CHAPTER 7

AN APPROACH FOR EVALUATING SOFTWARE QUALITY

Dr. Alaa Mohamed Fahmy

ABSTRACT

In order to takle the increasingly important issue of software quality, ISO/IEC 9162 was developed. This international standard sets out six quality characteristics, which are intended to be exhaustive. In order to successfully and practically apply ISO/IEC 9162, an evaluation framework is proposed in this paper. It is designed to allow technology changes without needing modification of its basic principles. The proposed framework is a four-step procedure for the planning, controlling and reporting on software evaluation. These steps are specifying of the evaluation requirements, evaluation design, conducting of the evaluation and reporting on the results achieved during the evaluation.

1. INTRODUCTION

Significant gains in the quality of software will not take place until there is a comprehensive model of software product quality available. Several different models of software product quality have been proposed [1-6]. While these models offer interesting insights into various apects of software quality they have been strong enough to stimulate siginificant gains in the quality of software or to gain wide acceptance.

Most recently the international standard ISO/IEC-9126 Software Product Evaluation Characteristics (1991) [7] has been put forward as a high-level framework for characterizing software product quality. ISO is the abbreviation for International Organization for Standardization and IEC is the abbreviation for International Electrotechnical Commission. This standard appears to have drawn considerably on the model originally proposed by Boehm [1]. While this standard can provide high-level guidance, it does not go nearly far enough to support building quality into

software. ISO/IEC-9162 is the software product evaluation standard from ISO. It defines six characteristics that describe with minimal overlap, software quality. It Provides the definition of the characteristics and associated quality evaluation process to be used when specifying the requirements for and evaluating the quality of software products thoughout their life cycle [7].

There is a wealth of knowledge about software quality available [8-17]. The greatest challenge in proposing any model for software product quality is to find a framework that can accommodate this knowledge in a way that is constructive, refinable, and intellectually manageable. The prime requirement of any such model is that it makes clear and direct links between high-level quality attributes and explicit product characteristics at all levels. Beyond this the model must provide:

* Systematic guidance for building quality into software.
* A means to systematically identify/classify software characteristics and quality defects.
* A structure that is understandable at a number of levels, refinable and adaptable.

When carrying out a software products evaluation, clients require reliable results which can be used as a sales aid. An evaluation must be of value to many parties of the software industry, including producers, vendors, users and to the community at large. This means that an evaluation procedure needs to be pertinent, widely applicable and cost-effective. By pertinent is meant that the properties chosen to be evaluated are meaningful to the clients.This means that the purpose of an evaluation, for example in the context of the functionality characteristics, is to assure that the computer program is a trusted implementation of the design specification, which in turn is a trusted transformation of the requirements specification. In other words, the purpose of the evaluation is to answer the question "Is the claimed functionality really available"?. This objective might be difficult to reach, especially taking into account the diverging interests of the client or even the user of the software product.

What kind of guidelines should be provided by the Evaluator's guide? This can be answered by asking the questions "What happens when a problem occurs during an evaluation"? and "What type of problems can occur"?. A missing documents is identified in the evaluation specification. A delayed document leads to a postponement of the respective evaluation activities. If there is a change of product representation, a new evaluation specification has to developed or an existing one has to be modified where applicable (e.g. change of programming language might impose a change of some or all evaluation techniques, which might in turn have an impact on the cost of evaluations). Other problems might be treated in a similar way.

2. ISO/IEC-9126 SOFTWARE QUALITY CHARACTERISTICS

ISO/IEC-9126 standard defines six characteristics that describe, with minimal overlap, software quality. These characteristics are **functionality, reliability, usability, efficiency, maintainability and portability.**

Specifying software quality for a product that has still to be developed is difficult for the purchaser or supplier. The purchaser needs to understand clearly and be able to communicate his/her requirements for the product to be developed. The supplier needs to be certain he/she understands the requirements and is able to assess with confidence whether it is possible to provide the product with the right level of software quality.

Consequently, ISO/IEC-9126 will serve to eliminate any misunderstanding between purchaser and supplier. This improvement in communication will do away with any rework required as a result of the software product not meeting the purchaser's requirements. Both the times taken to deliver the specified software product and the cost of development will be lower as a result of conformance to the ISO/IEC-9126 standard.

Functionality is the set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs. **Reliability** is the 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** is the set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users. **Efficiency** is the 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** is the set of attributes that bear on the effort needed to make specified modifications. **Portability** is the set of attributes that bear on the ability of software to be transferred from one environment to another.

