31

A Method for Software Evaluation with respect to Quality Standards

Dieter Welzel and Hans-Ludwig Hausen
GMD
53754 Sankt Augustin, Germany. Telephone: +49 2241 14-3224. Fax: +49 2241 14-3006. email: welzel@gmd.de

Abstract

Based on a set of case studies in eight European countries a method of software evaluation has been designed within ESPRIT Project SCOPE (Software CertificatiOn Programme Europe). This method deals with several types of information: software characteristics and metrics, product and process information, and evaluation techniques. In order to be applicable, the method is supported by a five step procedure which analyses the quality requirements, specifies, designs and conducts the evaluation, and finally, reports on the collection of all documents produced in an evaluation report. The thoroughness of software evaluation is expressed by evaluation levels; encapsulations of evaluation techniques, in order to measure a quality attribute and manage the whole process more easily, are described by evaluation modules. Two guides have been produced and have been submitted to the responsible ISO/IEC JTC1 sub-committee for review and inclusion in normative documents being developed for the application of ISO/IEC 9126. As terminology standards IEEE 610 and ISO 8402 were taken into consideration.This method proposed can work with customised models as well as with standards.

Keywords

software evaluation, requirements for quality, software characteristics, software metrics, evaluation techniques, verification and validation

1 INTRODUCTION

Quality of IT products is a key element of the European software industry. To be able to assess software a practical but well-founded method for software evaluation is required. With a repeatable and unbiased evaluation the quality can be improved and the productivity of the software development process can be increased. The method to be applied has to conform with international standards and to contribute to the work of the national, European and International standardisation bodies. Therefore, the ESPRIT Project SCOPE (Software CertificatiOn Programme Europe) was launched in 1989 in order to:

S. Bologna et al. (eds.), *Achieving Quality in Software*

Table 1 Objectives stated in ISO/IEC Guide 25

Repeatability	Repeated evaluation of the same product to the same evaluation specification by the same testing laboratory gives the same result.
Reproducibility	Repeated evaluation of the same product to the same evaluation specification by different testing laboratories gives the same result.
Impartiality	Evaluation is free from unfair bias towards achieving any particular result.
Objectivity	The evaluation result is obtained with the minimum of subjective judgment.

- develop and experiment with an evaluation procedure, that is both technically well defined and cost effective,
- promote the use of modern software engineering technology for use in software evaluation and certification, and finally to
- contribute to the improvement of the European software industry.

The SCOPE consortium consisted of partners from eight European countries: Denmark, Finland, France, Germany, Ireland, Italy, Spain and the United Kingdom. The partners came from academic institutes as well as from industry. The consortium brought in European state-of-the-art technology used by software houses and testing laboratories. The project ended in June 1993 (SCOPE Consortium, 1993).

2 REQUIREMENTS FOR EVALUATION

In order to achieve the objectives of SCOPE an evaluation method was designed and applied in several case studies. The evaluation method was further refined and validated in two waves of case studies. Experimentation profited from 27 case studies in eight European countries. The procedure proposed may be applied in connection with:

- first party evaluation, i. e. internal product evaluation,
- second party evaluation, i. e. acceptance evaluation on product delivery, or
- third party evaluation, i. e. independent evaluation by, for example, a testing laboratory.

In order to develop an applicable practical evaluation procedure further objectives stated in (ISO/IEC Guide 25, 1990) have been considered (see table 1).

At the beginning of the project there was no complete description for an evaluation. Therefore, several approaches had to have a strong impact in the development of the method. The approach of GQM (Goal/Question/Metric (Basili and Rombach, 1988) supports the finding of metrics. Guidance techniques for this have been developed in the ESPRIT projects AIM and PYRAMIDE. Combined with a multi-level scheme for quality assessment (Hausen, 1989) the specifying of product-based quality models can be described. For describing the evaluation techniques to be applied in order to measure the software product or part of it a knowledge-based approach (Neusser and Hausen, 1989)

was adopted, where both the features of a technique and the invocation of related methods and tools are defined in terms of production rules.

3 OBJECT OF EVALUATION

In order to perform an evaluation several types of information have to be distinguished and used in an evaluation procedure: software characteristics & metrics, product information, process information, and evaluation techniques.

Each information type is described separately in a model. The characteristics of a software product as well as of its software development process and the attached metrics define the quality model. For the software characteristics the six characteristics of (ISO/IEC 9126, 1991) are referred. They are functionality, reliability, usability, efficiency, maintainability and portability. Product and process information is defined by an information model. Requirements specification, system specification, programs or handbooks are all examples for documents containing product information; management report, quality assurance report or project file are examples for documents containing process information. The techniques and tools model embraces the evaluation techniques and tools that are to be used to evaluate software attributes. Evaluation techniques contain verification methods, validation techniques, measurement procedures and assessment methods.

Thus, an evaluation process is defined by the identified relationships between the different models. In order to reduce the variety of evaluation and to achieve a reasonable evaluation procedure standardised descriptions of the information types can either be selected or have to be developed.

