SIG Evaluation Criteria Reliability: *Guidance for producers*
Colophon

Rob van der Leek
+31 20 314 0950
r.vanderleek@sig.eu

Cathal Boogerd
+31 20 314 0950
c.boogerd@sig.eu

Version 1.0 – May 7, 2014
Table of contents

1 Introduction 4

2 Guidance for producers 5
   2.1 Fault Isolation .................................................. 6
   2.2 Transaction Handling ............................................. 6
   2.3 Redundancy .......................................................... 6
   2.4 Deployment Automation .......................................... 7
   2.5 System Autonomy .................................................. 7
   2.6 Reliability Testing ................................................ 7
   2.7 Failover .............................................................. 7

3 Glossary of terms 8
1 Introduction

This document describes the SIG evaluation criteria for reliability of software systems. These criteria are intended 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 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 provides explanation to software producers about the measurement method of SIG applied for evaluation. This document is not intended to be a guide to developing reliable software.
2 Guidance for producers

SIG’s evaluation criteria for reliability operationalize the ISO 25010 definition for reliability (see Chapter 4 for an explanation of reliability terms in ISO 25010). The criteria can be applied to both newly developed systems and systems that are already operational. SIG defines five increasing levels for software reliability, represented by one to five stars (more stars is better).

Following the ISO 25010 standard, the evaluation criteria define reliability as availability of a system. Availability is often measured as uptime or compliance with service level agreements. Such measurements comprise both the system and the surrounding organization. The SIG evaluation criteria focus only on the software system itself, and determine the degree to which that system supports reliable operation. To achieve this, a system should implement the necessary best practices:

- 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**: fitted with mechanisms to ensure a certain level of tolerance of errors, making sure that not every error results in 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. Error! Reference source not found. visualizes these characteristics as encapsulating layers that prevent errors from becoming noticeable to users of the systems.

Figure 1: ISO 25010 system characteristics for reliability.

Sufficient reliability measures on just one of these aspects can guarantee a high availability. The rationale for this is as follows: a mature system will produce few errors. Errors are deviations from the normal internal 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. A similar argument holds for the other two aspects: errors that are tolerated do not need to be recovered, and errors produced by an immature system without fault tolerance can still be recovered without impacting reliability.

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 a system or its processes that can objectively assessed by analysing process artefacts, design or source code. Figure 2 shows how the characteristics are operationalized by system properties. In
this table, a cross (X) indicates that a property has a major influence on a reliability characteristic.

<table>
<thead>
<tr>
<th>Maturity</th>
<th>Fault Isolation</th>
<th>Transaction Handling</th>
<th>Redundancy</th>
<th>Deployment Automation</th>
<th>System Autonomy</th>
<th>Reliability Testing</th>
<th>Failover</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault Tolerance</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Recoverability</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
</tbody>
</table>

Figure 2: Matrix that operationalizes the ISO 25010 definition for reliability.

For each system property, the threshold measurement values are provided that are required for attaining the level of 4 stars. Note that these clarifications are only meant as guidance for software producers, providing sufficient conditions for satisfying the measurement model that is used during evaluation by SIG.

### 2.1 Fault Isolation

Fault isolation is the extent to which a component-based system is able to prevent faults propagating from one to other components. In this context, a fault is something that a component cannot recover from immediately or at all, i.e. the component where the fault occurs is not able to fulfill its role in the system temporarily or permanently. 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 main functionalities offered by the system (i.e. the primary transactions). To attain the level of ★★★★★ for this property, all dependencies to critical components must be non-propagating and only direct (non-transitive) propagation to non-critical components is allowed.

### 2.2 Transaction Handling

Transaction handling is assessed by determining to which extent implemented user operations that fall under the primary transactions (see 2.1), are encapsulated in transactions. A transaction mechanism prevents inconsistent states and/or data in the presence of errors.

To achieve a level of ★★★★★, all critical components should be able to rollback state and data in case a transaction cannot be completed successfully.

### 2.3 Redundancy

Redundancy determines to which extent a system is vulnerable to a single point of failure. There are several levels of redundancy, ranging from redundant systems that can be deployed in completely isolated installations, redundant components that can be deployed multiple times and work together for the same system, to no redundancy at all.
To be eligible for a level of ★★★★☆ for this property, it should be possible to redundantly deploy all components (critical and non-critical) in the system.

2.4 Deployment Automation
Deployment automation indicates how quickly a system can be (re)deployed. The principle behind this metric is that manual work involved in the deployment of a system will slow it down. The faster a system can be (re)deployed, the faster new versions can be put in production, enabling a faster recovery from errors and failures. Furthermore, manual work (in the deployment as well as in other areas) increases the probability of misconfigurations and errors.

For a level of ★★★★☆, the deployment time for a system should be less than 60 minutes.

2.5 System Autonomy
System autonomy measures the dependency of a system to human intervention to stay operational. System autonomy is inversely dependent on the number of operator hours needed and exercised directly on the system. This is measured in operator hours per month. For instance, a system that needs 150 operator hours is less autonomous than a system that needs 10 operator hours per month.

To achieve ★★★★☆ on this property, the number of operator hours per month should be less than 4.

2.6 Reliability Testing
Reliability testing assesses to what extent the test loads resemble production loads and is dependent on the representativeness and the coverage of critical components. Reliability testing coverage also includes the extent to which a system was designed for failure.

To attain the level of ★★★★☆, all components should be tested, and load should be representative for at least the critical components.

2.7 Failover
Success and execution time of failovers is strongly dependent on the level of automation and the frequency of testing if the failover mode can be successfully achieved. 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 level of automation of failover, a surrogate measure is used: the frequency and level of automation of failover testing. This is based on the observation that to achieve a high automation on failover testing, a mature, reliable and automated failover mechanism is required.

To achieve ★★★★☆, failover of the system should be frequently automatically tested (every 1-4 weeks).
3 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.