INTERNATIONAL STANDARD

ISO/IEC 25023

First edition
2016-06-15

Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality

Ingénierie des systèmes et du logiciel — Exigences de qualité et évaluation des systèmes et du logiciel (SQuaRE) — Mesurage de la qualité du produit logiciel et du système

Reference number
ISO/IEC 25023:2016(E)

Some quality measures produce a result that is relative to a target value that needs to be established as part of requirements.

NOTE 2 Some measurements are normalized against the target value specified in a requirement specification, a design specification, or a user documentation. Such target value is able to be determined and required as the threshold by developers or maintainers to improve architecture, design, implementation, assembles, operational procedures, user interface or performance of the software product or system. The target value is also able to be specified as one of agreed requirements by acquirers and suppliers to specify quality requirements or to examine conformance for acquisition. A requirements specification is usually changed and revised during development and affects the quality measures based on it. Some of requirements to be specified might be missing or inconsistent, or some of the target values might be insufficient and need to be changed because it is very difficult to specify completely both of stated and implied needs derived from stakeholder or system requirements at the beginning of development. Accordingly, users of quality measures are expected to take account of evolving and revising a requirements specification and to apply quality measures not at once but iteratively during development and/or evaluation.

NOTE 3 Some quality measures (such as mean response time) can be difficult to interpret in isolation. The following are ways that quality measures can be applied so that they are easier to understand and interpret:

a) conformance: comparing measures with a specific business or usage requirements (e.g. the maximum acceptable response time is 0,5 seconds);

b) benchmarks: comparing measures with a benchmark for the same or a similar product or system used for the same purpose (e.g. the mean response time of the new system in no more than the mean response time of the old system);

c) time series: comparing trends over time (e.g. how does the mean response time change during the day).

7 Format used for documenting the quality measures

The following information is given for each quality measure in the tables in Clause 8:

a) ID: identification code of quality measure; each ID consists of the following three parts:

— abbreviated alphabetic code representing the quality characteristics as capital X and subcharacteristics as one capital X followed by lowercase x (for example, “PTb” denotes “Time behaviour” measures for “Performance efficiency”);

— serial number of sequential order within quality subcharacteristic;

— G (Generic) or S (Specific) expressing potential categories of quality measure; where, Generic measures can be used whenever appropriate and Specific measures could be used when relevant in a particular situation;

b) Name: quality measure name;

c) Description: the information provided by the quality measure;

d) Measurement function: mathematical formula showing how the quality measure elements are combined to produce the quality measure.

NOTE Useful QMEs which can be used frequently to construct quality measures are specified briefly in Annex B to help comprehend and apply measurement function for the quality measures.

8 System and software product quality measures

8.1 General

The quality measures in Clause 8 are listed by quality characteristics and subcharacteristics in the order used in ISO/IEC 25010.

Quality measures can be used with different evaluation techniques that could be chosen according to quality characteristics and evaluation rating levels depending on whether it is used as internal or external measures. Accordingly, some quality measures listed in Clause 8 can be used at different stages of evaluation such as static review of design specification or dynamic analysis of executable products.

Quality measures, which may be applicable, are not limited to these listed here. It is recommended to refer a specific measure or measurement from specific International Standards or guidelines. For example, functional size measurement is defined in ISO/IEC 14143 and an example of precise time efficiency measurement can be referred from ISO/IEC 14756.

NOTE 1 This list of quality measures is not finalized and might be revised in future versions of this International Standard. Readers of this International Standard are invited to provide feedback.

NOTE 2 In this clause, the word measure means quality measure unless otherwise mentioned. For example, “Functional suitability measures” means “Functional suitability quality measures”.

8.2 Functional suitability measures

Functional suitability measures are used to assess the degree to which a product or system provided functions that meet stated and implied needs when used under specified conditions.

NOTE 1 Functional suitability is concerned with whether the functions meet stated and implied needs.

NOTE 2 A function referred to here could be an elementary process as defined in functional user requirements in ISO/IEC 14143.

NOTE 3 Similar measures with other QMEs like functional size can be defined as a way to weight the result with better accuracy, as unit ratios do not indicate the quantum of functionality that is missing.