Which software characteristics and metrics are evaluated is a decision of the product provider who engages, for example, a testing laboratory. To support the identification of the metrics is a fundamental concern and the first step of each evaluation. This is the reason why the method proposed is called metric-based. Two basic concepts have been developed to simplify the evaluation: the evaluation levels and the evaluation modules.

4 EVALUATION LEVELS

Evaluation levels express the thoroughness of the evaluation in terms of the evaluation techniques to be applied. Each technique determines metrics and measurements. The specification of metrics selection is supported by three steps. Environmental, personal and economic aspects of the product to be evaluated give a first selection of an evaluation level (see table 2). There are four levels where D is the lowest level and A is the highest level.

Table 3 shows to which level the evaluation techniques are attached. The '+' notation in the table indicates the additional techniques when moving to a higher level. The next step requires the agreement on the metrics and their values. The required threshold value can be defined by using table 4. Generally, the contents of all tables are not fixed. They were developed by the industrial partners of SCOPE. But before an evaluation is started, the tables should be fixed and be the subject of a contract between the provider and the testing laboratory. Product-based standardised tables will simplify the process of identifying the quality requirements.

Table 2 Guideline for selecting an evaluation level

Level	Environment	Person	Economic	Application
D	small damage to property	no risk to people	negligible economic loss	entertainment, household
C	damage to property	few people disabled	significant economic loss	fire alarm, process control
B	recoverable environmental damage	threat to human lives	large economic loss	medical systems, financial systems
A	Unrecoverable environmental damage	many people killed	financial disaster	railway systems, nuclear systems

Table 3 Guideline for selecting evaluation techniques

	Level D	Level C	Level B	Level A
Functionality	functional testing (black box)	+ inspection of documents (check lists)	+ component testing (white box)	+ formal proof
Reliability	programming language facilities	+ fault tolerance analysis	+ reliability growth model	+ formal proof
Usability	user interface inspection	+ conformity to interface standards	+ laboratory testing	+ user mental model
Efficiency	execution time measurement	+ benchmark testing	+ algorithmic complexity	+ performance profiling analysis
Maintainability	inspection of documents (check lists)	+ static analysis	+ analysis of development process	+ traceability evaluation
Portability	analysis of installation	+ conformity to programming rules	+ environment constraints evaluation	+ program design evaluation

5 EVALUATION MODULES

The concept of an evaluation module was introduced to support the structuredness and manageability for the whole process. Without an appropriate structure evaluation would quickly become intractable, unwieldy and complex. Therefore, a well-structured, encapsulated description of software characteristics and the metrics and evaluation techniques attached to them had to be identified. Such a description lists the evaluation techniques applicable for software characteristics and names the product and process information required. It also defines the evaluation procedure and the format for reporting the results of applying the metrics and techniques. In addition the information necessary for an estimation of the costs is provided.

Thus, an evaluation module encapsulates

Table 4 Guideline for selecting metrics

level	technique	threshold metric	threshold value
...	...	...	...
A	white box testing	statement coverage	100 %
		branch coverage	95 %
		...	...
		condition coverage	90 %
		expression coverage	90 %
B	white box testing	statement coverage	95 %
		branch coverage	90 %
		...	...
		condition coverage	85 %
		expression coverage	85 %
C	white box testing	statement coverage	91 %
		branch coverage	85 %
		...	...
		condition coverage	not necessary %
		expression coverage	not necessary %
D	white box testing	statement coverage	85 %
		branch coverage	80 %
		...	...
		condition coverage	not necessary %
		expression coverage	not necessary %
...	...	...	...

- the definition of one or more atomic evaluation procedures applied to the product or process information in order to measure software characteristics or sub-characteristics,
- the attachment of metrics and evaluation levels to those characteristics, - the description of the assessment procedure to be applied,
- the format for reporting the results and cost figures.

In other words an evaluation module also contains, beside the information needed, the way of how measurements can be performed on (parts of) the software product. An example of an evaluation module is given in (Hausen and Welzel, 1993b, Annex) and an example in table 5.

6 STEPS OF EVALUATION

As an example of the evaluation method proposed a five step procedure was designed within the SCOPE project. The intended use of the procedure is for actually running an evaluation (including case studies). The view of an independent testing laboratory is taken. The procedure describes the activities carried out by the testing laboratory and the interaction between the testing laboratory and the client (e. g. producer, distributor,

Table 5 Example of an Evaluation Module

EM Requirements	
scope of application:	'off-the-shelf' end-user software products
software characteristic:	ISO/IEC 9126 Usability
evaluation level:	Level D
evaluation technique:	Inspection by checklist
EM Specification	
sub-characteristics:	installability-from-scratch (INST), learnability (LRN), use-efficiency (UE), customisability of interface by the user (CUS), experienced-user-migration-ease (UME)
metrics attached to sub-characteristics:	*Metric ID.*: 1.11 — *related to*: INST, UME
	selection constraint: new directories are automatically created
	Metric — *Value*
	IF new directories are automatically created, is the user informed? — yes, in all cases $-> 2$; no, in all cases $-> 0$
aggregation of metrics:	total number of 'points' per sub-characteristic
EM Application Procedure	
how to get the information to be able to answer the questions	
EM Application Report	
document the evaluation procedure, collect all measurements and assessment results, prepare a cost report	

buyer, or user). The client is the person or institution who negotiates the evaluation specifications with the testing laboratory.

