, Issue 1.0, DISC TickIT Office, 1993

3. CSA Position Paper, TickIT provides proven quality benefits to both customers and suppliers, CSSA, 1994

4. TickIT Certification: A Passport to International Trade, John Slater, Logica

5. Logica Report, Quality Management Standards for Software, UK Department of Trade and Industry, 1988

6. Price Waterhouse Report, Software Quality Standards: The Costs and Benefits, UK Department of Trade and Industry, 1988

7. ISO9001:1987 Quality Systems, Model for Quality Assurance in Design/Development, Production, Installation and Servicing, International Standards Organisation, 1987

8. ISO9000-3:1991 Quality Management and Quality Assurance Standards - Part 3. Guidelines for the Application of ISO9001 to the Development, Supply and Maintenance of Software, International Standards Organisation, 1991

9. Proposal, International Arrangement for IT Quality System Certification, Task Force, 1994

Process Improvement - How To Measure It

Nicholas Ashley
Principal Consultant

Mike Kelly
Technical Director

BRAMEUR Ltd.
Hampshire, UK

Abstract

This paper presents guidelines for setting up and monitoring a Process Improvement programme. It is not a complete 'do-it-yourself' manual - that would require a number of books! It is however intended to help anyone starting out in this domain, to understand what is required and to save some time in the early stages of programme implementation. The paper presents an overview of the theoretical foundation for a Process Improvement programme followed by guidelines on what is required for the programme framework and establishing the infrastructure, the latter concentrating on the measurement aspects. We also discuss briefly the human factors issues and finish with a look at some of the lessons learnt from actual Process Improvement and measurement programme implementations.

The paper is based primarily on the experience of Nicholas Ashley on two major measurement programme implementations supporting organisational process improvement plans, these being the IBM Tax Project for the Royal Thai Government and the System Development department of a major UK finance house. In addition the experiences of both authors on other measurement programmes have been drawn upon as well as those of the CEC funded MAST project to which both authors have contributed. All these projects used METKIT material as the basis of both programme planning and staff training.

Table of Contents

1 Introduction

> *Things can only get better, unless of course they get worse.*
>
> **Joseph Heller**

Process Improvement (PI)

Today, the main challenge facing managers is to optimise their development process by creating a development environment that enables project teams to work effectively.

Main Goal

This document indicates the main steps in how to create and sustain a development environment that:

- Enables project teams to work effectively to meet the business objectives
- is flexible enough to be adapted to the changing needs of its surrounding environment.

It provides a blueprint to enable a department to set up, run and sustain a Process Improvement programme.

Questions Answered

This document looks at the following questions:

- What is Process Improvement?
- Why is it necessary?
- What will it do for us?
- Will it work for us?
- How do we do it?

We do not address specifically the questions:

- How much will it cost?
- Has it been shown to work in other organisations?

These are, however, extremely important questions for anyone wanting to undertake the Process Improvement journey and they are addressed briefly in the accompanying tutorial.

Principles Of Process Improvement

- Every process can be measured, controlled and improved
- It is no point introducing changes unless you have the capability to measure their impact.
- Changes to a process should take into account its capability
- Process Improvement is a continuous activity
- Unless a mechanism to sustain a culture of continuous Process Improvement is implemented, a Process Improvement programme will fail.
- Identify problems and prioritise them, don't tackle everything at once.
- Process Improvement requires an integrated approach encapsulating the process itself, the culture of the organisation and people issues.

Integrated Approach

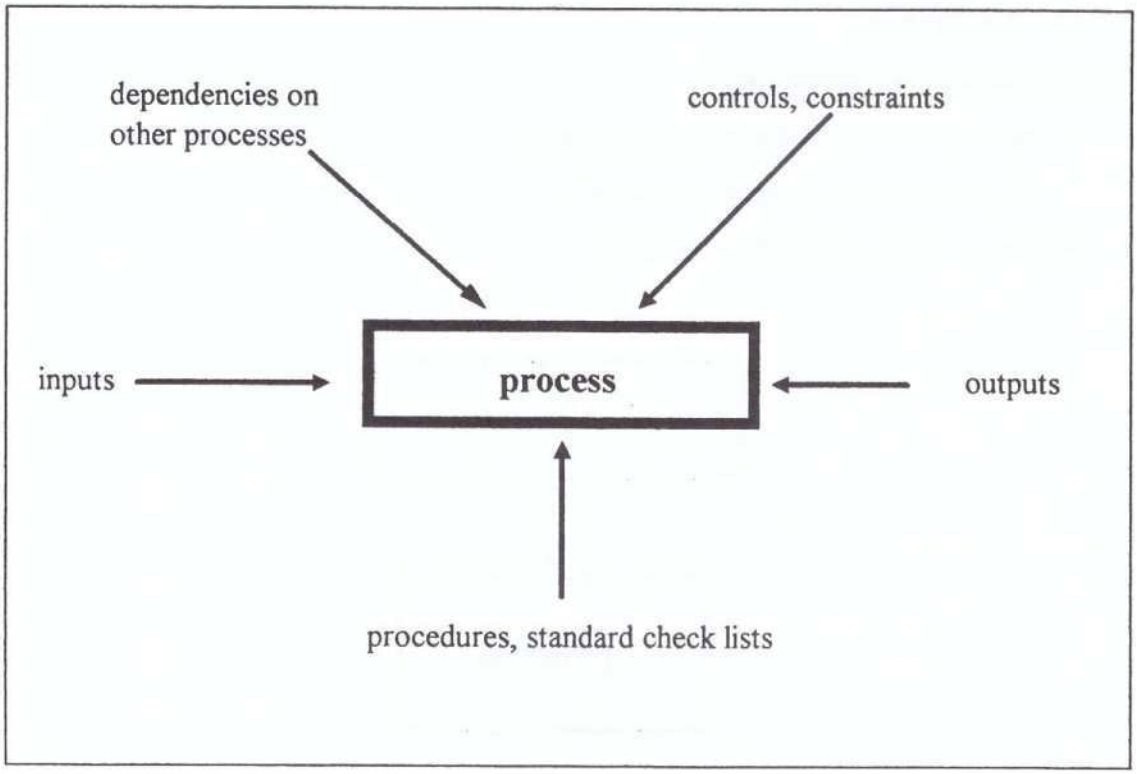


Figure 1 - Integrated Approach

Process Improvement is not simply about changing a process. It requires an integrated approach encapsulating the following:

- Changing the culture of the organisation to one where staff are proactive in suggesting changes and staff see the commitment of management
- People issues
  - personalities
  - skills

  - attitudes
  - anxieties
  - training
- Improving or re-engineering the process model.

Unless all three issues are addressed then a Process Improvement programme is almost certainly doomed to failure.

Our approach is based on a proved method which is flexible enough to be applied to any process. It is a combination of both incremental improvements together with re-engineering.

2 Theoretical Foundations

In this section we outline the theoretical foundations underpinning Process Improvement. First we discuss what we mean by a process.

What Is A Process?

Every task can be modelled as:

- the resources needed to perform the task
- the process used to carry out the task
- the products produced by the task

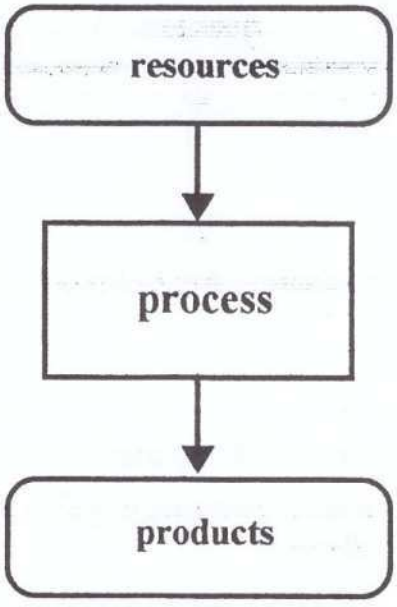


Figure 2 - High Level Process Model

This applies to any task such as writing a report, conducting a review or transforming a software specification into a software design. The high level qualitative goals of any organisation are to:

- make better use of their resources
- improve the efficiency of their processes
- improve the quality of their products.

It should be noted that a process may consist of a number of sub-processes. For example, the overall software development process can be considered as a collection of sub-processes, such as requirements capture, planning and estimating, design, reviews, etc.

What Is Process Capability?

A capable process is a process that produces products that satisfy the customer's requirements. Achieving process capability cannot be viewed in isolation when that process is an integral part of a larger process. If a target for a process is set that is beyond its normal capability, the only way to achieve it is to distort the process in such a way as to cause trouble in other related processes. For example, suppose management has set the following unrealistic target for the accuracy of the effort to develop software systems.

> For at least 95% of the projects, produce estimates of an effort from the requirements that are correct to within 5% of the actual effort expended

Management pressure to meet this unrealistic target may be so great that project managers are forced to reduce the effort spent on reviews and testing. The consequences are almost certain to have an adverse effect on the number of operational failures.

Therefore it is important that any quantifiable goals for a process should be challenging, yet achievable. This can only come about from validated data from the process itself, not from the unrealistic wishes of management.

What States Can A Process Be In?

A process can be in one or four states, see figure 1, namely:

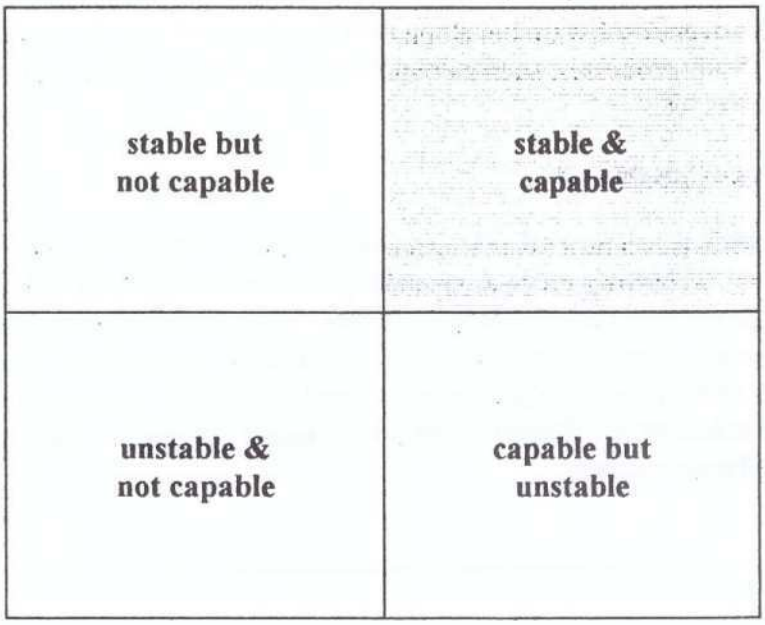


Figure 3 - States Of A Process

State 1 - Threshold: Stable But Not Capable

The process is stable but not fully capable. Management action is required to change the process to reduce common causes of variation.

State 2 - Ideal: Stable & Capable

The process is stable and capable. In other words the process is in control and the output conforms to the requirements.

State 3 Brink of Chaos: Capable But Not Stable

The process is capable but not stable. In other words the process is out of control although it has been producing outputs that conform to the requirements. In a capable but unstable process, quality and productivity can deteriorate rapidly.

State 4 - Chaos: Unstable & Not Capable

The process is unstable and not capable. In other words it is not possible to predict future behaviour and it is producing outputs that do not conform to the requirements. The process is susceptible to random fluctuations. In this state it is not possible to predict the future behaviour of the process. The first step out of this chaotic state is to identify and eliminate those special causes that are having a detrimental effect on the process.

What Is A Process Improvement Cycle?

Essentially, a Process Improvement cycle consists of the following:

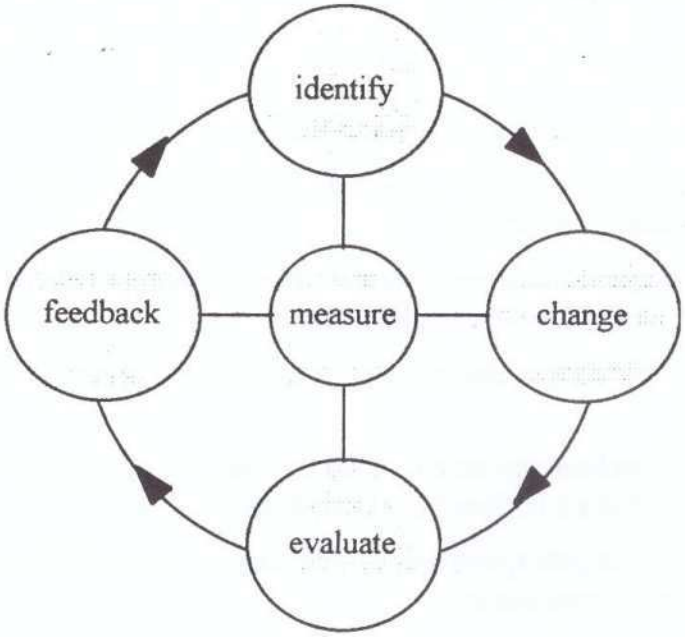


Figure 4 - Process Improvement Cycle

identify	First, use a measurement to establish the effectiveness of each key process, to identify where the main problems are and which processes need improving. Then determine the causes of the problems before introducing changes to try to improve the situation. For example, management may be worried about productivity because they think too much effort is being spent on rework
change	Make changes. Don't try to do everything at once, implement the highest priority changes first
evaluate	Assess the impact of the changes.
feedback	Use the results of the evaluation to inform managers and project teams. Now start all over again because Process Improvement is a continuous activity.

Incremental Improvement Versus Re-Engineering

We now look briefly at the difference between how to determine when a process needs to be improved via incremental improvements and when its needs to be re-engineered using a radically different process.

Incremental Approach

Improving a process incrementally requires the following steps:

- Define a set of Key performance Indicators that can be used to asses the performance of the process.
- Identify changes to the processes that need to be implemented in order to create an effective development environment.
- Identify people issues that need to be addressed in order to support the proposed changes.
- Establish baselines and set quantifiable targets for the Key Performance Indicators that are challenging yet achievable.
- Validate the targets against budget, timescales, the capability of the process and changes to be implemented
- Provide a step-by-step approach on how to implement the changes.

Re-Engineering Approach

The re-engineering approach on the other hand looks at the software development process as a whole by firstlt developing a process model. Once a model has been developed it is validated by performing the following analysis:

- Is each activity necessary or can it be eliminated?
- Can the activity be combined with another?
- Are the activities in the proper order?

What Is Process Modeling

Process modeling refers to the practice of explicitly representing the constituents of a process. There are explicit and implicit components of a process. Until now when people talk about process modeling it is mostly to do with the explicit, formally representable aspects of a process. The tacit or implicit (and perhaps the more important) aspects of the process have been largely ignored. The explicit components of a process are:

- activities that go to make up the process
- agents, roles and responsibilities
- feedback loops
- document flows and stores
- object/data entry

- control and management information flows
- process flows
- triggering and terminating conditions for each process step
- rules or conditions for carrying out each of the actions of the process

Process modeling also includes the determination of the quality assurance practices, standards affecting some aspects of the process, and the tools and methods involved in the process.

The tacit aspects of a process are the:

- socio-cultural practices in the organisation which affect how the process is really performed
- needs and anxieties of the process owners and other stakeholders that also affect how the process is carried out
- institutional / organisational variables that affect the process
- external factors that affect the organisation and how the process is performed

Processes can be modelled either textually or diagrammatically. The diagrammatic or graphical format has come to be synonymous with process modelling. There are many graphical formats for carrying out process , such as:

- Data Flow Diagrams (DFD)
- Entity relationship Diagrams (ERD)
- Role Activity Diagrams (RAD)
- Structured Systems Analysis (SSA) Diagrams

3 Framework For A Process Improvement Programme

In this section we provide an outline blueprint for setting up and running a Process Improvement programme. Before embarking on a Process Improvement programme you need to answer five fundamental questions:

Where are we ?

Where do we want to go ?

How do we get there ?

How do we know if we have got to where we want to go ?

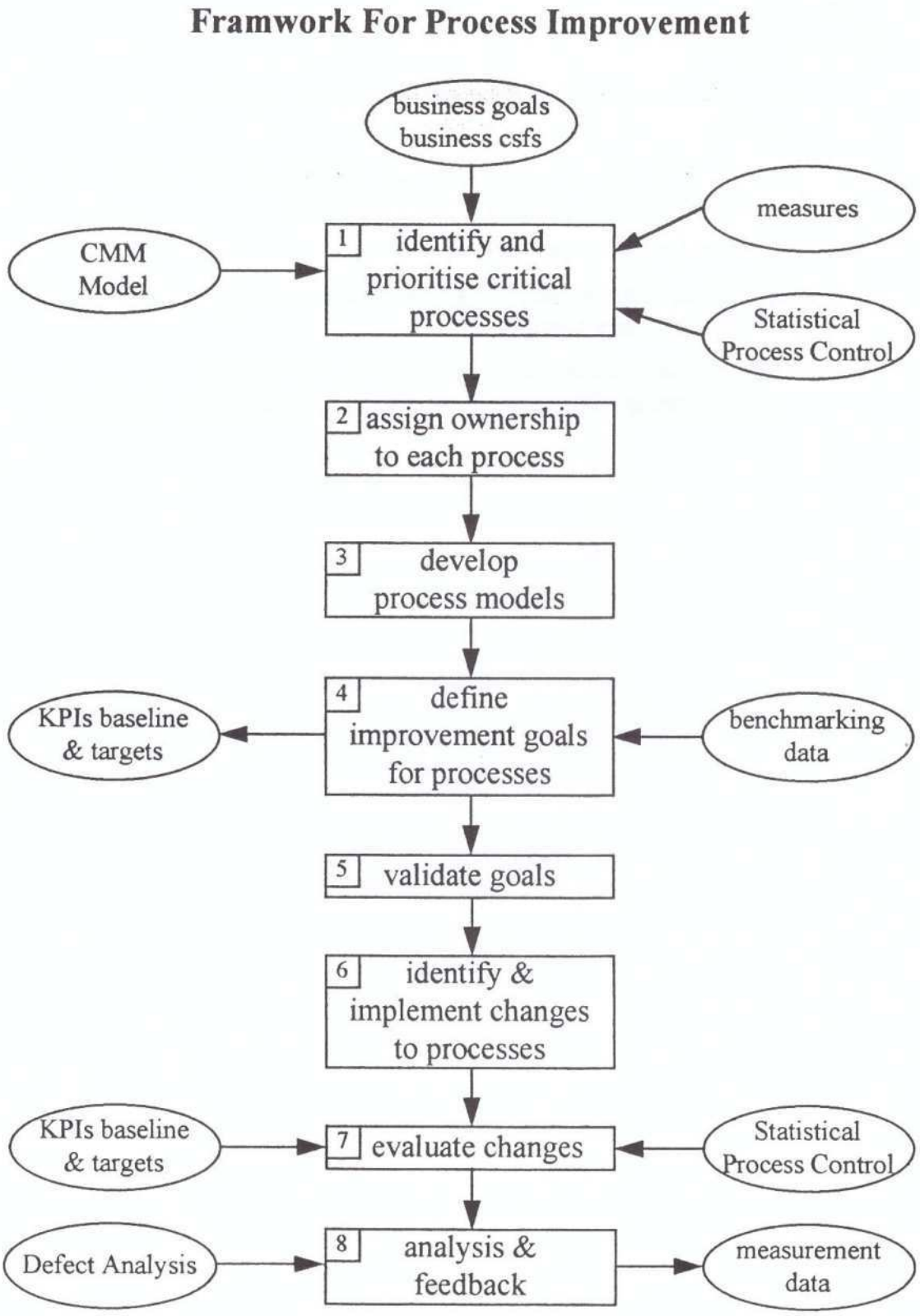
How do we compare against the competition ?

In order to answer these question you need to address the following issues:

- Are the goals of the Process Improvement programme aligned to our business goals and stated in quantifiable terms?
- Have we put into place all the infrastructure to support the Process Improvement programme
- Do the staff know why we are running the Process Improvement programme and do they know what is expected of them?
- What type of training should our staff be given?
- Are we able to measure the success of the Process Improvement programme?
- How do we define and validate quantifiable goals for a process?
- How do we define a set of Key Performance Indicators to assess the capability of a process?
- How do we set and validate targets for a process?
- How do we develop a model of an improvement programme for a process to enable an organisation to determine where it needs to go and how to get to where it wants to go?

These questions, and others like them, are all too often asked by most teams faced with the challenge of setting up and running a Process Improvement Programme. The framework for setting up a Process Improvement programme is illustrated in figure 5.

Figure 5 - Framework for Process Improvement



4 Infrastructure To Support A Process Improvement Programme

In this section we outline the infrastructure needed to support a Process Improvement programme.

The key stages are:

- Set up a Measurement Programme to support a Process Improvement programme
- Use Statistical Process Control to provide an objective means of improving the effectiveness and quality of a development process
- Use Defect Analysis as an improvement tool
- Assess the capability of each of the main processes using the approach adopted by the SEI Capability Maturity Model
- Identify and carry out the training that needs to be provided to both staff tasked with implementing a Process Improvement programme and project teams.

5 Management of Change

In this section we briefly discuss management of change - a crucial aspect of any Process Improvement programme.

In general, people try to resist change but adapt very quickly once they are convinced of the benefits of the change. When introducing a PI programme there are four main states that people go through.

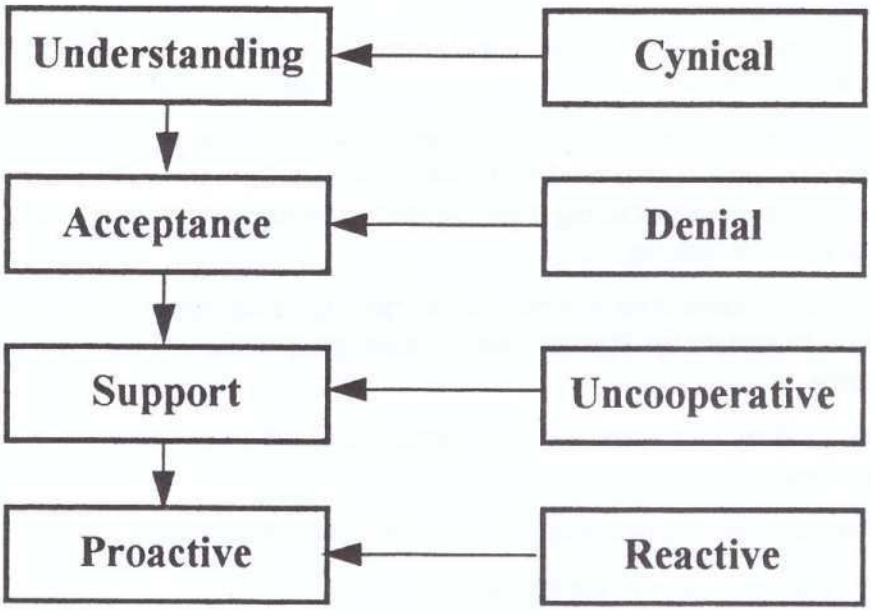


Figure 6 - Attitude Transition States

Firstly, understanding why the Process Improvement programme is necessary and its associated benefits. This may involve convincing people to suspend their disbeliefs.

Secondly, accepting that the Process Improvement programme is necessary and that it has been shown to work for other organisations. This will involve answering the question :

What's in it for me?

Thirdly, providing support, whether this is total support or partial support in the sense of - *I'll provide initial support, to enable you to convince me that it works.* In other words obtaining buy-in from managers and project teams.

Finally, the fourth state is where managers and project teams become proactive.

It is important to recognise that these states exist and to address them, otherwise the Process Improvement programme is unlikely to succeed.

During the first three stages you will inevitably encounter :

- denial that there is a problem
- resistance from cynics who are either openly hostile or are unconvinced by the benefits.

Cultural Issues

In this final section we discuss briefly the need to investigate a culture change to support the Process Improvement programme where PI becomes an integral part of the development culture.

Firstly Management need to ensure that project teams have a positive attitude towards continuous Process Improvement and recognise it is the responsibility of all the staff.

The greatest challenge with introducing a culture change is not in getting it accepted, but in sustaining momentum to support the culture change after the initial wave of enthusiasm has worn off. This then leads onto the question of how do you hold onto gains that are made and build on these gains to become even better.

There is no magic formula to becoming the best and remaining there. Instead there are fundamental principles that need to be continuously adhered to. The keys to helping to meet the challenge of sustaining a culture change to support an Process Improvement programme are as follows :

- Choose a leader who is committed to the culture change and has the dynamic drive to sustain the Process Improvement programme and motivate project teams.
- Set challenging targets that are attainable and not so unrealistic as to be de-motivating.
- Provide feedback to employees to show them what they have achieved
- Provide incentives to the employees
- Provide recognition of achievement
- Provide training to the workforce to help them become more effective
- Ensure that improvements are not just helping profits but benefiting all the employees by creating a better working environment that enables them to work more effectively.
- Create a company culture where Process Improvement is at the forefront of every employees perception of the company and where employees are :
  - given the opportunity to suggest improvements
  - listened to when they give advice on how they think something can be improved.

Finally, what happens when you achieve your ultimate goal of becoming the best of the best ? You will then have companies benchmarking against you, trying to oust you from your number one position. Sweet dreams are made of this.

6 Case Study experience

The recommendations in this paper have been developed based on the experience of the authors on a number of projects to implement measurement programmes in a variety of organisations as well as feedback from the activities of the CEC funded MAST project which is currently conducting technology transfer in software measurement based on the METKIT material.

The main projects used as the basis for this paper were:

- The IBM Tax system for the Royal Thai Government
- Measurement programme for a UK Assurance Company systems department
- Measurement programme for a UK bank
- Process Improvement programme for a UK bank IS department

These cases varied considerably in the levels of investment, level of external support used and extent of implementation, although in all cases the programmes implemented are still running and being improved.

The preceding sections of this paper indicate the steps and measures implemented in the case studies. In this section, however, we provide a brief summary of some of the lessons learnt from our experience of working with these organisations.

First as you can see there is a heavy emphasis on work with the financial market sector. It is our experience that although this sector may not be at the forefront of innovative technology, they are possibly the leading sector when it comes to wanting to improve and having sufficient budget and management commitment to do something about it.

In all cases management commitment has been strong and continuous - on those few projects we have been involved with where senior management commitment did not last or was spasmodic, the programmes invariably failed.

On one project, currently in progress, the organisation has invested heavily in process improvement and uses a considerable amount of external consultancy. The early lessons from this project were that:

- Good planning is essential - everyone must know what they are doing and the limits of their responsibility.
- Constant project management is vital
- Constant awareness of progress to all concerned (and in this case this meant 150 people) must be maintained
- Get a configuration management system in place as soon as possible

The actual processes, metrics, etc. while important would have been wasted without attention to these 4 points. With respect to configuration management, a good, well planned and automated system can significantly shorten timescales for improvement and in extreme cases will substitute for poor quality systems! For example, on a major project in one defence sector organisation improvement, while slow, was achieved with a good CM system and one which covered all the cracks, deficiencies and plain bad practise associated with their QMS, so much so that they continually obtained 9001 certification with no major non-conformance's.

Improving Software Quality through Quantitative Evaluation of Products and Processes

Gualtiero Bazzana*
Massimo Giunchi
Etnoteam S.p.A.
Via A. Bono Cairoli, 34
20127 - MILANO (ITALY)

G. Ru
Italtel SIT BUCT,
Via G. Reiss Romoli,
20019 Castelletto di Settimo
Milanese, (ITALY)

S. Scotto di Vettimo
Siemens Telecomunicazioni
Italia
SS Padana Superiore,
Cassina de Pecchi (ITALY)

This paper presents pragmatic approaches and experiences aiming at software product quality improvement based on quantitative measures.

Starting from the knowledge matured in the SCOPE Project, the paper presents the fundamentals of measurement-based SW product quality improvement and two case studies from the telecom application domain.

The former focuses on the improvement in maintainability reached by a data collection and analysis campaign adopting source code static analysis techniques.

The latter describes the improvement in efficiency derived from a simple yet profitable statistical analysis of the operational profile of the systems installed in field.

Last but not least, the paper gives hints on the relationship between process quality and product quality, in terms of:

- impact of process improvement practices on the quality of SW products, as derived from an European-wide survey;
- a method for tracking the net contribution of process improvement on product quality and business goals.

1. Software product quality evaluation and the ISO 9126 standard

The ISO/IEC 9126 is the result of the joint committee of ISO and IEC (International Electrotechnical Commission) and is published with the title "Information technology - Software product evaluation - Quality characteristics and guidelines for their use" [ISO 9126].

This standard gives a definition of software quality in terms of factors and ought to represent the end of a long lasting discussion on this subject ([GILL 92], [ARTH 85], [BOWE 85], [BOEH 78], [DEUT 88], [FORS 89], [GRAD 92] and [VMAR 90]). Its main content is the representation of quality of software as seen by software users. Six characteristics are defined in the standard as the building blocks of software product quality. They are:

- Functionality, "a set of attributes that bears on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs";
- Reliability, "a set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time";
- Usability, "a set of attributes that bear on the effort needed for use, and on the individual evaluation of such use, by stated or implied set of users";
- Efficiency, "a set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions";

- Maintainability, "a set of attributes that bear on the effort needed to make specified modifications";
- Portability, "a set of attributes that bear on the ability of software to be transferred from one environment to another".

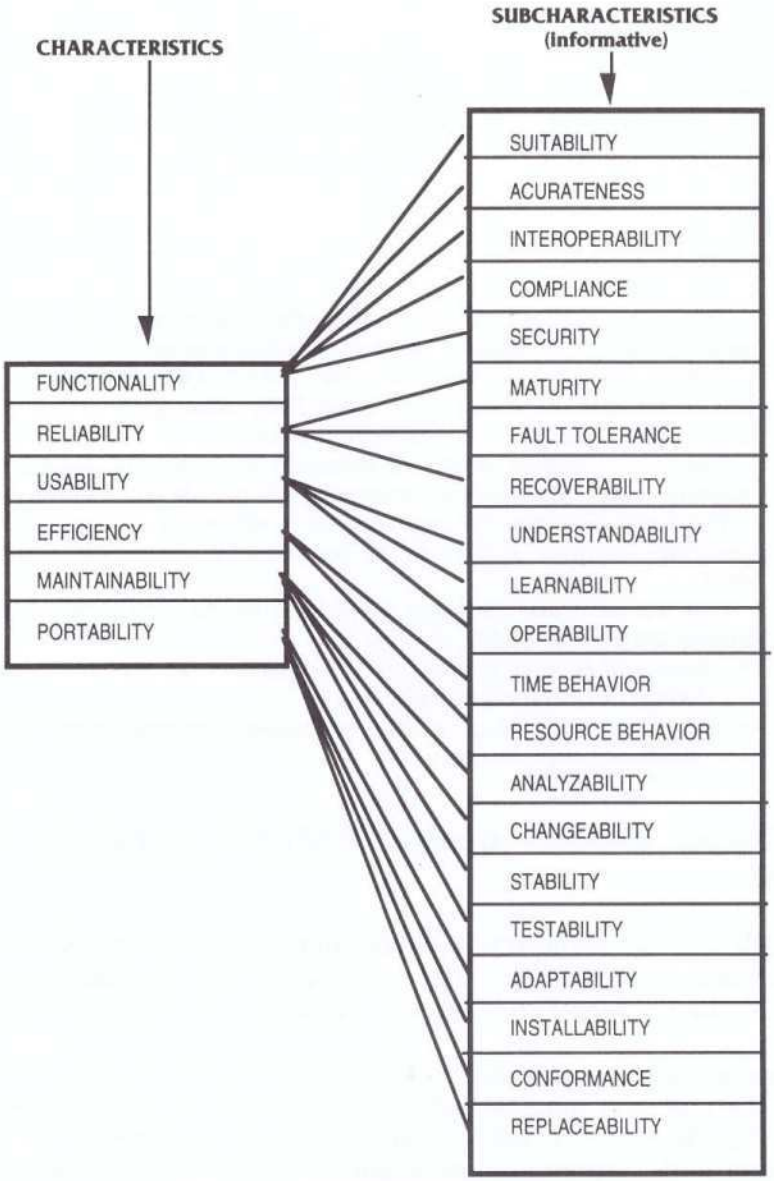


Figure 1: The ISO 9126 quality model

In addition, a short description of an "evaluation process model" is given in an informative appendix, that defines also a set of sub-characteristics that details the concepts of the above mentioned six characteristics. The evaluation process model proposed by ISO 9126 has been designed so that it may in principle be applied to any phase of the development life cycle for each component of the software product.

It consists of three main stages:

- Quality Requirement Definition;
- Evaluation Preparation;
- Evaluation Procedure.

It has to be noted that the process relies on the existence of a pool of (undeclared) techniques and metrics; as a consequence, a number of detailed activities are not present. In particular, analysis and validation of metrics are considered as feeding this pool, and thus not being part of the main process. In the following, a short explanation of the three stages of the evaluation process is given.

The purpose of the initial stage ("Quality requirement definition") is to specify requirements in terms of quality characteristics (and possibly sub-characteristics). Since a software product is composed of different components, the requirements may differ for the various components.

The purpose of the second stage ("Evaluation preparation") is to set up evaluation and to prepare its basis. It is refined into three steps:

- Quality metrics selection. Here metrics that correlate to the characteristics of the software product and allow direct measurement are established. Possible metrics are numberless: every quantifiable feature of software or interaction with its environment is a candidate metric.
- Rating level definition. The purpose of this activity is to define the scales onto which measured values will be mapped. Moreover, scale values must be divided into ranges corresponding to the levels of satisfaction of the requirements. Another time, no general values are possible; rather, they must be defined for each specific evaluation.
- Assessment criteria definition. This activity must prepare a procedure for summarising the results of the evaluation of the different characteristics. For instance, decision tables or weighted averages might be used. Managerial aspects such as time and costs may also be included in the procedure.

The last stage ("Evaluation procedure") is where the evaluation is actually performed in terms of:

- Measurement. The selected metrics are applied to the software product, obtaining values distributed on the defined scales.
- Rating. For each measured level, the rating level (i.e.: satisfaction) is determined.
- Assessment. The final step of the software evaluation process implies the summary of rated levels. By using the assessment criteria defined, a global result on the quality of the product is derived and then compared with managerial aspects (time, costs and so on) in order to take a decision.

The major evolution of the ISO 9126 are being accomplished through a number of guidelines for its usage, among which the one dedicated to the list of metrics and indicators is of particular interest [AZUM 93].

2. The SCOPE Project

An Esprit project has been dealing with the issues of software product quality evaluation and certification. This project is called SCOPE (Software CertificatiOn Programme in Europe, Esprit project n. 2151) [DENE 92]. The objectives of the project can be summarised as follows:

- to define procedures enabling the granting of a quality seal to the software when it complies with a certain set of quality attributes;
- to develop new efficient and cost effective evaluation technologies for the granting of this seal;
- to promote the use of modern software engineering technologies to be used during the development of the software and contributing to the delivery of the seal.

SCOPE involved partners from many European Community countries, belonging both to industry and academy. The project started on March 1989 and was concluded in June 1993. One of the main results of the SCOPE Project is the definition of an evaluation framework (composed of a model, a method and several techniques) that has been thoroughly experimented through a number of case studies. Moreover the SCOPE approach to evaluation has introduced an interesting concept for the practical implementation of quality evaluation: the Evaluation Modules (also known as Bricks) [ROBE 91]. An Evaluation Module is a precise guide-line describing the application of a specific evaluation technique on a given part of a software product with the objective to evaluate a specific characteristic of software quality in accordance with the ISO 9126 Standard. This modular approach to evaluation allows flexibility, can cope with technological/ methodological evolution and pushes towards the usage of such modules also during development and when specifications are defined. The advance of research in metrication and the new challenges presented by modern HW/SW technologies makes impossible to define a set of internationally accepted metrics based on statistical evidence of their soundness (following for instance the scheme proposed in [SCHN 92]); they would in fact become obsolete in a short time interval and would also probably not be applicable and meaningful outside the environment where they were devised. This is the reason why we think that it is better to have a library of evaluation modules that can be subjected to change in order to keep with evolution.

An evaluation module should be composed by two parts plus some optional annexes. The first part is what we call the evaluation specifications; within it, the following information must be given: general and specific definitions, target characteristic (among the six proposed by ISO 9126) and optional refinement into sub-characteristics, the evaluation technique to be used (inspection, execution analysis, static analysis or modelling), the documents required (e.g.: the product parts needed, such as: user's manual, source code, etc.), details of the assessment method and of its underlying "theory", identification of factors and metrics (unambiguous questions and formulae together with their target scales or units), clear identification of the data to be collected, cost information, required structure and elements of the evaluation report, references (to standards, to recognised theoretical work, etc.). A Specification is necessary for performing the assessment but might not be enough; this is the reason why examples of interpretation are also needed as a second part. These examples should give guidelines on thresholds, score computation and anything related to the pass/fail decision process. Some evaluation modules might also have optional annexes needed for tailoring to specific environments in terms of definitions (that is to say how do the defined items apply, given a particular product developed, for instance, in C++ and specified using SADT), data collection and tool selection. During the lifetime of the project more than one hundred such evaluation modules were defined; the most valuable ones were refined and

packed in a set of about twenty that cover all the six ISO 9126 characteristics and that are publicly available.

Besides, a co-ordinated work of standardisation within ISO is currently active in order to promote the application of the Evaluator's Guide [ISO 95], defining the phases that compose the software evaluation process, the evaluation levels that correspond to different requirements of quality and the evaluation techniques that can be applied depending on the considered evaluation level. This guide has been submitted to ISO/IEC/JTC1 SC7/WG6 and is intended to support the application of the ISO 9126.

As an outcome of the SCOPE Project, several services have been set-up for SW product quality evaluation with respect to ISO 9126; in particular an European agreement (known as "Euroscope") has been set-up among several companies in Europe in order to offer harmonised services for software product quality evaluation. Specific efforts are currently devoted to off-the-shelf packages and object-oriented code certification [CHEE 95].

For an extensive presentation of both technical and managerial aspects of SW product quality evaluation, the interested reader is referred to [BACH 94]; such book covers a wide range of topics on the subject, including:

- existing approaches to SW assessment;
- underlying principles of measurement;
- assessment techniques;
- metrics analysis and metrics databases;
- tools for data collection;
- case studies;
- management issues.

A more general introduction to software evaluation and certification can be found in [RAE 95].

3. Case study experiences

During the SCOPE Project a significant number of case studies were performed, as summarised in [BACH 94], demonstrating the pragmatic feasibility and the cost-effectiveness of the proposed approach. After the end of the project, the assessment techniques defined were brought to industry, both in the form of third-party evaluation services and in the form of internal usage by QA. In the following two case studies of measurement based product evaluation are summarised; both of them have been experienced in 1994 outside the context of the SCOPE Project, but rather as part of the activities of two big software producing units engaged in the development of challenging products in the telecoms application domain.

3.1 Maintainability evaluation and improvement

3.1.1 Environment description

The experience described in this paragraph has been matured at Siemens Telecomunicazioni Italia (STI), in the context of process/ product improvement activities focused on telecommunication systems for mobile phone handling, in accordance with the GSM International Standard, phase 2. In particular, maintainability evaluation has been applied to a SW development project characterised by the following data:

- Time scale for SW development and integration testing: 15 months (interspersed with several sub-releases, known as "builds");
- Product size: approximately 500 KLOC (Kilo Lines of Code), mainly in C language;
- Re-use from previous products: approximately 40%;
- Project staffing: approximately 60 Full Time Equivalent (including developers, testers, supervisors and support staff but excluding: configuration management, system test and project management staff).

As part of a significant process improvement effort focused on Quality Assurance and Quality Control practices [BAZZ 95] it was decided to define a set of indicators in order to follow a quantitative approach in project management and process improvement. In accordance with [GRAD 94], four major groups of indicators were devised:

- metrics used for project management: these are metrics whose primary goal is to put at the disposal of the project leader quantitative data to be used in order to take knowledgeable decisions during the life time of the project;
- metrics used for product evaluation: these are metrics whose primary goal is to single out critical parts of the SW product that might cause problems as far as maintainability and reliability are concerned;
- metrics used for derivation of baselines: these are metrics whose primary goal is to collect data useful for deriving site-specific models to be used in future projects as estimate rules;
- metrics used for validation of best practices: these are metrics whose primary goal is to quantitatively derive a judgement on the effectiveness of the process improvement initiatives, in order to decide whether to adopt them as part of the Quality Management System or to reject them.

In the following, the focus is given on metrics for product evaluation.

3.1.2 Goals of product evaluation

The goals of the product evaluation activity can be summarised as follows:

- to define quantitative rules in order to master the complexity of a SW product onto which massive maintenance is foreseen for several years;
- to evaluate the current quality level of the application, especially for those parts that showed to be error-prone in field and for those subsystems onto which major functional enhancements were planned (in both cases a decision had to be taken whether to restructure the existing SW or to re-engineer parts of it);
- to single-out (by means of Pareto analysis) SW parts that had to be subjected to manual code inspection;
- to provide designers with guide-lines and rules for the development of code easy to maintain;
- to derive a baseline for cross-comparison at company level.

3.1.3 Activities undertaken

The activities undertaken are summarised with reference to the steps of the evaluation procedure proposed by ISO 9126.

- Quality Requirement Definition
  In accordance with ISO 9126 and with the requirements expressed in the Quality Plan of the project, the target of evaluation was defined as the "Maintainability" characteristic, in terms of all its sub-characteristics: Analysability, Testability, Stability, Changeability.
- Evaluation Preparation
  - Quality metrics selection
    A number of metrics were defined (largely based on source code structural complexity [MCCA 76]), as described in the following table. Such metrics were related to coding rules and used as non-mandatory indications of which parts to re-engineer when new features had to be developed. The association to sub-characteristics was done in a such a way that the cardinality of the relationships is 1:N.

Goal	Metric	Definition
Analysability	Statement Size	Average number of operands and operator per statement
	Comment density	Percentage of comments versus number of statements
Testability	Cyclomatic number	As defined in [MCCA 76]; number of decision points + 1
Stability	Control Density	Percentage of control structure versus number of statements
	Number of Statements	Absolute number of statements within a function
Changeability	Nesting Levels	Maximum depth of nesting of control structures
	Max. Distance	Max. number of stat. from the start to the end of a structure

Table 1: Static analysis metrics for maintainability tracking

  - Rating level definition.
    As far as rating level definition is concerned, the following steps had to be performed: definition of lower and upper thresholds for metrics and backward integration of results: metrics --> sub-characteristics --> characteristics. The first aspect was accomplished starting from a set of reference threshold values and customising them by means of the derivation of metrics onto a statistical valid sample (about 30 KLOC) for which metrics were automatically collected and then their meaningfulness was manually validated by means of code inspection.

Metric	Lower Bound	Upper Bound
Statement Size	3	7
Comment density	0,2	0,65
Cyclomatic number	1	10
Control Density	0	0,2
Number of Statements	5	100
Nesting Levels	1	6
Max. Distance	0	25

Table 2: Thresholds for static analysis metrics

As far as integration algorithms are concerned, these were defined considering all possible combinations of values and defining a rating at characteristic level subdivided into five classes (as suggested by ISO 9126): poor, average, fair, good, excellent. In order to take into account the different importance of metrics, weighted composition methods were also used.

- Assessment criteria definition
  The metrics (calculated for each function contributing to the source code of the product) were integrated using weighted composition algorithms in order to derive a single "maintainability index" at various levels of granularity (function, process, functional area, processor, network element, product).
  The maintainability index was defined in the following way: $MI = \sum (w_i * n_i)$, where W_i is the weight associated to each class (poor = 0, average = 0.25, fair = 0.5, good = 0.75, excellent = 1) and N_i is the percentage of functions falling in that class. In this way MI spans in the range [0 .. 1], with 0 meaning a bad maintainability and 1 meaning an optimal maintainability. Having defined as goal a target level for MI greater than 0.70, the Quality Plan of the project stated that subsystems with a maintainability index below 0.6 had to be manually inspected in order to decide whether reverse engineering activities were needed.

- Evaluation Procedure
  - Measurement
    The selected metrics were automatically extracted from the software product, using the Logiscope static analyser; in order to keep the human overhead to a minimum, several batch programs were developed to run and control the jobs and to produce the reports.
  - Rating
    In order to produce outputs suitable for various classes of users (development supervisors, designers and QA team) the following reports were produced at different levels of granularity: overall quality report, list of functions subdivided by rating class, Kiviat graphs of average metric values, distribution of metrics, distribution of sub-characteristics, calling graphs among functions and, for each function falling either in the poor or in the average classes, Kiviat graphs of metrics and calling graph.
  - Assessment
    The final assessment stage involved several aspects: identification of critical components for which re-engineering activities were needed, selection of the sample for manual code inspection, analysis of the MI with respect to the defined target, study of the variation of the maintainability index during the development progress, validation of metrics and models. Such aspects are dealt in more details in the next paragraph.

3.1.4 Results obtained

The analysis of data showed that:
- the overall MI of the product was 0.74, thus exceeding the target value defined at the beginning of the work;
- despite this fact, some areas showed maintainability problems, with values for MI even below 0.60;
- areas with a low MI were areas historically error-prone or quite difficult to maintain.

Figure 2 shows the MI for the software subsystems residing on the Operation and Maintenance processor of the Base Station Controller Network Element.

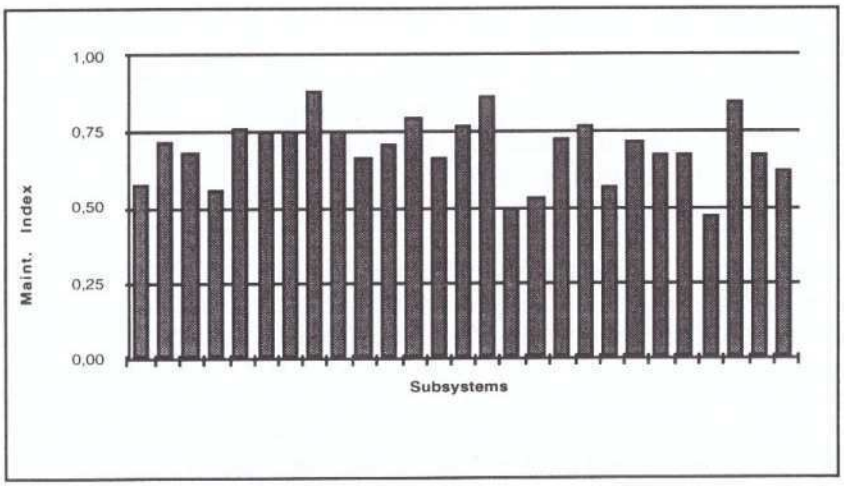


Figure 2: Maintainability index for several subsystems of the product under analysis

As already mentioned, the development of the system under analysis was planned in several builds; for this reason, it was quite interesting to analyse the trend of the MI across the various builds; Fig. 3 shows such trend across the three most significant builds for a number of selected design parts; the following aspects can be observed:

- areas where a major reverse engineering was made had a substantial growth of the maintainability index (e.g.: areas B and H);
- areas where specific structural changes were made in accordance with suggestions from static analysis metrics, also showed a significant bettering (e.g.: areas A and E);
- areas where static analysis metrics were anlayzed during development (even if no specific actions were adopted) succeeded in keeping to a minimum the impact of additional features within existing source code (e.g.: area D):
- areas were maintenance activities were massive starting from a brand new developed source code showed a good MI, with small diminishing along builds (e.g.: areas C, F and G)

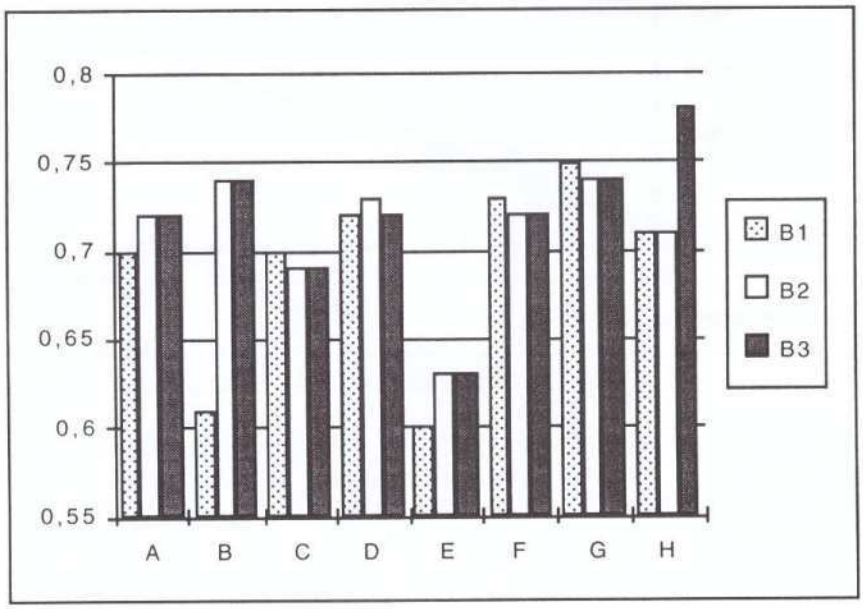


Figure 3. Maintainability index for some areas across various builds

The analysis of control graphs also provided useful feedback to designers, pointing out situations like:

- duplicated code that ought to be factorized in appropriate modules;
- multiple decision branches expanded too much;
- long modules that could have been easily broken down into sub-modules;
- questionable programming styles;
- usage of recursion in very complex functions.

As far as metrics validation is concerned, the defined metrics and thresholds were felt as adequate, in the sense that they succeeded in maximising the return on investment from re-engineering and code inspection activities, focusing the attention on the 20% of the modules that are likely to cause 80% of the maintenance troubles. Moreover, correlations were sought in order to validate the model and to check the results with other indicators of the measurement system; in particular the following aspects were found out:

- a correlation was found between the MI and the maintenance efforts;
- a strong correlation was found between the metric 'Number of statements' and the metric 'Cyclomatic number', thus demonstrating the multicollinearity of such metrics (see Fig. 4);
- no statistical valid correlation was found between the 'Maintainability Index' and, respectively, the 'Number of Statements' (correlation coefficient = - 0,58) and the 'Cyclomatic Number (correlation coefficient = - 0,40), thus demonstrating the higher informative value of the defined indicator with respect to the most widespread simple metrics normally used for tracking complexity and maintainability factors;
- a significant correlation (correlation coefficient = + 0,85) was found between the number of topological paths and the number of tests designed following a functional approach; this brought us to the idea that the testing coverage of the product (that was subjected to several thousands tests) was quite evenly distributed.

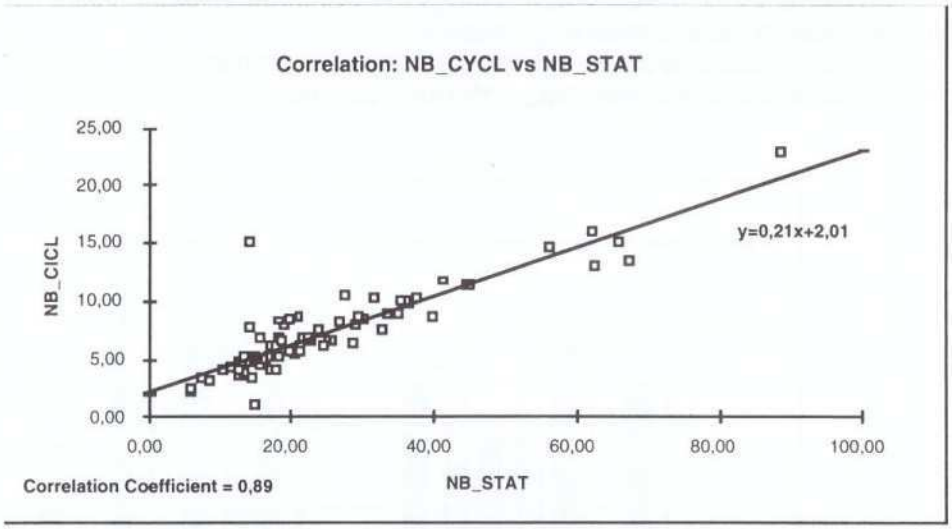


Figure 4: Multicollinearity of static analysis metrics

Correlation between MI and fault density was also analysed (see Fig. 5) resulting in a weak inverse correlation, meaning that the higher the MI, the better the product in terms also of reliability. This correlation is anyway thought to be not fully statistically valid and thus it will have to be analysed in more details in the future.

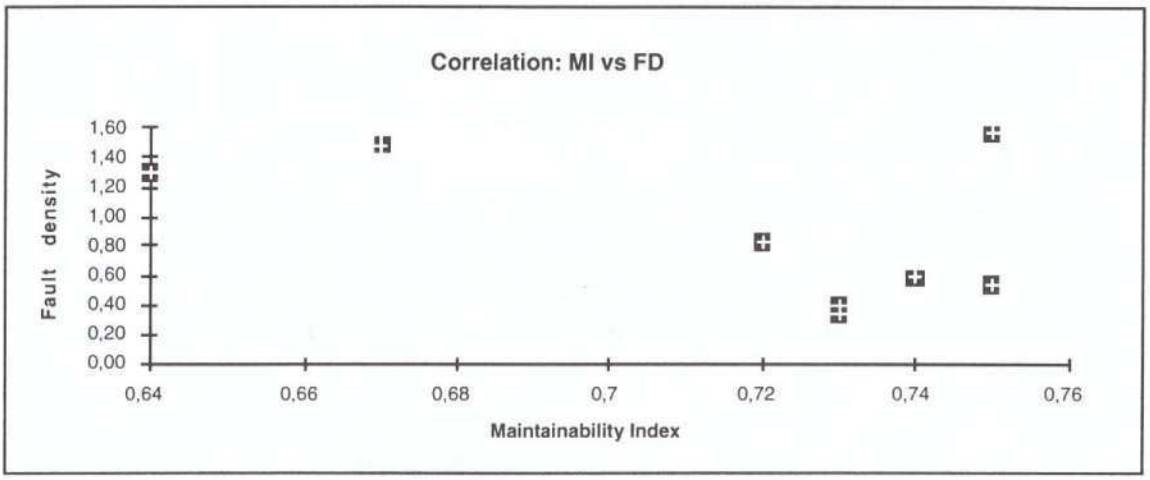


Figure 5: Correlation between Maintainability Index and Failure Density

3.1.5 Future steps

It is possible to say that the adoption of static analysis techniques was positive since designers focused their reverse engineering efforts on troublesome modules, in accordance with a Pareto strategy.

As a consequence of the results of the evaluation activities, the following steps have been planned:

- widespread adoption of the techniques and the tools experienced, that will be available to all designers as part of the software factory tool-kit;
- usage of the approach for guiding code inspection and reverse engineering activities;
- extension of the usage of the static analysis tool in order to derive also testing paths assuring a topological coverage during early testing phases;
- tuning of the adopted quality model.

3.2 Operability evaluation and improvement

3.2.1 Environment description

The experience described in this paragraph has been matured at Italtel SIT BUCT Linea UT, in the context of a long-lasting process/ product improvement effort [DAME 95]. In this context the main goal of the improvement program is the application of a Plan-Do-Check-Act scheme able to check and measure the products and the development process in a quantitative way, in order to single out, implement and monitor the improvement opportunities (as shown in Fig. 6).

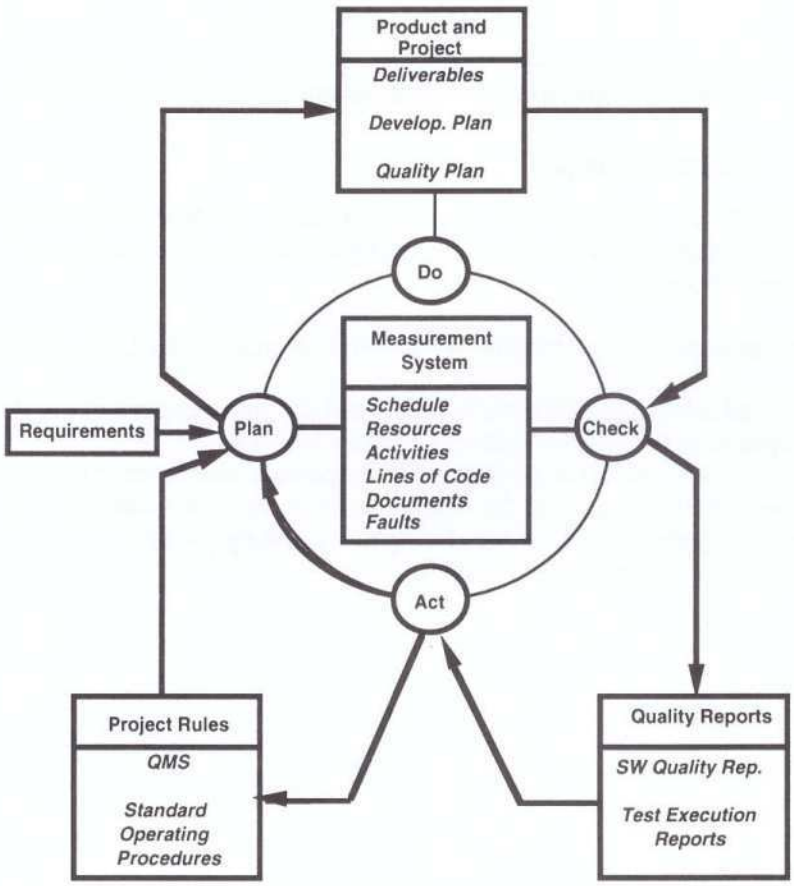


Figure 6: Application of PDCA to process/ product improvement

The improvement program has its roots in the Quality Management System and is constantly kept under control by means of a measurement system [DAME 93].

