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)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

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 ... iv

Introduction ... v

1 Scope ... 1
1.1 Characteristics ... 1
1.2 Level of evaluation ... 1
1.3 Technique ... 1
1.4 Applicability ... 2

2 Conformance ... 2

3 Normative references ... 2

4 Terms and definitions ... 3

5 Inputs and measures ... 3
5.1 Evaluation methodology ... 3
5.1.1 Practical considerations relating to the methodology ... 5
5.1.2 Disturbances ... 5
5.2 Input for the evaluation ... 8
5.2.1 The SUT description ... 8
5.2.2 The workload description ... 9
5.2.3 The fault load description ... 10
5.3 Data elements ... 11
5.3.1 Output from the baseline run ... 11
5.3.2 Output from the test run ... 11
5.3.3 Completion of the Autonomic Maturity Questionnaire ... 12
5.4 Quality Measures ... 12
5.4.1 Summary of the Quality Measures and Quality Measure Elements (QME) ... 12
5.4.2 Quality Measure - Resiliency ... 12
5.4.3 Quality Measure - Autonomic Recovery Index ... 13
5.4.4 Quality Measure Element (QME) - Number of transactions under disturbance ... 15
5.4.5 Quality Measure Element (QME) - Number of transactions under no disturbance ... 16
5.4.6 Quality Measure Element (QME) - Autonomic Maturity Score ... 16

6 Interpretation of results ... 17
6.1 Mapping of measures ... 17
6.2 Reporting ... 17
6.3 Application Procedure ... 17

Annex A (informative) Sample Report ... 18

Bibliography ... 37

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*.

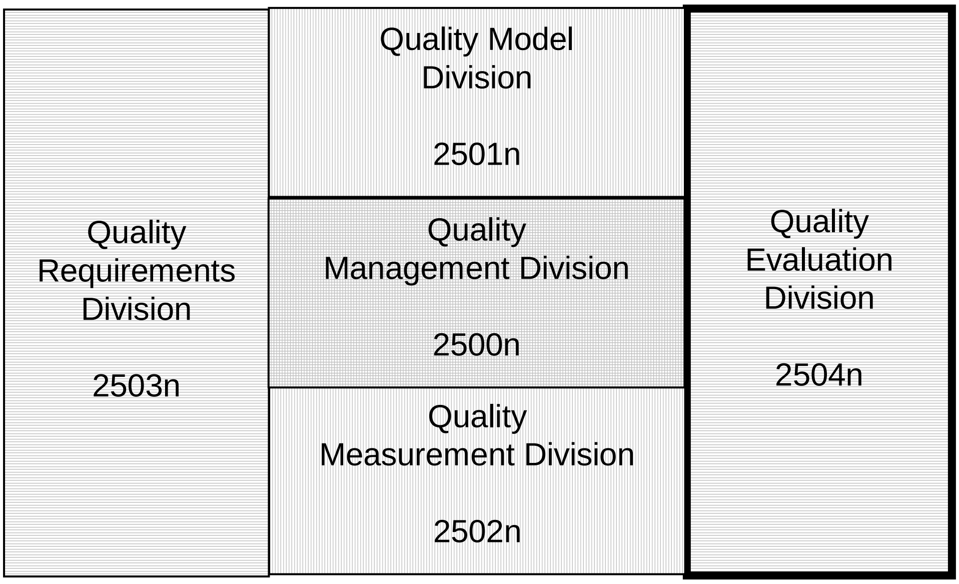
Quality Model Division
2501n
Quality Requirements Division
2503n
Quality Management Division
2500n
Quality Evaluation Division
2504n
Quality Measurement Division
2502n

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.

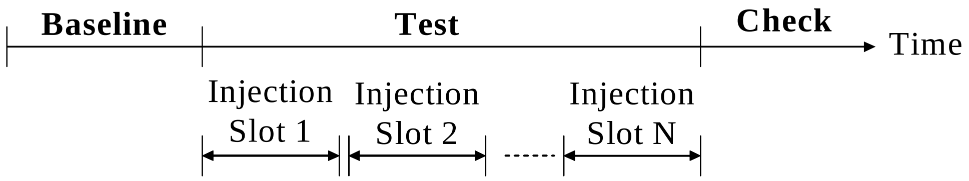

Figure 2 — Three Phases of the Evaluation Methodology

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.

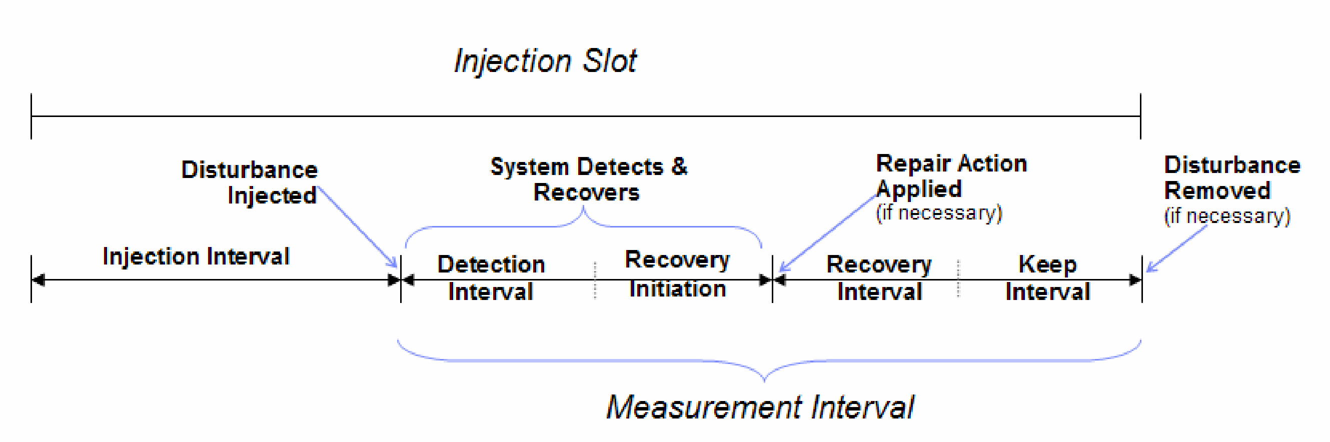

Figure 3 — Injection slot sub-intervals

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.

5.1.2.2 Resource contention

Disturbances in this category simulate the case in which resources on a machine in the SUT are exhausted because of an unexpected process, user action, or application error.

Table 2 — Disturbances for resource contention

Disturbance name	Description
Memory hog on DBMS, application, HTTP, and messaging servers	This fault scenario represents the case where all the physical memory on the system is exhausted. It is intended to simulate the situation in which a certain process in the machine stops being a good citizen and takes over all the physical memory. All the free physical memory of the system is taken up by the hog process. This disturbance is complicated by the virtual memory system, so the current implementation is to request all physical memory and randomly access within this memory to simulate page requests.
I/O hog on DBMS server	This fault scenario represents the case where the disk bandwidth of the physical drive containing the business data is saturated. It is intended to simulate the situation in which a certain process in the machine stops being a good citizen and creates unplanned heavy disk I/O activities. The disk actuator is busy servicing read or writes requests all the time. This shouldn't be confused with the case where the bandwidth of the I/O bus is saturated.
DBMS runaway query	This fault scenario represents the case where the DBMS is servicing a runaway query. It is intended to simulate the situation in which a long-running, resource-intensive query is accidentally kicked off during operation hours. It shouldn't be confused with a batch of smaller queries being executed.
Messaging server poison message flood	This fault scenario represents the case where the message queue is flooded with many poison messages. A poison message is a message that the receiving application is unable to process, possibly because of an unexpected message format. It is intended to simulate the situation in which the operator configures a wrong queue destination. A large number of poison messages are sent to the message queue. This shouldn't be confused with the case where the application is causing a queue overflow.
DBMS and messaging server storage exhaustion	This fault scenario represents the case where the system runs out of disk space. It is intended to simulate the situation in which a certain process in the machine stops being a good citizen and abuses the disk quota. All the disk space of the drives containing the business data is taken up by the hog process.
Network hog on HTTP, application, DBMS, and messaging servers	This fault scenario represents the case where the network link between two systems in the SUT is saturated with network traffic. It is intended to simulate the situation where a certain process in the machine stops being a good citizen and transfers excessive data on a critical network link. This test should be performed in a private network such as with its own private switch or contained within a network segment to avoid impacting other systems in the wider network.
Deadlock on DBMS server	This fault scenario represents the case in which a deadlock involving one or more applications leaves a significant number of resources (rows or tables) in the DBMS locked, making them inaccessible to all applications. Any queries on the DBMS that require these locked resources will not complete successfully.
Memory leak in a user application	This fault scenario represents the case in which a user application causes a memory leak that exhausts all available memory on the system. It is intended to simulate the case in which a poorly written application is deployed onto an application server.