Figure 1 provides an overview. It describes the sources of input for the evaluation and the steps of the evaluation procedure.

6.1 Analysing Evaluation Requirements

The evaluation requirements are formal records of the agreement between client and testing laboratory of what has to be achieved by the evaluation process. It provides a nominal list of software characteristics which are to be evaluated at which evaluation level and identifies the source of data and evidence which might be used in the evaluation process. Software characteristics may be functionality, reliability, usability, efficiency, maintainability, portability (from ISO/IEC 9126).

6.2 Specifying the Evaluation

The evaluation specification contains the more formal description of the evaluation requirements. It includes available documents identified and received items classified into product, process and (for the evaluation process) supportive information. The classification makes use of an information model which identifies the types of information needed for an evaluation (compare figure 2).

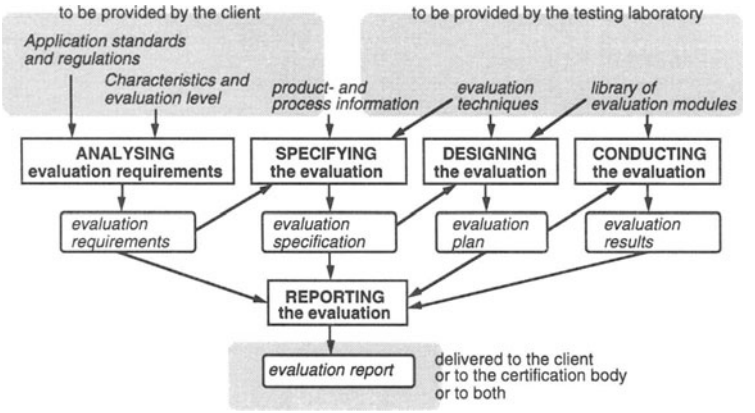

Figure 1 The Evaluation Procedure

The analysis of the product comprises two phases:

- identification of the product, and
- classification of the received items into product, process, and supportive information.

In the identification phase the following should be considered:

- document identifier
- document title
- condition of document (physical appearance, abnormalities)
- date of receipt
- legal implication of document handling (document security, confidentiality)

In the classification phase the items received are classified into the following:

- product information
- process information, and
- supportive information.

It is not required that the structure of the documentation received exactly follows the information model, but it must be possible to identify and extract the required information from the material received.The information model covers the development of a complete system which may include both hardware and software, but only the software part is subject for evaluation. In general, not all types of information are required for an evaluation. The required information depends on the selected characteristics and the corresponding evaluation levels.

The specification of the evaluation should be organised according to the quality char-

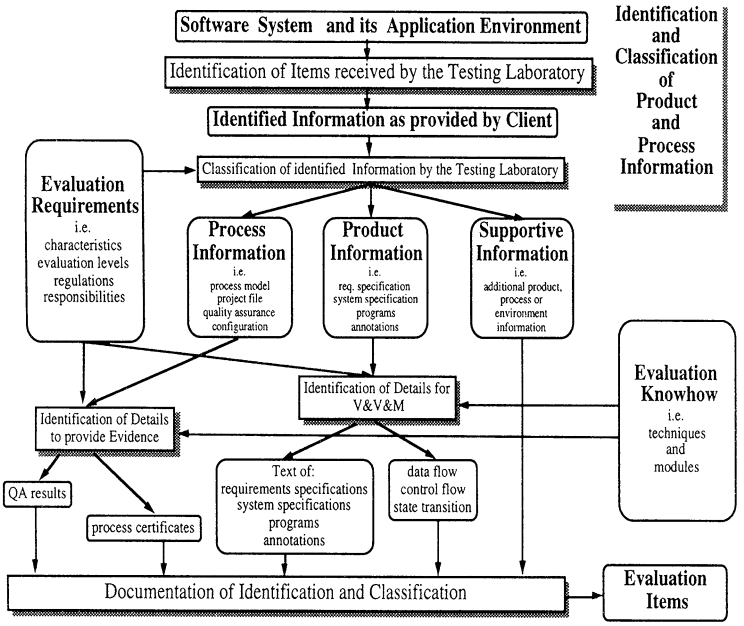

Figure 2 Specifying the Evaluation

acteristics, in this case the characteristics of ISO/IEC 9126. The evaluation specification associated with each characteristic must be formulated as a combination of the following types of statements:

- an exact reference to statements in a requirement specification document, user manual, or possibly other information, which should specify the program requirements that are to be evaluated,
- a statement about the software product which is either missing in the program specification or needs to be explained more carefully for the evaluation
- an exact reference to statements in identified standards and in regulations documents where additional program requirements are given which should also be included in the evaluation specification.

