Getting The Measure Of TickIT

Guidance and information about the emerging ISO measurement standards for improving software processes and how they relate to ISO 9001:2000.

Getting the measure of TickIT

Guidance and information about the emerging ISO measurement standards for improving software processes and how they relate to ISO 9001:2000

Please activate the 'Navigation Panel' button on your Acrobat Viewer to see document structure and select sections of interest. The document also contains active links to web sites.

DISC TickIT Office
389 Chiswick High Road
London, W4 4AL

Tel: +44 (0) 20 8996 7427
Fax: +44 (0) 20 8996 7429

Contents

1. Introduction

An important new element in the 2000 issue of ISO 9001 is the requirement to measure processes, products and quality objectives. Measurement of processes is seen as a critical component of process improvement. Measurement of product helps to assure the quality of deliverables and in turn, also helps in the improvement of the processes that contribute to product realization.

The purpose of this guide is to help those organizations who want to take software measurements seriously whilst also addressing the needs of ISO 9001:2000. It has been produced by the BSI DISC TickIT Guidance committee, BRD/3/1, to promote understanding and constructive use of measurement tools. The guidelines are intended to benefit all organizations involved in software development - not just those that are TickIT-registered. However, if an organization already has a quality system in place for ISO 9001 and TickIT registration, this guidance can be used to build on the existing regime by means of a combination of experience, best practice and the emerging ISO/IEC standards for software engineering. A second purpose of this guide is to serve as an introduction to a much more detailed work entitled "Quantitative software management - Software measurement methods", which provides more detail for the standards and processes discussed here. In both documents, extensive use is made of other publications dealing with measurement of software, and these are referenced in the appendices.

The history of effective software measurement is not encouraging. The software engineering community, including organizations certificated under the TickIT scheme, has tended to regard measurement as an easily avoided diversion, if performed at all, and having little impact on quality improvement.

ISO 9001:1994 clause 4.20, Statistical techniques, only required the user to consider the requirements of process and product measures. As written in the 1994 standard, this clause was not particularly helpful, hard to enforce, and often ignored without too much risk of gaining a nonconformity. In the past where measurements were attempted, they were often restricted to counting the number of errors reported and possibly how long it takes to correct them, but that, generally, was as far as it went. Very rarely were these figures taken and used proactively to support improvement.

ISO 9001:2000 however, recognizes that measurements are important, vital in fact, if organizations are going to have any chance of assessing, monitoring and improving the quality of their product or service.

Compliance with all the standards identified below is not necessary for improvements to be achieved, although organizations involved in software may well find that, increasingly, compliance requirements are being written into certain types of software contract. TickIT is intended to support and provide confidence of software quality for acquirers, developers and support personnel, with additional guidance for assessors. This document therefore attempts to provide similar support for these different but complementary views of software measurement.

Appendix A describes some of the terminology used to monitor software processes and products. In conventional parlance, the term 'metric' applies to the method and scale, whereas 'measurement' involves the use of metrics to assign values. In general literature, however, the two terms tend to be used inter-changeably. In this document, 'measurement' is used as the general term except where a method or scale is specifically intended.

2. What should be measured and why?

Users should not be measuring just to satisfy the requirements of ISO 9001:2000 clauses 8.2.3 and 8.2.4, and thus auditors. Measurement needs to be done with clear objectives in mind, and the parameters chosen so that both quantitative and qualitative benefits may be obtained. One of the Quality Management Principles of ISO 9000:2000 is "a factual approach to decision making". Use of measurement techniques will help to address this principle.

First of all, it is important to assess what aspects of software can sensibly be measured, economically and with what benefit? Next, having obtained the measurements, how can they be interpreted and made use of? A measurement programme that shows no evidence of benefit will soon be discarded, or, possibly worse, continued by personnel with no clear understanding of what is required, simply because it has become proceduralized. This will waste time and effort and degrade the prime objectives of measurement.

2.1 Quantifiable benefits of making measurements

Some of the more immediate - and quantifiable - benefits of measurements are:

- validating the completeness of a requirements definition,
- identifying and measuring compliance of software requirements,
- identifying and measuring attainment of software design objectives,
- identifying and measuring completeness of software testing objectives,

Next column

- identifying user acceptance criteria for a completed software product,
- providing a measure of confidence in specific quality requirements of the product, for example, reliability and functionality.

2.2 Standards and guidance related to measurements

Help is at hand in the form of a number of standards currently being developed by ISO and IEC. In particular, ISO/IEC 9126 Software Engineering - Product quality, ISO/IEC 14598 Software engineering – Product evaluation and ISO/IEC 15939 Information technology – Software measurement process framework. At the time of writing - February 2002, only ISO/IEC 14598 is fully issued in its final form but sufficient material has now published to be constructively used within a measurement programme. The incomplete standards are sufficiently advanced to enable a workable process structure to be established. The target dates for full publication are as follows:

- ISO/IEC 9126, which will eventually comprise an International Standard, three Technical Reports and several guidance documents, should be completed in 2002
- ISO/IEC15939 should be available in 2002.

One of the purposes of this document is to introduce software developers, maintainers and users to the contents and use of these standards and to relate them to ISO 9001:2000, the TickIT Guide and good software engineering practice. Significant changes in the content of any of the referenced standards will be addressed in further updates of this document.

In the following sections, there is a brief summary of each of these standards. Other standards relevant to software measurements are described in Appendix C. The intent is to place these documents in a consistent software-orientated framework and to indicate to users those that are the most appropriate to their needs. Strictly speaking, none of these standards is vital to the generation and use of effective software measures, but detailed study of their content will add considerably to the understanding of the overall processes and details. This guide sets out a clear framework to encourage software managers and quality personnel to set up suitable processes; the standards themselves will complete the detail.

2.2.1 ISO/IEC 9126 Software Engineering - Product quality

This standard has undergone a major revision since the 1991 edition. Not only has it been expanded into four sections but the processes for handling measurement data have been transferred to another standard, ISO/IEC 14598, as explained below.

ISO/IEC 9126 deals with several important aspects of quality measurements. The first is the development framework and the relationships of product quality to the development lifecycle. Concepts of quality in software change as development progresses, for example, *Required Product Quality* is as specified in the user requirements, *Design Quality* relates to the product as built and *Quality-in-Use* deals with the product as seen by users. *Quality-in-Use* generally depends on external measurements, whereas *Design Quality* tends to be focused on internal parameters. The four principal parts of ISO/IEC 9126 deal with the following topics:

- *Part 1: Quality model* - provides an introduction to the concepts and lays out the framework for the other parts.
- *Part 2: External metrics* - deals with external measurements made when validating the software in a system environment.
- *Part 3: Internal metrics* - deals with internal measurements made during the development phases.
- *Part 4: Quality-in-use metrics* - deals with user perception and acceptance of the software.