5.1.2.3 Loss of data

Disturbances in this category simulate a scenario in which business-critical data is lost.

Table 3 — Disturbances for loss of data

Disturbance name	Description
DBMS loss of file	This fault scenario represents the loss of database file which contains critical business data. It is intended to simulate the situation where an operator accidentally issues an OS command to delete the one or more database files that contain data for a particular database object. The DBMS can no longer address the file from the file system. This is different from an OS file handle loss, which is considered a bug in the OS
DBMS and messaging loss of disk	This fault scenario represents the loss of a physical hard drive that contains the business data. It is intended to simulate the case where a hard drive is damaged such that the disk controller marks the targeted hard drive as offline.

5.1.2.4 Load resolution

Disturbances in this category simulate a sudden increase in the workload on the system.

Table 4 — Disturbances for load resolution

Disturbance name	Description
Significantly increased load handling and resolution	This fault scenario represents the case where the load on the SUT increases drastically (generally about 10 times the previous load). It is intended to simulate the situation where a significantly heavy load is introduced because of a catastrophic event or failure of the primary system. The optimal result for this disturbance is to handle at least the same amount of business as before without being overwhelmed by the extreme increase in requests. Technologies that illustrate this characteristic would be flow control and quality of service monitors.

5.1.2.5 Detection of restart failure

Disturbances in this category simulate a situation in which an application or the component it depends on is corrupted and cannot be restarted.

Table 5 — Disturbances for restart failure

Disturbance name	Description
Process restart failure of DBMS, application, HTTP, and messaging servers	The fault scenario represents the case where the software component fails to restart. It is intended to simulate the case where a key file, or data, that is required during the start-up process is lost. When the software program is restarted, it fails at the point where the key file or data cannot be loaded.

5.2 Input for the evaluation

NOTE Section A.5 to A.8 in the sample report provides an example of the type of output documented here.

5.2.1 The SUT description

5.2.1.1 Specification of the hardware and OS configuration

The properties of the hardware architecture and configuration shall be described in sufficient details to allow replication of the hardware and OS configuration. These include but not limited to the following:

- vendor and model number;
- system availability date;
- CPU (processor type, number and speed (MHz/GHz) of the CPUs);
- cache (L1, L2, L3, etc);
- main memory (in Megabytes);
- disks and file system used;
- network interface;
- number of systems with this exact same configuration;
- OS (product name, vendor, and availability date);
- OS tuning parameters and options changed from the defaults;
- compilation and linkage options and run-time optimizations used to create/install OS;
- logical or physical partitioning used on this system to host software instances;
- which software components, application, and additional software from 5.2.1.2, 5.2.1.3, and 5.2.1.4 run on this hardware.

5.2.1.2 Specification of the software component configuration

The properties of the software components such as web server, application server, message server, database server, JVM, etc, that the applications use shall be described in sufficient details to allow replication of the software configuration. These include but not limited to the following:

- vendor name, product name and version, and availability date;
- tuning parameters and options changed from the defaults;
- compilation and linkage options and run-time optimization used to create/install the software component;
- number of instances on each system.

5.2.1.3 The application programs

All programs used by the emulated users shall be presented on a digital storage medium. These programs shall be ready for use on the SUT (either as an executable program or the complete source code). They shall be described in sufficient details to allow replication of the software configuration. These include but not limited to the following:

- vendor name, product name and version, and availability date;
- tuning parameters and options changed from the defaults;

- compilation and linkage options and run-time optimization used to create/install the software component;
- number of instances on each system.

5.2.1.4 Additional software required

A list of all additional software components or standard system software modules which are needed to run shall be described in sufficient details to allow replication of the software configuration. These include but not limited to the following:

- vendor name, product name and version, and availability date;
- tuning parameters and options changed from the defaults;
- compilation and linkage options and run-time optimization used to create/install the software component;
- number of instances on each system.

For the baseline run, this shall include the test driver that simulates the multiple users and drives the test scripts.

For the test run, this shall include the fault injection software.

5.2.1.5 The stored data

All data, which are needed by the programs for their correct working or which have any influence on the performance of the SUT so long as they are not contained in the descriptions of the task type input, shall be presented in their entirety on digital storage medium. They shall be formatted ready for immediate use and storage on the SUT without any further modification. Examples of such data can be:

- data files, required for a correct computation;
- output data files used by the programs, which are not empty when starting the test;
- the data of a data base system.

5.2.1.6 Additional information for proof

It is the responsibility of the tester to submit the results of the measurement to proof. Therefore the tester shall supply additional documents of his/her own choice in addition to the documents requested in this International Standard, which are suitable to repeat the measurement by an external person/group to attain the same results.

5.2.2 The workload description

5.2.2.1 The workload specification

Describe the workload in sufficient details to allow replication of the software configuration. These include but not limited to the following:

- description of the transaction or operation that will be tested;
- description of the test data;
- the test scripts.

5.2.2.2 The workload parameter set

The values for the set of parameters used to drive the workload shall be described in sufficient details to allow replication of the software configuration. This shall include any configuration parameters for the workload driver and the application that could affect the performance and application behaviour. These include but not limited to the following:

- the total number of user;
- the duration of the baseline run (in seconds), including the duration of the ramp up period, steady state, and shutdown period;
- the reporting interval (in seconds) where the transaction rate is reported during the run;
- the transaction mix of the workload (e.g. 10% New Order, 20% Status Query, etc);
- whether the transaction mix is maintained at each reporting interval, and if not, describe when each type of transactions is executed during the run;
- other configuration changes that determine how the workload will be run that might affect the repeatability and performance of the run;
- any configuration changes on the application that might affect the repeatability and performance of the run.

5.2.2.3 Parameter set for proving the consistency and stability of the workload

In order to properly evaluate the effect of disturbance injection, the baseline must be shown to be repeatable and consistent. The baseline shall be run 3 times and the following values shall be provided as proof of its repeatability and consistency.

- The required statistical significance of the measurement results as defined by the acquirer (e.g. the number of successful transactions completed in each of 3 baseline runs should not differ by more than 5%).
- The reported statistical significance of the measurement results.
- Identify any significant performance spikes and dips in any reporting intervals during the run that are greater than the required statistical significance defined by the acquirer. Explain what might be causing those. An explanation of any performance spikes and dips for a reporting interval during the run if they exceed the required statistical significance defined by the acquirer. Either the presence of spikes and dips in a reporting interval or the presence of different types of transactions at various reporting intervals may indicate a problem. The injection of a disturbance at such intervals could produce variation in performance and quality measures. This shall be noted in the report. However, as long as their occurrences are consistent in multiple runs the results are comparable.

5.2.3 The fault load description

5.2.3.1 The fault load specification

The list of the disturbances to be executed against the workload shall be described comprehensively and grouped by categories of disturbance as defined in 5.1.2. Disturbances shall be defined for software components that the application depends on such as the web server, application server, message server, the database server, etc.

5.2.3.2 The fault load parameter set

Each disturbance shall be run within an injection slot as defined in 5.1. The following values used in the injection slot shall be described:

- measurement interval (in seconds), also known as the duration of injection slot;
- injection interval (in seconds);
- detection interval (in seconds).