Within the many improvement activities undertaken in the last years, the following ones are directly related to product quality evaluation/ improvement:

- the adoption of ISO 9126 in the definition of the Quality Plan of each project;
- the optimisation of coding practices, in order to cover the following ISO 9126 characteristics: maintainability, efficiency, reliability, portability;
- development of proprietary tool-kits to check product quality in accordance with coding rules defined;

- automation of non-regression test suites;
- quantitative estimation of failure trend based on reliability growth models; this gave good results, with a percentage error between estimates and actual data in the range from 5 to 10% [BAZ3 93];
- root cause analysis of failures and scheduling slippages [DAME 96];
- statistical analysis of alarm logs produced in field.

In the following, details are given of the last issue mentioned.

3.2.2 Goals of product evaluation

The goal of statistical evaluation of alarm logs produced by operating switching in field can be summarised as follows:

- to improve the usability for operators, by reducing the number and typology of alarms (that are often too many or too verbose, thus resulting in excessive documentation that cannot be analysed thoroughly);
- to single out SW failures not notified by users but in any case detected by Operation and Maintenance procedures;
- to collect quantitative data describing the operational profile of the product in field.

3.2.3 Activities undertaken

The analysis started from the collection of alarm logs produced during one month (October 1994) by eight major switches operating in field, characterised by varying size and typology.

Data was then statistically analyzed, paying attention in particular to the following aspects:

- number of alarms notified;
- stratification by typology of alarm;
- stratification by class of switching;
- distribution within the day.

3.2.4 Results obtained

A first interesting thing to note was the fact that, among 900 types of events, a group of only 5 contributed to about the 40% of alarms notified, as shown in Fig. 7.

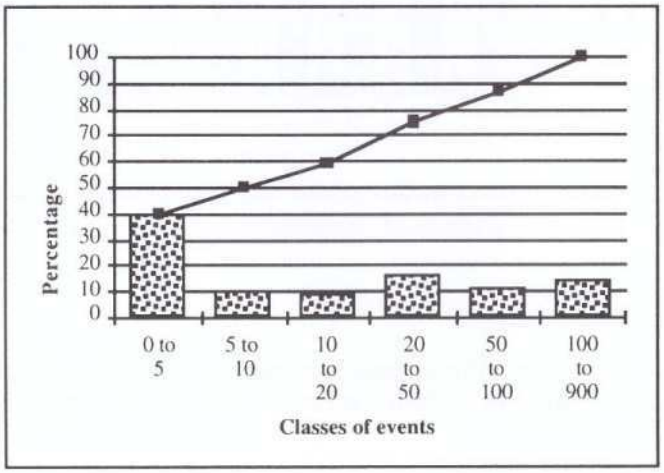


Figure 7: Pareto distribution of alarm logs

Moreover, among the first 20 events more frequently notified, it was notable to distinguish that only one was pertinent to software failures, whereas the others were related to periodic audits or to anomalies in the network.

The distribution of events during the day (see Fig. 8) was also very helpful to discriminate the events that caused the biggest overhead.

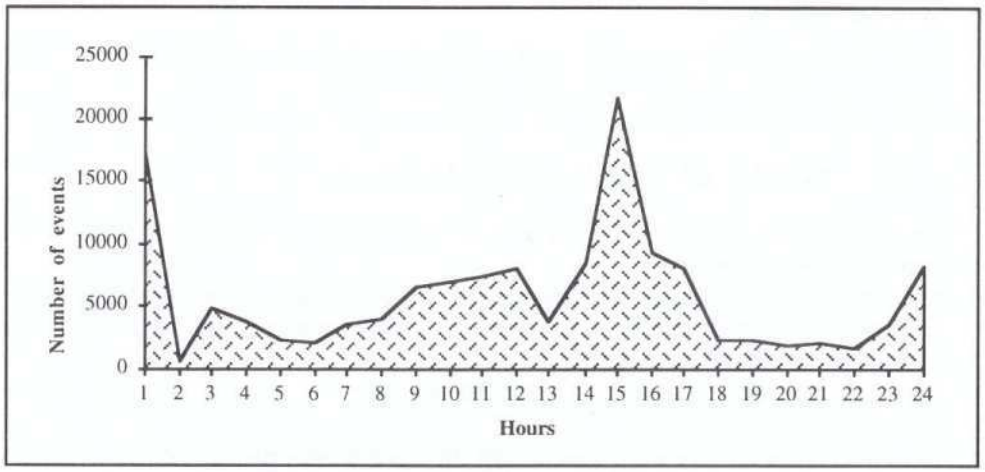


Figure 8: Distribution of events during the day

Fig. 9 shows the optimisation in operability (in terms of number of printed pages per month) that could be obtained simply by removing a very limited number of alarms that provided no major contribution in the monitoring of the status of the equipments.

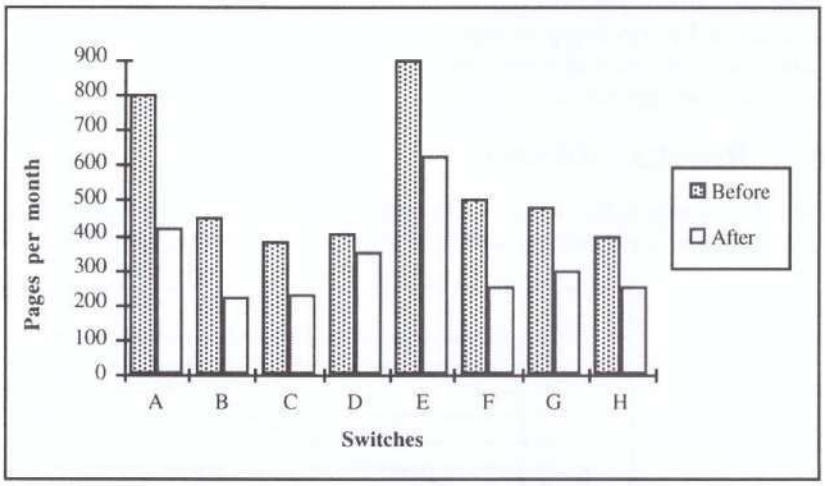


Figure 9: Effects of improvement actions onto operability issues

3.2.5 Future steps

It is possible to say that the adoption of the described techniques proved to be very useful since it provided valuable insights at both organisational and technical levels at a very low cost.

As a consequence, activities will be pursued with the following aims:
- application of such analysis in a systematic way;
- introduction of the technical mechanisms needed to better the operability;
- analysis of feedback on failures within root cause analysis activities.

4. A mature approach to product and process improvement

4.1 Process quality vs. product quality

The relationship between product and process evaluation/ improvement is somewhat controversial [VOLL 93]: which one gives the highest return on investment? Are both necessary? Is one the pre-condition for the other?

The feeling of IT representatives is traced by an European wide awareness survey (reported in [BAZ1 93]), that had the following goals:

- to evaluate the current situation of software product evaluation and certification and, more generally, of the sensibility about software quality problems;
- to identify the needs and demands for software product evaluation and certification;
- to understand which factors and agents can influence the dissemination of this software engineering practice;
- to get a feeling about the expected relationship between product and process certification;
- to study the level of knowledge of relevant standards in this field;
- to understand how potential users would expect an evaluation service to be carried out.

In particular, a critical look at the results concerning the relationship between product and process quality reveals the following interesting aspects:

- process certification according to ISO 9001 is felt as not enough to guarantee the quality of a specific software product (see Fig. 10);
- software product evaluation alone is felt as not enough to understand the maturity of the development environment and thus it is better to accompany it with a process assessment (Fig. 11), for instance by means of the Bootstrap approach [BOOT 94].

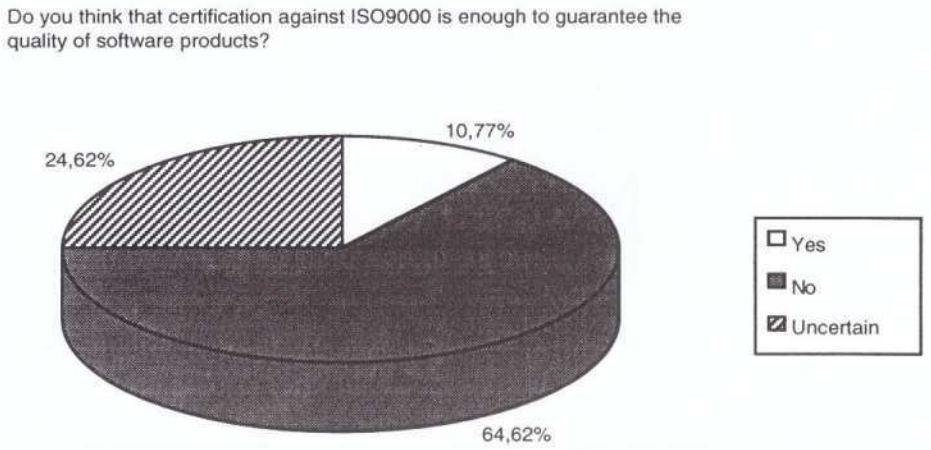


Figure 10: Appraisal of ISO 9000 certification with respect to the quality of delivered products

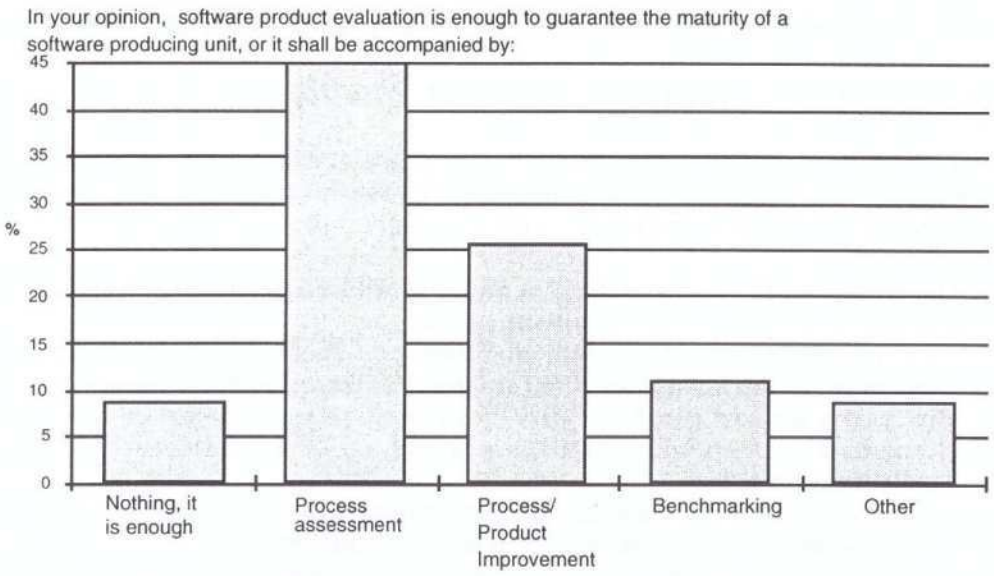


Figure 11: Software product evaluation and other issues

This concept is heavily underlined when ISO 9000 certification was considered: a very low percentage of the interviewed declared that such certification can give a sufficient guarantee of the quality of the delivered products. Product and process are closely linked and cannot be separated when quality is analysed: this is confirmed by Fig. 12, showing that most people ask for a combined assessment (both process and product).

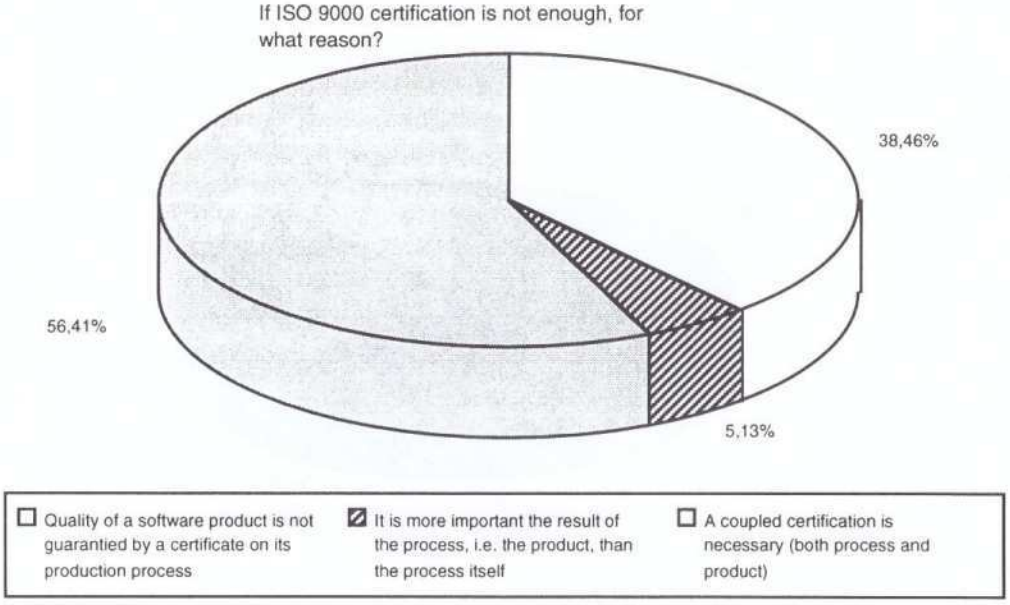


Figure 12: Reasons why ISO 9000 is felt necessary but not sufficient for guaranteeing software product quality

We stress the tight relationship that is perceived by all interviewed people, considering that no major difference in the judgement is evident looking at different roles or application domains or country (indeed, where ISO 9000 registration is more widespread - e.g. UK - the awareness of the need to combine it with product evaluation is particularly strong).

4.2 Tracking the effectiveness of process improvement onto product quality and business goals

Process improvement is sometimes advocated as the new silver bullet for software engineering. Indeed many companies are devoting a great deal of efforts and investments in order to set-up quality systems and raise the maturity of the software development process [HERB 94]. These efforts are based on the assumption that the best practices of software development have a positive impact on the ultimate goals of a software producing unit, namely: timeliness, productivity and quality. We feel that a slight problem might exist: it is very difficult to quantify the gains and make them tangible; moreover, it is even harder to find out quantitative relationships between process maturity level and the achievements at company-wide level. This is a problem for the widespread adoption of process improvement.

Several experiences (for instance [BAZ2 93] [HERB 94], [DAME 95], [CUSU 91], [SEL 91]) report the impacts in timeliness, productivity and quality due to the adoption of best software engineering practices. The reader is however always a bit sceptical since the effects are indirectly inferred: there is no quantitative evidence that the good results were in fact a consequence of process improvement. Might be it was due to a change of project manager, or to good luck, or whatsoever. What is meant is that the software engineering community needs quantitative data showing evidence of positive correlation between process maturity levels and project results. This would be much more effective than any theoretical assumptions about good engineering practices. The reader will recognise suddenly that this approach has an intrinsic problem: we find difficult to have quantitative values for process maturity and stability of the development process, in order to track improvements and their effects on final goals. This might be due to the following reasons:

- management-by-metrics schemes (see for instance: [PQMI 88], [BASI 81], [AMI 92]) usually start from company goals; thus they derive indicators that are related to business but are not directly related to the development process;
- standards related to quality management systems (such as ISO 9000/3 [ISO 9003]) take scarcely into account process stability and maturity. All you can know is whether a company is certified or not. This could allow to have a correlation in which the results of several companies (some of which already certified, some not yet) are compared and we might look whether clusters exist. This sort of chart, even if utterly interesting, is very difficult to draw since no two companies use the same set of indicators and thus their results are not comparable. Even if they were, such a diagram would not help a specific company in understanding whether its own QMS is working well;
- maturity level grading ([HUMP 89] [PAUL 91]), even if extremely useful, is characterised by levels that are a bit too far one from another. In order to successfully shift from one level to the following (and thus having two points on the desired diagram) you have to wait no less than two years. This is a time interval that is often too long for a company to monitor whether the right path has been chosen.

It is notable to see that almost all improvement programs warns you from making more than one change at a time to your process, otherwise you will not be able to assess its efficacy. This is due to our limited ability of assessing the effects of changes in the software development process. Unfortunately, this imposes unrealistic constraints: making one change at a time each two years will not advance the capabilities of software producing units to the level required by the market. As a consequence, current process assessment and improvement schemes are open to criticism with respect to their sparse data collection techniques [BOLL 92] or to the insufficient combination with total quality management findings [KUMI 93].

Bringing improvements to the development process of a complex software producing unit is a time consuming, trial-and-error activity. Unless the organisation is very rigid, you cannot think that process improvements become consolidated in a short time interval; rather, different attitudes will co-exist for quite a long time. Thus some projects might fully adopt new practices (these are commonly known as 'pilot projects'), some others could incorporate new issues to a limited extent, some others will keep on with the old practices; this can be due to various reasons, like: conclusion of a big project, severe schedule pressure, psychological resistance, etc.

The SEI has proposed a set of software measures [BAUM 92] that are compatible with the measurement practices of the Capability Maturity Model (CMM); within this set there are two very interesting process related indicators which provide information on the stability of the process by monitoring the number of requests to change the process and the number of waivers to the process. Such indicators are "Process Change Requests" and "Waivers from Process Standards", and their interpretation is as follows:

- "Process Change Requests": a large number of requests may indicate that the defined processes are not as efficient as believed or may not be appropriate to a particular project;
- "Waivers from Process Standards": a large number of waivers granted may indicate that the processes are actually undergoing ad hoc revisions and are not as stable as believed.

The idea of defining indicators aimed at keeping track of process stability is extremely worthy and should be expanded towards statistical process control techniques. In the following a proposal is made for a process indicator, named "Process Standardisation", aiming at providing more insights within the stability of the development process and the effects of process improvements. The "Process Standardisation" indicator measures the level of adherence to what defined in the Quality Management System. In order to do so, for each activity of the development life cycle, waivers to the QMS are taken into account by means of the definition of typical 'behaviours'. Behaviours represent the possible attitudes through which an activity can be exploited. Such behaviours are specific for each software producing unit (SPU). It is felt that four or five behaviours are normally sufficient: one corresponding to the thorough application of the QMS principles and of the latest process improvement actions, the others attached to typical pre-defined waivers. Waivers can embrace all aspects of the development process but are most likely to be directed towards: the presence/ absence of documentation, the way documents are written, the bounds among activities, the adoption of specific techniques, etc. At Italtel SIT BUCT, for instance, all behaviours are defined within an Operating Procedure. An example of the different behaviours defined for the system test activities is sketched in the table below.

Behaviour	Testing activity
1	No constraint has to be respected and no document has to be produced
2	The only constraint for starting test execution is the completion of conditioning tests. Test reports can be produced in a simplified way.
3	The constraints for starting test execution are: • completion of conditioning tests; • completion of coding and integration testing activities. Test reports have to be produced in accordance to QMS
4	According to the QMS (integral adoption of all standard operating procedures)
5	Fully accordance to the QMS and automation of test-suites for non-regression

Tab. 3: Behaviours and Waivers - an example

The experiences matured at Italtel SIT BUCT Linea UT and are part of a long-lasting process/ product improvement effort that has been supported and sponsored by S. Dal Monte, U. Ferrari and G. Damele. For the specific improvement actions described in the paper we have to thank: MC. Aletti, F. Aquilio, MG. Corti, L. Giovanelli, G. Panzeri, G. Pisano, F. Pompili, D. Scrignaro, G. Vailati.

The experiences matured at Siemens Telecomunicazioni Italia fall within a major effort in software quality management applied to a mobile phone development project, supported and sponsored by G. Cecchetto, E. Pietralunga and G. Vulpetti. For the specific experience described in the paper we are indebted to: O. Balestrini, L. Barbieri, P. Bettoni, R. Delmiglio, G. Falzoni, B. Ferri, S. Finetti, A. Lora, A. Manini, B. Marelli, B. Montanari, L. Travaglini.

Finally we have to thank O. Fouillouze and M. Maiocchi for the support given in the set-up and supervision of activities.

7. References

[AMI 92] A. Kuntzman-Combelles, P. Comer, J. Holdsworth, S. Shirlaw
"Metrics Users' Handbook"
AMI Project, Cambridge, 1992

[ARTH 85] L.J. Arthur
"Measuring Programmer Quality"
John Wiley and Sons, 1985

[AZUM 93] M Azuma
"Information Technology - Software Product Evaluation - Indicators and metrics", Working Draft within ISO/JTC1/SC7/WG6, Project 7.13.3, 1993

[BACH 94] R. Bache, G. Bazzana
"Software metrics for product assessment"
Mc Graw Hill, 1994

[BASI 81] V. Basili, D. Weiss
"A methodology for collecting valid software engineering data"
IEEE Trans. on Software Engineering, Vol. se-10, November 1981

[BAUM 92] J.H. Baumert, M.S. McWhinney
"Software measures and the Capability Maturity Model"
CMU/SEI-92-TR-25, September 1992

[BAZ1 93] G. Bazzana, R. Brigliadori, O. Andersen, T. Jokela
"ISO 9000 and ISO 9126: friends or foes?"
Proceedings of IEEE Software Engineering Standards Symposium, Brighton, September 1993

[BAZ2 93] G. Bazzana, P. Caliman, D. Gandini, R. Lancellotti, P. Marino
"Software management by metrics: practical experiences in Italy"
10th CSR Workshop, Amsterdam, October 1993
Published in:
Norman Fenton, Robin Whitty and Yoshinori Iizuka
"Software Quality Assurance and Measurement - A worldwide perspective"
International Thomson Computer Press, 1995

[BAZ3 93] G. Bazzana, G. Damele, M. Maiocchi, G. Zontini
"Applying Software Reliability Models to a large industrial dataset"
Information and Software Technology, Dec. 1993

[BAZZ 95] G.Bazzana, R. Delmiglio, A. Lora, O. Balestrini, S Finetti
"Quantifying the Benefits of Software Testing: an Experience Report from the GSM Application Domain",
Proceedings of Objective Quality Conference, Florence, May 1995
Published by Springer-Verlag in "Lecture Notes in Computer Science", N° 926

[BOEH 78] B.W. Boehm et al.
"Characteristics of Software Quality"
TRW series of Software Technologies, Vol. 1, North Holland, 1978

[BOLL 92] T.B. Bollinger, C. McGowan
"A Critical Look at Software Capability Evaluations"
IEEE Software, July 1992

[BOOT 94] P. Kuvaia, J. Simila, L. Krzanik, A. Bicego, S. Saukkonen, G. Koch
"Software process assessment & improvement: the Bootstrap Approach", Blackwell, 1994

[BOWE 85] T.P. Bowen, G.B. Wigle, J.T. Tsai
"Specification of Software Quality Attributes" Volumes I, II and III
Rome Air Development Centre, RADC-TR-85-37, 1985

[CHEE 95] M. Cheek
"ESPRIT's Legacy of Software Evaluation and Certification",
IEEE Software, May 1995, pages 90-91

[CUSU 91] M.A. Cusumano
"Japan's software factories"
Oxford University Press 1991

[DAME 93] G. Damele, G. Bazzana, M. Giunchi, G. Rumi
"Setting-up and using a metrics program for process improvement"
AQuIS'93, Venice, October 1993

[DAME 95] G. Damele, G. Bazzana, M. Giunchi, G. Caielli, M. Maiocchi, F. Andreis
"Quantifying the Benefits of Software Process Improvement in Italtel Linea UT Exchange"
International Switching Symposium 95, Berlin, April 1995

[DAME 96] G. Damele, G. Bazzana, F. Andreis, F. Aquilio, S. Arnoldi, M. Giunchi
"Process Improvement through Root Cause Analysis"
Accepted for presentation at AQUIS '96, Florence, January 1996

[DENE 92] B.De Neumann, G. Bazzana
"A Methodology for the Evaluation / Certification of Software"
3rd European Conference on Software Quality Assurance, Madrid, November 1992

[DEUT 88] M.S. Deutsch, R.R. Willis
"Software Quality Engineering"
Prentice-Hall, 1988

[FORS 89] T. Forse
"Qualimetrie des systèmes complexes"
Les Editions d'Organisation, 1989

[GILL 92] A. C. Gillies
"Software quality - Theory and management
Chapman & Hall Computing, 1992

[GRAD 92] R.B. Grady
"Practical software metrics for project management and process improvement"
Prentice-Hall, 1992

[GRAD 94] R.B. Grady
"Successfully applying software metrics"
IEEE SW, Vol.27, No.9, September 1994

[HERB 94] J. Herbsleb et al.
"Benefits of CMM-based Software Process Improvement: Initial Results"
SEI Technical Report, August 1994

[HUMP 89] W.S. Humphrey
"Managing the Software Process"
Addison-Wesley, 1989

[ISO 9000-3] ISO 9000-3
"Quality Management and Quality Assurance Standards - Part 3: Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software"
ISO, September 1990

[ISO 9126] ISO/IEC 9126
"Information technology - Software evaluation - Quality characteristics and guide-lines for their use"
ISO, December 1991

[ISO 95] ISO/ IEC JTC1/ SC7 N1317 (Editor: P. Robert)
"ISO/ IEC CD 14598 - 5.2 Information Technology - Evaluation of software product - Part 5: Evaluator's Guide""
ISO Committee Draft, Jan 1995

[KUMI 93] H. Kumi
"Quality Management by ISO-9000 and by TQM"
Proceedings of EOQ 93, Helsinki

[MCCA 76] T.J. McCabe
"A complexity measure"
IEEE Transactions on SW Engineering, 1976

[MOLL 92] K.H. Moeller, D. Paulish
"Software Metrics: a practitioner's approach to improved software develoment"
Chapman & Hall, 1992

[PAUL 91] M.C. Paulk, B. Curtis, M.B. Chrissis, E. Laverill, J. Bamberg, T.C. Kasse, C. Timothy, K. Konrad, J.R. Perdue, C.V. Weber, J.V. Withey
"Capability Maturity Model for Software"
SEI, Carneige Mellon University, Pittsburgh, 1991
CMU/SEI-91-TR-24, ADA240603

[PQMI 88] AT&T
"Process Quality Management and Improvement Guidelines"
AT&T Quality Steering Committee, Issue 1.1, 1988

[RAE 95] A. Rae, P. Robert, H. L. Hausen
"Software Evaluation for Certification - Principles, Practice and Legal Liability"
Mc Graw-Hill, 1995

[ROBE 91] P. Robert, F. Seigneur
"A modular approach for software product assessment - The brick concept"
In 2ème Rencontre Qualité Logiciel & Eurometrics 91. Actes et catalogue de l'exposition (1991)

[SCHN 92] N. Schneidewind
"Methodology for Validating Software Metrics",
IEEE Transactions on Sw Engineering, Vol.18, No.5, May 1992, pp. 410-422

[SEL 91] SEL, NASA Goddard Space Flight Center
"Software Engineering Laboratory Relationships, models and management rules"
SEL-91-001, February 1991

[VMAR 90] A.Von Maryhauser
"Software engineering methods and management"
Academic Press, 1990

[VOLL 93] T.E. Vollman
"Software quality assessment and standards"
IEEE Computer, June 1993

Object-Oriented Software Measures

Horst Zuse

Technische Universität Berlin
Fachbereich Informatik
Franklinstraße 28/29
FR 5-3
10587 Berlin
Gemany
E-mail: zuse at cs.tu-berlin.de
Phone: +49-30-314-73439
Fax: +49-30-314-21103

Keywords

Measure LOC, software measures, software measurement, measurement theory, object-oriented software measures, concatenation operations, Dempster-Shafer Function of Belief, De Finetti axioms.

Abstract

In this paper foundations of the properties of object-oriented software measures are presented. The criteria for the properties of object-oriented software measures are characterized with several concatenation operations between objects, classes, methods, etc. Concatenation operations can be used as a tool to give numbers an interpretation above the ordinal scale level. The result of this investigation is that software measures for object-oriented techniques have completely other properties than measures for imperative languages. It is shown that many of the measures in the object-oriented programming area follow the properties of the *Dempster-Shafer Measure of Belief* in Artificial Intelligence.

1 Introduction

In order to show the properties of software measures, we use measurement theory which allows to translate mathematical / numerical properties of measures back to empirical (intuitive) properties. We can do this under the assumption that the considered measure can be used as an ordinal scale which implies that we have a homomorphism. many measures related to imperative languages assume an extensive structure.

We demonstrate that object-oriented software measures have totally different properties as software measures for imperative languages. In order to have criteria for object-oriented software measures above the ordinal level, we use the Function of Belief and the De Finetti axioms.

The paper is structured as follows: In Section 2 we introduce some aspects of object-oriented measures, in Section 3 we introduce empirical conditions behind software measures, in Section 4 we introduce very briefly measurement theory, in Section 5 we discuss properties of object-oriented measures, in Section 5 we show the behavior of object-oriented measures related to the Function of Belief, and the De Finetti axioms. In Section 7 the approach of Section 6 is applied to an object-oriented measure, the results are presented in Section 8, Section 9 shows the conclusions and Section 10 contains the used references. The attachment gives a brief introduction into the basic concepts of software measurement.

2 Object-Oriented Software Measurement

In literature more than two hundred measures /FETC95/ for applications in the area of object-oriented programming can be found. This process of defining new measures is not finished today. The reason for such a lot of

measures may be problems in understanding object-oriented programs.

A very early investigation of OO-Measures can be found by Rocacher /ROCA88/. In 1989 Morris /MORR89/ discussed software measures for an object-oriented application, Bieman /BIEM91/ discussed software measures for software reuse in an object-oriented, environment, Lake et al. /LAKE92/ discussed measures for C++ applications, Chidamber et al. /CHID94/ evaluated different Smalltalk applications, Sharble et al. /SHAR93/ discussed measures for an object-oriented design (OOD), Li /LI..93/, /LI..93a/ evaluated ADA-Programs, and Chen and Lu /CHEN93/ evaluated OO-Measures related to the OOD-method of Booch /BOOC91/. Karner /KARN93/ wrote a master thesis of measurement in an object-oriented environment. Other books or papers of this area are /DUMK92/, /DUMK94/, /CALD91/, /JENK93/, /JENS91/, /LARA90/, /LORE93/, /LORE94/, /RAIN91/, /BARN93/, /MART94/, /MART94a/, /TEGA92/, /TEGA92a/, /ABRE93/, /ABRE94/, /WHIT92/, /LAKE94/, /TAYL93/, /ZUSE94c/, /ZUSE95/.

We will not discuss object-oriented measures here since it would be beyond the scope of this paper. We refer to the diploma thesis of Fetcke /FETC95/, who investigated more than one hundred object-oriented software measures from literature.

3 Empirical Conditions behind Software Measures

We explain or characterize the properties of software measures with empirical conditions. There are several reasons to consider software measurement from an empirical view. We collected some of them.

- Empirical conditions or statements are more intuitive for the user than mathematical statements.

- In reality empirical statements are considered together with the Measures LOC and the Measures of McCabe, for example the term *difficulty of maintenance*, or *complexity* of a program.

- Quality characteristics of software are also described by empirical statements in the ISO 9126 norm /ISO.91a/.

- Different formal definitions of a measure can be reduced to one empirical condition, for example wholeness (the sum should be greater than the sum of the parts, see also Section 4 and the attachment) can be reduced numerically to an additive measure without modifying the empirical assumptions.

- Another example is the normalization of software measures to numbers between 0 and 1. Assume, we have a Measure u which can be used as a ratio scale. In order to get numbers between $0 \leq u \leq 1$ the following Measure u' is proposed: $u' = 1 - (1/(1+u))$ It is important to mention here, if the Measure u can be used as a ratio scale, the Measure u' cannot be used as a ratio scale. However, both Measures u and u' assume the same empirical conditions. That means, the Measure u' is a numerical modification of the Measure u without changing the empirical meaning of the measure.

- The GQM (Goal-Question-Measure paradigm) /BASI84/ defines goals of measurement empirically, the questions are also empirically, but the measure are formally. The missing link between measures goals and questions are the empirical relational systems of measures. It is important to have the same language for the goals, the questions and the measures in order to select a measure. Here it is obviously, that the translation of mathematical properties back to empirical properties by measurement theory is very helpful.

- Discussing measurement scales, like ordinal, interval, ratio or absolute scales, we need both: the empirical and formal conditions (See the attachment).

- The calculation of correlations (for example with Spearman, Kendall Tau or Pearson) between a software measure and an external variable, like costs of maintenance, requires an empirical interpretation of the measure and the external variable. The reason is that the use of correlation coefficients requires certain scales levels. Scales are defined by empirical and numerical relational systems (Scale types are defined by admissible transformations). The correlation coefficient itself is used to compare empirical relational systems.

4 Measurement Theory

There exists a theory, called measurement theory, which gives conditions how to combine empirical conditions with numerical conditions. Measurement theory is also a proper theory to translate mathematical properties of measures back to empirical properties. Readers who are interested to get a deeper understanding of measurement theory we refer to Roberts /ROBE79/, to Krantz et al. /KRAN71/, to the attachment at the end of this paper, and to some works of the author, like /ZUSE91/, /ZUSE92/, /BOLL93/ and /ZUSE94b/.

4.1.1 Software Measures for Imperative Languages

In order to explain the empirical interpretation of the numbers, we give a very brief introduction using the Measure LOC = |N|, where N is the set of nodes in a flowgraph. Let us assume we have two Flowgraphs P1 and P2 and we have

LOC(P1)=200 and LOC(P2)=400.

From a mathematical view we can say: Flowgraph P1 has less lines-of-code than Flowgraph P2. We also can write

LOC(P1) < LOC(P2).

This statement is always true and cannot be falsified. In this context the numbers 200 and 400 do not have any empirical meaning., they are only mathematical statements.

However, if we give the numbers of the Measure LOC an empirical interpretation like *difficulty of maintenance* then the Measure LOC expresses that Flowgraph P1 is easier to maintain than Flowgraph P2. This empirical statement can be falsified and the following views are possible, too.

- P1 is equally difficult to maintain than P2.
- P2 is less difficult to maintain than P1.

More formally we denote the relation *equally or more difficult to maintain* with •≥. The statement P1 •> P2, means that P1 is more difficult to maintain than P2, and the statement P1 ≈ P2 means that P1 and P2 are equally difficult to maintain. Combining ≈ and •> we write P1 •≥ P2, and denote this empirical relation as: *equally or more difficult to maintain*.

4.1.2 Extensive Structure

The extensive structure consists of a set of empirical axioms. It also exists the Theorem 1.2 in measurement theory that says: a measure, which is additive, assumes an extensive structure. The extensive structure consists of five empirical conditions, also denoted as axioms:

Weak order.
Axiom of weak positivity.
Axiom of weak associativity.
Axiom of weak commutativity.
Axiom of weak monotonicity.
Archimedian axiom.

For flowgraphs, the picture below summarizes the axioms of the extensive structure for flowgraphs.

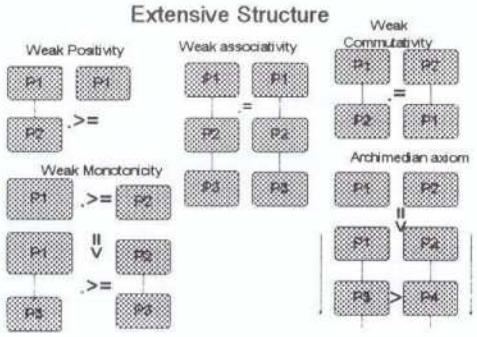


Figure 4.1: Extensive structure for flowgraphs. The weak order has been left out because it is not based on a concatenation operation.

The extensive structure leads to the (additive) ratio scale. For many measures in the area of structured programming we can find the properties of the extensive structure for software measures.

Important to Notice:
One of the major criterion of software measures for imperative languages was whether they assume an extensive structure or not.

We now will show, that we do not have such properties for object-oriented software measures.

5 Measurement Theory and Object-Oriented Software Measures

In this Section foundations of the properties of object-oriented software measures are presented. The criteria for the properties of object-oriented software measures are characterized with several concatenation operations between classes. Concatenation operations can be used as a tool to give numbers an interpretation above the ordinal scale level. The result of this investigation is that software measures for object-oriented techniques have completely other properties than measures for imperative languages. It is shown that many of the measures in the object-oriented programming area follow the *Dempster-Shafer Function of Belief* in Artificial Intelligence and the *DeFinetti axioms*. A detailed discussion of this approach can be found in Zuse et al. /ZUSE95/ and Fetcke /FETC95/.

5.1 Introduction

During the past more than two hundred software measures for object-oriented design and programming have been proposed. However, there is a uncertainty of proper definitions of such measures. As Churcher et al. /CHUR95/ point out: *In our own work on OO metrics, we have found the lack of standard terminology and practices to be a major obstacle to ready comprehension and comparison of the work of others. The varied nature of OO languages means that great care must be taken in comparing results. We believe that the ad hoc approach that has characterized conventional software metrics work must be avoided to enable progress to be made on the more changing problem of developing OO metrics. This suggests that empirical and theoretical considerations should receive equal weighting in the development of new metrics.*

We agree to this view. In the area of imperative languages many fundamental concepts of software measurement were not understood. One the one side concerns this the definition of measures. The Measure LOC is such an example, where no standardized definition of a line of code exist. The Halstead measures are a further example, because the definition of operators and operands are not defined properly. On the other side, there exists a lack of education of fundamental concepts of measurement and measures. This lack of good education of people leads to a wrong use of software measures and a wrong procedure of the investigation of software measures in all areas of this discipline People propose properties of software measures more in an intuitive way or by a feeling, than in a scientific way.

The investigation of Chidamber et al. /CHID91/, /CHID94/ shows some properties of object-oriented software measures. Chidamber et al. use the Weyuker /WEYU88/ properties. They investigate their measures in the direction of the Weyuker Property 9, which also is called wholeness. However, wholeness can be modified to an additive property /BOLL93/, /ZUSE94/. Having an additive measure, every axiom of the extensive structure is fulfilled. Most of the object-oriented measures do not fulfill idempotency. In this case we have no extensive structure and wholeness cannot be fulfilled either. Another problem is that the Weyuker properties are not compatible (See Chapter 6 /ZUSE91/)..

5.2 Abstraction Levels

Since standardized definitions of object-oriented software measures are very poor, we discuss fundamental concepts of object-oriented software measurement on a very high abstraction level. People may criticize that, but if measurement is not understood on a high level abstraction, we never can understand the fundamentals of software measurement on lower abstraction levels

In order to discuss object-oriented software measures, we have to discuss the abstraction level which we use for measurement in the object-oriented area. Abstraction levels are also called models of a program or a software system. For example, the Measure of McCabe is based on the abstraction level of a flowgraph, and the Measures of Halstead /HALS77/ are based on a quadruple (n1, n2, N1, N2). This quadruple describes another abstraction level. We discuss four levels of abstraction in the area of object-oriented measures.

5.2.1 Classes

The first level of abstraction is the class level. A class describes the properties of objects with attributes (often called instance variables) and methods. A class has a set of attributes and a set of methods defined in it. In Figure 4.1 the abstraction of a class as a picture is presented.

5.2.2 Methods

The behavior of an object is characterized by the methods defined in its class. A method contains the code of an OO program. Methods can be seen as being subroutines, which are invoked by messages sent to an object. In some OO-systems methods are constructed similar as subroutines in structured programming. The idea, in our case of measurement, is therefore to treat methods as procedures in software measurement.

A method in our view (measurement view) consists of a sequence of statements with a single entry point and a single exit point. The sequence of statements is executed each time the method is called. That means, we can consider the control flow (flowgraph) of the method.

Additionally, a method can be parameterized, having a list of parameters. Parameters can be input, output or both. So we can consider some data flow, additionally. In Figure 4.1 the abstraction of a class with methods is presented.

5.3 Structures On Classes

Of course, classes are not the major structure in OO-systems. Classes are related to each other by two types of relationship: *uses relationships* and *inheritance relationships* /BOOC91/. Two other types of relationship will not be considered here, namely *meta classes* and *generic classes*. These constructs are not under consideration by any proposed software measure we found.

5.3.1 Inheritances Relationships

A class A inherits the properties / attributes of class B. The properties of Class B become additional properties of Class A without being defined in Class A again. In turn, Class B is a generalization of Class A.

In Figure 4.1 we have a class vehicle that is able to move. Car and bicycle are specialization's of vehicle, both are able to move too. The method move and the attribute wheels are inherited along the thick arrow from vehicle to its subclasses. These properties are not denoted for the inheriting class.

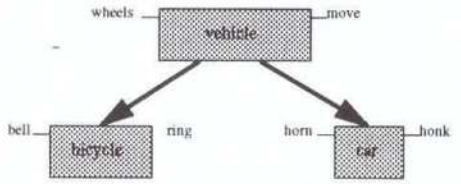


Figure 5.1: Classes, methods, and Inheritance.

We see, that we represent a class by a rectangle. The inheritance is represented by a thick arrow, which points to the class which inherits. Methods are represented by a thin line assigned to the rectangle and a name of the method.

The specialized classes have additional properties, i.e. a bike can ring it's bell. A class can inherit properties from more than one class. This case is called multiple inheritance and is shown in Figure 4.2

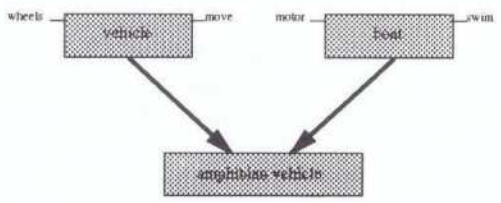


Figure 5.2: Multiple inheritance.

Here, the class amphibian vehicle is a special case of vehicle and of boat. So an amphibian vehicle can move and swim.

5.3.2 Uses Relationships

A Class A uses class B, if Class A accesses objects of Class B. Let us consider a Class library with a method borrow a book to borrow a book, where books is a class itself, then the Class book uses the Class Books. The use of a class can be hidden by the implementation, the Class Library could use a Class List, without the appearance in the interface of Class Library. Booch divided these two cases into *uses for interface* and *uses for implementation*. We denote the uses relationship with a thin arrow to the used class and leave the kind of use unspecified.

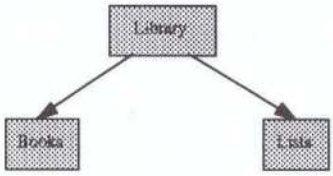


Figure 5.3: Uses relationships.

5.4 OO-System Level

Taking classes and the two relationships on classes (inheritance and uses relationship) together we can speak of an OO system. That is a set of classes and the uses and inheritance relation on this set.

Typically, none of the two relations builds a complete graph on the set of classes. In contrast, both can build a couple of unconnected subgraphs in one system. In order to be able to cope with structures on classes, we define two subsets of systems.

These are

a) *inheritance hierarchies* with one single root. The graph has to be connected, multiple inheritance is allowed.

b) *uses hierarchies* with one single main class that is the only class that is not used by any other class in that graph. The graph must be acyclic and connected.

A system can consist of several of such subsystems. We discuss the properties of measures related to the constructs as described above. Both inheritance and uses hierarchies contain the single class as a special case.

5.5 Concatenation Operations for Object-oriented Programs and their Properties

We now consider concatenation operations. Before discussing the concatenation operations we shall speak of some properties of operations in general. As discussed earlier, extensive measurement (extensive structure) is assumed by additive measures.

5.5.1 Extensive Structure

Above, we defined the extensive structure which consists of the axioms weak positivity, weak associativity, weak commutativity, weak monotonicity and the Archimedian axiom. For our considerations of object-oriented measures we need a further axiom called idempotency.

5.5.1.1 Idempotency

Additionally, a further concatenation operation is essential in the area of object-oriented programming, and that is idempotency.

Definition 4.x (Idempotency)
A concatenation rule is called idempotent, if for all objects a holds:

$$a \circ a = a.$$

®
When ever a concatenation operation is idempotent, there is **no way** that the Archimedian axiom can be fulfilled. The consequence is that we have no extensive structure.

Hence, having idempotency, we cannot come to the ratio scale via a concatenation operation. Some of the concatenation operations discussed in the following sections in fact are idempotent, so that we have to consider other structures than the extensive structure. These new structures are *belief structures* and the *De Finetti axioms*.

5.5.2 Weyuker Properties

As mentioned above, some authors /CHID91/, /CHID94/ use the Weyuker Properties to investigate the properties of their measures via concatenation operations. Again, we show, which axioms of the extensive structure are violated by the Weyuker properties.

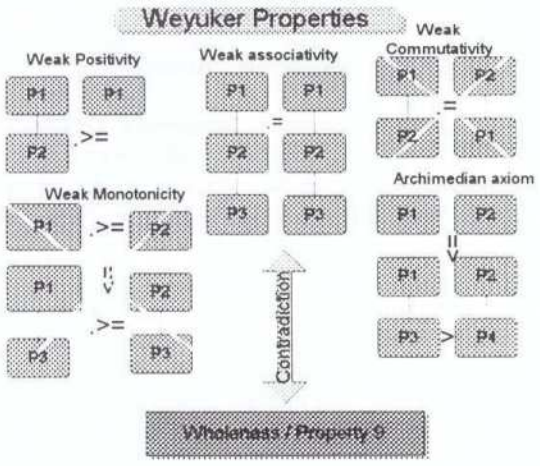


Figure 5.4: Violation of the Axiom of weak commutativity, and weak monotonicity of the extensive structure, and the requirement of wholeness is a contradiction.

Another problem is that the Weyuker Properties are not compatible. Property 9 is not compatible with the axiom of weak commutativity and weak monotonicity of the extensive structure /ZUSE91, Chapter 6).

5.5.3 Concatenation Operations for Methods

Since many OO programming languages are imperative in their methods, it is possible to apply traditional intra-modular metrics on method level. More than ninety of such measures based on flowgraphs were investigated in /ZUSE91/ in the context of the sequential concatenation operations BSEQ. 7.

5.5.4 Concatenation Operations on the Class Level

Chidamber et. al. introduce in /CHID91/ and /CHID94/ a concatenation operation for classes. They use concatenation operations in order to study the properties of their measures. We will use the concatenation operations of Chidamber et al. for our studies, too, and we consider some additional concatenation operations.

5.5.4.1 CUNI: Class Unification

The unification of two classes has no means of expression in OO models. As necessary for being an concatenation operation the result of combining two classes is one single new class. This new class combines all the properties the two single classes. If we concatenate Classes A and B to a new class C, we denote this with

C=CUNI(A,B).

Doing this, we unify the sets of properties of the both Classes A and B to the set of properties of the new Class C. Properties of a class are it's attributes (or instance variables) and methods. The *sets* of properties are *unified*, so identical methods or attributes occur only once. Considering the Classes A and B in Figure 4.7 we see, that both classes have the Variable a and Method M_1 in common. The new Class C has these variable and method only once.

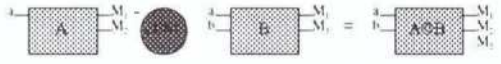


Figure 5.5: Class unification.

We now consider the implications of concatenation operations on the inheritance relation.

Inheritance Relationship

Within a class declaration we usually find information on (direct) superclasses. We don't find information on subclasses, as we can freely define subclasses to existing classes. We can consider subclasses to some extent to be a form of a property of a class, as for example for polymorphism. Furthermore, some measures proposed in literature use subclasses to characterize a class. For this reason, we consider the inheritance relation in both directions.

We require that the combined class inherits all the properties which any of the two original classes inherited. So any superclass of either A or B or both must be a superclass of the concatenation operation CUNI(A,B). This can lead to multiple inheritance. Consistently, we define all the direct subclasses of A and B to be direct subclasses of the unified class. To illustrate this definition, we consider some cases including those discussed in /CHID94/

.

The first case discussed by Chidamber et. al. assumes two Classes A and B having the same direct superclass (father) V. The resulting class has the same father V. The direct subclasses (children) of A and B respectively also are children of the unified class. Figure 4.5 illustrates this case.

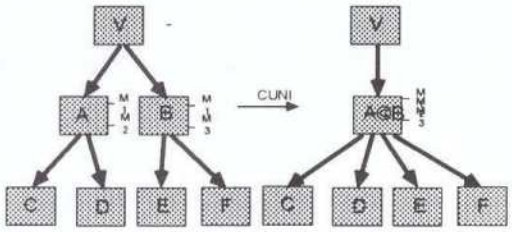


Figure 5.6: Unification CUNI (A,B) of sibling classes.

If Class B is a direct subclass of Class A, these will be combined as follows. The father V of A also was an indirect superclass of B and is now the father of CUNI(A,B). The children are again all children of A and B, in this case the children of B do not inherit more properties as they inherited already the properties of A via B (See Figure 4.6).

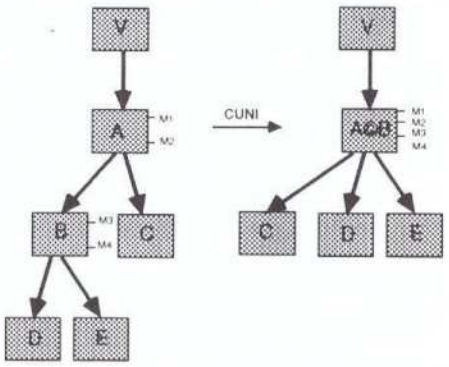


Figure. 5.7: Unification CUNI(A,B) with a direct subclass.

Classes that have separate fathers have to be combined as shown in figure Figure 4.7. The combined class inherits from both superclasses via multiple inheritance. The children are again all the children of A and B, here C multiply inherits from A and B, of course it is only once a child of CUNI(A,B).

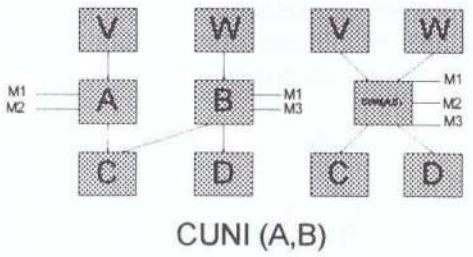


Figure 5.8: Unification CUNI(A,B) of unrelated classes.

We now consider the implications of CUNI for uses relations. The behavior is very similar as described above for inheritance. All the classes used by either one ore both of the classes combined will be used by the unified class. This lies already in the unification of the properties defined in the class. Analogously, we define all classes that used any single or both separate classes to use the combined class. This won't cause problems as those classes need not use the additional parts of the new class but the original parts of that class already used. The figures below illustrate this case with two examples.

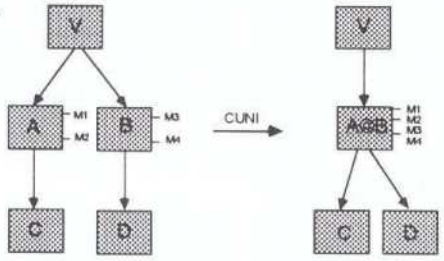


Figure 5.9: Concatenation operation CUNI(A,B) with uses relationships.

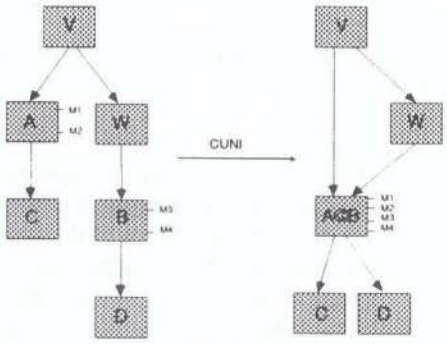


Figure 5.10: CUNI with uses relationships

We now consider the properties of the concatenation operation CUNI.

Properties of CUNI

Since the unification of the sets of attributes and the sets of methods are independent of the ordering in which these sets are unified, and no order is defined neither for the uses nor inheritance relation, CUNI is commutative and it is also associative.

Interesting is, that CUNI is also idempotent:

$$A \circ A = A,$$

where A is a set of objects. Assuming idempotency by a concatenation operation has the consequence, that the a measure does not assume an extensive structure which has the consequence that we cannot come to the ratio scale.

Of course, this property of CUNI is not by an accident. CUNI has been defined as a *unification* and we know that unification of sets is idempotent. This reflects an advantage of object-oriented design. Similar classes should be combined to one class with the consequence of less maintenance effort.

Chidamber et. al. check, whether their measures agree with the Weyuker properties. If we keep in mind, that CUNI is idempotent, it is not surprising that none of their six measures agree with wholeness, which is defined as:

$$u(A \circ B) \geq u(A) + u(B),$$

for all A, B $\in$ **A.** If idempotency is fulfilled, we cannot have wholeness, as Weyuker requires that. In fact, if A=B, we obtain for any measure μ

$$u(CUNI(A,A) = u(A).$$

As we mentioned above, this is due to CUNI is a kind of an unification. Our suggestion is to add a second combination operation that corresponds to intersection of sets.

5.5.5 Class Intersection (CINT)

We have seen, that the unification of classes leads to a class which includes all the properties, methods and attributes which are common for the both single classes.

The result of an intersection of two classes is that the common properties of both classes are reduced to one property in the combined class. We illustrate this with the following picture. Assume two Classes A and B with variables and methods. The new Class C, combined by Classes A and B only consists of the variable a and method M1.

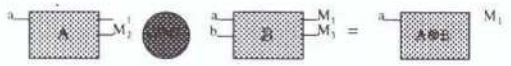


Figure 5.11: CINT: Intersection of classes.

As for CUNI, we now consider the implications of CINT for the inheritance relationship. It is important to say that CINT is not a combination rule to build new stand alone classes. We are discussing the consequences for a concatenation operation which is defined as CINT. Later we will discuss generalization, then we will see an application of CINT in reality.

CINT intersects the sets of properties of classes. We now require that the intersected class inherits all the properties that both classes inherited. These properties are again common to both classes. And we require that every subclass that are common to both classes will be subclasses of the intersection.

That leads to somewhat non-intuitive situations as shown in Figure 4.21. Here, G is an indirect superclass of A and B and so the intersected class CINT (A,B) will inherit from G but neither from V nor W. And the only common subclass is E, which becomes a child of the intersection.

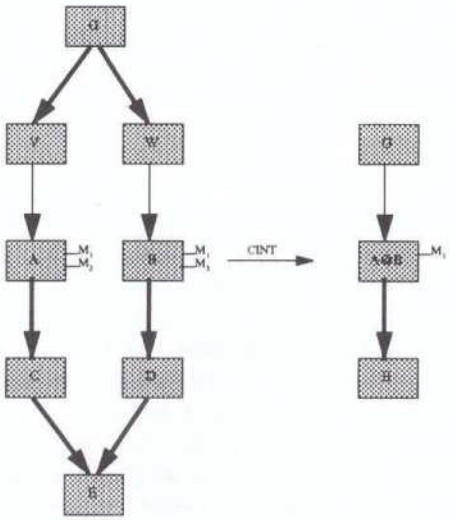


Figure 5.12: Class intersection and inheritance relationships.

Again, for the uses relation our consideration are very similar. Only those classes which use both classes use the intersection and as an implication of the intersection of properties. Only those classes used by both classes will be used by CINT(A,B). Unlike inheritance indirect use does not propagate along the vertices of the uses relation.

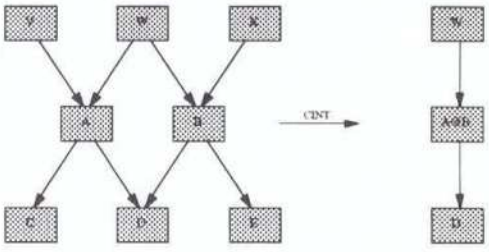


Figure 5.13: Class intersection and uses relationships.

We now have two combination operations CUNI and CINT among classes and we find a couple of measures including some of the measures proposed by Chidamber et. al. holding the following condition

$$\text{C-NOM(CUNI (A,B))} = \text{C-NOM(A)} + \text{C-NOM(B)} - \text{C-NOM(CINT(A,B))}.$$

where C-NOM is the number of methods in a class.

5.5.6 The Empty Class

We now define the empty Class ∅ to be a class with no properties, i. e. empty sets of attributes and methods. Furthermore, the empty class may not inherit any property and no class inherits from ∅ . It is also not used by any class and of course does not use any class. Having defined an empty class we can say that two classes A and B are disjunct, i. e. they don't have any property in common. It holds CINT(A,B) = ∅

6 Belief Structures

As we have shown above, measures which are additive, assume an extensive structure. They can be used as an additive ratio scale. In the object-oriented environment the behavior of measures related to concatenation operations is different. Many measures do not assume an extensive structure which has the consequence that they cannot be used as an (additive) ratio scale. Other measures assume idempotency, which implies, that they do not assume the Archimedian axiom, which implies no extensive structure, either.

