OpenAIS

Open Architectures for Intelligent Solid State Lighting Systems

Risk analysis report and required test setups
(Deliverable: D4.3)

Version 2.0, 18 July 2016
# Table of Contents

0 INTRODUCTION ........................................................................................................... 4  
  0.1 Document context .......................................................................................... 4  
  0.2 Scope .................................................................................................................. 4  
  0.3 Objective .......................................................................................................... 4  

1 RISK ANALYSIS ........................................................................................................ 6  
  1.1 Risk Management Process ............................................................................... 6  
  1.2 Risk Assessment - Preliminaries ...................................................................... 6  
     1.2.1 DEFINITION .......................................................................................... 6  
     1.2.2 PURPOSE .............................................................................................. 7  
     1.2.3 PROCESS ............................................................................................... 7  
     1.2.3.1 Risk identification .............................................................................. 7  
     1.2.3.2 Risk analysis ...................................................................................... 8  
     1.2.3.3 Risk evaluation ................................................................................... 8  
  1.2.4 SELECTION OF RISK ASSESSMENT TECHNIQUES ................................... 8  
  1.3 Risk Assessment .............................................................................................. 10  
     1.3.1 OPENAIS SYSTEM ............................................................................... 10  
     1.3.2 OUR APPROACH ................................................................................. 11  
     1.3.2.1 FMEA ............................................................................................... 11  
  1.4 Risk Analysis Report ....................................................................................... 12  
     1.4.1 RISK REGISTER .................................................................................... 12  
  1.5 Risk Mitigation ................................................................................................. 14  
     1.5.1 MITIGATION PLAN ................................................................................ 14  
     1.5.1.1 First step - Mitigation action workshop .............................................. 14  
     1.5.1.2 Next steps .......................................................................................... 15  

2 RISK-BASED TESTING ............................................................................................ 17  
  2.1 The test plan for the identified high risks ....................................................... 17  

3 REQUIRED TEST SETUPS ..................................................................................... 19  
  3.1 The M16 test setup ........................................................................................... 19  
  3.2 The final test setup .......................................................................................... 21  
  3.3 System view of test setup ................................................................................ 21  
     3.3.1 TEST MANAGEMENT SYSTEM ............................................................. 22  
  3.4 Test Architecture .............................................................................................. 23  
     3.4.1 TEST EXECUTION AND VERIFICATION .............................................. 23  
     3.4.2 TEST ARCHITECTURE CONTEXT ....................................................... 24  
     3.4.3 OBJECT TYPES ................................................................................... 24  
     3.4.4 ACTUATOR TYPES ............................................................................... 25  
     3.4.5 SENSOR TYPES ................................................................................... 25  
  3.5 Connections to BMS domain .......................................................................... 25  

4 TEST FRAMEWORK ................................................................................................. 27
4.1 Test Automation Framework ................................................ 27
4.2 Virtual devices ....................................................................... 27
4.3 Controlled OpenAIS devices ................................................ 27
  4.3.1 WIRELESS NETWORK TRAFFIC GENERATOR ................. 28
4.4 Network emulation .............................................................. 29
4.5 Remote Testing .................................................................... 29
5 REFERENCES ........................................................................... 31
1. Introduction

1.1 Document context

This document is part of the Work Package 4 “Integration of components” and is titled “Risk analysis report and required test setups”. It has relation to the other deliverables given below:

- WP4: D4.4 Integration Approach/Process[OPENAIS_D4.4] providing information on:
  - The test architecture and system to be used during integration
  - The identified risk to be verified during integration

- WP2: D2.3 Final Architecture [OPENAIS_D2.3] providing information on:
  - Devices and related objects being part of the architecture.
  - The communication infrastructure
  - Commissioning aspects
  - Security aspects

- WP1: D1.4 Solution Specification providing information on:
  - Pilot system requirements
  - Architectural requirements
  - Usability requirements
  - Scenarios, use cases and story boards

1.2 Scope

The scope of the document is related to a pilot system design that is built according the OpenAIS reference architecture. As many system designs can be derived from this reference architecture, the focus for the risk assessment, will be on the intended pilot system. As the pilot customer is not yet known at the time of writing of this document, a reference system is assumed, based on an existing system installation of one of the project members.