This decomposition of quality reflects the user's view and introduces the concept of quality in use. Users are mainly interested in using the software product, and evaluate software mostly from the viewpoint of the performance and the service it provides, rather than on the basis of internal aspects of the development process. The process of development requires the user and the developer to use same software quality characteristics since they apply to requirement and acceptance. When developing off-the-shelf software, the implied needs must be reflected in the quality requirement.

Since developers are responsible for producing software which will satisfy quality requirements, they are interested in the intermediate product quality as will as in the final product quality. In order to evaluate the intermediate product quality at each phase of the development cycle, the developers have to use different metrics for the same characteristic because the same metrics are applicable to all phases of the cycle. A manager is typically more interested in the overall quality rather than in a specific quality characteristic, and for this reason will need to assign weights, reflecting business requirements to the individual characteristics. The manager may also need to balance the quality improvement with management criteria such as schedule delay or cost overrun, because he/she wishes to optimize quality within limited cost, human resources and time-frame.

ISO/IEC-9126 suggest a further decomposition of each characteristic into a set of subcharacteristics. These sub- characteristics are a step closer to the quantitative, technical aspects of the software product. Figure1, shows these sub-characteristics.

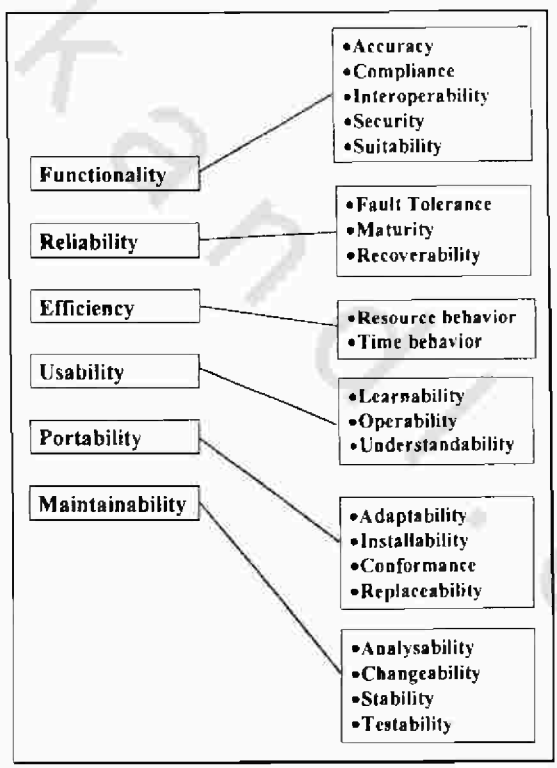

Figure 1 : sub-characteristics of ISO/IEC-9126 standards

As shown in figure 1, these sub-characteristics are accuracy, compliance, interoperability, security, and suitability for functionality. Reliability sub-characteristics are fault tolerance, maturity and recoverability. Efficiency sub-characteristics are resource behavior and time behavior. Usability sub-characteristics are learnablility, operability and understandability. Protability sub-caracteristics are adaptability, installability, conformance and replaceability. Maintainability sub-characteristics are analysability, changeability, stability and testability.

Suitability is an attribute of software that bears on the presence and appropriateness of a set of functions for specified tasks. Accuracy is an attribute of software that bear on the provision of right or agreed upon results or effects. Interoperability is an attribute of software that bear on its ability to interact with specified systems. Compliance is an attribute of software that make the software adhere to application related standard or conversions or regulations in laws and similar prescriptions. *Security* is attributes of software that bear on its ability to prevent unauthorized access, whether accidental or deliberate, to programs and data.

Maturity is an attribute of software that bear on the frequency of failure by faults in the software. *Fault tolerance* is an attribute of software that bear on its ability to maintain a specified level of performance in cases of software faults or of infringements of specified interface. *Recoverability* is an attribute of software that bear on the capability to re-establish its level of performance and recover the data directly affected in case of a failure and on the time and effort needed for it.

Time behavior is an attribute of software that bear on response and processing times and on throughout rates in performing its function. *Resource behavior* is an attribute of software that bear on the amount of resources used and the duration of such use in performing its functions.

Understandability is an attribute of software that bear on the user's efforts for recognizing the logical concept and its applicability. *Learnability* is an attribute of software that bear on the user's effort for learning its application (for example, operation control, input, output).*Operability* is an attribute of software that bear on the user's effort for operation and operation control.