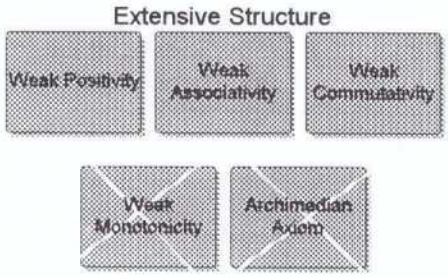


Figure 6.1: No Extensive Structure for the most object-oriented measures.

Mostly, the axioms of weak monotonicity and the Archimedian axiom are violated. In order to discuss measurement structures in the object-oriented area above the ordinal scale level, we consider qualitative and quantitative probabilities and belief structures or belief functions. The goal of discussing such properties is to get an empirical interpretation of number above the ordinal scale level.

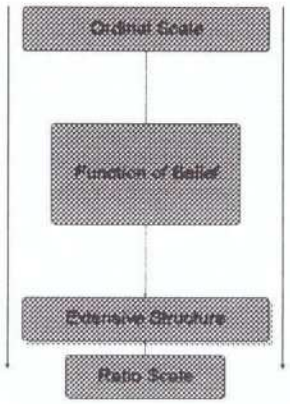


Figure 6.2: Ordinal scale, function of belief, and extensive structure.

The picture above shows the goal of the new axiom systems. Having only an ordinal scale level, we have a low level of measurement structures. We are far away from the empirical conditions of the extensive structure. The axioms of the function of belief shall help to get more empirical conditions for the interpretation of measurement values above the ordinal scale.

6.1 Belief Functions

There exist a well known quantitative approach, the theory of belief functions, which has generated considerable interest in recent years /SHAF76/, /SHAF87/. In this theory, belief may be interpreted as a generalization of probability. Also belief functions provide a useful and effective tool for the quantification of subjective, personal judgments. It is the view of many researchers that humans frequently reason in qualitative rather quantitative terms /BHAT86/. A function of belief describes the belief in a hypothesis and the belief in combination of hypotheses. More information's about belief functions can be found in /SHAF76/, /BUER87/ and /WONG91

In 1995 Zuse et al. /ZUSE95/ presented an approach for characterizing object-oriented software measures with a modification of the function of belief. We now leave out all the mathematics of belief functions and discuss some thought experiments (Gedankenexperimente) related to object oriented software measures. A detailed discussion can be found in Fetcke /FETC95/. We illustrate this with some pictures.

6.1.1 Empirical Properties based on Belief Functions

Before we can consider empirical conditions for object-oriented software measures, we have to define the sets of attributes and methods.

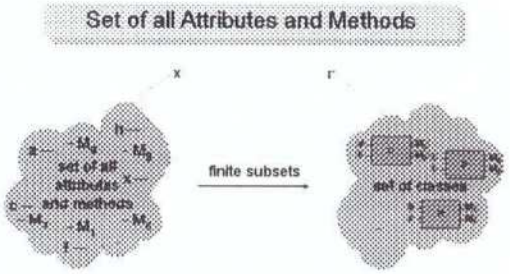


Figure 6.3: Explanation of the sets X and the set of Classes Γ.

The picture above shows the used sets for our investigation. The left *cloud* shows the set of all attributes and methods. The right *cloud* shows the set of classes, which structure the attributes and methods.

We now consider special cases / conditions of the behavior of classes. From /ZUSE95/ we present some empirical axioms. The first axioms is called the axiom partial monotonicity.

6.1.2 Partial Monotonicity of Qualitative Belief

The axiom / condition of partial monotonicity gives criteria for an empirical interpretation of software measures above the ordinal scale level. We can investigate object-oriented software measures whether they fulfill the property below or not.

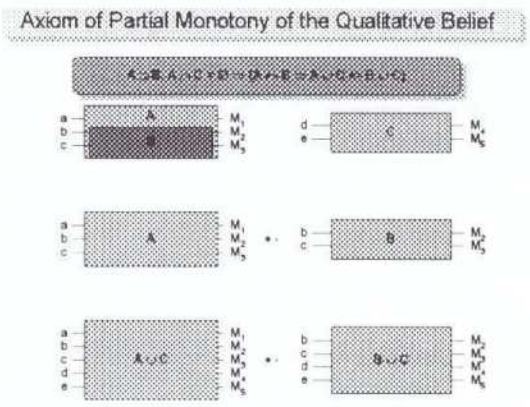


Figure 6.4: Axiom of partial monotonictiy.

Assume we have a Class A consisting of attributes and methods. Class consists of the attributes a, b, c and the methods M1, M2, and M3. Class B consists of the attributes b, c and the methods M1 and M2. It follows that Class B is a real subset of A. We write this as $A \supset B$.

Furthermore, we have a third Class C, consisting of the attributes d, e, and the methods M4 and M5. We see, that Class C does not have any attributes or methods common with Class A. The intersection of the both Classes A and C is the empty set. We write this as $A \cap C = \emptyset$.

We now assume that we have an empirical relation $\bullet\geq$. We denote this relation as *equally or more difficult to maintain*. Assuming $A \bullet\geq B$, what means Class A is *equally or more difficult to maintain* than B, it follows that $A \cup C$ is also equally or more difficult to maintain than $B \cup C$. We write this as $A \bullet\geq B \Rightarrow A \cup C \bullet\geq B \cup C$. The Classes in Rows 2 and 3 show this behavior. If a user has this view of maintenance of classes then he has to look for a Measure u which fulfills his view. We can write this as:

$$A \bullet\geq B \Rightarrow A \cup C \bullet\geq B \cup C \Leftrightarrow$$
$$u(A) \bullet\geq u(B) \Rightarrow u(A \cup C) \bullet\geq u(B \cup C).$$

We call this a homomorphism related to the axiom of partial monotonicty. It is important to mention that the condition above only holds in the direction of =>.

6.1.3 Dominance Axiom

The next picture illustrates the dominance axiom.

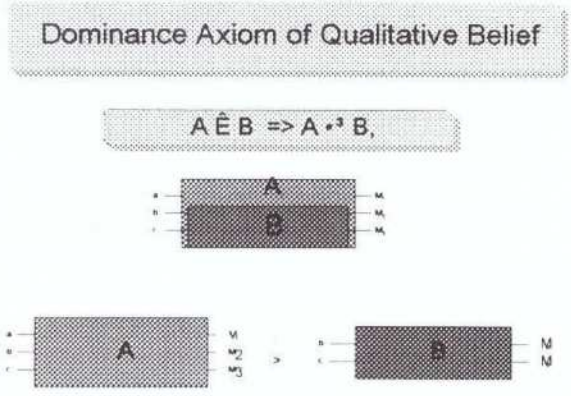


Figure 6.5: Dominance axiom.

The dominance axiom shows the following: Assume Class B is a subset of Class A. Then it

follows that Class A is equally or more difficult to maintain than Class B. We write this as: $A \supseteq B \Rightarrow A \bullet\geq B$. This is another important criterion for an object-oriented software measure.

6.1.4 Partial Monotonicity of DeFinetti

We now show one axiom derived by the DeFinetti axioms. It is called axiom of partial monotonicity. We illustrate this with a picture.

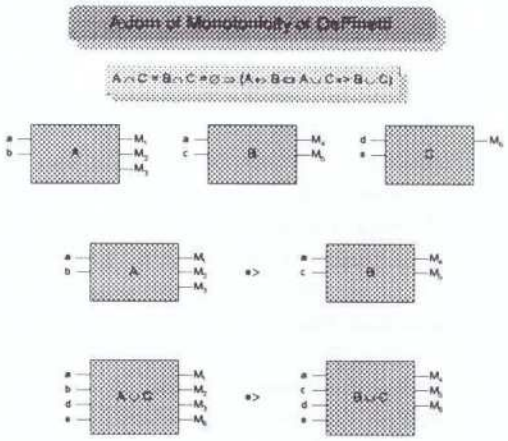


Figure 6.6: Axiom of Monotonicity of the DeFinetti axioms.

The axiom of partial monotonicity of DeFinetti is stronger than the axiom of partial monotonicity as discussed above. Assume, we have three Classes A, B and C with attributes and methods. We also assume it holds that the Classes A and C, and B and C do not have common attributes and methods. We write this as: $A \cap C = \emptyset$, and $B \cap C = \emptyset$. Both together we write as: $A \cap C = B \cap C = \emptyset$.

We assume that Class A is equally or more difficult to maintain than B ($A \bullet\geq B$). If holds $A \bullet\geq B$ then we expect that also holds: $A \cup C \bullet\geq B \cup C$. We write the complete axiom as:

$$A \bullet\geq B <=> A \cup C \bullet\geq B \cup C.$$

This axiom holds for both directions. It is illustrated in the picture above in Rows 2 and 3.

6.1.5 Modified Function of Belief

Finally, we show a stronger condition, called the modified function of belief, which includes the conditions above.

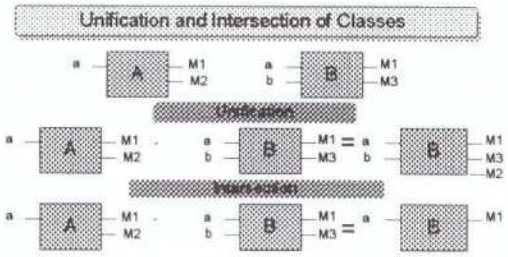


Figure 6.7: Unification and Intersection of Classes A and B.

In the most cases object-oriented measures fulfill the following condition

$$u(A \cup B) = u(A) + u(B) - u(A \cap B).$$

The unification of Class $A \cup B$ and the intersection of Classes $A \cap B$ are shown in the picture above. The discussed conditions above are implied by this condition. The condition above says that a measure which follows the condition above is only additive if no common attributes and methods are existing. That is mostly not the case in the object-oriented environment.

6.1.6 Summary

We now see, that we have additional axioms / conditions / criteria to analyze object-oriented software measures. The starting point for our investigation was a set of attributes and methods which we did structure to classes. Then we presented four conditions and discussed them. These conditions describe object-oriented software measures above the ordinal scale level.

7 Investigations of some Object-Oriented Measures

We now apply the discussed conditions above to the Measure C-CBO of Chidamber et al. /CHID94/.

Measure C-CBO

The Measure CBO of Chidamber et al. /CHID94/. It is defined as:

C-CBO (C) is the number of classes that are coupled with the Class C. Two classes are coupled to each other if methods of one class methods or instance variables of other classes use. We illustrate this with an example.

In the picture below Class A is coupled with the four Classes V, W, C, and D. It holds C-CBO(A) = 4.

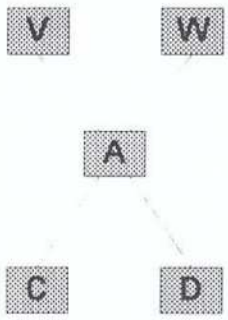


Figure 7.1: Class A is coupled with four other classes. Class A is used by the Classes V and W and Class A uses itself the Classes C and D. It holds CBO(A) = 4.

The behavior of this measure can be described with the class intersection. The modified axioms of the function of belief are fulfilled, it holds for the Classes A and B.

CBO (CUNI(A, B)) = CBO (A) + CBO (B) - CBO (CINT(A , B)).

We demonstrate this with the following example.

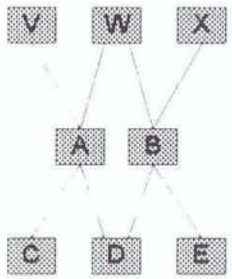


Figure 7.2: CBO related to the concatenation operations CUNI and CINT.

Figure 7.3: CBO related to the concatenation operations CUNI and CINT.

Figure 7.4: CBO related to the concatenation operations CUNI and CINT.

It holds:

CBO (A) = 4
CBO (B) = 4
CBO (CINT(A,B)) = 2
CBO (CUNI(A,B)) = 4 + 4 - 2 = 6.

The consequence is, that the Measure C-CBO fulfills the condition

$$u(A \cup B) = u(A) + u(B) - u(A \cap B).$$

In /ZUSE95/ and /FETC95/ more measures are investigated with the properties above.

8 Results

We now summarize the results of our investigation:

1. The Measures LOC is additive related to the sequential concatenation operation BSEQ which implies that the Measure LOC assumes the conditions of the extensive structure. The conditions of the extensive structure are the weak order, the axiom of weak positivity, the axiom of weak associativity, the axiom of weak commutativity, the axiom of weak monotonicity, and the Archimedian axiom.

2. The Measure LOC and the Measure MCC-V2 can be used as a ratio scale.

3. Object-oriented measures have more complicated properties related to concatenation operations. In the area of object-oriented measures we can define many concatenation operations, like CUNI and CINT. Concatenation operations are useful to get more informations of the meaning of the numbers of measures above the ordinal scale level.

4. We demonstrated that object-oriented measures mostly do not assume an extensive structure which blocks the way to the ratio scale. The reason is idempotency, which implies that the Archimedian axiom is not fulfilled. .

5. We showed the properties for object-oriented techniques if the investigated measures assume intersection and unification properties.

6. We mentioned, that measures in the object-oriented environment have the property of idempotency and the properties of dominance, partial monotonicity and monotonicity of DeFinetti. These conditions help the user of object-oriented measures to understand their properties in a better way.

9 Conclusion

Object-oriented software measures assume an extensive structure. We showed, that in the area of object-oriented measures we can find properties of idempotency, unification and intersection of sets, the function of belief, and the De Finetti axioms.

Attachment: Foundations of Software Measurement

In this attachment we introduce basic concepts of measurement theory and show the application to software measures. We use the text of Krantz et al. /KRAN71/, Luce et al. /LUCE90/ and Roberts /ROBE79/. Proofs of the theorems and a more detailed discussion of measurement theory related to software measures can be found in /ZUSE91/, /ZUSE92/, /ZUSE92b/, /ZUSE94c, /ZUSE95/, and /BOLL93/. An application of measurement theory to more than ninety intra-modular software complexity measures can also be found in /ZUSE91/.

It should be mentioned here that it is a widely spread misunderstanding that measurement theory is only useful to determine the scales like nominal, ordinal, interval and ratio scale. This is more a side-effect of measurement theory. The major advantage of measurement theory lies in hypotheses about reality.

1.1 Basic Concepts of Measurement Theory

First of all we want to introduce the notation of an empirical, a numerical relational system and a scale. Let

$\mathbf{A} = (A, \bullet\geq, o)$

be an empirical relational system, where A is a non-empty set of empirical objects, •> is an empirical relations on A and o a binary operation on A (Of course, there are more than one relation and binary operation possible).

According to Luce et al. /LUCE90/, p.270, we assume for an empirical relational system **A** *that there is a well-established empirical interpretation for the elements of* ***A*** *and for each relation Si of* ***A***. We also assume the same for the binary operations.

Let further

$\mathbf{B} = (\Re, \geq, +)$

be a formal relational system, where $\Re$, are the real numbers, ≥, a relation on B, and + a closed binary operation on $\Re$. (Of course, there are more than one relation and binary operations possible). We also include the case that there are no relations or no operations.

The sign + means here the following: for a concatenation of two Flowgraphs P1 o P2 holds the following formula:

$$u(P1 \circ P2) = u(P1) + u(P2),$$

where u is a software measure and P1, P2ε **P**

A **measure** is a mapping u: A -> B such that the following holds for all P1, P2 ε A:

$$P1 \bullet\geq P2 \Leftrightarrow u(P1) \geq u(P2)$$
and
$$u(a \circ b) = u(a) + u(b)$$

Then the Triple (**A**, **B**, u) is called a **scale**. According to this definition we see that measurement assumes a homomorphism.

Given two relational system **A** and **B** we can ask whether there exists a measure such that (**A**, **B**, u) is a scale. This problem is called the **representation problem**. If such a measure exists we can ask how uniquely the measure is defined. This problem is called the **uniqueness problem**. The uniqueness problem leads to the definition of scale types such as ordinal or ratio scale.

Let g: X ->Y and h: Y -> Z be mappings. Then hg denotes the composed mapping hg(x)=h(g(x)) for x ε X.

Definition 1.1 (Admissible Transformation):

Let (**A**, **B**, u) be a scale. A mapping g: A -> B is an admissible transformation iff (**A**, **B**, g) is also a scale.

Real scales are classified according to the admissible transformations.

Name of the Scale	**Transformation g**
Nominal Scale	Any one to one g
Ordinal Scale	g: Strictly increasing function
Interval Scale	$g(x) = a\,x + b, \quad a > 0$
Ratio Scale	$g(x) = a\,x, \quad a > 0$
Absolute Scale	$g(x) = x$

Figure 1.1: Scale types of real scales. It is a hierarchy of scale types. The lowest one is the nominal scale and the highest one is the absolute scale.

Meaningfulness

However, the major question for the user is: how does he know what scale type is assumed and what are the conditions for the use of a measure on a certain scale level. Or equivalently, how does a measure and reality look like which creates numbers which can be transformed by a certain admissible transformation of scales.

Admissible transformations also lead to the definitions of meaningful statements /ROBE79/, p.58. *A statement with measurement values is meaningful iff its truth or falsity value is invariant to admissible transformations.* Meaningfulness guarantees that the truth value of statements with measurement values is invariant to scale transformations. For example if we say that the distance D1 is twice as long as distance D2, then this statement is true or false no matter whether length is measured in meters or yards. These problems are existing in the area of software measures too. This is the case if we want to make statements with measurement values for example after having applied statistical methods. We want to explain this problem by a statement which was given by Pressman /PRES92/ (Relation >) and similarly by Weyuker /WEYU88/.

Example 1.1 (Wholeness)

Let P1 and P2 be program bodies combined in some way to P1 o P2 (See for an example Section 2.3 and Figure 2.2). Let be a software complexity measure. Then the requirement for software complexity by Pressman is

$$u(P1 \circ P2) > u(P1) + u(P2).$$

This statement is not meaningful for an interval scale. If we apply the admissible transformation of the interval scale g(x)=a x+b, then we get:

$$a\,u(P1 \circ P2) + b > a\,u(P1) + b + a\,u(P2) + b,$$

for all $a>0$ and for all $b \varepsilon \Re$. Hence, the truth or falsity of the statement is not invariant to this type of admissible transformation. Hence the statement is meaningful for a ratio scale.

We see that meaningfulness depends on admissible transformations. The admissible transformations depend on the relational systems under consideration. Hence it is important to study the conditions which should hold on the relational systems in order to have a certain class of admissible transformations or equivalently to have a certain scale type.

In Bollmann and Zuse /BOLL93/ and Zuse /ZUSE94a/ we showed that wholeness is a pseudo property without any empirical meaning. we also showed that there can exists ratio scales which are not additive.

1.2 Ordinal and Ratio Scale

We now introduce the conditions for the use of software measures as an ordinal and a ratio scale. Firstly, we consider the conditions of the ordinal scale.

1.2.1 Ordinal Scale

In order to describe a measure as an ordinal scale we introduce the **weak order** which is a binary relation that is transitive and complete:

$P \bullet\!\geq P'$, $P' \bullet\!\geq P'' \Rightarrow P \bullet\!\geq P''$ transitivity

$P \bullet\!\geq P'$ or $P' \bullet\!\geq P$ completeness (connectedness),

for all P, P', P'' ε **P**, where **P** is the set of flowgraphs ($\bullet\!\geq$ is a binary empirical relation, like *equal or more complex).*

In /ROBE79/, p.110, we find the following theorem which we can be applied directly to flowgraphs.

Theorem 1.1:

Suppose (**P**, $\bullet\!\geq$) is an empirical relational system, where **P** is a non-empty countable set

of flowgraphs and where •≥ is a binary relation on **P**. Then there exists a function u: P -> ℜ, with

$$P \bullet\geq P' <=> u(P) \geq u(P')$$

for all P, P' ε **P**, iff •≥ is a weak order. If such a homomorphism exists, then

$$((\mathbf{P}, \bullet\geq), (\Re, \geq), \mathbf{u}))$$

is an ordinal scale.

1.2.2 Ratio Scale

In order to come to the ratio scale the relational system (**P**, •≥) has to be extended to (**P**, •≥, o). We want to introduce the following notation for P, P'ε **P**: P ≈ P' iff P •≥ P', and P' •≥ P, P •> P' iff P •≥ P', and not P' •≥ P, where •≥ is the weak order and means equally or more complex, •> means more complex, and ≈ means equally complex.

Theorem 1.2 /KRAN71/, p.74:

Let **P** be a non empty set, •≥ is a binary relation on **P**, and o a closed binary operation on **P**. Then (**P**, •≥, o) is a closed extensive structure iff there exists a real-valued function on P such that for all a, b ε **P**

$$(1)\ a \bullet\geq b <=> u(a) \geq u(b)$$
and
$$(2)\ u(a \circ b) = u(a) + u(b)$$

Another function u' satisfies (1) and (2) iff there exists >0 such that

$$u'(a) = u(a).$$

The statement u' (a) = u(a) gives the admissible transformation for an additive ratio scale. We see that Theorem 1.2 gives us conditions for the additive ratio scale.

In Zuse /ZUSE91/, p.57 an extensive structure, as proposed by Bollmann /BOLL84/, is presented.

Definition 1.3: (Extensive Structure):

Let **P** be a non-empty set, •≥ binary relation on **P**, and o a closed binary operation on **P**. The relational system (**P**, •≥, o) is an **extensive structure** if and only if the following axioms hold for all P1,..,P4 ε **P**.

A1': (**P**,•≥) is a weak order

A2': P1 o (P2 o P3) ≈ (P1 o P2) o P3, axiom of weak associativity

A3': P1 o P2 ≈ P2 o P1, axiom of weak commutativity

A4': P1 •≥ P2 => P1 o P3 •≥ P2 o P3 axiom of weak monotonicity

A5': If P1 •> P2 then for any P3,P4 there exists a natural number n, such that nP1 o P3 •> nP2 o P4, Archimedian Axiom
®

The axioms of the extensive structure describe empirical conditions related to concatenation operations of objects. The axiom of weak positivity is also an axiom of the extensive structure. In the next Section we will show how this approach can be applied to software complexity measures.

1.3 Measurement Theory and Software Measures

We now show very briefly how measurement theory can be applied to software complexity measures. We will illustrate this with the Measures of McCabe. We specialize the general measurement theory approach for software complexity. We consider the empirical relational system (**P** , •≥) for the ordinal scale and (**P** , •≥, o) for the ratio scale, where **P** is the set of flowgraphs, •≥ the binary relation *equal or more complex than*, and o is a binary operation.

In measurement theory a concatenation operation is defined as:

$$P \times P \rightarrow P.$$

where A is the set of flowgraphs. As binary operations we consider here the sequential combination BSEQ for flowgraphs.

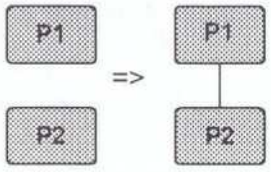


Figure 1.2: The binary operation BSEQ=P1 o P2 of two arbitrary flowgraphs P1 and P2, and the sequential concatenation operation of a structured chart.

1.3.1 Application of Measurement Theory to the Measure of McCabe

Firstly, we choose the Measures of McCabe /ZUSE91/, p.151, with

$$MCC\text{-}V=|E| - |N| + 2 \text{ , and}$$

$$MCC\text{-}V2=|E| - |N| + 1$$

to demonstrate our method. McCabe derived a software complexity measure from graph theory using the definition of the cyclomatic number. McCabe interpreted the cyclomatic number as the "minimum number of paths" in the flowgraph. He argued that the minimum number of paths determines the complexity (Called by McCabe: cyclomatic complexity) of the program: *The overall strategy will be to measure the complexity of a program by computing the number of linearly independent paths v(G), control the "size" of programs by setting an upper limit to v(G) (instead of using just physical size), and use the cyclomatic complexity as the basis for a testing methodology.*

The cyclomatic number is only a value of a mathematical function. Considering the Triple (**A, B, u**) for a scale, only the part (**B,**) is used, but measurement considers always the Triple (**A**, **B**, u) which includes the empirical relational system **A**. However, interpreting the cyclomatic number as the complexity of a flowgraph gives this number an empirical interpretation. Having an empirical interpretation of the numbers (cyclomatic complexity) the use of measurement theory is justified. We consider the empirical relational system (**P**, $\bullet\geq$) for the ordinal scale and (**P**, $\bullet\geq$,o) for the ratio scale, where **P** is the set of flowgraphs, o is the sequential combination BSEQ of flowgraphs as defined in Figure 1.2, and MCC-V2: is the additive Measure MCC-V2=|E| - |N| + 1 of McCabe with

$$MCC\text{-}V2(P1 \circ P2) = MCC\text{-}V2(P1) + MCC\text{-}V2(P2).$$

1.3.1.1 The Measure of McCabe as an Ordinal Scale

The Measure MCC-V2 can be used as an ordinal scale if the empirical ranking order corresponds with the ranking order of the Measure of McCabe. The Measure MCC-V=|E|-|N|+2 of McCabe is a strictly monotonic transformation of the Measure MCC-V2. A strictly monotonic function is the admissible transformation of the ordinal scale, it does not change the ranking order.

1.3.1.2 The Measures of McCabe as a Ratio scale

In order to give the conditions for the use of the Measure MCC-V2 as a ratio scale we use Theorem 1.2. If o is the sequential combination BSEQ of flowgraphs, if complexity is transitive and complete and if the conditions e1, e2 and e3 hold (what means that

$$((A, \bullet\geq), (\Re, \geq), MCC\text{-}V2)$$

is an ordinal scale) then

$$(\mathbf{P}, \bullet\geq, o)$$

is a closed extensive structure. In this case

$$((\mathbf{P}, \bullet\geq, o), (\Re, \geq, +), MCC\text{-}V2),$$

where MCC-V2=|E| - |N| + 1 is an additive ratio scale. The Measure MCC-V2 is a strictly monotonic transformation of: MCC-V. In other words: the Measure MCC-V2=|E| - |N| + 1can be used as a ratio scale because it is additive related to the concatenation operation BSEQ.

1.3.1.3 The Measure of McCabe as an Absolute Scale

There is a confusing situation in literature about the absolute scale. It is not true that the absolute scale is described by counting. We now show the conditions for the use of the Measure of McCabe as an absolute scale. Very often people say that the Measure of McCabe measures the cyclomatic number and that is an absolute scale. That is wrong, because we can transform the cyclomatic number by an admissible transformation. We can say that we have twelve independent path, but we also can say that we have one dozen independent paths.

In order to characterize the Measure of McCabe as an absolute scale, we have to introduce an additional concatenation operation BALT. Above, we introduced the sequential concatenation operation BSEQ, additionally we introduce the concatenation operation BALT, which is defined as:

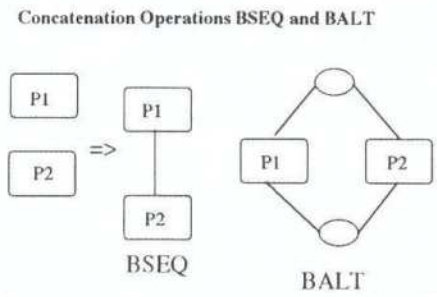


Figure 1.5: Concatenation operations BSEQ and BALT, where two Flowgraphs P1 and P2 are combined to a new Flowgraph BSEQ = P1 o P2 or BALT = P1 • P2, with P1, P2 ∈ **P.**

In order to come to the absolute scale we consider the following empirical relational system:

$$\mathbf{A} = (P, \bullet\geq, o, \bullet),$$

where P is the set of flowgraphs, •≥ an empirical relation, like equally or more difficult to maintain, o is the sequential concatenation operation BSEQ and • the concatenation operation BALT. We consider the Measure MCC-V2 of McCabe which is defined as:

$$\text{MCC-V2 (P)} = |E| - |N| + 1.$$

For the sequential concatenation operation BSEQ = P1 o P2 the combination rule is:

$$\text{MCC-V2 (P1 o P2)} = \text{MCC-V2 (P1)} + \text{MCC-V2(P2)}.$$

It is easy to see that the Measure MCC-V2 is additive related to the concatenation operation BSEQ. The combination rule for the concatenation operation BALT = P1 • P2 is:

$$\text{MCC-V2 (P1 • P2)} = \text{MCC-V2(P1)} + \text{MCC-V2 (P2)} + 1.$$

We now consider whether both combination rules are meaningful related to admissible transformation the ratio scale. It is easy to see that the combination rule for the concatenation operation BSEQ is meaningful related to the admissible transformation of the ratio scale. We illustrate this here:

$$a\,\text{MCC-V2 (P1 o P2)} = a\,\text{MCC-V2 (P1)} + a\,\text{MCC-V2(P2)}.$$

We transform each measurement value with the admissible transformation of the ratio scale (a>0). It is easy to see that the combination rule is invariant to the admissible transformation of the ratio scale.

We now consider the combination rule for the concatenation operation BALT and apply the admissible transformation of the ratio scale.:

$$a\,\text{MCC-V2 (P1 • P2)} = a\,\text{MCC-V2(P1)} + a\,\text{MCC-V2 (P2)} + 1.$$

It is easy to see that the statement above is not invariant to the admissible transformation of the ratio scale (a>0). The statement above is invariant for a=1, and that is the admissible transformation of the absolute scale.

This shows that the Measure MCC-V2 can be used as an absolute scale if we consider the following empirical (**A**) and numerical (**B**) relational systems:

$$\mathbf{A} = (P, \bullet\geq, o, \bullet), \text{ and}$$
$$\mathbf{B} = (R, \geq, +, \text{MCC-V2(P1)} + a\,\text{MCC-V2 (P2)})$$

and the Measure MCC-V2 To use a measure as an absolute scale depends on the behavior of the concatenation operations related to the admissible transformations of the ratio scale. If one of the both combination rules are not invariant to the admissible transformation of the ratio scale, then we have an absolute scale (without any proof).

2 Concatenation Rules and Independence Conditions

Another important criteria for software measures are the independence conditions which are based on concatenation rules /ZUSE92/. Let us assume that $((A, \bullet\geq),(\Re, \geq), \mathbf{u})$ is an ordinal scale. We want to discuss the question whether there exists a binary operation • such that

$$u(P1 \text{ o } P2) = u(P1) \bullet u(P2)$$

for all P1, P2 ε A. We denote • as a combination rule. The formula above is identical with the formula

$$\mu(P1 \text{ o } P2) = f(\mu(P1), \mu(P2)).$$

The answer is given in the following theorems.

Theorem C1: There exists such a • iff P1≈ P2 => P1 o P ≈ P2 o P, and P1 ≈ P2 => P o P1 ≈ P o P2, for all P1, P2, P ε **P**

Theorem C2:
There exists such a • iff P1 ≈ P2 <=> P1 o P ≈ P2 o P, and P1 ≈ P2 <=> P o P1 ≈ P o P2, for all P1, P2, P ε **P**

Theorem C3: There exists such a • iff P1 •≥ P2 => P1 o P •≥ P2 o P, and P1 •≥ P2 => P o P1 •≥ P o P2, for all P1, P2, P ε **P**

Theorem C4: There exists such a • iff P1 •≥ P2 <=> P1 o P •≥ P2 o P, and P1 •≥ P2 <=> P o P1 •≥ P o P2, for all P1, P2, P ε **P**

The condition C1-C4 are hierarchical ordered, that means C4 implies C3, C2, and C3, C2 imply C1. The independence conditions are very important in the context of prediction and validation of measures. For more information see /ZUSE94a/.

3 Validation and Prediction

If we have two ratio scales, one of the measure M and the other one of the external variable V, then the only one function which is possible between two ratio scales is the following formula:

$$V(P) = a\,M^{b}.$$

with a,b>0. This formula has to be validated if we want to use a measure, which assumes a ratio scale, as a predictor for an external variable C, which can be also used as a ratio scale. The formula above is known as the basic COCOMO-Model /BOEH81/. For more information's see Bollmann et al. /BOLL93/ and Zuse /ZUSE94a/.

10 References

/ABRE93/ Abreu, F.B.:
Metrics for Object-oriented Environment. Proceedings of the Third International Conference on Software Quality, Lake Tahoe, Nevada, October 4-6, 1993, pp. 67-75.

/ABRE94/ Abreu, F. B.:
Object-Oriented Software Engineering: Measuring and Controlling the Development Process. Proceedings of the 4th International Conference on Software Quality, Washington D.C., October 3-5, 1994

/BARN93/ Barns, Michael, G.:
Inheriting Software Metrics. Journal of Object-Orriented Programming, November-December 1993, pp. 27-34.

/BASI84/ Basili, V.; Perricone, Barry T.:
Software Errors and Complexity: An Empirical Investigation. Communications of the ACM, Volume 27, No. 1, January 1984, pp. 42-52.

/BIEM91/ Bieman, J.M.:
Deriving Measures of Software Reuse in Object Oriented Systems. Technical Report #CS91-112, July 1991, Colorado State University, Fort Collins/ Colorado, USA.

/BOLL84/ Bollmann, Peter:
Two Axioms for Evaluation Measures in Information Retrieval. Research and Development in Information Retrieval, ACM, British Computer Society Workshop Series, pp. 233-246, 1984

/BOLL93/ Bollmann-Sdorra, P.; Zuse, H.:
Prediction Models and Software Complexity Measures from a Measurement Theoretic View. Proceedings of the 3rd International Software Quality Conference, Lake Tahoe, Nevada, October 4-7, 1993.

/BOOC91/ Booch, G.:
Object-Oriented Design with Applications. Benjamin/Cummings, 1991.

/BUER87/ Bürger, Hans-Christoph:
Axiomatische Beschreibung von Glaubensfunktionen. Diploma Thesis, TU-Berlin, Dezember 1987.

/CALD91/ Caldiera, G.; Basili, V.:
Identifying and Quantifying Reuseable Software Components. IEEE Software, Feb. 1991, pp. 61-70. Also in: Arnold, Robert, S.: Software Reegineering. IEEE Computer Society, 1992, pp. 485-494.

/CHEN93/ Chen, J.Y.; Lu, J.F.:
A new Metric for Object-oriented Design. Journal of Information and Software Technology, Volume 35., No. 4, April 1993, pp.232-240.

/CHID94/ Chidamber, Shyam, R.; Kemerer, Chris, F.:
A Metrics Suite for Object Oriented Design. IEEE Transactions on Software Engineering, Volume 20, No. 6, June 1994, pp. 476-493.

/DUMK92/ Dumke, Reiner:
Softwareentwicklung nach Maß- Schätzen - Messen - Bewerten. Vieweg Verlag, 1992.

/DUMK94/ Dumke, Reiner; Zuse, Horst:
Software-Metriken in der Objektorientierten Software-Entwicklung. In: Editor: Prof. Lehnert: Wartung von Wissensbasierten Systemen, Hänsel-Hohenhausen, Deutsche Hochschulschriften 561, 1994.

/FETC95/ Fetcke, Thomas:
Software Metriken bei der Object-orientierten Programmierung. Diploma thesis, Gesellschaft für Mathematik und Datenverarbeitung (GMD), St. Augustin, and TU-Berlin, 1995.

/HALS77/ Halstead, M.H.:
Elements of Software Science. New York, Elsevier North-Holland, 1977.

/ISO.91a/ ISO/IEC Standard:
ISO 9126 Software Product Evaluation - Quality Characteristics and Guidelines for Their Use, 1991.

/JENK93/ Jenkins, J.:
Software Metrics won't eliminate the Productivity Crisis. American Programmer, Volume 6, Feb. 1993, pp. 2-5.

/JENS91/ Jensen, R.; Barteley, J.:
Parametric Estimation of Programming Effort: An Object-oriented Model. Journal of Systems and Software, Volume 15, 1991, pp. 107-114.

/KARN93/ Karner, Gustav:
Metrics for Objectory. Master Thesis at the Linkšping University, S-581 83 Linkšping, Sweden, 1993.

/KOLE93/ Kolewe, Ralph:
Metrics in Object-Oriented Design and Programming. Software Development, October 1993, pp. 53-62.

/KRAN71/ Krantz, David H.; Luce, R. Duncan; Suppes; Patrick; Tversky, Amos:
Foundations of Measurement - Additive and Polynominal Representation, Academic Press, Volume 1, 1971

/LAKE92/ Lake, Al:

A Software Complexity Metric for C++. Annual Oregon Workshop on Software Metrics, March 22-24, 1992, Silver Falls, Oregon, USA.

/LAKE94/ Lake, Al:
Use of Factor Analysis to Develop OOP Software Complexity Metrics. Annual Oregon Workshop on Software Metrics, April 10.12, 1994, Silver Falls, Oregon, USA.

/LARA90/ Laranjeira, L.:
Software Size Estimation of Object-Oriented Systems. IEEE Transactions on Software Engineering, May 1990, pp. 510-522.

/LEDG75/ Ledgard, Henry F.; Marcotty, Michael:
A Genealogy of Control Structures. Communications of the ACM, Volume 18, No. 11, November 1975.

/LI.93/ Li, Wei; Henry, Sallie:
Object-Oriented Metrics that Predict Maintainability.
Journal of System and Software, Volume 23, No. 2,
November 1993, pp. 111-122

/LI.93a/ Li, W.; Henry, S.:
Maintenance Metrics for the Object Oriented Paradigm. Proceedings of the First International Software Metrics Symposium, may 21-22, Baltimore/USA, 1993, pp. 52-60.

/LORE93/ Lorenz, Mark:
Object-Oriented Software Development - A Practical Guide. Prentice Hall, 1993.

/LORE94/ Lorenz, Mark; Lorenz, Mark:
Object-Oriented Software Metrics. - A Practical Guide. Prentice Hall, 1994.

/LUCE90/ Luce, R. Duncan; Krantz, David H.; Suppes, Patrick; Tversky, Amos:
Foundations of Measurement. Volume 3, Academic Press, 1990

/MART94/ Martin, Robert:
OO Design Quality Metrics - An Analysis of Dependicies. R.C. M. Consulting Inc.

/MART94a/ Martin, Robert:
OO Design Quality Metrics - An Analysis of Dependencies. Working Paper, 1994, (will appear in Proceedings of OOPSLA 94 in ACM SIGPLAN Notices).

/MORR89/ Morris, Kenneth, L.:
Metrics for Object-Oriented Software Development Environments. Massachusetts Institute of Technology, Master of Science in Management, May 1989.

/PRES92/ Pressman, Roger S.:
Software Engineering: A Practitioner's Approach. Third Edition, McGraw Hill, 1992.

/RAIN91/ Rains, E.:
Function Points in an ADA Object-Oriented Design? OOPS Messenger, ACM Press, Volume 2, No. 4, October 1991.

/ROBE79/ Roberts, Fred S.:
Measurement Theory with Applications to Decisionmaking, Utility, and the Social Sciences. Encyclopedia of Mathematics and its Applications Addison Wesley Publishing Company, 1979.

/ROCA88/ Rocacher, Daniel:
Metrics Definitions for Smalltalk. Project ESPRIT 1257, MUSE WP9A, 1988.

/SHAF76/ Shafer, Glenn:
A Mathematical Theory of Evidence. Princeton University Press, 1976.

/SHAF87/ Shafer, Glenn:
Belief Functions and Possibility Measures. In Analysis of Fuzzy Information, Volume 1, Mathematics and Logic, J.C. Bezdek, Ed. Boca Raton, FL, CRC Press, 1987, pp. 51-84.

/SHAR93/ Sharble, Robert, C.; Cohen, Samuel, S.:
The Object-Oriented Brewery: A Comparison of Two Object-Oriented Development Methods. ACM SIGSOFT SOFTWARE ENGINEERING NOTES, Volume 18, No. 2, April 1993, pp. 60-73.

/TAYL93/ Taylor, D.A.:
Software Metrics for Object Technology. Object Magazine, Mar-Apr. 1993, pp. 22-28.

/TEGA92/ Tegarden, David, P.; Sheetz, Steven, D.; Monarchi, David, E.:
A Software Model of Object-Oriented Systems. Decision Support Systems: The International Journal, 7/1992.

/TEGA92a/ Tegarden, David, P.; Sheetz, Steven, D.; Monarchi, David, E.:
Effectiveness of traditional Software Metrics for Object-oriented Systems. Proceedings HICSS-92, San Diego, 1992.

/WEYU88/ Weyuker, Elaine J.:
Evaluating Software Complexity Measures. IEEE Transactions of Software Engineering Volume 14, No. 9, September. 88.

/WHIT92/ Whitmire, Scott, A.:
Measuring Complexity in Object-Oriented Software. Third International Conference on Applications of Software Measures (ASM92), November 1992, La Jolla, California. In: Workshops in Computing: T.Denvir, R.Herman and R.Whitty (Eds.): Proceedings of the BCS-FACS Workshop on Formal Aspects of Measurement, South Bank University, London, May 5, 1991. Series Edited by Professor C.J. Rijsbergen. ISBN 3-540-19788-5. Springer Verlag London Ltd, Springer House, 8 Alexandra Road, Wimbledon, London SW19 7JZ, UK, 1992, pp. 116-141.

/WONG91/ Wong, S., K., M.; Yao, Y., Y.; Bollmann-Sdorra, P.; Bürger, H.,C.:
Axiomatization of Quantitative Belief Structure. IEEE Transaction on Systems, Man and Cybernetics, Volume 21, No. 4, July/August 1991.

/ZUSE91/ Zuse, Horst:
Software Complexity: Measures and Methods. DeGruyter Publisher 1991, Berlin, New York, 605 pages, 498 figures.

/ZUSE92/ Zuse, Horst; Bollmann-Sdorra, Peter:
Measurement Theory and Software Measures. In: Workshops in Computing: T.Denvir, R.Herman and R.Whitty (Eds.): Proceedings of the BCS-FACS Workshop on Formal Aspects of Measurement, South Bank University, London, May 5, 1991. Series Edited by Professor C.J. Rijsbergen. ISBN 3-540-19788-5. Springer Publisher.

/ZUSE92b/ Zuse, Horst:
Measuring Factors Contributing to Software Maintenance Complexity. Proceedings of the 2nd International Conference on Software Quality, Triangle Research Park, NC, October 4-7, 1992. ASQC (American Society for Quality Control) 611 East Wisconsin Avenue, Milwaukee, Wisconsin 53202, USA, pp. 178-190.

/ZUSE94a/ Zuse, Horst:
Software Complexity Metrics/Analysis. Marciniak, John, J. (Editor-in-Chief): Encyclopedia of Software Engineering, Volume I, John Wiley & Sons, Inc. 1994, pp. 131-166.

/ZUSE94c/ Zuse, Horst:
Foundations of the Validation of Object-Oriented Software Measures. In: Theorie und Praxis der Softwaremessung (Dumke, R.; Zuse, H. (Editors), Deutsche Universitätsverlag DUV, Gabler - Vieweg - Westdeutscher Verlag, 1994, pp. 136-214.

/ZUSE95/ Zuse, Horst; Fetcke, Thomas:

Properties of Object-oriented Software Measures. Proceedings of the Annual Oregon Workshop on Software Metrics (AOWSM), Silver State Park, June 3-5, 1995.

About the Author:

Horst Zuse received his B.S. degree in electrical engineering from the Technische Universität of Berlin, Germany, in 1970, the Diploma degree in electrical engineering from the Technische Universität in 1973, and the Ph.D. Degree in computer science from the Technische Universität of Berlin in 1985. Since 1975 he is senior research scientist with the Technische Universität Berlin. His research interests are information retrieval systems, software engineering, software metrics and the measurement of "complexity" and "quality" of software during the software life-cycle. From 1987 to 1988 he was for one year with IBM Thomas J. Watson Research in Yorktown Heights. His research work there was software metrics, too. In 1991 he published the book: Software Complexity - Measures and Methods (De Gruyter Publisher). From 1989 till 1992 he was with the ESPRIT II Project 2384 METKIT (Metric-Educational- Toolkit) of the European Commission. Since 1990 he gives several times a year seminars about software metrics for people of the industry within the scope of DECollege and ORACLE. He also gave many presentations and seminars about software measurement on conferences in US and Canada. Now, his research interests are validation of software metrics, evaluation of cost estimation and prediction models, application of software metrics in the whole software life-cycle and standardization of software measures. From August 1, 1994 to December 31, 1994 he is invited as a researcher by the *Gesellschaft für Mathematik und Datenverarbeitung (GMD)* in St. Augustin in Germany.

The PERFECT Approach to Continuous Improvement using Experience

PERFECT Consortium
edited by
Anders Gustavsson and Christer Mattsson
Q-Labs AB, Sweden

ABSTRACT

We present the PERFECT Improvement Approach (PIA) which is aimed at supporting organisations in achieving better understanding of how the different elements of their software development processes affect the quality of the software product. The PIA is developed within the PERFECT project which is an ESPRIT-III-project. The PERFECT project started in October 1993 and will end in September 1996 and is now starting its second cycle. The approach taken for the PIA is to package existing and in the project developed research approaches together with existing and developed tools and methods. The methods will allow an organisation using the PIA to learn and understand what impact the software processes have on the product developed. In the PIA, pilot projects and production projects are executed for the results to be analysed for further, future improvements. The project has so far resulted in the PIA and a platform with a modelling language, APEL, supporting the PIA and its methods. The fundamental concepts of the PIA are based on the well-known QIP (Quality Improvement Paradigm), GQM (Goal Question Metric) and Experience Factory. To these methods and models have been added and developed functionalities and processes supporting organisations in performing continuos process improvement. For the GQM-paradigm a process with activities has been developed for how to perform metrics derivations and analysis. Since the PERFECT project is application driven the methodology developed has been used by the applications in the project which has resulted in ample feedback steering the research and development in appropriate directions. In this paper the PIA is presented and how the PIA experience organisation supports continuous improvement efforts. We also discuss how such improvement programs can be started using the PIA.

1. Introduction

1.1 Background

The problem of the software industry of today is the low ability to meet customer requirements and the low expectations on quality. The software industry, in simplified terms, still have the following problems:

- projects cannot satisfy and/or predict quality, cost, and time requirements,
- the quality of software products cannot be demonstrated or certified at the time of delivery,
- organisations cannot infuse new technologies in a controlled way,
- improvement efforts often fail due to e.g. expectations on quick results, inappropriate improvement goals etc.

The typical organisation today has realised the connection between the software development process and the quality of the delivered software products. A clear trend in the software community is to focus on the processes of the organisation and to initiate and establish different types of software process improvement programs. Reasons for wanting

to implement a process improvement program can be to cope with the problems stated above but also if new software quality standards are to be applied in the organisation. These new standards could be the result of entering new markets or the result of changed requirements from customers.

One important step for an organisation running improvement programs is to analyse and understand the elements of its current software development process. The gained understanding can then be used to improve the organisations ability to meet the various requirements put on software, e.g. quality requirements. Examining the problems of software industry in more detail, there exists a common set of often addressed problems that have to be considered in a process improvement program.

Examples of these are:

- Vague insight into how the development process affects the quality of the software products.
- Limited support for defining process improvement plans.
- Absence of cost-benefit models.
- Poor ability to use and adapt the current state of the art in process improvement technology.
- Insufficient engineering processes for reduction of defects.
- Lack of adaptable platforms for support.

There is also a common understanding in the software development community that there is no single method or model adequately supporting continuous process improvement. The purpose of the PERFECT project is to develop a generic model for systematic process improvement of software development processes solving the problems stated in the previous paragraph. Continuous process improvement in the PERFECT sense means the presence of experience feedback loops on both project level and organisational level. These loops aid in using the outcome of a project as an instrument for changing the subsequent instantiations and executions of projects. This model for continuous software process improvement is developed within PERFECT, using known and developed models and methods, integrating these and developing new methods and models as needed.

The outline of this paper is as follows: the rest of this section is devoted to present the PERFECT project, goals, objectives and partners involved. Section 2 presents the PERFECT Improvement Approach, the PIA, with its goals, principles, methods and models. Section 2 also presents the PIA experience organisation with its functionalities. Section 3 discuses how to start an improvement programme using the PIA. Section 4 discusses some experiences from usage and sections 5 concludes the presentation.

1.2 The PERFECT project

A process improvement process implies that methods and models for improvement supports the continuous and systematic feedback about the performance of the software processes. Methods and models for systematic quality improvement are methods which utilise both assessment methods to build a base and set goals for improvements as well as methods for defining measurable goals with their measurements. These methods and models include feedback mechanisms in order to exploit and utilise measurements.

The overall goal of the PERFECT project is to provide means for industry to achieve this in their improvement efforts. PERFECT is aiming at supporting the ability to continuously improve the software development process in order to achieve specified quantitative quality goals. The insights of the characteristics of the software industry have resulted in a set of goals for the PERFECT project which take into account the software characteristics and the problematic areas of the software industry. Usage of the PIA will specifically affect organisations in such way that they will be able to define and implement a software process improvement plan based on quantitative quality goals, improve the organisation's processes by feedback from development projects and previous improvement cycles, and forecast and act on quality problems based on experiences in previous projects. Moreover they will be able to package process technology for future use within the organisation.

1.3 Contributions in the PERFECT project

The development within the PERFECT project rests on sound principles presented below and well-known methods and technologies like the QIP [Basi94b], the GQM [Basi93c] and the experience factory concept [Basi93a]. Within the PERFECT project these have been put together forming the base for the PIA. Furthermore a abstract process language, APEL, has been designed and implemented on a platform supporting and automating some of the methods designed. This platform is implemented using two tools, Process Weaver and Adele, with added functionality and a graphical user interface. The main contributions with respect to methodologies in PERFECT is the integration of methods and models forming a platform supported process improvement approach developed considering the requirements and needs of European industry. Furthermore the QIP, GQM, and the experience factory have been added more detail to making them more usable together. Details have been added in the form of activities and processes assisting and guiding through continuous process improvement programmes. The are organised into what is called the PERFECT Experience Factory (PEF), where organisational and project support processes reside and interact with the experience base. The PEF is a refined and structtured development of the experience factory presented in [Basi93a]. For the GQM a detailed process have been developed guiding the user in deriving metrics, building GQM-plans and measurement plan ans subsequently analysing the measurements against goals and questions. One new concept developed is the use of experience feedback loops on different levels, i.e. organisational and project. The loops on project level provide on-line feedback and guidance to project in execution. Methods and models for analysing impact of changes and improvements have also been developed within the project. Most important is integrating it all and using it in the applications, giving feedback on how to proceed with the development and refinement of developed methods, technologies and the platform.

1.4 The PERFECT consortium

The PERFECT project is application driven which means that the experiences and feedback from the applications should direct the work within the project. The aim of the applications is to provide the technology assessment framework for the project. Based on the results of the methodology and platform work a continuous improvement and evaluation process is introduced and an evaluation of developed methodology, framework and platform will take place, resulting in new inputs for the continued work. Partners in the consortium are:

- CAP Gemini Innovation, Grenoble, France [Platform]

- Q-Labs, Lund, Sweden [Methodology, Applications]
    - Lund University, Sweden [subcontractor]
- University of Grenoble, France [Platform]
- University of Kaiserslautern, Germany [Methodology, Platform]
- Bosch, Schwieberdingen, Germany [Applications]
- Siemens, Oslo, Norway [Applications]
    - SINTEF, Trondheim, Norway [subcontractor]
- Daimler-Benz, Ulm, Germany [Methodology]

The project started in October 1993 and has a duration of 36 months. The final result of PERFECT will be delivered in the form of a handbook at the end of September 1996.

2. The PERFECT Improvement Approach

In this section the PERFECT Improvement Approach (PIA) is presented via the following:

- a set of goals derived from most common problems in software development, giving the motivation for an organisation to be interested in the PIA,
- a set of basic principles for process improvement, setting the context and mind-set,
- a set of generic models (e.g. process models, organisational models) for process improvement at various levels of detail,
- and a set of candidate technologies (i.e. techniques and tools) supporting process improvement aspects.

The goals are set from problems and requirements found in the software developing industry. From these goals some basic and fundamental principles are derived forming the framework and reference points for the work within PERFECT w.r.t. developing the PIA. All the sets described above form the experience base of the PERFECT project.

PIA is aimed at supporting industrial organisations in their attempt to improve the maturity of their software development processes. This short overview of the PIA motivates the need for improvement in the software domain, describes mandatory principles for improvement used within PERFECT, describes a high-level improvement process, summarises a first set of support technologies (i.e. techniques and tools), summarises existing experiences with PIA, and suggests possible ways to get started using PIA.

2.1 Motivation, goals, and principles for PIA

Motivations and goals

Referring to the problems stated in the introduction any software development and software maintenance organisation must have as a goal to eliminate the problems hinder or obstruct their business success. These goals are the main driver for the improvement effort within an organisation. A non-inclusive list of goals can be:

- Gain insight into how the development process affects the quality of the software products.
- Achieve support for defining process improvement plans.
- Develop cost-benefit models.
- Increase the ability to transfer state of the art in process improvement technology.
- Develop engineering processes for reduction of defects.
- Learn from experience in past projects.
- Measure performance indicators against goals.

Wanting to achieve goals of this kind is the motivation for PERFECT and for organisations for using the PERFECT Improvement Approach.

Process improvement principles

Concepts of improvement exist in various engineering domains. The concept of Total Quality Management (TQM) is well known from the manufacturing domain [Feig91]. The question is how we can adapt ideas such as TQM to our domain without neglecting the specific characteristics of the software domain [Basi93a]. Over the past 10 to 15 years, much progress has been made towards understanding the software domain. The following characteristics need to be kept in mind [Basi95]:

- software discipline is experimental (i.e. we need to constantly gain experience from development projects, and provide it for reuse in future projects) [Romb93]
- most development techniques are human-based (i.e., their effects are not easily repeatable)
- software is developed not produced/manufactured (i.e. each development process is unique)
- currently there is a lack of explicit models of processes, products and other relevant aspects (i.e. effects are not predictable or repeatable)
- each software is different in terms of goals and contexts (i.e. development approaches, measures etc. may be different)
- packaging of explicit experience to enhance its reuse potential requires additional resources outside the normal project environments (i.e., additional but integrated organisation needs to be established)

PIA Principles

From improvement approaches which have been successfully adapted to these characteristics of the software domain, we have adopted the following PIA principles [Basi95]:

- The primary focus is on product improvement; process improvement has to be justified as a means towards product improvement
- Improvement has to be a continuous process based on the scientific principles (e.g. Quality Improvement Paradigm [Basi94b])

- Improvement has to be a continuous process based on the scientific principles (e.g. Quality Improvement Paradigm [Basi94b])
- Projects are the main vehicle for continuous process improvement since they provide the natural basis for integrating learning and reuse of learned experience
- Improvement effort has to be committed throughout the organisation and supplied with sufficient budget and resources
- Measurement is essential and has to be tied to the specific goals and characteristics of a development project
- Organisations need to instantiate the scientific principal (cf. QIP) in order to achieve project-specific goals as well as experience building goals of an organisation
- The proper implementation of improvement requires resources for the analysis and packaging of explicit experiences

These principles are based on lessons learned from throughout the software development community. They form the rationale for the process model and organisational model developed within the PERFECT project as being the elements of the PIA method.

2.2 PIA methods and their underlying models

The PIA methods consist of models for a variety of aspects, e.g. a PIA process model and a PIA organisation model. The PIA-process model is the conceptual framework for how activities are performed, what steps are executed in different methods. All processes developed should be derived from this PIA process model. The PIA organisational model form the conceptual model for managing the improvement efforts and the experiences of the organisation.

The PIA methods start with a high-level view where the highest level of the PIA-process model is based on the QIP, [Basi94b]. On that level there can be different *views of* the PIA-Process, for instance a Project-Level-Process (project view), GQM-Process, (measurement view) and an Organisational-Level-Process (organisational view). This views are in turn *refined into* appropriate level of detail for usage. As an example the PIA-Process model can have the view GQM-Process, which in turn can be refined into a GQM-metrics-derivation-process.

For each company instantiating the PIA the will be company specific *specialisations* of the models, i.e. how did they go about it. On all levels the models are integrated forming the PIA.

In the remainder of this section the overall PIA process model, the PIA organisational model and the existing views and refinements are presented. In section 2.3 the PIA technologies supporting the process improvement approach are enumerated.

High-level PIA Models

The PIA models are of two kinds, the PIA-process model and the Organisational model, as stated above. The top level PIA process model is based on the QIP (Quality Improvement Paradigm), defined by Basili et al, see for instance [Basi94b]. The QIP consists of the following six steps:

1. Characterise
2. Set Goals
3. Choose Models
4. Execute
5. Analyse
6. Package

This process model is used as "source" process model when developing and deriving different views, for instance the measurement view.

The (top-level) PIA organisational model reflects the need for extra-project resources (i.e. strategic improvement management, project support, and experience package engineering). The organisational model is shown in figure 1 with its parts, the Software development project organisations and the experience factory organisation with the experience base and the functionalities.

The experience base consists of the different models needed such as:

- Process models,
- Product models,
- and Quality/Cost models.