If multiple disturbances are to be executed in sequence without stopping and restarting the workload, the disturbances and their sequence of execution shall be described.

5.2.3.3 The Autonomic Maturity Questionnaire

A questionnaire composing of the questions defined in 5.4.3 shall be created for each disturbance to be executed.

5.3 Data elements

Note Section A.9 to A.11 in the sample report provides an example of the type of output documented here.

5.3.1 Output from the baseline run

This is the output from a normal execution of the workload against the system. The output shall provide the following:

- the workload parameter set that was used in the run and recorded in the output;
- measurement interval;
- reporting interval (e.g. every 30 seconds, or every 10 minutes, etc, throughout the workload);
- total transaction completed without error within the measurement interval, and between reporting interval;
- total transaction completed with errors within the measurement interval;
- other system performance information including the CPU utilization and i/o utilization of major components used in the SUT (e.g. web server, application server, database server, etc) at each interval for comparison with a SUT.

5.3.2 Output from the test run

This is the output from the workload against the system while under disturbance Injections. This output shall provide the following for each injection slots:

- the workload parameter set that was used in the run and recorded in the output;
- measurement interval (in seconds);
- injection Interval (in seconds);
- whether manual recovery was used to disable the disturbance and recover the system to the steady state like that of a baseline run;
- reporting interval (in seconds, e.g. every 30 seconds, or every 10 minutes, etc throughout the test);
- total transaction completed without error within the measurement interval and between each reporting interval;
- total transaction completed with errors within the measurement interval;
- other system performance information including the CPU utilization and i/o utilization of major components used in the SUT (e.g. web server, application server, database server, etc) at each reporting interval for comparison with a baseline test;

- Whether the check phase was performed to verify the integrity of the system. If performed, state the method of verification and result. If not performed, state why this is not necessary.
- The quality measure of resiliency shall be reported for each disturbance and over the entire set of disturbances.

5.3.3 Completion of the Autonomic Maturity Questionnaire

A completed autonomic maturity questionnaire on how the problem is detected, analyzed, and resolved shall be provided.

The autonomic maturity score (a quality measure element) shall be reported for each disturbance and over the entire set of disturbances. The quality measure of autonomic maturity index shall be reported.

5.4 Quality Measures

NOTE Section A.2 and A.10 in the sample report provides an example of the type of output documented here.

5.4.1 Summary of the Quality Measures and Quality Measure Elements (QME)

Quality Characteristic - Reliability

- Quality Sub-characteristics - Recoverability
  - Quality Measure - Resiliency
    - QME - Number of transaction under disturbance - number of transactions completed without error within a measurement interval where disturbance(s) are injected
    - QME - Number of transactions under no disturbance - number of transactions completed without error within a measurement interval where disturbances are not injected
  - Quality Measure - Autonomic recovery index
    - QME - Autonomic maturity score

The following clauses describe the new quality measures and quality measure elements.

5.4.2 Quality Measure - Resiliency

The *Resiliency* is a quantitative measure of the quality of service under test. It describes the relationship between the throughput when the system is fouled with a disturbance and the throughput when it is not infected.

Measure Name -- Resiliency

Purpose of Measure – How resilient is the system when it encounters disturbance?

Method of application – refer to 5.1

Measurement, formula, and data element computations – For each disturbance, calculate

P_i / P_{base} *where*

P_i = number of transactions completed without error within a measurement interval of an injection slot where disturbance(s) were injected.
P_{base} = number of transactions completed without error within the measurement interval (of a baseline run where disturbances(s) were not injected

The overall Resiliency is calculated by taking the average of the Resiliency for each injection slot.

Interpretation of measured value – 0 <= x. The higher the better. (In practice, the value is likely 0 <= x <= 1, although in theory *x* is possible to attain a value > 1 if the system under disturbance attains a higher throughput than the one without.)

Measure scale type –- Absolute

Measure type – P_i = Count, P_{base} = Count

Input to measurement – Test report

ISO/IEC 12207:2008 SLCP Reference – 6.4.5 System Integration Process, 6.4.6 System Qualification Testing process, 6.4.9 Software Operation Process

Target audience – Acquirer, Supplier, Developer, Maintainer

5.4.3 Quality Measure - Autonomic Recovery Index

The *Autonomic Recovery Index* is a qualitative measure of the level of autonomic capability for recovery.

Measure Name -- Autonomic Recovery Index

Purpose of Measure -- How well does the software product detect, analyse, and resolve disturbances?

Method of application –

For each disturbance, observe the behaviour of the system in detecting, analysing, and resolving the disturbance which are faults or events, then answers a set of questions from a questionnaire to obtain the score.

The score for each disturbance shall be calculated based on answers to the questions on the Autonomic Maturity Questionnaire answered by the test operator. Each answer is given a value based on the increasing autonomic level of the response of the system as follows:

Table 6 — Autonomic Level

Autonomic Level	Description
Basic	Rely on reports, product, and manual actions to manage IT components
Managed	Management software in place to provide facilitation and automation of IT tasks
Predictive	Individual components and systems management tools able to analyze changes and recommend actions
Adaptive	IT components collectively able to monitor, analyze, and take action with minimal human intervention
Autonomic	IT components collectively and automatically managed by business rules and policies

Each question shall have one of six answers each with a different score that correspond to the Autonomic Level as follows:

- A is awarded 0 points (basic)
- B0 is awarded 0.5 points (basic/managed)
- B is awarded 1 point (managed)
- C is awarded 2 points (predictive)
- D is awarded 3 points (adaptive)
- E is awarded 4 points (autonomic)

NOTE The value of the points awarded could be adjusted based on experience, customer preference, and context.

The following questions shall be used for each disturbance:

- How is the disturbance detected?

  A. The help desk calls the operators to tell them about a rash of complaints.

  B0. The operators detect the problem themselves by monitoring multiple data sources.

  B. The operators detect the problem themselves by monitoring a single data source.

  C. The autonomic manager notifies the operator of a possible problem.

  D. The autonomic manager detects the problem without human involvement.

  E. Same as D. Chosen only if the answer to the “How is the disturbance analysed?” question is also E, i.e. when the system monitors and correlates data based on business rules and policies that allow actions to be taken without human involvement.

- How is the disturbance analyzed?

  A. The operator collects and analyzes multiple sources of system-generated data.

  B. The operator analyzes data from a single management tool.

  C. The system monitors and correlates data that leads to recommended recovery actions.

  D. The system monitors and correlates data that allows actions to be taken without human involvement.

  E. The system monitors and correlates data based on business rules and policies that allow actions to be taken without human involvement.

- What is the action taken?

  A. The operator performs the required procedures and issues the commands on each affected resource individually.

  B. The operator performs the required procedures and issues the commands on a centralized management console.

  C. The operator approves and initiates the recovery actions.

D. The autonomic system initiates the recovery actions. No human action is needed.

E. Same as D. Chosen only if the answer to the "How is the disturbance analysed?" question is also E, i.e. when the system monitors and correlates data based on business rules and policies that allow actions to be taken without human involvement.

Each disturbance shall produce an autonomic maturity score based on the average of the 3 answers above. The autonomic maturity score for every question for every disturbance shall be disclosed.

The overall Autonomic Recovery Index is the average score of all injection slots normalized to the highest autonomic level possible (i.e. 4). The result is a value between 0 and 1.

A value of 0 indicates that the autonomic capabilities of the system are basic (manually managed by reports, product manuals, and manual actions). A value of one indicates that the system is autonomic (automatically manages itself to achieve business objectives).

Measurement, formula, and data element computations – For each disturbance, take the average point score for the questions and then divide by the maximum score of 4.

Interpretation of measured value - 0 <= x <= 1. The closer to 1.0 is the better.