Only functional and non-functional requirements mentioned or referred to in the specification are subjects for evaluation. Therefore the evaluation specification must be detailed and complete.

Based on the classified items and the evaluation level a first feasibility study can be performed.

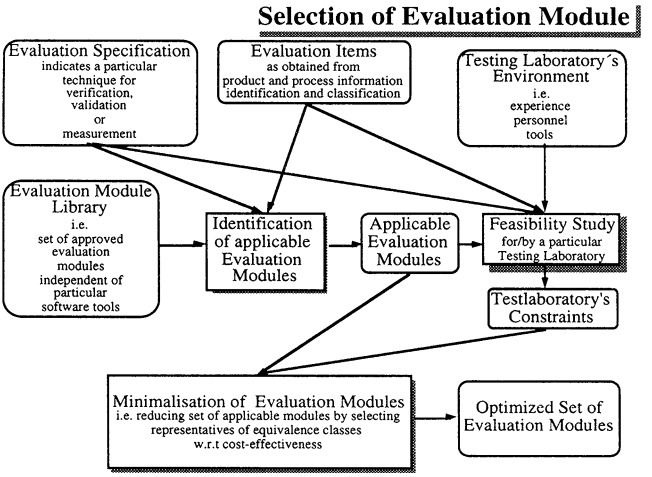

Figure 3 Selection of Evaluation Modules

6.3 Designing the Evaluation

In the design step the evaluation modules are selected from the evaluation module library. The selecting process is implied by two criteria:

- the module must be known and recognised to be useful in the evaluation of the characteristic for which it is to be used,
- the module must be applicable to the product part on which it is to be used.

However, this set of modules may not be optimal for carrying out the evaluation. Some modules may be redundant and some may be missing. It must be decided whether new modules should be developed or whether missing modules can be substituted by a combination of existing modules. The purpose is to make the final planning of these modules for the evaluation. The planning must be done in order to optimise the coverage of and the cost of conducting the evaluation.

The optimised set of modules requires product, process and perhaps supportive information as imposed by their input interface. So a refinement and adoption step may be necessary to relate the information needed by the modules to the items identified by the application of the information model.

The evaluation plan includes a list of modules to be applied. Each evaluation module includes information from which the cost of its application can be derived. Hence, it is possible to give a fairly good estimate of the cost of conducting the evaluation at this point.

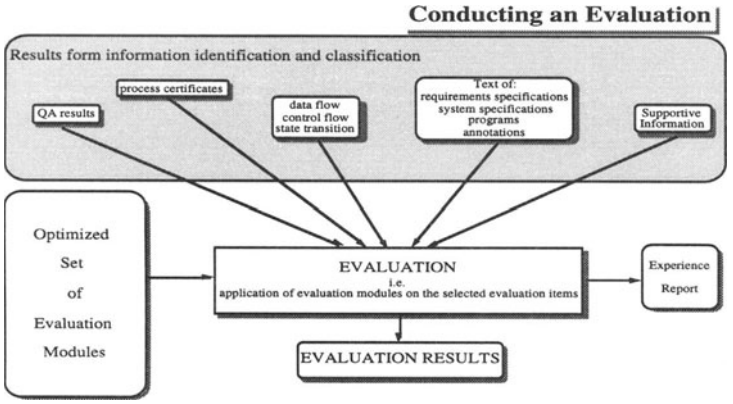

Figure 4 Conducting an Evaluation

6.4 Conducting the evaluation

Conducting the evaluation then comprises the application of the set of optimised evaluation modules on the related documents and collecting for each of them the results of validation, verification, measurement and assessment.

Measurements can be manual, computer aided (e.g. using a check list manager for applying check-lists), or automatic (e.g. measuring complexity in a source code component using a static analyser).

The main task is to collect the measurement result and also to keep any information about the measured product part that could be helpful for an acceptance decision.

The selected evaluation modules are applied according to the schedule given in the plan. The results of applying the individual modules are recorded in the evaluation report. Observations made during the process also have to be included in the evaluation report.

The application of an evaluation module comprises three steps:

- measurement according to metrics identified by the module,
- assessment by comparing measurement results with the acceptance criteria,
- recording the measurements results and results of the assessment.

Depending on the results of evaluation modules an aggregation of the module results is necessary.

6.5 Reporting the Evaluation

The final step of the evaluation is that of producing the evaluation report. The table of contents of the report follows the steps described above, and each of the steps are documented during the evaluation process.

The following table of contents is suggested for the report:

1. Preface - identification of producer and evaluator
2. Evaluation requirements - product overview, quality characteristics, evaluation level
3. Evaluation specification - identification and classification of items, detailed specification
4. Evaluation plan - selected evaluation modules, evaluation process planning
5. Evaluation results - results of applying evaluation modules
6. Conclusion of the evaluation results - including signature, responsibilities, limits of results, distribution of report

7 CASE STUDIES

The SCOPE project used 30% of its total effort to conduct case studies. This was in order to gain practical experience with the evaluation procedure produced, and also to ensure that the approach can be applied in practice. The case studies were conducted in two phases. In the first phase six case studies were carried out in an experimental fashion. They tried out different approaches to software evaluation together with different evaluation techniques. The case studies of this phase were selected based on their availability and not on any particular selection criteria. The experiences from these evaluation experiments were carefully analysed. The result of the exercise was a stepwise procedure for conducting an evaluation.