Adaptability is an attribute of software that bear on the opportunity for its adaptation to different specified environment without applying other actions than those provided for this purpose for the software considered. *Installability* is an attribute of software that bear on the effort needed to install the software in a specified environment. *Conformance* is an attribute of software that make the software adhere to standards or conventions relating to portability. *Replacebility* is an attribute of software that bear on the opportunity and effort of using it in the place of specified other software in the environment of that software.

Analysability is an attribute of software that bear on the effort needed for diagnosis of deficiencies or causes of failures, or for identification of parts to be modified. *Changeability* is an attribute of software that bear on the effort needed for modification, fault removal or for environmental change. *Stability* is an attribute of software that bear on the risk of unexpected effect of modification. *Testability* is an attribute of software that bear on the effort needed for validating the modified software.

3. EVALUATION FRAMEWORK ACTIVITIES AND DOCUMENTS

The purpose of a software product evaluation is to provide any interested party with quantitative results concerning a software product that are comprehensible, acceptable and trustworthy. This paper presents a procedure, which defines the activities of analysing the evaluation requirements, specifying, designing and performing the evaluation, and reporting on the results of the evaluation. Figure 2, shows the main activities and documents within the proposed evaluation framework.

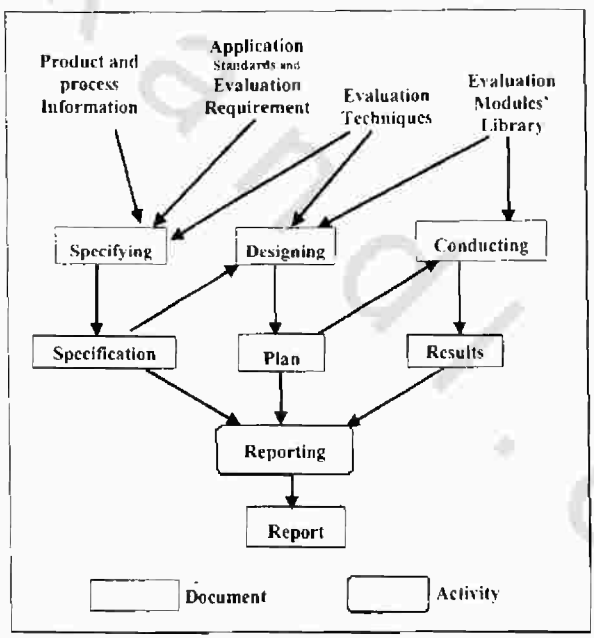

Figure 2 : The Main Activities and Documents within the Proposed Evaluation Framework.

The aim of this evaluation framework is to assist testing laboratories when providing the evaluation services, software producers in planning the evaluation, software customers when specifying certification of the evaluation in their contracts

with their clients, and software users when studying and understanding the results. Evaluation is conducted through a well-defined, step-wise procedure which, to achieve flexibility, is based on a library of evaluation modules. The standardisation of these modules is beyond the scope of this paper.

The procedure is widely applicable for evaluating a software product, i.e. is capable for dealing with any type of software product ranging form off-the-shelf packages developed for a general customer base, through projects commissioned by a customer, to embedded software. This is obtained by using a level approach to quality characteristics and providing a flexible view on classifying product and process information.

The cost-effectiveness of an evaluation procedure is related to the level of confidence required and this level will vary with the critical aspects of the software product usage and the reason for conducting an evaluation. Guidance is provided on how to make an estimate of the costs of evaluation as early as possible in the procedure.

The procedure complies with reqirements stated in ISO/IEC guide 25, which provides general guidance for the operation of testing laboratories and lays down rules for creating a framework for the evaluation of all types of products including software products.

Conformancy testing is not defined in this paper but can be applied if a normative quality model, identifying quality characteristics, metrics and evaluation techniques, is given.

4. OVERVIEW OF THE EVALUATION PROCEDURE

The evaluation procedure presented comprises four steps, which divide the evaluation into distinguished activities as shown in figure 2. Specifying the evaluation finds out the conditions and constraints of the evaluation and defines the objectives in addition to the formal description of the evaluation requirements, consists of the identification of available documents and classification of items received into product, process and (for the evaluation process) supportive information in order to identify which evaluation needs are to be performed. The evaluation techniques are specified in order to determine the information required and identify the items which are missing. They are formal records of the agreement between client and test laboratory of what has to be achieved by the evaluation process. They provide a nominal lists of software quality characteristics to be evaluated at a specific evaluation level and identifies the source of data and evidence to be used in the evaluation process.