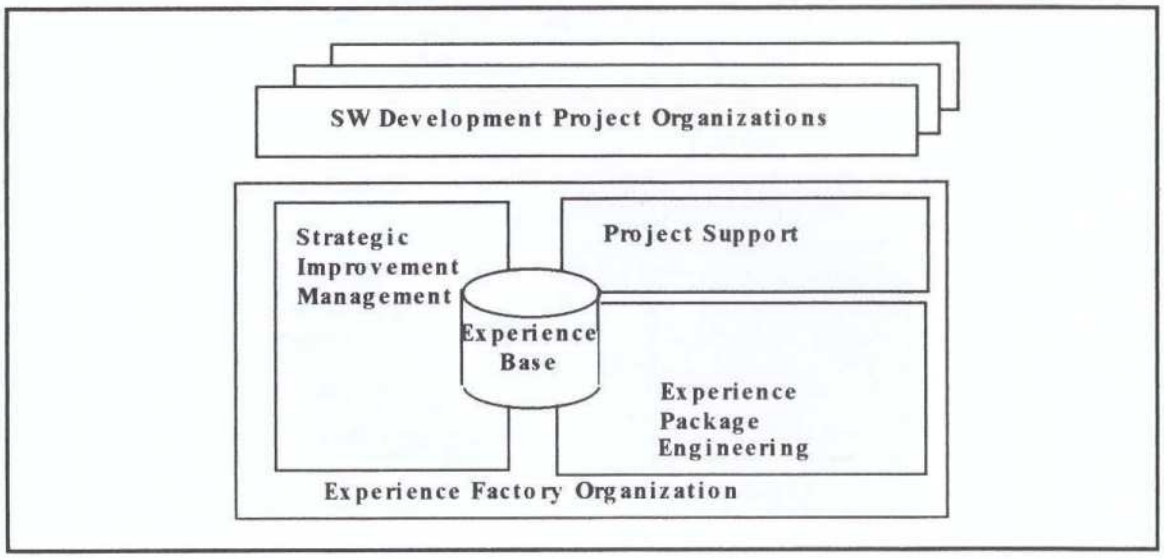


Figure 1: Organisational model of the PIA

The PIA process model and the PIA organisational model are integrated into one overall PIA model, which is shown below in figure 2 in the upper part of the PIA system model.

Views and Refinements of the PIA elements

Views of the PIA elements include:

- GQM process model: a specialisation of the PIA process model from a measurement perspective

- Project-level improvement [D22A, WP22b]: a specialisation of the PIA process model from a project perspective
- Organisation-level improvement [D22A, WP22b]: a specialisation of the PIA process model from an organisation perspective

Refinements of PIA elements include:

- Experience factory model [WP22b]: a refinement of the PIA experience factory organisation in figure 1

It is clear that each process and model is generic in the sense that it generalises from specific environments and/or projects. Different instantiations are possible, including the choice of supporting techniques and tools.

2.3 PIA TECHNOLOGIES

The current example set of PIA technologies contains:

- PIA System Model (figure 2).

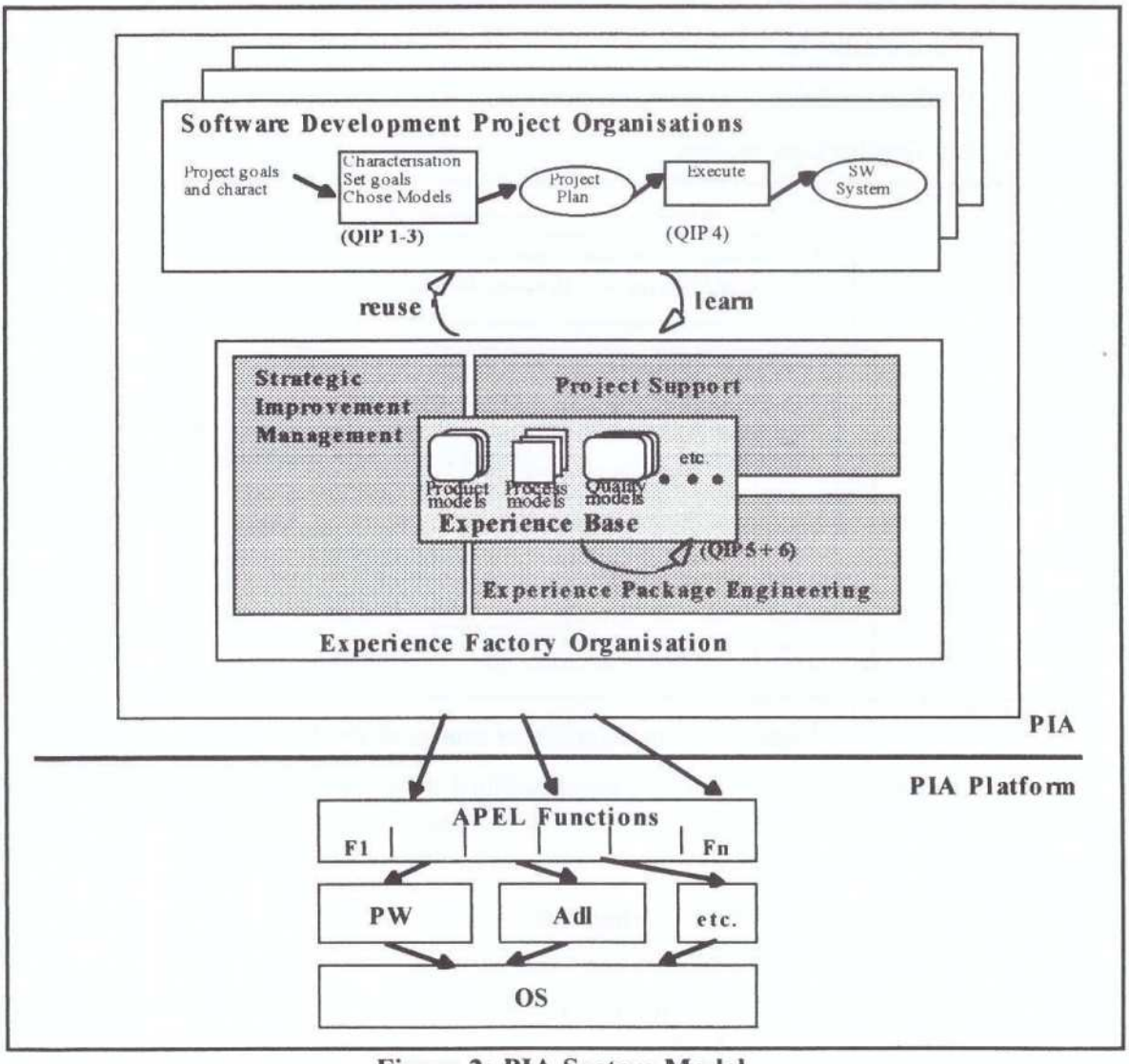


Figure 2: PIA System Model

- GQM templates and abstraction sheets as notations to document the results of the GQM process [D25B]
- Two GQM editors (one from UKL; one from the ESSI-project Cemp)
- Several development process modelling notations (MVP-L from UKL, PW language from CGInn, APEL V2)
- A set of APEL editors
- Survey of commercially available Measurement tools
- Process execution tools (Weadele)
- Product configuration management tools (Adele)

2.4 PIA Experience Factory Organisation

The Experience Factory Organisation contains three functionalities for support of organisational improvement programmes, software development projects and for management of the experience base of the organisations. Below these functionalities and the processes are described.

The "Strategic Improvement Management" Function

The purpose of "Strategic Improvement Management" functions of the PIA, is to manage the organisation's processes strategically and effectively, in order to be organisational assets.

This includes how to systematically increase an organisation's ability to fulfil its business goals by small step improvements of its software development processes, how to identify new technology in the area of software engineering that is or will be most favourable for the organisation's ability to fulfil its business goals, and how to introduce selected new technology into the organisation in an manner that will maximise the technology's positive effects on the organisation.

The Strategic Improvement Management function contains three processes, namely:

- **"New Process Technology Investigation" Process:** The purpose of the "New Technology Investigation" process is to systematically recommend new technology in the area of SE that will be most favourable for the organisation's ability to fulfil its business goals.
- **"New Process Technology Introduction" Process:** The purpose of the "New Process Technology Introduction" process is to systematically introduce selected new technology into the organisation.
- **"Improvement Programme" Process:** The purpose of the "Improvement Programme" process is to systematically encourage and support an organisation to fulfil its business goals by evolutionary process improvements.

The "Experience Package Engineering" Function

The purpose of the "Experience Package Engineering" function of the PIA, is to establish, maintain, and reuse processes and the experience of their use.

This includes how to define and document a process so that projects in the organisation can take full advantage of it. It also includes how to augment the processes with the necessary definitions and models to store and visualise properties of the process. This also includes how to use the gathered experience of the processes and their performance to improve the process parts that are shown to be less efficient.

The Experience Package Engineering function contains three processes, namely

- **"Process Development" Process:** The "Process Development" process describes how to document a process to make it a useful and valuable asset to the organisation. The purpose of the "process development" process is to develop, document and maintain a reusable software development process.
- **"Experience Package Development" Process:** The "Experience Package Development" process describes how to augment the process with the quality models and measurement definitions that makes it possible to get quantitative experience of the process' performance. The purpose of the "Experience Package Development" is to augment a process with features for gathering experience from its use.
- **"Experience Package Improvement" Process:** The "Experience Package Improvement" process describes how to improve specific properties of a process based on the experience of its performance. The purpose of the "experience package improvement process" is to systematically improve a process based on experience of its use.

The "Project Support" Function

The purpose of the "Project Support" function of the PIA, is to support the effective use and adaption of the software development process in a software project based on the organisation's experiences.

This includes how to introduce a process into a project and make it stay in use. It also includes how to understand the properties of a used process in a project by characterising, gathering data and drawing conclusions from the project and its performance. A third part of the support of projects is to guide the projects by providing experience from other projects that can help show the connection between possible actions and their consequences for the projects. To be able to do this it is necessary to have a good quantified understanding of the process and its performance to substantiate the credibility of the experiences.

The Project Support function contains three processes, namely:

- **"Process Introduction and Use" Process:** The "Process Introduction" process describes how to introduce a process into a project and how to make the process stay in use. It includes support for how to tailor, plan, instantiate, train, inform, coach, measure the process or based on the process. The purpose of the "Process Introduction" process is to support the introduction of a software development process into a project and to continuously support and motivate its use.
- **"Process Understanding" Process:** The "Process Understanding" process describes how to understand the properties of a process used in a project by characterising, gathering data and drawing conclusions from a project applying it.

The purpose of the "Process Understanding" process is to understand a process by monitoring and analysing its use, including packaging of the experience.

- **"Process Guidance" Process:** The "Process Guidance" process describes how to guide a project by providing experience from other projects that will help support the project doing the most effective actions in the most efficient way, showing the connection between possible actions and their consequences for the project's performance. The purpose of the "Process Guidance" process is to guide a project towards achieving its goals by providing experience from previous use of the process (i.e. process and resource experience).

2.5 Summary of PIA presentation

The PIA and the platform have been presented in a very brief fashion in the previous sections. In this limited space it is not possible to go deeper into detail. Although resting on sound principles and using known technologies and methods new details have been developed and added. One major goal has been to integrate and make usable to industry an approach for continuous process improvement with support for goal-oriented measurement and comprehensive use of experiences.

3. Successive introduction

There is no single best way to introduce the PIA into a company environment. It depends very much on the current maturity and the maturity aimed for depending on time and resources available. In simple words, the starting point and target point of quality improvement decides on the appropriate start-up. By the time the organisation gets used to running improvement programmes and with increased experience, the focus of the improvement programmes can become more advanced.

Each organisation has to select, tailor and refine the processes from the set of PIA processes. Depending on the goals, characteristics and maturity of the organisation and project plans for the project these processes should be instantiated to a set of processes that forms the process improvement programme.

Examples instantiation include:

- The "Immature" organisation that starts with selecting processes for establishing the infrastructures and baselining the current organisation and projects.
- The "Process guidance" organisation has some limited experiences collected and is reusing this information for guiding the projects.
- The "Systematic process improvement" organisation is a mature organisation with the ideas of the basic principles implemented and in operation. This includes processes for reuse of experiences in all stages, both in terms of understanding, guiding and the introduction of new technologies. An example road map for systematic process improvement is presented in figure 3.

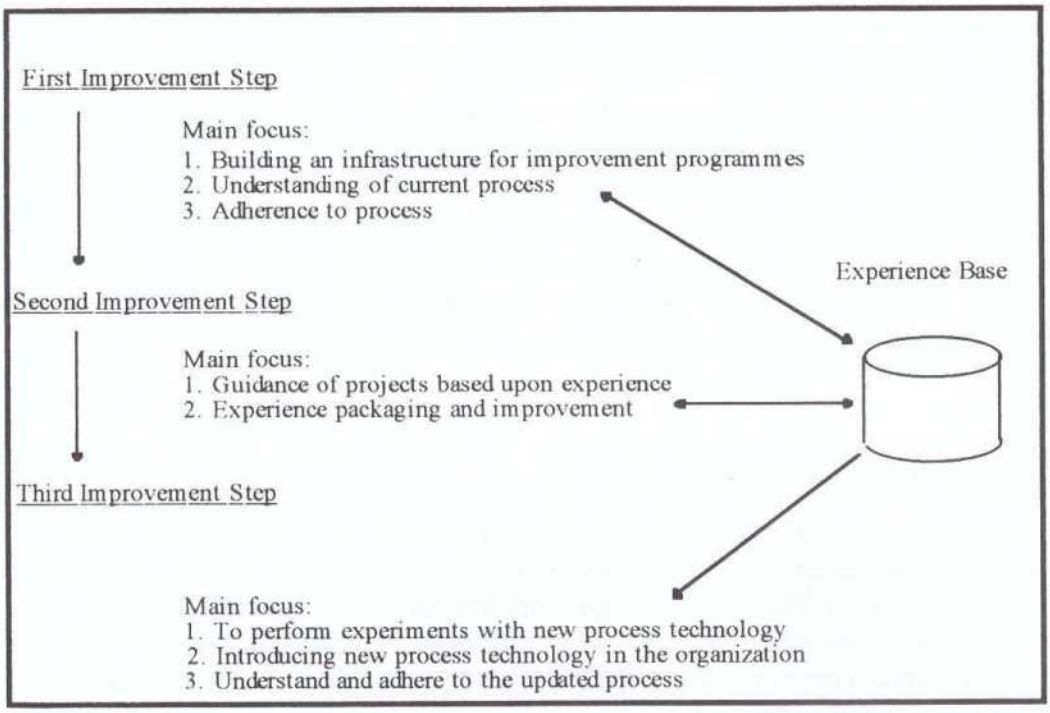


Figure 3: Successive Introduction Scenario

3.1 Example road map for systematic process improvement

Looking at figure 3 the main focus of the first improvement step is "only" to get a proper baseline of the currently used process and then to achieve adherence to a commonly agreed process. According to the figure, the subsequent improvement step have a more demanding focus.

In the second improvement step experiences from the first programme is now used to guide projects since experiences have been collected forming an understanding of the processes used.

In the third improvement step the organisation have such control that new technology can be evaluated and introduced in the organisation.

However, independent on the specific situation, a few lessons seem to be general [Basi95]:

- repeatable improvements are mostly achieved by continual, sustained process change; not technology breakthroughs,
- emphasise product improvement as the goal; not process alone,
- major goal of PIA is self-improvement; not external comparison,
- PIA should be started small in order to learn from experience using the CMM terminology: an organisation should act as a level-5 organisation from the beginning and let measurement drive needed process changes as appropriate for the organisation to accept the fact that it takes time to fully adopt new technologies.

4. Experience from usage

The experiences from using the PERFECT improvement approach within the project are the experiences from the applications. The project is now (July 1995) in the phase of starting the second half of project. Experiences from applications are now used to set requirements on remaining activities and for updating deliverables from the first half. The fundamentals have however been used extensively in other earlier projects, such as NASA/SEL.

Current example applications and instantiations of PIA include:

NASA/SEL (pre-Perfect version) [Basi93b], [Basi94a].

Bosch (D), Siemens (N), Q-Labs (S) as PERFECT-internal application projects.

Schlumberger (NL), DEC (I) from the ESSI project CEMP.

Siemens, Allianz [WP22a], and others through collaborations with the Software-Technology-Transfer-Institute at UKL (STTI-KL).

Experiences from all these sites are being collected, analysed and provided.

5. Conclusions

In this paper we have presented the ESPRIT project PERFECT and its proposed improvement approach. The goals and motivations for the project were lead into fundamental principles based on lessons learned and experiences. The presented PIA (PERFECT Improvement Approach) has as major principles to support continuous improvement of the software products and hence the processes via learning from experiences. This gathering of experiences is done by the use of goal oriented measurement, always related to quantitative goals on both organisational and project levels. Given this, the PIA gives methods for organisations to understand the different factors affecting their development processes and their impact on their product quality. The PIA also supports subsequent improvement and how these should be guided in the organisation.

Given as input, from organisations willing to improve, their characteristics, goals and constraints it is possible to select the best suited PIA forming the PERFECT Experience Factory Organisation of that company. PERFECT also supplies support for how to enter new technologies and how to launch improvement programs within an organisation. The PERFECT project is now in the phase of ending the first, and starting the second and last cycle. Since the project is application driven the PERFECT experience base learns from the applications use of developed methods and platform. This will eventually be published at the end of the project as a handbook.

6. References

[Basi93a] Victor R. Basili. The Experience Factory and its relationship to other improvement paradigms. In Ian Sommerville and Manfred Paul, editors, Proceedings of the 4th European Software Engineering Conference, pages 68–83. Lecture Notes in Computer Science Nr. 717, Springer-Verlag, 1993.

[Basi93b] Victor R. Basili. The maturing of the Quality Improvement Paradigm in the SEL. pages 39–59. NASA Goddard Space Flight Center, Greenbelt MD 20771, 1993.

[Basi93c] Victor. R. Basili, Applying the GQM Paradigm in the Experience factory, in 10th Annual CSR Workshop, October 1993, To appear as part of a book entitled Software Quality Assurance: A Worldwide Perspective to be published by Chapman and Hall.

[Basi94a] Victor R. Basili and Scott Green. Software process evolution at the SEL. IEEE Software, 11(4):58–66, July 1994.

[Basi94b] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. Experience Factory. In John J. Marciniak, editor, Encyclopedia of Software Engineering, volume 1, pages 469–476. John Wiley & Sons, 1994.

[Basi95] Victor R. Basili and Frank McGarry. The Experience Factory: How to build and run one, April 1995. Tutorial given at the 17th International Conference on Software Engineering (ICSE17), Seattle, Washington, USA.

[BR88] Victor R. Basili and H. Dieter Rombach. The TAME Project: Towards Improvement-oriented Software Environments. IEEE Transactions on Software Engineering, SE-14(6):758-773, June 1988.

[D22A] PERFECT Consortium, D22A. 'The PERFECT Approach for Improving Software Processes'. ESPRIT Project No. 9090, 1994.

[FH93] P. Feiler, W.S Humprey, Software Process Development and Enactment: Concepts and definitions, in Proceedings of the second international conference on the software process. Berlin Germany, February 25-26, 1993

[Feig91] Armand V. Feigenbaum. Total Quality Control. McGraw Hill, NY, 1991.

[Hois94] Barbara Hoisl. A process model for planning GQM-based measurement. Technical Report STTI-94-06-E, Software-Technology-Transfer-Initiative Kaiserslautern, University of Kaiserslautern, 67653 Kaiserslautern, Germany, April 1994..

[Rom91] H. Dieter Rombach. Practical Benefits of Goal-oriented Measurement. In N. Fenton and B. Littlewood, eds., Software Reliability and Metrics. Elsevier Applied Science, London, 1991.

[Romb93] H. Dieter Rombach, Victor R. Basili, and Richard W. Selby, editors. Experimental Software Engineering Issues: A critical assessment and future directions. Lecture Notes in Computer Science Nr. 706, Springer-Verlag, 1993.

[WP22b] PERFECT Consortium, WP222PEF. 'The PERFECT Experience Factory Model rev A'. ESPRIT Project No. 9090, 1995.

ESI ISCN

APS COMETT

Practical Improvement of Software Processes and Products

ESI-ISCN '95 : Measurement and Training Based Process Improvement
September 11-12, Vienna, Austria

ami

ALCATEL

CMS - BRITISH STEEL PLC

BRAMEUR SZTAKI SIEMENS

OBJECTIF TECHNOLOGIE

UNIVERSITY OF MARYLAND

ETNOTEAM PROCESS INC

FESTO

SCOPE

metkit

ISO 9001 VIT

SPICE

BOOTSTRAP Institute

ESPITI

European Software Process Improvement Training Initiative

Approaches

ami provides an overall improvement framework based on the quantitative approach "Assess-Analyse-Metricate-Improve". In ISCN' 95 ami will evaluate how the improvement paradigm works using different assessment methodologies.

Bootstrap is a method for assessing and quantitatively evaluating process quality attributes. In ISCN' 95 a project combining ISO and Bootstrap to an auditing method will be presented.

EQA The European Quality Award provides a guideline and a questionnaire for self-assessment within an organisation and to compare these results internationally. In ISCN' 95 Alcatel will present it's experience with Total Quality Management and the use of EQA as a tool to achieve this.

Metkit is a set of integrated training courses that demonstrate how to implement process improvement programmes and how to measure processes and their products.

Scope has developed procedures and a guide for the evaluation of software product quality. In ISCN' 95 Scope will present a statistical analysis of the operational profile of systems installed in field.

SPICE develops an ISO standard for software process improvement and capability evaluation including a baseline practices guide, a process assessment guide, and a capability determination guide.

TickIT procedures ensure that ISO 9001 audits of software development organisations are performed by IT-competent auditors working for TickIT accredited certification bodies. Guidance material is published in the "TickIT Guide"

Active Integration

Seminar participants are entitled to submit with their registration one page of questions, issues, problem statements or experiences for inclusion in the workshop.

Organisations

ESI The European Software Institute is a major industry initiative dedicated to supporting its member and European industry in improving their customer's competitiveness by promoting and disseminating best practices.

ISCN The main goal of the International Software Consulting Network is to set up a resource pool of experts involved in well known process improvement projects and initiatives such as ami, Bootstrap, Metkit, Scope, Spice, TickIT, and to establish teams of experts from these projects.

The ISCN Co-ordination Office has established formal procedures for new membership, team selection, project co-ordination and network maintenance. Each expert's details are stored in an expert skill database based on a standard expert skill profile. Each customer describes his requirements based on a standard customer requirement report. The database supports the mapping of customer requirement data onto expert skill data.

ISCN guarantees that the experts they promote are highly skilled, well qualified, experienced consultants. Details of ISCN membership is available from the ISCN Co-ordination office.

Who Should Attend and Why?

The ISCN conference series provides continuous information about the evolution of different European strategies, projects, and the experience of users with regard to the implementation of process improvement practices.

Participants will see the process improvement field from three different aspects: Strategic European Programmes, Methodologies, and Industrial Experience. The ultimate goal is that all participants learn how the methodologies can be implemented in their own organisation and what the benefit in measurable terms will be.

The workshop will enable participants to discuss with the experts and the industrial users how to utilise the methodologies to achieve the optimum benefits.

Es wird ersucht, diese für eine Person gültige Einladung
beim Eintritt in das Rathaus unbedingt vorzuweisen.

Please note: This invitation is valid for one person only
and presentation of card is necessary for admittance to City Hall.

Cette invitation valable pour une personne sera à présenter
à l'entrée de l'Hôtel de Ville.

Der Bürgermeister der Bundeshauptstadt Wien

beehrt sich, für

Montag, den 11. September 1995, um 20 Uhr

anlässlich der

ESI-ISCN '95 Konferenz

zu einem

Cocktailempfang

im Wappensaal des Wiener Rathauses

einzuladen

Zugang: 1, Lichtenfelsgasse 2, Feststiege II

Conference Chair : Takis Katsoulakos

Chairman of the Second Day´s Proceedings: Hans Jürgen Kugler

08:30 *Demonstration Stands and Coffee*

Networks and Services	Quality Tools and Methods
ISCN	BOOTSTRAP
ESI	Compita Ltd
	SPR
	APAC

Industrial Experience

09:45	Making the Best the Standard	Brian **Hepworth** CMS - British Steel
10:15	Alcatel - Total Quality Management and the European Quality Award	Günther **Waldner** Alcatel
10:45	**Coffee Break**	
11:00	Essence and Accidents in SEI-style Assessments	Ken **Dymond** Process Inc.
11:30	FESTO - Experience with ESSI Application Experiment ODB	Günther **Rutschek** Festo
12.00	Business Decision Problems Supported by Software Product and Process Assessment	Miklos **Biro** MTA Sztaki
12:30	**Lunch**	
14:00	Siemens Process Improvement Approach	Thomas **Mehner** Siemens
14:30	Quantitative Approach to Software Process Improvement	Annie **Combelles** Objectif Technologie

Networks & Services

15:00	International Software Consulting Network - Managing Expert Networks	Micheál **Mac Airchinnigh** ISCN
15:30	ESI´s Products and Services	Günther **Koch** ESI
16:00	**Coffee Break**	
16:15	Workshop - How do I benefit ?	
20:00	Dinner at traditional Viennese „Heuriger“	

Conference Chair : Takis Katsoulakos

Chairman of the Second Day´s Proceedings: Hans Jürgen Kugler

Organising Committee

Co-ordinating Board	**Richard Messnarz**	**(IST, ISCN)**
	Debbie Penney	**(ISCN)**
Support	**Andreas Bolin**	**(IST, ISCN)**
	Tina Glaser	**(IST)**
	Werner Kerschenbauer	**(IST, ISCN)**
	Michael Loidl	**(ISCN)**
	Markus Pöschl	**(ISCN)**

Programme Committee

Chair: Richard Messnarz (Austria)

Micheal Mac an Airchinnigh	**(Ireland)**
Gualtiero Bazzana	**(Italy)**
Adriana Bicego	**(Italy)**
Martin Brett	**(Germany)**
Christophe Debou	**(Belgium)**
Alec Dorling	**(Spain)**
Hans Jürgen Kugler	**(Spain)**
Pasi Kuvaja	**(Finland)**
Jean Martin Simon	**(France)**
Christer Mattsson	**(Sweden)**
Colin Tully	**(UK)**
Günter Waldner	**(Austria)**

Main Organisers

ESI	European Software Institute, Spain
I.S.C.N.	International Software Consulting Network, Ireland
IST	Institute for Software Technology, Austria

Supporting Organisations

Alcatel
ami User Group
APAC GmbH, Austria
APS Leonardo, Austria
Brameur, UK
CCC Software Professionals, Finland
Centre de Recherche Public Henri Tudor, Luxembourg
Colin Tully Ass., UK
Center for Software Engineering, Regional ESPITI Organiser, Ireland
Etnoteam, Italy
Hibernia Learning Partnership, Ireland
INESC, Portugal
Intersoft, Greece
Leansoft Oy, Finland
Mari Northern Ireland
Network for Women in Technology in Europe (WITEC)
Österreichischer Verband der Wirtschaftsingenieure, Austria
Q-Set Ltd., Ireland
Research Center Seibersdorf, Regional ESPITI Organiser, Austria
SERC, Netherlands
University of Iceland
University of Linz, Regional ESPITI Organiser, Austria
University of Maryland, USA
Unix User Group Austria

MAKING THE BEST THE STANDARD

Users Experiences of operating an ISO 9001 compliant Quality Management System
and Total Quality Management culture

BY

BRIAN HEPWORTH, QUALITY MANAGER,

CMS

(Part of BRITISH STEEL plc, UK)

Summary:

This paper examines the 'people in quality 'perceptions based on the experiences at CMS, part of British Steel plc. Over the past six years CMS have operated an outstanding customer satisfaction ethos to achieve competitive advantage based on an ISO 9001 compliant quality system and a Total Quality Management culture.

Through internal and external assessment it has been possible to identify areas requiring review and, as necessary, appropriate corrective action introduced. Metrics studies, customer perception surveys and improved problem/enquiry, change and configuration management projects have all provided improved staff motivation and customer satisfaction.

The onward journey into quality improvement has continued with the Total Quality Management programme (TQM) requiring cultural change involving everyone in the organisation. This could only be achieved by top down directives and commitment from all levels, including management. Experience indicates that managing the management commitment is a difficult but very necessary process in a quality programme.

1. Introduction

At the 3rd European Conference on Software Quality held in Madrid on 3-6th November 1992 the paper presented was entitled 'Experiences of Certification and Operating the Total I.T. Function to ISO 9001' and detailed the experiences at CMS of establishing and operating a successful Quality Management System (QMS). The foundations of formal procedures and standards were recognised as being essential for developing the quality culture to achieve outstanding customer satisfaction.

Whilst certification, with the subsequent continuous (third party) assessment, provides a sound basis for monitoring the effectiveness of the quality management system it must never be considered that certification is the ultimate quality culture achievement.

At CMS establishing and operating a formal QMS was recognised as only the first stage of the drive to provide improved information technology products and services.

CMS has existed for over twenty years, created by the combining of seven information processing units with different development and hardware platforms.

2. Establishing the Quality Management System.

The senior management of CMS established study groups to investigate the future role of CMS and a number of observations were made.

a) the differences between the best and worst projects was very significant.

b) people operated within the ability of the system to direct them

c) there was a need to make the best practices 'the standard'.

d) external assessment was necessary to prevent erosion of the quality system.

The decision was made, in 1988, to implement a quality management system compliant with ISO 9001 and subject to external (third party) assessment and registration.

The quality strategy at CMS has three defined aspects.

Procedural	-	establishing the procedures to achieve the standard of service promised. (Figure 1)
Continuous Improvement	-	establishing a culture to encourage improvement and to develop qualitative and quantitative measurements of the quality of service and implement the actions necessary for outstanding customer satisfaction.
Managed and Optimised	-	the integration of the results of quality control actions into the prediction of quality variances and effective avoidance of non conformities.

Procedural

The critical factors for the QMS quality initiative were defined as:

Creation of formal procedures, based on current best practice.

Generic 'single page' procedures would be prepared for all necessary activities.

User written procedures to ensure user ownership

A Quality Plan, prepared at the start of each project, would define the quality requirements.

The training of everyone in the organisation would be a critical success factor.

The QMS initiative would be defined as a project

Management commitment was an essential

Everyone in the organisation needed to be involved

People are the key to the quality of any product or service

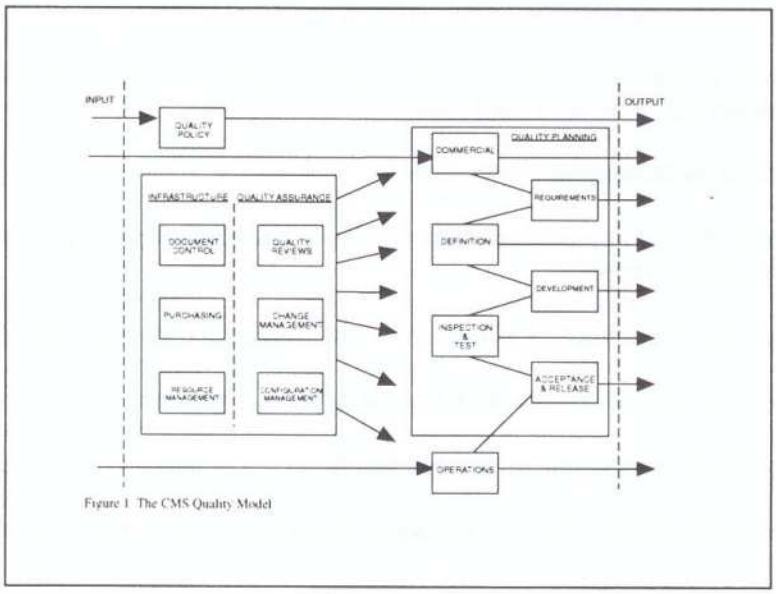


Figure 1 CMS Quality Model

The benefits to be gained from operating a procedural quality management system were recognised as being very significant. By reducing the amount of rework, increasing on-time delivery, ensuring the customer requirements were fully understood at the start of a project, and raising the level of achievement to that of the best practices could result in improving productivity by more than twenty five percent.

The quality initiative provided the opportunity for everyone in CMS to focus on the ways of working at managerial, departmental, organisational and individual level and to open up avenues of communication.

At a time of extreme pressure, due to a full order book and an increasing workload, a new operating culture was designed and introduced by the workforce, whilst continuing to meet customer demands. A significant achievement in a period of a little over twelve months.

On the basis of the Quality Manual the procedures were identified and an organisation of subproject teams as a series of 'missions' was established. Each mission was under the control of a Senior (Blue Card) Manager. (Figure 2)

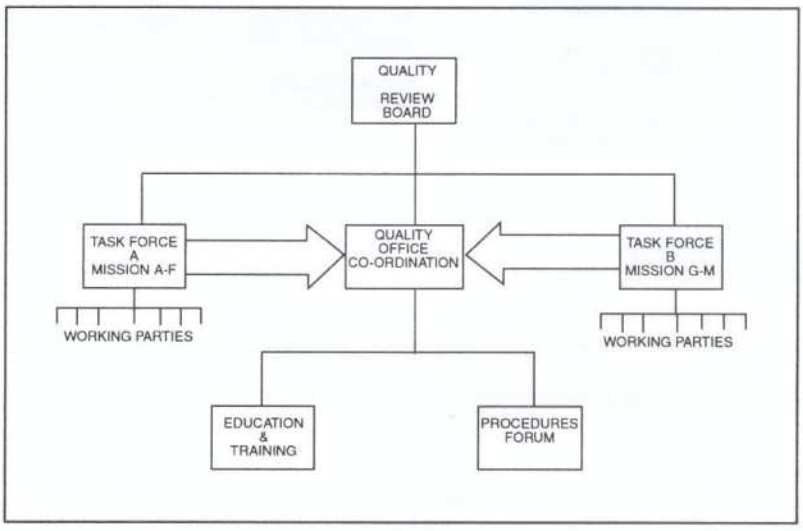


Figure 2: Quality Initiative Organisation

The mission controllers were tasked with creating teams of users to address a set of procedures to develop the detailed steps. The target for all procedures was 'one page'.

The responsibilities of the mission controllers were:

a) to deliver all necessary procedures for the designated area

b) ensure that users define and take ownership of the procedures.

All procedures had to be seen to be written and owned by the users; it was important that the procedures should not be seen as being imposed by the Quality Office. Visible commitment and energetic leadership by each of the mission controllers and every member of each procedure definition team was essential.

General policy on creating effective procedure writing teams was that the ideal membership of a team was five, and that teams should be picked by the mission controller. There were no absolutes on the wording of procedures, nor was there a finite list of procedures. Effectiveness in use was the only success criteria. It was emphasised that the procedures need to be generic so that they could continue to reflect the dynamic nature of the business.

It also became apparent that there was a need for a change of emphasis on 'people' within CMS to a culture that recognises customers rather than users. To this end a commercial section was introduced into the CMS organisation structure.

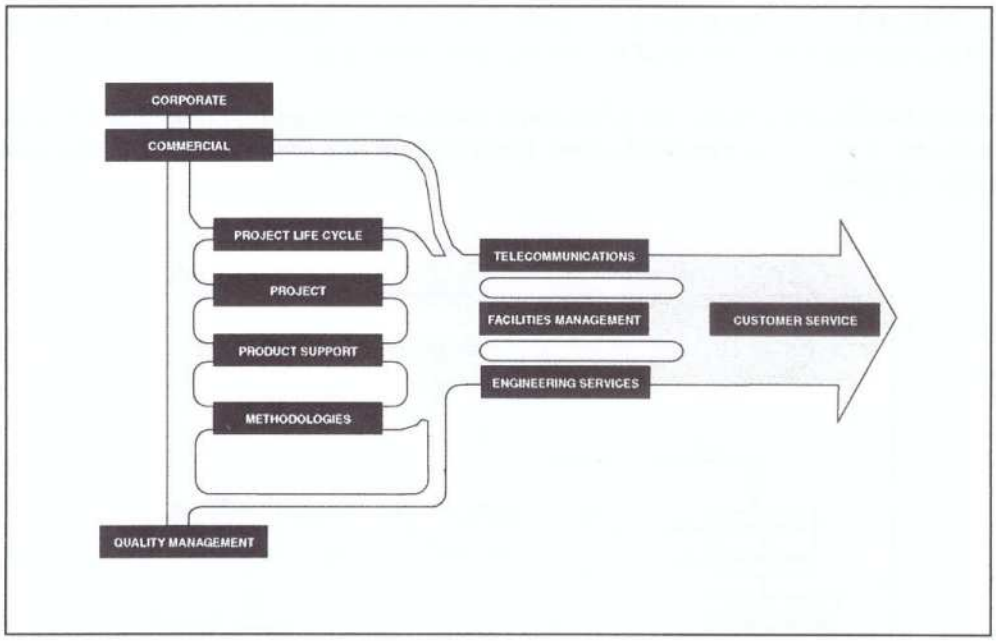


Figure 3: Procedures Energy Flow

The energy flow of the formalised procedures was directed towards improved customer services to ensure outstanding customer satisfaction. (Figure 3)

The Quality Manager's responsibility was to provide the coordinating and interfacing activity, to be the catalyst for change and at times to be the referee.

The greatest accolade that can be awarded to any Quality Manager by users has to be

> "we wrote the Quality Management System, not you".

3. Benefits of the certificated QMS

It is difficult to identify specific features of the quality management system that have contributed most to the development of the quality culture as the 'normal way of working'. However, the adoption of quality reviews at each stage of the design input and output, supported by a visible product and project review schedule as part of the quality plans, have provided significant direct benefits to CMS from the quality culture.

Over a period of time a clear perspective of the improvements that have been achieved can be gained by reflecting on how CMS used to operate.

Reduced error correction costs

A closer partnership with the customer has ensured a clearer definition and understanding of the requirements by both the customer and supplier. This process has been greatly assisted by the introduction of quality reviews throughout the design, development and implementation process to build in quality. This has meant higher costs at the front end but substantially lower costs further down the life cycle. Analysis of the defects arising has been used to highlight areas for improvement.

Reduction in Project Overruns

More ontime deliveries (regularly better than 98.5%) has not only delighted our customers; it has also made resource management an easier task.

Improved Maintainability

The 'formalised', but flexible way of working has now become the standard and has resulted in the provision of detailed project files with fully documented systems that are much easier to support and maintain.

Improved Staff Morale

Working for a 'quality' company generates a strong feeling of commitment and readiness to try that extra bit harder to ensure the standards are maintained. Nobody goes to work to produce a nonquality product. Recognising that the QMS provides a sound basis for ensuring quality is inbuilt has been a major boost to staff morale.

Improved Customer Service

Customer perception surveys and project survey report returns indicate that the CMS/customer relationship has significantly improved. The areas previously causing concern with the customer have been addressed and are now 'satisfactory' or 'good'. We have also been able to improve some of the 'good' to 'very good' and even 'excellent'.

Basis for TQM

Quality is dynamic and the QMS has provided the foundations on which to build an expanding total quality culture. These foundations have been underpinned by a whole series of improvement programmes, metrics, project management, change control, training, and re-organisation. (Figure 4)

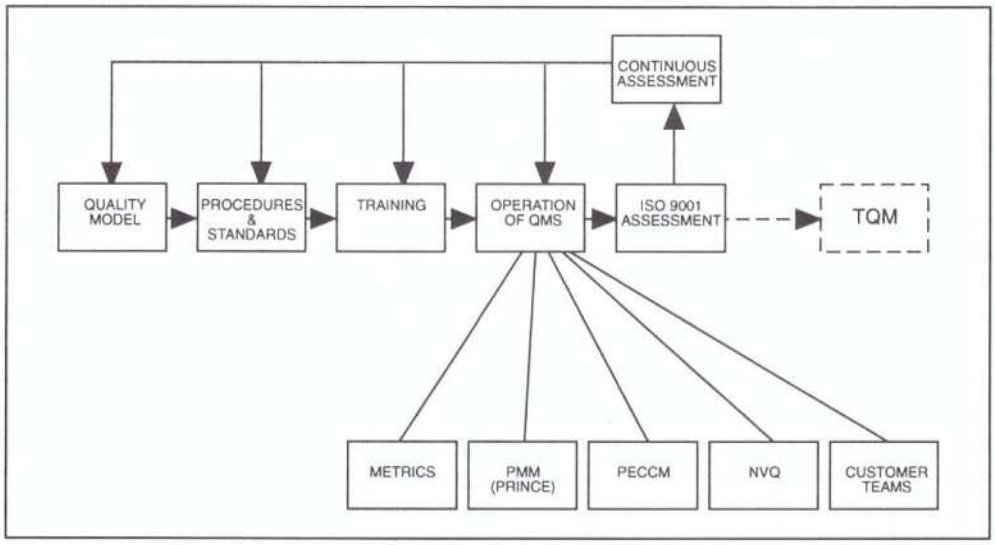


Figure 4 : CMS Processes Model

The main ingredient for success was the total commitment by all concerned.

Since the initial assessment the quality management system has been subject to two management re-organisations, three restructuring initiatives, the introduction of TQM and a leading edge change control system and project management methodology.

The QMS itself has undergone twenty amendment updates and been subject to eighty five internal audits and seventeen external audits.

The results of the external audits over the past four years have highlighted four main areas for concern;

weaknesses in the project management methodology, hence the introduction of a formal project management methodology (PRINCE).

the lack of evaluation of software development tools has resulted in the establishment of the development control centre responsible for evaluation.

the need for a metrics programme to support the effectiveness of the QMS.

recognition that the quality management system must be rigorously adhered to at all levels in the organisation.

The strength of operating a quality system compliant with ISO 9001 and subject to third party auditing has been proved beyond doubt, as have, the benefits. In the last three years the level of business available has fallen dramatically due to the recession, particularly in the traditional customer base.

CMS have maintained and grown the business platform, continued to provide year on year improvement in profits, and at the same time, changed the business profile from seventy percent internal to seventy percent external business.

A significant lesson learned since operating a formal QMS has been that there is no room for complacency, constant vigilance is necessary if the benefits of the QMS are to be continually gleaned.

Early in 1994 CMS successfully completed a reaffirmation of the effectiveness of the quality system under the auspices of a three year TickIT re-assessment by BSI QA.

Summaries of the nonconformances relative to ISO 9001 paragraphs are shown in Figures 5 and 6.

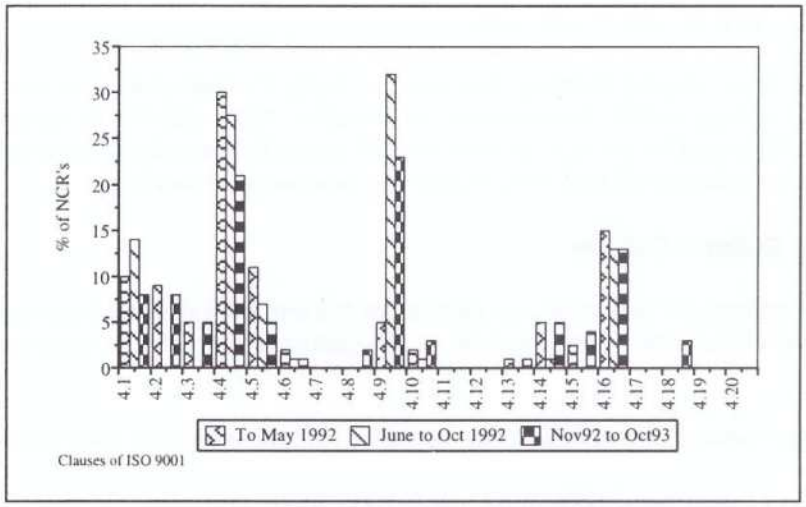


Figure 5: Internal Audit Nonconformances over Time Periods

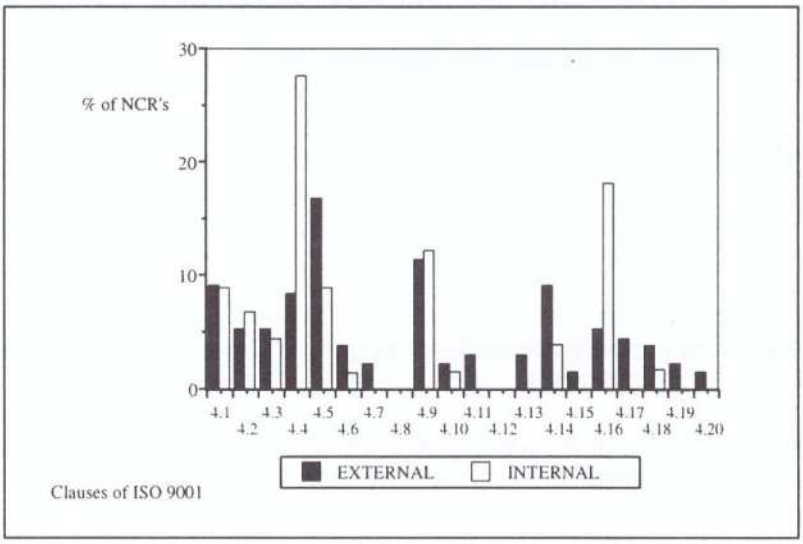


Figure 6: External/Internal Audit Nonconformance Comparisons

4. Continuous Improvement

There are a number of alternative approaches to developing a quality culture

a) Accredited certification to ISO 9001
b) Total Quality Management
c) Business Process Reengineering
d) SEI CMM Capability Maturity Model
e) Benchmarking
f) SPICE
g) European Quality Model (self assessment)
h) Rapid Application Development (RAD)

all of which are complementary to each other.

The first criteria of a continuous improvement programme is to know where you are and to identify the baseline conditions under which the organisation operates. CMS did this by building the procedural QMS as the foundations of future quality initiatives. This was followed by extending the quality culture to encourage participation by all staff into developing new ways of working.

5. Managing the Cultural Change

With the firm foundations of the procedural aspects for the provision of quality services established, the next stage was to explore the wider aspects of the cultural change from hierarchical to consensus management.

The most important aspect of quality at CMS has always been customer satisfaction.

> "To achieve customer satisfaction by delivering timely and cost effective business solutions and IT services through precise performance"

was the original Mission Statement.

This was updated in 1992 to provide a clearer customer commitment in a briefer format

> **"To achieve Outstanding Customer Satisfaction"**

6. Total Quality Management.

The immediate challenge to the change from hierarchical management to a culture of consensus management is at the heart of total quality philosophy.

A decision was taken by the Senior Management to distinguish the TQM programme from the QMS procedures by establishing a quality approach not just to the way a task is performed, but to the way a task is thought about in the first place.

The TQM initiative was launched as a seperate initiative from the currently operational ISO 9001 quality system, rather than an as extention of the quality management system.

A series of one day training programmes were introduced, the main purpose of which was for individuals to identify a program of action to demonstrate the 'Power to Perform'. Follow up reviews were carried out six months later. Other activities included the introduction of a suggestion scheme; continuous improvement schemes; departmental purpose analysis; customer perception surveys.

At the TQM launch seminars the cost of quality was identified and a working evaluation developed. Typically, from each course, the cost of quality in CMS was identified at around 33 to 38%. Whilst there was an implication that cost of quality would be reduced, no target values or dates were set. In addition it was stated that there may be no results for ten years or more; not a very inspiring scenario. Lists of facilitators had been identified but at the time of the launch their responsibilities were undefined and no formal training had taken place. This was subsequently rectified but the TQM image had been tarnished.

The culture changes of Total Quality (TQM) were promoted under five themes

Leadership	at all levels remove the barriers to achievement (openness)
People	only through people can the CMS mission be achieved
Customers	outstanding customer satisfaction through partnership
Value	value for money
Innovation	encourage ideas by all staff in every aspect of the business

The overall message to everyone in CMS was 'Power to Perform'

As in many other organisations, the TQM programme has been subject to a variable level of acceptance and achievement. In the case of CMS, despite a significant training programme, this has resulted in three 'relaunches' of the initiative. Some confusion exists in the workforce regarding the apparent dual and separate quality activities, of QMS and TQM. This has now been addressed and continuous business improvement is currently functioning by combining the procedural and cultural aspects.

Working parties have been established to report on TQM 'health check' aspects:

Review DPA/DIP
Meaningful metrics
Publicise success
Time allocation to TQM
Is This Quality? (ITQ) How to handle?
Attitude! Response to TQM Survey
TQM infrastructure
CMS response to recent Customer Satisfaction Survey
Other companies' experience

TQM quarterly reporting bulletins are issued based on the following metrics :

Customer Perspective

Excellent Service	Number of Customer Complaints
	Customer Project Reviews
Reliability	Operational Availability
	Number of Problems Raised
Responsive Supply	% Projects completed on time
	Problem/Enquiry Resolution
	Quotation Response
	Complaint Resolution
Courteous Response	Telephone answering

Internal Business Perspective

Rewarding Careers	Staff Turnover
Happy Healthy Staff	Absenteeism
Efficient staff utilisation	Profit per Employee
Operating Efficiency	Sales Ratio

Financial Perspective		
	Survive	Cash Generation Return on Capital Profitability
	Succeed	Sales
Innovation and Learning Perspective		
	Investment in training	Training spend as % of Sales Training spend per Employee
	Effective Training	Course Ratings

7. Measurement

The metrics programme has concentrated largely on the customer aspects of the provision of services rather than the measurement of the effectiveness of the QMS in software development.
This has been due to the ability to focus specific measurement directly to outstanding customer service.

Early in 1991 a project was launched to evaluate the customer perception of CMS,based on the application of the model postulated by Parasurman et al (1). with a requirement to identify useful metrics that could be introduced to CMS on an ongoing basis.

Detailed analysis of the returns was made and these were used to identify quality improvement actions. (Figures 7,8,9 and 10). The latest analysis has indicated that the most serious areas of weakness have shown significant improvement.

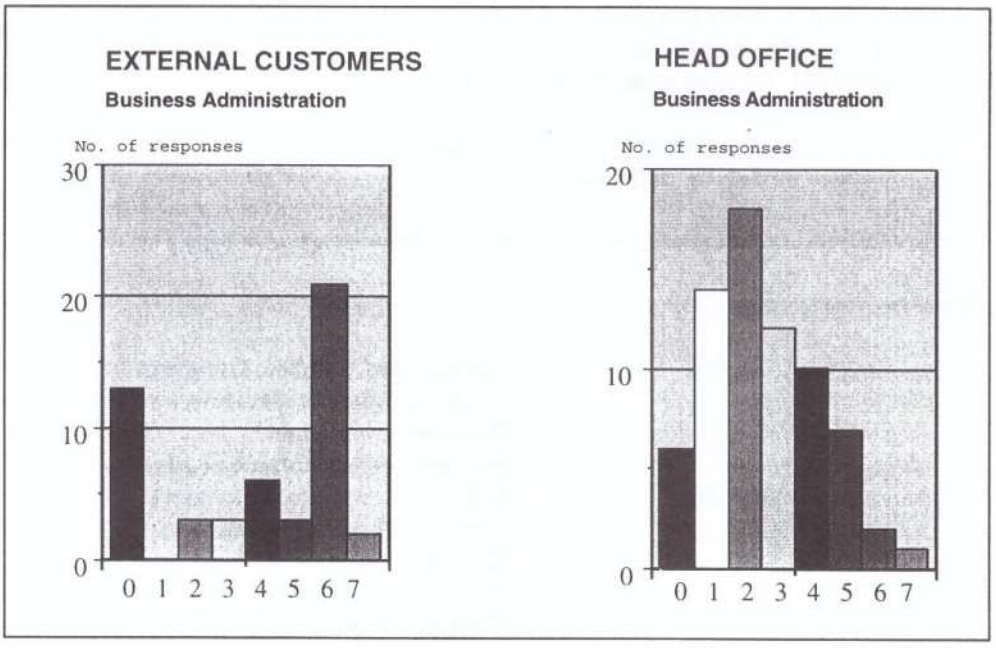


Figure 7&8: Comparison of Customer Responses - Business Admininistration.

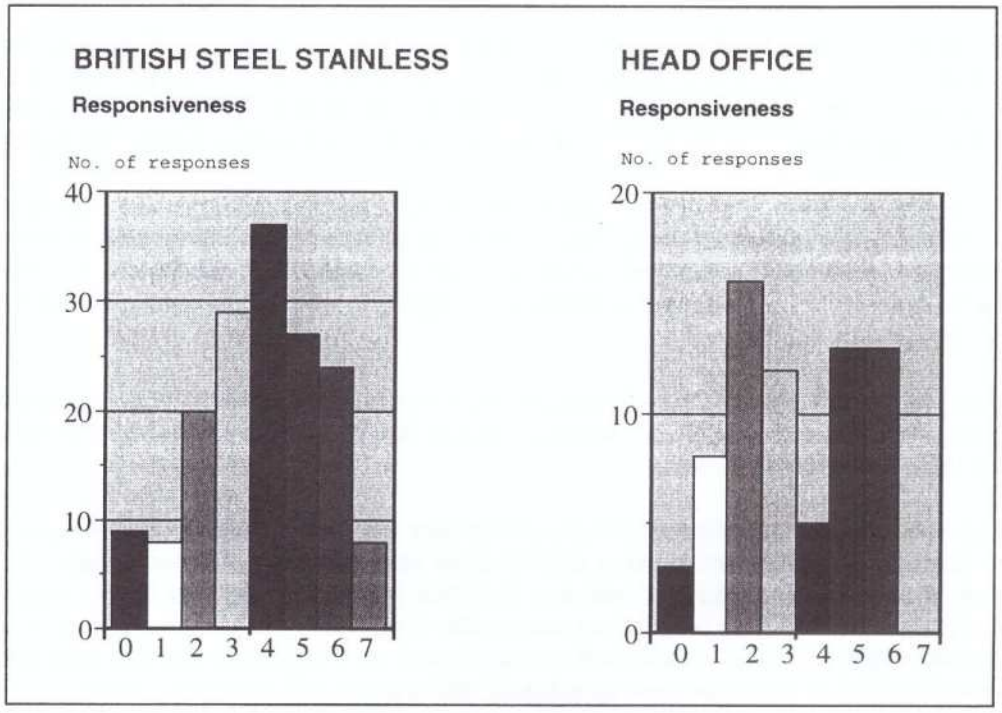
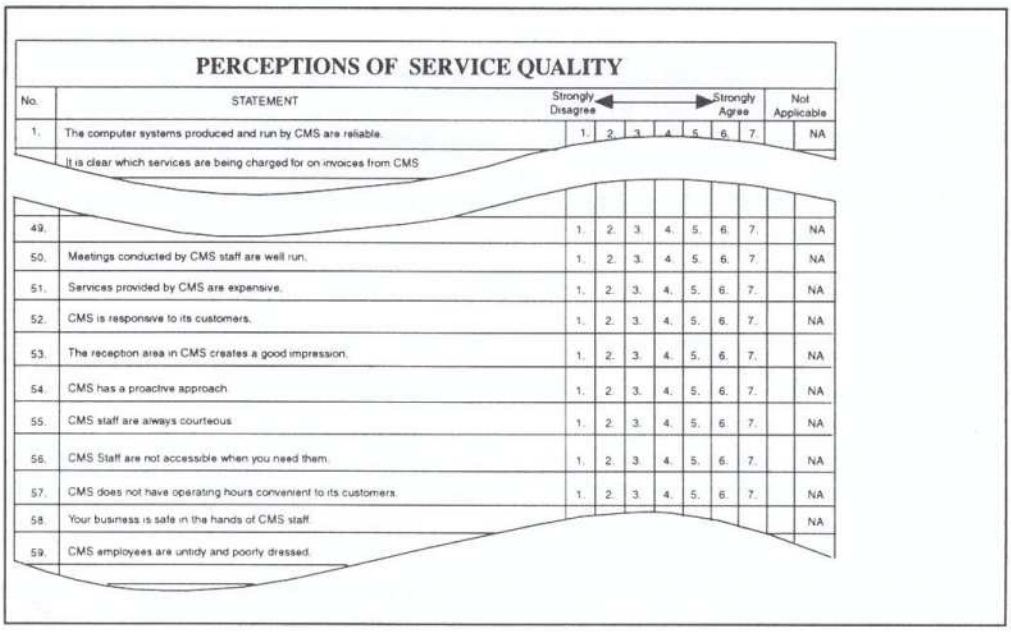


Figure 9&10: Comparison of Customer Responses - Responsiveness

The questionnaire devised to test the customer perception was orientated around ten topics with a series of five questions on each topic and corresponding reworded questions of the opposing view to test the validity of the answers. These were then randomly mixed within the questionnaire. A seven scale rating, 1 = disagree, 7 = agree, was used as the mechanism for the customer to report the perception of service.(Figure 11)



PERCEPTIONS OF SERVICE QUALITY

No.	STATEMENT	Strongly Disagree ◄ ► Strongly Agree							Not Applicable
1.	The computer systems produced and run by CMS are reliable.	1.	2.	3.	4.	5.	6.	7.	NA
	It is clear which services are being charged for on invoices from CMS								
49.		1.	2.	3.	4.	5.	6.	7.	NA
50.	Meetings conducted by CMS staff are well run.	1.	2.	3.	4.	5.	6.	7.	NA
51.	Services provided by CMS are expensive.	1.	2.	3.	4.	5.	6.	7.	NA
52.	CMS is responsive to its customers.	1.	2.	3.	4.	5.	6.	7.	NA
53.	The reception area in CMS creates a good impression.	1.	2.	3.	4.	5.	6.	7.	NA
54.	CMS has a proactive approach	1.	2.	3.	4.	5.	6.	7.	NA
55.	CMS staff are always courteous	1.	2.	3.	4.	5.	6.	7.	NA
56.	CMS Staff are not accessible when you need them.	1.	2.	3.	4.	5.	6.	7.	NA
57.	CMS does not have operating hours convenient to its customers.	1.	2.	3.	4.	5.	6.	7.	NA
58.	Your business is safe in the hands of CMS staff.								NA
59.	CMS employees are untidy and poorly dressed.								