In the second phase 21 case studies were carried out. The main objective was to demonstrate the practical feasibility of the evaluation method proposed as described in the procedure. To achieve this the case studies were selected according to different criteria:

- They should be concerned with products representative of those most likely to be in need of evaluation and certification.
- Evaluation techniques, application areas, and fields of software engineering should be well covered.
- The evaluation procedure should be tested to demonstrate that it is practical and robust.

As a consequence of this careful selection process, the resulting set of case studies covered a wide range of applications including administrative and technical systems, software tools, communications protocols, and embedded systems, see table 6. In addition a wide range of commonly applied development approaches was covered. This included standard third generation life-cycles, prototypes and systems developed in 4GL. The trial evaluations covered the quality characteristics defined in ISO/IEC 9126 with a focus on functionality, maintainability, and usability. Most evaluations were at the low to medium level of stringency. That is, most case studies conducted evaluation at the C and D levels and only a few at the B and A levels. The actual distribution of case studies on levels and characteristics is shown in table 7 which reflects the actual demand of the case study providers for thoroughness of the particular evaluation.

A number of different evaluation modules were tried out. 19 out of 21 case studies used checklist-based evaluation modules. They are applicable in most cases, easy to use and very flexible. Static analysis tools were applied in 11 case studies. It is not always possible

Table 6 Case Study Application Areas

Case study product application include:	
Process control	Accounting
Electronic mail	Traffic control
Medical Application	Stock management
Phone exchange	Operating systems
Desktop publishing	Management info system
Electronic point of sale	Process monitoring
Picture generation	Message handling
Image processing	Graphical analysis
Fire alarm	

Table 7 Distribution of Case Studies on Levels and Characteristics

	Level	D Level	C Level B	Level A
Functionality	4	13	1	(1)
Reliability		1		
Usability	7	6		
Efficiency				
Maintainability	2	13	1	
Portability	3	2		

to apply such tools, but when it is possible they are thorough and efficient. A variety of other evaluation techniques were used in 7 case studies. These techniques, which often require specific application support, include Petri net analysis of software specifications and reliability modeling. An overview of evaluation techniques and evaluation modules applied in the case studies is given in table 8.

The main objective of the second phase case studies was to allow the collection of practical experiences with the evaluation procedure, and to ensure that the approach is applicable in practice. This was indeed the main conclusion in most of the case studies, each of which gave feedback on many aspects of the evaluation procedure. They identified points with a need for refinement of the procedure. These refinements were then implemented in the procedure. The efficiency and effectiveness of the evaluation method was assessed by monitoring the effort incurred from applying the evaluation modules as well as their impact on the result of the evaluation. All evaluation modules were tried out in one or more of the studies. In conclusion, the case studies successfully achieved their goals.

Essentially all the case study providers were very positive towards the evaluation process, the results, and the experience they had gained through their participation. The case studies showed that it is feasible to carry out software product evaluation according the procedure proposed.

Table 8 Evaluation Techniques and Evaluation Modules used in Case Studies

Evaluation techniques applied	Evaluation modules applied
Checklists used in 19 case studies - easy to use - subjective results - applicable in most cases	*checklists used to asses* Requirements (5 Modules), Design (9), Source code (16), Test documentation (5), User manual (10), Safety/security aspects (8)
Static analysis tools used in 11 case studies - efficient and thorough - meaning of measurement - value not clear - application not always possible	*Static and dynamic analysis* - application of Logiscope and QUALMS - measurement of structural parameters - measurement of test coverage
Other techniques used in 7 case studies - inspection, interviews, tools - each applied in one case study - application support necessary	*Petrinet Analysis* - application of Design/CPN *Reliability Analysis* - application of SW reliability modeling programs

8 RESULTS OF THE CASE STUDIES

It is reasonable to expect that the software products evaluated in the case studies were representatives of the high quality part of the software available on the market. However only about half of the evaluated products successfully passed the acceptance criteria. Many of the products had a pronounced lack of quality. Often documents necessary for the evaluation were completely missing or the contents were clearly unsatisfactory. Obtaining design documentation was especially difficult in many case studies. Other problems encountered were missing functionality and omissions in general.

These case studies which worked with their own reduced version of the evaluation procedure asked for more details and for objective decision support. Therefore it was decided to include all possible details into the an evaluator's guide.

The application of the evaluation modules also resulted in many comments. Checklist based evaluation modules were most popular in the case studies and consequently most experiences were accumulated for these evaluation modules. One conclusion was that if checklists are carefully designed with thorough explanations to each question and if the checklist includes at least 25-30 questions, then the subjectivity involved in this evaluation technique is within acceptable limits.