In the step designing the evaluation process, evaluation modules which define the applicable evaluation techniques are identified and linked to the respective characteristics. The selection and ordering of evaluation modules and the estimation of costs complete the evaluation design. Performing the evaluation by the testing laboratory, then comprises the application of the set of optimised evaluation modules on the related documents and collecting for each of them the results.

The last step in the evaluation procedure is the reporting of all results achieved during the evaluation requirement analysis, specification, design and performing. The evaluation procedure considers the following basic types of information shown in figure 3:

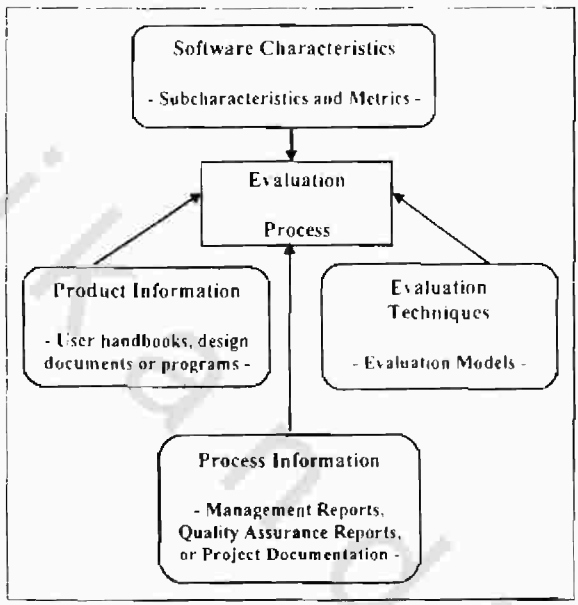

Figure 3 :Information Needed for Software Evaluation

*** Software Characteristics**

Set of software characteristics, which are of interest and should be evaluated. A break down into subcharacteristics and metrics may be helpful for comprehension.

*** product Information**

Including documents such as user handbook, documents, or code.

*** Process Information**

Including document such as management report, quality assurance report, or project file.

*** Evaluation Techniques**

Set of evaluation techiques, encapsulated in evaluation modules, which are to be used in order to evaluate the software product for a software characteristic.

4.1 Specifying the Evaluation :

An evaluation specification is derived by negotiation and agreement with the client of the evaluation and from available information about the software product, i.e. product description, requirement specifications, user manual, etc. Also references to application specific standards and guidelines can be included.

The evaluation specification shall state unambiguously the quality attributes of the software product (functional and non-functional) to be evaluated. The quality attributes of the software product shall be categorised according to the software characteristics given by ISO/IEC-9126. Furthermore, for each quality attribute the required value shall be specified. The list of attributes will be possibly modified by the findings of the product analysis. This list represents the necessary compromise between the characteristics of the product the client feels it is important to evaluate, and a more comprehensive list of characteristics that an evaluator might feel to be the appropriate set of attributes which should be evaluated.

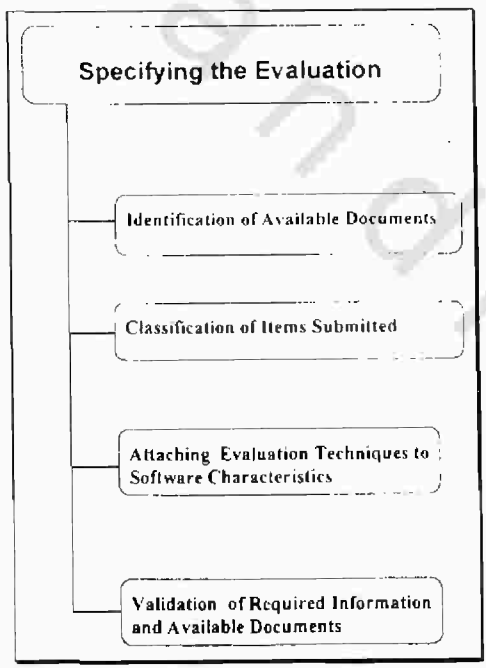

Figure 4 : Activities of specifying the Evaluation.

123

As shown in figure 4, specifying the evaluation consists of the following activities:

- Identification of available documents.
- Classification of items submitted.
- Attaching evaluation techniques to software characterictics.
- Validation of required information and available documents.