Measure scale type –- Absolute

Measure type – Count

Input to measurement – User monitoring record

ISO/IEC 12207:2008 Reference – 6.4.5 System Integration Process, 6.4.6 System Qualification Testing process, 6.4.9 Software Operation Process

Target audience – Acquirer, Supplier, Developer, Maintainer

5.4.4 Quality Measure Element (QME) - Number of transactions under disturbance

Table 7 – QME – Number of transactions under disturbance

Information	Description
Specific QME name	Number of transactions under disturbance
Specific QME id	
Definition	Number of transactions completed without error within a measurement interval where disturbance(s) were injected
Measurement method	Count
Detail	Obtain the transaction counts from the operation or test report of the workload. See 5.4.1.
Documentation	
Aspect – Measurement scale	Ratio
Aspect – Measurement focus	External
Aspect – Measurement method	Objective
Inputs	Operation report Test report
Used for	Resiliency

5.4.5 Quality Measure Element (QME) - Number of transactions under no disturbance

Table 8 — QME – Number of transactions under no disturbance

Information	Description
Specific QME name	Number of transactions under no disturbance
Specific QME id	
Definition	Number of transactions completed without error within a measurement interval where disturbance(s) were not injected. See 5.4.1 of ISO/IEC 25045.
Measurement method	Count
Detail	Obtain the transaction counts from the operation or test report of the workload.
Documentation	
Aspect – Measurement scale	Ratio
Aspect – Measurement focus	External
Aspect – Measurement method	Objective
Inputs	Operation report Test report
Used for	Resiliency

5.4.6 Quality Measure Element (QME) - Autonomic Maturity Score

Table 9 — QME – Autonomic Maturity Score

Information	Description
Specific QME name	Autonomic Maturity Score
Specific QME id	
Definition	This is a score based on the answer to a question in the autonomic maturity questionnaire by assigning a different numeric score to one of six answers.
Measurement method	Count
Detail	See 5.4.2 of ISO/IEC 25045
Documentation	
Aspect – Measurement scale	Ratio
Aspect – Measurement focus	External
Aspect – Measurement method	Subjective
Inputs	Operation report Test report Monitoring of the test
Used for	Autonomic Recovery Index

6 Interpretation of results

6.1 Mapping of measures

Both Resiliency and Autonomic Recovery Index have a value from 0 to 1, and the closer to 1, the better the result.

6.2 Reporting

The report provides the following.

- An executive summary about the result. Describe an overview of the software systems being tested, and a summary of the result and key findings.
- A score card listing the score of every disturbance and the overall score for resiliency and autonomic recovery index.
- A summary of the reaction to disturbances by listing the ones that caused a crash (i.e. a system or software component shutdown), a hang, (i.e. system does not response), invalid result, degrade performance, and the ones that had no noticeable impact to performance (i.e. has no impact to resiliency).
- Description of findings and recommendation, including strength and weakness or opportunity for improvement for the components that form the SUT.
- Describe the products used in the SUT, including the software, hardware, OS, and network,
- Data elements specified in 5.3 and quality measures specified in 5.4.
- The report shall also provide graphs for the baseline and the test for each disturbance with X-axis for time and Y-axis for transaction rate to visualize each measurement interval and injection slot.
- The questionnaire used in the evaluation of the autonomic recovery index shall be included. Each section of the report will include extract of the relevant section in clause 5 to provide a context.

For a sample report, please see Annex A.

6.3 Application Procedure

The application procedure is not applicable to this International Standard.

Annex A
(informative)

Sample Report

A.1 Executive Summary

Describe an overview of the software systems being tested, and a summary of the result and key findings.

NOTE The scope of each clause is described in italic. In an actual report they can be deleted.

This document contains the results and product recommendations from the recoverability testing of the open source Day Trader 2.0 application on a software system solution composing of HHHH v1.6 HTTP server, AAAA v1.6 application server, and DDDD v6.0 DBMS server.

This software system solution achieves a resiliency score of 0.81 (out of 1.00) and a autonomic recovery index score of 0.63 (out of 1.00). This is an improvement over the previous recoverability test score of 0.75 and 0.55 respectively for Day Trader 1.0 application on HHHH v1.5 HTTP server, AAAA v1.6 application server, and DDDD v5.0 DBMS server.

The improvement in the resiliency score is due to the improvement in the application to distribute work to other nodes in the cluster when one of its nodes goes down. The improvement in the autonomic recovery index score is due to the automatic failover of the failed node to the rest of the cluster with no operator intervention.

More details can be found in the following clauses in the report.

A.2 Score Card

A score card listing the score of every disturbance and the overall score for resiliency and autonomic recovery index

This scorecard shows the contribution that each fault made to the Resiliency and Autonomic Recovery Index scores of the system. The Resiliency score is a value from 0 to 1 that reflects the ability of the system to service requests while the disturbance is applied. The Autonomic Recovery Index contribution is the value from 0 to 4 that describes the level of autonomic maturity exhibited by the system while the disturbance is applied.

NOTE The following numbers are fictitious.

Table 10 — Score Card

Disturbance	Contribution to Resiliency Score	Contribution to Autonomic Recovery Index Score	Comment
Unexpected Shutdown			
0101 Process shutdown- DBMS	0.64	1	
0102 Process shutdown- Message Server	0.63	1	
0103 Process shutdown HTTP Server	1.00	3	
0104 Process shutdown – Application Server	0.98	3	
0105 OS shutdown- DBMS	0.51	1	
0106 OS shutdown-Message Server and HTTP Server	0.51	1	
0107 OS shutdown - Application Server	0.99	3	
Loss of Data			
0201 Loss of table- DBMS	0.39	0	
0202 Loss of file- DBMS	0.44	0	
0203 Loss of disk- DBMS	1.00	3	
0204 Loss of disk-Message Server	0.44	0	
Resource Contention			
0301 CPU hog-DBMS	1.00	3	
0302 CPU hog-Message Server and HTTP Server	0.91	0	
0303 CPU hog –WAS	0.91	0	
0304 Mem hog-DBMS	0.89	0	
0305 Memory hog-Message Server and HTTP Server	0.96	0	
0306 Mem hog-WAS	1.00	3	
0307 I/O hog – DBMS	1.00	3	
0308 Disk hog-DBMS	0.99	3	
0309 Disk hog-Message Server	1.00	3	
0310 Runaway query – DBMS	1.00	0	
0311 Poison message – Message Server	0.71	0	
Load Resolution			
0401 Moderate load surge with 2x users	0.89	2	
0402 Significant load surge with 10x users	0.50	2	
Detection of Restart Failure			
0501 Process restart-DBMS	0.63	0	
0502 Process restart-Message Server	0.59	0	
0503 Process restart-HTTP Server	1.00	3	
0504 Process restart - Application Server	1.00	3	
0505 OS restart-DBMS	0.36	0	
0506 OS restart-Message Server	0.42	0	
0507 OS restart – Appl Server	0.99	0	
0508 OS restart – HTTP Server	0.99	0	
Combined Score	0.81	0.63 (2.52/4)	

A.3 Summary of Reaction to Disturbances

A summary of the reaction to disturbances by listing the ones that caused a crash (i.e. a system or software component shutdown), a hang, (i.e. system does not response), invalid result, degrade performance, and the ones that had no noticeable impact to performance (i.e. has no impact to resiliency)

Table 11 — Summary of Reaction to Disturbances

Reaction to disturbances	Number of disturbances	Disturbance
Crash	18	Xxxx, xxxx, xxxx, ….
Hang	1	0311 Disk hog for DBMS server (Table run out of storage)
Invalid result	0	
Degrade performance	19	Xxxx, xxxx, xxxx, ….
No impact to performance	6	Xxxx, xxxx, xxxx, ….
Improve performance	0	
Total	44	

A.4 Findings and Recommendations

Describe a list of findings and recommendation, including strength and weakness or opportunity for improvement for the components that form the SUT.

A.4.1 Application Server

Strength:

- When an application server within a cluster is down, the workload quickly rebalance among the remaining servers with minimal failure of transactions.

Weakness / Opportunity for Improvement:

- The cluster does not detect and bypass “sick” application server that suffers a loss of capacity due to various disturbances. This results in a loss of overall capacity of the cluster since too much work is directed to servers with slow response time.

A.4.2 Defects

A list of application and product defects identified and their current status.