1.3 Objective

There are two main chapters in this document, one covering risk assessment and another one covering the test setups. Both related to the OpenAIS system to be built for a pilot

The risk assessment is based on a Failure Mode and Effect Analysis (chapter 2.3.2.1) and related risk analysis, and will be categorized according to impact on users of the system, and failure effect on the system.

The test setup (Chapter 4 Required test setups) is built in two phases, the first one (M16 test setup) is a small test setup to start with, which will gradually be extended with new features and devices, growing towards a bigger final test setup (M33 test setup), as a representative for the actual pilot system. The final test setup will also allow integration with existing Building Management Systems (BMS).

The test setups are based on a test architecture that allows for continuous integration and testing, using test automation frameworks.

Based on the outcome of the risk assessment a test strategy will be defined for each of the identified risks. Chapter 3 Risk-based testing will describe the test strategies to apply.
2. Risk Analysis

*Risk* is a measure of the extent to which an entity is threatened by a potential circumstance or event, and is typically a function of: (i) the adverse impacts that would arise if the circumstance or event occurs; and (ii) the likelihood of occurrence [NIST].

2.1 Risk Management Process

Risk management processes include: (i) framing risk; (ii) assessing risk; (iii) responding to risk; and (iv) monitoring risk [NIST]. Figure 1 illustrates the four steps in the risk management process, including the information and communication flows necessary to make the process work effectively. The first component of risk management addresses how organizations frame risk or establish a risk context — that is, describing the environment in which risk-based decisions are made. It produces a risk management strategy that addresses how organizations intend to assess risk, respond to risk, and monitor risk. The second component of risk management addresses how organizations assess risk within the context of the organizational risk frame. The third component of risk management addresses how organizations respond to risk once that risk is determined based on the results of a risk assessment. The fourth component of risk management addresses how organizations monitor risk over time.

![Figure 1 the big picture: Risk Management Process [NIST]](image)

The focus of this deliverable is on the *risk assessment* component of a risk management process.

2.2 Risk Assessment - Preliminaries

This section briefly describes the background of risk assessment by providing the definition, purpose and process referring to the ISO/IEC 31010 standard [IEC 31010:2009].

2.2.1 Definition

*Risk assessment* is the process of identifying, estimating, and prioritizing risks. Assessing risk requires the careful analysis of threats and failures to determine the extent to which circumstances or events could adversely impact a system and the likelihood that such circumstances or events will occur.
2.2.2 Purpose

The risk assessment provides a structured qualitative assessment of the operational environment of the system. It identifies sensitivities, threats, vulnerabilities and risks. The assessment recommends cost-effective safeguards to mitigate risks, threats and associated exploitable vulnerabilities. This is summarized in IEC 31010 as follows:

*The purpose of risk assessment is to provide evidence-based information and analysis to make informed decisions on how to treat particular risks and how to select between options.* [IEC 31010:2009]

2.2.3 Process

Risk assessment process involves risk identification, risk analysis and risk evaluation.

![Figure 2 Risk assessment in risk management process [IEC 31010:2009]](image)

Error! Reference source not found. shows the steps followed in risk assessments within the context of risk management process.

2.2.3.1 Risk identification

Risk identification is the process of finding, recognizing and recording risks [IEC 31010:2009]. Here we identify what might happen or what situations might exist that affect the achievement of given objectives. The process includes identifying the causes and sources of the risk, events, situations or circumstances which could have a material impact upon objectives and the nature of that impact.

Risk identification methods can include [IEC 31010:2009]:

- evidence-based methods (e.g. check-lists and reviews of historical data)
- systematic team approaches (follows a systematic process by means of a structured set of prompts or questions)
- inductive reasoning techniques such as HAZOP [IEC 61882:2001].
Various supporting techniques can be used to improve accuracy and completeness in risk identification, including brainstorming, and Delphi methodology [Linstone]. Once a risk is identified, any existing controls such as design features, processes and systems are identified.