The evaluation specifications need not be concerned with whether any attribute can or cannot be measured, i.e. whether particular evaluation modules are available.

Neither should the specification be totally influenced by the wishes of the client expressed in the evaluation requirement. If, at the product analysis stage, some aspect of the product is suspect and is deemed necessary to perform an investigation, then this should be included in the evaluation specification. It will, of course, still be the right of the client to withdraw the product from evaluation if the more extensive evaluation is not agreed.

4.2. Designing the Evaluation :

The aim of designing the evaluation is to produce an evaluation plan by specifying the set of evaluation modules to be used according to the evaluation specification. It describes the selection procedure of evaluation modules, the adaptation of the informtion classified to the information used by the selected modules, the planning of the evaluation, as well as the cost of evalution. The clause concludes with a description of the interaction with legal procedures. Figure 5, shows the activities of designing the evaluation.

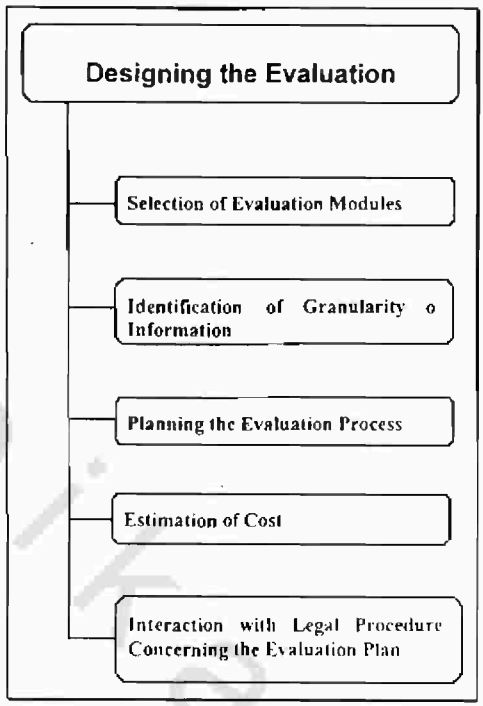

Figure 5: Activities of Designing the Evaluation

After determining the evaluation techniques, that is, identifying which software characteristics should be considered and how thoroughly, and after identifying the technical constraints of the product (like product type, design or program language), a set of applicable evaluation modules are selected from an evaluation module library.

The selection process requires that the evaluation module library be searched for evaluation modules, which will be useful in the evaluation of each attribute for the target product. This directly implies two criteria to be applied in the selection of evaluation modules : **First,** the module must be known and recognized to be useful in the evaluation of the attribute for which it is to be used for. **Secondly,** the module must be applicable to the product part on which it is to be used. For example, many 3GL software metrics cannot be applied to object oriented or rule based software.

However, this set of evaluation modules may not be optimal for carrying out the evaluation. Some modules may be redundant, some other modules may be missing. It must be decided whether new modules must be developed or whether missing

modules can be substitiuted by a combination. The planning will be done in order to optimize the coverage of evaluation of carrying through the evaluation.

To reduce the set of applicable modules to a minimal set of modules needed for the evaluation, an equivalence relation might be introduced on the set of (all) modules in order to use representative modules which minimize the cost of the evaluation with respect to the applicable evaluation techniques and tools.

In the optimization step, representatives of the applicable modules are selected. Each testing laboratory can use those modules which have lowest cost. With the results of the application of modules, an evaluation statement for the software product can be made.

Figure 6, shows the selection procedure. With information of the evatluation specification, the evaluation items and the evaluation module library, a set of applicable evaluation modules can be identified. In the step "Minimization of Evaluattion Modules" a set of representatives of equivalence classes selected according to the specific constrains of the testing laboratory and to cost-effectiveness.

The optimized set of evaluation modules requires product, process and perhaps supportive information as imposed by the input interface of the evaluation modules. A refinement and adaptation step might be necessary, as shown in figure 7, to relate the information needed by the modules to the evaluation items identified by the information classification.

Planning the evaluation process includes the usual activities to be performed at the beginning of and during a project which are:

- Identifying the evaluation activities.
- Identifying resources (human, tools,... etc.).
- Allocating resources to activities.
- Scheduling and reporting progress.