N/A

A.5 Input for the evaluation

A.5.1 The SUT description

A.5.1.1 Specification of the hardware architecture and configuration

The properties of the hardware architecture and configuration shall be described in sufficient details to allow replication of the hardware and OS configuration. These include but not limited to the following:

- *vendor and model number;*
- *system availability date;*
- *CPU (processor type, number and speed (MHz/GHz) of the CPUs);*
- *cache (L1, L2, L3, etc);*
- *main memory (in Megabytes);*
- *disks and file system used;*
- *network interface;*
- *number of systems with this exact same configuration;*
- *OS (product name, vendor, and availability date);*
- *OS tuning parameters and options changed from the defaults;*
- *compilation and linkage options and run-time optimizations used to create/install OS if different from defaults*
- *tuning information if differ from defaults*
- *logical or physical partitioning used on this system to host software instances;*
- *which software components, application, and additional software from 5.2.1.2, 5.2.1.3, and 5.2.1.4 run on this hardware.*

Database server hardware:

Number of Systems:	1
Hardware Vendor:	IBM Corporation
Model Name:	IBM System p5 550
Processor:	POWER5+
MHz:	2100
Number of CPUs:	4 cores, 2 chips, 2 cores/chip (SMT on)
Memory (MB):	16384
L1 Cache:	64KB(I)+32KB(D) on chip per core
L2 Cache:	1920KB on chip per chip
Other Cache:	36MB off chip per DCM, 2 DCMs per SUT
OS Vendor:	IBM
OS Name:	IBM AIX 5L v5.3
Disks:	1x36GB SCSI, 10K RPM
Network Interface:	2 built-in Gigabit Ethernet ports
Other Hardware:	1 x IBM 4Gb dual-port Fibre Channel HBA connected to two IBM System Storage DS4700 storage controllers containing 28 x 36GB disk drives
H/W Available:	Aug-2006
O/S Available:	Aug-2006

Notes / Tuning Information

```
vmo -o lgpg_regions=454 -o lgpg_size=16777216 -o v_pinshm=1
aioo -o maxservers=100 -o maxreqs=16384 -o fsfastpath=1
```

<Add additional hardware for other components in the SUT, such as application server, test load simulator, fault kit driver, Ethernet switch, etc.>

A.5.1.2 Specification of the system software configuration

The properties of the software components such as web server, application server, message server, database server, JVM, etc, that the applications use shall be described in sufficient details to allow replication of the software configuration. These include but not limited to the following:

- *vendor name, product name and version, and availability date;*
- *tuning parameters and options changed from the defaults;.*
- compilation and linkage options and run-time optimization used to create/install the software component;
- number of instances on each system.

Database server software:

Number of Instance:	1 (1 instance per database server hardware)
Software Vendor:	IBM Corporation
Product Name:	IBM DB2 Universal Database 9.1
Availability Date:	Feb-2007

Notes / Tuning Information:
Tuning performed by db2tune.sh script. See attached appendix XXX ……<This is just an example. Appendix XXX does not exist.>

<Add additional software for other components in the SUT, such as application server, HTTP server, test load simulator, fault kit driver, etc.>

A.5.1.3 The application programs

All programs used by the emulated users shall be presented on a digital storage medium. These programs shall be ready for use on the SUT (either as an executable program or the complete source code). They shall be described in sufficient details to allow replication of the software configuration. These include but not limited to the following:

- *vendor name, product name and version, and availability date;*
- *tuning parameters and options changed from the defaults;*
- *compilation and linkage options and run-time optimization used to create/install the software component;*
- number of instances on each system

DayTrader 2.0

- The application used by this test is the open source Day Trader 2.0 application available from the Apache Geroninmo site at http://cwiki.apache.org/GMOxDOC10/day-trader.html . It was downloaded and stored in server XXXX under the directory /xxxxx/yyy/daytrader. Please refer to /xxxxx/yyy/daytrader in server XXXX for the make file and the configuration parameters used. One instance of DayTrader was started on server XXXX.

A.5.1.4 Additional software required

A list of all additional software components or standard system software modules which are needed to run shall be described in sufficient details to allow replication of the software configuration. These include but not limited to the following:

- *vendor name, product name and version, and availability date;*
- *tuning parameters and options changed from the defaults;*
- *compilation and linkage options and run-time optimization used to create/install the software component;*
- *number of instances on each system.*

For the baseline run, this shall include the test driver that simulates the multiple users and drives the test scripts.

For the test run, this shall include the fault injection software.

Performance workload tester:

Number of Instance:	2 (1 instance per test driver server hardware)
Software Vendor:	IBM Corporation
Product Name:	Rational Performance Tester 8.1
Availability Date:	April-2009

Fault load driver kit:

Number of Instance:	2 (1 instance per fault load server hardware)
Software Vendor:	Locally developed by department XXXX
Product Name:	Resiliency Tester 1.0
Availability Date:	April-2009

A.5.1.5 The stored data

All data, which are needed by the programs for their correct working or which have any influence on the performance of the SUT so long as they are not contained in the descriptions of the task type input, shall be presented in their entirety on digital storage medium. They shall be formatted ready for immediate use and storage on the SUT without any further modification. Examples of such data can be:

- *data files, required for a correct computation;*
- *output data files used by the programs, which are not empty when starting the test;*
- *the data of a data base system.*

The database is generated by the Day Trader application for 500,000 accounts as part of its data generation procedure.

A.5.1.6 Additional information for proof

It is the responsibility of the tester to submit the results of the measurement to proof. Therefore the tester shall supply additional documents of his own choice in addition to the documents requested in this International Standard, which are suitable to repeat the measurement by an external person/group to attain the same results.

N/A

A.6 The workload description

A.6.1 The workload specification

Describe the workload in sufficient details to allow replication of the software configuration. These include but not limited to the following:

- *description of the transaction or operation that will be tested;*
- *description of the test data;*
- *the test scripts;*

The application used by this test is the open source Day Trader 2.0 application available from the Apache Geroninmo site at http://cwiki.apache.org/GMOxDOC10/day-trader.html. See the website for the workload specification.

A.6.2 The workload parameter set

The values for the set of parameters used to drive the workload shall be described in sufficient details to allow replication of the software configuration. This shall include any configuration parameters for the workload driver and the application that could affect the performance and application behaviour. These include but not limited to the following:

- *the total number of user;*
- *the duration of the baseline run (in seconds), including the duration of the ramp up period, steady state, and shutdown period;*
- *the reporting interval (in seconds) where the transaction rate is reported during the run;*
- *the transaction mix of the workload (e.g. 10% New Order, 20% Status Query, etc);*
- *whether the transaction mix is maintained at each reporting interval, and if not, describe when each type of transactions is executed during the run;*
- *other configuration changes that determine how the workload will be run that might affect the repeatability and performance of the run;*
- *any configuration changes on the application that might affect the repeatability and performance of the run.*

Number of users = 100

Duration of baseline run = 3300 seconds

Reporting interval = 30 seconds

Transaction mix = 70/30 read/write transaction (normal mix)

Transaction mix maintained in each reporting interval during steady state, with no more than 5% variation.

A.6.2.1 Parameter set for proving the consistency and stability of the workload

In order to properly evaluate the effect of disturbance injection, the baseline must be shown to be repeatable and consistent. The baseline shall be run 3 times and the following values shall be provided as proof of its repeatability and consistency.

- *The required statistical significance of the measurement results as defined by the acquirer (e.g. the number of successful transactions completed in each of 3 baseline runs should not differ by more than 5%).*
- *The reported statistical significance of the measurement results.*
- *Identify any significant performance spikes and dips in any reporting intervals during the run that are greater than the required statistical significance defined by the acquirer. Explain what might be causing those. An explanation of any performance spikes and dips for a reporting interval during the run if they exceed the required statistical significance defined by the acquirer.*

> *Either the presence of spikes and dips in a reporting interval or the presence of different types of transactions at various reporting intervals may indicate a problem. The injection of a disturbance at such intervals could produce variation in performance and quality measures. This shall be noted in the report. However, as long as their occurrences are consistent in multiple runs the results are comparable.*

3 baseline runs were performed with the following result:

- Run 1 - 15000.45 page/seconds
- Run 2 - 15600.67 pages/seconds
- Run 3 - 15580.45 pages/seconds

The runs are consistent and stable since their differences between the lowest and highest results are within the limit at 3.8%.