Static analysis techniques were experimented with in half of the case studies. This technique gives objective measures but their interpretation was considered as being difficult. However, the evaluation modules applying these techniques were very efficient for identifying program modules containing problems.

The evaluation modules applied in the case studies were not developed and documented in a common way. This resulted in confusion and misunderstandings which could have been avoided. Therefore the need for a guideline for producing evaluation modules became evident.

This ultimately led to the presentation of the guide that describes how to design, produce and maintain an evaluation module. The procedure of developing an evaluation module comprises five steps. After analysing the requirements of the module to be developed (step one) the module is to be specified (step two). The writing of the module (step three) has to follow the required evaluation module structure. A validation of the module (step four) ensures the fulfillment of its requirements . Validation comprises both a technical review to ensure that the module represents state-of-the-art and practical trials on real software products to ensure the modules' applicability in practice. Finally, the module has to be embedded into the Evaluation Module Library (step five).

The case studies stated the need of storing the experience gained in a data base in order to make them available for further investigations which could lead to an improvement of evaluation procedures or particular techniques. In order to achieve an effective evaluation process it is necessary to reflect experience gained with evaluation modules and the module library, appropriateness of levels and software characteristics, appropriateness of product representation, appropriateness of process representation, calculation of actual costs in order to improve cost estimates, appropriateness of the evaluation method.

The case studies have shown that the evaluation procedure can already be used in a wide range of contexts such as:

- *Product Certification:* Software product certification can be defined and performed in compliance with the various standards and constraints using a fully defined evaluation procedure.
- *Independent Evaluation:* Software product evaluation can be performed by an independent testing laboratory according to the Evaluator's Guide.
- *Acceptance Testing:* Departments in charge of performing acceptance testing of delivered software products could use the Evaluator's Guide to assist them when specifying and organising their activity.
- *Contractual Requirements:* Specifying a software to be subcontracted could be complemented by the technical quality requirements to be met and by appending the set of Evaluation Modules to be used for the final acceptance testing.
- *Product Ranking:* The comparison of two software products regarding quality can be performed by comparing how they behave against the results of a fixed and repeatable evaluation procedure such as the one developed.

One of the most important aspects of the resulting technology is its potential ability to adapt to the ever changing world of software engineering. Among the various foreseeable changes, we have considered, for instance

- *Evaluation Tools:* As the evaluation activity grows and matures, many new supporting (software) tools and products will appear, the integration of which within our framework will have to be as straightforward as possible while still preserving the know-how of the parties involved,
- *Evaluation Techniques:* Similarly, the overall evaluation technology itself will progress, while, hopefully, not making the fundamental results of the task obsolete,
- *Development Technology:* The need to be able to adapt to ever more programming languages or environments has been a constant driver to the design of the documents as for instance, the Evaluation Modules and their structures illustrate,

- *Harmonisation with Other Fields:* The search for quality is one of the major drivers of a lot of work in the software fields. Security and safety domains become more and more important. Thus, we must keep an eye on future harmonisation and convergence.

With this flexibility the circumstances of any testing laboratory can be taken into consideration. The testing laboratory can find out which kind of software evaluation it can offer.

What are the final conclusion form the case studies? First, an evaluation by improper qualified personnel, immature methods and without the Evaluator's Guide (or a similar guidance) produce results which are not useful for management. Therefore, both a certified procedure and a certified staff are required. Secondly, most of the evaluation tools available were immature because, amongst other difficulties, different tools produced different results for the same metric. This shows the need for well-instrumented metrics and measurements. Consequently, metrics and measurements have to be standardised. Product-type specific metrics are needed. Finally the integration of the evaluation procedure into software engineering process models (such as VORGEHENSMODELL, SSADM, MERISE) is considered as being necessary to ensure at quality assurance.

9 STANDARDISATION OF THE METHOD

With the five step procedure guidance is provided on what has to be done, how the work has to be carried out and how it has to be documented. The procedure supports planning, designing and controlling of an evaluation process tailored to specific circumstances. To help the evaluator two guides have been produced (Hausen and Welzel,1993b, and ISO/IEC 9126: Guides to software evaluation, 1993) that have been submitted to ISO/IEC JTC1/SC7 "Software Engineering" for review in WG 6 "Evaluation and Metrics" (Begh et al., 1993). They are parts of normative documents currently being produced for the practical application of the International standard (ISO/IEC 9126, 1991). They are dedicated to particular aspects:

- an Evaluator's Guide (EG), which describe the five-step procedure.
- a Guide to Developing, Documenting and Validating an Evaluation Module (GDDV), which describes how to design, create and maintain an evaluation module.

The evaluation process is defined using the EG and applies evaluation modules which are developed and documented according to the GDDV.

Although the present guides explicitly refer to ISO/IEC 9126 similar quality models can be applied without severe changes to the underlying evaluation procedure.

Figure 5 shows how the Evaluators Guide and the Evaluation Module Development Guide fit into the set of guides being discussed in ISO/IEC JTC1/SC7/WG6.

To demonstrate an application of the Evaluation Module Development Guide, an example of an evaluation module, which defines usability evaluation, was formatted along the proposed evaluation module structure and was included in the guide (Hausen and Welzel, 1993b, part 2).

