Software Improvement Group

# SIG EVALUATION CRITERIA RELIABILITY

## Guidance for producers

SOFTWAREIMPROVEMENTGROUP.COM

**Colophon**

December 21, 2020

Software Improvement Group
+31 20 314 0950
info@softwareimprovementgroup.com

# TABLE OF CONTENTS

# 1. INTRODUCTION

This document describes the SIG evaluation criteria for reliability of software intensive systems. These criteria are defined for the standardized evaluation of the reliability of product, process and operational support of a software system. The purpose of such evaluation is to provide an instrument to architects and developers for guiding improvement of the products they create and enhance, and to acquirers for comparing, selecting, and accepting pre-developed software.

This guidance document outlines the measurement method that SIG applies for the evaluation. This document is not intended as a guide for developing reliable software, still it includes tangible criteria that can be accounted for during the definition of the non-functional aspects of the system.

## 2. ISO 25010 RELIABILITY

The ISO 25010 standard defines reliability as one of the attributes of software product quality (see Chapter 4 for the explanation of reliability according to ISO 25010). Consequently, the ISO 25010 standard identifies four reliability characteristics: **Availability**, **Maturity**, **Fault Tolerance** and **Recoverability**. Availability is mostly defined within a service level agreement (SLA) and is determined as total uptime per given time period. Maturity, Fault tolerance and Recoverability consist of the characteristics proper to the system and supported by its surrounding organization and processes.

The SIG evaluation for reliability implements the ISO 25010 definition through criteria that can be applied to both newly developed systems and systems that are already operational. The SIG evaluation criteria focus primarily on the software system itself, and determine the degree to which the system supports reliable operation. In this context, reliability best practices are expressed for each of the reliability characteristics:

- The system is **mature**: it is thoroughly tested and has a low manual maintenance effort, minimizing the number of potential errors in production;
- The system is **fault-tolerant**: it is fitted with mechanisms to ensure a certain level of tolerance to faults, making sure that a fault does not lead to a system failure;
- The system is **recoverable**: should it fail despite all efforts, it has mechanisms to either recover fully automatically or support human intervention for fast recovery.

These three characteristics maturity, fault tolerance and recoverability are the primary system characteristics in the ISO 25010 standard to influence the externally perceived availability. Figure 1 visualizes these characteristics as encapsulating layers that act to prevent and/or overcome errors, faults and failures in the system.

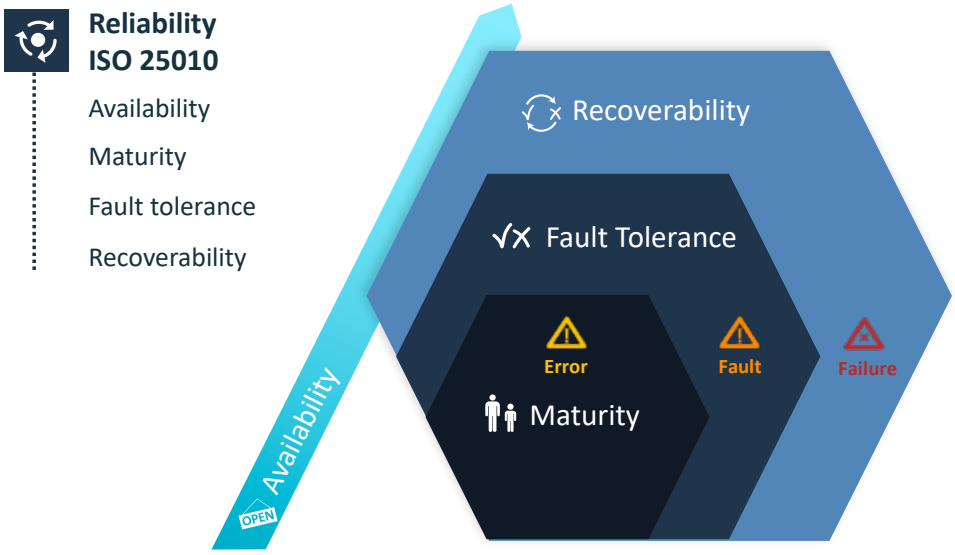


*Figure 1: ISO 25010 system characteristics for reliability.*

Sufficient reliability measures on one of the characteristics can attain high availability. The rationale for this is that a mature system will produce few errors. Errors are deviations from the normal system operations, but not yet visible as a system failure. With few errors the possibility of failures is low, and therefore the system is reliable, independently of its fault tolerance and recoverability. Similar arguments hold for the other two aspects: high tolerance to faults prevents that faults lead to a failure, and an immature system that exhibits errors and has low fault tolerance is still reliable if it can be recovered within the constraints defined for availability.