Figure 11: Perceptions of Service Questionnaire

Questionnaires were sent to one hundred customer representatives, targeted at a typical cross section of business managers, users of IT services, and financial controllers.

There was some trepidation at issuing a questionnaire with one hundred questions as to level of returns that would be received, however, the fears were unfounded as a seventy three percent return was achieved. This high return was due to two main factors, a professional approach to the design and format, also, the questionnaire was tested with a small representative sample before the final issue.

The measurement of quality in development has not achieved a corresponding action programme, for example, the defect cause and effect measures indicate that all defects are recognised at the point in the life cycle at which they occur. Whilst this can be considered an excellent result there is some doubt as to the effectiveness of the measurement process. Investigation of alternative measures are currently being considered.

Contrary to the teaching of Deming, the measures used for delivery on time, response to quotations, customer project reviews and financial achievements, are all linked to bonus payment, as a result there may be some 'massaging of the figures'.

Allied with the need to develop effective measurement was the recognition that the relative size of development projects was unknown, making comparisons very difficult in an environment of mainly enhancements to very large operational systems. Function Point Counting has been introduced to establish an effective base line for productivity and quality measures. Attempts to introduce function points in the early 1980s had been abandoned as it was seen as a high overhead with little return for the effort expended. The initiative is currently entering the analysis and feedback phase.

8. Making sure Middle Management commitment is effective

Continuous improvement will only succeed if for all staff the immediate instinct becomes "How can **I** achieve improvements" rather than "how can **they** improve..."

It is important that management continually inject cultural initiatives to 'restoke the flames of involvement' and provide the necessary leadership.

Recent examples directly involving management both in the design and participation of training programmes have been :-

Team working courses, focused on improved working between sections
Creative Thinking Courses, using imaginative techniques to generate new ideas
Leadership seminars, recognising and developing leadership skills

Striving for quality must be a continuous dynamic process in any organisation; it is also a process that is continually under threat from

loss of momentum
apathy
"flavour of the month" syndrome
management back out.

Few managers will exhibit all these facets but there will be occasions when one or more predominates. The believers need encouragement to develop continuous improvement but all this can be jeopardised by it being seen that some managers 'get away with it'.

Recognising the potential for such problems in implementing a quality culture, CMS launched the QMS with the key objectives:

a) Quality is dynamic, continuous improvement is an essential feature

b) Excellence without bureaucracy
c) Improved customer service
d) To gain commercial advantage.

The mechanism for driving the continuous improvement of the QMS was through the Quality Review Board (QRB) and Quality Representatives Committee (QRC), supported by internal and external audits.

The Quality Manager reports the effectiveness of the QMS through the QRB. Board membership includes Senior Management from each of the business, commercial and administrative sections. Senior Managers have a designated responsibility for the operational QMS and continuous improvement.

The Quality Representatives Committee provides for a cross section of the CMS community to highlight problems and to recommend changes in the implemented QMS.

Management commitment is recognised as a prime requirement for the success of a quality initiative. However, the commitment is easy to give; the management actions must support the words.

9. Results Driven Incentives

Developing results driven incentives for the quality programme includes the following:

a) People incentives - monetary and nonmonetary.
b) People empowerment
    ITQ (is this quality?)
    Suggestion Scheme
    Continuous Improvement
    National Vocational Qualifications (NVQ).
c) Customer Teams.
d) External Awards.

There were some implications as to the 'people aspects' of quality in the QMS training programme with some individual benefits identified. However, there were no overt references to the importance of results driven reward structures. In hindsight the programme may have appeared all 'stick' and no 'carrot'.

The recognition of CMS as a quality organisation with the award of ISO 9001, provided collective reward for achievement as part of the organisation, but not to individuals.

This was addressed by the introduction of nonmonetary Customer Service and Technical Achievement Awards to staff and, more recently, a Total Quality Award. Customers are responsible for recommending the customer service awards.

Three of the four constituent items of the CMS quarterly bonus are quality related, being: service availability; delivery; and customer perception of service; all of which directly contribute towards achieving outstanding customer satisfaction.

An early attempt to promote innovation was the ITQ (Is This Quality?) scheme, where any none quality activity, requirement or effect, could be flagged for action. Unfortunately, two vital elements were not included:

a) financial evaluation (not financial reward)
b) responsibility for ownership of an ITQ.

A suggestion scheme, with monetary benefits, was introduced and initially this was very popular, even though most monetary rewards were quite small.

All the above have since been replaced by the continuous improvement scheme, which is a nonmonetary scheme. Although the committee can recommend monetary awards.

People empowerment as an incentive has operated in a number of ways, including the development of an open leadership profile through all levels of management. The leadership training courses, run by consultants in organisational development were considered excellent by everyone who attended.

In developing the certificated QMS it was recognised that there were people in the organisation who were qualified to do their job by experience rather than via formal examination or other means. This failure to recognise these skills was deemed a weakness. To meet the requirements of ISO 9001 that "appropriate records of training shall be maintained", an assessment and recording system was introduced. This became part of the UK IT National Vocational Qualification (NVQ) scheme for the assessment of competence as defined by the job role, a basic essential that other assessment/appraisal systems have never really addressed at a fundamental level. CMS became the first NVQ IT industry assessment centre: a significant results rewarded innovation spin off from the quality programme.

Staff appraisals, conducted half yearly, include upwards and downwards assessment.

Over the past two years a significant aim of reorganisation at CMS has been the 'flattening' of the organisation structure and the creation of multi discipline and multi skilled teams established to meet the customers' specific requirements under the direction of the team manager. Revenue and costs are the responsibility of the team manager. The business manager provides the commercial interface between the customer and CMS.
The requirements of the customer are supported by service level agreements. As the customer teams develop, where necessary, the QMS for a customer team will reflect the specific needs of that customer via the use of local procedures.

External awards are regarded at CMS as being an important aspect of the quality culture.
They serve a number of purposes:

a) ensure the quality activities are maintained and prevent complacency
b) advertise our commitment to customers and prospects
c) improve staff morale.

Since ISO 9001 certification, CMS have achieved a number of awards:

National Training Award (1990) for the QMS training programme
Regional Training Award (Individual, 1991) for the Training Records System
Master Cutler TQM Award (1992)
Investors in People Award (1993)
National Vocational Qualification (NVQ) Assessment Centre (1993).

We are currently developing self assessment to the European Quality Award criteria with the intention of external assessment in 1995/6.

At CMS external awards are seen as mechanism to consolidate the various initiatives, for example, the submission to the Investors In People Award (IIP) referenced over seventy positive actions related to quality and staff deployment.

10. The Future: Managed and Optimised

The SEI Capability Maturity Model is envisaged as being the next significant step in the development of the quality culture at CMS. (Figure 12)

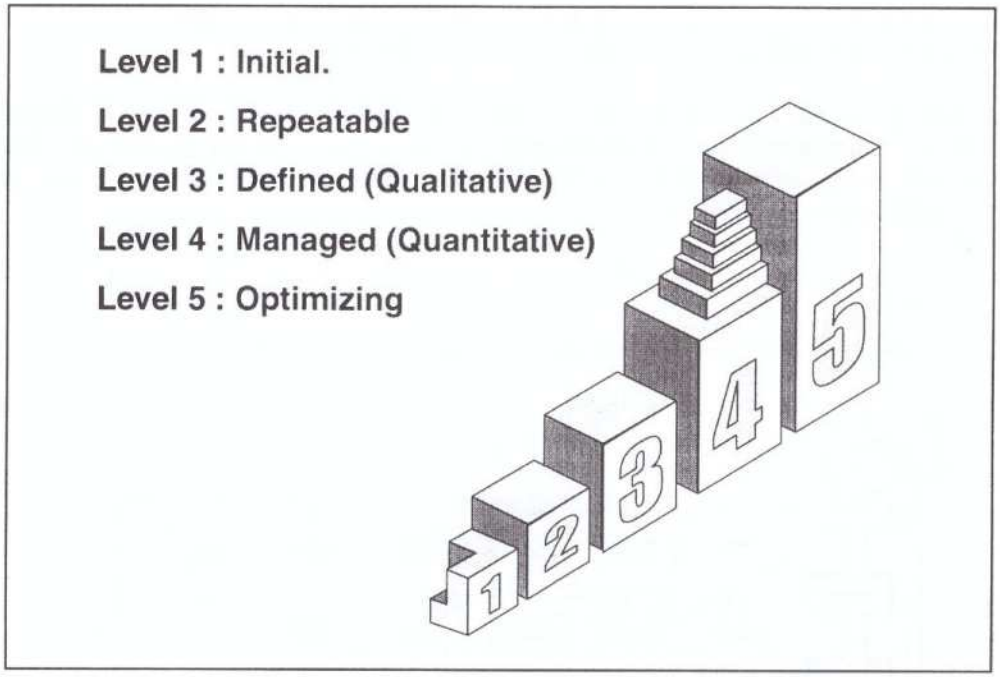


Figure 12 : SEI-CMM Capability Maturity Model

In some small measures the integration of the results of previous quality actions are used to predict quality variances and effective avoidance in the provision of products and services. However the practicalities of implementing the managed and ultimately the optimising level requires a comprehensive and sustainable metrics programme.

It is the aim of CMS to achieve the status of level four in the SEI Capability Maturity Model before the end of 1995. Current status with the operation of defined level of processes is level three, however, this can be visualised as a very long plateau needing considerable effort to move to the next level.

The SEI CMM level four, managed, is essentially quantitative, offering reasonable statistical control over quality. The process architecture allows accurate measurement comparing like with like and isolating variables. The key actions required are:

automatic gathering of process data
economically justified investments
quantitative quality and productivity prediction and tracking

Process optimisation operates at all levels of process maturity. Data is available to improve the optimising process itself. Investment in process improvement is natural because cost justification can be quantitatively accomplished and improvements predicted. The key actions needed at level five are:

constancy of purpose
continual improvement

11. Conclusion

The CMS approach to quality has been to establish a 'way of working' foundation through the introduction of a quality management system based on best practices.

Certification of the QMS to ISO 9001 was considered essential for two reasons:

It provided external continuous assessment of the operation of the QMS
It ensured that the QMS would continue to be developed and maintained.

Certification has provided major benefits. During the turbulent times of major changes and reorganisations it is commendable that the QMS has continued to provide the guiding light. Recent full three year reassessment, as required by the TickIT certification, was successful but it also identified aspects of the QMS needing attention.

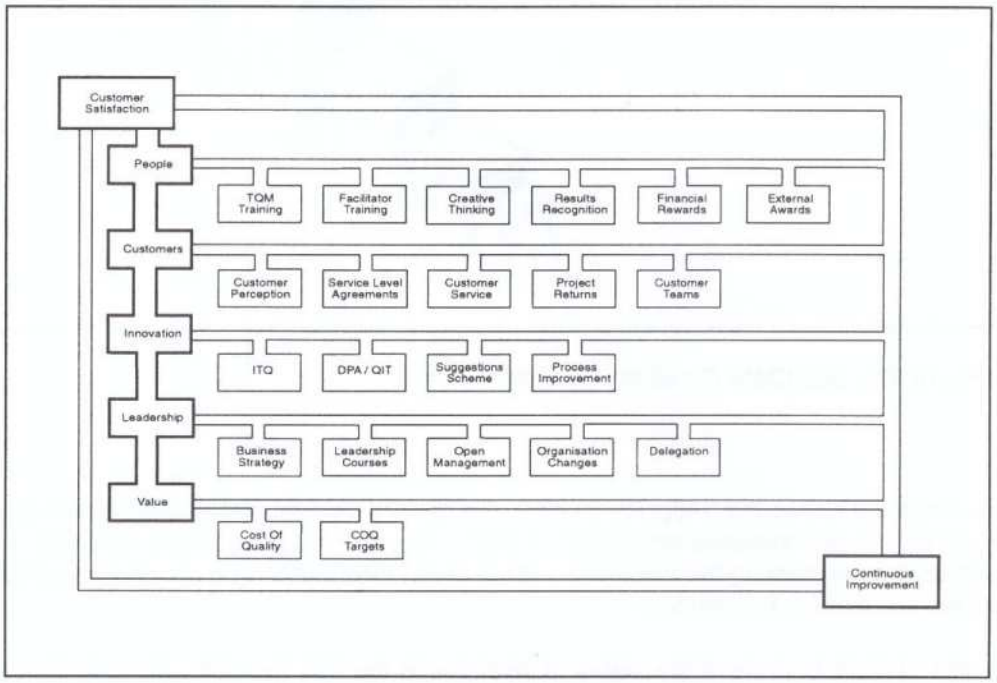


Figure 13: TQM Roadmap

With the benefit of hindsight it is easy to see that the introduction of the Total Quality Management failed to take full advantage of the solid foundations of the QMS.

It is essential to define the 'TQM road map' (Figure 13) for the direction of the TQM activities to ensure the continued commitment from all levels of management and the continued support of all staff during the cultural changes.

The real lesson that has been learned is that it is essential to keep working at achieving further improvements. Each initiative, however small, contributes to the overall quality, reputation, and success, of the business. Management must provide the leadership and direction by visible involvement, whilst at the same time, ensuring there is full empowerment of people.

With the full commitment of management the introduction of a total quality culture is difficult; without it impossible.

Reference

1. Parasurman, Zeithami V, and Berry L (1985); "A conceptual model of service quality and its implications for future research." published in The Journal of Marketing (Fall), 41-50.

Total Quality Management and The European Quality Award

G. Waldner
Alcatel Austria AG
Scheydgasse 41
A-1210 Vienna

Self-Assessment is a comprehensive, systematic and regular review of an organisation's activities and results referenced against a model of business excellence for example the European Model for Total Quality Management. The Self-Assessment Process allows the organisation to discern clearly its strengths and areas in which improvements can be made and culminates in planned improvement actions which are then monitored for progress. Increasingly, companies in Europe accept that Total Quality Management is a way of managing a business to gain competitive advantage thereby ensuring longer term success - meeting the needs of their customers, employees, financial and other stake-holders and the community at large.

Alcatel Austria was the first and till now the only Austrian company to apply for The European Quality Award in 1993, 1994 and 1995. The purpose of this publication is to promote the value of Self-Assessment as a key process for driving business improvement and the benefits to be gained from carrying out internal Self-Assessments and from applying for the award.

The History

Alcatel Austria was founded in 1884 and thus is one of Austria's oldest industrial companies. In 1925 the company became a member of the ITT Corporation. Since the fusion of the telecom group of ITT with that of the French Companie Generale d'Electrique in the year 1987, the group is called Alcatel Alsthom. Our company is a subsidiary of Alcatel N.V. which is part of the Alcatel Alsthom Group.

Alcatel Austria's Total Quality Approach

Alcatel Austria's Total Quality initiative has a long history. It began in the 1970s with the introduction of the ITT Quality Program *Quality Improvement through Defect Prevention* which was conducted by its famous Group Quality Manager **PHILIP CROSBY**. Recognition, certificates and Quality awards of those previous days are still present in many departments of our company to remained staff of our long-standing experience in the field of Total Quality. In the early 1990s our board of directors stepped up the pace of Alcatel Austria's drive to Total Quality with the implementation of Time Based Management and ISO 9001. As first Austrian representative within the telecommunication sector we received external certification for the whole company in 1991 and have maintained our certificate through a reaudit in 1994.

Having achieved ISO 9001 a new challenge was sought to drive continuous improvement. A new process that could clearly discern strengths from opportunities for improvements was necessary. Therefore we set in motion a Total Quality process using the criteria of The European Quality Award. Since 1992 self-assessments play a major part in our business review, because we view the model of the European Foundation for Quality Management (EFQM) as a business model. It is a quality model that links all the aspects of the business, enabling our employees to drive continuous business improvement through Total Quality Management.

Figure 1

ALCATEL

Total Quality Milestones

1990 Start of Time Based Management

1991 ISO 9001 Certification

1992 Start of Customer, Employee Surveys

1992 Implementation of a Quality Cost System

1992 Start of EQA Self-Assessment Process

1993 1st Application for the EQA

1994 Start of Business Process Reengineering

1994 2nd Application for the EQA

1994 Start of SEI Assessments

1995 3rd Application for the EQA

Alcatel Austria Graphic Center

The European Model for Total Quality Management

In order to undertake Self-Assessment, an underlying framework is necessary. An ideal framework is the European Model for Total Quality Management. Although each organisation is unique, this model provides a generic framework of criteria that can be applied widely to any organisation or component part of an organisation.

The European Model for Total Quality Management underlies The European Quality Award and is based on the following premise:

Customer Satisfaction, People Satisfaction and Impact on Society are achieved through Leadership driving, Policy and Strategy, People Management, Resources and Processes, leading ultimately to excellence in Business Results.
The Enablers criteria are concerned with how the organisation approaches each of the criterion parts and the Results criteria are concerned with what the organisation has achieved and is achieving.

Figure 2

Each of the nine elements shown in the model is a criterion that can be used to assess the organisation's progress towards Total Quality. The percentages shown are those for the purpose of The European Quality Award. By using the weightings, an organisation has the additional benefit of being able to compare its score profile with the *Best in Europe*. An application for The European Quality Award will be assessed and scored on a scale from 0 to 1000 points. The model and percentages were derived following a wide consultation exercise across Europe and are reviewed annually by the European Foundation for Quality Management.

To achieve a permanent Total Quality culture which conforms to Alcatel Austria's Visions and strategy, managers use the above mentioned method:

Alcatel Austria's Self-Assessment Process

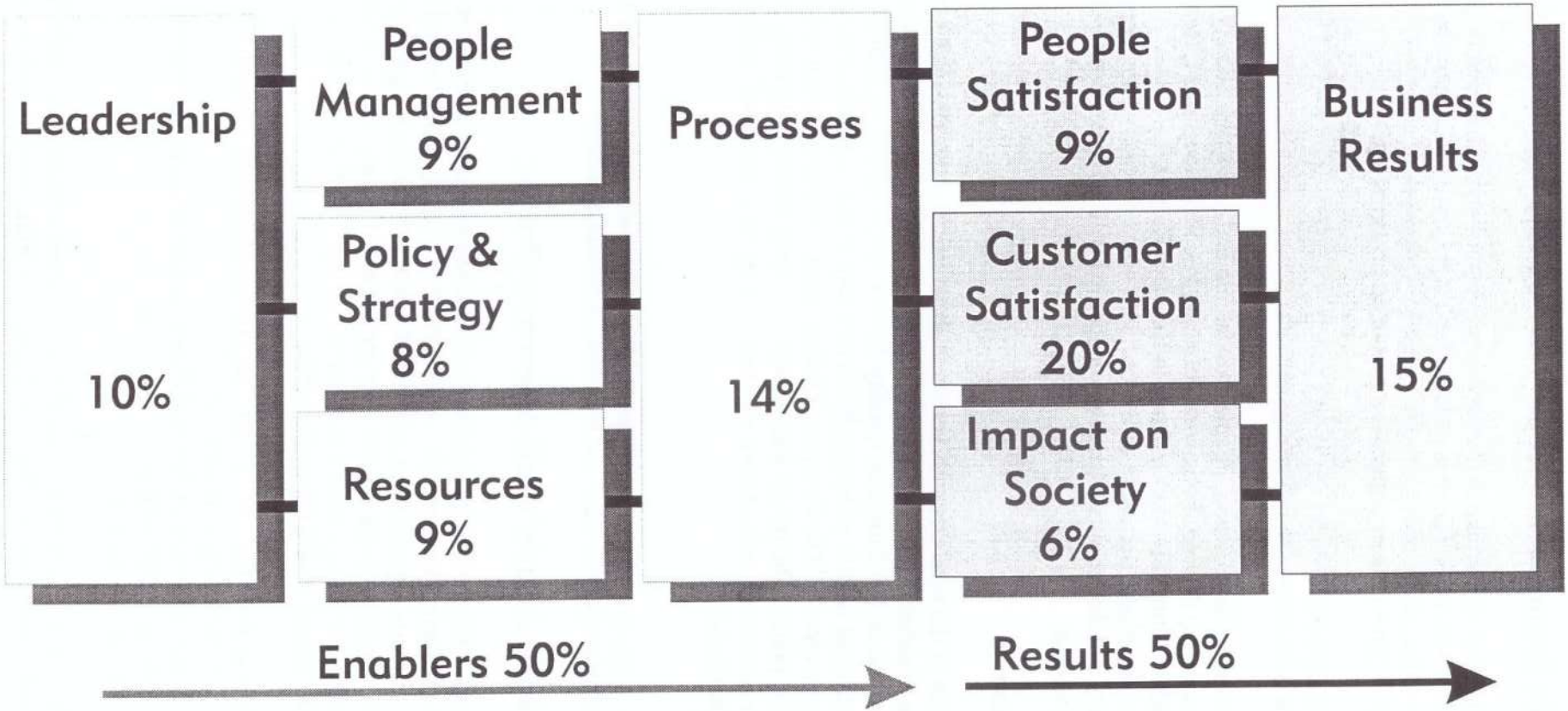
Since 1992 the European Quality Award Model has been recognised by the board of directors as a guideline for the entire Alcatel Austria Total Quality Approach and as the driver for all continuous improvement activities. On basis of a written suggestion by the Commission of the EU, Alcatel Austria was the first and till now the only Austrian company to apply for TEQA. Because of this process managers are strongly involved in assessing awareness of Total Quality and reviewing its progress.

Main steps of the self-assessment process which is to conform to the EFQM Model:
1. Self-Assessment Document
2. Evaluation through Internal Assessors (score, areas for improvement)
3. Management Review
4. Corrective Actions

The self-assessment document is written every year by the TQM-Manager and subsequently handed over to internally trained assessors from every division (cross section).

The European Quality Award Model

9 Elements (33 Criterion Parts)



The European Model for Total Quality Management

In order to undertake Self-Assessment, an underlying framework is necessary. An ideal framework is the European Model for Total Quality Management. Although each organisation is unique, this model provides a generic framework of criteria that can be applied widely to any organisation or component part of an organisation.

The European Model for Total Quality Management underlies The European Quality Award and is based on the following premise:

Customer Satisfaction, People Satisfaction and Impact on Society are achieved through Leadership driving, Policy and Strategy, People Management, Resources and Processes, leading ultimately to excellence in Business Results.
The Enablers criteria are concerned with how the organisation approaches each of the criterion parts and the Results criteria are concerned with what the organisation has achieved and is achieving.

Figure 2

Each of the nine elements shown in the model is a criterion that can be used to assess the organisation's progress towards Total Quality. The percentages shown are those for the purpose of The European Quality Award. By using the weightings, an organisation has the additional benefit of being able to compare its score profile with the *Best in Europe*. An application for The European Quality Award will be assessed and scored on a scale from 0 to 1000 points. The model and percentages were derived following a wide consultation exercise across Europe and are reviewed annually by the European Foundation for Quality Management.

To achieve a permanent Total Quality culture which conforms to Alcatel Austria's Visions and strategy, managers use the above mentioned method:

Alcatel Austria's Self-Assessment Process

Since 1992 the European Quality Award Model has been recognised by the board of directors as a guideline for the entire Alcatel Austria Total Quality Approach and as the driver for all continuous improvement activities. On basis of a written suggestion by the Commission of the EU, Alcatel Austria was the first and till now the only Austrian company to apply for TEQA. Because of this process managers are strongly involved in assessing awareness of Total Quality and reviewing its progress.

Main steps of the self-assessment process which is to conform to the EFQM Model:
1. Self-Assessment Document
2. Evaluation through Internal Assessors (score, areas for improvement)
3. Management Review
4. Corrective Actions

The self-assessment document is written every year by the TQM-Manager and subsequently handed over to internally trained assessors from every division (cross section).

The self assessment training is conducted by an external consultant. By means of a case study the internal assessors learn to identify strengths and areas for improvement and the use of TEQA scoring process. They also learn how to allocate their points for all enablers and results criteria. After the individual appraisal of the self-assessment document by each team member they meet for one day to agree on a consensus view of all strength, areas for improvement and the score. The TQM-Manager and the senior assessor present the results of the self assessment to the board of directors. This review meeting is entirely devoted to self-assessment issues. The board of directors assigns priorities and agrees upon corrective actions by means of single tasks or improvement projects and their results are included in the next application document.

Key Elements of a Successful Self-Assessment Process

- The success of the process depends substantially on the preparation of the self assessment document. The originator of this document has the task of representing the overall performance of the company on not more than 75 pages. If the internal assessors don't recognise their own company in this document the whole self-assessment process will get a poor image. It is very hard for the originator to satisfy this requirement because all managers see the company from different points of view. Therefore the contents of the document should be defined in a way that allows verification during an audit or site visit. Thus the score of the assessment also will depend on the know how and performance of the originator. Note: A poor originator will make out of a 500 points company perhaps a 450 points description. On the other hand, a strong originator perhaps a 550 points document.

- For a better understanding of the requirements of the EFQM Model the originator should read as many case studies as possible to take all company-relevant enablers and results into consideration and to describe the contents exactly in conformance with the scoring charts. The case studies help the originator to identify existing or potential TQM activities in his company.

- Provide external self assessment training for at least one manager (e.g. EFQM Assessors Training for three days) to guarantee sufficient know how in your company. This well-trained manager should be also responsible for the preparation of the self-assessment document.

- Select carefully your internal assessors team. Nominate a cross-functional team (8 assessors) out of experienced middle managers who cover all relevant subjects of the 9 Elements. At least one member of the board of directors should join this team to provide sufficient information from the top level. An external expert should assist the team to help reduce conflicts or misunderstandings. The key manager in the assessors team is the senior assessor who is responsible for moderating the group and for time management. It depends essentially on him if the consensus meeting will be a success or not. Nominate a well-known moderator within your company who has the skill to lead a team and to overrule them if necessary without any lasting conflicts.

- Carry out a one day consensus meeting to discuss the scoring range of the assessors after their individual assessment and to define appropriate areas for improvement.

Example of our self assessment in 1995:

- Scoring range before consensus(32 assessors): 240 Points
- Scoring range after consensus: 120 Points

- Difference of total score before and after consensus: 40 Points
- Difference of total score (max, min) between our 4 teams (after consensus): 10 %

- After your company has carried out one or two internal self assessments try an application for TEQA or a national quality award to learn and to verify your own performance and to get advanced ideas for corrective actions. Give your internal assessors and your management the feeling that your internal results will basically conform to that of external experts.

Because of the positive experience in terms of the self-assessment process within the company, the number of internal assessors involved in this process has increased since 1993 from 2 to 30.
Man days spent for this process (score, consensus) increased from 6 days in 1993 to 90 days in 1995.
All relevant divisions, levels and areas, i.e. marketing, development, operations, procurement, distribution, human resources and quality assurance, participate in this process. At least one member of the board of directors is actively involved in each assessment team.

Deviation Between Internal and External Self Assessment Results

To avoid too much deviation between the internal and the EFQM results the following aspects are important:

- Carry out training for all internal assessors for one day.
- Use one Enabler and one Result criterion for this training and have a consensus meeting. It is an advantage if your assessors evaluate, for example, criteria 3 (people management) and 7 (people satisfaction) of your own self assessment document. They will find out strengths, areas for improvement and the score in conformance with the EFQM scoring charts. It is not necessary to confront the assessors with a complete case study. Middle and top managers don't need more days of training because they are able to understand this simple process in one day.
- Commission an external trainer (if possible an EFQM assessor).
- Explain the Blue Card very carefully. You will have the greatest deviation if assessors do not exactly take this scoring grid into consideration for their individual evaluation. Tell them that they should always take a look at this card during their individual assessment of the 33 criteria parts of the EFQM Model.
- The Lead Assessor should steadily handle the Blue Card during the consensus meeting, particularly after completion of a whole element, if the scoring range of the assessors is as much as 40 percent, or if the discussion allows no consensus.
- Be sure that the writing team of the document and the assessing team consist of different managers. Otherwise the writer will defend the contents of the document during the consensus meeting.
- Say to the assessors that a good company will achieve a total score of 500 points and quality prize winners achieve not more than 600 - 650 points. So you can avoid too unrealistic individual scores.
- A good approach shows a deviation of not more than 10 percent between the internal and external results.

The figure below shows the deviation of our assessors' score from those of the EFQM assessors for each TEQA criterion in 1993 and 1994:

	1993	1994
1. Leadership	+20%	-10%
2. Policy & Strategy	+30%	0
3. People Management	+10%	0
4. Resources	+10%	-10%
5. Processes	0	-10%
6. Customer Satisfaction	0	+10%
7. People Satisfaction	0	0
8. Impact on Society	0	0
9. Business Results	+30%	+20%
Average deviation	±11%	± 7%

Developing a Strategy for Effective Use of Internal and External Assessments

- The real start of the process is the handing over of the company self assessment document to the internal assessors. Therefore it is better to start the assessors training when your description is completed. The assessors will be aware of the logic and non-bureaucratic manner of this business model through this training. Afterwards they like to know the performance of their company in comparison with the requirements of the EFQM Model. You will not have any problems to nominate internal assessors, to train them, to motivate them for a two-day individual study of the company's performance, to bring them together in a consensus meeting for discussion of scores and areas for improvement, and to present finally the results of the overall process to the board of directors during a management review. But you will perhaps have great problems in starting the process itself. It is not necessary to inform the whole staff about the start of this process or to nominate a team for the descriptions if there are manpower problems within the company or there is a saturation because of a lot of different management projects and initiatives in recent years. You can start this process easily with the informal nomination of a project leader and the informal commitment of the top management. The only assumption is that the orignator of the document knows the company very well to define all relevant contents.

- Change your internal assessors each year. In this way, you have the chance to use this process also as a kind of quality training and to spread the TQM philosophy within your whole organisation. Note that the learning experience during this process is perhaps more valuable for your company than the rigid definition of improvement projects. Because of the individual feedback of many assessors the originator can further improve his document for the next assessment and also for a potential application for TEQA. But don't change your senior assessors because they have an important function to improve and stabilise the consensus process. Their professional use of the Blue Card is key success factor.

- Note that the name TEQA may be an unfortunate term. Parts of your organisation will believe that the winning of the Award is the only intention of the management. There would be no problems if the EFQM would replace the word *award* with the word *benchmark*.This would perfectly meet the intention of the company's management and avoid a lot of senseless discussions within or outside the company. Avoid the word *award* whenever possible.

- The intention to win TEQA can be a dangerous strategy for top management. If they announce that the organisation will, for example, prepare itself three years and then apply to win TEQA the chance is very high that the top management will be a loser in the eyes of the employees. Remain realistic - to win TEQA could be a useful vision but not a specific strategic target. Avoid any pressure in this direction. Don't forget that the strategic target is to benchmark your performance, to verify your internal self-assessment results, to get feedback from at least 7 international TQM experts, to get a comprehensive and non-bureaucratic overview about your company in terms of strengths and areas for improvement, to provide excellent business training for your managers, and finally to reduce your distance from the best TQM companies in Europe through continuous improvement activities.

- Involve at least one member of the board of directors in your self assessment team. They know best the strategic direction of the company and thus help to optimise the contents of the document and the self assessment process itself. Note that the output of internal or external consensus meetings normally show areas for improvement which concerns primarily the activity of these managers. So it is up to them to define corrective actions or not. Don't forget that the EFQM Model is only a guideline for Total Quality Management. The strategic thinking of the top managers and the culture of the company determine which areas for improvement are relevant and which not. It is not efficient to follow this model 100 % if the demands in terms of customers, change and competition require other actions for the company. For example, Alcatel Austria is dramatically expanding operations in Eastern Europe. In this type of pioneering situation it is difficult to follow the EFQM Model. Take those areas for improvement into consideration which support best your firm's strategic direction and do not be overly concerned if you do not satisfy all other model's requirements.

- One major outcome of this process is that a company needs comprehensive benchmarking activities. You will never get more than 500 points if you cannot show the value of your results and trends in comparison with other companies, business sector averages or best in class practices. It is also difficult to get these benchmarks easily or immediately. Therefore start as soon as possible with this activity which is quite important for every company. Note that it is easier to increase the score for the enablers criteria than for the results criteria. To avoid a major inequality between these two criteria parts you need primarily benchmarks.

Business Benefits for the Organisation

The following aspects are relevant for each organisation:

- Self assessments which conform to the EFQM Model allow a comparison with the best TQM companies in Europe. We know exactly how many applicants (in percent) for TEQA have a better performance per each criterion than Alcatel Austria. These benchmarks give us more certainty for the formulation of corrective actions.

- We use self assessments as a business training activity. Therefore we increased the number of internal assessors from 8 to 32 (4 teams) to give more employees the chance to get a good overview of our company. Because of self assessments, managers understand better the loops between enablers and results, for example, the relationship between policy & strategy, critical processes and business results. The learning experience for all managers

who are involved in this process is more valuable than the rigid definition of corrective actions.

- Because of the EFQM Model managers understand that Total Quality does not mean any additional activities under the term TQM. Total Quality means the satisfying of all stakeholders: customers, suppliers, employees, shareholders, society. This is our daily business itself.

- Managers recognise the importance of benchmarking and they are more interested in the comparison of their own process performance with that of competitors or with the business sector's average.

- The self assessment results are a good basis for the definition of improvement projects. Without the use of a TQM model you have a high risk of losing the orientation for your improvement program.

- By using of the EFQM Model you can easily increase the motivation for Total Quality. It should be the intention of any manager to reduce the distance to the best TQM companies in Europe, to be the best national TQM company or to be the best subsidiary within a corporation in this subject.

- The EFQM Feedback Report is very helpful. Don't forget that each of the 7 international assessors work at least 3 days with your company, in sum 21 days, but you pay only a minimal fee for this TQM consulting activity.
- The self-assessment approach is very powerful, non-bureaucratic and easy for everyone to understand.

References

- The European Quality Award Application Brochure and the Self-Assessment Guidlines 1995;

  European Foundation for Quality Management
  1200 Brussels, Avenue des Pleiades 19
  Telefax +32 2 779 12 37

- The Self-Assessment Handbook (1994), Chris Hakes, Chapman & Hall
- The Rank Xerox European Quality Award Submission Document 1992
- The D2D European Quality Award Submission Document 1994
- The Use of Quality Award Criteria and Models for Self-Assessment Purposes (Proceeding of the First European Forum on Quality Self-Assessment, Milano, 1994)
- Benchmarking for Competitive Advantage (1993), Tony Bendell, Financial Times Pitman Publishing

Biographical Details

Dipl. Ing. Günther Waldner

Born 1961 in Austria, degree in petroleum engineering; from 1989-1991 implementation of ISO 9001 at a Philips company; since 1991 Total Quality Manager for Alcatel Austria; Auditor for ÖQS; Assessor for TEQA 1995; Trainer on ISO 9000 and Self Assessments in several Austrian Management Institutes.

The European Model for Total Quality Management

In order to undertake Self-Assessment, an underlying framework is necessary. An ideal framework is the European Model for Total Quality Management. Although each organisation is unique, this model provides a generic framework of criteria that can be applied widely to any organisation or component part of an organisation.

The European Model for Total Quality Management underlies The European Quality Award and is based on the following premise:

Customer Satisfaction, People Satisfaction and Impact on Society are achieved through Leadership driving, Policy and Strategy, People Management, Resources and Processes, leading ultimately to excellence in Business Results.
The Enablers criteria are concerned with how the organisation approaches each of the criterion parts and the Results criteria are concerned with what the organisation has achieved and is achieving.

Figure 2

Each of the nine elements shown in the model is a criterion that can be used to assess the organisation's progress towards Total Quality. The percentages shown are those for the purpose of The European Quality Award. By using the weightings, an organisation has the additional benefit of being able to compare its score profile with the *Best in Europe*. An application for The European Quality Award will be assessed and scored on a scale from 0 to 1000 points. The model and percentages were derived following a wide consultation exercise across Europe and are reviewed annually by the European Foundation for Quality Management.

To achieve a permanent Total Quality culture which conforms to Alcatel Austria's Visions and strategy, managers use the above mentioned method:

Alcatel Austria's Self-Assessment Process

Since 1992 the European Quality Award Model has been recognised by the board of directors as a guideline for the entire Alcatel Austria Total Quality Approach and as the driver for all continuous improvement activities. On basis of a written suggestion by the Commission of the EU, Alcatel Austria was the first and till now the only Austrian company to apply for TEQA. Because of this process managers are strongly involved in assessing awareness of Total Quality and reviewing its progress.

Main steps of the self-assessment process which is to conform to the EFQM Model:
1. Self-Assessment Document
2. Evaluation through Internal Assessors (score, areas for improvement)
3. Management Review
4. Corrective Actions

The self-assessment document is written every year by the TQM-Manager and subsequently handed over to internally trained assessors from every division (cross section).

The self assessment training is conducted by an external consultant. By means of a case study the internal assessors learn to identify strengths and areas for improvement and the use of TEQA scoring process. They also learn how to allocate their points for all enablers and results criteria. After the individual appraisal of the self-assessment document by each team member they meet for one day to agree on a consensus view of all strength, areas for improvement and the score. The TQM-Manager and the senior assessor present the results of the self assessment to the board of directors. This review meeting is entirely devoted to self-assessment issues. The board of directors assigns priorities and agrees upon corrective actions by means of single tasks or improvement projects and their results are included in the next application document.

Key Elements of a Successful Self-Assessment Process

- The success of the process depends substantially on the preparation of the self assessment document. The originator of this document has the task of representing the overall performance of the company on not more than 75 pages. If the internal assessors don't recognise their own company in this document the whole self-assessment process will get a poor image. It is very hard for the originator to satisfy this requirement because all managers see the company from different points of view. Therefore the contents of the document should be defined in a way that allows verification during an audit or site visit. Thus the score of the assessment also will depend on the know how and performance of the originator. Note: A poor originator will make out of a 500 points company perhaps a 450 points description. On the other hand, a strong originator perhaps a 550 points document.

- For a better understanding of the requirements of the EFQM Model the originator should read as many case studies as possible to take all company-relevant enablers and results into consideration and to describe the contents exactly in conformance with the scoring charts. The case studies help the originator to identify existing or potential TQM activities in his company.

- Provide external self assessment training for at least one manager (e.g. EFQM Assessors Training for three days) to guarantee sufficient know how in your company. This well-trained manager should be also responsible for the preparation of the self-assessment document.

- Select carefully your internal assessors team. Nominate a cross-functional team (8 assessors) out of experienced middle managers who cover all relevant subjects of the 9 Elements. At least one member of the board of directors should join this team to provide sufficient information from the top level. An external expert should assist the team to help reduce conflicts or misunderstandings. The key manager in the assessors team is the senior assessor who is responsible for moderating the group and for time management. It depends essentially on him if the consensus meeting will be a success or not. Nominate a well-known moderator within your company who has the skill to lead a team and to overrule them if necessary without any lasting conflicts.

- Carry out a one day consensus meeting to discuss the scoring range of the assessors after their individual assessment and to define appropriate areas for improvement.

Example of our self assessment in 1995:

- Scoring range before consensus(32 assessors): 240 Points

- Scoring range after consensus: 120 Points
- Difference of total score before and after consensus: 40 Points
- Difference of total score (max, min) between our 4 teams (after consensus): 10 %

- After your company has carried out one or two internal self assessments try an application for TEQA or a national quality award to learn and to verify your own performance and to get advanced ideas for corrective actions. Give your internal assessors and your management the feeling that your internal results will basically conform to that of external experts.

Because of the positive experience in terms of the self-assessment process within the company, the number of internal assessors involved in this process has increased since 1993 from 2 to 30.
Man days spent for this process (score, consensus) increased from 6 days in 1993 to 90 days in 1995.
All relevant divisions, levels and areas, i.e. marketing, development, operations, procurement, distribution, human resources and quality assurance, participate in this process. At least one member of the board of directors is actively involved in each assessment team.

Deviation Between Internal and External Self Assessment Results

To avoid too much deviation between the internal and the EFQM results the following aspects are important:

- Carry out training for all internal assessors for one day.
- Use one Enabler and one Result criterion for this training and have a consensus meeting. It is an advantage if your assessors evaluate, for example, criteria 3 (people management) and 7 (people satisfaction) of your own self assessment document. They will find out strengths, areas for improvement and the score in conformance with the EFQM scoring charts. It is not necessary to confront the assessors with a complete case study. Middle and top managers don't need more days of training because they are able to understand this simple process in one day.
- Commission an external trainer (if possible an EFQM assessor).
- Explain the Blue Card very carefully. You will have the greatest deviation if assessors do not exactly take this scoring grid into consideration for their individual evaluation. Tell them that they should always take a look at this card during their individual assessment of the 33 criteria parts of the EFQM Model.
- The Lead Assessor should steadily handle the Blue Card during the consensus meeting, particularly after completion of a whole element, if the scoring range of the assessors is as much as 40 percent, or if the discussion allows no consensus.
- Be sure that the writing team of the document and the assessing team consist of different managers. Otherwise the writer will defend the contents of the document during the consensus meeting.
- Say to the assessors that a good company will achieve a total score of 500 points and quality prize winners achieve not more than 600 - 650 points. So you can avoid too unrealistic individual scores.
- A good approach shows a deviation of not more than 10 percent between the internal and external results.

The figure below shows the deviation of our assessors' score from those of the EFQM assessors for each TEQA criterion in 1993 and 1994:

	1993	1994
1. Leadership	+20%	-10%
2. Policy & Strategy	+30%	0
3. People Management	+10%	0
4. Resources	+10%	-10%
5. Processes	0	-10%
6. Customer Satisfaction	0	+10%
7. People Satisfaction	0	0
8. Impact on Society	0	0
9. Business Results	+30%	+20%
Average deviation	±11%	± 7%

Developing a Strategy for Effective Use of Internal and External Assessments

- The real start of the process is the handing over of the company self assessment document to the internal assessors. Therefore it is better to start the assessors training when your description is completed. The assessors will be aware of the logic and non-bureaucratic manner of this business model through this training. Afterwards they like to know the performance of their company in comparison with the requirements of the EFQM Model. You will not have any problems to nominate internal assessors, to train them, to motivate them for a two-day individual study of the company's performance, to bring them together in a consensus meeting for discussion of scores and areas for improvement, and to present finally the results of the overall process to the board of directors during a management review. But you will perhaps have great problems in starting the process itself. It is not necessary to inform the whole staff about the start of this process or to nominate a team for the descriptions if there are manpower problems within the company or there is a saturation because of a lot of different management projects and initiatives in recent years. You can start this process easily with the informal nomination of a project leader and the informal commitment of the top management. The only assumption is that the orignator of the document knows the company very well to define all relevant contents.

- Change your internal assessors each year. In this way, you have the chance to use this process also as a kind of quality training and to spread the TQM philosophy within your whole organisation. Note that the learning experience during this process is perhaps more valuable for your company than the rigid definition of improvement projects. Because of the individual feedback of many assessors the originator can further improve his document for the next assessment and also for a potential application for TEQA. But don't change your senior assessors because they have an important function to improve and stabilise the consensus process. Their professional use of the Blue Card is key success factor.

- Note that the name TEQA may be an unfortunate term. Parts of your organisation will believe that the winning of the Award is the only intention of the management. There would be no problems if the EFQM would replace the word *award* with the word *benchmark*.This would perfectly meet the intention of the company's management and avoid a lot of senseless discussions within or outside the company. Avoid the word *award* whenever possible.

- The intention to win TEQA can be a dangerous strategy for top management. If they announce that the organisation will, for example, prepare itself three years and then apply to win TEQA the chance is very high that the top management will be a loser in the eyes of the employees. Remain realistic - to win TEQA could be a useful vision but not a specific strategic target. Avoid any pressure in this direction. Don't forget that the strategic target is to benchmark your performance, to verify your internal self-assessment results, to get feedback from at least 7 international TQM experts, to get a comprehensive and non-bureaucratic overview about your company in terms of strengths and areas for improvement, to provide excellent business training for your managers, and finally to reduce your distance from the best TQM companies in Europe through continuous improvement activities.

- Involve at least one member of the board of directors in your self assessment team. They know best the strategic direction of the company and thus help to optimise the contents of the document and the self assessment process itself. Note that the output of internal or external consensus meetings normally show areas for improvement which concerns primarily the activity of these managers. So it is up to them to define corrective actions or not. Don't forget that the EFQM Model is only a guideline for Total Quality Management. The strategic thinking of the top managers and the culture of the company determine which areas for improvement are relevant and which not. It is not efficient to follow this model 100 % if the demands in terms of customers, change and competition require other actions for the company. For example, Alcatel Austria is dramatically expanding operations in Eastern Europe. In this type of pioneering situation it is difficult to follow the EFQM Model. Take those areas for improvement into consideration which support best your firm's strategic direction and do not be overly concerned if you do not satisfy all other model's requirements.

- One major outcome of this process is that a company needs comprehensive benchmarking activities. You will never get more than 500 points if you cannot show the value of your results and trends in comparison with other companies, business sector averages or best in class practices. It is also difficult to get these benchmarks easily or immediately. Therefore start as soon as possible with this activity which is quite important for every company. Note that it is easier to increase the score for the enablers criteria than for the results criteria. To avoid a major inequality between these two criteria parts you need primarily benchmarks.

Business Benefits for the Organisation

The following aspects are relevant for each organisation:

- Self assessments which conform to the EFQM Model allow a comparison with the best TQM companies in Europe. We know exactly how many applicants (in percent) for TEQA have a better performance per each criterion than Alcatel Austria. These benchmarks give us more certainty for the formulation of corrective actions.

- We use self assessments as a business training activity. Therefore we increased the number of internal assessors from 8 to 32 (4 teams) to give more employees the chance to get a good overview of our company. Because of self assessments, managers understand better the loops between enablers and results, for example, the relationship between policy & strategy, critical processes and business results. The learning experience for all managers

who are involved in this process is more valuable than the rigid definition of corrective actions.

- Because of the EFQM Model managers understand that Total Quality does not mean any additional activities under the term TQM. Total Quality means the satisfying of all stakeholders: customers, suppliers, employees, shareholders, society. This is our daily business itself.

- Managers recognise the importance of benchmarking and they are more interested in the comparison of their own process performance with that of competitors or with the business sector's average.

- The self assessment results are a good basis for the definition of improvement projects. Without the use of a TQM model you have a high risk of losing the orientation for your improvement program.

- By using of the EFQM Model you can easily increase the motivation for Total Quality. It should be the intention of any manager to reduce the distance to the best TQM companies in Europe, to be the best national TQM company or to be the best subsidiary within a corporation in this subject.

- The EFQM Feedback Report is very helpful. Don't forget that each of the 7 international assessors work at least 3 days with your company, in sum 21 days, but you pay only a minimal fee for this TQM consulting activity.
- The self-assessment approach is very powerful, non-bureaucratic and easy for everyone to understand.

References

- The European Quality Award Application Brochure and the Self-Assessment Guidlines 1995;

  European Foundation for Quality Management
  1200 Brussels, Avenue des Pleiades 19
  Telefax +32 2 779 12 37

- The Self-Assessment Handbook (1994), Chris Hakes, Chapman & Hall
- The Rank Xerox European Quality Award Submission Document 1992
- The D2D European Quality Award Submission Document 1994
- The Use of Quality Award Criteria and Models for Self-Assessment Purposes (Proceeding of the First European Forum on Quality Self-Assessment, Milano, 1994)
- Benchmarking for Competitive Advantage (1993), Tony Bendell, Financial Times Pitman Publishing

Biographical Details

Dipl. Ing. Günther Waldner

Born 1961 in Austria, degree in petroleum engineering; from 1989-1991 implementation of ISO 9001 at a Philips company; since 1991 Total Quality Manager for Alcatel Austria; Auditor for ÖQS; Assessor for TEQA 1995; Trainer on ISO 9000 and Self Assessments in several Austrian Management Institutes.

Essence and Accidents in SEI-style Assessments or "Maybe This Time the Voice of the Engineer Will Be Heard"

Ken Dymond
Process Inc US

Abstract

Software process assessments (SPAs), in the style evolved by the Software Engineering Institute of Pittsburgh from 1987 to now, have been used world-wide to start process improvement in software organizations. Successful assessments depend, in the opinion of the author, on the statement quoted in the title above -- assessments are the voice of the working level in software companies. This article explains how assessment provides that voice and the context for which assessment was designed. The article will also describe some ways in which assessment accidents have been mistaken for essentials.

Section 1. Introduction

I will begin this article by explaining how the quote in the title came about. This statement was made by an engineer at a one-day briefing on assessment and process improvement I was giving with a colleague. The scene was a meeting room in a large company in a European city. The audience consisted of about 45 people from the company, mostly high-level software managers. The front rows were all occupied by managers. As it happened and as I found out, there were some engineers from the working level sitting in the back row.

My colleague and I from the US had spent the day explaining what software process assessment is all about, what happens during the assessment and why, and the effects assessment is intended to have on a software organization. I was becoming a little frustrated as the day wore on because I felt I was not communicating the message about assessments. The attitude of the managers, mostly everybody in the room, was that "We will do this assessment because the top manager [who was not present] ordered it done, and we will be successful at it. Never mind the buy-in and these other things you are talking about [and that I will outline for you in this paper]."

This attitude, not expressed in these very words but evident all the same, meant that my colleague and I were not successful in conveying to this organization what assessment was for. Finally, just before we finished about 5 PM, I invited the audience to state any concerns they had for doing an assessment. A person from the back row raised his hand and said "Maybe this time the voice of the engineer will be heard." Quite courageous, I thought, given that most of the other persons in the audience were this man's superiors and were of the official "party line" view of assessment that I had been hearing all day. That engineer's bold statement was my reward for a day of toil, for his words meant that the message of assessment was understood by at least one person in the room, and that person was a representative of the group assessment was meant to directly help.

The background of that engineer's statement was one that assessment teams often hear. Software practitioners -- engineers -- often describe the background this way: "We have had plenty of quality initiatives around. Management starts them all the time. But nothing ever changes. The agenda is always management's and nobody ever asks the people who do the work what they think should be changed." In the US one says, "Management hoists the quality flag up the flagpole, everybody salutes, and then we go back to business as usual."

Now I want to explain the message of assessment, why assessments, in my view, continue to be used successfully worldwide, as well as some of the misunderstandings that have arisen about assessment.

Section 2. Assessment context

Assessment was designed for and should be practiced in a context of continual process improvement. The context can be pictured as in Figure 1.

The Process Improvement Cycle: IAEIL
(Initiate, Agree, Establish, Implement, Leverage)

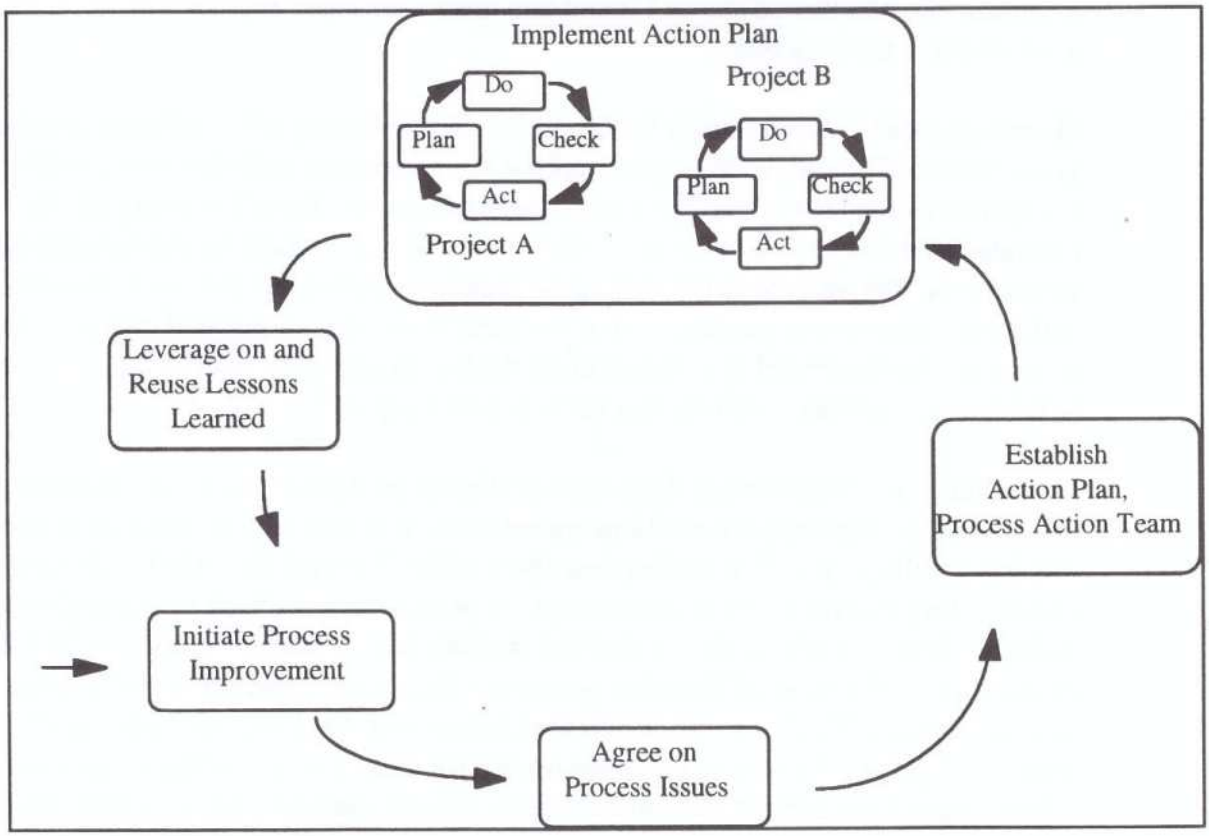


Figure 1

Assessment is one of the best ways to perform the second step, "Agree on Process Issues". But no part of the process improvement cycle can operate unless it is acted upon by people in the organization. The work has to be done by and with people, and so we must add them to the cycle.

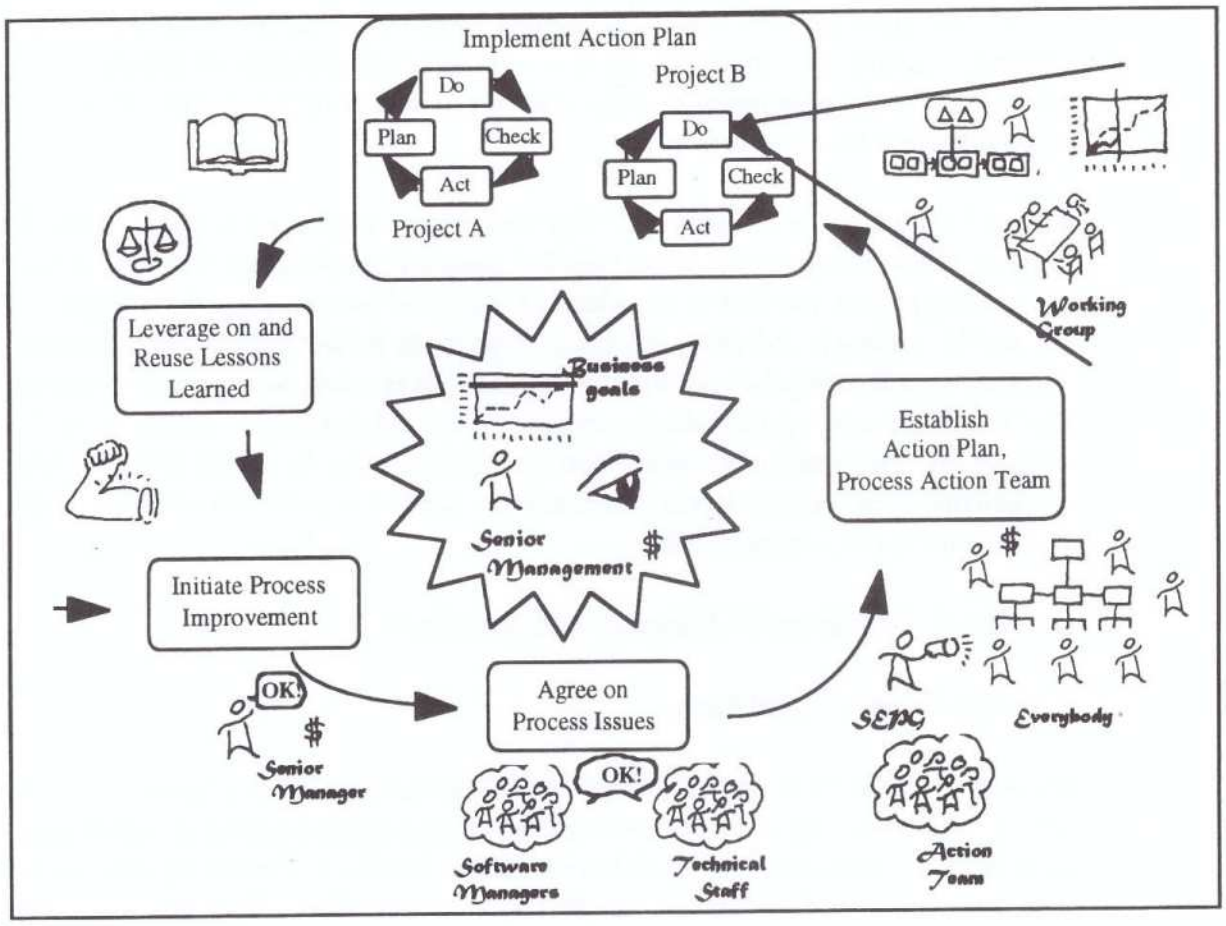


Figure 2

The senior manager initiates the improvement effort and supports it with resources (note the dollar sign), including a company's most valuable asset, its people. Assessment is the next and vital step that generates agreement from management and work force on the process issues that are hampering their work the most. If that step is omitted or performed in an appraisal method that doesn't produce the assessment consensus, the consequences can be failure for the improvement effort to follow (if one does follow). Even if an action plan is written without a preceding assessment to focus the issues, it is

likely to contain the agenda of management, not the workers. And if the action plan does not remove the most important bottlenecks of the workers, the changes merely add to the workload instead of rationalizing and streamlining it.