A.7 The fault load description

A.7.1 The fault load specification

The list of the disturbances to be executed against the workload shall be described comprehensively and grouped by categories of disturbance as defined in 5.1.2. Disturbances shall be defined for software components that the application depends on such as the web server, application server, message server, the database server, etc.

A.7.1.1 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 12 — Disturbances for unexpected shutdown

Disturbance	Target	Fault Description
0101 Abrupt OS shutdown on the DBMS server	DBMS	Psshutdown \\<hostname>
0102 Abrupt OS shutdown on the application server	Appl Server	Psshutdown \\<hostname>
0103 Abrupt OS shutdown on the messaging server	Msg Server	Psshutdown \\<hostname>
0104 Abrupt OS shutdown on the HTTP server	HTTP	Psshutdown \\<hostname>
0105 Abrupt process shutdown of DBMS	DBMS	pskill \\xxx yyyy.exe
0106 Abrupt process shutdown of application server	Appl Server	pskill \\aaa bbbb.exe
0107 Abrupt process shutdown of messaging server	Msg Server	pskill \\ccc dddd.exe
0108 Abrupt process shutdown of HTTP server	HTTP Server	pskill \\eee ffff.exe
0109 Abrupt network shutdown on the DBMS	DBMS	ifconfig eth0 down
0110 Abrupt network shutdown on the application server	Appl Server	ifconfig eth0 down
0111 Abrupt network shutdown on the messaging server	Msg Server	ifconfig eth0 down
0112 Abrupt network shutdown on the HTTP server	HTTP Server	ifconfig eth0 down

A.7.1.2 Loss of Data

Disturbances in this category simulate a scenario in which business-critical data is lost.

Table 13 — Disturbances for resource contention

Disturbance	Target	Fault Description
0201 Loss of table	DBMS	DBMS drop table
0202 Loss of database file	DBMS	forcedel
0203 Loss of disk for database table	DBMS	Ipssend setstate DDD (disable a disk in a RAID)
0204 Loss of disk for message queue	Msg Server	forcedsk

A.7.1.3 Resource Contention

Disturbances in this category simulate the case in which resources on a machine in the SUT are exhausted because of an unexpected process, user action, or application error.

Table 14 — Disturbances for resource contention

Disturbance	Target	Fault Description
0301 CPU hog on DBMS server	DBMS	CpuHog.exe - for(;;);
0302 CPU hog on application server	Appl Server	CpuHog.exe - for(;;);
0303 CPU hog on messaging server	Msg Server	CpuHog.exe - for(;;);
0304 CPU hog on HTTP server	HTTP Server	CpuHog.exe - for(;;);
0305 Memory hog on DBMS server	DBMS	MemHog.exe - pHuge = malloc(HUGE_VALUE)
0306 Memory hog on application server	Appl Server	MemHog.exe - pHuge = malloc(HUGE_VALUE)
0307 Memory hog on messaging server	Msg Server	MemHog.exe - pHuge = malloc(HUGE_VALUE)
0308 Memory hog on HTTP server	HTTP Server	MemHog.exe - pHuge = malloc(HUGE_VALUE)
0309 I/O hog on DBMS server	DBMS	IoHog.exe - copy or write BIG_FILE across the database disks.
0310 Disk hog for messaging server (Message queue run out of storage)	Msg Server	Diskhog.exe
0311 Disk hog for DBMS server (Table run out of storage)	DBMS	Diskhog.exe
0312 Deadlock for DBMS server	DBMS	Deadlock.exe <resource1> <resource2>
0313 Memory hog of an user application	Appl Server	LeakCity.ear
0314 Run away query inside DBMS	DBMS	Bigquery.exe
0315 Poison Message	Msg Server	Bigmsg.exe
0316 Network hog on DBMS server	DBMS	NetworkHog.exe <hostname>
0317 Network hog on application server	Appl Server	NetworkHog.exe <hostname>
0318 Network hog on messaging server	Msg Server	NetworkHog.exe <hostname>

A.7.1.4 Load Resolution

Disturbances in this category simulate a sudden increase in the workload on the system

Table 15 — Disturbances for load resolution

Disturbance	**Target**	**Fault Description**
0401 Moderate load surge with 2x users	All	Stress.exe -client TWO_TIMES
0402 Significant load surge with 10x users	All	Stress.exe -client TEN_TIMES

A.7.1.5 Detection of Restart Failure

Disturbances in this category simulate a situation in which an application or the component it depends on is corrupted and cannot be restarted.

Table 16 — Disturbances for restart failure

Disturbance	**Target**	**Fault Example**
0501 OS restart failure of the DBMS server	DBMS	rm boot.ini
0502 OS restart failure of the application server	Appl Server	rm boot.ini
0503 OS restart failure of the messaging server	Message Server	rm boot.ini
0504 OS restart failure of the HTTP server	HTTP	rm boot.ini
0505 Restart failure of the DBMS	DBMS	Rm DBMSsyscs.exe
0506 Restart failure of the application server	Appl Server	Rm startserver.bat
0507 Restart failure of the queue manager	Message Server	rm strmqm
0508 Restart failure of the HTTP server	HTTP	rm admin.conf

Comment: These tests are executed by repeating the Unexpected Shutdown faults and renaming a key file in the particular component. It does not test the case where a corrupted database backup causes the database restart to fail.

A.7.2 The fault load parameter set

Each disturbance shall be run within an injection slot as defined in 5.1. The following values used in the injection slot shall be described:

- *measurement interval (in seconds), also known as the duration of injection slot;*
- *injection interval (in seconds);*
- *detection interval (in seconds).*

If multiple disturbances are to be executed in sequence without stopping and restarting the workload, the disturbances and their sequence of execution shall be described.

The following fault testing parameters were used by the disturbance testing unless specified otherwise in the individual test details:

- Measurement interval = 3000 seconds
- Injection interval = 600 seconds
- Detection interval = 1200 seconds

All disturbances were executed with a stopping and starting of the workload.

A.8 The Autonomic Maturity Questionnaire

A questionnaire composing of the questions defined in 5.4.3 shall be created for each disturbance to be executed.

Done. See A.11.

A.9 Output from the baseline run

This is the output from a normal execution of the workload against the system. The output shall provide the following:

- *the workload parameter set that was used in the run and recorded in the output;*
- *measurement interval;*
- *reporting interval (e.g. every 30 seconds, or every 10 minutes, etc, throughout the workload);*
- *total transaction completed without error within the measurement interval, and between reporting interval;*
- *total transaction completed with errors within the measurement interval;*
- *other system performance information including the CPU utilization and i/o utilization of major components used in the SUT (e.g. web server, application server, database server, etc) at each interval for comparison with a SUT.*