8.2.1 Functional completeness measures

Functional completeness measures are used to assess the degree to which the set of functions covers all the specified tasks and user objectives.

Table 1 — Functional completeness measures

ID	Name	Description	Measurement function
FCp-1-G	**Functional coverage**	What proportion of the specified functions has been implemented?	X = 1 – A/B A = Number of functions missing B = Number of functions specified
NOTE 1 Functions can be specified in a requirement specification, a design specification, a user manual or all of these. NOTE 2 A missing function is detected when the system or software product does not have the ability to perform a function that is specified.			

8.2.2 Functional correctness measures

Functional correctness measures are used to assess the degree to which a product or system provides the correct results with the needed degree of precision.

Table 2 — Functional correctness measures

ID	Name	Description	Measurement function
FCr-1-G	**Functional correctness**	What proportion of functions provides the correct results?	X = 1 – A/B A = Number of functions that are incorrect B = Number of functions considered
NOTE 1 An incorrect function is one that does not provide a reasonable and acceptable outcome to achieve the specific intended objective. NOTE 2 The functions considered for evaluation may be all the functions of a product or a specific set of functions required for a particular usage. NOTE 3 Developer or maintainer possibly examines an individual function by reviewing or testing and determines whether the function successfully provides suitable outcomes to specific objectives as defined in the requirements specification or not. In such a case, the degree of correctness is determined per an individual function.			

8.2.3 Functional appropriateness measures

Functional appropriateness measures are used to assess the degree to which the functions facilitate the accomplishment of specified tasks and objectives.

Table 3 — Functional appropriateness measures

ID	Name	Description	Measurement function
FAp-1-G	**Functional appropriateness of usage objective**	What proportion of the functions required by the user provides appropriate outcome to achieve a specific usage objective?	X = 1 – A/B A = Number of functions missing or incorrect among those that are required for achieving a specific usage objective B = Number of functions required for achieving a specific usage objective
NOTE 1 This function will typically be considered for the most important or most frequently identified usage objectives. Thus, this quality measure is first calculated for each of the defined usage objectives that can be pursued in the system, and then the next quality measure, i.e. FAp-2-G "Functional appropriateness of the system", can be calculated collectively across all usage objectives to provide a system measure. NOTE 2 Users of this International Standard could also consider measuring the proportion of user objectives that are achievable in order to get a better understanding of the actual impact on user's intended usage.			
FAp-2-G	**Functional appropriateness of system**	What proportion of the functions required by the users to achieve their objectives provides appropriate outcome?	$X = \sum_{i=1 \text{ to } n} A_i / n$ A_i = Appropriateness score for usage objective i, that is, the measured value of FAp-1-G for i-th specific usage objective n = Number of usage objectives

8.3 Performance efficiency measures

Performance efficiency measures are used to assess the performance relative to the amount of resources used under stated conditions. Resources can include other software products, the software and hardware configuration of the system, and materials (e.g. print paper, storage media).

NOTE 1 The performance efficiency measure is affected strongly and fluctuates depending on the conditions of use, such as load of processing data, frequency of use, number of connecting sites and so on. Therefore, performance efficiency measures might include the ratio of estimated or measured value with error fluctuation to the designed value with allowed error fluctuation range required by specification. It is recommended to list and to investigate the role played by factors such as "CPU" and memory used by other software, network traffic, and scheduled background processes. Possible fluctuations and valid ranges for estimated or measured values can be established and compared to requirement specifications.

NOTE 2 It is also recommended that a task be identified and defined to be suitable for performance efficiency or capacity measures; for example, a transaction as a task for a business application, a switching or data packet sent as a task for a communication application, an event control as a task for a control application and an output of data produced by a user callable function as a task for a common user application.

8.3.1 Time behaviour measures

Time behaviour measures are used to assess the degree to which the response and processing times and throughput rates of a product or system when performing its functions meet the requirements.