Circulation of the guides through ISO/IEC provides a world-wide audience which could not have otherwise been reached. It also increases the awareness of the concept of third

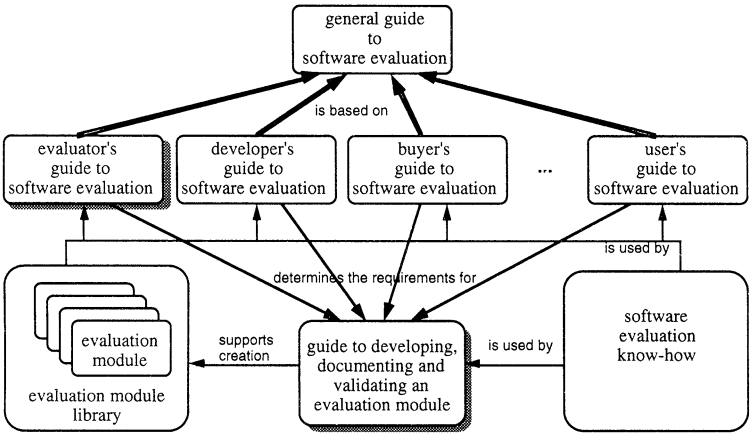

Figure 5 Guides supporting the Evaluation Method

party software evaluation and hopefully also the demand for this service similar to the situation of the ISO 9000 series of quality management standards.

10 PERSPECTIVES

The results supporting the evaluation process and the development of high quality evaluation modules, have been made available to both industry and academia. European as well as non-European industry have expressed much interest, especially in the method. Some small and medium enterprises as well as some large system integrators have adopted or are about to restructure their quality assurance with respect to the method developed. In addition, academia are considering other results, such as the way of modeling products, process and quality, for inclusion in educational circles and further research projects.

Based on the experience gained in the SCOPE project DELTA has implemented a commercial evaluation service, MicroScope, which is an instantiation of a subset of the Evaluator's Guide. The MicroScope approach (Kyster, 1993) follows closely the evaluation procedure as described, but a full set of evaluation modules is not yet available for commercial use. Therefore, commercial evaluations can only be offered for some combinations of quality characteristics and evaluation levels.

At the moment several organisations in Europe are setting up a network of testing laboratories offering harmonised software product evaluations according to ISO standards viz. the Evaluator's Guide. Feedback from the case studies showed that the software industry accepted the software evaluation concept. Furthermore the need for a certification scheme based on an approach like the Evaluator's Guide has been expressed. Such a scheme is expected to be implemented in the near future.

The evaluation method itself is specified by production rules (Hausen and Welzel,

1993a). Such a (semi-) formal description improves the possibility of using the computer itself to automate (or to assist a human in performing) some of the tasks associated with the process. Therefore, in addition, a concept of an advisory system has been designed (Hausen, 1992).

The rule-base specification allows a translation to PROLOG predicates in order to validate the evaluation method itself and to run a software evaluation. For efficiency reasons this might be implemented into an object-oriented software engineering data base system, such as the European Portable Common Tool Environment PCTE (ECMA standard 149, 1991). A feasibility study has shown that this can be achieved.

The usage of the evaluation method within a software development project is more highlighted in (Welzel and Hausen, 1994) and (Welzel, 1993). The effects to the organisational regulations of the project as well as of the whole company are still under research. The installation of the method has already contributed to process improvement (business re-engineering).

For specifying the Quality of Service (QoS) - network service - the techniques of this evaluation methods has been applied. The quality model for QoS uses characteristics of ISO/IEC 9126 and standards valid for network service (Bogen, Hausen, Worst, 1994).

REFERENCES

Basili, V.R. and Rombach, H.D. (1988), The TAME Project: Towards Improvement-Oriented Software Environments. *IEEE Transactions on Software Engineering*, 14/6, 758-73.

Begh, J., Hausen, H.-L. and Welzel, D. (1993) A Practitioners Guide to Evaluation of Software. *Proceedings of the IEEE Software Engineering Standards Symposium*, September 1993, IEEE Computer Society, p. 282-8.

Bogen, M.; Hausen, H.-L. and Worst, R. (1994) Handling of QoS Characteristics. *Computer Networks for Research in Europe*, a supplement to Computer Networks and ISDN Systems, Volume 26 (1994), Supplement 2,3, pp. 107-18, Elsevier Science Publishers B.V.

ECMA standard 149 (1991). *Portable Common Tool Environment* (PCTE), Abstract Specification, European Computer Manufacturers Association.

Hausen, H.-L. (1989) Yet Another Quality and Productivity Modeling | YAQUAPMO |. *ACM, IEEE, HICSS-22, Hawaii International, Conference on System Sciences* (ed. B.C. Shriver), Hawaii, January 1989, p. 978-987.

Hausen, H.-L. (1992) A Specification of an Assessment and Certification Advisor.*ACM, Annual Conference of the ACM* (ed. J.P. Agrawal, V. Kumar, V. Wallentine), March 1992, Kansas City, MO, 20 p.