Mappings between internal, external and quality-in-use measurements need to be defined by the developing organization; these will never be absolute, but can be developed into a usable set of quality parameters.

Product (that is software) quality can be measured internally by static measures such as compliance with coding standards and program structure. External measures usually involve measurement of the software in use or under test, for example, its performance, ability to recover from errors, and ease of use. So, if quality-in-use (what the customer sees) is taken as the prime objective, product quality built in during development becomes a direct contributor. Product quality in turn is directly influenced by the quality of development processes. This is the concept used in ISO/IEC 9126 to support the quality model.

Six characteristics go into describing the overall quality model in ISO/IEC 9126:

- *Functionality* - the features the product delivers,
- *Usability* - how easy and effective operators find the product in use,
- *Reliability* - the confidence that may be held in the product's continued and accurate operation,
- *Portability* - whether or not the software can be effectively ported to another environment,

Next column

- *Efficiency* - an assessment by some defined criteria as to whether the software makes adequate use of resources,
- *Maintainability* - how easy the software is to update or modify.

Each of these characteristics is further divided into sub-characteristics, for example, efficiency comprises time behaviour and resource utilization, and reliability covers maturity, fault tolerance and recoverability. Obviously, not all characteristics will be applicable in all situations and once again it is up to the developer, user, acquirer, or possibly evaluator to consider what measures affect the quality-in-use criteria.

Quality-in-use characteristics include the following four topics:

- *Effectiveness* - enabling users to achieve specified goals with accuracy and completeness,
- *Productivity* - the expenditure of resources against benefits achieved,
- *Safety* - levels of risk of harm to people, business, property or the environment,
- *Satisfaction* - the capability of the software to satisfy user needs.

2.2.2 ISO/IEC 14598 Information technology - Software product evaluation

Products, even software products, are only part of the story. It is also necessary to consider measures of the processes used to design, build, test and generally control the product. Here, ISO/IEC 14598, in conjunction with ISO/IEC 12207 Information technology - Software life-cycle processes and ISO/IEC TR 15504 Information technology - Software process assessment, are the most relevant. For software products, ISO/IEC TR 15504 identifies those process-related measurements and activities required to provide quality.

The standard is in six parts and provides guidance from the perspectives of developer, acquirer and evaluator:

- *Part 1: General overview*
- *Part 2: Planning and management*

  This covers the development of measurement plans and the planning and development of evaluation activities.
- *Part 3: Process for developers*

  This covers planning and evaluation of internal and external measures for ensuring product quality is incorporated at the development stage.
- *Part 4: Process for acquirers*

  This covers aspects of quality of use to purchasers and users of software products.

Next column

- *Part 5: Process for evaluators*

  This covers the establishment of evaluation requirements, the specification, design and conduct of the evaluation activities and finally the evaluation report.
- *Part 6: Documentation and evaluation modules*

  This deals with the packaging of quality aspects dealing with specific module evaluation techniques. Also, it allows the complexity of dealing with measurement assessments to be managed effectively.

  So, for example, a measurement technique as described below, say requirements traceability, is identified and scoped, inputs and analysis are defined, and the report format is described.

The six parts provide a general framework plus supporting information. The contents of this standard are extended further later in this document.

There is another project, SQuaRE - Software product Quality Requirements and Evaluation, which is at a very early stage. This aims to consolidate the ISO/IEC 9126 and ISO/IEC 14598 standards with additional framework and guidance material.

2.2.3 ISO/IEC 15939 Information technology - Software measurement process framework

This identifies the essential requirements for a measurement programme within a process improvement activity, as defined in ISO/IEC TR 15504, using the terminology of ISO/IEC 12207. It describes an iterative improvement loop with four key process areas:

- establishing and maintaining management commitment,
- planning the measurement processes,
- performing the measurement processes,
- evaluating measurements.

Gradually, one can begin to see to see how all the various software engineering standards under development fit together and can be utilized to form a coherent measurement strategy.

However, ISO standards are not the only potential source of information. The Application of Metrics in Industry (AMI) guide, for example, provides valuable assistance in devising a measurement programme. The Software Engineering Institute have a series of measurement handbooks and another set of guides, Practical Software Measurement (PSM), sponsored by the US DoD are also available. Further details are provided in Appendix D.

Now that these tools are available, it is possible to start to construct a measurement and improvement programme that is both compatible and compliant with TickIT and ISO 9001:2000.

3. Developing and applying a measurement programme within an ISO 9001-compliant quality management system.

This section deals with the general criteria for a measurement programme and how this can relate to specific ISO 9001:2000 clauses. Subsequent sections explain the methods and terminology derived from the standards in more detail.

3.1 Measurements and quality objectives

ISO 9001:2000 requires that quality objectives shall be measurable - this is the first basic requirement for an organization. Measurement issues at this level may or may not relate to specific software characteristics - other corporate goals are equally applicable. However, whatever product and process measurements are applied, they should be compatible with and lend support to the quality objectives. The Quality Policy needs to provide a suitable framework for the objectives to be established and reviewed, and also needs to identify management commitment for compliance and improvement. In other words, for the commitment to improvement to be real, relevant objectives for a software organization, including software quality, must exist and be measured.

Taking measures from every project may not be the most appropriate approach - at least initially. It is better to focus on those activities that can be shown to be potential beneficiaries of the measurement process, where the gains can be quantified against the additional expenditure made.

3.2 Measurement planning

It is vital that a measurement programme enjoys the full support of senior management with clearly identified budgets and personnel time allocated. It is not sufficient just to state this as policy; it also needs to be demonstrated in practice. Management commitment and the provision of adequate resources are key requirements of ISO 9001:2000.

3.2.1 What needs to be covered

In order to demonstrate a coordinated approach, there should be a high level plan dealing with software measures – referred to here as a Measurement Plan -

Next column

which in turn is supported and referenced by project or other quality plans as appropriate.

The planning of measurement activities should be evident in development projects and support activities. Quality plans for such activities should identify not only what measures are to be taken - consistent with the organizations objectives - at what stages and how, but also how they are collated and fed back into an improvement process. It should therefore be evident from the plan that the measures collected are part of this framework and provide some tangible benefit - not just collected because they seem to be the easiest measures to make.

