INTERNATIONAL STANDARD

ISO/IEC 25045

First edition
2010-09-01

Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation module for recoverability

Ingénierie des systèmes et du logiciel — Exigences de qualité et évaluation des systèmes et du logiciel (SQuaRE) — Module d'évaluation pour la possibilité de récupération

Reference number
ISO/IEC 25045:2010(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 25045 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*.

ISO/IEC 25045 is one of the SQuaRE series of International Standards, which consists of the following divisions under the general title *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE):*

- Quality Management Division (ISO/IEC 2500n),
- Quality Model Division (ISO/IEC 2501n),
- Quality Measurement Division (ISO/IEC 2502n),
- Quality Requirements Division (ISO/IEC 2503n),
- Quality Evaluation Division (ISO/IEC 2504n).

Introduction

The evaluation of software product quality is vital to both the acquisition and development of software that meets quality requirements. The relative importance of the various characteristics of software quality depends on the mission or objectives of the system of which it is a part; software products need to be evaluated to decide whether relevant quality characteristics meet the requirements of the system.

The essential parts of software quality evaluation are a quality model, the method of evaluation, software measurement, and supporting tools. To develop good software, quality requirements should be specified, the software quality assurance process should be planned, implemented and controlled, and both intermediate products and end products should be evaluated.

This International Standard is part of the SQuaRE series of International Standards. It contains general requirements for specification and evaluation of systems and software quality and clarifies the associated general concepts. It provides a framework for evaluating the quality of software products and states the requirements for methods of software product measurement and evaluation.

The general goal of creating the SQuaRE series of International Standards is to move to a logically organized, enriched and unified series covering two main processes: software quality requirements specification and software quality evaluation, supported by a software quality measurement process. The purpose of the SQuaRE series of International Standards is to assist those developing and acquiring software products with the specification and evaluation of quality requirements. It establishes criteria for the specification of systems and software quality requirements, their measurement, and evaluation. It includes a two-part quality model for aligning customer definitions of quality with attributes of the development process. In addition, the series provides recommended measures of software product quality attributes that can be used by developers, acquirers, and evaluators.

SQuaRE provides

- terms and definitions,
- reference models,
- a general guide,
- individual division guides, and
- International Standards for requirements specification, planning and management, measurement and evaluation purposes.

SQuaRE includes International Standards on quality model and measures, as well as on quality requirements and evaluation.

SQuaRE replaces the current ISO/IEC 9126 series and the ISO/IEC 14598 series.

ISO/IEC 25040, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation reference model and guide* will replace a part of ISO/IEC 14598-1, *Information technology — Software product evaluation — Part 1: General overview*.

ISO/IEC 25041, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation modules* will replace ISO/IEC 14598-6, *Software engineering — Product evaluation — Documentation of evaluation modules*.

ISO/IEC 25001, *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Planning and management* replaces ISO/IEC 14598-2, *Software engineering — Product evaluation — Part 2: Planning and management*.

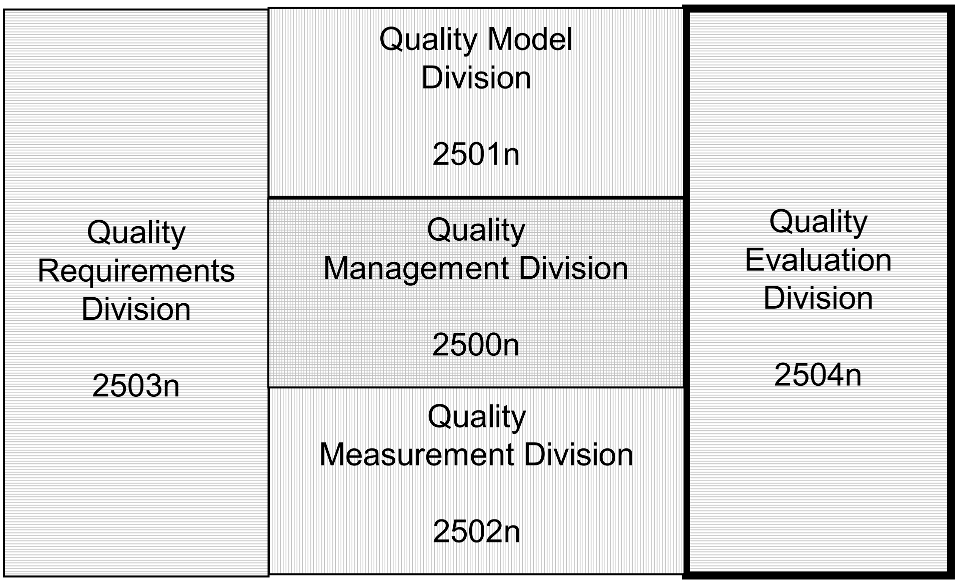

Figure 1 – Organization of the SQuaRE series of International Standards

Figure 1 illustrates the organization of the SQuaRE series, representing families of standards, also called divisions.