Table 4 — Time behaviour measures

ID	Name	Description	Measurement function
PTb-1-G	**Mean response time**	How long is the mean time taken by the system to respond to a user task or system task?	$X = \sum_{i=1 \text{ to } n} (A_i) / n$ A_i = Time taken by the system to respond to a specific user task or system task at i-th measurement n = Number of responses measured
PTb-2-G	**Response time adequacy**	How well does the system response time meet the specified target?	X = A/B A = Mean response time measured by PTb-1-G B = Target response time specified
NOTE 1 Result of a smaller value is better and less than or equal to 1 is good. NOTE 2 Response time is the time from the submission of a request until the first response is produced, i.e. the time it takes to start responding, not the time it take to output the response. NOTE 3 An alternative to this measure is nth percentile response time under expected load conditions. It is also useful to apply it on individual functions or classes of functions.			
PTb-3-G	**Mean turnaround time**	What is the mean time taken for completion of a job or an asynchronous process?	$X = \sum_{i=1 \text{ to } n} (B_i - A_i) / n$ A_i = Time of starting a job i B_i = Time of completing the job i n = Number of measurements
PTb-4-G	**Turnaround time adequacy**	How well does the turnaround time meet the specified targets?	X = A/B A = Mean turnaround time measured by PTb-3-G B = Target turnaround time specified
NOTE 1 Result of a smaller value is better and less than or equal to 1 is good. NOTE 2 In the case of a pipeline (e.g. a systems chain), the elapsed time in each stage of the pipeline has to be considered and bottlenecks in one stage can affect overall turnaround time. NOTE 3 It is recommended to use this measure in conjunction with specified payload and/or workload.			

Table 4 *(continued)*

ID	Name	Description	Measurement function
PTb-5-G	**Mean throughput**	What is the mean number of jobs completed per unit time?	$X = \sum_{i=1 \text{ to } n} (A_i / B_i) / n$ A_i = Number of jobs completed during the i-th observation time B_i = i-th observation time period n = Number of observations
NOTE 1 Jobs could be fine-grained operations like microprocessor operations or coarse grained transaction processing units like those defined by Transaction Processing Performance Council (TPC) or higher level abstractions like functions. So, the results of this measure when used in different contexts should be interpreted appropriately. NOTE 2 Mean throughput is able to be compared to a target threshold of throughput to calculate the throughput adequacy. When such a target threshold under specific condition is specified as one of requirements, the result value is required to be larger than 1.			

8.3.2 Resource utilization measures

Resource utilization measures are used to assess the degree to which the amounts and types of resources used by a product or system when performing its functions meet the requirements.

Table 5 — Resource utilization measures

ID	Name	Description	Measurement function
PRu-1-G	**Mean processor utilization**	How much processor time is used to execute a given set of tasks compared to the operation time?	$X = \sum_{i=1 \text{ to } n} (A_i / B_i) / n$ A_i = Processor time actually used to execute a given set of tasks in observation i B_i = Operation time to perform the tasks in observation i n = Number of observations
NOTE Result value varies from greater than 0 to 1. Usually, the smaller is better.			
PRu-2-G	**Mean memory utilization**	How much of memory is used to execute a given set of tasks compared to the available memory?	$X = \sum_{i=1 \text{ to } n} (A_i / B_i) / n$ A_i = Size of memory actually used to perform a given set of tasks for i-th sample processing B_i = Size of memory available to perform the tasks during i-th sample processing n = Number of samples processed
NOTE Result value varies from greater than 0 to 1. Usually, the smaller is better.			
PRu-3-G	**Mean I/O devices utilization**	How much of I/O device busy time is used to perform a given set of tasks compared to the I/O operation time?	$X = \sum_{i=1 \text{ to } n} (A_i / B_i) / n$ A_i = Duration of I/O device(s) busy time to perform a given set of tasks for i-th observation B_i = Duration of I/O operations to perform the tasks for i-th observation n = Number of observations

Table 5 *(continued)*

ID	Name	Description	Measurement function
NOTE 1 Result value varies from greater than 0 to1. Usually, the smaller is better. NOTE 2 Busy time means the period of time during which a system or a device is actually working.			
PRu-4-S	**Bandwidth utilization**	What proportion of the available bandwidth is utilized to perform a given set of tasks?	X = A/B A = Bandwidth of actual transmission measured over time to perform a given set of tasks B = Bandwidth capacity available to perform a given set of tasks
NOTE 1 In case there is a concern whether the relevant type of resource is well utilized during specific time period or not, for example, to complete specified tasks with maximum resource utilization by avoiding interrupting processing, the result value of closer to optimal is better. In this case, the optimal value depends on the circumstance. NOTE 2 The measurer has to consider the possible communication traffic limitations (e.g. dropping or throttling) which can affect the resulting statistical values including average.			