Aspects covered by the Measurement Plan would typically be:

- the rationale for the measurement activities and what types of project are to be addressed, (for example new developments, upgrades or support activities) - there must be a clear overall purpose for conducting a measurement programme, which must also relate to the organizational quality objectives, and a senior management sponsor should be identified,
- an overview of all the measurement methods to be employed, together with collation activities - this can be a combination of review, test, assessment, report feedback as well as statistical relationships,
- any general normalization methods to be adopted - there should also be some reference to the measurement scales employed (see Appendix A),
- the overall framework within which each measure fits - this would typically address the internal, external and quality-in-use measures discussed in ISO/IEC 9126 and there should be clear and demonstrable correlation between these different parameters, with limitations of the approach identified (however, the characteristics identified in ISO/IEC 9126 should not necessarily be considered definitive for all situations - some may not be relevant for all products, or additional ones could be introduced),
- the identification of an improvement programme including planned reports and reviews - this may in fact be part of the same document.

3.2.2 A phased approach

The approach to measurement planning outlined within the AMI framework and the ISO/IEC TR 15504 and 15939 documents focuses on the Deming Plan-Do-Act-Check (PDCA) approach - as does ISO 9004: 2000. For software, an interpretation of this approach could be as follows:

Phase 1: Requirements assessment phase

Initially, one needs to identify the organization's needs, business drivers or goals, for example, general improvements in software quality, time to market, and development and support costs. At first, such goals are likely to be somewhat nebulous and difficult to quantify but, as the analysis begins, sub-goals, which are quantifiable, can be developed. Once again, reference to defined quality objectives should be made.

The model provided by ISO/IEC 9126 can help with this: for example, the need for maintainability breaks down into analysability, changeability, stability, testability and compliance. After some initial analysis, one can begin to apply quantifiable parameters to these characteristics. Ensuring that the module structure matches functional requirements, defined processing paths through code, and identifiable equivalence classes for both test data and functions can support testability.

The first draft of the Measurement Plan can now be drawn up. This should be reviewed by the parties concerned and agreement on these basic objectives reached before the next stage.

Phase 2: Analysis phase

Within this phase, the sub-characteristics and accompanying goals are fleshed out with identifiable measurements, current product and processes are assessed for compliance to these requirements, benchmarks are defined and a metrics database - at least in structure – established. The Measurement Plan is extended to cover these aspects.

Phase 3: Metricate phase

With the basic metrics identified, a template for more focused measurement can be developed. Process and product assessments are now taken based on this template, so it is possible to see a means of feeding back requirements into the measurements processes. The Measurement Plan now includes assessment schedules and detailed measurement methods.

Phase 4: Improvement Phase

Comparison of measurement results with the initial goals can now be made. The Measurement Plan is updated to focus on those areas where improvement is needed to meet the defined goals. The Requirements Assessment phase may now be re-visited, comparing and refining goals and building a quantified quality and capability profile for the organization.

Next column

Figure 1 provides a diagrammatic view of this process which, to be effective, needs to cover both the product and the processes needed to develop and support that product.

3.2.3 Some pacing issues

The extent of the measurement activity across the organization needs to be considered carefully. Individual projects are often too time or cost sensitive to allow resources to be committed to measurement activities in what initially amounts to an experiment. Although the use of measurements will ultimately permit closer monitoring and improved quality and productivity on such projects, it is probably better to focus initially on less critical areas. The skills should be built up gradually, to limit the effort expended until the benefits are clear. The most important action is to make a start!

Throughout the processes described above, it is important to identify and retain the support of senior management who are sponsoring the activities, this is one of the key activities identified in ISO/IEC 15939. Tangible benefits may not become apparent until the second iteration or even later, so some act of faith is often needed. The Measurement Plan should be reviewed and updated frequently and measures made of the resources expended as well as the software and process aspects. This information will be needed later to put the final benefits in perspective.

3.2.4 Operating the Plan

The Measurement Plan, shown in Figure 1, is key to the programme. It is continuously updated and improved throughout the cycle and should be clearly referenced in the quality plans of any projects forming part of the initiative. The diagram also identifies the metrics database described above. It is important that a consistent and accessible structure for collecting measurements is established at an early stage. This repository provides a consistent recording structure and assists in normalization of results.

The Measurements Plan will also need to cover those processes that contribute to measurements collection, such as specification review, software inspection, testing, rework etc. The standard procedures covering these processes will need to accommodate the new requirements.

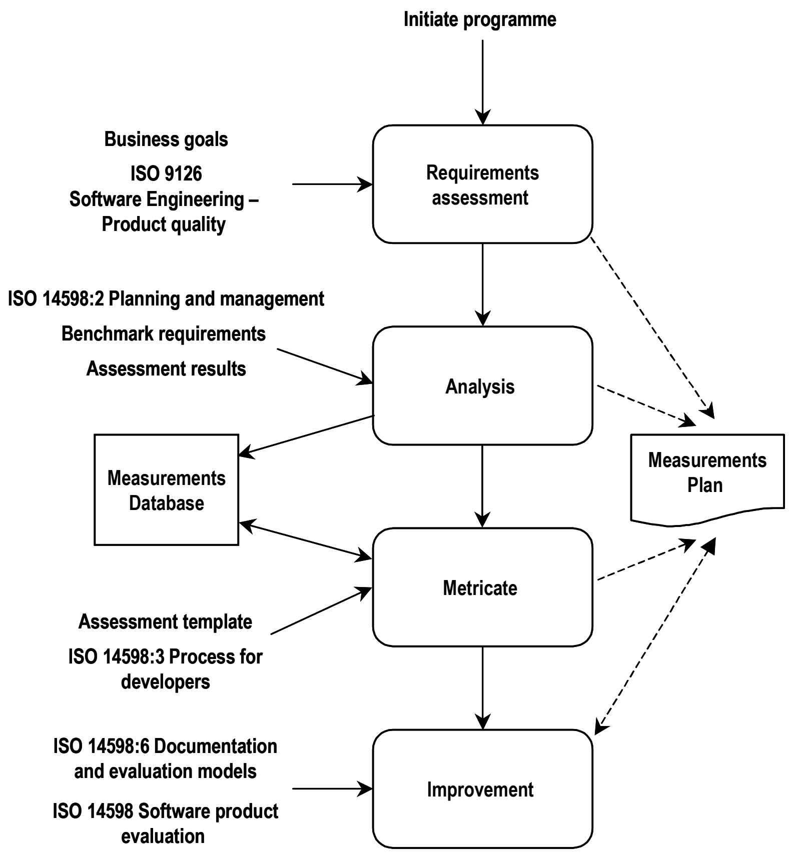
Initiate programme
Business goals
ISO 9126
Software Engineering – Product quality
Requirements assessment
ISO 14598:2 Planning and management
Benchmark requirements
Assessment results
Analysis
Measurements Database
Measurements Plan
Metricate
Assessment template
ISO 14598:3 Process for developers
ISO 14598:6 Documentation and evaluation models
ISO 14598 Software product evaluation
Improvement

Figure 1: Measurement planning process

3.3 Measurements and ISO 9001:2000 requirements

It is now possible to see how a measurement and associated improvement programme fits in with ISO 9001:2000 requirements. Each relevant clause will be considered in turn.