The evaluation plan including the list of evaluation modules to be applied. Each module includes information from which the cost of its application can be derived. Hence, it is easy to calculate the total cost of the evaluation. However, the testing laboratories act in a competitive market, and therefore the actual price of the evaluation may differ substantially from the calculated cost. Furthermore, in some cases, the cost may also include the whole, or part, of the development cost of a new module.

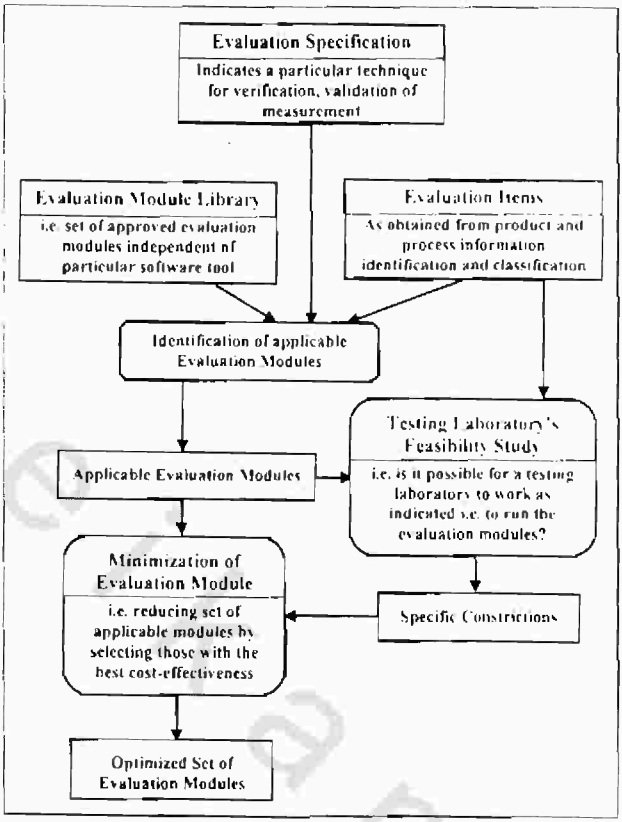

Figure 6 : Selection of Evaluation Modules.

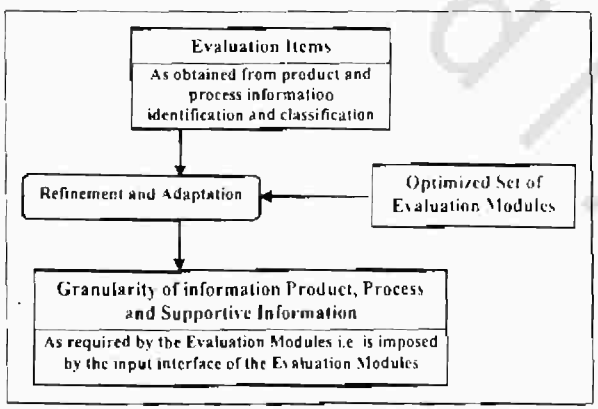

Figure 7: Identification of Granularity of Information .

A first rough estimation of costs is obtained based on the evaluation levels. Here a refinement of the estimation based on the selection of evaluation modules is considered. Each module includes data from which the cost of its application can be derived. From this information the cost of implementing the evaluation plan is estimated. Costs might be divided into cost of personnel and cost of applying tools.

In the interaction with legal procedures concerning the evaluation plan, the client requests the production of an evaluation plan. This act constitutes the client's acceptance of the testing laboratory's offer to draw up the evaluation plan subject to the terms and conditions specified in the evaluation specification. The evaluation plan may restate many of the provisions of the evaluation specification but will also:

- Identify specific evaluation modules to be applied.
- Prescribe a timetable for the evaluation process.
- Provide for logistics of the evaluation process.

The testing laboratory may offer to conduct an evaluation according to the evaluation plan.

4.3. Conducting the Evaluation :

The implementation of the evaluation plan means applying the evaluation modules on the related product parts and collecting for each of them the application results. The output will be a collection of measurement reports resulting from the applicatition of the modules. The execution of the evaluation modules and the interpretation of the results migh be supported by a testing laboratory's workbench, assistant or advisor.

Measurements can be manual, computer aided (e.g., using a check list manager for applying check lists), or automatic (e. g. measuring the 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 (measurement data) about the measured product part that could be helpful for an acceptance decision to be taken. Figure 8, shows the activities to be conducted by the evaluation conduction.