2.2.3.2 Risk analysis
Risk analysis is about developing an understanding of the risk. Risk analysis consists of determining the consequences and their probabilities for identified risk events, taking into account the presence (or absence) and the effectiveness of any existing controls [IEC 31010:2009]. The consequences and their probabilities are then combined to determine a level of risk. Methods used in analysing risks can be qualitative, semi-quantitative or quantitative. Qualitative assessment uses significance levels such as “high”, “medium” and “low”, semi-quantitative methods use numerical rating scales and quantitative analysis estimates practical values in specific units.

2.2.3.3 Risk evaluation
Risk evaluation involves comparing estimated levels of risk with risk criteria defined when the context was established, in order to determine the significance of the level and type of risk and then make decisions about future actions. Decisions may include: whether a risk needs treatment, priorities for treatment, whether an activity should be undertaken, etc.

2.2.4 Selection of risk assessment techniques
Risk assessment may be undertaken in varying degrees of depth and detail and using one or many methods ranging from simple to complex. Some of the factors influencing the selection of risk assessment techniques are:

- The objectives of the study and the needs of decision-makers
- The type and range of risks
- Degree of expertise, human and other resources and the availability of information and data

Table 1 provides a comparison on the applicability of risk assessment tools.
<table>
<thead>
<tr>
<th>Tools and techniques</th>
<th>Risk assessment process</th>
<th>Risk identification</th>
<th>Risk analysis</th>
<th>Risk level evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Risk identification</td>
<td>Consequence</td>
<td>Probability</td>
<td>Level of risk</td>
</tr>
<tr>
<td>Brainstorming</td>
<td>SA&lt;sup&gt;1&lt;/sup&gt;</td>
<td>NA&lt;sup&gt;2&lt;/sup&gt;</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Delphi</td>
<td>SA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Check-lists</td>
<td>SA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Primary hazard analysis</td>
<td>SA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Hazard and operability studies (HAZOP)</td>
<td>SA</td>
<td>SA</td>
<td>A&lt;sup&gt;3&lt;/sup&gt;</td>
<td>A</td>
</tr>
<tr>
<td>Hazard Analysis and Critical Control Points (HACCP)</td>
<td>SA</td>
<td>SA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Environmental risk assessment</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
</tr>
<tr>
<td>Structure « What if? » (SWIFT)</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
</tr>
<tr>
<td>Scenario analysis</td>
<td>SA</td>
<td>SA</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Business impact analysis</td>
<td>A</td>
<td>SA</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Root cause analysis</td>
<td>NA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
</tr>
<tr>
<td>Failure Mode Effect Analysis (FMEA)</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
</tr>
<tr>
<td>Fault tree analysis</td>
<td>A</td>
<td>NA</td>
<td>SA</td>
<td>A</td>
</tr>
<tr>
<td>Event tree analysis</td>
<td>A</td>
<td>SA</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Cause and consequence analysis</td>
<td>A</td>
<td>SA</td>
<td>SA</td>
<td>A</td>
</tr>
<tr>
<td>Cause-and-effect analysis</td>
<td>SA</td>
<td>SA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Layer protection analysis (LOPA)</td>
<td>A</td>
<td>SA</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Decision tree</td>
<td>NA</td>
<td>SA</td>
<td>SA</td>
<td>A</td>
</tr>
<tr>
<td>Human reliability analysis</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
</tr>
<tr>
<td>Bow tie analysis</td>
<td>NA</td>
<td>A</td>
<td>SA</td>
<td>A</td>
</tr>
<tr>
<td>Reliability centred maintenance</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
</tr>
<tr>
<td>Sneak circuit analysis</td>
<td>A</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Markov analysis</td>
<td>A</td>
<td>SA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Monte Carlo simulation</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Bayesian statistics and Bayes Nets</td>
<td>NA</td>
<td>SA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>FN curves</td>
<td>A</td>
<td>SA</td>
<td>SA</td>
<td>A</td>
</tr>
<tr>
<td>Risk indices</td>
<td>A</td>
<td>SA</td>
<td>SA</td>
<td>A</td>
</tr>
<tr>
<td>Consequence/probability matrix</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
<td>SA</td>
</tr>
<tr>
<td>Cost/benefit analysis</td>
<td>A</td>
<td>SA</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Multi-criteria decision analysis (MCDA)</td>
<td>A</td>
<td>SA</td>
<td>A</td>
<td>SA</td>
</tr>
</tbody>
</table>