8.3.3 Capacity measures

Capacity measures are used to assess the degree to which the maximum limits of a product or system parameter meet the requirements.

NOTE 1 Capacity measures are expected to be measured through dynamic analysis, such as volume testing of the system, or can be measured by system integration testing or simulation. Maximum value and distribution of the duration can be investigated for many cases of static analysis, dynamic testing or operations.

NOTE 2 The maximum limit is expected to be specified as a target value which can theoretically be beyond a possible realistic value.

Table 6 — Capacity measures

ID	Name	Description	Measurement function
PCa-1-G	**Transaction processing capacity**	How many transactions can be processed per unit time?	X = A/B A = Number of transactions completed during observation time B = Duration of observation
NOTE 1 Result value varies from 0 to maximum limit. Usually, the larger is better. NOTE 2 This measure can be useful only if there is sufficient workload to test. NOTE 3 Task can be alternatively used, as well as transaction.			
PCa-2-G	**User access capacity**	How many users can access the system simultaneously at a certain time?	$X = \sum_{i=1 \text{ to } n} A_i / n$ A_i = Maximum number of users who can simultaneously access the system at i-th observation n = Number of observations

Table 6 *(continued)*

ID	Name	Description	Measurement function
NOTE Result value varies from 0 to maximum limit. Usually, the result of larger value is better.			
PCa-3-S	**User access increase adequacy**	How many users can be added successfully per unit time?	X = A/B A = Number of users successfully added during observation time B = Duration of observation
NOTE 1 Result value varies from 0 to maximum limit. Usually, the larger is better. NOTE 2 This measure indicates the degree to which the capability of software or system to have enough capacity to accept accesses from a lot of users, even during rapid increase of users in a given moment, e.g. an extremely large number of users could simultaneously access the system or software in an instance through the internet.			

8.4 Compatibility measures

Compatibility measures are used to assess the degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment.

8.4.1 Co-existence measures

Co-existence measures are used to assess the degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.

Table 7 — Co-existence measures

ID	Name	Description	Measurement function
CCo-1-G	**Co-existence with other products**	What proportion of specified software products can share the environment with this software product without adverse impact on their quality characteristics or functionality?	X = A/B A = Number of other specified software products with which this product can co-exist B = Number of other software products specified to co-exist with this product in the operation environment

8.4.2 Interoperability measures

Interoperability measures are used to assess the degree to which two or more systems, products or components can exchange information and successfully use the information that has been exchanged.

Table 13 — User interface aesthetics measures

ID	Name	Description	Measurement function
UIn-1-S	**Appearance aesthetics of user interfaces**	To what extent are user interfaces and the overall design aesthetically pleasing in appearance?	X = A/B A = Number of display interfaces aesthetically pleasing to the users in appearance B = Number of display interfaces
NOTE 1 An internal or external user interface aesthetics quality measure is used to assess the appearance of the user interfaces and will be influenced by factors such as screen design and colour. This is particularly important for consumer products. NOTE 2 Good colour combinations can help users to quickly read the text or identify the image. Then, it can be helpful for better aesthetics measurement to address bad colour combinations, such as light blue on grey, red on orange, green on blue and so on. NOTE 3 This quality measure often depends on an individual of users. Then, either expertise usability designers or testers on behalf of users, or representatives from target user groups are expected to be involved to measure this.			

8.5.6 Accessibility measures

Accessibility measures are used to assess the degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.

NOTE For the additional criteria for accessibility, refer to ISO 9241-171.