Based on the results of evaluation modules, a synthesis of all measurement reports might be necessary. This can be done by one of the following two possibilities :

- Application of so called "Assessment Modules" where each of them defines a multidimensional assessment method.
- "Summarizing" output of all evaluation modules by :

- Identifcation of a list of "metrics" applied (in case of checklists, it is the collection of questions applied).
- "Achievement inforamtion" i.e. coverage of checklists.
- Rating the results i.e. mapping onto the categories excellent, good, fair, poor (or others).

The actual results are to be compared with the required results specified in the evaluation specification. An evaluation conclusion shall be done according the pass/fail decision criteria- if they are defined. If the decision is pass the evaluation specifications and the actual resuls should be only repeated. If the decision is fail, the measurement result that drove this decision should be highlighted.

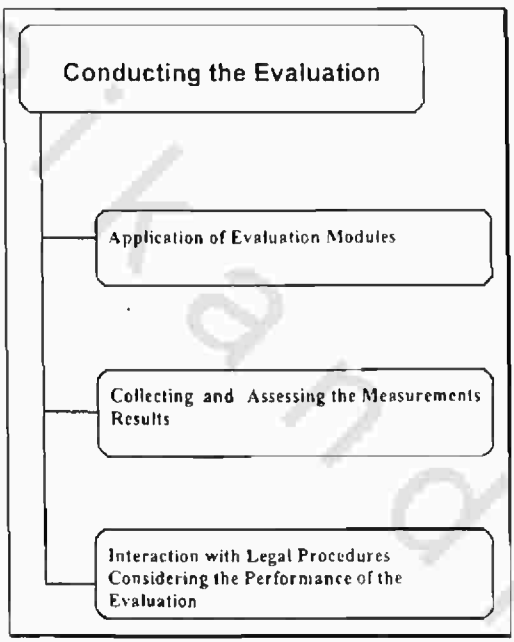

Figure 8 : The Actvities of Conducting the Evaluation.

In most cases, the evaluation may be expected to follow provisions and procedures laid down in the evaluation specification and plan. From the legal standpoint, the attempt must be made to identify and provide for potential problems. The most significant problem would be that the evaluation plan cannot be implemented. This may be caused by a number of factors of circumstances including the possibilities that :

- Deficiencies appear in the evaluation plan.
- The software may not be as specified.
- Third party intervention may prevent performance.

4.4. Reporting the Evaluation :

The last activity of the evaluation process is to collect all results, document the process and structure them in the evaluation report. The evaluation report compiles all outputs from the individual modules, including all observations made during the evaluation process, into the evaluation results, which are a part of the evaluation report.

The evaluation shall be reported accurately, clearly, unambiguously and objectively, as required by ISO/IEC Guide [11]. This includes all the information necessary for the interpretation of the evaluation results and all information required by the method used in order to be able to control application of evaluation procedure and the suitability of decisions taken.

5. STRUCTURE OF THE EVALUATION REPORT

A proposed structure of the evaluation report including its items is shown in Figure 9.

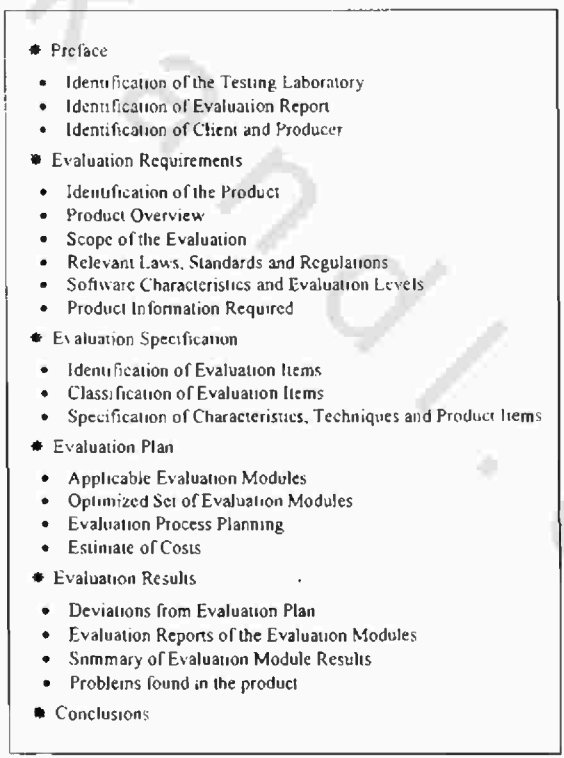

Figure 9 : The Proposed Stucture of the Evaluation Report.