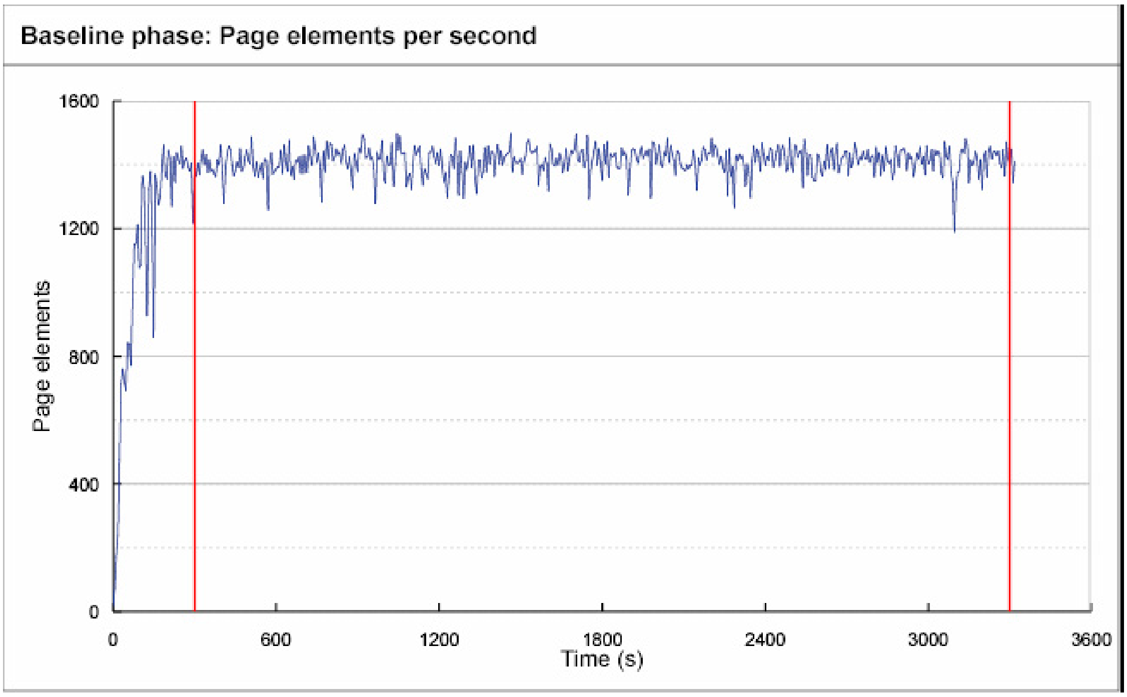

Figure 4 — Baseline phase: Page element per second

Workload parameters:

- Number of users = 100
- Transaction mix = 70/30 read/write transaction (normal mix)
- Ramp up = 300 seconds
- Measurement Interval = 3000 seconds (total test interval minus ramp-up = 3300 – 300 = 3000 seconds
- Reporting interval = 30 seconds

Workload result:

- Throughput = 1500.45 pages/second
- Total number of transaction executed over measurement interval = 4,505,374
- Total number of transaction successful = 4,501,350
- Total number of transaction failed = 4,024 (0.9% due to normal deadlock, etc, no unexpected errors)

Utilization:

Table 17 — Baseline phase: Average CPU and I/O utilization

Component	Average CPU Utilization	Average I/O Utilization
HTTP Server	80%	5%
Application Server	80%	10%
Message Server	50%	20%
Database Server	80%	20%

A.10 Output from the test run, and results of quality measures

This is the output from the workload against the system while under disturbance Injections. This output shall provide the following for each injection slots:

- *the workload parameter set that was used in the run and recorded in the output;*
- *measurement interval (in seconds);*
- *injection Interval (in seconds);*
- *whether manual recovery was used to disable the disturbance and recover the system to the steady state like that of a baseline run;*
- *reporting interval (in seconds, e.g. every 30 seconds, or every 10 minutes, etc throughout the test);*
- *total transaction completed without error within the measurement interval and between each reporting interval;*
- *total transaction completed with errors within the measurement interval;*
- *other system performance information including the CPU utilization and i/o utilization of major components used in the SUT (e.g. web server, application server, database server, etc) at each reporting interval for comparison with a baseline test;*
- *Whether the check phase was performed to verify the integrity of the system. If performed, state the method of verification and result. If not performed, state why this is not necessary.*
- *The quality measure of resiliency shall be reported for each disturbance and over the entire set of disturbances.*

The Quality Measure Resiliency is a quantitative measure of the quality of service under test. It describes the relationship between the throughput when the system is fouled with a disturbance and the throughput when it is not infected.

For each disturbance, calculate

$\boldsymbol{P_i / P_{base}}$ *where*

P_i = *number of transactions completed without error within a measurement interval of an injection slot where disturbance(s) were injected.*
P_{base} = *number of transactions completed without error within the measurement interval (of a baseline run where disturbances(s) were not injected.*

The overall Resiliency is calculated by taking the average of the Resiliency for each injection slot.

The quality measure Autonomic Recovery Index is a qualitative measure of the level of autonomic capability for recovery. Five levels of autonomic maturity are defined in the table below. For each disturbance, the evaluation is based on how the problem is detected, analyzed, and resolved according to a set of questions and the answer is mapped to a score associated with the autonomic maturity level. See A.11 for more details.

NOTE Only the result from one disturbance is shown here as a sample.

A.10.1 Unexpected Shutdown

A.10.1.1 Abrupt OS shutdown for DBMS

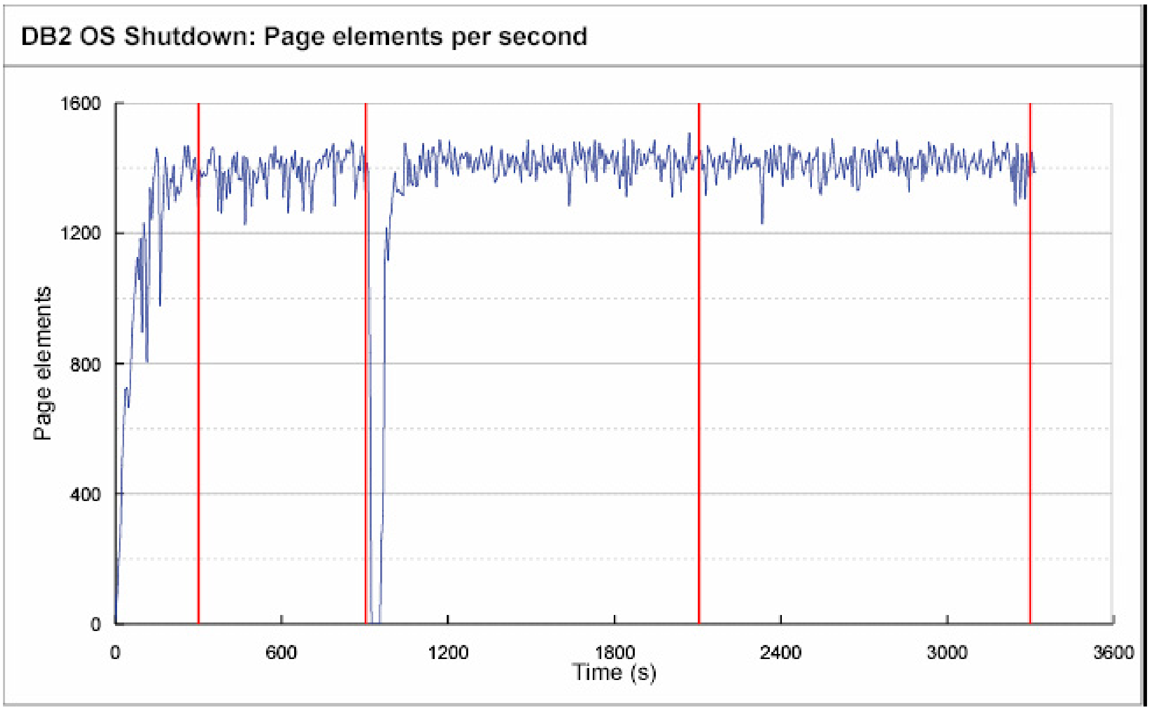

Figure 5 — DBMS OS Shutdown: Page elements per second

Workload parameters:

- Same as baseline run

Workload result:

- Throughput = 1440.43 pages/second
- Total number of transaction executed over measurement interval = 4,502,075
- Total number of transaction successful = 4,321,290
- Total number of transaction failed = 180,785 (due to database server down, normal deadlock, etc, nothing unexpected)

Utilization:

- CPU and I/O graphs can be found in server X directory /xxx/yyy/911

Check phase:

- Performed by comparing application log, test report, and the row count of tables provided by dbcheck.exe. The result matched. The output of all three can be found in server X, directory /xxxx/yyyy/911.