Hausen, H.-L. and Welzel, D. (1993a) A Rule-Based Specification of Software Evaluation and Certification - Formal Model -. *SCOPE report*, SC.93/019, Version 02, GMD Sankt Augustin, May 1993.

Hausen, H.-L. and Welzel, D. (1993b) Guides to Software Evaluation (comprising: The Evaluators Guide, The Evaluation Module Development Guide). *Arbeitspapiere der GMD*, No. 746, GMD Sankt Augustin, April 1993.

Hausen, H.-L. and Welzel, D. (1994) Evaluating Software concurrently with and after its Developement. in: *Proceedings of the 11th International Conference on Testing Com-*

puter Software, Washington, DC, June 13-16, 1994.

IEEE 610, IEEE standard 610.12-1990, (1990). *IEEE Standard Glossary of Software Engineering Terminology*, Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017, USA, December 1990.

ISO/IEC Guide 25 (1990). *General Requirements for the Competence of Calibration and Testing Laboratories*, International Standards Organization, International Electrotechnical Commission.

ISO/IEC 8402, International standard, (1990). *Quality Concepts and Terminology* Part One: Generic Terms and Definitions, International Organization for Standardization, International Electrotechnical Commission, December 1990.

ISO/IEC 9126, International standard, (1991). Information technology - Software product evaluation - Quality characteristics and guidelines for their use, International Organization for Standardization, International Electrotechnical Commission.

ISO/IEC 9126: Guides to software evaluation, International standards (1993). part 5: The evaluator's guide, ISO/IEC JTC1/SC7 N1136, July 20, 1993, part 7: Guide to Developing, Documenting and Validating Evaluation Modules, *ISO/IEC JTC1/SC7/WG6 Technical report*, July 26, 1993, International Organization for Standardization, International Electrotechnical Commission.

Neusser, H.-J. and Hausen, H.-L. (1989). Knowledge Based Handling of Methods and Tools, *ACM, IEEE, HICSS-22, Hawaii International, Conference on System Sciences* (ed. B.C. Shriver, B. C.), Hawaii, January 1989, p. 142-51.

Kyster, H. (1993). MicroScope: The Evaluation of Software Quality, DELTA, Venlighedsvej 4, 2970 Hoersholm, Denmark.

SCOPE Consortium (1993). SCOPE Technology Report, *SCOPE report*, SC.93/009, Version 03, GMD Sankt Augustin, May 1993 (revised July 1993).

Welzel, D. (1993). A rule-based process representation for software process evaluation, in: *Information and Software Technology*, Volume 35, Number 10, pp. 603-10, Butterworth Heineman, October 1993.

APPENDIX: TERMINOLOGY USED

The terminology used based on (IEEE 610, 1990) and (ISO 8402, 1990). The following terms are mentioned in order to extend and classify the terminology.

- assessment of software: Process of comparing the values obtained from the measurements with quality requirements.
- classified software: Software which is classified according to product, process and supportive information or other characteristics.
- client: Person or institution (e.g. producer, distributor, buyer, or user) who requests/ negotiates the evaluation.
- evaluation module Encapsulation of the definition of an evaluation (sub-) method applied on a product or process information in order to measure software characteristics or subcharacteristics by applying metrics, checking pass/fail criteria, delivering evaluation report and cost report.
- evaluation level: 1. Grade which is defined by a set of evaluation techniques to be applied and the thresholds of quality metrics being obtained by these techniques.

2. Identification of (subcharacteristics and) metrics and attachment of metrics to subcharacteristics and definition of acceptance criteria by selecting rating levels for each metric and reference to (sub-) evaluation method to be applied to obtain a metric.

- evaluation report: Final document of the software evaluation. It is progressively completed during the whole evaluation process and consists of four parts: - evaluation requirement, - evaluation specification, - evaluation plan, and - evaluation result.
- evaluation item: Entity being evaluated.
- identified software: Software which is identified by document identifier, title, condition, and of date of arrival as well as handling information.
- measurement: Application of a metric for product quality or process productivity.
- process information: Entities obtained during the software process.
- product information: Entities constituting a complete or part of a software product.
- software evaluation: Process which comprises validation and verification, measurement and assessment of software.
- supportive information: Entities which are not evaluated but which are necessary for an evaluation.

Authors

Dieter Welzel
received his degree in Computer Science from the University of Bonn. Since 1990 he has been working at GMD Institute for Application-Oriented Software and System Technology, Sankt Augustin. His main activities involve interface notions for concurrent systems with modular structure, process modeling and evaluation of software processes.
E-mail: welzel@gmd.de

Hans-Ludwig Hausen
received degrees in Electrical Engineering and in Computer Science from Technical University of Berlin, where he also served as a lecturer in Computer Science. At present he holds the position of a senior scientist at GMD Institute for Application-Oriented Software and System Technology, Sankt Augustin. His main interest are computer aided software engineering and software quality assurance.
E-mail: hausen@gmd.de