Table 14 — Accessibility measures

ID	Name	Description	Measurement function
UAc-1-G	**Accessibility for users with disabilities**	To what extent can potential users with specific disabilities successfully use the system (with assistive technology if appropriate)?	X = A/B A = Number of functions successfully usable by the users with a specific disability B = Number of functions implemented
NOTE 1 Specific disabilities include cognitive disability, physical disability, hearing/voice disability, and visual disability. NOTE 2 The range of capabilities includes disabilities associated with age. NOTE 3 Any person becomes possibly a user with limited cognitive, physical, hearing or visual ability under specific situations or environments, for example, in darkness, in low atmospheric pressure at high altitude, in water and so on.			
UAc-2-S	**Supported languages adequacy**	What proportion of needed languages is supported?	X = A/B A = Number of languages actually supported B = Number of languages needed to be supported
NOTE When users are trying to use a system or software with different language from their own native one, they frequently suffer from operational errors and sometimes give up to achieve their intended goals. Such case is one of decreasing accessibility and caused by misunderstanding of description and messages. Then, it has to be considered, specified and implemented, which languages are to be supported for possible variation of users.			

8.6 Reliability measures

Reliability measures are used to assess the degree to which a system, product or component performs specified functions under specified conditions for a specified period of time.

Internal reliability measures are used for predicting if the completed system/software product in question will satisfy prescribed reliability needs during the development of the system/software product.

External reliability quality measures are used to assess attributes related to the behaviour of the system of which the software is a part during execution testing to indicate the extent of reliability of the software in that system during operation. Systems and software are not distinguished from each other in most cases.

8.6.1 Maturity measures

Maturity measures are used to assess the degree to which a system, product or component meets the needs for reliability under normal operation.

NOTE The concept of maturity can also be applied to other quality characteristics to indicate the degree to which they meet the required needs under normal operation (see ISO/IEC 25010).

Table 15 — Maturity measures

ID	Name	Description	Measurement function
RMa-1-G	**Fault correction**	What proportion of detected reliability-related faults has been corrected?	X = A/B A = Number of reliability-related faults corrected in design /coding/testing phase B = Number of reliability-related faults detected in design/coding/testing phase
NOTE For example, inadequate error handling is a kind of reliability-related faults.			
RMa-2-G	**Mean time between failure (MTBF)**	What is the MTBF during the system/software operation?	X = A/B A = Operation time B = Number of system/software failures actually occurred
NOTE 1 Result value varies from 0 to infinite. Usually, the larger is better. NOTE 2 MTBF itself can be used to compare the reliabilities of different systems or software products.			
RMa-3-G	**Failure rate**	What is the average number of failures during a defined period?	X = A/B A = Number of failures detected during observation time B = Duration of observation
NOTE 1 The period used in this measure could be different for testing and operations purposes, which refers to actual usage or testing time. NOTE 2 A reliability estimation model can use this measure as an input. NOTE 3 The usefulness of this quality measure depends on the adequacy of test cases or the extent of system usage during testing, e.g. normal, exceptional and abnormal cases.			
RMa-4-S	**Test coverage**	What percentage of the system or software capabilities, operational scenarios or functions that are included in their associated test suites are actually performed?	X = A/B A = Number of system or software capabilities, operational scenarios or functions that are actually performed B = Number of system or software capabilities, operational scenarios or functions which are included in their associated test suites

8.6.2 Availability measures

Availability measures are used to assess the degree to which a system, product or component is operational and accessible when required for use.

Table 16 — Availability measures

ID	Name	Description	Measurement function
RAv-1-G	**System availability**	For what proportion of the scheduled system operational time is the system actually available?	X = A/B A = System operation time actually provided B = System operation time specified in the operation schedule
NOTE This measure can be extended to special days, such as holidays and weekend, in addition to regular operational days.			
RAv-2-G	**Mean down time**	How long does the system stay unavailable when a failure occurs?	X = A/B A = Total down time B = Number of breakdowns observed
NOTE 1 Result value varies from 0 to infinite. Usually, the smaller is better. NOTE 2 Externally, availability can be assessed by the proportion of total time during which the system, product or component is in an up state. Availability is therefore a combination of maturity (which governs the frequency of failure), fault tolerance and recoverability (which governs the length of down time following each failure).			

8.6.3 Fault tolerance measures

Fault tolerance measures are used to assess the degree to which a system, product or component operates as intended despite the presence of hardware or software faults.

NOTE An internal or external fault tolerance measure can be related to the system/software products' capability of maintaining a specified performance level in cases of operation faults or infringement of its specified interface.