<sup>1</sup> Strongly applicable  <sup>2</sup> Not applicable  <sup>3</sup> Applicable
2.3 Risk Assessment

2.3.1 OpenAIS System

As an input for our risk analysis we need a system or a system design. In this early phase of the OpenAIS project, we only have a reference architecture [OPENAIS_D2.3] and lack the needed system. To come up with a system out of the reference architecture, we decided to focus on the pilot system that would be designed for M33. Based on our expectation of the pilot, we made assumptions about the system and proceeded with the design. We took the floor plan and lighting design of one floor of a building on the High Tech Campus for the system under analysis.

Figure 3 shows the realization view of the system expected for M33; this system was used for our analysis.
2.3.2 Our Approach

2.3.2.1 FMEA

We have seen in Section 2.2.4 that there are various risk assessment techniques with varying complexity and extent of resources and data required for analysis. Among them Failure Mode and Effect Analysis (FMEA), a technique which identifies failure modes and their effects, is rated high for risk identification, risk analysis and risk evaluation. Considering the relevance and suitability for our risk analysis, we picked this method.

In the FMEA process, first we look at the hierarchical decomposition of the system into basic elements and start the analysis from the lowest level elements and proceed in a bottom-up fashion. For each system element (subsystem, component) we look at the functions of the element in all of its operational modes, and check if any failure of the element may result in any unacceptable system effect. A failure mode effect at a lower level may then become a cause of a failure mode of an item in the next higher level. Hence we look at its local effect and also identify the end effect on the system. The severity of the failure is also estimated. The next step is to identify the causes of the failure and also to estimate the probability of occurrence of the failure modes. In certain analysis, as a last step the methods to detect the failures are identified and probability of detection is estimated.
The potential risks, \( R \), of the various failure modes is measured in as follows:

\[
R = S \times P,
\]

(1)

where \( P \) denotes the probability of occurrence and \( S \) stands for the severity of the failure mode.

In the analysis where the additional step to distinguish the level of failure detection, the risks of the various failure modes is presented by a Risk Priority Number (RPN)

\[
R = S \times O \times D,
\]

(2)

where \( O \) is the rank of the occurrence of the failure mode, \( S \) is the rank of the severity of the failure mode and \( D \) is the rank of the likelihood the failure will be detected before the system reaches the end-user/customer.

A one-day FMEA analysis workshop has been organized with participants from OpenAIS WP4 team, OpenAIS architects and an external expert. The details of the FMEA analysis and workshops are described in Deliverable D4.2 (due in M12).

### 2.4 Risk Analysis Report

We have seen in Section 2.2.3 that risk assessment process involves risk identification, risk analysis, and risk evaluation. For our risk assessment, FMEA has been chosen (Section 2.3.2). In this section we focus on the reporting part of risk analysis step in the risk assessment process. The results from FMEA workshop [OpenAIS_D4.2] are used for reporting.

#### 2.4.1 Risk Register

A risk register is a tool used in risk analysis phase to report risk events. Risk factors analysed and quantified are reported in the risk register. It provides an indication of the risk levels, their acceptability and corresponding actions needed.

The FMEA report given Deliverable D4.2 [OpenAIS_D4.2] provides the RPN calculated as in equation Error! Reference source not found. The larger the RPN the worse is the risk; but a direct mapping of RPN to risk levels is often not considered in the literature. Finding out the acceptable risk is generally a complicated and multifaceted issue. One of the principles used to determine the acceptable risk is the ALARP principle (“As low as reasonably practicable”) [Rausand]. This is shown in risk matrix given in Error! Not a valid bookmark self-reference., where the occurrence and severity are used to identify the coarse risk levels as indicated by the colour codes given in Table 3. The occurrence and severity rankings are based on IEC 60812:2006 [IEC 60812:2006] (as used in FMEA analysis given in Deliverable D4.2). Table 3 also shows the acceptability and action needed for each risk level [Rausand].