The divisions within SQuaRE model are:

- **ISO/IEC 2500n - Quality Management Division.** The International Standards that form this division define all common models, terms and definitions further referred to by all other International Standards from the SQuaRE series. Referring paths (guidance through SQuaRE documents) and high level practical suggestions in applying proper standards to specific application cases offer help to all types of users. The division also provides requirements and guidance for a supporting function which is responsible for the management of software product requirements specification and evaluation.

- **ISO/IEC 2501n - Quality Model Division**. The International Standard that forms this division presents a detailed quality model including internal, external and quality in use characteristics. Furthermore, the internal and external software quality characteristics are decomposed into sub-characteristics. Practical guidance on the use of the quality model is also provided.

- **ISO/IEC 2502n - Quality Measurement Division**. The International Standards that form this division include a software product quality measurement reference model, mathematical definitions of quality measures, and practical guidance for their application. Presented measures apply to internal software quality, external software quality and quality in use. Measurement primitives forming foundations for the latter measures are defined and presented.

- **ISO/IEC 2503n - Quality Requirements Division**. The International Standard that forms this division helps in specifying quality requirements. These quality requirements can be used in the process of quality requirements elicitation for a software product to be developed or as input for an evaluation process. The requirements definition process is mapped to technical processes defined in ISO/IEC 15288, *Systems and software engineering — System life cycle processes*.

- **ISO/IEC 2504n - Quality Evaluation Division.** The International Standards that form this division provide requirements, recommendations and guidelines for software product evaluation, whether performed by evaluators, acquirers or developers. The support for documenting a measure as an Evaluation Module is also presented.

This International Standard is part of the Quality Evaluation Division (ISO/IEC 2504n), which consists of the following International Standards (see Figure 2).

- ISO/IEC 25040[1)], *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation reference model and guide*, contains general requirements for specification and evaluation of software quality and clarifies the general concepts. It provides a process description for evaluating the quality of software products and states the requirements for the application of this process. The evaluation process is the basis for software product quality evaluation for different purposes and approaches. Therefore, the process can be used for the evaluation of quality in use, external software quality and internal software quality. It can also be applied to evaluate the quality of pre-developed software or custom software during its development process. The software product quality evaluation can be conducted by an acquirer, a developer organization, a supplier or an independent third party evaluator.

- ISO/IEC 25041[2)], *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation modules*, defines the structure and content of the documentation to be used to describe an evaluation module. These evaluation modules contain the specification of the quality model (i.e. characteristics, sub-characteristics and corresponding internal, external or quality in use measures), the associated data and information about the planned application of the model and the information about its actual application. Appropriate evaluation modules are selected for each evaluation. In some cases, it might be necessary to develop new evaluation modules. Guidance for developing new evaluation modules is found in ISO/IEC 25041. This International Standard can also be used by organizations producing new evaluation modules.

- ISO/IEC 25045, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation module for recoverability* provides the specification to evaluate the sub-characteristic of recoverability defined under the characteristic of reliability of the quality model. The ability of a software product and thereby a system to remain available or to recover within an acceptable timeframe from disturbance has always been important since a down time often has economic and other consequences. The emphasis in recent years has extended to the autonomic ability of the software product and thereby a system to be self-managed with minimal involvement by human operators. There are interests in the user domain and industry on how well a software product and thereby a system handles such disturbances in the way it detects, analyses, adjusts or recovers. This International Standard determines the quality measures of resiliency and autonomic recovery index when the information system composed of one or more software products' execution transactions is subjected to a series of disturbances. A disturbance could be an operational fault (e.g. an abrupt shutdown of an operating system process that brings down a system) or an event (e.g. a significant increase of users to the system).

1) To be published.

2) Under preparation.

Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation module for recoverability

1 Scope

This International Standard is one of the SQuaRE series of International Standards, which contains general requirements for specification and evaluation of systems and software quality and clarifies the associated general concepts. SQuaRE provides a framework for evaluating the quality of software products and states the requirements for methods of software product measurement and evaluation.

This International Standard uses a methodology involving two types of evaluation for recoverability. One part of the method makes use of the disturbance injection methodology and a list of disturbances based on common categories of operational faults and events to evaluate the quality measure of resiliency. The second quality measure is based on a set of questions that is defined for each disturbance to evaluate the quality measure of autonomic recovery index by assessing how well the system detects, analyses, and resolves the disturbance without human intervention.

This International Standard is applicable to information systems executing transactions in a system supporting single or multiple concurrent users, where speedy recovery and ease of managing recovery is important to the acquirer, owner/operator, and the developer.

1.1 Characteristics

This evaluation module measures the quality measures defined under the following characteristic and sub-characteristics of the quality model as defined in ISO/IEC 9126-1:2001.

NOTE The reference to ISO/IEC 9126-1 will be replaced by a reference to ISO/IEC 25010 when published.

Characteristic – Reliability

Sub-characteristic – Recoverability

Quality measure – Resiliency

Quality measure – Autonomic recovery index

1.2 Level of evaluation

Level D as defined in ISO/IEC 14598-5. This evaluation is intended for a system with executable products.

NOTE The reference to ISO/IEC 14598-5 will be replaced by a reference to ISO/IEC 25040 when published.

1.3 Technique

A disturbance injection methodology is a test methodology where disturbances are injected against the application and other components of the system while it is running a workload of interest to the acquirer. A disturbance injection methodology and a list of disturbances based on common categories of operational faults and events are used to evaluate the quality measure of Resiliency. For each disturbance, the Resiliency of the system is calculated based on the ratio between the number of transactions that complete successfully

while the system is under disturbance and the number of transactions that complete successfully in a system that does not encounter the disturbance. A set of disturbances is defined under the following categories:

- Unexpected shutdown — e.g. abrupt operating system (OS) shutdown, process shutdown, network shutdown;
- Resource contention — e.g. CPU/memory/IO hogs, memory leak, database management system (DBMS) runaway query, DBMS deadlock, DBMS and queuing server storage exhaustion;
- Loss of data — e.g. DBMS loss of data, DBMS loss of file, DBMS and queuing server loss of disk;
- Load resolution — e.g. a moderate or significant increase of users or workload;
- Restart failures — e.g. restart failure on OS and middleware server process.

Other disturbance categories may be identified if appropriate.

A set of questions to assess how well the system detects, analyses, and resolves the disturbance is defined for each disturbance to evaluate the quality measure of autonomic recovery index. A score is calculated for each disturbance based on the answers to those questions.

The overall Resiliency and autonomic recovery index are calculated respectively as an average of those individual scores.

The detailed evaluation methodology involved is given in 5.1.

1.4 Applicability

This evaluation module is applicable to an information system that involves a software product and other software components. The information system must have a workload that has a consistently reproducible performance result to properly assess the impact of disturbance and recovery.

The evaluation module can be used in the following situations:

a) evaluation as part of the system verification testing;

b) evaluation against the test environment of a production system to gauge recoverability and identify weakness;

c) evaluation of the recoverability of different solutions proposed by vendors using a common workload.

The evaluation result is only applicable to the specific release and configuration of the software and hardware components on which they were evaluated. Two results are comparable if they use the same workload and workload parameter set defined in 5.2.2.2 and fault load and fault load parameter set defined in 5.2.3.2 for the evaluation.

2 Conformance

An evaluation of the recoverability of a software product conforms to this International Standard if it complies with Clause 5.

3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 25000:2005, *Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*

4 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 25000 and the following apply.

4.1
performance baseline
result from a normal execution of a performance workload against a system without performing disturbance injection

4.2
disturbance
operational fault (e.g. an abrupt shutdown of an OS process that brings down a system) or event (e.g. a significant increase of users to the system), or anything that could change the state of the system

NOTE For the context of this evaluation module, the disturbances are limited to external faults or events, rather than internal faults that required modifying the application or OS code.