Table 17 — Fault tolerance measures

ID	Name	Description	Measurement function
RFt-1-G	**Failure avoidance**	What proportion of fault patterns has been brought under control to avoid critical and serious failures?	X = A/B A = Number of avoided critical and serious failure occurrences (based on test cases) B = Number of executed test cases of fault pattern (almost causing failure) during testing
RFt-2-S	**Redundancy of components**	What proportion of system components is installed redundantly to avoid system failure?	X = A/B A = Number of system components redundantly installed B = Number of system components
NOTE For example, in many safety-critical systems, some parts of the control system could be duplicated with the intention of increasing reliability of the system.			
RFt-3-S	**Mean fault notification time**	How quickly does the system report the occurrence of faults?	$X = \sum_{i=1 \text{ to } n} (A_i - B_i) / n$ A_i = Time at which the fault i is reported by the system B_i = Time at which fault i is detected n = Number of faults detected
NOTE Result value varies from 0 to infinite. Usually, the closer to 0 is the better.			

8.6.4 Recoverability measures

Recoverability measures are used to assess the degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.

Table 18 — Recoverability measures

ID	Name	Description	Measurement function
RRe-1-G	**Mean recovery time**	How long does it take for the software/system to recover from failure?	$X = \sum_{i=1 \text{ to } n} A_i / n$ A_i = Total time to recover the downed software /system and re-initiate operation for each failure i n = Number of failures
NOTE 1 Result value varies from 0 to infinite. Usually, the smaller is better. NOTE 2 When this quality measure is compared to a target threshold for mean recovery time, that is specified in agreed requirements by acquirer and supplier, the measure is able to be used to examine conformance.			
RRe-2-S	**Backup data completeness**	What proportion of data items is backed up regularly?	X = A/B A = Number of data items actually backed up regularly B = Number of data items requiring backup for error recovery

8.7 Security measures

Security measures are used to assess the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.

NOTE 1 Penetration tests can be performed to simulate an attack because such a security attack does not normally occur in the usual testing.

NOTE 2 Security protection requirements vary widely from the case of a stand-alone system to the case of a system connected to the Internet. The determination of the required security functions and the assurance of their effectiveness have been addressed extensively in related International Standards. The user of this International Standard has to determine what kind of security functions need to be used in each case depending on the level of risk.

8.7.1 Confidentiality measures

Confidentiality measures are used to assess the degree to which a product or system ensures that data are accessible only to those authorized to have access.

Table 19 — Confidentiality measures

ID	Name	Description	Measurement function
SCo-1-G	**Access controllability**	What proportion of confidential data items are protected from unauthorized accesses?	X = 1 – A/B A = Number of confidential data items that can be accessed without authorization B = Number of data items that require access control
SCo-2-G	**Data encryption correctness**	How correctly is the encryption/decryption of data items implemented as stated in the requirement specification?	X = A/B A = Number of data items encrypted/decrypted correctly B = Number of data items that require encryption/decryption
NOTE For the details of related data quality, refer to Cnf-I-1 in ISO/IEC 25024.			
SCo-3-S	**Strength of cryptographic algorithms**	What proportion of cryptographic algorithms has been well-vetted?	X = 1 – A/B A = Number of cryptographic algorithms broken or unacceptably risky in use B = Number of cryptographic algorithms used
NOTE 1 It is important to select a well-vetted algorithm that is currently considered to be strong by experts in the field and to select well-tested implementations. As with some cryptographic mechanisms, the source code has to be available for analysis. For example, US government systems require FIPS 140-2 certification. NOTE 2 There are other ways of measuring the strength of cryptographic algorithms, for example, using ethical hacking.			

8.7.2 Integrity measures

Integrity measures are used to assess the degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.