3.3.1 Quality management system – general requirements (clause 4.1)

This clause requires the organization to monitor, measure and analyse its quality management processes. This could also be interpreted as including any out-sourced processes affecting product conformity. A note explains that the set of processes needed for the quality management system should include those for process measurement.

3.3.2 Control of records (clause 4.2.4)

Records of, for example, test results, problem reports, change requests and audit reports are all valuable and the related procedures should contribute to and feature these as part of the measurement programme. It is important to ensure that database measurements may be traced to the originating records.

3.3.3 Quality policy (clause 5.3)

Quality policy must include a commitment to continually improve the effectiveness of the quality management system.

3.3.4 Planning – quality objectives (clause 5.4.1)

This clause states that quality objectives must be measurable and consistent with the quality policy and product requirements. Examples of measurable objectives might be process capability levels or specific targets for product characteristics such as software usability, reliability or functionality. Quality objectives would be expected to be consistent with any improvement plan. Note that measurable quality objectives also apply to product.

3.3.5 Competence, awareness and training (clause 6.2.2)

The training necessary for measurement collection and analysis techniques should be identified and incorporated into a training plan. Appropriate training records need to be kept.

3.3.6 Planning of product realization (clause 7.1)

Here, there is a reference to product quality objectives and requirements and, by referring back to 5.4.1, the clear statement that these must be measurable. The outcome of this activity would normally be a quality plan addressing the measurement activities necessary to achieve product realization. (Note that product in ISO 9001:2000 terms relates to hardware, software, processed material and services.)

3.3.7 Review of requirements related to product (clause 7.2.2)

Performance criteria, for example efficiency, reliability, and specific performance parameters, may be built into a purchase specification, in which case, measurements may be needed to demonstrate eventual contract compliance. It follows that the implications of performance measurements should be assessed when contractual requirements are reviewed. Performance measurements from previous contracts may be used to check the feasibility of the contractual terms before any commitment is made to supply the product.

3.3.8 Design and development planning (clause 7.3.1)

Design processes need to incorporate requirements for measurement collection (initially under quality planning as described above, but eventually incorporated into the procedures). For example, design specifications need to demonstrate compliance to quality characteristics and sub-characteristics (as per ISO/IEC 9126), and design reviews and code inspections need to quantify appropriate measures. Design changes need to take into account measurement requirements.

3.3.9 Design and development verification (clause 7.3.5)

Product verification may, for example, address design criteria. For software this could include coding structure and provisions for testability. Measures can be designed to quantify these aspects.

3.3.10 Design and development validation (clause 7.3.6)

Validation must be performed to assure that the product is capable of fulfilling requirements. Measurement of in-use aspects of the product, such as functionality, can help to provide this assurance.

3.3.11 Control of production and service provision (clause 7.5.1)

This states that production and services must be planned and carried out under controlled conditions, which include the implementation of monitoring and measurement (of product and services). Measurements collected from servicing activities may be important for certain organizations to achieve their goals and it should be evident (in the procedures or via quality planning) that this is so.

3.3.12 Validation of processes for production and service provision (clause 7.5.2)

Production or service processes must be validated where it is not possible to subsequently measure or monitor their output. This means that the performance of such processes must be predicted, for example, by performing process capability measurements. This is applicable in areas such as disaster recovery where process deficiencies only become apparent after delivery.

3.3.13 Identification and traceability (clause 7.5.3)

This clause requires the organization to identify product status with respect to monitoring and measurement requirements. For a software development organization, effective configuration management, linking measurement results to identifiable product baselines, is required. A degree of traceability is required between the product measurements in the database and versions of the product. There is little point making detailed measures if they cannot be related to specific versions of the product. This is particularly relevant when software is complex, when very small changes can have a dramatic impact. The extent and method of handling this aspect of configuration control should be described in the Measurement Plan, the Configuration Plan or individual quality plans.

3.3.14 Control of monitoring and measuring devices (clause 7.6)

This relates to the tools for monitoring and measurement and would normally be associated with physical product and measurement devices. The clause makes clear, however, that the suitability of software (whether as a component of the measuring device, or a stand alone tool) must be confirmed. For software measurement tools, appropriate methods of data collection and presentation need to be used, tools must be compatible with the software or system being measured and the results must be validated.

3.3.15 Measurement, analysis and improvement – general (clause 8.1)

Organizations must plan and take measurements to ensure conformity of both the product and the quality system and to achieve process improvement. Appropriate measurement methods and/or statistical techniques must be identified and used. This implies an

Next column

organization-wide planning activity linked to process improvement rather than a focus on specific projects.

3.3.16 Customer satisfaction (clause 8.2.1)

Customer satisfaction, (or perception of satisfaction), must be one of the performance measures made when assessing the quality management system. The methodology used must be defined, that is the parameters to be measured and the methods of data collection and analysis. A method of determining customer satisfaction could be derived from some of the measures employed for measuring product-in-use quality, (as described in ISO/IEC 9126 part 4).

3.3.17 Internal audit (clause 8.2.2)

The collection and assessment of product measurements would normally be carried out independently of internal quality audits. However, measurements related to process assessment need to be coordinated, so both the overall and detailed planning for internal audit should cater for measurement requirements. Furthermore, whereas compliance audits are generally based on random sampling, measurement sampling needs to be on a more statistical basis. These factors should be considered during audit planning.

3.3.18 Monitoring and measurement of processes (clause 8.2.3)

Measurement of processes should be applied, where applicable, to demonstrate achievement of planned results. This relates to all of the software primary, organizational and support processes that are applicable to the organization, as defined in ISO/IEC 12207, as well as any non-software processes covered by the quality management system.