4.3
injection slot
point where the recoverability of the system under test (SUT) is tested by injecting a disturbance while a workload is being run

5 Inputs and measures

5.1 Evaluation methodology

The evaluation shall follow the methodology outlined below utilizing an existing performance workload, injecting disturbances which are faults or events as the workload is executing, and measuring the performance under disturbance as compared to a stable environment.

The evaluation methodology consists of three phases, as outlined in Figure 2 below. These are the Baseline phase, the Test phase, and the Check phase. Note that prior to running a Baseline phase or Test phase, the workload must be allowed to ramp up to steady state, in which the workload runs at a consistent level of performance.

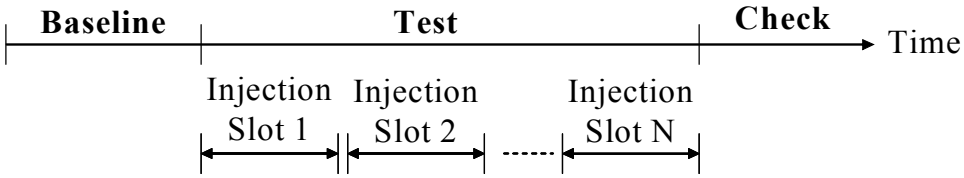

Figure 2 — Three Phases of the Evaluation Methodology

The **Baseline phase** determines the operational characteristics of the system in the absence of the injected perturbations. This baseline phase is run to generate a performance baseline that shall be used to compare the result from the test phase, and shall comply with all requirements defined by the performance workload.

The **Test phase** determines the operational characteristics of the system when the workload is run in the presence of the disturbances. This phase shall use the same setup and configuration as the Baseline phase.

The Test phase is divided into a number of consecutive Disturbance *Injection Slots*. These injection slots shall be run one after another in a specified sequence.

The **Check phase** ensures that the reaction of the system to the disturbance did not affect the integrity of the system. During this phase, a check shall be made to ensure that the system is in a consistent state.

During each injection slot, the fault load driver initiates the injection of a disturbance into the system under test (SUT). Ideally, the SUT detects the problem and responds to it. This response can consist of either fixing the problem or bypassing the problem by transferring work to a standby machine without resolving the original problem. If the SUT is not capable of detecting and then either fixing or bypassing the problem automatically, the fault load driver waits an appropriate interval of time, to simulate the time it takes for human operator intervention, and initiates an appropriate human-simulated operation to recover from the problem.

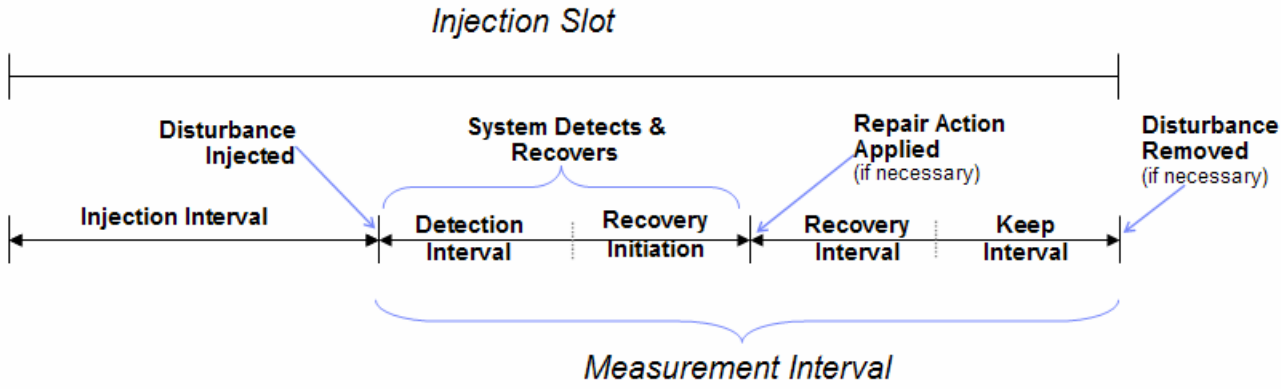

Figure 3 — Injection slot sub-intervals

As Figure 3 demonstrates, each injection slot consists of five sub-intervals.

- The **Injection Interval** is the predefined time that the system is allowed to run at steady state before a particular fault is injected into the SUT. The benchmark driver waits for the predefined injection interval before injecting the fault. The purpose of the injection interval is to demonstrate that the system is functioning correctly before any disturbance is injected.