After the assessment, which produces the consensus on process problems, as many people as possible at all levels should participate in the "Establish" step. This wide participation continues the example of the collaborative activity demonstrated by the assessment in deciding what process problems the action plan will address. A process action team should be the coordinator among working groups focusing on process issues in projects, senior and middle management (which supports and oversees the action plan), and the rest of the organization (which expects to hear good news about improvement and to participate when their turn comes).

Figure 2 shows, in the "Implement" step a working group on a particular phase of a project. The working group, of perhaps 2-5 people from the project and the action team, are helping project workers to introduce an improved process step (the triangle replacing the smaller rectangle indicating a process step) in the project phase. Each working group is following a Deming cycle of Plan-Do-Check-Act to implement a process change, with the "Plan" step coming from the organization's overall action plan arising from the assessment. The result of Working Groups implementing the action plan should be an incremental improvement in the organization's capability to deliver software on time, on budget, and with increased quality.

Section 3. Assessment: Essence and Accidents

3.1. The Process Question

It is easy to miss the point and the outcome of assessment by emphasizing one of its accidental elements as if it were the essence. (I once heard a consultant speak at a conference on assessment and indignantly accuse the SEI of concealing the fact that the SPA was nothing but a means to start an improvement effort.) The remaining sections of this paper try to distinguish the essential features of assessment -- those activities and the manner of conducting them -- that make assessment what it is, from the accidents -- those activities that may be present in any given assessment but need not be.

What is assessment? Software process assessment is a formal and collaborative protocol by which management in a software organization can ask the working level technical people about the process bottlenecks they face in their work.

Why must this simple Process Question be asked with a formal protocol? Why can't management just walk around and ask the "troops"? Why can't management just send out a survey and get the answers?

In most organizations, if you, the manager, walk around and ask the people who work for you about their process problems, they will at the least edit their answers according to what they think you want to hear. So you won't be hearing about their process problems but the ones you have been talking about or hinting at yourself. This distance between manager and worker is a fact of life in many hierarchical organizations. It is not a bad or good fact, merely a natural effect when the boss, who has power over you and can determine your job life, asks you about work performance

Plenty of managers have sent out surveys, usually delegated to a subordinate or to the Human Resources staff or the "quality" group. The usual result is enthusiastic answers to the first round of the survey then disillusionment by the troops when nothing much happens ("We run the quality flag up the flagpole...."). And how many surveys about process problems generate the commitment to divert resources to solving the problems?

Another reason why it is not a simple matter to question subordinates about their work problems has to do with the culture of engineering. Engineers and software workers in general, tend to be a "can-do" bunch of people. They think that with enough resources, and if management will let them alone, they can do wondrous things. Usually they are right. They can do wondrous things, like meet an impossible delivery schedule, but at a high cost in overwork, disrupted home lives because of long hours, and burn out. Also, the high cost means that heroic efforts can't be repeated too often. But management keeps expecting heroic efforts and the technical staff keeps delivering them at the horrendous price.

Still another reason to use a formal protocol to ask software workers about their process problems is that all kinds of answers are given, and some cooperative method is needed to organize the responses. Conducting the assessment in a collaborative way to obtain the views of project leaders and technical workers is one way of organizing the process issues. Another organizing factor of assessments is the Capability Maturity Model (CMM) for software, the generic process standard for systematically improving the software process.[1]

Section 3.2. What activities happen in assessment and what are their outcomes?

Assessment has been characterized as a social ritual.[2] When we try to describe an assessment solely in terms of what happens, we are apt to miss its social nature. In this section I will describe briefly the assessment activities, then in the next two sections point out what is essential about the social ritual involved.

Figure 3 is a version of the basic diagram the SEI and assessment teams had used from about 1987 to 1994 to describe assessment practice. The SPA generally takes 5 days and consists of the activities in the diagram. (I exclude the separate phases of preparation,

training, and any ensuing final report.) These activities can be grouped into major phases: formal opening (opening meeting); gathering information (questionnaire analysis, project leader interviews, and functional area discussions); preparing and testing findings (this happens in two sub-phases: (1) preliminary findings and feedback to project leaders and (2) final findings draft and develop briefing through the dry runs); presentation of findings (formal conclusion); and preparation for producing a final report and action plan (draft recommendations, assessment debrief, and next steps).

A team of 5-9 people, trained for the specific assessment, conducts the activities of the 5 days. Most assessment team members are from the organization assessed, but on a first assessment one or two outsiders who are authorized by the SEI as lead assessors may be on the team as guides.

Activities in the Typical 5-day SEI-style Software Process Assessment (SPA)
(Interactions between assessment team and interviewees are outlined with solid borders)

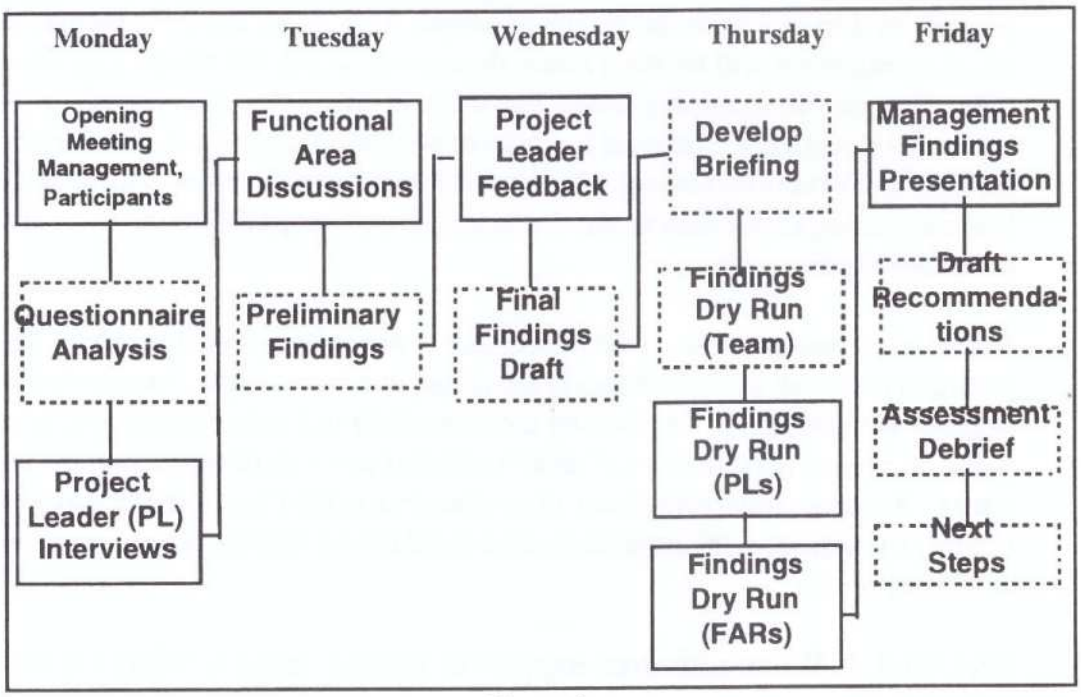


Figure 3

The activities diagram doesn't convey the meaning or experience of assessment. We say in training courses that hearing about assessment is like going to lectures on swimming. The student really doesn't grasp the essence of that activity until he or she enters the water. Assessment is learned by doing.

The essential assessment outcomes are the agreement on the most important process bottlenecks and the commitment to improve them. In other words, assessment produces, almost always, buy-in throughout the software organization for a team effort at process

improvement. On whose part does this agreement, commitment, and buy-in occur? On the part of senior management and the software work force. Thus, the result of using this formal technique for management to ask, and the work force to answer, The Process Question, produces much more than a list of findings at the final presentation.

Section 3.3. What produces the assessment outcomes?

If, as I said above, assessment is a formal technique for management to ask workers about their process problems, one of the crucial activities is the opening meeting. The senior manager must be present and must state publicly the reasons why the assessment is being undertaken and why he or she is asking for the support and enthusiastic participation of everyone. Likewise, the senior manager must attend the management findings presentation on Friday. This formal closing fulfills a commitment the senior manager makes in the opening meeting to hear the process issues of the work force -- that is, to listen to the answers generated by this formal technique for asking The Process Question.

Both the formal opening and closing meetings are "big tent" activities, where everyone involved in software -- the whole site, if possible -- are gathered to witness the commitment of the organization including senior management to do something about the process issues the work force has highlighted in the assessment. The opening and closing meetings are the formal occasion for the senior manager to join the improvement team.

The work force drives the assessment activities and thereby join the team effort during the phases of information-gathering and of preparing and testing findings.

In the project leader discussions, the people responsible for getting software produced on time have the chance to describe their process problems to the assessment team. These discussions are highly structured: one project leader at a time (from a total of 4 to 5), responds to 20 or so open-ended questions about software engineering practices[3]. These questions are based on the process areas in the Capability Maturity Model.

In the Functional Area Representative discussions, the people who actually do the work of software -- designers, coders, testers, quality assurance staff, configuration control staff, documentation specialists -- have the chance to air their process problems to the team. These "FAR" discussions, as they have been called, are for the most part without structure except for what the FARs bring to it. The assessment team listens for about 1 and 1/2 hours and intervenes only to equalize the amount of speaking time between FARs who are more talkative and those who are shy.

The phase of preparing draft findings is done by the assessment team alone, but the practitioners and project leaders -- the interviewees -- test the findings for the team. There are separate feedback sessions for project leaders -- one project leader at a time with the team -- of "preliminary findings." The preliminary findings are rough statements

of the process issues the team has summarized from the individual interviews with 4 or 5 project leaders and from the 4 (usually) group discussions with about 8 practitioners each, amounting to a total of about 35 people -- for most sites a fair cross-section of viewpoints on the software process.

The team next drafts the findings briefing, no longer preliminary but in the format to be delivered at the Management Findings Presentation. The team takes into account the feedback of the project leaders on the preliminary findings in drafting the briefing. The team also now applies the Capability Maturity Model (CMM) as a context for the findings. Thus the process issues that concern the interviewees the most are expressed in terms of the CMM whenever possible. This is because process practices at a certain maturity level must be in regular use as a foundation for installing higher level practices, according to the CMM. In any case, the interviewees' process issues, whether expressed in CMM terms or not, become assessment findings -- the voice of the engineer must be heard. Assessment findings are thus also a means of interpreting or customizing the generic CMM to an organization's particular culture, business market, and application domain.

The feedback sessions with project leaders are the assessment team's first test of whether the process issues gathered from the interviewees are the right ones. The team presents the draft briefing 3 times in rehearsals -- called "dry runs" in the American idiom -- in advance of the management briefing. The first dry run is by the assessment team leader to the rest of the team (merely a practice session for the leader to become comfortable with the rhythm and wording of the briefing). There are two more rehearsals for the interviewees. One rehearsal is for all the project leaders together; the other dry run is for all the FARs together.

At both of the latter rehearsals, the team leader presents the draft briefing while the rest of the team writes down comments, reactions, and suggestions from the audience of project leaders or FARs. These rehearsals allow the interviewees to see whether the team has captured their issues and to offer corrections where it has not.

After collecting the comments from the dry run audiences, the team revises the content of the findings as indicated by suggestions from the interviewees and prepares for the management presentation the next day.

Section 3.4. What activities are essential to the assessment and how must they be conducted?

To produce the agreement on process issues and the buy-in to do something about them throughout the whole software organization, the assessment must include certain activities which must be conducted in a collaborative way. I will point out from my experience what elements are essential and what makes them collaborative -- a team effort.

I will describe in this section the social ritual in these activities that make the assessment different from any other way of appraising an organization's software process.

First of all, the opening and closing meetings are crucial because they bring together, in an unusual and startling format, as many people as possible who are involved in the organization's software process. Clearly something big is happening: a large part -- perhaps all -- of the work force is present. The top manager is there, usually with subordinate levels of management who do not wish to miss what the boss might say. At the opening meeting the team leader explains briefly the assessment process. Many workers will be skeptical ("We run the quality flag up the flagpole...."). To allay their skepticism, the senior manager stands up and says publicly that he or she supports the assessment as well as the ensuing improvement effort. And -- very importantly -- the senior manager asks the interviewees to be candid with the team and promises that the findings and process issues will not be attributed to any person in the organization. No one will be blamed for freely expressing their view of process problems. The senior manager's presence and spoken words start the thinking among the work force that perhaps this time the quality effort will be different.

Next, the face-to-face interviews with the team take place. The interviews are evidence to the people involved that their opinions are being sought and that those opinions, in a non-attributive fashion, will be made known to management. The importance of their opinions is demonstrated by the team writing down the interviewees' statements. The promise of non-attribution -- confidentiality of the source of any information the team obtains from interviewees -- is a promise at this point in the assessment, not a demonstrated fact. But there is evidence that the team will keep its promise.

An effort has been made so that there are no reporting relationships between team members and interviewees or among the interviewees. This means there are no managers of any interviewees on the team. And the practitioners have been selected so they are all peers in terms of the organization's hierarchy. Sometimes it happens that a manager is on the team, usually because a small organization just does not have enough people to fill all the team and interviewee roles. When this occurs, the manager-team member steps out of the room before an interview starts. The team leader then asks the interviewees in that group if any one of them would feel uncomfortable or be less candid if the manager-team member were present during the discussion. (Often the discussion may concern an area managed by the absent team member.) If even one person does object, the manager-team member is excluded from that interview. The rest of the team may not share their notes from that interview with the manager-team member (unless the notes are "scrubbed" so that no source of information is revealed). This absolute concern by the team for confidentiality is a sine qua non of the assessment method and evidence to the interviewees that they will not be penalized for giving their true opinions about process problems in their work -- even if the processes causing their problems are favorite ones of their boss!

In the FAR discussions, assessment teams often see a transformation among the interviewees. FAR discussions tend to be lively. Imagine a group of about 8 engineers or technical people, fond of their work and full of suggestions about how to do their work better. They are given free rein for an hour and a half to express their opinions about work problems and how to solve them. None of their managers are present. There is a team of 5 to 9 people who are going to listen to them carefully and take down their opinions. In most organizations nothing like this has ever happened before.

Because no managers are present and because the assessment team has no agenda except a minimal one for this session, the interviewees conduct the discussion. It is entirely theirs. Usually someone starts by talking about a particular problem they are having, say, with not enough time for testing. Someone else may state a similar problem. For a while, the discussion may follow this thread. Then someone may say, "We had that same problem on our project and this is what we did about it. ...." Another person may say: "I didn't know about that. Can we talk about that later? I'd like to find out more about your solution."

This last exchange marks the transition I just referred to: from airing problems to sharing solutions. The most subtle part of the transformation is from skepticism about the assessment to generating the buy-in for a team effort at changing the work process. In many companies the moment marks the first time people from different projects are working together to improve the organization's capability. The transition is an exhilarating one for members of the assessment team. Experienced assessors watch for it to happen during the FAR discussions and consider it part of the "magic" of assessment.

These free-wheeling FAR discussions have been carefully conducted to produce that "magic" -- in other words, the magic is part of the method. The FARs have been selected to include opinion leaders among the work force, people who are respected by their peers and sought out for advice. In case the discussion falters, the assessment team is trained to prompt gently. Just before the end of the session, the team leader asks the "Big Question" which all the interviewees answer one-by-one: "If you could change one thing about your work [and team leaders traditionally add: "other than your boss or your boss's salary"], what would you change?" The team leader then asks each practitioner about strengths of the organization, perhaps with the words: "If everything about the way you work were going to change tomorrow but you could keep one thing the same, what would that be?"[4]

In this way, the team gets a sampling of opinion on the most crucial process problems as well as on process strengths that other parts of the organization might adopt in the follow-on improvement effort. Process strengths and problems, if corroborated in the dry runs, become ingredients in the findings briefing. This is another way that assessment tries to make sure that the voice of the engineer is heard.

The rehearsals of the draft findings are conducted to make sure the team is conveying to management the voice of the whole software organization. Project leaders, interviewed before singly but now in a group, have a chance to approve process findings and to suggest changes before upper management sees them. Neither their managers nor their subordinates -- many of whom were among the FARs -- are present in the dry run for project leaders. Similarly in the findings dry run for the FARs, none of their managers are present.

In these two dry runs, the team usually discovers that the assessment method has worked because practitioners typically suggest few changes to wording, but the changes they do suggest are important ones for nuance and context. Though rehearsing the words for the team to use the next day is vital, there is another essential outcome of the dry runs that is just as important: the practitioners come to "own" the findings. By joining in the dry runs after giving of their time and their opinions in the previous face-to-face meetings of the week, the interviewees now become virtual members of the assessment team. The dry runs are the next-to-the-last step in forging a living example of how the whole software organization can become a process improvement team.

Consider the "social psychology" of the assessed organization at this point, the evening before the culminating presentation to management. On this Thursday night, all the interviewees know substantially what the briefing will contain, except for last-minute changes the team might make as a result of suggestions during the dry runs. The interviewees, as virtual team members, are asked not to discuss the briefing with anyone else until after the final briefing the next day. The most important person who does not yet know the findings is the senior manager. That person has deliberately put himself or herself in this psychologically exposed position in order to give room to the organization to answer the question about process problems. To get that answer freely and without "adulteration" by management's views, the senior manager has sponsored the assessment and has thereby become a member of the improvement team. The senior manager also thereby expresses his or her trust in the validity of the work force's viewpoints. And the workers -- project leaders and FARs - understand by these actions that management is empowering them to improve their work life, productivity, and product quality by going through the assessment.

Finally, on assessment day 5, the last essential piece occurs at the final findings presentation. Here the answer to management's question -- "What process problems are you facing in your work?" -- is presented by the team leader. This short briefing, about 45 minutes, is the high point of the assessment. And the psychological and social dynamics have been set in place by the assessment method.

Of all the people in the room, the 35 or so interviewees and the 5 to 9 assessment team members already know the content of the findings. The senior manager doesn't know

what the assessment results are, but many of the people who work under his or her direction have helped to craft the findings. Other members of the organization have come to hear the findings; they have had to make allowance in their work schedule all week for their colleagues who were being interviewed or going through a dry run and are usually curious by this time to hear the result. At least some part of everyone's attention is on the senior manager to judge his or her reaction while the team leader presents the findings, about 7 in number, one-by-one.

And what is the result of the intense activities of assessment week? What is the outcome of the assessment and why was it worth it?

The result the team expects -- and almost always gets -- is for heads in the audience to nod agreement with the findings. The nods are signs that the team's findings are the ones the audience wanted to convey to management and to the rest of the organization. The findings are the main problems the work force faces every day, has to solve with process "patches", and the ones that hamper their creativity and productivity, lower morale and lessen the competitive software capability of the organization.

An even better sign -- and this happens occasionally -- is for someone in the audience, usually a manager who has not been interviewed, to challenge one of the findings. If you are a first time assessment team leader, you may be a little shocked by this challenge and apprehensive about responding. What do you do? You wait a moment before answering. The chances are likely that someone in the audience -- an interviewee perhaps -- will respond for you and say "Yes, this finding is true!" (Remember the engineer's statement quoted in the title of this article.) And if not? You should recall for the audience what you and the team have done in the past week: you have followed the assessment method, designed to produce consensus on process issues; you have rehearsed these findings with knowledgeable people who gave your team this information in the first place; you know that these findings represent the working level issues. At this moment no one in the organization has a better understanding of the process issues than the team.

Finally the briefing comes to an end. The team has prompted the senior manager beforehand to stand up and say a few words at this point. Usually managers do not need this prompting because they recognize full well the improvement impetus that can come from the unity of opinion in the room. The typical statement of a senior manager at this point goes something like this: "The findings are no surprise, people have known about them a long time. [No wonder, since there have probably been undercurrents about these problems going on for a long time, but this is the first time that the whole organization can recognize them as problems.] The next step is for us to capitalize on the assessment, form a process improvement team, and build an action plan together. With the help and input of everyone, we'll write that plan and implement it and change those processes so we can make a difference in our capability to deliver software better than our competitors [or for the business goal appropriate to the organization's market]. Finally, I commit to

implementing the action plan and expect and want everyone to participate appropriately."

This short findings briefing and the reaction of the audience and the senior manager may seem a small outcome for a large investment. What has been produced to justify the investment? Consider what the organization has achieved:

1) Agreement on the main process issues in the whole organization produced in a systematic way.

2) Buy-in for starting the improvement effort:
- from the work force who were asked their issues by management in a compelling way, face-to face by their peers, not in a mere survey;
- from the senior manager who demonstrated his or her commitment by
  -- funding the assessment;
  -- participating at the opening and closing meetings;
  -- publicly promising support for an improvement program.

3) A worked example -- the assessment -- of how the whole organization can collaborate in an improvement effort.

For my part, as assessment team leader, I appreciate the courage and human understanding of the senior manager who undertakes to put himself or herself in the psychologically vulnerable position resulting from the assessment -- even though momentary. Senior managers are not supposed to show any weakness. But most senior managers instinctively understand the opportunity that assessment, in the style I have outlined above, gives them to focus the whole organization on process improvement and to form a team composed of management and technical workers to carry out improvement.

Assessment is a radically different event than most organizations are used to and marks a cultural change. This culture change is very much a part of the method, because genuine process improvement is a definite break with past ways of operating for most software organizations who have been concentrating --in response to market pressure and rightly so -- on delivering the product to the customer.

Of course, it is possible to just go through the motions of an assessment and not really produce the three outcomes above. Sometimes this happens, usually because there is not a shared understanding between management and work force as to the purpose of the assessment, or because one of the accidents discussed in the next section is considered as the focus of the assessment. But much more often the assessment "magic" happens at the final briefing.

Section 3.5. What is accidental in assessment?

It is important to look at the non-essential elements in assessment activities, especially at this time when other "appraisal"[5] methods are being developed. Many organizations are developing or practicing various appraisal methods: the SEI itself in the CMM-Based Appraisal for Internal Process Improvement or CBA-IPI (initially perhaps with not enough public input from experienced team leaders, judging by remarks at the first meeting of the CBA-IPI Users' Group[6]); ISO with SPICE; ISO with the revised 9001; and governments with their various kinds of evaluations of suppliers.

Another reason to examine what is accidental as opposed to essential in assessments is that there continues to be discussions in the literature and at conferences about this or that failing of the assessment method. Many of these discussions miss the point by focusing on an accident of the assessment method. And many of these discussions have sparked other appraisals, which while perhaps effective for their own purposes are less likely to produce the assessment "magic". I will discuss three viewpoints that in my opinion mistake accidents for essentials and try to describe why these viewpoints stray from the essential.

3.5.1. Assessment is the maturity questionnaire.

In the early days of the SEI assessment method, from 1987 to 1992, many people thought that the most public aspect of assessment, the questionnaire on process maturity, was the entire method. This view mistook the part for the whole. This view is understandable since those of us conducting assessments at the time hardly paused to describe adequately the essence of what we were experiencing. Also, it took assessment practitioners time to understand the nature of the effects assessment produced and to sort out essence from the accidents.

For most of those years, the locus classicus of both assessment and evaluation (the software capability evaluation or SCE, a US Department of Defense software process audit) was the technical report SEI/CMU-87-TR-23 "A Method for Assessing the Software Engineering Capability of Contractors" by Watts Humphrey and William Sweet. This short report described well the purpose of assessment to help contractors prepare process improvement plans to meet DoD requirements for software capability of contractors. A large part of the report contained a list of 101 questions used by both assessment and evaluation teams, that is, in both collaborative self-appraisals and in adversarial, external audits. The fact that the U.S. government, a very large customer market, was going to use SCEs based on the questionnaire to evaluate software suppliers meant that US industry focused on the questionnaire as the essence of both assessment and evaluation.

In assessment practice, the answers to the questions were used by the assessment team to get an initial estimate of the software maturity level of the site. This estimate helped the team to draft the 20 or so open-ended questions to ask of each project leader in the brief time (about an hour) of an interview. Depending on the maturity level of the questionnaire responses from 4 or 5 project leaders, the team would know whether to concentrate on process maturity issues from level 2, 3, 4 or 5 in interviews with project leaders. Thus the questionnaire became a tool to manage the time and content of the project leader interviews.

After the project leader interviews and FAR discussions were completed, the team also reviewed the initial questionnaire responses to see if the estimate of maturity level was confirmed by the information-gathering done face-to-face. The information from live discussions has always been considered by most experienced assessors as more reliable than responses to the questionnaire. The questionnaire is like a ladder the team uses to climb to a certain plateau of information and then discards as it continues the assessment. Those of us who trained assessment teams at the SEI made this point during the training classes, but it has never been emphasized to my knowledge in the published literature. Other assessment methods that rely heavily on a questionnaire are not likely to achieve the buy-in from practitioners that asking them face-to-face in a collaborative way produces. Assessments can and have been done without a questionnaire, but an appraisal without collaborative interviews is not an assessment

3.5.2. The main purpose of assessment is to gather information.

Audits are very good at finding out detailed information on software practices at a site. Audits are performed by outsiders who examine documents and who may interview practitioners. Because audits are performed by outsiders to ensure objectivity, audit teams almost never include practitioners of the process under view. Assessment teams must include peers of the interviewees (provided the free flow of opinions is not hampered by a reporting relationship between a team member and an interviewee, as mentioned above in Section 3.4).

There are three reasons why assessment teams include practitioners of the site's software process: first, assessment is a living example that the site people can perform a process improvement activity in the midst of the schedule pressures inherent in the software industry; second, peers feel more comfortable talking to peers about their process problems rather than to outsiders or to managers; and third, some of the team members will more than likely serve on a process action team (in the "Establish" step of the improvement cycle), and the assessment interviews introduce them to a cross-section of the site's software work, a viewpoint which no one at the site may have had before. Thus, assessments do gather information but for specific future improvement actions at the site.

3.5.3. Assessment results should be quantitative.

It is likely that quantitative process findings are valuable for a site that has already gone through the improvement cycle once and has a basis of comparison. But in the usual case of an organization starting its first improvement effort, there are few baseline process metrics collected and thus little or no data to report. It is also apparent that a large database of quantitative process information is needed to make comparisons of software capability across companies and internationally.[7] Questionnaire responses and the maturity levels have been collected by the SEI for studies of trends in assessment results.[8] While comparison data helps to get a macro view of software capabilities on an industry, on a national or international scale, the purpose of assessment is to start an improvement effort on the micro scale of one site. For that purpose, industry comparisons may be interesting but are largely irrelevant to the systematic actions an organization may take for improving its software capability.

It is sometimes argued that managers need quantitative information about the return on investment to process improvement before they will undertake something like an assessment. My experience has been that in some cases such data may help but even data will not convince a reluctant manager ("Yes, those numbers are impressive, but our organization is different.") There seems to be no "silver bullet" or magic remedy that removes a senior manager's skepticism. Some managers take a long time to finally agree after repeated approaches by their subordinates or by hearing the idea at conferences. Others seem to grasp quickly the value of process improvement and see assessment as its starting point and as an opportunity to establish a relationship of regular two-way communication with the work force. Of course, managers have the ultimate responsibility for their organizations and with their knowledge of current business conditions may rightly judge that the added stress assessment brings is inopportune at the time.

My last argument on this point is that assessment is less about science, which broad data gathering helps, than about an organization's internal transfer of technology. Process improvement is different from scientific research. If you wait to start a process improvement effort until all the evidence of its value is in -- as seems to be urged by the research view of software engineering[9] -- your company may have lost advantage to competitors who have the head start that assessment provides.

Summary

My intention in this article was to show the nature of assessment: that it is a method for a senior manager to ask his or her organization for a candid view of their process problems and to forge an organization-wide, team impetus for improving that process. This dual purpose captures the double-, even triple-, acting nature of the collaborative style of the SEI assessment. Though apparently only one activity may be going on at a time, there is a great deal of parallel processing in assessment. For example, the FAR

discussions not only gather information about process problems facing practitioners but also generate some solutions from colleagues in the discussion and start the idea that improvement is possible. The FAR discussions also embody an example of projects cooperating across organization boundaries to reuse their best process and technology practices.

Assessment SEI-style, is a powerful means for senior managers to turn the attention of a whole organization away from daily routine for a time and onto long-term process improvement. For most organizations who have not started a systematic improvement effort, an assessment is a routine-breaking way of beginning. The unique nature of assessment and the results it produces should be understood by managers and practitioners contemplating its use and by national and international committees developing alternate appraisal methods.

---

[1] See Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, and Charles V. Weber, "Capability Maturity Model for Software, Version 1.1", CMU/SEI-93-TR-24, Software Engineering Institute, Pittsburgh, PA, February, 1993. Also Mark C. Paulk, Charles V. Weber, Suzanne M. Garcia, Mary Beth Chrissis, and Marilyn W. Bush, "Key Practices of the Capability Maturity Model, Version 1.1", CMU/SEI-93-TR-25, Software Engineering Institute, Pittsburgh, PA, February, 1993. See also Kenneth M. Dymond, *A Guide to the CMM*, Process Inc US. Annapolis, Maryland, 1995.

[2] Remark made by Ron Radice, then Director of the SEI's Process Program, in a presentation at the EFDPMA Conference on Software Process Improvement and Automation, 4 November, 1993, Washington, DC.

[3] An open-ended question is one that asks for an explanation as opposed to a one-word answer. For example, the question "How is project planing handled on your project?" invites a description of a process, whereas "Do you do project planning?" invites a "yes," or "no," or "always," etc., as a response.

[4] For this particular formulation of the "strengths" question, I am indebted to Jeff Bleiberg of American Management Systems, Inc., who introduced it to me on an assessment we conducted together.

[5] I use "appraisal" to mean the most inclusive category for all methods of examining the software process, from the most collaborative and internally performed, like assessment, to the most stringent and externally performed, like an audit.

[6] The CBA-IPI Users' Group was organized in August 1994 by SEI-authorized lead assessors to provide an independent, expert view of developments in the appraisal method and offer lessons learned from practitioners to the SEI.

[7] Capers Jones, "Flaws: Gaps in SEI Programs," *Software Development*, March 1995, 41-48.

[8] See for example Watts S. Humphrey, David H. Kitson and Tim C. Kasse, "The State of Software Engineering Practice: A Preliminary Report," Software Engineering Institute, CMU/SEI-89-TR-1, February 1989. Also, David H. Kitson and Steve Masters, "An Analysis of SEI Software Process Assessment Results: 1987-1991," Software Engineering Institute, CMU/SEI-92-TR-24, July 1992.

[9] See the article by Capers Jones cited above; also Robert L. Glass, "The Software-Research Crisis," IEEE *Software*, Nov. 1994, 42-47.

ISCN 95

August 29th , 1995

Gerhard Rutschek
FESTO GesmbH
Software Development
Lützowgasse 12
A-1141 Vienna, Austria
Phone: +43/1/91075/333
Fax: +43/1/91075/250

Festo - Experience with ESSI Application Experiment ODB

Abstract

This paper describes three years experience with process assessment, process improvement planning and implementation of improvements within an ESSI application experiment at the software centre FESTO Vienna in co-operation with the TU-Graz. The baseline project for this application experiment was the software tool ODB (On-line Data Base). Starting points were the results and the problems which arose in the project VIP (Visualisation of Industrial Process).
The improvements performed resulted in the design of a `Lean Life Cycle and Team Model` for small software engineering teams with less than ten people. This team model is based on the recent concept of `Programming by Contract` describing a defined model in which single modules are given to sub- and subsubcontractors with clearly defined roles, responsibilities, interfaces, acceptance criteria, and contracts.

The following issues will be discussed:

- Introducing the company FESTO and their Software department in Vienna
- Showing up the problems which arose in the project VIP
- Results of a `Bootstrap Assessment` performed in January 1992
- Improvement goals and activities planned and carried out
- Discussion of the `Lean Life Cycle and Team Model` and the concept of `programming by contract`

1. Introduction to Festo Vienna and the SWE Profit Centre

The company FESTO has 50 locations in different countries with about 3250 employees. It's target areas include pneumatic, cybernetic, didactic and tooltechnic.The product range covers everything from individual components such as valves, cylinders and accessories, through to pre-assembled and ready-to-connect modules including customer-specific designs. Of the 800 million DM turnover 1% is used for vocational and further training programmes and 5% for research and development. FESTO is on the way to a world-wide ISO 9001 certification. For example, the FESTO quality assurance system meets the requirements of this standard in all areas from design, development, production and assembly right through to after sales service.

FESTO Vienna has 130 employees. The firm has partners in nearly all eastern countries and co-ordinates the eastern market. The software engineering department named SWE consists of 5 employees and 3 part-time workers, in total there are approx. 10 SW engineers employed. Both hardware and software projects are carried out in this department, among them are some special line solutions, e. g. for banks.

FESTO Austria have extremely close links with neighbouring countries of the former eastern block. The first joint ventures were started in 1982. Today, in addition to numerous agencies, representatives and automation centres, FESTO Austria has wholly-owned subsidiaries in Hungary, Bulgaria, the Czech Republic, Slovakia, Poland, Russia, Romania, the Ukraine, Slovenia and Croatia. An agency office has been established in Tashkent to provide services for Usbekistan. In all these countries, we offer our full range of products and services.

FESTO`s main emphasis is in the field of process automation and one of the major tasks of the software division in Vienna is to develop software to control and give an overview to industrial pro[illegible]. The two process ware tools VIP (Visualization and Control Program) and OPC (On line Data Base) were developed in this division. The software division in Vienna, the "Profit centre" for FESTO Vienna and the eastern market, is providing services in data bases, OOBC, object orientation and distributed systems. It aims to become the competence centre for software engineering.

The principal organisation of the customer (case) project and the CASE tools is as shown in the figure below.

Figure 1: The connections between the CASE tools in the customer project

2. The Starting Point for Process Improvement

The developers of VIP and ODB were hardware experts and had limited knowledge of software models.

VIP (Visualisation of Industrial Processes) specifically deals with the issues of industrial processes and has implemented a system which provides a graphical interface to overview data and processes. The project started in 1988 and has now completed it's life-cycle. The first version was delivered at the end of 1988. Due to major problems, a second version with many modifications was finished in March 1990. Further versions were released in the middle of 1991 and at the end of 1993.

VIP was developed with MODULA2 under MS-DOS and represents a PC-based control and visualisation system. It works in a DOS, NOVELL and desqview environment. At present around 1000 licences have been sold abroad. VIP required 18 man-years and consists of 1 MB runtime-code, with all tools it requires approximately 5-6 MB. VIP also exists in a mini and a didactic version.

During the development of VIP the following problems occurred:

* There was no structured documentation.
  After the introduction of SA (Structured Analysis) no efficient updating of design charts and documents was carried out. Working with a missing structured design resulted in vague definitions and descriptions of the interfaces. The actual documents remained and were infrequently updated.

* Project planning and controlling
  There was no efficient project tracking, so major problems occurred during the out-sourcing. This was mainly due to unclear contracts and functional descriptions.

* Quality system
  There was no visible work process, and a life cycle model was also missing.

* Problem tracking and solving
  There are three categories of errors: functional errors, which can be reproduced, concurrent errors, which are not easy to reproduce, and user errors, which are easy to correct, but need time to be reconstructed (even when the user can`t define how he caused the problem). No error documentation was prepared or problem analysis performed and therefore the monolithical construction of the program caused problems during maintenance.

In the current project ODB (On-line Data Base) a strategy was established to avoid these problems. ODB is a commonly accessible object-oriented datapool for on-line programs and realises a data connection between different operating systems. It was programmed in C++, 4GL languages and Visual Basic. It is an expandable toolbox, which works under WINDOWS-NT, WfW, and Windows95. To increase the compatibility of the toolbox main standard interfaces were employed.

The figure below illustrates the percent amount of changes in the project VIP per year. These numbers are estimates because during VIP there was no automated bug-tracking function.
As you can see, starting with the second delivered version in 1990 nearly 40% of the corrections were done after delivery.

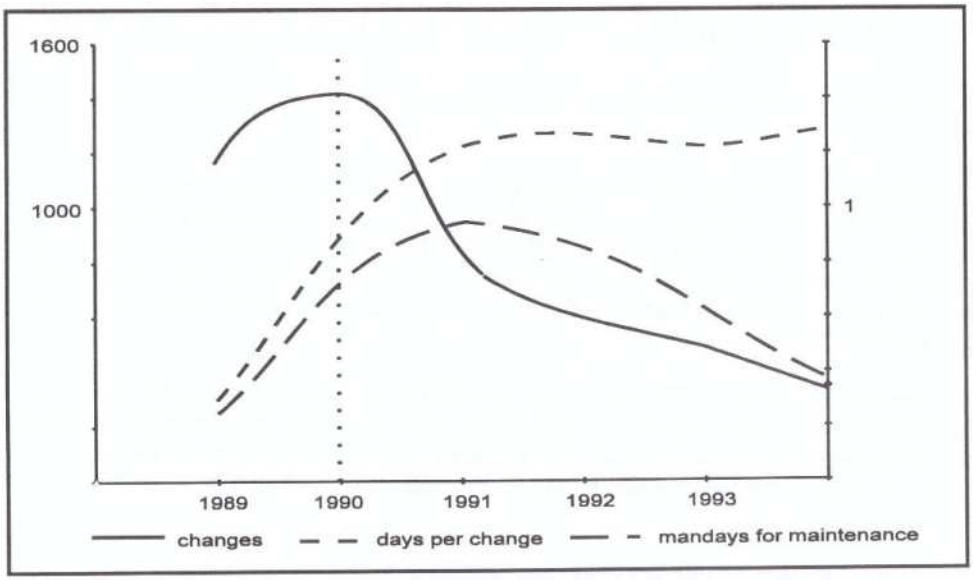


illustration 2 estimated datas about maintenance situation

A big problem was that VIP was implemented as a monolithical program. The description of the internal interfaces was poor and so it was hard to find the module responsible for a bug. In a monolithically built system an adaptation or change based on new requirements is very expensive, you have to step through the whole system. Changing parts or installing new modules is very complicated. In the later versions we designed an overlay system, but again no distribution in separate and independent tasks was done. The system consists of one main part, not split into smaller ones which would communicate together by well defined interfaces. In case of a change the whole system had to be reconciled and delivered. Also from the aspect of re-use it was not possible to separate functions from the system.

In conclusion it was clear that the VIP project had problems with maintenance and re-use. From October 94 till July 95 we measured the time consumption for the maintenance of VIP and the measurement showed that an amount of 29 man weeks was blocked for maintenance.

In January 1992 a Bootstrap Assessment of the software division took place. The strengths and weaknesses of the software division and of the VIP-project were analysed and an Action Plan to improve the software engineering process was established.

Maturity level
The project VIP was on a maturity level 1.75, which means that methods to perform management and software development are being applied. However, some basic methods are missing (methods for analysis and design, methods for project management, methods for quality management).

Comments on the Organisation
+ Commitment of the management
+ Use of control functions
- High dependence on the qualification of individual engineers.
  This will cause problems when one of these engineers leaves the company.

Comments on the Methodology
+ Development plan
  But a negative point is that there are no formal procedures to produce development plans.
+ Configuration control mechanism
+ Acceptance testing, but

- No concept for module and integration testing
- No development model
- No quality plan
- No test plan

Comments on the Technology

+ Use of tools for configuration management
+ Prototyping
+ Developed in-house test data generator
- No tools supporting analysis and design

Recommended Actions for Project VIP :

Organisation

To improve the quality system the software division started looking for international standards, refined the organisational model and produced a quality plan to be followed in all subsequent projects. This should be aim at the visibility of activities, progress and defined procedures for reviews, planning, introducing new technology and training.

Methodology

We have started learning management methods to produce development plans and applied control methods, like design and code reviews, checklists and management reviews. For analysis and documentation, case methods were introduced and the change to a more professional design was executed. In addition new testing methods were implemented, such as test plan, test documentation, test execution and failure reports. This will lead to a better understanding of the complex system, more efficient maintenance and a reduction in the number of errors.

Technology

Although it was recommended the existing management-tool was not reactivated, the use of PERT plans, milestones, resource plan, training plans, control plans was recommended. Without the use of these plans the project management was not effective.

3. Improvement Planning and ESSI Proposal

ESSI Proposal and schedule

The ESSI Project started in May 1993. At this time Austria was not yet a member of the EU. This resulted in 15-20% additional expense, because FESTO needed to create a new starting point for quality protected software engineering in small teams (Lean Life Cycle Model) and to reconstruct the ESSI-project. Tools, SW-licences, training and dissemination were allowable costs.

The results of the Bootstrap Assessment showed the current status of the SWE practice. One of the main conclusions from the assessment was the required status of the SWE practice upon completion of the experiment. Aspired goals were the evolution of a phased plan, good commercial impacts and the transferability of the experience to other projects.

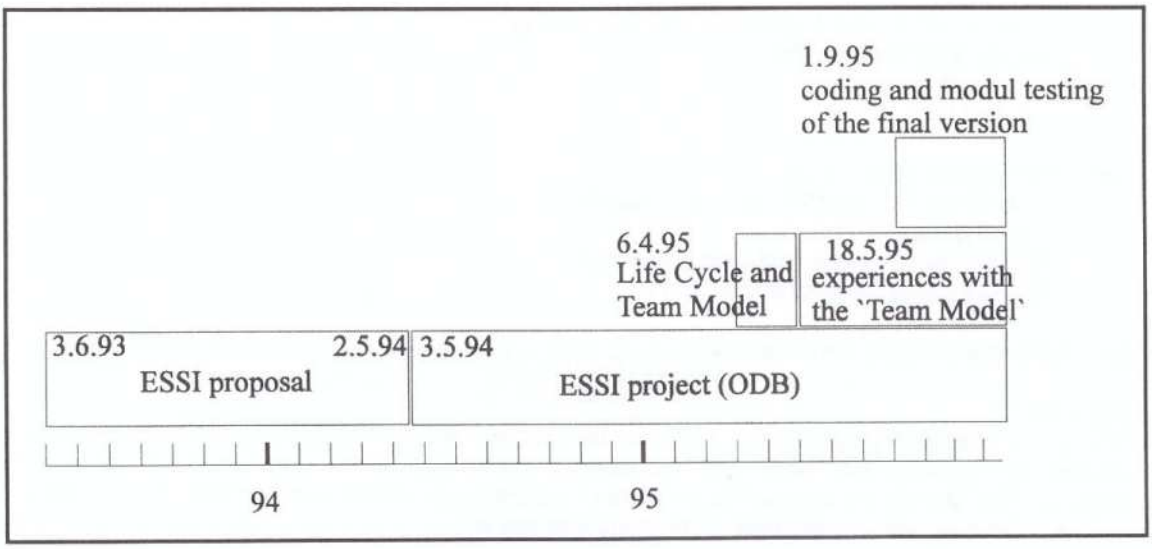


illustration 3 timeschedule of the ESSI-project

Improvement Goals

Working on the experiences of the VIP project we defined two main goals for the ESSI project.

- **Reducing the maintenance costs**
- **Increasing the Re-use**

Improvement Activities

The concept of the new system ODB is completely different from the VIP. It is planned as a toolbox with an open system architecture. To realise this structure we chose a database based system with a client server model. The idea of toolbox means, that the system will never be finished and is open for further enhancements based on standardised interfaces. It will be extended with some new functionality based on well proved functions. The open system also allows the purchase of modules with expert knowledge and include it in the system.

The complete system is realised as a distributed system. Each functionality has it's own EXE-program or DLL with a well defined interface. This concept makes it easier to find out which module is responsible for a bug. By debugging the interfaces between different modules and simulating the interfaces a powerful way for catching real-time-bugs is obtained.

Therefore we hope to reduce the amount of software changes and the cost of such changes. Parallel development in the project is also possible with a concept like this, because each group can emulate the interface from a task which will be developed by another group.

Breaking down the entire system into manageable tasks is also a good way to realise out-sourcing and helps the project management. You can plan small tasks and can out-source these modules easily, as the outface of the module is defined in the conditions of the contract. The interface of the task is a technical part of the contract for an out-sourcing group - programming by contract. The implementation inside the module is not so important, if the programmer follows the documentation, test, and quality assurance rules defined in the ´Life Cycle and Team Model´. The outface of the module is very important, as all the other modules are based on this interface.
In the fast changing world of software development under Windows you have to implement prototyping. In our opinion a good way for re-using the work of prototypes is to split it into small

tasks, because only with a concept like this can you emulate some functionality by, for example, making dummy-DLL's.

The costs of a module will be measured in the future to detect early critical modules and also to measure the quality of the team developing the module.

The ODB is a toolbox for process control systems with an open architecture. It is realised as a distributed system with standard interfaces. The concept is that each functionality is programmed as a separate task with a small and well defined interface to the remainder of the system (object oriented philosophy). This idea follows the strategy of cohesion, coupling and information hiding.

The figure below shows the data flow diagram of the data acquisition part.

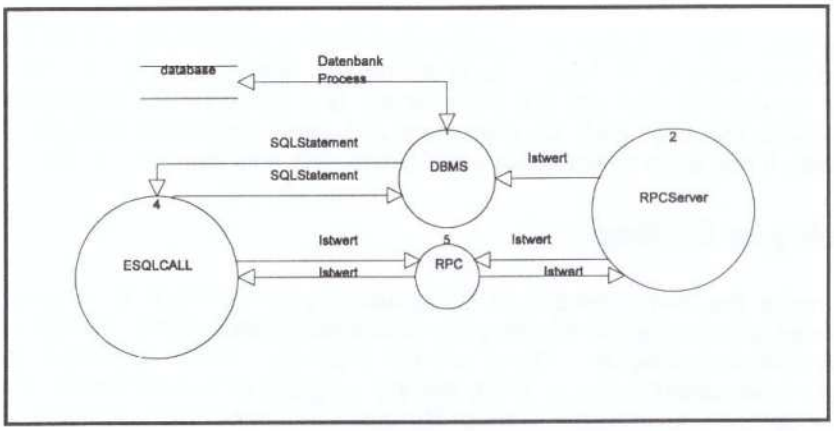


illustration 4 data flow diagram of the data acquisition part

Improvement Achievements

The estimated productivity of the VIP project was about 17 Lines Of Code / man day. Developing the ODB prototype under UNIX, we reached a productivity of 38 LOC / man day not including the documentation requirements of the ´Life Cycle and Team Model´ which were not implemented at this time.

We anticipate a decrease in productivity when we "go live" with the Life Cycle Model to only 25 LOC / man day, and we hope that through re-use the productivity will start to increase again by middle of 1996.

By using structured documentation (defined in the Life Cycle Model) a better base for discussions is provided. The discussions with the customer also became easier. Each member of the team has clearly defined responsibilities for a specified task (i.e. he is his own project manager), therefore parallel development is possible.

Starting with the UNIX-prototype we followed the strategy of exact task and interface definitions. It was possible to save the knowledge for later versions. The WindowsNT prototype has exactly the same interface as the UNIX-prototype. The parallel development of the connection to the database with the integrity of the references was also possible. Later we adapted the database to the SQL Server from Microsoft and we only had to change about 5 % of the code of the connection module - this was a feature of the ODBC interface. In order to speed up the updating to the database, the connection module was only done and so the application did not have to be changed.

The first experiment with an outside developer group (an external company) was then started, using the concept of 'programming by contract' and in the next few weeks we will obtain the first results.

We reduced the maintenance of VIP to nearly 1 man.

In the next few weeks we will install a version control system including a bug tracking system. So we will work on each change request and bug report with this system to make statistics of the quality of each module (= subcontractor). Also members of the team will play the role of a rnodule owner and subcontractor in the future.

4. The Life Cycle and Team Model

The ´Life Cycle and Team Model´ is a guideline containing instructions of how software should be developed and which kind of documents during the development should be provided. The model based on experiences with the ODB-project is a guideline for employees in all future projects. It adapts the ESA PSS 05 guidelines and uses structured analysis and structured design methods. It has been designed for small teams with less than 10 people.

Programming by Contract

As a team model the new approach of ´Programming by Contract´ is employed. A module corresponds with a data-process following the structured analysis (SA) notation and has clearly defined data- and control-inputs. These are the incoming arrows in the SA, and produce a couple of data- and control-outputs, which are the outgoing arrows in the SA. Each module is delegated to a person who is responsible for it's successful completion, and a kind of contract is made: Each module has a clear input- and output-situation and the supplier has only a guarantee for this pre-defined input- and output-situation.

A list of subcontractors (which contractor is responsible for which module) is documented in a management file which will be updated constantly. For the project manager the management file is the basic instrument to supervise the project, he can easily identify who has taken which module and under which contracting conditions. He can also see which contracts have already been completed, which module parts of the system were already generated, tested, taken over and integrated.

Each module owner refines his module with the SA and presents the newly designed modules to the subcontractors. The project manager has to ensure that when the module contracts are being established they will not cause any deadlock situations.

Each module owner tests his own module and documents the testdata used. The acceptance criteria for a module include the documentation of testdata in test protocols. Two kinds of modules can be distinguished: elemental baseline modules, which are the building blocks of the integration, and integrated modules, which consist of baseline modules. During the acceptance of the elemental modules a complete test is required. During the integration stage the complete testing is more difficult, therefore well defined testdata, but not necessarily completed testing is required. All documented testdata should be given to the maintenance team making their task of re-testing easier.
It should be mentioned, that using ´Programming by Contract´ leads to a clearly defined handling unit. An error handling unit, based on an error configuration file, is recommended. In the configuration file new errors are registered with a number and a warning/error classification.

Model Overview

Three mega-processes were designed : project acquisition, realisation and development, and maintenance. Each mega-process is further refined by CASE charts until one clear role with responsibilities is assigned to each work process. The interfaces between the mega-processes have been clearly defined.

The **project acquisition** process comprises: establishing an offer, the review and confirmation of an order, the definition of ´0% to 100% requirements´ in a first version requirements document, the resource estimation, and the official project establishment. The first version of the requirements document, for instance, is taken over and completed in the mega-process. and documents them in a checklist.

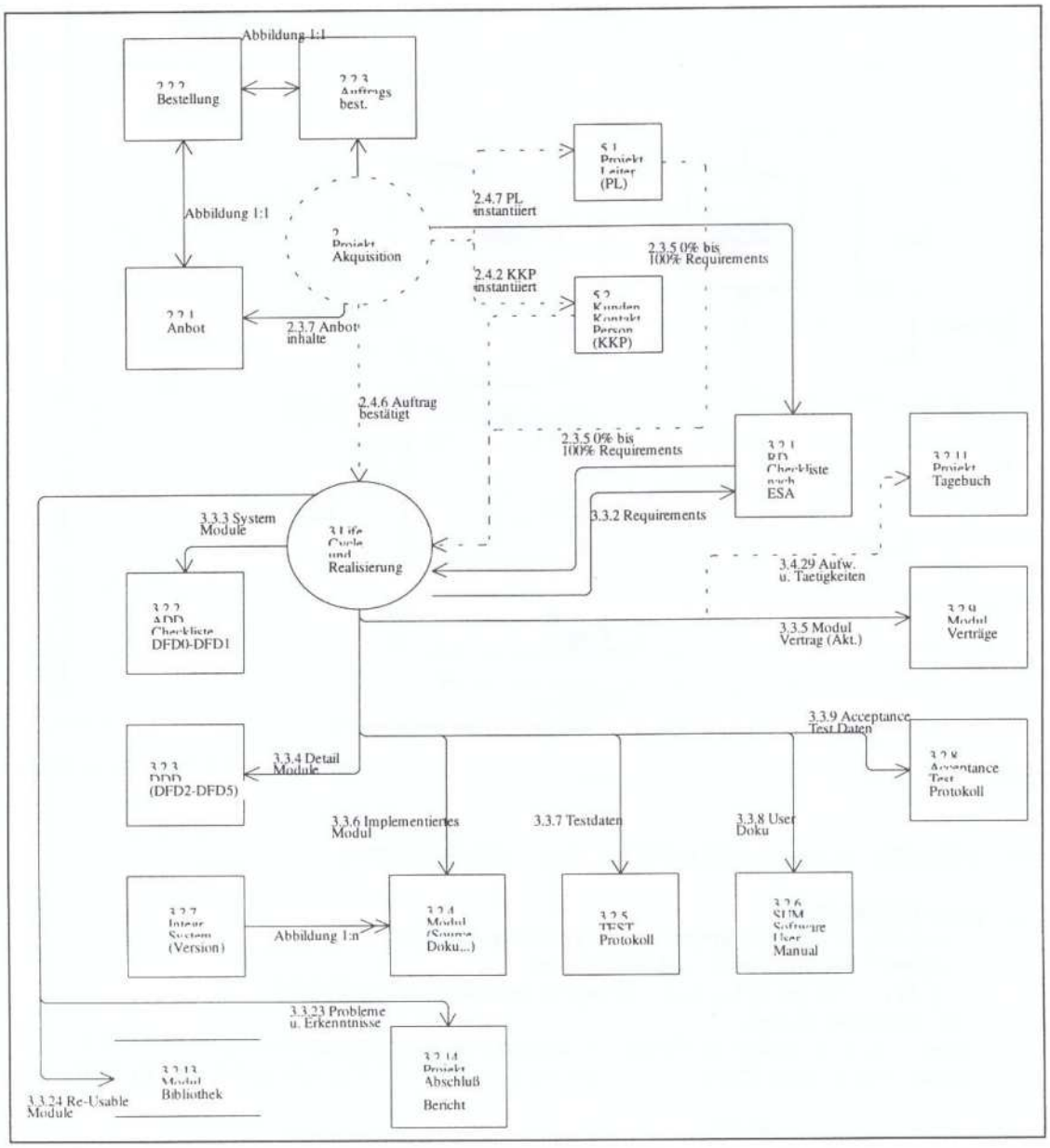


illustration 5 megaphases without maintenance

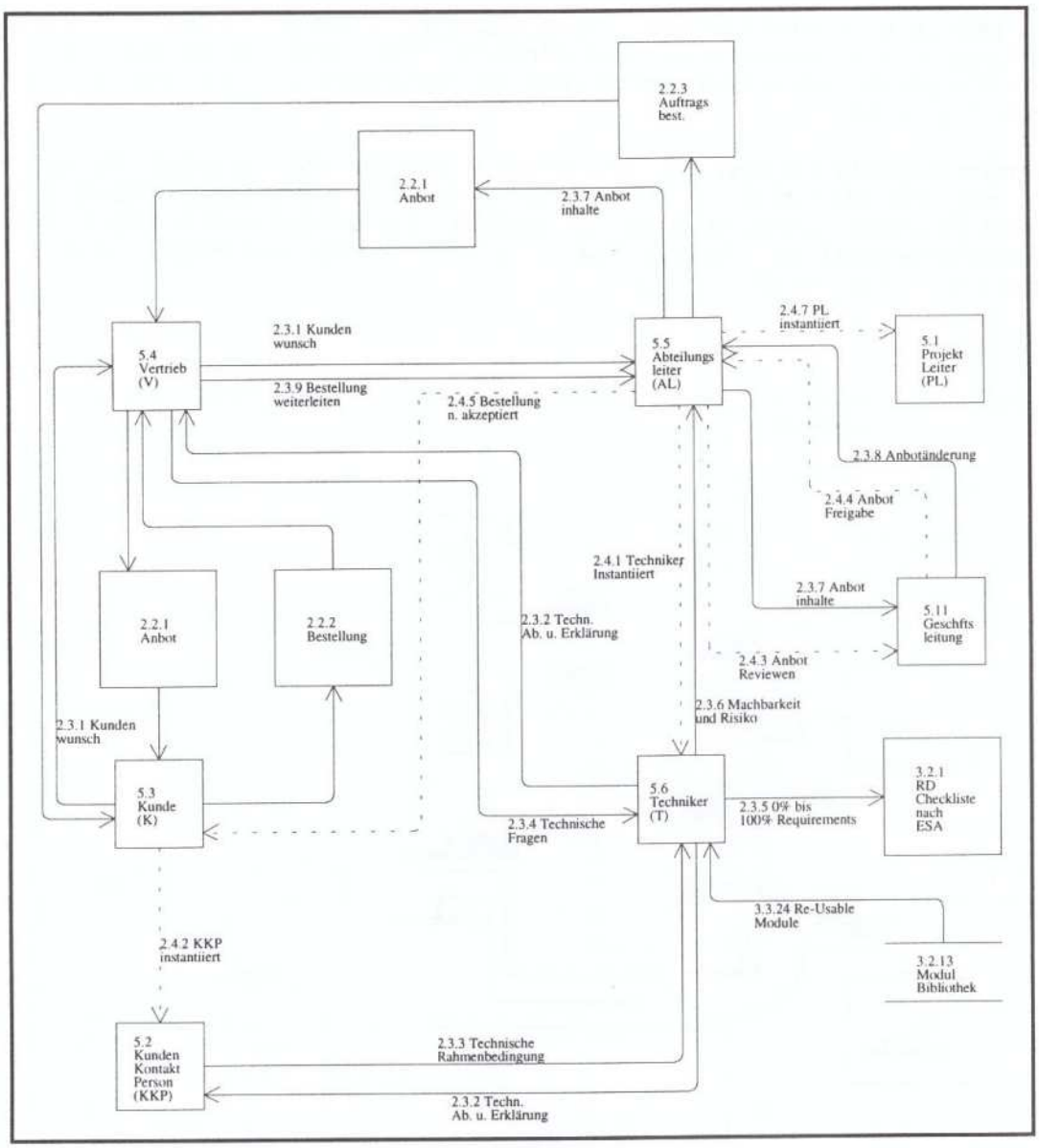


illustration 6 role-play for project acquisition

In the **realisation phase** the following documents are established:

Requirements Document

- Overview of the customer´s requirements and goals
- Technical requirements and restrictions
- The RD is the basis for establishing the contract, review and for the acceptance test.
- description of design verification or project management details are documented separately (review forms, test protocols, standard PERT plans based on workflows)

Architectural Design Document

- Rough general view of the design
- Main functions of the system
- System modules, interfaces, and data flows

- Basic document for the detailed design document
- Description of how the software system is arranged to meet the needs outlined in the requirements document
- Describes the entire software structure, the single software components, interfaces and data required.
- The ADD should be detailed enough in order that a costs- and expense estimation can be extracted, which has a reliability of within 10%.
- In 'Programming by Contract' the description of the single modules within the ADD is the basis for the module contract.