# 3. SIG RELIABILITY MODEL

To determine the maturity, fault tolerance and recoverability of a system, SIG analyses a number of system properties that influence these characteristics. System properties are traits of the system that can be objectively assessed through process artefacts, design or source code. Figure 2 shows the identified system properties. A rectangle (■) indicates that a property influences a reliability characteristic:

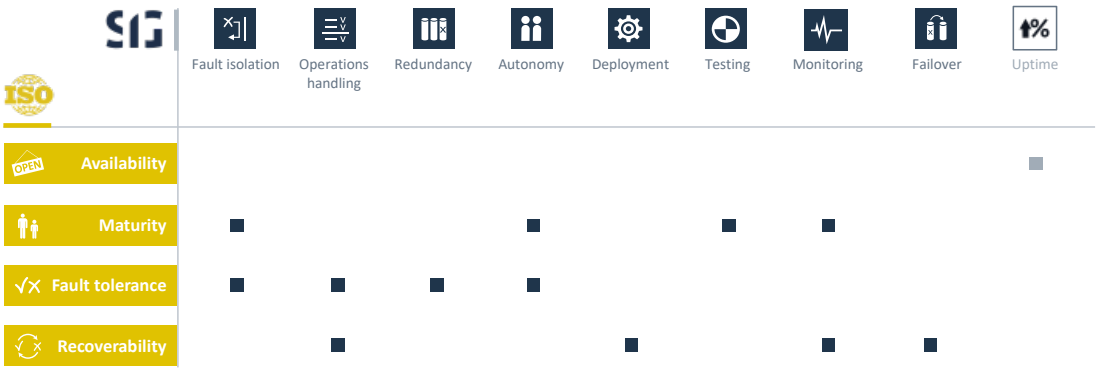


*Figure 2: Mapping of the system properties in the SIG reliability model to the ISO 25010 reliability characteristics*

Each system property is explained as to provide clarifications on the purpose of the SIG Reliability model. Note that these clarifications are meant as guidance for software producers to understand the conditions that satisfy the measurement model.

## 3.1 SYSTEM PROPERTIES

### 3.1.1 Fault isolation

Fault isolation is the degree to which faults in a component are contained. Proper fault isolation **prevents faults from propagating** through the system. A fault is the identified or hypothesized cause of the software failure, often referred to as bug. Ideally, a fault in a (critical) component is contained immediately, such that it does not affect other parts of the system and does not lead to failures and/or total system outage. If a fault occurs in component A, then the components that depend on A should be able to notice it and gracefully inform upstream that a service is not available, instead of 1) waiting and retrying indefinitely or, worse, 2) stop functioning correctly themselves.

When assessing fault isolation, a distinction is made between critical and non-critical components. Critical components are those that are involved in serving the primary functions of the system. These critical components should not be directly affected when other components exhibit problems. This implicitly means that the system architecture is such that the dependencies between components do not result in propagation of errors from one component to another. This implies that error management of the system is mature in terms of error detection and prevention mechanisms, hierarchy in handling errors/exceptions and completeness in the provision of information when logging errors, as to aid in incident management. Detection and prevention mechanisms vary from one system to another but often involve timeout and retries, health checks, containment and pushback mechanisms.

### 3.1.2 Operations Handling

Operations in the system refer to the execution of the main functions and transactions provided by the system. This property assesses the degree to which execution of operations is consistent. The focus in this system property lies in the way operations are handled. Consistent handling of system operations **ensures correct and predictable behaviour of the system**. Inconsistent handling of operations introduces unexpected outcomes that often lead to

faulty behaviour and failures. This translates to whether operations are complex and lengthy or whether these are kept simple according to robustness principles such as economy of mechanisms, atomicity and idem-potency. Next, operations are executed on data and reliability requires validation of data on boundary values prior to execution, i.e. there is no need to run (lengthy) operations with faulty input data to discover in a later stage that this input causes unreliable output, better to ensure that input is valid prior to processing it. Another aspect in handling operations is whether execution is smooth or (unnecessary) interrupted. Interruption in processing, such as using blocking calls that interfere with the normal flow of execution, must be avoided and/or replaced by better mechanisms that guarantee eventual consistency and correct sequence of execution. Additionally, it should be possible to revert an erroneous execution and reinstate the previous correct state and data as a way to recover from an error. This requires that the implemented operation mechanisms prevent inconsistent states and/or data in the event of errors.

### 3.1.3 Redundancy

Redundancy determines to which extent a system is vulnerable to single points of failure. It is obtained by the **duplication** of components and/or services of the system in the same location, **as a means to increase its fault-tolerance**. There are several levels of redundancy, ranging from redundant systems that can be deployed in isolated installations, redundant components that can be deployed as multiple instances to work together and share the load, to no redundancy at all.