3.3.19 Monitoring and measurement of product (clause 8.2.4)

Product requirements must be verified by measurement at appropriate stages of product realization in accordance with the product realization plan as described in clause 7.1 A quality model approach similar to that described in ISO/IEC 9126 is therefore applicable where the measuring activity is integrated into the development lifecycle phases. In essence, this is the key ISO 9001:2000 clause relating to the purpose of this document. Where software is the product, quality characteristics of that product, as described in the ISO/IEC 9126 model, provide a structured means of measurement and therefore a statement of its fitness for purpose.

3.3.20 Analysis of data (clause 8.4)

Data must be collected and analysed to determine the suitability and effectiveness of the quality management system and to allow improvements to be identified. As a minimum, this information must include: customer satisfaction (or dissatisfaction), conformance to product requirements, product and process characteristics and trends, and supplier performance. Some form of measurement activity is therefore required in each of these areas.

3.3.21 Continual improvement (clause 8.5.1)

This clause requires the organization to continually improve the effectiveness of the quality management system in line with the declared quality policy and quality objectives. Use should be made of audit results, data analysis, corrective and preventive actions and management review activities. Improvements in processes should be backed up by quantitative measures.

3.3.22 Preventive actions (clause 8.5.3)

The collection, analysis and feedback of measurement results are an important aspect of both identifying preventive action requirements and demonstrating that that the actions taken have been effective. The data should demonstrate, for example, how design and testing defects can be reduced by measurement analysis, and how design processes can benefit.

4. Using software measurements

This section looks at specific processes for handling software measurements from the viewpoint of the three categories of software users – the developer, the acquirer and the evaluator, or assessor. Using the concepts described in the new standards, it discusses how these can be incorporated into both an ISO 9001:2000-compliant quality management system and the Measurements Plan described above. There are very close interrelations between these three categories, and developers, in particular, would benefit by considering all three aspects. Figure 2 shows how simple relationships between measurement and conventional project documents might be established.

The range of software activity and the way software affects business and everyday life is now so extensive that no general guide can hope to address all aspects. This is as true for measurements as any other software engineering activity. The processes described here make no attempt to describe any one particular type of software development process or supporting environment. Therefore, users of this guidance should consider how to incorporate the following generic processes into their own specific environment, bearing in mind the identified improvement objectives.

Next column

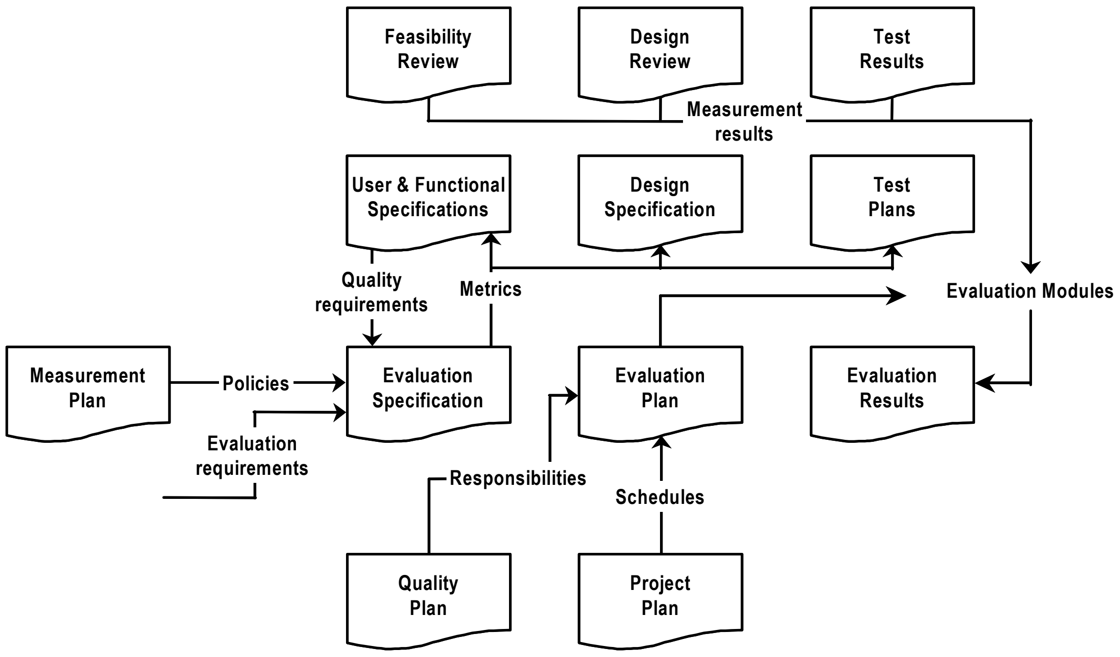

Figure 2: Measurement relationships between project documents

4.1 Measurement processes for developers

Development, in this context, applies to initial product development, incorporation of third party products and maintenance by product support personnel. ISO/IEC 14598 part 3: Process for developers, is a key information source.

Assuming the fundamental quality characteristics have been identified and the necessary measurement framework established, a number of key process stages can then be defined.

4.1.1 Organization

The organizational aspects of development and support need to be addressed as part of the overall quality system and Measurement Plan as far as they are appropriate. ISO/IEC 14598 Part 2: Planning and management, adds support in this area and explicitly covers:

- preparation of a policy statement,
- definition of organizational and improvement objectives,
- identification of technology,
- assignments of responsibilities,
- identification and implementation of evaluation techniques - both for developed and acquired software,
- technology transfer - training, data collection, tools,
- collation and management of improvements within the organization.

Much of the above will have been covered already - in a general sense at least - by ISO 9001: 2000. Further enhancements should feature in the Measurement Plan.

4.1.2 Project planning and quality requirements

Again, normal ISO 9001:2000 and TickIT requirements apply. A development or support life cycle needs to be established and documented in quality planning or other documents. Using the software quality requirements model, any conflicts should be identified and resolved and a feasibility analysis conducted. It is vital to assess whether the product and measurement requirements are technically feasible, reasonable and achievable (given budget and time constraints), complementary and verifiable.

4.1.3 Specifications

Here, the developer maps quality requirements, both external and internal, to the specification. So, for external requirements, say interoperability with other systems over specified interfaces, the necessary internal characteristics are identified in terms of system attributes and metric scales with targets then defined. In the case of interoperability, these might be the degree of compliance with the interfacing standard, provision for tracking data across an interface, or the verified ability to work with multiple variants of an external system.