Detailed Design Document
- Completes and refines the ADD
- Detailed description of structures and characteristics of the single ADD-components and various sub and subsubmodules

Software User Manual
- Contains the user information for installation, learning and use of the software product.

Test Protocol

Acceptance Test Protocol
- Detailed description of the software being installed and results of the acceptance test
- All functions requested in ATP were tested (software, hardware, documentation)
- A positive AT confirms the realisation of all requests made in RD.

During realisation a system is developed consisting of a couple of self-sufficient tasks with their own version numbers. The **maintenance** team is provided with all results produced.

The maintenance process uses error reports, change requests, change reports, versions of systems and tasks are defined and documented. Test notes are archived and the user manual adapted. This phase has a lot of interfaces to the configuration management process, because a version survey, i.e. which changes lead from one version to another, and also which version was delivered to which customer, has to be managed.

Accompanied Management Processes

PERT-Plans

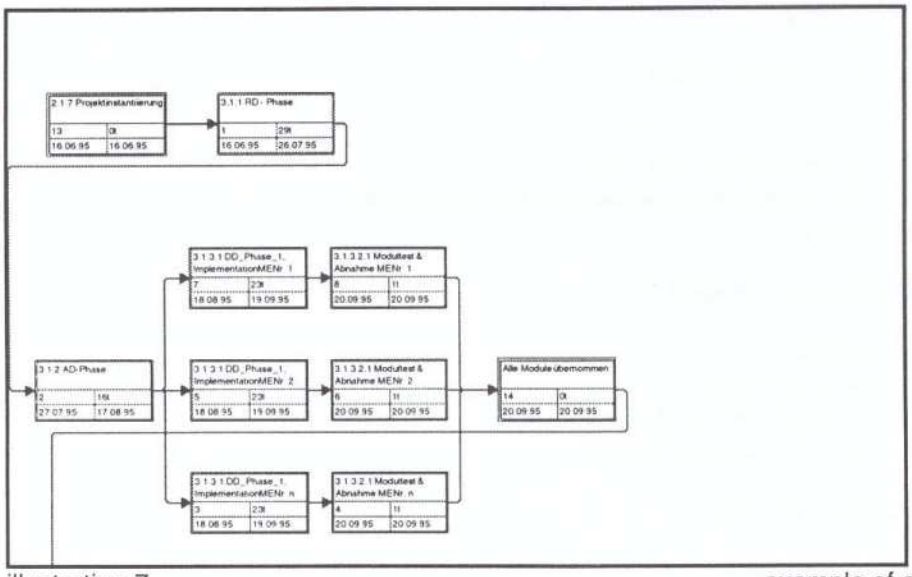


illustration 7 example of a PERT-plan

Datacollection-Plan

The collection of data is necessary for getting answers to the following questions:

- What effort do the modules effect in maintenance?
- In which modules do the most errors appear?
- How big are the modules?
- How critical was the error?
- How far were delivery dates for the modules overrun?
- Which part of a module was constructed by re-use?
- How is the correlation between effort and size of a module calculated?

The project manager writes the project diary, which documents the dates, activities and efforts, based on the completed module contracts. Each module contract contains a start date, a delivery date and approximate effort. After the acceptance the real effort and the real delivery days are written in the project diary.

Each module owner tests his own module and protocols all testdata. In this protocol the effort of testing is noted. The project leader makes an acceptance test protocol, in which the type and number of errors are documented.

Roles

Roles are actors standing behind the processes. They carry out the activities designated to any given process. Below are some examples of roles and their different tasks:

Project Manager
- Planning the project (time, effort, resources)
- Ascertains the requirements together with the client contact, ensuring the requirements made during the project acquisition are refined and completed.
- Designs the context diagrams, the system modules and constructs the first module description
- Analyses errors
- Gives contracts to module owners
- Oversees the project using the module contracts
- Writes the project diary
- Prepares the project conclusion report

Customer Contact
- Clears technical requirements with the client
- Keeps contact with the project manager during the entire project
- Reviews the requirements

Module owner of non elemental module
- Refine the design
- Description of the module
- Separates out elemental modules
- Module description in DDD with CASE
- Gives module contracts to submodule owner

Module owner of elemental module
- Description in DDD
- Implementation of the module

- Tests the module and constructs TP

Reviewer
- Internal RD review
- Internal ADD review

5. Conclusions

Personal Opinion and Conclusions

In October 1994 we made a radical change to the ESSI project. We moved away from a heavy documentation orientation and defined a lean Life Cycle Model for small teams (less than 10 people). This model also defines the roles in the Life Cycle under the aspects of multifunctionality and out-sourcing. The only way to define such a model is to do it with the entire team. Each member gives his experiences and suggestions as an input to the model.

The new open architecture approach also changes the teamwork structure, because each one is now the project manager for his own work. The team motivation is better and will be fixed with the experience phase of the Life Cycle Model. In the case of re-use the efficiency will be increased and the commercial benefits will be better.

Future Plans

When we have finished the ESSI project we will do a Re-Assessment with Bootstrap. At present we are checking the response and requests for this idea and plan to make a Self and Distance Learning Product based on the experience with the Life Cycle Model. We have submitted a LEONARDO proposal covering this area.

For the final report of ESSI we want to make a Measurement and Benefit Analysis. The methodology for this will be the basis for the future. A final presentation will be given at the ESSI Conference in the first quarter of 1996.

Business Decision Problems Supported by Software Product and Process Assessment

Miklós Biró, Éva Feuer, Tibor Remzsô, Piroska Turchányi

Computer and Automation Institute

Hungarian Academy of Sciences

Budapest, Lágymányosi u. 11. H-1111 Hungary

Tel +361 269 8270

Fax +361 269 8269

e-mail: miklos.biro@sztaki.hu

Abstract:

This paper analyses critical business decision problems supported by software product and process assessment which can potentially benefit from a systematic application of multiple criteria decision making theory and techniques. An additional purpose of the paper is to provide a new approach to the topic with a didactic introduction appealing to a business oriented audience as well. The first issue is product quality where standards support purchasing type decisions. The decision problem of an external customer seeking reliable partners is supported by ISO 9000 certification which is performed according to a well defined list of decision criteria. This list is helpful for the decision but does not tell how to proceed in case of failure on one or several of the criteria. The *BOOTSTRAP* software process assessment methodology provides an action plan which supports the management in the detailed choice of the most appropriate steps and decisions depending on the actual standing of the organization.

1. Introduction

Multiple Criteria Decision Making (MCDM) is a classical research field whose objective is the improvement of decision maker performance when there are several alternatives and a variety of criteria for evaluating and comparing them. The subject of this paper is to show how Software Product Evaluation and Software Process Improvement methods can benefit from the systematic application of MCDM theory and techniques.

The fundamental notions of the MCDM field are introduced in the following section with a focus on software quality. What kinds of decisions are supported by

software product quality evaluations based on more or less defined systems of decision criteria? This question is analyzed in the third section. Are there critical decisions which are not supported by software product quality evaluations? Two different types of such critical decisions are given in the fourth and fifth sections. The last section contains a summary and directions for further research.

2. Multiple Criteria Decision Making (MCDM)

MCDM research starts back in the 18th century with Daniel Bernoulli [Bernoulli, 1738]. He argued that a person's utility of wealth is not a linear function of the money measure of his wealth. In our century, it was first von Neumann and Morgenstern [von Neumann, Morgenstern, 1944] who gave a scientific articulation to MCDM. They referred to "several conflicting maximum problems". Since then, due in part to the general successes of the operations research/management science field, the literature of MCDM has exhibited a huge growth.

With a mind trained in the traps of the software development process, we can discover however, that many approaches are strongly biased by the solution methods they offer themselves. An analogous problem is encountered within the software life-cycle, when user requirements specification is not expressly separated from software requirements specification.

There are of course approaches which discipline themselves to focus first on the analysis of the real nature of decision problems. These include [Tversky, 1969] and [Zeleny, 1982]. In this paper, we introduce the MCDM approach following [Zeleny, 1982] but focusing on the concepts relevant to Software Product and Process Assessments.

Multiple criteria decisions involve alternatives which are usually evaluated on the basis of a hierarchical system of criteria. There is a large variety of evaluation process types. The alternatives for example are most of the time to be ranked, but there are cases where only a single alternative has to be accepted.

The starting point of the introduction to MCDM must be the clarification of the fundamental concepts which are *attributes, objectives, goals, criteria, strategies and alternatives.* The subject of the following sections is essentially the discussion of the high relevance of these notions to software quality and process assessment. These terms are typically used however not defined in the otherwise meticulous standards.

Attributes are the descriptors of reality. They can be identified and measured essentially independently from the decision maker's or user's requirements. *Objectives* consist of individual attributes or sets of attributes together with their directions of preference. They are clearly identifiable with the decision maker's or user's requirements. Two or more objectives can form higher level objectives resulting in a hierarchy which helps overcoming the cognitive psychological barriers in case of more than 7±2 simultaneous objectives [Biró, Maros, 1992]. *Goals* mean target levels of achievement defined in terms of either attributes or objectives. They are determined by the decision maker's or user's requirements. *Criteria* is a general term for the attributes, objectives, or goals which are judged relevant in a given *decision situation* by an *individual* or a *group*.

A quality standard can be considered as the result of a group consensus on a system of relevant quality criteria.

Strategies are the means of changing the current situation. *Alternatives* are strategies which are candidates to be employed for changing the current situation according to a current system of citeria. Alternatives are usually expected to be mutually exclusive. Nevertheless, if the decision situation allows for the simultaneous selection of several alternatives, then it is enough to state that there is at least one pair of alternatives which are mutually exclusive, otherwise there is no real decision to be made.

3. Product Quality Evaluation

The evaluation of the quality of a *software product* is one of the basic issues in information technology. Many books [Gilb, 1988] [Gillies, 1992] [Boehm, 1981] and papers deal with this subject, suggesting MCDM approaches to a smaller or larger extent. A system of decision criteria is summarized in the ISO/IEC 9126 standard. The first level of the criterion hierarchy contains the following elements: *functionality, reliability, usability, maintainability, portability,* and *efficiency*. This standard should be considered as the result of a group consensus on a system of relevant criteria. Nevertheless, there is no consensus for the moment about subcharacteristics for example, which are only provided as an illustrative annex to the standard. In MCDM terms, the ISO/IEC 9126 standard defines high level objectives only without specifying the corresponding lower level measurable attributes themselves, thus currently, it fails to satisfy its fundamental purpose of assisting consistent software

product quality evaluation. An approach to filling this gap is provided by the SCOPE guide for software product quality evaluation [Boegh, 1994].

The decisions that ISO/IEC 9126 intends to support are the following:

- Does the software requirements specification adequately reflect the user requirements?
- Does the developed software satisfy the user requirements?

From the MCDM point of view, the alternatives are software products which can be evaluated and ranked by calculating the weighted average of their scores with respect to the different criteria. The weighted average is also called *status* of the alternative. The weights and value scales should be carefully determined so that the status represents a true characterization of the alternatives on the basis of the hierarchical system of criteria.

Alternatives	*product*1	Result: status related to the ISO/IEC 9126 standard
	*product*2	
	• .	
	*product*n	
Criteria	*functionality*	Evaluation: scaling, weights
	reliability	
	usability	
	maintainability	
	portability,	
	efficiency	

There are fundamental business decisions which are not supported by ISO/IEC 9126 or other systems of product quality criteria, and which are in the focus of the following sections.

4. Decisions Depending on Sustainable Reliability

The fact that a company was able to eliminate defects from some of its products does not mean that those same defects will not reoccur in future software products.

A fundamental business issue is by consequent *the customer's decision problem:*

- Is the supplier able to sustain the reliability of its production?

ISO-9000 certification is intended to support the above decision by focusing on the process rather than on the product. The standard generally applied to the software development process is ISO 9001 which is complemented by the ISO 9000-3 notes for guidance on the application of ISO 9001 to software development.

ISO 9001 lists 20 quality system requirements which can be considered as a set of high level criteria for the above decision problem.

These criteria have more or less detailed subcriteria spelled out in the standard's subparagraphs on multiple levels. Thus, we are again faced with a hierarchical system of criteria. Here the lowest level attributes are mostly measurable in binary terms, that is only the existence of required policies and practices has to be checked.

ISO 9000-3, formulated in the form of guidelines for the application of ISO 9001 to the development, supply and maintenance of software, is essentially a restructuring and reformulating of the system of criteria from top to bottom but finally covering all of the ISO 9001 requirements.

	ISO 9001	**ISO 9000-3**
Alternatives	Certification (Yes or No)	Certification (Yes or No)
Criteria	• Tree structure • 20 elements (high level criteria) • Multiple subparagraphs (lower level criteria)	Same as ISO 9001 in a different structure.

5. Decisions Related to Software Process Improvement

Certification provides little support for improvement. The system of requirements is rather large and their simultaneous satisfaction may mean an inhibitive burden to the company. In addition, conformance must be maintained after registration as well, which can be best achieved by implementing continuous improvement. By consequent, the natural question before or after certification is how a company can improve its capability for reliably producing high quality products. This is *the supplier's decision problem:*

- How can we improve the reliability of the production ?

While the customer's decision problem discussed in the preceding section involved only two alternatives (yes or no), this decision problem admits a large number of

alternative courses of action the choice among which is supported by the *BOOTSTRAP* method for software process assessment and improvement.

BOOTSTRAP was a European ESPRIT project (5441) which was carried out and finished in February 1993 by a consortium of European software companies and universities. The aim of this project was to develop a method for software process assessment and improvement adapted to the European software industry. *BOOTSTRAP* designed a very detailed process quality attribute hierarchy and enhanced the questionnaire based on the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) [Paulk, Curtis, Chrissis, 1991,1993] by taking into account the ISO 9000-3 guidelines for software quality and the ESA (European Space Agency) PSS-05 software engineering standards. In addition, the questions are answered on a four choice percentage scale instead of the yes-no choices of the SEI method.

Contrary to the yes-no conclusions of ISO 9000, CMM provided an ordinal scale for measuring process maturity, which made it possible to move further than Software Capability Evaluation (SCE), to the possibility of prioritizing improvement efforts.

In CMM, the maturity levels are decomposed into *key process areas* which are targeted by specific yes-no questions during capability evaluation. From the decision making perspective, the key process areas are *high level criteria* which lead to the decision to be made about the maturity level of the evaluated software producing unit as a whole. The *decision alternatives* in this case are the *maturity levels* themselves.

SCE

Alternatives	Five maturity levels	Result: maturity level status (integer)
Criteria	Key process areas	Evaluation: binary

As pointed out in [Haase, Messnarz, Koch, Kugler, Degrinis, 1994], a CMM key process area is only valid within one maturity level, thus, it cannot be used for mapping separate attributes onto a maturity level scale. *BOOTSTRAP*, on the other hand, results in a profile containing the maturity-levels of the individual attributes.

***BOOTSTRAP* for process assessment**

Alternatives	Profiles with five possible maturity levels for each attribute	Result: refined maturity level status (the overall status between two consecutive maturity levels, and the maturity levels of process attributes)
Criteria	Detailed hierarchy of process-quality attributes	Evaluation: refined scaling and evaluation algorithm

When, instead of evaluation, the decision problem is improvement related, a *strategy* must be worked out. The *key process areas* or *BOOTSTRAP* attributes become decision alternatives as the potential components of the improvement action plan: which key process areas or which attribute(s) should be improved in order to achieve the goal, for example maturity level 4. In this case, the number of alternatives is considerable.

***BOOTSTRAP* for process improvement**

Alternatives	Process-quality attributes to include into the action plan	Result: Action plan
Criteria	Maturity levels of process-quality attributes	Evaluation: Trained *BOOTSTRAP* assessors

Having based the work on the ISO 9000-3 standard, *BOOTSTRAP* can determine about 85% of the ISO attributes as well whether they are satisfied or not [Biró, Feuer, Haase, Koch, Kugler, Messnarz, Remzsô, 1993], [Haase, Messnarz, Koch, Kugler, Degrinis, 1994]. Therefore *BOOTSTRAP* can also be used as a decision support tool for undertaking ISO certification.

***BOOTSTRAP* for ISO certification**

Alternatives	satisfied or not	Result: satisfied or not
Criteria	85% of ISO 9000-3 attributes	Evaluation: mapping algorithm

6. Conclusion and directions for further research

We analyzed critical software related business decision problems. Different types of criteria and alternatives were recognized in each case. Some of the criteria are standardized, like ISO/IEC 9126 and ISO 9001. Methods rank from simple binary evaluation to multiple level weighted average. The table below summarizes the decision criteria and alternatives in case of various approaches.

Summary table	Alternatives	Criteria
ISO 9000	yes or no	20 elements
SCE	Global maturity level	Key process areas
SEI Improvement	Key process areas to improve	Global maturity level
BOOTSTRAP assessment	Profiles with five possible maturity levels for each attribute	Detailed hierarchy of process-quality attributes
BOOTSTRAP improvement	Improvement action plan	Profiles with five possible maturity levels for each attribute

In the nineties, a new trend started in the field of MCDM. Instead of providing an exact but strict evaluation procedure, flexible learning tools based on visual interactive technology have been developed, for example [Angehrn, 1991], [Biró, Csáki, Vermes, 1991] and [Csáki, 1995]. The large variety of MCDM problems justifies the development of humanized multiple criteria decision systems where the decision makers are allowed to learn about the decision problem itself while experimenting with both its formulation and solution. These kinds of systems merely put different tools and instruments at the decision makers' disposal. The user is allowed to define the set of alternatives, the set of criteria, the evaluation method. He can trace the effect of any change in one problem component to any other problem component. Consequently, with the help of such a flexible system for the case of process improvement, the decision maker could determine its preferred improvement action plan in a most creative and challenging way.

BOOTSTRAP can be considered as a special MCDM method for comparing and evaluating alternatives on the basis of hierarchically structured criteria. It would be highly attractive to provide a flexible working environment in which the direct or indirect effect of any changes to problem components could be automatically

monitored. Such a feature would certainly increase the interest and commitment of the participants in the assessment and improvement process.

References

[Angehrn, 1991] Angehrn, A.: Designing Humanized Systems, Human Systems Management 10 (1991) pp. 221-231.

[Bernoulli, 1738] Bernoulli,D. Specimen Theoriae Novae de Mensura Sortis. Comment.Acad.Sci.Imper.Petropolitanae, 5(1738)175-192. English translation by L.Sommer, Exposition of a New Theory on the Measurement of Risk. Econometrica, 22(1954)23-36.

[Biró, Csáki, Vermes, 1991] Biró,M.; Csáki,P.; Vermes,M. WINGDSS Group Decision Support System under MS-Windows. In: Proceedings of the Second Conference on Artificial Intelligence (ed. by I.Fekete and P.Koch). (John von Neumann Society for Computer Sciences, Budapest, Hungary, 1991) pp.263-274.

[Biró, Maros, 1992] Biró,M.; Maros,I. The Use of Deep Knowledge from the Perspectives of Cooperative Problem Solving, Systems Modeling, and Cognitive Psychology. In: Shifting Paradigms in Software Engineering (ed. R. Mittermeir). (Springer-Verlag, Wien, New York, 1992) pp.56-67.

[Biró, Feuer, Haase, Koch, Kugler, Messnarz, Remzsô, 1993] Biró,M.; Feuer,É.; Haase,V.; Koch,G.R.; Kugler,H.J.; Messnarz,R.; Remzsô,T. *BOOTSTRAP* and ISCN a current look at the European Software Quality Network. In: The Challenge of Networking: Connecting Equipment, Humans, Institutions (ed. by D. Sima, G. Haring). (R.Oldenbourg, Wien, München, 1993) pp.97-106.

[Boegh, 1994] Boegh, J.: SCOPE: A guide for Software Product Quality Evaluation, Proceedings of the ISCN '94 Conference on Practical Improvement of Software Processes and Products, (Dublin, Ireland, 1994).

[Boehm, 1981] Boehm, B.: Software Engineering Economics. Prentice Hall, Englewood Cliffs, NJ 1981

[Csáki, 1995] Csáki, P., Csiszár, L., Fölsz, F., Keller, K., Mészáros, Cs., Rapcsák, T., Turchányi, P.: A flexible framework for group decision support, Annals of Operations Research, Special Volume on APMOD, *accepted*

[Gilb, 1988] Gilb, T.: Principles of Software Engineering Management, Addision-Wesley 1988.

[Gillies, 1992] Gillies, A.C.: Software Quality Theory and Management, Chapman & Hall, 1992

[Haase, Messnarz, Koch, Kugler, Degrinis, 1994] Haase,V.; Messnarz,R.; Koch,G.; Kugler,H.J.; Degrinis,P. Bootstrap: Fine-Tuning Process Assessment. IEEE Software (July 1994) pp.25-35.

[Paulk, Curtis, Chrissis, 1991] Paulk, M.C., Curtis, C. and Chrissis, M.B.: Capability Maturity Model for Software, Software Engineeering Institute, Carnegie Mellon University, Pittsburgh, August 1991.

[Paulk, Curtis, Chrissis, 1993] Paulk, M.C., Curtis, C., Chrissis, M.B.and Weber,C.V.: Capability Maturity Model, Version 1.1, IEEE Software, July 1993, pp.18-27.

[Tversky, 1969] Tversky,A. Intransivity of Preferences. Psychological Review, 76(1969)31-48.

[von Neumann, Morgenstern, 1944] von Neumann,J. and Morgenstern,O. Theory of Games and Economic Behavior. Princeton University Press, Princeton, NJ. 1944.

[Zeleny, 1982] Zeleny,M. Multiple Criteria Decision Making. McGraw-Hill Book Company, 1982.

July 21st, 1995
ISCN'95: Measurement and Training Based Process Improvement
September 11 - 12, Vienna, Austria

Thomas Mehner, Axel Völker
Siemens AG
Corporate Research and Development
Application Center Software
ZFE T ACS
Otto-Hahn-Ring 6
81739 Munich, Germany
Phone: ++49/89/636-47253
Fax: -44424
e-mail: Thomas.Mehner@zfe.siemens.de

Siemens Process Improvement Approach

Abstract

The Siemens Software Process Assessment Approach has been presented in detail at the ISCN'94 [6]. It is based on the Capability Maturity Model (CMM) of the Software Engineering Institute (SEI) [3]. Extensions have been made according to the BOOTSTRAP results and Siemens specific requirements targetting on Process Improvement in industry. The focus of those assessments is not ranking, but detailled identification of strengthes and weaknesses. Typically 60 to 120 recommendations are derived, clustered and prioritized to provide an adequate basis for an improvement program for specific business units.

The focus of this paper is on the Improvement Program OPAL, which is part of the strategic top-Program of Siemens AG.

The following topics will be disccussed:

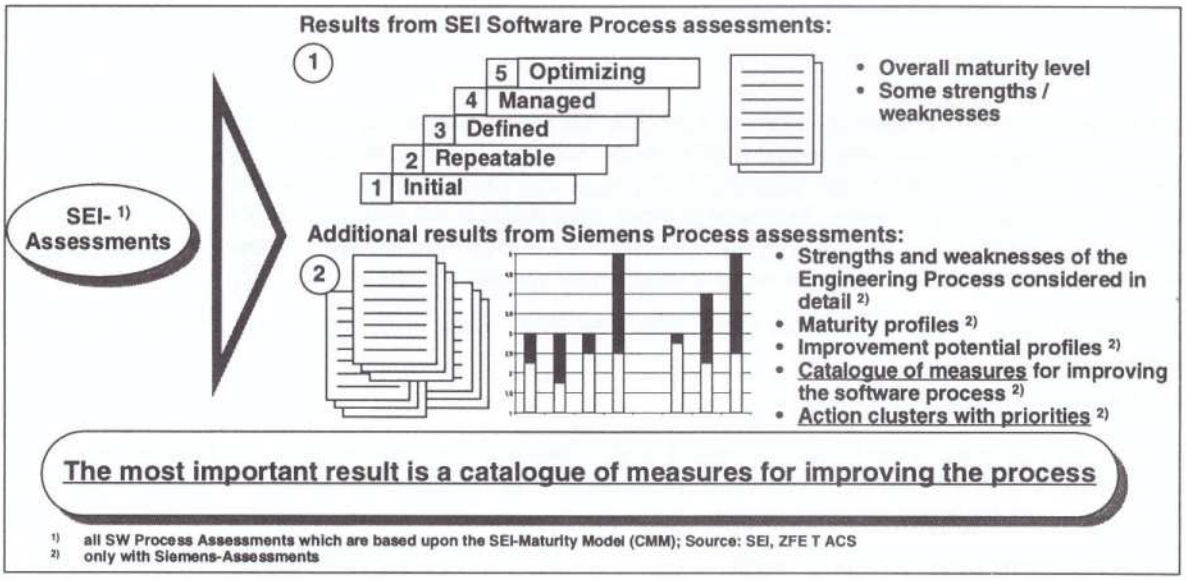
- Contents and goals of the OPAL program. For all Siemens divisions where SW is a relevant factor, an assessment with a following improvement program shall put the maturity level for SW-development on a sound basis for industrial development.
- Enhancements in the assessment methodology since ISCN'94. New application fields can be covered by facultative add on questions (System Engineering, Configuration Engineering, reuse). In addition to the assessment method described last year, a shorter process was defined which is more convenient for smaller organisations of up to 50 SW-developers.
- The phase of starting an improvement program begins with the final presentation of the assessment results. Procedure and the various possibilities for participation of the Application Center Software will be described.
- Experiences with improvement programs within Siemens AG in those programs are the final point of discussion.

1 Introduction

The Application Center Software is part of the Siemens Corporate Research and Development. Our main task is the enhancement of the quality and productivity of SW-and Systems Engineering within Siemens divisions and Siemens Operating Companies world wide. The focus is on the underlying processes and the quality management. Assessments are used for a first evaluation of the status of an organisation.

The Siemens Software Process Assessment has been presented in detail at the ISCN'94 [6]. It is based on the SEI Capability Maturity Model (CMM) and the BOOTSTRAP results. Furthermore ISO 9000- and special company-aspects are considered [1,2,3,4,5]. In this assessment an overall maturity level is computed for reasons of international compareability and for setting global goals. The main effort is spent on

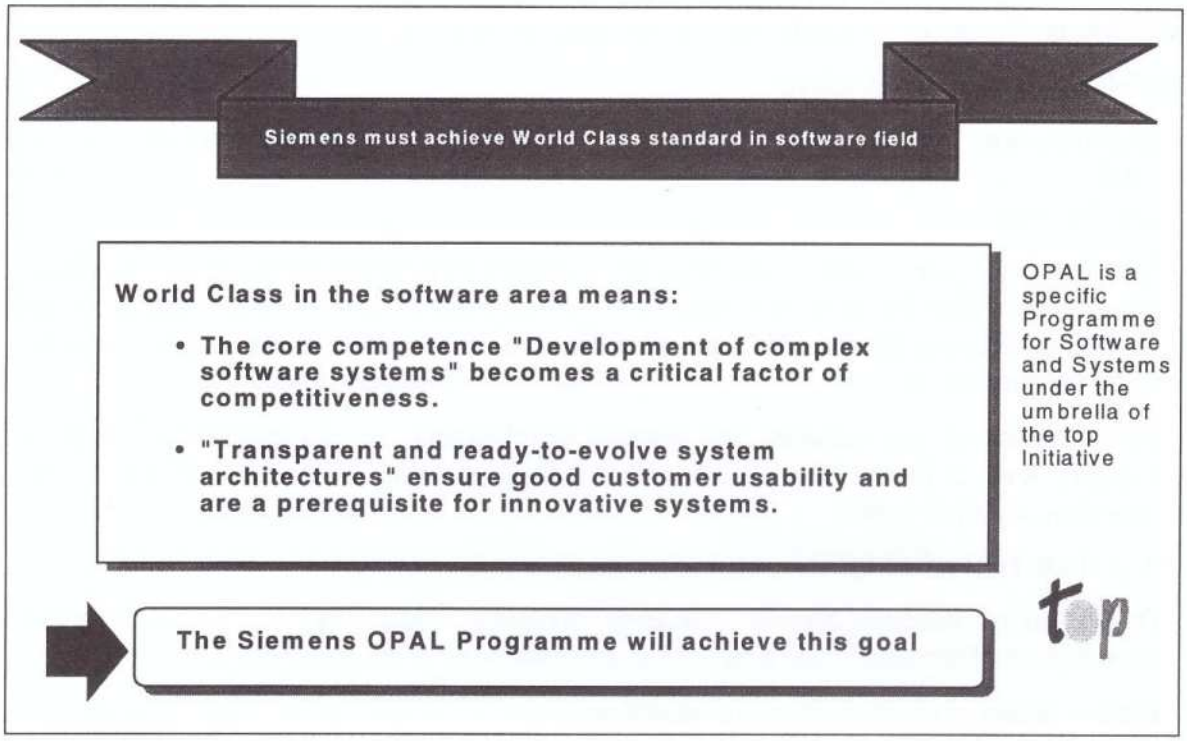
- detailed profiles and descriptions of strengthes and weaknesses throughout 25 process related areas,
- the derivation of recommendations for improvement actions,
- the further processing of recommendations (clustering, prioritising, classification, timely structuring) to generate a stable basis for an improvement program.



Picture 1: Comparison of SEI and Siemens Assessments

The focus of this paper is on the Improvement Program OPAL, which is part of the strategic top-Program of Siemens AG.

2 Contents and Goals of OPAL



Picture 2: OPAL Program

In 1994 the Siemens AG started the "top-Initiative" (time optimised processes"). for improving the competitiveness of all divisions of the company. The goal is to reach top positions on the global market.

The general impact of software at Siemens is steadily increasing. A considerable proportion of the returns are now software related. Software has strategic importance within a number of divisions. In some business areas there has been a shift from electromechanics towards electronics and thus towards software. Efforts for development and maintenance require billions of german marks. Some products require up to several thousand software developers. Development is often distributed over different sites and countries. Many systems are long lasting and evolutionary. Requirements w.r.t. quality, development time and cost are more and more stringent.

The situation described above led to the creation of the strategic OPAL-program which runs under the umbrella of the top-Initiative. The name „Optimise process´ and architecture´s leadership" describes the two focus points of the program:

- Assessment and Improvement of system architectures (this aspect is not covered here)
- Assessment and Improvement of System Engineering and under lying processes

For all Siemens divisions where SW is a relevant factor, an assessment with a following improvement program shall put the maturity level for SW-development on a sound basis for industrial development.

The main goal of OPAL is the fast and cost efficient development of good quality software. The focus is on software not only as a standalone product but by increasing impact as components of systems or development instrument for systems.

The following approach is taken:

- Determination of strategic importance of software in all key business areas. Analysis of competitors with regard to efficiency improvements in software development.
- Execution of a process assessment in all key business areas where software is of strategic importance within 18 months. Classification of engineering processes according to the Capability Maturity Model (SEI) and identification of the potential for improvement by using the Siemens-uniform assessment methods for all Siemens Groups.
- Industrialising of the software and systems development, i.e. achievement of at least maturity level 3 for all key business areas, where software is strategically extremely important within 3 years.

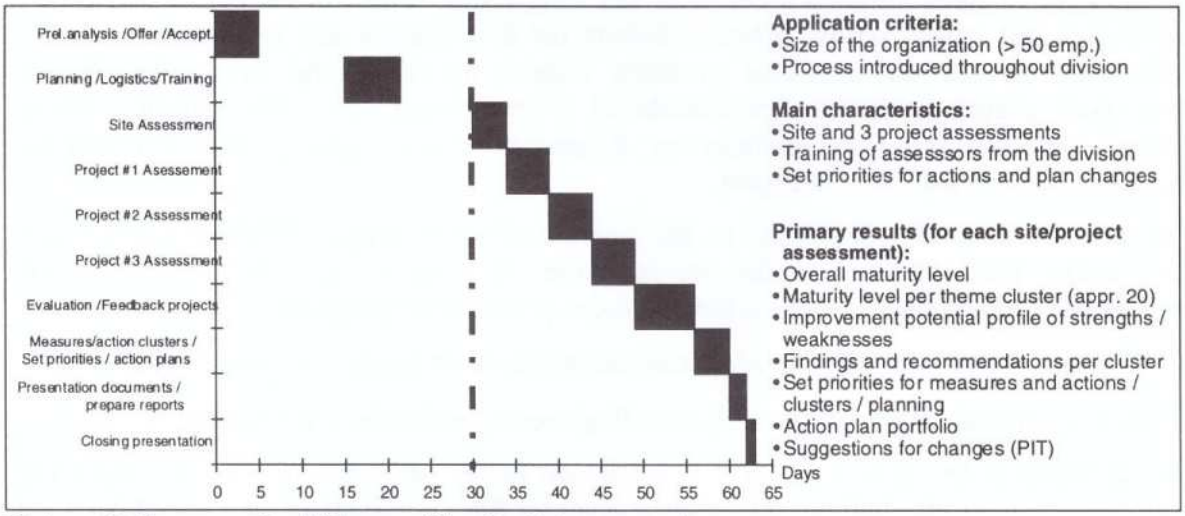
This goal can be reached by:

- Derivation of business specific efficiency programs which lead to the wide-spread introduction of measures based on the recommendations of the assessment.
- Establishment of continuous improvement processes in all key business areas relevant for software development.

Both activities can be supported by the Application Center Software.

3 Enhancements to the Siemens Assessment Methodology

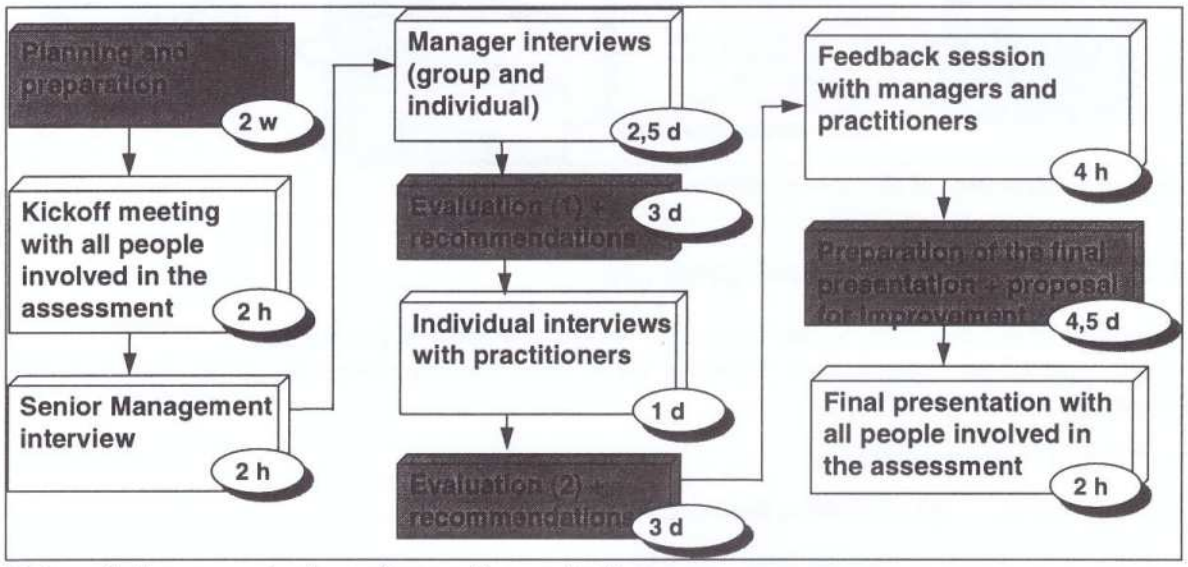
Since its presentation at ISCN'94 the Siemens Assessment Methodology has been enhanced to make it appropriate for all application fields within Siemens.



Picture 3: Framework of Siemens Standard Assessment

For small organisations with up to 50 software developers a different scheme had to be defined. In comparison to the method described last year the following changes have been introduced and successfully applied:

- The separate assessment of site and projects was given up. Our experience is that in smaller organisations the difference of defined and documented process and procedures is smaller and the persons responsible are not strictly different. For this reason the site and project aspects are treated together in the assessment interviews and in the evaluation.
- The separate assessment of different projects (two to four) is done for large organisations to get a representative picture. For smaller organisations members from different projects can be represented in the interview groups and the information can be gathered in one assessment.
- The participation of interview partners in the assessment was reduced to about half the amount.
- Time needed for the complete assessment from kick-off to final presentation was reduced to four weeks (from about 6 weeks for a standard assessment).



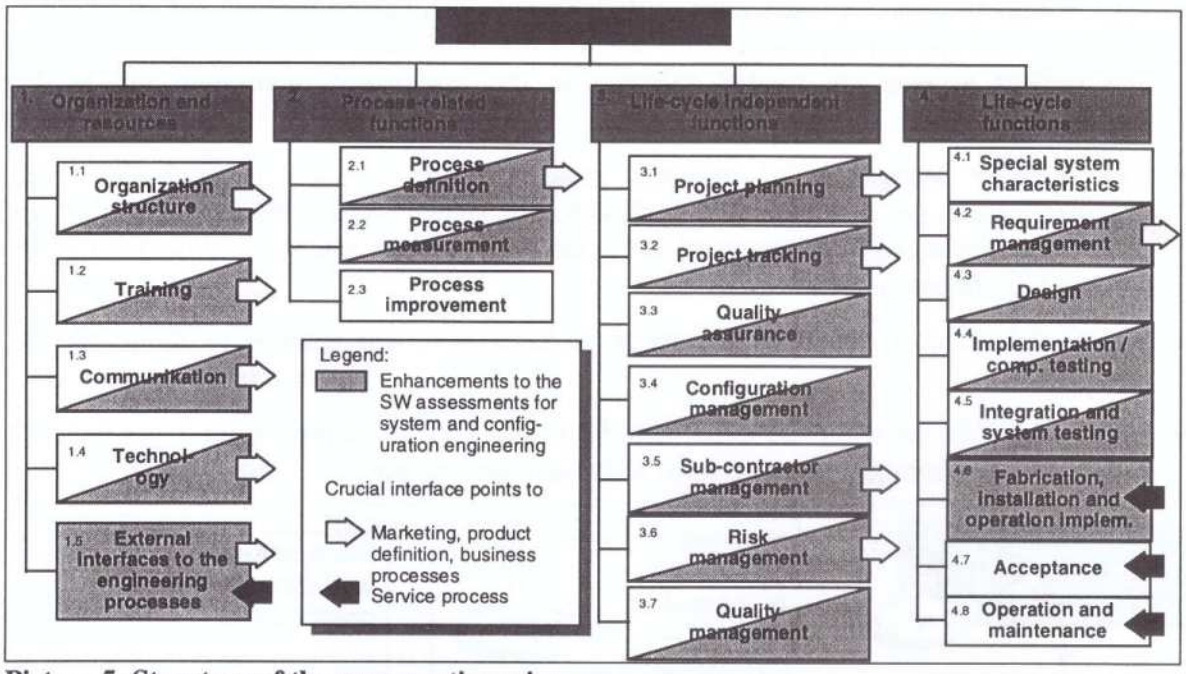
Picture 4: Assessment scheme for small organisations

We did not change the involvement of levels of management and staff in the interviews. For each organisation which wants to be assessed, the two forms of assessment are presented in an executive briefing and according to the seize and homogeneity the decision for one of the two assessment schemes is struck.

In many assessments we made the experience, that some important parts of the business process were not sufficiently covered by our questionnaire respectively the underlying CMM [3]. So major extensions had to be introduced:

- The interfaces at the beginning and at the end of the development process had to be included in more detail: marketing, product definition and service.
- Most division's business is in systems engineering up to plant engineering respectively "streamlines" of versions and variants of products/systems are delivered. For this reason facultative add on questions had to be introduced for many theme clusters.

Picture 5 the structure of our questionnaire. White boxes are unchanged compared to the initial software engineering version. Grey boxes are completely new and the others have been supplemented.



Picture 5: Structure of the new questionnaire

4 Starting an improvement program

The results of an assessment are the main input for the improvement program:
A catalogue of recommendations (typically 60 to 120) lists all recommendations. For each the following attributes are added:

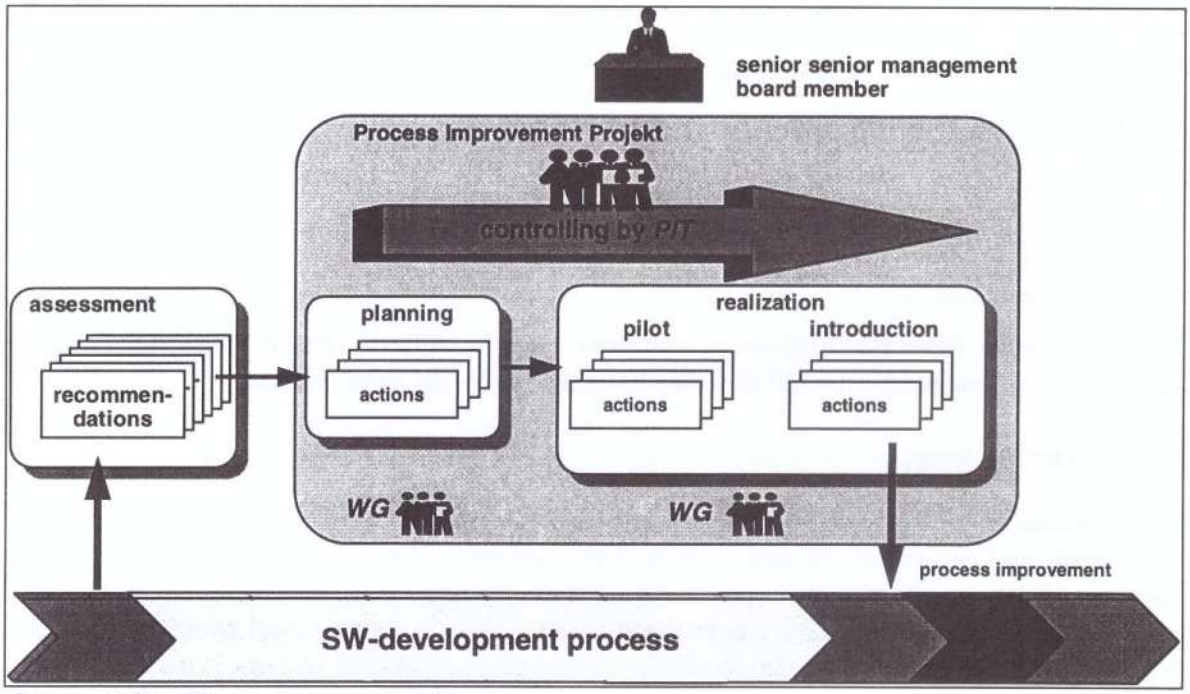
- Priority (three degrees);
- effort needed (rough estimation, three degrees);
- time frame (rough estimation, three degrees);
- action cluster (assignment of single recommendations to functional topics, which can be treated by a work group, the action clusters are prioritised in a portfolio matrix);
- special criteria (depends on the assessed organisation; e.g. relevance for ISO 9000, scope of recommendation (single group, whole organisation));

This is a sound basis for developing an improvement project. As a management input the goals of the organisation and the effort available must be given. With this information the recommendations can be selected and sequenced, so that the highest benefits (impact on improvement) can be expected for a given effort. Actions can be planed and responsible persons (e.g. work groups) can be assigned. The realisation can be done on a pilot basis for actions with complex implications or on broad basis when solutions are quite clear.

If an improvement program shall be successful, it must be a project. This means:

- A project organisation exists (e.g. Process Improvement Team and work groups, responsibilities, budget, effort and time assigned);

- a goal oriented planning for the realisation of improvement actions exists;
- realisation is tracked by PIT and senior management;
- the results are measured by a metric system which has to be defined in accordance with the goals of the program and specific requirements of the organisation.



Picture 6: Starting an improvement program

There are three options for the participation of the Application Center Software (ACS) in business specific improvement program,:

1. Especially large organisations with sufficient qualified resources are able to plan and implement a program without external help. In this case a deadline for reaching substantial improvement goals is fixed and a reassessment takes place after this time.
2. For other organisations there is a lack of knowledge about special topics, where weaknesses have been identified in the assessment. In this case the ACS has standard concepts available for some topics which can be adopted to organisation specific requirements and can be introduced quite fast (configuration management, formal inspections, risk management, QA for external deliveries, defect prevention, project planing). This also includes training and coaching. For other topics co-operation with technology labs of our central research division is available.
3. For smaller organisations the availability of resources and know how for an improvement program can be a problem. In this case the whole program is planned and managed by ACS.

5 Experiences with OPAL

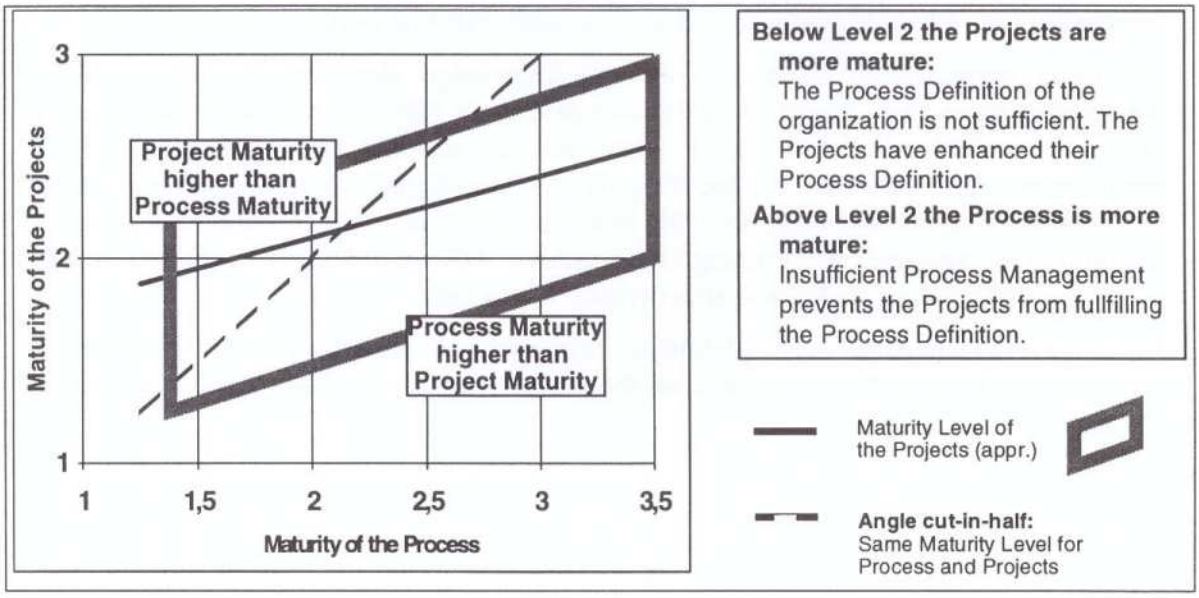
Up to now more than 20 assessments (covering about 50 projects) have been finished. For most of them an improvement program started within reasonable time (which should not be longer than 2 months!). We covered the following branches (Siemens divisions or associated companies):

- Telecommunications (switiching, mobile networks) with very large systems;
- power supply and power distribution;
- traffic systems with very high security requirements;
- automation;
- medical systems;
- automobile technics.

The assessments have taken place at company sites in Austria, Brasil (planed), France, Germany, Italy, Switzerland and USA. We have the following experiences:

5.1 Maturity level

By our standard assessment we examine the defined process valid for organisations (site assessment) and separately the practices of 2 to 5 projects (project assessment). In case of processes of organisations below level 2, projects are in most cases more mature rather than processes (site assessment). This means the globally defined process is not adequate and the projects have to add individual and local regulations and procedures to cope with their basic needs.
In case of processes of organisations above level 2 the defined process is generally more mature rather than projects. Here procedures are well defined (e.g. by a central QA-group) however the implementation in the daily life is only partly successful.
There have been processes with maturity level above 3, but up to now no project reached level 3.



Picture 7: Maturity level distribution

5.2 Frequent Weaknesses

Insufficient QA:

In some cases QA is only formally defined with insufficient resources or roughly undefined by insufficient competencies. In such cases QA can not support the development, high test costs and/or low quality is the consequence.

Early Phases:

Development starts on an unstable basis. Interworking of development and marketing, sales and service for specifying the requirements is insufficient. Improvement actions so far mostly focused on implementation and test.

Project Management:

Estimations are mostly done on an individual basis without defined method or tool support. Updating project plans is not sufficiently widespread. In many organisations progress tracking is based on accounting data (consumption of resources) instead of tracking of work products. As a consequence the missing of milestones is recognised rather late (fire fighting instead of preventive action).

Metrics:

Basic data is often incorrect and incomplete. A clear connection to goals (general goals of the business or specific goals of an improvement program) and actions (threshold conditions with mandatory activities in case of deviations) is missing [7].
As a consequence it is difficult to define and track quantitative goals for an improvement program.

5.3 Typical problems for improvement programs

If the start of an improvement program is postponed after an assessment („Schedules and resources do not allow to start now!"), the probability of a successful start later decreases significantly. After an assessment the staff members build up strong expectations for change. Problems described in the final presentation are well in mind. If this momentum is missed by waiting some month the general opinion tends to be: Nothing will happen.

Selection of PIT members is a critical task. Personality, technical knowledge and sufficient competencies in the organisation are needed.

Senior management must show continuously interest in and support for the improvement program (sponsorship). This implies e.g.

- direct regular reporting based on project plan and the defined metric system with visible feedback,
- personal presence in all relevant meetings
- introduction of special rewarding systems
- public announcement of a reassessment after 2 or 3 years.

All members of the organisation must have the impression that this is a project with high priority and that persons involved have full support by senior management in case of conflicts.

The improvement must be a project with adequate resources.

All employees should be periodically informed about what is just done, what is planned and what has already been reached (success stories) to keep up an attitude open for change. This can be done by a

regular newsletter, by information meetings which must provide a possibility for presenting and discussing problems frankly.

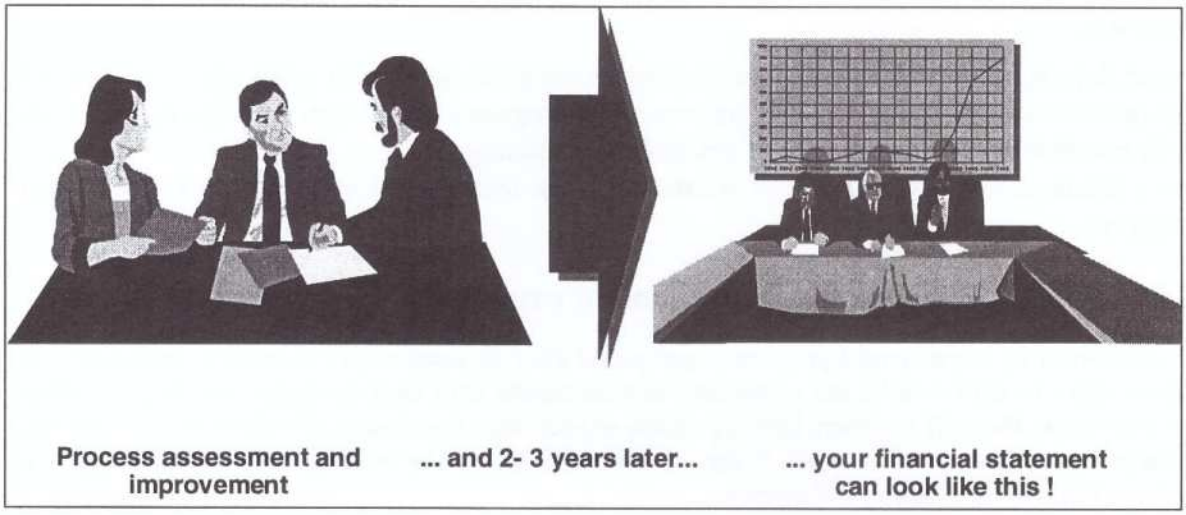
In some cases organisational structures (little kingdoms) prevent global solutions without very strong top management support.

Success in increasing efficiency can produce fear of loosing manpower. We always have to emphasise that this improvement enables the organisation to keep in market and to develop more or more complex products by the same staff.

5.4 Cost/Benefit Calculation for OPAL

The experience of our work shows that the effort for an assessment-based improvement program consumes about 3 - 7%) of the total development effort. This includes the assessment, the work done by a PIT and work groups and the introduction of measures (training, investment, learning). According to our experiences the efficiency improvement of a successful program can be around 30% within 2 to 3 years. The measurement of the improvement has to be done according to the goals of an organisation. It can be a reduction of error density after delivery, development cost or cycle time for development.