6. CONCLUSION

The paper provides a description of a software evaluation procedure. This evaluation procedure is a four-step procedure. The description is presented in terms of actions and results in order to cover the practical issues of planning, controlling and reporting on an evaluation.

The evaluation procedure proposed is adaptable to circumstances of any testing entity (available personnel, evaluation techniques and tools), and flexible to relevant standards of product type dependent quality requirements and of software engineering processes, and to legal or contract issues.

The evaluation framework proposed is designed to allow technology changes without needing modification of its basic principles. This is a consequence of the approach taken to use evaluation modules. Although, the evaluation framwork explicitly refers to ISO/IEC- 9126, other quality models can be applied without any serious change to the evaluation procedure.

Running a product evaluation provides information that can help to estimate the costs, improve evalution modules, or to identify needs for new evaluation techniques. The experience gained, therefore, should be stored in a database for the benefit of future evaluations.

One prime objective for developing this evaluation framework was to ensure that an evaluation process is effective. In order to achieve this goal, many issues have been considered, **first;** experience with evaluation modules and the module library, **second;** appropriateness of levels and software characteristics, **third;** appropriateness of product and process representation, **fourth;** calculation of actual costs in order to improve cost estimates, and appropriateness of the evaluation procedure.

REFERENCES

[1] B. W. Kernighan, and P. J. Plaugher; "The Elements of Programming Style", McGraw-Hill, N. Y. (1974).

[2] T. P. Bowen; "Specification of Software Quality Attributes", Rome Laboratory, New York, Technical Report, RADC-TR- 85-37. Vols. 1-3, 1976.

[3] B. W. Boehm, J. R. Brown, M. Lipow, G. J. Macleod, and M. J. Merritt; "Characteristics of Software Quality", North-Holland, Amesterdam, 1978.

[4] B. Kitchenham; "Towards a Constructive Quality Model", Soft. Eng. Journal, 105-112, July 1987.

[5] M. Deutsch, R. Willis; "Software Quality Engineering", Prentice-Hall, Englewood - Cliffs, NT: 1988.

[6] R. Nance, "Software Quality Indicators: An Holistic Approach to Measurement", Proc. 4th Annual Software Quality Workshop, Alexandria Bay, New York, August 1992.

[7] ISO/IEC Standard, ISO-9126 Software Product Evaluation - Quality Characteristics and Guidelines for their Use, 1991.

[8] M. C. Paulk, B. Curtis, M. Beth; "Capability Maturity Model for Software", Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, August 1991.

[9] H. L. Housen, N. Cacutalua, D. A. Welzel; "Method of Software Evaluation and Certification- Draft", Arbeitspapiere der GMD, No. 571, GMD Sankt Augustin, September 1991.

[10] IEEE Std 610. 12-1990, "IEEE Standard Glossary of Software Engineering Terminology", Institute of Electrical and Electronics Engineers, Inc., NY, December 1990.

[11] ISO/IEC Guide 25; "General Requirements for the Competence of Calibration and Testing Laboratories", International Standards Organization. International Electrotechnical Commission, 1990.

[12] International Standard ISO/CD 8402-1; "Quality Concepts and Terminology Part One: Generic Terms and Definitions", International Organization for Standardization. December 1990.

[13] Interantional Standard ISO 9000-3; "Quality Management and Quality Assurance Standards-Part 3: Guidelines for the application of ISO 9001 to the de-

velopment, supply and maintenance of software", International Organization for Standardization, 1991.

[14] International Standard ISO/IEC 9126; "Information Technology - Software Products Evaluation- Quality Characteristics and guidelines for their use", International Organization for standardization, Interanational Electrotechnical Commission, 1991.

[15] Working document for technical report of ISO/IEC JTCI/WG6; "Software Quality Evaluation Guide part 1: General Guide", by M. Azuma, Draft-Version 3, October 1992.

[16] I. J. Lloyd, and M. J. Simpson; "Fourth Report on the Legal Aspects of SCOPE" SCOPE document SC 92/003/STH.il.ms/T.2.5/.DT/01, 25th February 1992.

[17] SCOPE Consortium, edited by H. L. Hausen, N. Cacutalua, and D. Welzel' "A Method for Software Evaluation and Certification - The Information Model - Refinement of the Phase One Model", SCOPE report SC. 92/016/GMD. HCW/T2.1.1.2./RP/02, GMD Sankt Augustin, Germany, February 1992.