- The **Detection Interval** is the time from when a fault is injected to the time when a fault is detected. For an SUT that is not capable of detecting a fault automatically, the driver will be configured to wait for a predefined Detection Interval before initiating a recovery action. This is to simulate the time it takes for the human operator to detect a fault.

- The **Recovery Initiation Interval** is the time from when a fault is detected to the time when a recovery action begins. For an SUT that is not capable of detecting the fault or initiating a recovery action automatically, the driver will be configured to wait for a predefined Recovery Initiation Interval before initiating the recovery action. This is to simulate the time it takes for a human operator to initiate recovery.

- The **Recovery Interval** is the time that it takes the system to perform recovery.

- The **Keep Interval** is the time to ramp-up again and run at steady state after the recovery. This is the time remaining in a measurement interval. If a steady state is not achieved or is lower than that prior to the Disturbance Injection, this should be noted in the report.

It is important to note two things.

- First, the breakdown of the slot interval into sub-intervals is for expository purposes only. During a benchmark run, the benchmark driver only distinguishes the boundaries of these sub-intervals when the SUT requires simulated human intervention.

- And second, only the operations processed during the last four of the five intervals are part of the measurement interval, and are, therefore, counted when calculating the throughput for the run.

5.1.1 Practical considerations relating to the methodology

A test is more controllable if each injection slot is run in isolation such that the system is stopped, reset, and started and ramped-up between each injection slot. This will require the Check phase after each injection slot instead of after all Injection Slots as described in Figure 2.

If the customer wants or agrees (such as to speed up the test or to see how the system react to multiple disturbance that occurs one after another), the test could be setup to run some injection slots one after another without stopping, resetting, starting, and ramping up to steady state between each injection slot. This might be suitable for disturbances that do not bring down the SUT. Otherwise the resulting database recovery would take much longer due to the need to recover for all prior transactions from previous injection slots. If the test is to be run to compare different systems, and the injection slots are not run in isolation, the specific sequence and grouping of injection slots should be used for all the systems.

The interval length of the run depends on the workload. Larger workload with higher throughput tends to require a longer ramp-up period to reach the steady state where an injection slot could begin. The following is one example that had been used to provide a balance between efficiency and the need to allow a SUT sufficient time to detect and repair from the injected disturbances: For a baseline run allow the system to warm up for 5 minutes, and then use a 50-minute Baseline Phase. For a test run, allow the system to warm up for 5 minutes, and then use a 50-minute Test Phase, which is broken up into 10 minutes for the Injection Interval, 20 minutes for the combined Detection Interval and Recovery Initiation Interval, and 20 minutes for Recovery Interval and Keep Interval.

5.1.2 Disturbances

The disturbances and categories of disturbances in the execution runs are not intended to be comprehensive, and the user of this International Standard can extend the list based on experience and context. The list of disturbances is intended to cover common operation faults and events, where some disturbances could be due to operator mistakes or even malicious action but the list does not handle security issues. It is not the intention of this evaluation module to evaluate system security.

All five disturbance categories described below shall be used for conformance.

5.1.2.1 Unexpected shutdown

Disturbances in this category simulate the unexpected shutdown of an OS, one or more application processes, or the network link between components in the SUT.

Table 1 — Disturbances for unexpected shutdown

Disturbance name	Description
Abrupt OS shutdown for DBMS, application, HTTP, and messaging servers	This fault scenario represents the shutdown of the server OS. It is intended to simulate the situation where an operator accidentally issues an OS shutdown command either remotely or at the console. All the processes on the server are stopped and the OS is halted gracefully. This is different from a system crash due to a software defect, a power failure (which is tied to the hardware), or accidentally shutting down by using the power switch.
Abrupt process shutdown for DBMS, application, HTTP, and messaging servers	This fault scenario represents the shutdown of one or more processes supplying the component of the SUT. It is intended to simulate the situation where an operator accidentally issues an OS command to end the processes. This is different from issuing a command to the processes to inform them of the need to terminate. The only alert provided to the processes that "the end is near" is that supplied by the OS to all processes that are to be ended. (E.g. signal 9 in Linux).
Network shutdown for DBMS, application, HTTP, and messaging servers	This fault scenario represents the shutdown of the network link between critical components of the SUT. It is intended to simulate the situation where the network becomes unavailable because of a pulled cable, faulty switch, or OS level loss of network control.