The program pays off very soon:



Picture 8: OPAL results

6 Literatur

[1] Watts S. Humphrey, Managing the Software Process, SEI, Addison Wesley, 1989

[2] SEI, The Role of Assessment in Software Process Improvement, CMU/SEI-89-TR-3, CMU, 1989

[3] SEI, Capability Maturity Model, CMU/SEI-91-TR-24, CMU, 1991

[4] SEI, SoftwareEngineering Process Group Guide, CMU/SEI-90-TR-24, CMU, 1990

[5] Software Process Unit Assessment Questionnaire, BOOTSTRAP, 2i Industrial Informatics, Freiburg, 1991

[6] Axel Völker, Sortware Process Assessments at Siemens as a Basis for Process Improvement in Industry, Proceedings of the ISCN'94 Conference, ISCN Ltd., Dublin, Ireland

[7] K.H. Möller / D.J.Paulisch, Sortware-Metriken in der Praxis, Handbuch der Informatik 5.4, Oldenbourg Verlag München, 1993

Quantitative Approach to Software Process Improvement

Annie KUNTZMANN-COMBELLES

OBJECTIF Technologie, F

ami history

ESPRIT project
three phases
products

ami promotion

- handbook diffusion (more than 2 500, 43 countries)
- de Facto newsletter (nb 12)
- referenced by SEI
- ami user group structure
- ami user group meeting (15 countries)
- 2 days workshop in April 1995

ONE OF THE CEC SUCCESS STORIES

Why ?

sponsors (CEC and european industrial community)

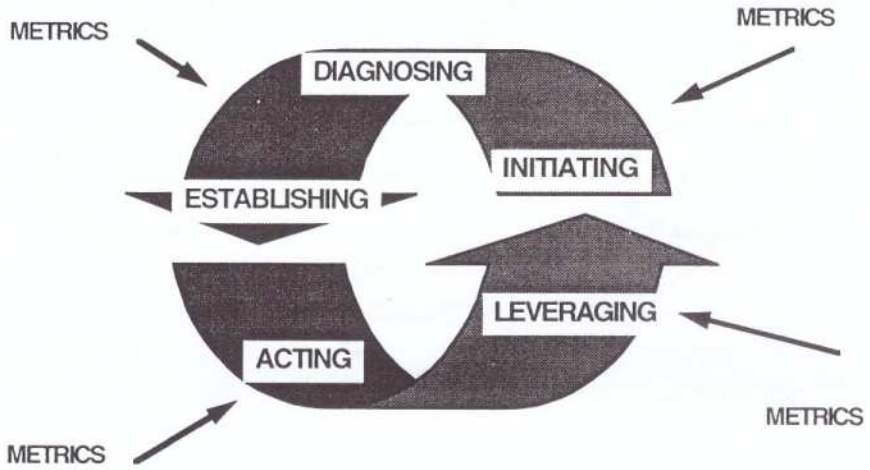
motivated and experienced project team

customer need exists

dedicated promoter

pragmatic and effective method

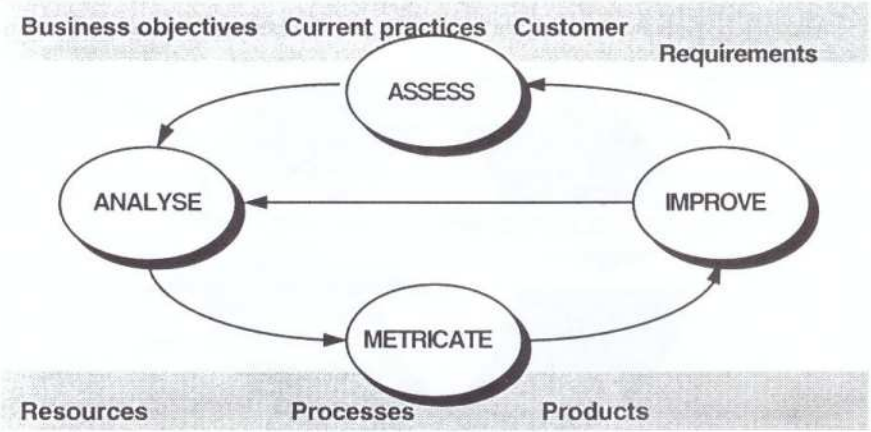
ami and Software Process Improvement



Questions

same metrics for each phase ?
how to define them ?
how to exploit them ?
how to relate them with business objectives ?

The ami route map



Initiating SPI with ami

- structure brain storm
- quantify the role of software in the business development
- analyse the origin of weaknesses
- understand

Establishing with ami

- put priorities among the domains to be improved
- select actions
- quantify detailed targets

Acting with ami

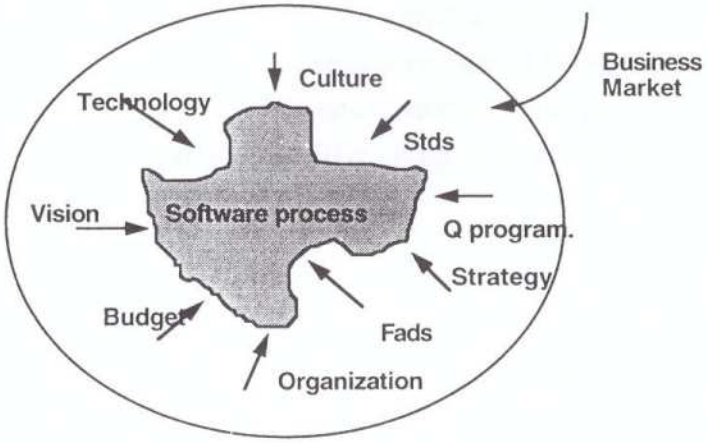
- monitor actions
- react to observations
- up date action plans
- check that goals will be reached

Leveraging with ami

- document and analyse lessons
- accumulate experiences

ami is generic

the software process is influenced by



Example (1)

BUSINESS GOAL

To reduce the time to market

Competitivity
Market shares

Telecommunications market
Multimedia
Large distribution

Example (2)

Results of the assessment

- Requirements are managed
- Project planning is an ad hoc process, not reliable, not repetitive
- Project tracking is not fully adequate
- Quality assurance is almost absent
- Configuration management is adequate

Example (3)

Goals tree

Example (4)

Make reliable estimates

processes : estimation
requirements management
review

products : effort estimation
planning
resources milestones

From goals to metrics (1)

estimation

Q1.1	Does a policy exists ? is followed ?
Q1.2	Are concurrent estimations achieved and compared ?
Q1.3	Is it using past data ?
Q1.4	Are past data validated ?
Q1.5	Is the volume and quality of past data sufficient ?
Q1.6	Is a margin evaluated ?
Q1.7	What is the effort spent to estimate ?
Q1.8	How many times re-estimations are conducted ?

Questions/metrics

Q1.1	Does a policy exists ? is followed ?	% projects
Q1.2	Are concurrent estimations achieved and compared ?	% concurrent estimations
Q1.3	Is it using past data ?	
Q1.4	Are past data validated ?	volume of data
Q1.5	Is the volume and quality of past data sufficient ?	
Q1.6	Is a margin evaluated ?	% projects/margin
Q1.7	What is the effort spent to estimate ?	% of total effort
Q1.8	How many times re-estimations are conducted ?	# of reestimations

From goals to metrics (2)

requirements management

- Q2.1 Do adeqaute requirements exist ?
- Q2.2 Is interpretation needed ?
- Q2.3 How many changes may occur after project start ?
- Q2.4 Are the changes mandatory ?
- Q2.5 Is the impact of a change assessed prior to decision ?

review

- Q3.1 Does the review policy exists ?is followed ?
- Q3.2 Frequency of rework of the estimation ?
- Q3.3 What is the effort dedicated to review ?

requirements management

Q2.1	Do adeqaute requirements exist ?	% completeness % of ambiguities
Q2.2	Is interpretation needed ?	
Q2.3	How many changes may occur after project start ?	# changes
Q2.4	Are the changes mandatory ?	
Q2.5	Is the impact of a change assessed prior to decision ?	% impact evaluated

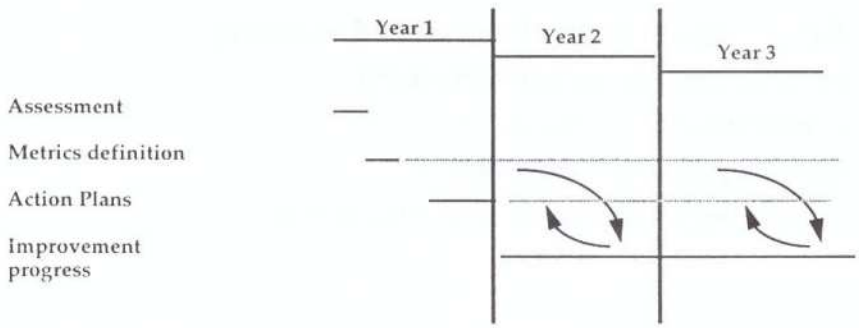
review

- Q3.1 Does the review policy exists ?is followed ?
- Q3.2 Frequency of rework of the estimation ?
- Q3.3 What is the effort dedicated to review ?

Example (5)
Actions

- install a data base and collection mechanisms
- review the estimation procedure
- organise systematic reviews of estimation
- claim for margin calculation
- define a process - adapted to customer - to resist to requirements changes

Schedule



Improvement Results

regarding Estimation Process

Before Actions

overrun between 15% and 20% for 70% of projects
lack of re-estimation

By the end of year 2

overrun 10% for about 40% of projects
less than 10% overrun for the others
quarterly re-estimations

project = development of a new product
or new version of an existing product

Key success factors

top management involvement and follow up
middle management involvement
committed action plans
metrics promoter (consultant)
leveraging on the basis of observations

Cost/benefits analysis

effort for SPI (4% of annual turn over)

effort for measurement (0,2% of annual turn over)

cost of non quality (25% of annual turn over)

measurement of improvement

ami CAN HELP

International Software Consulting Network - Managing Expert Networks

Richard Messnarz
Director of Process Development
ISCN Ltd, Dublin, Ireland

Micheal Mac an Airchinnigh
Trinity College
Dublin, Ireland

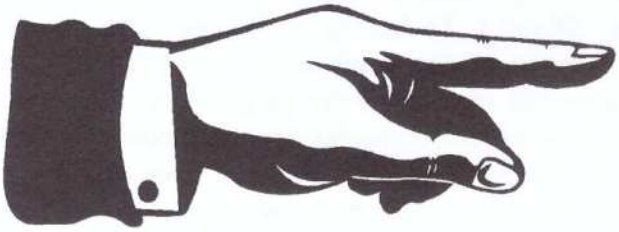
Paul Decrinis, Ronald Fellmann, Debbie Penney
ISCN Ltd.
Dublin, Ireland

Abstract. The main goal of I.S.C.N. - the International Software Consulting Network - is to set up a resource pool of experts involved in well known process improvement projects and initiatives such as ami, Bootstrap, CMM, Metkit, Pyramid, Scope, Spice, TickIT, etc. ([2], [4], [5], [6], [8], [10], [11], [12]) and to establish teams of experts for industry. This will lead to the exchange of knowledge and ideas in the process improvement field and will increase the benefits by combined use of the strengths of the different methodologies. As a quality oriented organisation ISCN [1] has established an annual conference series, to promote process improvement to industry. The first of these conferences was held in Dublin 1994, and as a result of the interest in ISCN '94 the next conference was planned for September 1995 in Vienna.
ISCN has progressed from the ISCN '94 seminar and is now set up as a registered company in Ireland. The co-ordination office is based in Dublin where the initial work of database compilation and standards for expert selection were implemented.
ISCN's mission statement is described in section 1, the structure of the co-ordination office and the expert pool and services in section 2, ISCN's work programme in section 3, the standards and procedures of the network in section 4, and the contact details and types of co-operation in section 5 of this paper.

1. ISCN's Mission

ISCN's Mission and Strategy. There are several methods for process assessment, measurement, and improvement, which can fit together in an overall process improvement framework. ISCN plans to encourage the combined use of such approaches and methodologies by using effective teamwork with experts from different projects and fields.
ISCN's strategy is to establish (i) an expert forum that will discuss new ideas at ISCN's conference series, and (ii) a resource pool which contains highly qualified experts who are interested in co-operating as team members. The members of the network are sourced internationally thus promoting transnational co-operations in the process improvement sector.

ISCN's Goals.



* *To establish a network of experts*
* *To provide teams of these experts to industry*
* *To co-ordinate and oversee the work of these teams*
* *To organise an annual process improvement conference*
* *To establish an electronic software engineering newspaper*

2. ISCN´s Structure

The Co-ordinating Procedures. The ISCN Co-ordination Office has established formal procedures for new membership, team selection, project co-ordination and network maintenance. Each expert´s details are stored in an Expert Skill Database (ESD) based on a standard Expert Skill Profile (ESP). Each customer describes his problems/requirements based on a standard Customer Problem/Requirements Report (CPR). The expert skill database supports the mapping of customer problem/requirements data onto expert skill data.

Qualification Procedure. ISCN members have to satisfy clearly defined requirements. Each expert´s customer references, project experience, trainer experience, publications, and skills are evaluated and checked according to a standard workflow. Experts are classified as senior consultant, consultant or junior consultant depending on their level of experience. If applicants do not satisfy all the criteria necessary their application will be rejected. This is not a permanent rejection and should their experience or skill level increase they may re-apply. This way ISCN ensures that the experts they promote are highly skilled, well qualified, experienced consultants.

ISCN´s Resource Pool. At present the ISCN resource pool contains experts with the following methodologies: ami, Bootstrap, ISO, MOOD, Scope, SEI/CMM, SPICE, and TickIT. Services which ISCN can offer to industry include also test planning and management, configuration management, object oriented design, formal methods, business planning, etc. In addition ISCN has established co-operation agreements with companies from Austria, Ireland, Finland, and US to offer their services through the ISCN network.

ISCN´s Conference Series and Expert Forum. The ISCN conference series addresses all aspects of process improvement strategies and methodologies - process analysis, process modelling, product quality evaluation, and practical process improvement. The first day is devoted to presenting improvement methodologies, while on the second day case studies are presented by industrial users to demonstrate personal experiences with the implementation of the methodologies. In the workshop all course participants have the opportunity to discuss their improvement requirements with the experts and the industrial users. To date two conferences have been organised - ISCN ´94, ESI-ISCN ´95 - the practical experiences from companies like Alcatel, Brameur, Festo, Motorola, Objectif Technologie, Process Inc., Q-Set, Siemens were presented. The papers given at the conferences are twinned - one methodology one industrial experience - this introduces the participants to the most recent methodologies and also shows how other companies have implemented them. Based on the conferences organised by ISCN an expert forum has been established. The purpose of this forum is to discuss new and recent strategies, trends, and developments in the process improvement sector. It is set up as a "talk back" facility on the Internet allowing ISCN members and consultants interested in becoming ISCN members to interact with each other.

The Co-ordination Office. The co-ordination office (Fig. 2) is the focal point of ISCN Ltd, managing customers queries, evaluating new membership applications, and using the expert skill database to establish teams. The co-ordination office is the main

organiser for the conference, and in future there will be a separate conference department (Fig. 1). This conference department will offer organisation services to members of the network who are planning conferences, seminars or workshops (e.g. the SPICE Symposium planned for May 1996 in Vienna).

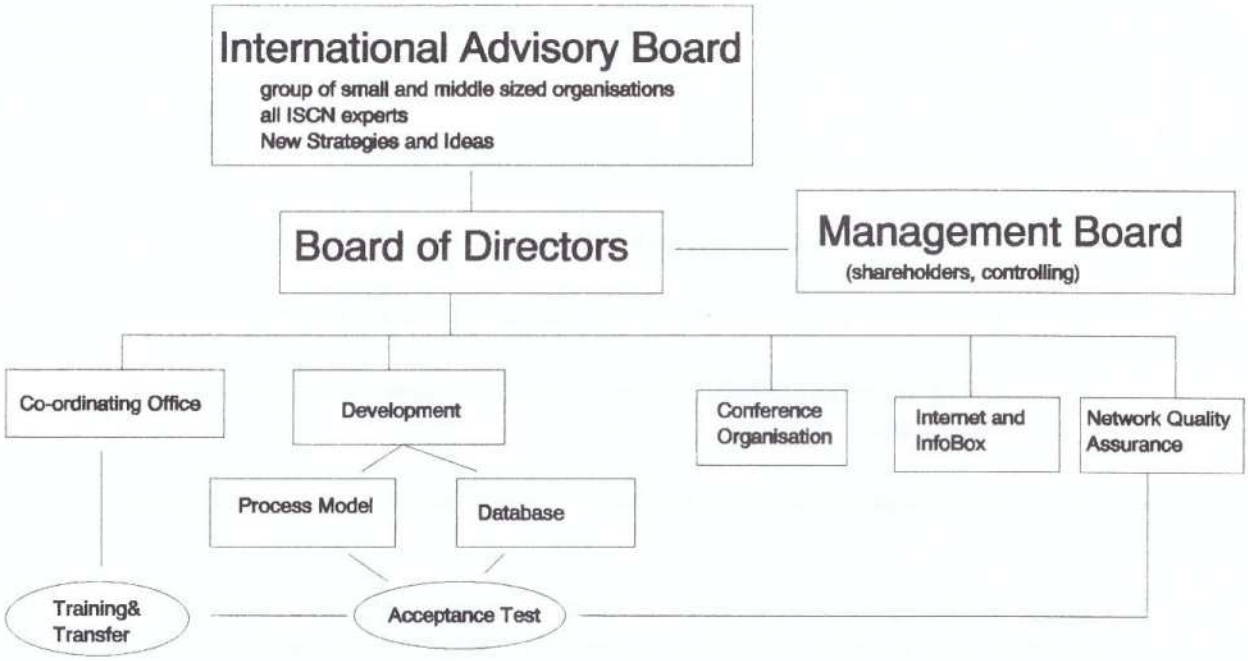


Fig. 1: ISCN's Organigram, 1995

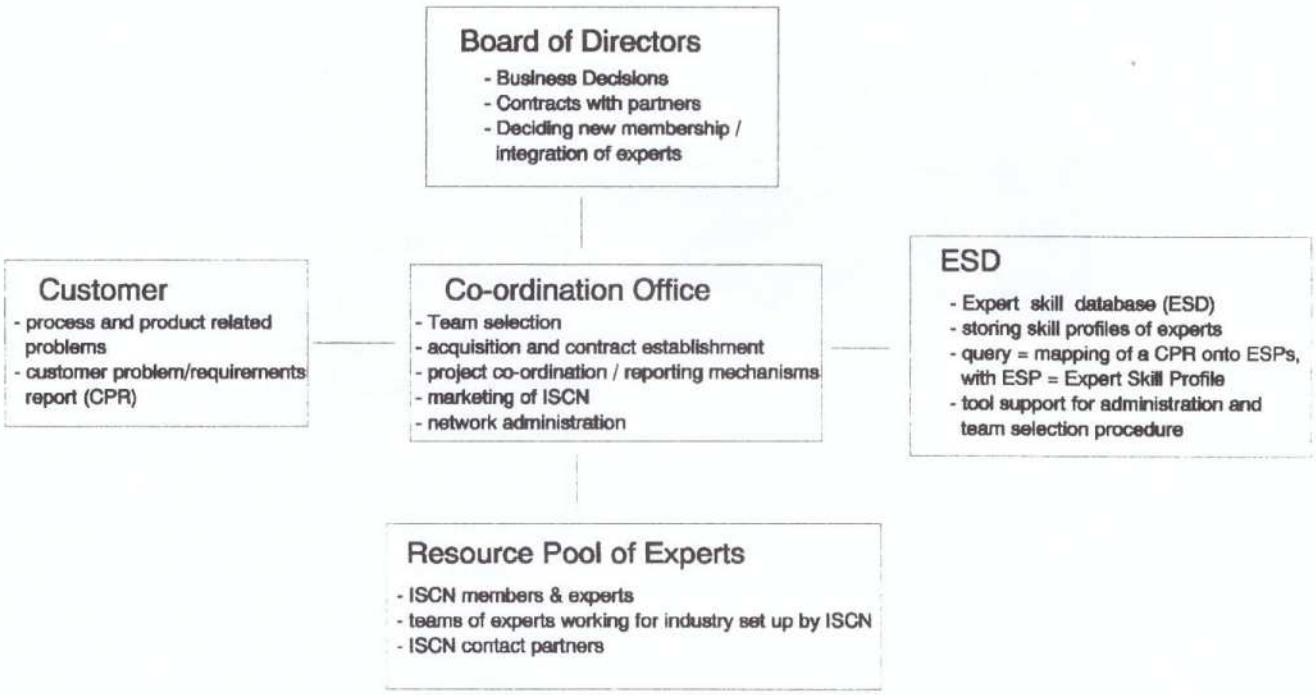


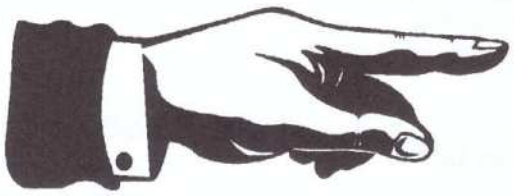
Fig. 2: The Co-ordination Office, 1994

The international advisory board consists of all ISCN experts, five companies from Austria, Ireland, Finland, and U.S., together with two Leonardo partners (formerly Comett), APS Leonardo and Hibernia UETP. The advisory board members do not make decisions but provide the directors with recommendations to aid with decision making. The directors - one Irish and one Austrian - are responsible for establishing business plans, deciding on new memberships, and the development and management of new projects. The following are examples of some of the ISCN projects already carried out:-

- ISCN procedure manual and workflow development
- ESD Expert Skill Database development
- Conference organisation manual and workflow development
- IBOX Information BOX development - electronic software engineering newspaper
- NQA Network Quality Assurance manual and workflow development

Based on the resource pool and co-operation agreements with organisations, ISCN is currently able to offer the following services. We expect to enhance the portfolio of services over the coming years by including experts from other methodologies and projects such as Trillium, Perfect, etc.

ISCN´s Services



** Strategic Planning*
** Business Planning*
** Bootstrap Assessments in Co-operation with Leansoft Oy*
**SEI Training and Assessments in Co-operation with Process Inc.*
**ISO Audits in Co-operation with APAC GmbH.*
**Combined Assessments Using More Than One of the Above Mentioned Approaches*
**SPICE Introduction and Training*
**TickIT Trainings in Co-operation with Q-Set Ltd.*
**Process Measurement Training*
**SCOPE Product Quality Improvement and Evaluation*
**Framework to Plan and Perform Process Improvement Based on the GQM Paradigm*
**Process Improvement and Change Management*
**Introduction to Total Quality Management*
**Test Planning and Management*
**Configuration Management*
**Formal Methods for Safety Critical Systems*

3. ISCN's Work Programme

3.1 Projects Performed in 1994

ISCN '94 Conference. To establish a first expert forum and to evaluate the interest in a network such as ISCN we performed the ISCN '94 conference "Practical Improvement of Software Processes and Products" in May 1994 in Dublin. This project was partly funded by the EU under the Comett programme. About 80 IT managers, consultants, and researchers participated in this conference, 90% of which were from industry. We received very positive feedback for such an initiative and thus decided (i) to organise a follow-up event in Vienna in 1995, and (ii) to continue establishing a network.

ISCN Modelling Project. In order to obtain a professional network it was necessary to define workflows and procedures for new membership, team selection, project co-ordination, and network maintenance. In addition we defined ISCN's mission statement, goals, and future structure. The outcome of this project was "The ISCN Procedure Manual" which clearly outlines the management tasks to be carried out by the co-ordinating office. Therefore ensuring that the co-ordinating processes start at a maturity level of 3, leading to a professional level of services.

Expert Skill Database Development. In parallel to the ISCN modelling project we designed and developed an Expert Skill Database (ESD) purpose designed to assign problems/requirements described in the Customer Problem/Requirements Report (CPR) to expert fields contained in the Expert Skill Profile (ESP). This allowed us to develop queries which automatically provide the co-ordination manager with a list of experts who meet the needs of the customer's problems/requirements. We developed two types of queries: (i) a "skill query" to obtain a list of experts who satisfy the required expert fields in order to complete the job, and (ii) a restriction query which takes into account restrictive data such as native language, experience in years as a manager, consultant, engineer, publication range (national, continental, international), minimum academic degree, etc. Both queries can be combined to display experts who satisfy the expert fields and all required restrictions. These queries can also be performed separately. The query inputs can be adapted, or modified, and then re-run. This way the database represents a decision support system for the co-ordinating manager, helping to find the most appropriate expert team. The ESA software engineering standards were applied for the development of this project.

3.2 Projects Performed in 1995

Establishing the Resource Pool. After running the 1994 projects we included information about ISCN in the ESI-ISCN '95 leaflet which was distributed in March 1995 to about 18,000 International IT managers, researchers, and consultants. We received about 80 membership requests, and more than 20 membership applications. So far the ISCN resource pool contains 25 members coming from different process improvement fields such as SEI/CMM, Bootstrap, TickIT, Scope, SPICE, etc. By the end of 1995 our aim is to have 50 members and our ultimate target is have 100 leading edge consultants.

Currently the resource pool contains 60% senior consultants, 40% consultants, and no junior consultants. In future ISCN plans to include more junior consultants who can work under senior consultants thus reducing the overall cost to customers.

ESI-ISCN '95 Conference. At ESI-ISCN 'Measurement and Training Based Process Improvement" the ISCN network will be presented for the first time as a legal entity with an initial resource pool offering a full range of services to industry. This conference combines European strategies (ESI, ESSI, ESPITI, Leonardo), methodologies (ami, Bootstrap, Metkit Scope, SPICE, TickIT), and industrial experience (Alcatel, CMS-British Steel, Brameur, ETNOteam, Festo, Obejctif Technologie, Process Inc., Siemens). There will be 30 presenters and we expect an attendance level of approx. 100 managers for all areas of the IT sector (industry, research and universities).

Consulting Projects. After establishing an initial resource pool ISCN started to offer services to industry. Already we have performed assessments [10] at large German organisations, establishing improvement plans, and offering other services like TickIT Training, and the GQM [4] approach for improvement planning based on the ami model. In the future we plan to establish about 15 to 20 integrated improvement teams per year. The results of these projects will be summarised in an annual report, which is made available to all ISCN members as part of their subscription.

Joint Projects and Proposals. To continue the idea of the combined use of different improvement approaches we contributed two Leonardo proposals:-

A set of comprehensive training modules and material for process improvement, combining different industrial approaches.	Software Process Improvement handbook, conference series and expert network.
PICO (**P**rocess **I**mprovement **C**ombined Appr**O**ach)	SPI-CONE

PICO. In Industry different approaches for planning process improvement and for the selection of best practice have been developed. In order for a company to remain competitive it is essential to establish effective procedures in all departments. These procedures require constant up-dating, therefore we plan to combine the experiences of the major approaches in process improvement and establish a fully integrated training package combining the strengths of the different methodologies. The courses themselves will have a configurable format consisting of a maximum of 6 modules (based on the results of recognised European projects; ami, Bootstrap, Metkit, SPICE, ISO and TickIT and the experiences of industrial users). It will then be configured by a procedure which will formally identify the skills and requirements of the attendees and select the proper set of modules based on the experience and skill data collected. This will allow the customer to mix-and-match the methodologies to suit their own specific requirements. Each module will take a minimum of half a day and a maximum of one day to perform. The project partners in these proposals are: ISCN Ltd., Leansoft Oy, Q-Set Ltd., Brameur, ami User Group, APS Leonardo, Hibernia Learning Partnership.

SPI-CONE. The objectives of the proposal are to write a software process improvement handbook, to organise a series of annual conferences and tutorials promoting new ideas and approaches in the area of software process improvement, and to establish a network of experts to offer state-of-the-art knowledge and services. The main target group for these services are International SME's.
In recent times the leading edge knowledge and new ideas for process improvement have been created in practical companies struggling with tight time-schedules, costly budgets for large software products, and ever growing competition in the marketplace. It is essential for them to improve their ability to predict and control their software production process in an attempt to survive in the market. The process improvement handbook will collect the most up-to-date knowledge from R&D projects and industrial experiences to apply their results to practice. This knowledge is captured via establishing a network of experts coming from the main R&D projects and software industry. The R&D projects behind the book will be ami, BOOTSTRAP, Metkit, SPICE and TickIT. The conferences and network will offer the possibility to capture new ideas in the improvement area at an embryonic stage and disseminate them back to industry and academia.

Company Brochure and Marketing. Based on the ISCN Procedure Manual, the conference leaflets, the resource pool, and the projects performed we started to design and produce a company brochure in August 1995. The brochure will be distributed at the ESI-ISCN '95 conference. In the autumn we plan to distribute it to about 18,000 IT managers, researchers, and consultants, and to start email and WWW marketing in US.

IBOX Project. ISCN has recently established an WWW server and has developed an "admin 1.0" software which supports an electronic software engineering newspaper on an Internet server. This newspaper will contain strategic and current information concerning software engineering and improvement planning. It will also include a summary of all ISCN projects and the results of these projects. There are plans to set up an open discussion forum "ISCN Talk-Back" where people from all over the world will be able to interact with ISCN members via Internet. Members who are interested in home pages on ISCN's information box will be able to buy these at a standard annual fee, with an additional fee for modifications to the page.

COM Project. The Conference Organisation Manual (COM Vers. 1.0) has been developed based on the experiences of organising the ISCN '94 and ESI-ISCN '95 conferences. The COM describes a phase specific and workflow based model for organising software engineering conferences like the ISCN conference series. It also contains guidelines for cost estimation (break even point analysis) and risk estimation. In addition there are PERT plans based on the workflows identified, therefore standardising the schedules for conference organisation.

3.3 Projects Planned for 1996

- To develop a CAD (Conference Administration Database) which works in a LAN and WAN

- To develop a training programme which allows people interested in ISCN to get information (we send an exe file and they execute a distance learning programme)

☎ 15-20 integrated teams for industry emphasising improvement and assessment

Conference Organisation : ISCN '96, SPICE Symposium '96

PICO and SPI-CONE: Process improvement handbook and combined training course

Electronic software engineering newspaper, WWW and selling home pages to members, the electronic newspaper will be co-financed by the Austrian Leonardo partner.

4. The Standards and Procedures of the Network

Defined Processes. A major goal was to start ISCN with defined processes at maturity level 3. The ISCN Procedure Manual project was based on business engineering and process modelling principles [4]. Business Engineering starts with the definition of strategic and business goals. From these goals subgoals are derived. Work flows have to be established, which as an output, produce the necessary data, conditions, or pre-conditions to be able to achieve the goals. The achievement of the goals should be verified and measured. These steps were followed when setting up the ISCN co-ordinating office.
In modelling work procedures we differentiated between data objects (a document, a form), organisational entities which execute work processes, and conditional objects which control when work processes are to be executed.

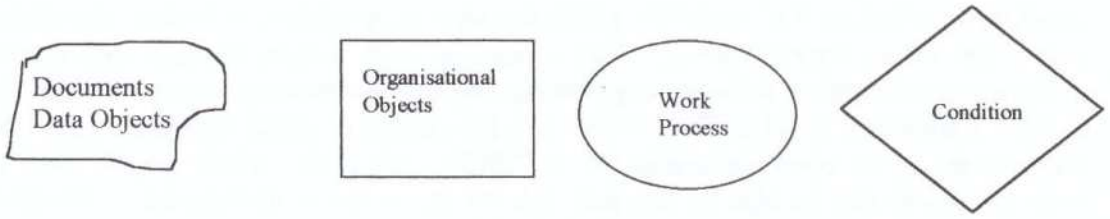


Fig. 3: Basic Modelling Objects [3]

New Membership Procedure. This procedure is applied when integrating new experts into the resource pool. The prospective expert has to contribute a standard ESP-form (Expert Skill Profile) and send it to the ISCN co-ordination office. Based on the defined membership criteria the skill profile is reviewed by the co-ordination office. The co-ordination office provides the management board with the ESP evaluation and classification based on standard evaluation checklists. The management board then decides whether to integrate the expert, and passes instructions to the ISCN co-ordination office to enter the expert's ESP into the ESD (Expert Skill Database) and include his details in the ISCN resource pool. The co-ordination office will issue a ISCN membership certificate to the new expert, showing his status and date of incorporation. If the management board decides not to include the expert, an official letter will be sent back to the applicant stating why the membership application was not accepted. This does not disqualify the person from applying again.

Team Selection. The customer describes his problems/requirements using a standard CPR-form (Customer Problem Report). The formalised CPR is the basis for performing queries in the ESD (Expert Skill Database) and for obtaining a list of qualified ISCN consultants to solve the customer's problem/requirements. Based on the list of experts obtained and on the customer's CPR the experts are contacted and the project is discussed. When a team of available experts has been contacted and established they receive the following information:- the customer's CPR, the names and the expertise of the other team members, and a proposal of how to co-operate within the team. Once this information has been provided to the experts they will contact the customer and establish work contracts. ISCN plays no part in forming the work contract between the team and the customer. Therefore allowing ISCN to remain independent and to promote each consultant equally.

Modelling Approach. For each work procedure an organisational diagram was established which illustrates all organisational entities and which data objects are exchanged between these entities (Fig. 4). Using the approach of "Levelling" each organisational entity was ticked and refined using a workflow diagram which shows which work processes have to be executed and in which sequence, and which data objects are to be used, created, modified (Fig. 5). Some work processes are so complex that it was necessary to tick and refine them more than once to obtain a more detailed work flow chart (Fig. 6). Workflow Diagrams also contain "conditions" which control if/when work processes are to be executed (Fig. 6).

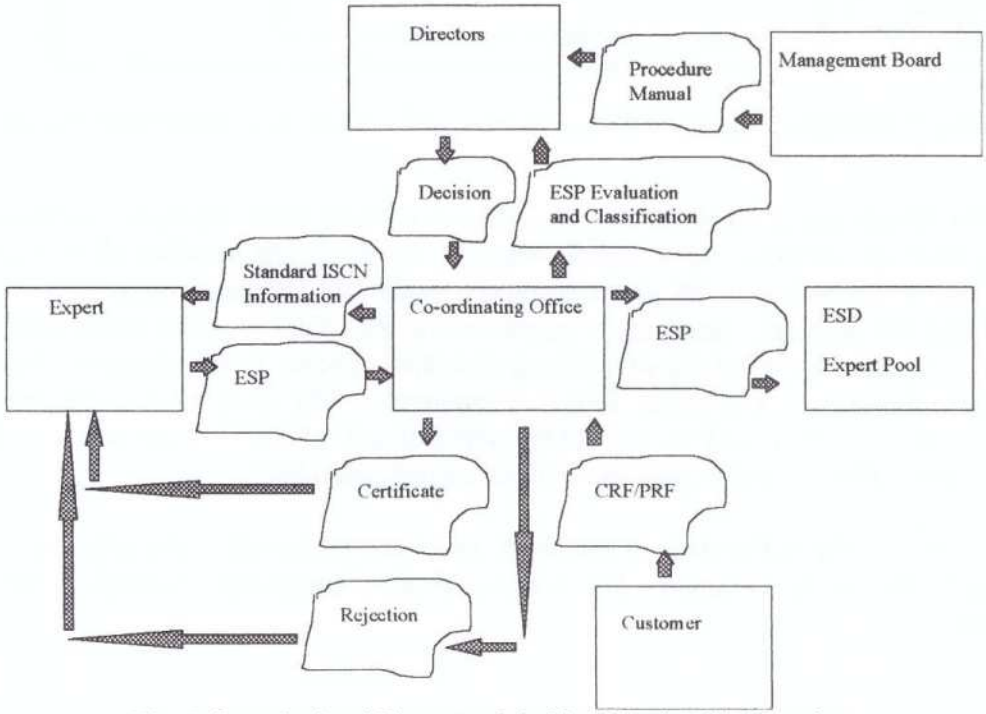


Fig. 4: Organisational Diagram of the New Membership Procedure

The procedure manual clearly describes the membership criteria, differentiating between mandatory and secondary criteria. To become a junior consultant an applicant must satisfy all type A and at least 1 type C criteria (Figs. 5 and 6). To become a consultant an applicant must fulfil all type A and all type C criteria. In order to become a senior consultant an applicant must satisfy all type A, C and S criteria. The procedure manual also provides checklists and decision forms to evaluate the correct classification of the expert's data. For example, the references (project,

training, publication) are cross-checked with the ticked expert fields, and if these do not correspond with each other it is not accepted as a proper reference. In addition we take into account the CRF/PRF (Customer/Project Reference Form) to evaluate if the referenced projects/trainings were successful.

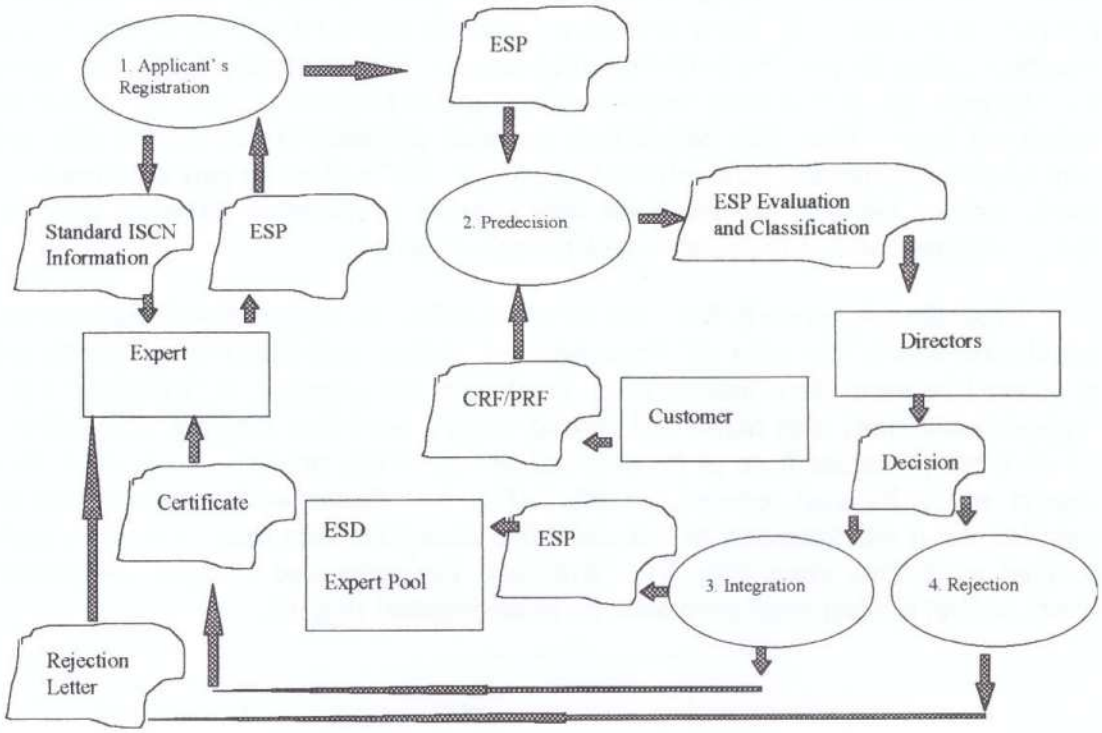


Fig. 5: Workflow Diagram for the Co-ordinating Office in the New Membership Procedure

In the "Predecision" process the co-ordination office uses standard checklists and evaluation procedures:- to evaluate whether the applicant satisfies all membership criteria, to cross-check the selected expert fields and the references given, and to evaluate the customer satisfaction based on the PRF/CRFs. Once these steps have been completed an initial classification is awarded. These classifications are rejection, junior consultant, consultant, senior consultant. The first ESP evaluation and classification is provided to the management board of ISCN who have to decide whether to integrate the expert and at what classification level.

Fig. 6 shows the refinement of the work process "Predecision" decomposing it into three further work processes. Fig. 6 also contains "conditions" controlling the work flows.

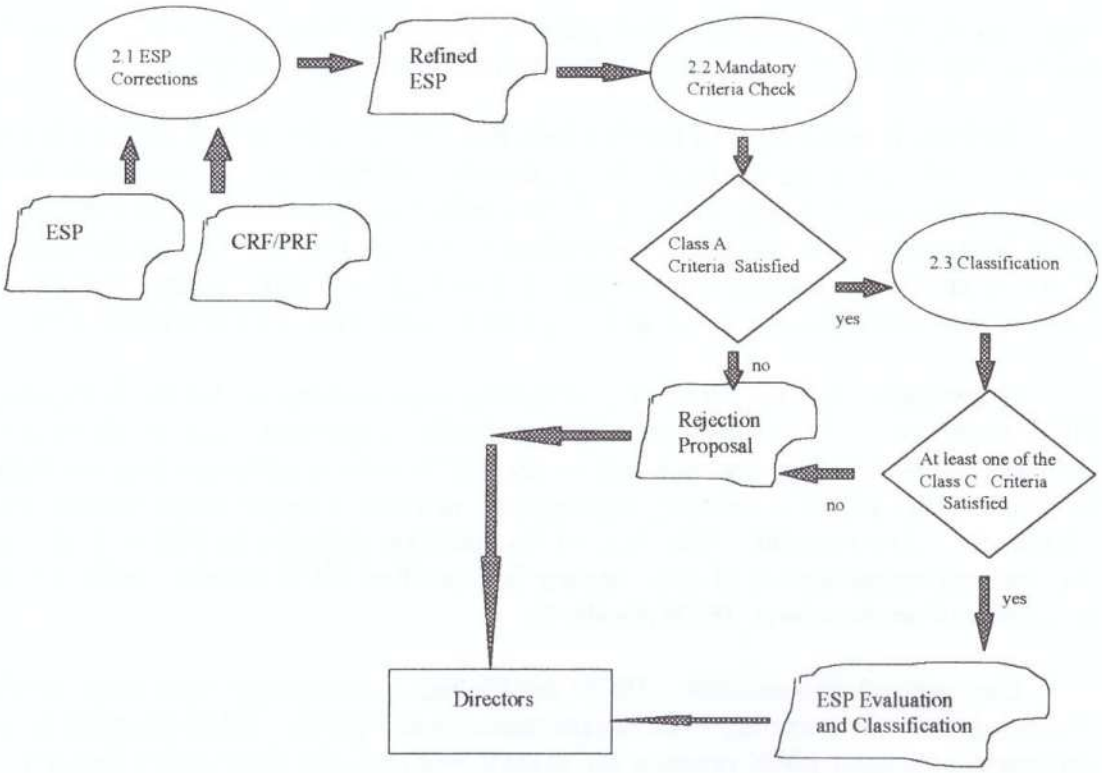


Fig. 6: Levelling Concerning the Work Process "Predecision" in Fig. 5

5. Contact Details and Types of Co-operation

ISCN Co-ordinating Office
I.S.C.N. Ltd.
Florence House
1 Florence Villas
Bray, Co. Wicklow
Ireland
Tel.: ++353 1 286 1583
Fax: ++353 1 286 5078
email: iscn@genrix.ie

If you are interested in becoming a member or if you are interested in ISCN's services please contact the co-ordinating office.

Currently we provide three types of co-operation.

Membership of the Resource Pool. ISCN experts go through a standard qualification procedure and receive an ISCN membership certificate. Each expert is promoted to industry and will be integrated into improvement teams. The annual membership fee is 200 ECU, which includes a price reduction of 20% for conferences

organised by ISCN (e.g. SPICE Symposium '96, ISCN '96), and an annual report summarising ISCN's activities and experience from improvement projects.

To Rent Home Pages on ISCN's Info Box. This facility is offered to all ISCN members. The aim is to get members to advertise and promote their companies and services under the umbrella of ISCN. Each product presented will be reviewed and tested by a group of at least two independent experts before insertion on the server. There is an annual subscription charge of 400 ECU for this facility. Non-ISCN members can also avail of this facility for an annual subscription charge of 800 ECU.

Organisational Co-operations. Consulting organisations whose employees are ISCN members can also use the ISCN umbrella to promote their products and services. Their services are included in the ISCN service portfolio and company brochure. This allows consulting organisations to reach a larger target market and address the correct people. This type of co-operation requires (i) that at least one management representative of the company is a certified ISCN member, and (ii) that the quality of services meet ISCN standards.

Contractual Framework. ISCN established a contractual framework which allows independent consultants and organisations to co-operate. When establishing an improvement project ISCN contacts the experts and provides them with information. This includes whether the consultant prefers (i) to work under the name of ISCN Ltd, (ii) to establish his own work contract with the customer in which case ISCN earns a provision, or (iii) to work under his own organisation's name - if the expert is an employee and not an independent consultant - in which case ISCN also earns a provision. Each ISCN expert confirms to adhere to ISCN membership rules which define standard quality and reporting guidelines including a formal evaluation of the customer satisfaction at the end of the project.

References

[1] Biro M., Kugler H-J., Messnarz R., et. al., BOOTSTRAP and ISCN - A Current Look at a European Software Quality Network, in: *The Challenge of Networking, Proceedings of the CON'93 Conference*, Oldenbourg, 1993

[2] Boegh J., SCOPE - A Guide for Software Product Quality Evaluation, in: *Proceedings of the ISCN '94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd. , Dublin, Ireland, 1994

[3] Chroust G., Modelle der SW Entwicklung, Oldenbourg, München 1992

[4] Debou C., ami - A New Paradigm for Software Process Improvement, in: *Proceedings of the ISCN '94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd. , Dublin, Ireland, 1994

[5] Galimberti R., BOOTSTRAP Institute - Services and First Results, in: *Proceedings of the ISCN '94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd. , Dublin, Ireland, 1994

[6] Haase V., Messnarz R., Koch G., Kugler H-J., Decrinis P., BOOTSTRAP: Fine Tuning Process Assessment, *IEEE Software*, pp. 25-35, July 1994

[7] Haase V., Messnarz R., Cachia R.M., Software Process Improvement by Measurement, in (ed.) Mittermeir R., *Shifting Paradigms in Software Engineering*, pp. 32 - 41, Springer Verlag, Wien, New York, Sept. 1992

[8] Kelly M., METKIT - Approaches, Experiences and Results, in: *Proceedings of the ISCN '94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd. , Dublin, Ireland, 1994

[9] Koch G., The ESI Approach - European Strategies for Software Process and Product Improvement, in: *Proceedings of the ISCN '94 Conference on Practical Improvement of Software Processes and Products*, ISCN Ltd. , Dublin, Ireland, 1994

[10] Messnarz R., Kugler H-J., BOOTSTRAP and ISO 9000: From the Software Process to Software Quality, in: *Proceedings of the APSEC'94 Conference*, Comput. Soc. Press of the IEEE, Tokyo, Japan 1994

[11] PYRAMID Consortium, ESPRIT Project 5425, *Quantitative Management: Get a Grip on Software!*, December 1991

[12] SPICE, *Process Assessment Guide*, Product Description, Version 1.0, ISO/IEC JTC1/SC7/WG10, September 1994

ESI's Products & Services

Günter Koch
Managing Director of
European Software Institute (ESI)
Parque Tecnológico de Zamudio
Building 204
E-48016 Bilbao
Bizkaia
Spain

Abstract: This article, submitted to the ESI-ISCN'95 Conference, describes in a top-down fashion how ESI develops its products and services in order to maintain them as attractive as possible for its members. Starting from the mission statement and from the envisaged membership structure a rationale is given for the positioning of ESI in its market and for its decisions on a set of products and services seen to be relevant for its mission implementation. A key contribution is on how ESI makes transfer of know-how into its members' companies effectively happen.

1. Mission, vision and their operationalisation

From its founding members ESI received end of 1993 the following **mission statement**:
ESI's mission is to support its members and European Industry to improve competitiveness by promoting and disseminating best practice in software engineering (and management).

This statement is continuously adapted to market needs, as in these days *(SLIDE 1)* such adaption is exercised by ESI establishing a professional business plan guaranteeing for the institute's future.

This mission statement gives the management of ESI a clear guidance what to consider for its operation, however, it does not provide a strategy which, in addition, has been developed by ESI's Board and management first by means of formulation of a **vision**.
(SLIDE 2)

This vision says, that ESI shall evolve from acting as an "agent of change" who is transferring best practice know how, to become ...

... a **Centre of Excellence**
... a responsive **Service Organisation**
... an **Awareness Creator**
... an **Organiser of International Co-operation Networks**

especially in the field of software & systems engineering & management (where software is interpreted from its technical instantiation in terms of IT programs up to its organisational application. This includes modelling and supporting of business processes in highly intelligent and complex environments).

ESI strives for becoming one of the world's foremost knowledge centres in the field of effective "softwareisation" and "virtualisation" of European industry.

2. ESI's members' structure

ESI's membership classification has been developed along two dimensions, by deciding

1. if an organisation is a supplier of software or of software intensive systems or if it is a user taking a maximum of profit from software supporting his/her business. Large companies today show both characteristics.
2. if ESI's "clients", i.e. its members are large organisations supporting the institute substantially as a sponsor or if they are members participating in the standard Corporate Membership program.

The simple classification scheme represented in (*SLIDE 3*) also indicates, that w.r.t. the final application domain of software, ESI is not limited and therefore offers its benefits to a wide range of vertical market participants. Supported through some market research studies, ESI foresees with a certain probability to attract specifically members with preference from privileged sectors such as

- Finance and Banking
- Telecommunication Operation, Products and Services
- Large Administrations
- Transport and Traffic and
- Production Industries

3. Competitiveness Factors

ESI's mission statement requests that it works for the benefit of its members' and European industry's competitiveness. So ESI's prime concern must be to identify the competitiveness factors in general for all software dependent companies and in specific for each client = member of ESI.

At a first glance these factors may look different for software suppliers compared to software users. However, knowing that still 2/3 of all software is "made" in house, i.e. at the users organisations, the difference in interest between the two groups is not too far. On the level of ESI this raises not a problem at all as both, software suppliers, which in Europe are solution suppliers, and software users have the same interests: to deliver software releases at high frequency, in time, at low cost, i.e. with an average productivity and at highest state-of-the art quality. These first level competitiveness criteria are shared across all ESI members and therefore are in the constant focus of ESI's management. Above this basic level more specific or even advanced factors count, however, these might be so much individual per each actor in the world-wide market that ESI may only refer to them through individual projects in co-operation with such member. Such higher order competitiveness criteria might be innovativeness w.r.t. a specific technology or methodology, communicativeness of the respective enterprise

versus its clients, responsiveness to new market requests, deep knowledge in the core field of competence or, since recent years receiving highest attention: the potential, ability and speed of organisational learning.
(SLIDE 4)

4. Measurability of Competitiveness Factors

Most of the competitiveness factors mentioned may only be of qualitative nature and may be perceived intuitively and as such give continuously reason for disputes between the competent actors who have to produce such qualities.

In a rationalistic and highly competitive market culture the identification and comparison of competitiveness factors or any other qualitative criteria is mandatory in order to position and to orientate any organisation such as are the members of ESI. Quantification of criteria through metrics has become a must, although within an infant discipline like software & systems engineering & management this may be not easy to conceive and to master. Without going to the details, reference is made to one successful supplier of software intensive systems, Motorola (USA), who has chosen the criteria of customer satisfaction, productivity, technology (roadmap) attainment, cycle time, and process qualities as their top level target and control parameters. *(SLIDE 5)*

To identify and customize such aggregation of metrics is one goal of ESI's benchmarking projects, through which ESI members are made fit for competition.

5. ESI's Role and Position

Besides the identification of competitiveness criteria ESI is expected to master successfully all kinds of transfer processes, in specific exchange of information about how to influence competitiveness by "leapfrog" innovations and by introducing successful methods of continuous improvements.

The question in this context is, which of the potential phase-dependent roles in a sequential transfer model *(SLIDE 6)* ESI may take: Research (R), Development (D), Productisation (P), Transfer (T), Use (U). The answer is simple to give: ESI's position is between R&D and U and therefore is to be associated to be a typical transfer agent and facilitator. In concrete, ESI has to make its distinction between being an applied R&D institute which it is also and not yet being a commercial consultancy organisation. *SLIDE 7* shows this position also by correlating such positioning to the dimension of penetration of an upcoming market, where ESI is positioned not to conflict with commercial consultants or consulting software houses, who themselves are preparing their end customers for the "next wave" business. ESI's positioning "at the early" end, i.e. where markets are rather to be opened than being developed or mature was confirmed by a study which was run by ESI with the support of Gartner Group, the result of which was that, e.g. "software process improvement" (SPI) is a market for which awareness creation has to be initiated rather than it is developed or even longing for existing products.

A specific conflict arises for ESI from the fact that not all of its members are in the same state of advancement. E.g. process maturity assessments following the model of SEI CMM have been completed in the so called systems industries already some three to four years ago, whereas some of ESI's members of user type such as in the financial sector have started to identify their position not earlier than since one year. This delay in transfer, however, makes the time span ESI has to cover by its services, which leads to potential conflicts with those members who are themselves actively approaching the markets of future professional services such as in assessment and improvement services *(SLIDE 8)*.

Overall ESI in its early days has decided to get its organisation started in the first year (1994), to create its products and services in the second and to serve its markets professionally in the third period. Today ESI s in transition from phase two to three.

Once gained momentum, ESI due to its positioning will have to continually evaluate the responses and requests from the market, i.e. will have to evaluate the feedbacked information from the receivers of its products & services in order to initiate subsequent new development and transfer cycles. This permanent adoption shall keep ESI in an always competitive mode which also means that ESI has permanently to adapt its program due to its members requirements.

6. ESI and its Members - Learning Organisations

As argued earlier, ability, speed and efficiency in learning will become one of the key competitiveness factors in future and so ESI naturally is challenged to find best ways, means and practices to implement what today is called the **Learning Organisation**. This is "an organisation specially skilled at creating, acquiring and transferring knowledge as well as modifying continuously its behaviour to reflect new knowledge and insights required to realise (new) business strategies". Although this requirement is universal to all industries, ESI has to find the more specific means and ways appropriate for software suppliers and users in order to drive their learning processes better than their competitors.

Creation of knowledge on organisational as well as individual level can be motivated and achieved in multiple ways in many different cultural contexts. E.g. learning may take place by

- **doing i.e. by processual application of knowledge,**
- **by using e.g. a new method, technology or application,**
- **by failing in acting**
- **by observing and concluding from someone acting**
  or
- **by listening and adoption.**

In a more concrete sense, one operational way to learn is through institutional education and training, which at ESI are established functions. These

functions today are completed by means of educative technology, in specific integrated information & telecommunication technology actually introduced under headlines such as multimedia or, more specific, computer supported education. *SLIDE 9* presents some communication options in teaching, cross-referenced to some of the training contents as to be introduced by ESI to its customers. The message from this chart is that for each specific learning context a specific application of one of the techniques introduced deserves a preference. We assume that a well composed combination of such media for a given receiver profile will lead to optimal learning results.

7. ESI's services for making the learning change happen

ESI as an institution offers different **products** (which ESI uses occasionally also as the term for its **services**) for transferring and communicating knowledge between member companies and from the outside world to its members' community. Such institutional services are

- **information services** from standard searches up to the creation of on demand dossiers
- **education & training** courses such as a postgraduate course in software engineering for universities, a doctorate qualification programmes as well as specific courses in additional qualifications of project and quality managers, engineers and designers of information systems and their contents.
- co-operative or individual technology & methodology, benchmarking and **improvement projects** together with its member companies
- **specific consultancy** and coaching, including assessments, benchmarkings and subsequent improvement plan implementations, applied to organisations
- **events** like conferences and workshop for exchanging experience and best practice know-how.

These services, all shaped for know-how transfer & exchange, are reflected by the three profit centre type organisational units called "Dissemination Services", "Member Projects" and "Consultancy (Package) Services", plus a complementary and extra unit called "SPICE" which is ESI's first strategic programme developing a full scope consultancy package including assessment benchmarking and improvement methods *(SLIDE 10)*.

8. Survey on ESI's Products & Services (Slide 11)

From a members point of view, the products and services "from the shelf" are ESI's basis and economic "raison d'être". They are presented in a catalogue under the category headlines

- Assessment
- Best Practice Based Positioning
- Improvement
- General Services

and

- Development of the future

where this last one is composed of three new strategic projects, which form each a catalytic framework for new products and services. These three future preparing projects are

- The "Learning Software Organisation", providing for a framework of changing the key competitiveness parameters of acquisition and exploitation of company knowledge.
- The "Global Software Enterprise", which is a composition of an organisational reference framework, basic telecooperation infrastructures and process implementations and allocations on a global scale, demonstrating the feasibility of a world-wide distributed, virtual software enterprise.
- The architecturing of the "R-A-C-P paradigm" which stands for a new approach in software & systems composition derived from requirements ® and then making use of standard architectures (A) which are to be realised, i.e. implemented by either reusable, prefabricated components (C) or by available products from the shelf (P). Focus of ESI's contribution will be the provision of an architecture patterns catalogue covering broadly systems architectures covering application domains which are typical for and at ESI's members organisations.

The products resulting from these three catalysing projects are not all yet well defined, however, this is one characteristics of prospective projects, that they rather convey new ideas and approaches than they already foresee the future in complete clearness and consistency by the predefinition of a firm set of deliverables and results to be acheved. In a sense, these projects prepare markets by giving future users of IT/software solutions a vision which technically will soon become feasible.

9. Joining ESI

ESI is a membership organisation completely open for all kinds and sizes of companies willing to participate in a programme the main purpose of which is to exchange, communicate and acquire knowledge on best practices in software & systems engineering and management. Due to different scopes and scales of interest and commitments, ESI today has to offer two levels of membership:

1. The "base level" Corporate Membership offering access to all of ESI's products and services as presented in ESI's catalogue. With the Corporate Membership fee of today 5,000,- ECU, a member establishes an annual "credit account" at ESI, for which the Corporate Member can "buy" ESI's services and products at preference tariffs.
2. The "upper level" Sponsoring Membership, providing the member with a series of additional advantages, such as

    - stronger representation and influence on Board level, to give ESI the direction decided by its members.

- ESI taking over all expenses, for official affiliations at ESI's headquarters in Bilbao.
- preference in the so called secondment of employees delegated from Sponsoring Members' organisations into ESI's co-operation projects which are performed at the institute at its cost (to the limitation of the annual membership fee).
- access to all the products and services as exactly offered to the Corporate Members at same conditions.

In other words: Corporate Membership is designed for enabling smaller organisation to develop on their own by short-cut imports of ready-to-use products and services, whereas Sponsoring Membership allows for far reaching and intensive preparation for competitiveness. Both, Corporate and Sponsoring Members form the unique ESI union which was founded to make its members provably winners.