### 3.1.4 Autonomy

System autonomy measures the degree to which a system is able to **provide its expected services without reliance on human intervention**. Generally, systems with a high degree of autonomy are cost efficient, less prone to human errors and have faster recovery times.
System autonomy is inversely dependent on the effort needed and exercised by operators directly on the system to ensure continued operation. This consists of the number of operating hours required for regular system maintenance such as clearing logs and archiving data and for addressing shortcomings in behaviour and results. Autonomy can be quantified in operator hours per month. For instance, a system that needs 150 operator hours is less autonomous than a system that needs 8 operator hours per month.

### 3.1.5 Deployment

Deployment indicates how quickly a system can be (re)deployed. The ease and speed with which the deployment of the system to the production environment can be achieved is an indication of the **ability of the system to recover from failures**. The objective is to reduce the time required for deployment of releases and patches by automating the deployment process, minimizing human interactions and enabling faster recovery.
The rationale behind assessing deployment as a contributor to reliability is that reliance on manual steps in system deployment will slow down its recovery in the event of a failure. The faster a system can be deployed, the faster new versions can be put in production, enabling a fast recovery from errors and failures. Furthermore, manual work (in the deployment as well as in other areas) increases the probability of misconfigurations and errors.

### 3.1.6 Testing

Testing is an indication of the maturity of the system and reflects the extent to which the system is able to withhold expected peak loads in production. It is an indication of how thoroughly the system can be stressed while still meeting its requirements. The objective is to uncover most, if not all, detectable faults.
System testing depends on the representativeness of the conducted system tests in assessing the correctness of the behaviour of the system in combination with the level of automation and the coverage of the paths of system execution. It is determined by assessing code coverage through unit testing and integration testing, and by examining the strategies applied for load testing.

### 3.1.7 Monitoring

Monitoring measures the degree to which **system operations, events and resources** are monitored. System monitoring in the context of reliability entails the presence of mechanisms in the system and its environment for timely detection of issues and bottlenecks and for accurate collection and analysis of evidence that aids in incident management and root-cause analysis. This involves but is not limited to the collection of system logs, events, alarms, and the aggregation and correlation of evidence in order to provide insights in the reliability of executions, resource thresholds, faults and failures. Monitoring influences both maturity and recoverability, i.e. a mature system is equipped with mature monitoring mechanisms and monitoring mechanisms support recovery of the system in the event of a failure.

### 3.1.8 Failover

Failover measures the degree to which the system can recover from an outage using a standby deployment. It reflects the ability to switch to a standby/backup system in a secondary location in case of an outage, in order to achieve fast recovery and ensure business continuity.
Success and execution time of failovers is strongly dependent on the level of automation and the assurance with which the failover mode can be successfully achieved. This latter requires frequent testing of the system failover. For instance, recovery of a system that has manual failover requiring 15 operators tested once every two years is less likely to be successful and will take longer than a system which has an active-active configuration that continuously synchronizes.

To determine the contribution of failover in the reliability of the system through recoverability, a number of aspects is considered. This pertains to the existence of a failover deployment within the constraints defined for disaster recovery and business continuity, the synchronization of releases and data between primary and failover site, the frequency and level of automation of failover testing, and the SLA uptime directives (e.g. in cloud).

## 3.2 SYSTEM RATING

In the measurement model, each of the properties is rated based on the SIG's system findings and risks with respect to the degree to which reliability best practices (controls) are implemented. This is expressed in a rating from 1 to 5 stars to reflect the quality of the system. The evaluation is based on methodical assessment by SIG technical experts. Inputs for the evaluation are: source code, the system's architecture and design, and details on deployment and operation.

Generally, the star rating is interpreted as follows:

- > ★★★★★ – reliability controls are well implemented
- > ★★★★☆ – reliability controls are mostly implemented
- > ★★★☆☆ – reliability controls are partially implemented
- > ★★☆☆☆ – reliability controls are minimal
- > ★☆☆☆☆ – reliability controls are missing

## 4. GLOSSARY OF TERMS

The following terms from the ISO 25010 standard are used in this document.

***Reliability***
The degree to which a system, product or component performs specified functions under specified conditions for a specified period of time.

***Availability***
The degree to which a system, product or component is operational and accessible when required for use.

***Maturity***
The degree to which a system, product or component meets needs for reliability under normal operation.

***Fault Tolerance***
The degree to which a system, product or component operates as intended despite the presence of hardware or software faults.

***Recoverability***
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.

Software Improvement Group

Fred. Roeskestraat 115
1076 EE Amsterdam
The Netherlands

www.softwareimprovementgroup.com
marketing@softwareimprovementgroup.com

Getting software right for a healthier digital world