Table 20 — Integrity measures

ID	Name	Description	Measurement function
SIn-1-G	**Data integrity**	To what extent is the data corruption or modification by unauthorized access prevented?	X = 1 – A/B A = Number of data items which are actually corrupted by unauthorized access B = Number of data items for which data corruption or modification have to be prevented
SIn-2-G	**Internal data corruption prevention**	To what extent are the available prevention methods for data corruption implemented?	X = A/B A = Number of data corruption prevention methods actually implemented B = Number of data corruption prevention methods available and recommended
NOTE Examples of internal methods for data corruption prevention are back up data frequently, compare data to reference data periodically, store data in multiple mirror sites.			
SIn-3-S	**Buffer overflow prevention**	What portion of memory accesses with user input in software modules has been done bounds checking for preventing buffer overflow?	X = A/B A = Number of memory accesses with user input that are bounds checked B = Number of memory accesses with user input in software modules
NOTE A buffer overflow occurs when data written to a buffer corrupts data values in memory addresses adjacent to the destination buffer due to insufficient bounds checking. This can occur when copying data from one buffer to another without first checking that the data fits within the destination buffer.			

8.7.3 Non-repudiation measures

Non-repudiation measures are used to assess the degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later.

Table 21 — Non-repudiation measures

ID	Name	Description	Measurement function
SNo-1-G	**Digital signature usage**	What proportion of events requiring non-repudiation is processed using digital signature?	X = A/B A = Number of events that ensure non-repudiation using digital signature B = Number of events requiring non-repudiation using digital signature
NOTE Certificates and security algorithms are also helpful to improve non-repudiation.			

8.7.4 Accountability measures

Accountability measures are used to assess the degree to which the actions of an entity can be traced uniquely to the entity.

Table 22 — Accountability measures

ID	Name	Description	Measurement function
SAc-1-G	**User audit trail completeness**	How complete is the audit trail concerning the user access to the system or data?	X = A/B A = Number of accesses recorded in all logs B = Number of accesses to system or data actually tested
SAc-2-S	**System log retention**	For what percent of the required retention period is the system log retained in stable storage?	X = A/B A = Duration for which the system log is actually retained in stable storage B = Retention period specified for keeping the system log in stable storage
NOTE 1 A stable storage is a classification of computer data storage technology that guarantees atomicity for any given write operation and allows software to be written that is robust against some hardware and power failures. Most often, stable storage functionality is achieved by mirroring data on separate disks via RAID technology. NOTE 2 Result value varies from 0 to infinite. Usually, larger than 1 is better.			

8.7.5 Authenticity measures

Authenticity measures are used to assess the degree to which the identity of a subject or resource can be proved to be the one claimed.

Table 23 — Authenticity measures

ID	Name	Description	Measurement function
SAu-1-G	**Authentication mechanism sufficiency**	How well does the system authenticate the identity of a subject?	X = A/B A = Number of authentication mechanisms provided (e.g., User ID/password or IC card) B = Number of authentication mechanisms specified
NOTE What is relevant for security is the strength of the authentication model and the ability to have multi-level multi-factor authentication and threat detection. Number of factors and degree of authenticity of provided protocol can also be used as authenticity measure.			
SAu-2-S	**Authentication rules conformity**	What proportion of the required authentication rules is established?	X = A/B A = Number of authentication rules implemented B = Number of authentication rules specified

8.8 Maintainability measures

Maintainability measures are used to assess the degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers.

8.8.1 Modularity measures

Modularity measures are used to assess the degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.

Table 24 — Modularity measures

ID	Name	Description	Measurement function
MMo-1-G	**Coupling of components**	How strongly are the components independent and how many components are free from impacts from changes to other components in a system or computer program?	X = A/B A = Number of components which are implemented with no impact on others B = Number of specified components which are required to be independent
NOTE Such a threshold is helpful to determine whether the degree of impact from changes of other components is minimal or not, for example, the frequency of changes of the component caused by changes of other components or the number of externally shared data bases that the component directly accesses.			
MMo-2-S	**Cyclomatic complexity adequacy**	How many software modules have acceptable cyclomatic complexity?	X = 1– A/B A = Number of software modules which have a cyclomatic complexity score that exceeds the specified threshold B = Number of software modules implemented
NOTE Such a threshold is used to determine whether a value of cyclomatic complexity is acceptable or not for each module. This is defined by each project or organization and is possibly a different value for a programming language, a type of module or function.			

8.8.2 Reusability measures

Reusability measures are used to assess the degree to which an asset can be used in more than one system or in building other assets.