It is important, also, that a consistent measurement environment is defined, so appropriate and repeatable conditions need to be identified. Once again, the feasibility analysis should be reviewed and any dependencies between attribute values identified.

The outcome of this phase in terms of measurement requirements should be some kind of mapping between requirement specifications, external quality requirements, corresponding internal quality requirements and finally the specified attributes and their metric scales and target values that contribute to the quantification of the software quality. This may seem like a 'per project' or 'per product' approach and, initially, that may well be the case. As the process matures however, specific product measurements need to be placed within the overall organizational framework so that consistent results can be analysed.

4.1.4 Design and planning

As part of design planning, the procedures required for data collection and analysis need to be defined. So, the plan would include scheduling, allocation of responsibilities, use of tools, databases and any specialist training required. The measurement precision and any statistical modelling techniques, including input data, normalization and sampling strategies, should be specified (see ISO/IEC 14598-6 regarding evaluation modules). Design planning and the procedures should also consider how the results of measurement might impact on development. So, contingency actions, the need for additional (or possibly less or more focused) review and testing, and the identification of improvement opportunities should all be considered.

4.1.5 Build and test

During the build and testing phases, actual measurements are collected, appropriate degrees of analysis performed and necessary actions taken. The emphasis during software coding will be on the internal metrics and their attributes, since this is the stage at which they can be measured and their results influence the external characteristics, which are directly measured during the testing and any evaluation stages. So, these aspects of the measurement activity need to be demonstrated in the design reviews and testing plans. At each stage of development, a picture is built up, first focusing on the internal and then the external quality characteristics that define overall product quality and that both complement and are validated by the test results and user experience.

Next column

As a final project phase, an overall review should be conducted to determine the overall effectiveness of the measurement collection exercise, to identify costs against gains, to establish the validity of the metrics used and to identify where benefits could be obtained on future projects. Where the activity is a support function, the results of this review can be fed back directly into future product releases.

4.2 Measurement processes for acquirers

Acquirers would normally be purchasers of complete software packages, either developed to specific requirements (bespoke) or pre-developed for a more general market (commercial off-the-shelf - COTS – products). They could, however, be developers wishing to integrate standard products into their own software design, developers looking to subcontract a part of their development activity to a third party, or developers wanting to use specific software tools.

ISO/IEC 14598-4 Part 4: Process for acquirers, provides a key reference source for developing this area. It is based on the acquisition process of ISO/IEC 12207 and hence relates to ISO 9001:2000 via the mapping provided in the TickIT Guide. There is no reason, however, why the basic components for this process cannot be incorporated into a development project's testing and validation activity; Figure 2 extends this idea with simple relationships between documents and records.

The basis of the process relies on establishing integrity levels and, here, ISO/IEC 15026 Information technology - System and software integrity levels, provides a useful model.

In ISO/IEC 14598-4, four stages are defined and these are discussed below:

4.2.1 Establishment of requirements

The scope of the evaluation needs to be established. Once again, the requirements for software quality defined in ISO/IEC 9126 can be used as a starting point, but other aspects such as cost and regulatory compliance may also need consideration. These are matched to integrity requirements with priority levels, reporting requirements and possibly thresholds. The timing of the evaluation needs to be consistent with objectives; too early an assessment on a development project will not provide a complete picture, whereas too late an assessment may be of limited use.

4.2.2 Evaluation specification

In drafting the Evaluation Specification, the following should be considered:

- the quality requirements to be evaluated, correlated to quality-in-use and external metrics, with priority and acceptance thresholds defined,
- scope of coverage and test cases where applicable, reference to evaluation modules (ISO/IEC 14598-6)
- measurement collection methods, information required and method of analysis.

ISO/IEC 14598-4 gives additional guidance for developing the Evaluation Specification, including methods for recording and correlating data.

4.2.3 Evaluation design

The type of evaluation will depend on the type of software being evaluated. Software under development may be covered at discrete points in development, or on completion. Off-the-shelf software, obviously, is more restricted. In either situation, some form of planning is needed. An Evaluation Plan needs to consider:

- needs for access to product documentation, development tools and personnel,
- costs and necessary expertise required,
- evaluation schedule and contingency arrangements, key milestones and criteria for evaluation decisions,
- reporting methods and tools, procedures for validation and standardization over future projects.

Once again, ISO/IEC 14598-4 provides fuller details and support material.

4.2.4 Evaluation execution

It goes without saying that the evaluation needs to be formally recorded, and normal ISO 9001:2000 requirements for quality planning apply. For some evaluations, this could simply be a record in a logbook; others will need to cover the full scope of project recording including:

- the results themselves and traceability to the product and its configuration information,
- analysis, results and decision records,
- problems, measurement limitations and any compromises made against original objectives,
- conclusions, both of the evaluation results and on the methods employed.

4.3 Evaluating software measurements

ISO/IEC 14598-5 Part 5: Process for evaluators, deals with the systematic evaluation of software using the measurement principles laid out in other parts of this standard.

Next column

Principal objectives of evaluation include:

- repeatability and reproducibility - assessment of the same software, using the same measurement criteria should provide repeatable results, whether or not the same evaluators are used,
- impartiality and objectivity - there should be no bias of the results, which should be factual within the confines of the evaluation requirements.

It should be noted that evaluations based on measurement criteria are fundamentally different from internal ISO 9001 audits, which are based on sampling against conformance requirements. Evaluation as defined here is a process of taking and analysing measurements.

It is clear that requirements for evaluation follow the same basic processes as defined above in those for acquisition. A change of bias is evident in the fact that issues of legal requirements, accessibility and confidentiality are addressed. Similarly, there are references to measurement tool requirements. Specific evaluation requirements to the four stages are summarized below.

4.3.1 Evaluation requirements

Requirements should additionally define:

- the extent of coverage,
- the evaluation objectives and methods of reporting,
- the evaluator qualifications and independence required.

4.3.2 Evaluation specification

Specifications should additionally cover:

- definition of the scope and format of the metrics used, identifying how these may be derived from the product requirements,
- identification of non-deterministic measurements to ensure that the required levels of repeatability and objectivity are obtained,
- identification of any methods of cross-correlation of the measurement results.

Three sub-activities are identified concerning the Evaluation Specification:

- analysing the product description – identifying these key components for evaluation,
- specifying the measurements to be performed - identifying product components in terms of the characteristics and sub-characteristics needed for quality measurement - this should result in a formal metric for the evaluation,

- verifying the resulting specification against the evaluation requirements - a cross check of produced metrics against requirements, ensuring sufficient and complete compliance and ensuring the proposed measurements are consistent with the current state of the art and software engineering standards.