Quality measures:

- Resiliency = 4321290 / 4501350 = 0.96
- Autonomic Recovery Index = 3.33 / 4.0 = 0.83

Analysis and comment:

- The database implemented AIX HACMP for failover and DBMS HADR for a hot standby database.
- HACMP detects the shutdown of the OS and automates failover to the standby HADR database.
- HACMP also sends an email notifying the DBA of the failover, and advising that the DBA restart the failed primary machine

A.10.X.X <add result for other disturbances>

……

A.10.X <add result for another disturbance category>

……

A.10.X.X <add result for other disturbances>

……

A.11 Completion of the Autonomic Maturity Questionnaire

The Autonomic Recovery Index is a qualitative measure of the level of autonomic capability for recovery.

Measure Name -- Autonomic Recovery Index

Purpose of Measure -- How well does the software product detect, analyse, and resolve disturbances?

Method of application –

> *For each disturbance, observe the behaviour of the system in detecting, analysing, and resolving the disturbance which are faults or events, then answers a set of questions from a questionnaire to obtain the score.*
>
> *The score for each disturbance shall be calculated based on answers to the questions on the Autonomic Maturity Questionnaire answered by the test operator. Each answer is given a value based on the increasing autonomic level of the response of the system as follows:*

Table 18 — Autonomic Level

Autonomic Level	***Description***
Basic	*Rely on reports, product, and manual actions to manage IT components*
Managed	*Management software in place to provide facilitation and automation of IT tasks*
Predictive	*Individual components and systems management tools able to analyze changes and recommend actions*
Adaptive	*IT components collectively able to monitor, analyze, and take action with minimal human intervention*
Autonomic	*IT components collectively and automatically managed by business rules and policies*

Each question shall have one of six answers each with a different score that correspond to the Autonomic Level as follows:

- *A is awarded 0 points (basic)*
- *B0 is awarded 0.5 points (basic/managed)*

- *B is awarded 1 point (managed)*
- *C is awarded 2 points (predictive)*
- *D is awarded 3 points (adaptive)*
- *E is awarded 4 points (autonomic)*

Note: The value of the points awarded could be adjusted based on experience, customer preference, and context.

The following questions shall be used for each disturbance:

How is the disturbance detected?

A. The help desk calls the operators to tell them about a rash of complaints.

B0. The operators detect the problem themselves by monitoring multiple data sources.

B. The operators detect the problem themselves by monitoring a single data source.

C. The autonomic manager notifies the operator of a possible problem.

D. The autonomic manager detects the problem without human involvement.

How is the disturbance analyzed?

A. The operator collects and analyzes multiple sources of system-generated data.

B. The operator analyzes data from a single management tool.

C. The system monitors and correlates data that leads to recommended recovery actions.

D. The system monitors and correlates data that allows actions to be taken without human involvement.

E. The system monitors and correlates data based on business rules and policies that allow actions to be taken without human involvement.

What is the action taken?

A. The operator performs the required procedures and issues the commands on each affected resource individually.

B. The operator performs the required procedures and issues the commands on a centralized management console.

C. The operator approves and initiates the recovery actions.

D. The autonomic system initiates the recovery actions. No human action is needed.

Each disturbance shall produce an autonomic maturity score based on the average of the 3 answers above. The autonomic maturity score for every question for every disturbance shall be disclosed.

The overall Autonomic Recovery Index is the average score of all injection slots normalized to the highest autonomic level possible (i.e. 4). The result is a value between 0 and 1.

A value of 0 indicates that the autonomic capabilities of the system are basic (manually managed by reports, product manuals, and manual actions). A value of one indicates that the system is autonomic (automatically manages itself to achieve business objectives).

A.11.1 Unexpected Shutdown

Table 19 — Questionnaire result for unexpected shutdown

Disturbance	Detect	Analyze	Action	Score	Comment
0101 Abrupt OS shutdown on the DBMS server					
0102 Abrupt OS shutdown on the application server					
0103 Abrupt OS shutdown on the messaging server					
0104 Abrupt OS shutdown on the HTTP server					
0105 Abrupt process shutdown of DBMS					
0106 Abrupt process shutdown of application server					
0107 Abrupt process shutdown of messaging server					
0108 Abrupt process shutdown of HTTP server					
0109 Abrupt network shutdown on the DBMS					
0110 Abrupt network shutdown on the application server					
0111 Abrupt network shutdown on the messaging server					
0112 Abrupt network shutdown on the HTTP server					

A.11.1.1 Loss of Data

Table 20 — Questionnaire result for loss of data

Disturbance	Detect	Analyze	Action	Score	Comment
0201 Loss of table					
0202 Loss of database file					
0203 Loss of disk for database table					
0204 Loss of disk for message queue					

A.11.1.2 Resource Contention

Table 21 — Questionnaire result for resource contention

Disturbance	Detect	Analyze	Action	Score	Comment
0301 CPU hog on DBMS server					
0302 CPU hog on application server					
0303 CPU hog on messaging server					
0304 CPU hog on HTTP server					
0305 Memory hog on DBMS server					
0306 Memory hog on application server					
0307 Memory hog on messaging server					
0308 Memory hog on HTTP server					
0309 I/O hog on DBMS server					
0310 Disk hog for messaging server (Message queue run out of storage)					
0311 Disk hog for DBMS server (Table run out of storage)					
0312 Deadlock for DBMS server					
0313 Memory hog of an user application					
0314 Run away query inside DBMS					
0315 Poison Message					
0316 Network hog on DBMS server					
0317 Network hog on application server					
0318 Network hog on messaging server					

A.11.1.3 Load Resolution

Table 22 — Questionnaire result for load resolution

Disturbance	Detect	Analyze	Action	Score	Comment
0401 Moderate load surge with 2x users					
0402 Significant load surge with 10x users					

A.11.1.4 Detection of Restart Failure

Table 23 — Questionnaire result for detection of restart failure

Disturbance	Detect	Analyze	Action	Score	Comment
0501 OS restart failure of the DBMS server					
0502 OS restart failure of the application server					
0503 OS restart failure of the messaging server					
0504 OS restart failure of the HTTP server					
0505 Restart failure of the DBMS					
0506 Restart failure of the application server					
0507 Restart failure of the queue manager					
0508 Restart failure of the HTTP server					

Autonomic Recovery Index = Average of all Scores / Highest autonomic score possible = (   ) / 4.0 = (   )

Bibliography

[1] ISO/IEC 9126-1:2001[3], *Software engineering — Product quality — Part 1: Quality model*

[2] ISO/IEC TR 9126-2:2003[4], *Software engineering — Product quality — Part 2: External metrics*

[3] ISO/IEC TR 9126-3:2003[5], *Software engineering — Product quality — Part 3: Internal metrics*

[4] ISO/IEC TR 9126-4:2004[6], *Software engineering — Product quality — Part 4: Quality in use metrics*

[5] ISO/IEC 12207:2008, *Systems and software engineering — Software life cycle processes*

[6] ISO/IEC 14598-1:1999[7], *Information technology — Software product evaluation — Part 1: General overview*

[7] ISO/IEC 14598-6:2001, *Software engineering — Product evaluation — Part 6: Documentation of evaluation modules*

[8] ISO/IEC 15026:1998, *Information technology — System and software integrity levels*

[9] ISO/IEC 15288:2008, *Systems and software engineering — System life cycle processes*

[10] ISO/IEC 15939:2007, *Systems and software engineering — Measurement process*

[11] ISO/IEC 25001:2007, *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Planning and management*

[12] ISO/IEC 25010:—[8], *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Quality model and guide*

[13] ISO/IEC 25020:2007, *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Measurement reference model and guide*

[14] ISO/IEC TR 25021:2007, *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements*

[15] ISO/IEC 25040:—[8], *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation reference model and guide*

[16] ISO/IEC 25051:2006, *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing*

[3] Upon publication, ISO/IEC 25010 will replace ISO/IEC 9126-1.

[4] Upon publication, ISO/IEC 25023 will replace ISO/IEC 9126-2.

[5] Upon publication, ISO/IEC 25022 will replace ISO/IEC 9126-3.

[6] Upon publication, ISO/IEC 25024 will replace ISO/IEC 9126-4.

[7] Upon publication, ISO/IEC 25040 will replace ISO/IEC 14598-1.

[8] To be published.

ICS 35.080

Price based on 37 pages