4.3.3 Evaluation plan and evaluation modules

Coordinated measurement activities are a key feature of effective evaluation and here the plan needs to provide an evaluation schedule that provides optimum information when conducted during development. This needs careful and experienced planning, taking into account the development lifecycle, development stage and users needs. It is recommended that evaluation modules, as defined in ISO/IEC 14598-6, are used to provide a consistent and repeatable reporting format.

Evaluation modules are a key component of ISO/IEC 14598 and a means of handling measurement complexity. In particular they provide:

- visibility of the information needed to assess specific quality requirements,
- documentation of the necessary interfaces with measurement tools.

ISO/IEC 14598-6 deals with the documentation requirements and breaks evaluation modules into the following six components:

- *Introduction* - this covers document control, relationships with other documents, technical requirements and a rationale for the module,
- *Scope* - this relates to the quality characteristics, or sub-characteristics that are addressed, the level of evaluation (taking into account the importance of the characteristic, the evaluation techniques used, including any necessary theory) and the applicability of the module,
- *References*,
- *Required definitions*,
- *Inputs required* - data to be collected and metrics to be calculated,
- *Information on interpreting results*.

4.3.4 Evaluation results

This stage addresses the generation of the evaluation report including independent review of evaluation results. Normally the final report would be preceded by a draft so that the personnel involved with the product can provide feedback on the evaluation.

Next column

5. Taking measurement concepts further

The discussion in this document has been based, primarily, on the measurement of identified quality characteristics as described in ISO/IEC 9126. The ISO/IEC Joint Technical Committee, JTC1, have also developed a number of technical reports (normally regarded as embryonic standards) dealing with the further classification of software and using functional size measurement to determine internal and external processing and interfacing requirements. ISO/IEC TR 12182 Information technology - Categorization of software identifies sixteen classifications. These are further developed, using the basis of functional measures in ISO/IEC TR 14143 Information technology - Software measurement - Functional size measurement, particularly in part 5, which deals with the determination of functional domains. An example of this would be to assign a measure of defect density to particular characteristics based on size measurements, for example, the defect density of interface processing functions. This in turn could relate to common characteristics and eventually to quality measures as described in ISO/IEC 9126.

6. The future

The revised ISO 9001:2000 has placed a much greater emphasis on processes and their continued improvement, and on the quantitative assessment of product.

Organizations now need to show how they manage the quality of their product; producing software that is reviewed and tested solely against the functional requirements will no longer be sufficient. Identification of product quality characteristics and their repeated and systematic measurement and feedback in the manner described in this document will, in due course, become an accepted aspect of software development.

As mentioned earlier, the ISO/IEC 9126 and ISO/IEC 14598 standards, upon which a large part of this guide is based, are themselves the subject of further development. BRD/3/1 will therefore monitor these developments and update this document as necessary.

A further, more detailed guide by the same author is in preparation and will be available from BSI. This is entitled "Quantitative software management - software measurement methods". It deals with the topics addressed here and treats the actual processes of measurement in greater depth, but with less emphasis on ISO 9001:2000 and TickIT.

Appendix A - Measurement terminology

The following is a selection of some of the common terminology used in software measurement. References are given where these terms are taken from ISO/IEC standards.

- ***Direct measure:*** A measure of an attribute that does not depend upon a measure of any other attribute (ISO/IEC 14598-1).
- ***External measure:*** An indirect measure of a product derived from measures of the behaviour of the system of which it is a part (ISO/IEC 14598-1).
- ***External quality:*** The extent to which a product satisfies stated and implied needs when used under specified conditions (ISO/IEC 14598-1).
- ***Indicator:*** An indirect measure that can be used to estimate or predict another measure (ISO/IEC 14598-1).
- ***Indirect measure:*** A measure of an attribute that is derived from measures of one or more other attributes (ISO/IEC 14598-1).
- ***Internal measure:*** A measure derived from the product itself, either direct or indirect; it is not derived from measures of the behaviour of the system of which it is a part (ISO/IEC 14598-1).
- ***Internal quality:*** The totality of attributes of a product that determine its ability to satisfy stated and implied needs when used under specified conditions (ISO/IEC 14598-1).
- ***Measure (noun):*** The number or category assigned to an attribute of an entity by making a measurement (ISO/IEC 14598-1).
- ***Measurement attribute:*** A measurable physical or abstract property of an entity.
- ***Measurement scale:*** A scale that constrains the type of data analysis that can be performed on it. Generally there are four basic measurement scale types to consider:
  - *Nominal* - values are categorical, that is they have no measurable purpose other than categorizing some parameter, for example, numbers applied to a football teams players,
  - *Ordinal* - values have a ranking, but other than order, no relative relationships can be drawn, for example, assigning defects to a severity level,
  - *Interval* - values have defined distances between them, such as elapsed time, but no zero value is defined,
  - *Ratio* - values include a zero, such as electrical power measures, which can be multiplied and divided.

  A fifth scale - absolute - is also sometimes applied, which is similar to ratio and allows equality with other scales of the same type.
- ***Measurement:*** The process of assigning a number or category to an entity to describe an attribute of that entity.
- ***Metric:*** A measurement scale and the method used for measurement.
- ***Quality-in-use:*** The quality that is perceived by users when the software is actually used in the users' environment. It can be measured by effectiveness, task efficiency and satisfaction (ISO/IEC 9126 -1).
- ***Quality model:*** The set of characteristics and the relationships between them that provide the basis for specifying quality requirements and evaluating quality (ISO/IEC 14598-1).
- ***Rating level:*** A scale point on an ordinal scale that is anchored to a range of values on another ordinal, interval or ratio scale measure (ISO/IEC 14598-1).
- ***Rating:*** The action of mapping the measured value to the appropriate rating level - used to determine the rating level associated with the software for a specific quality characteristic (ISO/IEC 14598-1).

Appendix B - A summary of measurement techniques

This section gives a brief summary of some typical measures that can be applied to software. The list is not exhaustive, nor is it applicable in all circumstances. The standard IEEE 982.1-1998 Standard dictionary of measures to produce reliable software gives a much fuller description of these measures.

Measurements in design

- ***Defect (or error) density*** - number of errors per defined code size (usually K lines of code), inspection measure.
- ***Cause and effect graphing*** - used to identify incomplete or ambiguous requirements. This is a combinational mapping of input conditions to expected outcomes, a measure of design completeness, once the ambiguities are resolved, the design is considered complete.
- ***Requirements traceability*** - a measure of the completeness of the design in terms of satisfied requirements.
- ***Defect index*** - usually weighted by severity, provides a measure of the correctness as the software passes through design stages.
- ***Hours per defect detection*** - a measure of the efficiency of the design review and code inspection processes.
- ***Number of conflicting requirements*** - used to detect conflicts by mapping inputs and outputs to varying and conflicting requirements.
- ***Number of input and exit points per module*** - measure of the complexity of modules, summated over the product.
- ***Halstead complexity measure*** - determines the relative complexity of code modules by measuring the number of operands and operators in a program.

Measurements in testing

- ***Fault density*** - number of faults found over a specified period per defined code size (usually K lines of code), test measure.
- ***Cumulative failure profile*** - graphical interpretation of suitability of code for use based on fault density at given security levels of failure, may be used as predictive measure.
- ***Functional or modular test coverage*** - total requirements measure over those tested.
- ***Error distribution*** - attempts to classify errors in terms of lifecycle phase. Analysis helps to determine where most development effort is expended.

Measurements in support

- ***Fault days*** - period known faults remain in system from creation until their removal.

Measurements in maintenance

- ***Software maturity index*** - a functional index based on releases. Intended to show the maturity progression of the software in terms of functionality introduced as the product matures.

Appendix C - Other useful standards

ISO/IEC 12119 Information technology - Software packages - Quality requirements and testing

Although referenced by some of the other standards, such as ISO/IEC 14598-5, which focus on the product description, user documentation and programs, this focuses on a different approach to quality requirements. The product description, for example, requires statements on functionality, reliability and usability without the structured characteristics of ISO/IEC 9126, although many of the details are similar. The testing section discusses test pre-requisites, records and test report requirements.

ISO/IEC 12207 Information technology - Software life cycle processes

This is a key standard in the overall software engineering framework, of which measurement requirements form a part. It describes a comprehensive set of software processes under three main categories: primary processes, supporting processes and organizational processes. This structure is directly utilized by most of the other standards described here, in particular ISO/IEC TR 15504. The standard was originally issued in 1995 and is currently being updated with an amendment. It was intended principally as the basis of a contractual relationship between parties developing, acquiring and using software and is intended to be tailored as required. It is supported by a guide, ISO/IEC TR 15271.

ISO/IEC 14143 Information technology - Software measurement - Functional size measurement

This uses Base Functional Components (BFC) to describe a sizing mechanism, which is independent of technology, much like function points.

ISO/IEC 14756 Information technology - Measurement and rating of performance of computer-based software systems

This defines how user orientated performance of computer based systems can be measured and rated. It puts into a mathematical framework a mechanism for consistently measuring both hardware and software performance aspects of computer systems.

ISO/IEC 15026 Information technology - System and software integrity levels

This standard establishes requirements for the determination of integrity levels for software and for systems that use software. By identifying the requirements for system and software level determination, the software integrity requirements can be determined. The integrity level assigned is either the degree of reliability of a mitigating function or a limit on the frequency of failure, that is, the degree of confidence that can be put on the overall system not to fail. Analysis of risk, comprising three phases: risk analysis, risk evaluation and control, is a key process in assessing integrity levels; these reflect the worst-case risk associated with the system.

As in the concept of ISO/IEC 12207, where a contractual two party situation is identified, here an independent integrity assurance authority is envisaged with which the developer or user of the system negotiates assigned integrity levels and the degree of control required. Extensive references to IEC 60300-3-9 (Dependability management - Part 3: Application guide - Section 9: Risk analysis of technological systems) are provided.

Overall the standard is an excellent example of how the basic software processes in ISO/IEC 12207 can be built onto and tailored for a specific need.

ISO/IEC TR 15504 Information technology - Software process assessment

This is a nine-part work, (currently under revision), dealing with capability determination, an assessment framework, assessor qualifications and process improvement. It is currently issued as a Technical Report, a precursor to becoming a full international standard, and is intended to be a framework to which other capability models, such as Bootstrap and the Capability Maturity Model (CMM) comply. ISO/IEC TR 15504 also includes its own assessment model but its use is not mandatory, nor a pre-cursor to using and conforming to the standard.

As indicated above, the assessment process in ISO/IEC TR 15504 maps directly onto that found in ISO/IEC 12207 (in fact, it actually extends ISO/IEC 12207 in a number of key areas). ISO/IEC TR 15504, which originated from the SPICE project, is primarily intended for organizations wishing to measure and improve their own software processes rather than for external evaluation, although such an approach is entirely compatible and adopts a maturity scale, similar to, but more complex than that used by the CMM. ISO/IEC TR 15504 is specifically not intended to be the basis of a certification scheme like ISO 9001.

BS 7925 Software testing

This standard deals principally with low level component testing and describes many useful concepts such as equivalence portioning, state transition testing and cause-effect graphing, together with examples, as well as the plans and documents involved. It comes in two parts, vocabulary and component testing. It should fit in well with quality requirements testing as described in this document.

Appendix D - Reference material

In addition to the standards discussed above, the following further reading material may be of interest:

AMI Handbook (Addison-Wesley, 1996, ISBN 0201877465)

FENTON and PFLEEGER: *Software metrics - a rigorous and practical approach* (PWS Publishing Company, ISBN 0534954251)

PSM: Practical Software Measurement

This organization is supported by the US Army and has an extensive seven-part guide on software measurements. It is available from:
www.psmsc.com

Software Engineering Institute publications

The following SEI documents provide excellent guidance and can be obtained from:
www.sei.cmu.edu/sema

Software Measurement for DoD Systems: Recommendations for Initial Core Measures (CMU/SEI-92-TR-19)

Software Size Measurement: A Framework for Counting Source Statements (CMU/SEI-92-TR-20)

Software Effort and Schedule Measurement: A Framework for Counting Staff-hours and Reporting Schedule Information (CMU/SEI-92-TR-21)

Software Quality Measurement: A Framework for Counting Problems and Defects (CMU/SEI-92-TR-22)

Goal-Driven Software Measurement - A Guidebook (CMU/SEI-96-HB-002)

Practical Software Measurement: Measuring for Process Management and Improvement (CMU/SEI-97-HB-003)

IEEE standards

982.1-1988 IEEE Standard Dictionary of Measures to Produce Reliable Software

982.2-1988 IEEE Guide for the Use of IEEE Standard Dictionary of Measures to Produce Reliable Software