<table>
<thead>
<tr>
<th>Occurrence/Severity</th>
<th>Remote (1)</th>
<th>Low (2-3)</th>
<th>Moderate (4-6)</th>
<th>High (7-8)</th>
<th>Very high (9-10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hazardous (9-10)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Very) High</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Risk Matrix
Table 3: Risk levels and actions

<table>
<thead>
<tr>
<th>Colour Code</th>
<th>Risk level</th>
<th>Acceptability</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Low</td>
<td>Acceptable</td>
<td>Only ALARP actions considered</td>
</tr>
<tr>
<td>Moderate</td>
<td>Moderate</td>
<td>Acceptable</td>
<td>Use ALARP principle and consider further</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>investigations</td>
</tr>
<tr>
<td>High</td>
<td>High</td>
<td>Not acceptable</td>
<td>Risk reducing measures required</td>
</tr>
</tbody>
</table>

The FMEA report given Deliverable D4.2 [OpenAIS_D4.2] provides the RPN calculated as in equation Error! Reference source not found.. The larger the RPN the worse is the risk; but a direct mapping of RPN to risk levels is often not considered in the literature. Finding out the acceptable risk is generally a complicated and multifaceted issue. One of the principles used to determine the acceptable risk is the ALARP principle (“As low as reasonably practicable”) [Rausand]. This is shown in risk matrix given in Error! Not a valid bookmark self-reference., where the occurrence and severity are used to identify the coarse risk levels as indicated by the colour codes given in Table 3. The occurrence and severity rankings are based on IEC 60812:2006 [IEC 60812:2006] (as used in FMEA analysis given in Deliverable D4.2). Table 3 also shows the acceptability and action needed for each risk level [Rausand].

and Table 3 show that one to one mapping of R or RPN to risk-level is difficult. E.g. if the severity-level is Hazardous and occurrence is low, R will be between 18 and 30 and RPN will be between 90 and 150 assuming a moderate (5) detectability. The risk level in this case is high and risk reducing measures are required. Now consider, a low severity-level and high occurrence. R will be between 28 and 40 and RPN will be between 90 and 150 (assuming a moderate (5) detectability).

To filter in the entries for unacceptable risks in a large FMEA table, we could consider all entries with R $\geq$ 18 (18 is lowest possible number for unacceptable risk). However, careful attention has been given to include entries with detectability above moderate level ($\geq$ 5). The corresponding RPN will be $\geq$ 90. Based on these filters, we checked each entry in the FMEA table and categorized them according to the risk level. The risk register shown in Table 4 highlights the findings with reported risk levels of high and above where risk reducing measures required. It also shows moderate ones (orange colour) with detectability issues (i.e. RPN $\geq$ 90).

Table 4: Risk Register from FMEA analysis

<table>
<thead>
<tr>
<th>Risk No./Type</th>
<th>Risk Event</th>
<th>Concerns</th>
<th>Risk Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cannot configure the device (e.g. floor controller) during installation and</td>
<td>IT issues arising during commissioning</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>commissioning due to connectivity issues (e.g. discovery not working)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Wireless devices’ (luminaires, sensors etc.) communication interrupted by short interferences

2

Wrong placement of devices (e.g. sensors) during installations

3

Wrong linking of devices (e.g. sensors) during commissioning

4

Light not responding to UI Apps after getting authorized

5

PoE switch component damage/failure/power issues

6

Communication with cloud services are broken and gaps in data received

7

Communication issues due to security errors (no keys, key expiry) affecting lighting control

8

Wrong configuration (e.g. border router, PoE switch) during installation

10

Border router not accessible due to broadcast storm

11

<table>
<thead>
<tr>
<th>Concern</th>
<th>Risk Mitigation</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 IT issues arising during commissioning</td>
<td>Note: IT network and IT support may become available only after lighting installations in the traditional workflow. Hence, if no contractual agreement can be made to ensure</td>
<td></td>
</tr>
</tbody>
</table>

2.5 Risk Mitigation

An important outcome of a risk analysis is the identification of high risk areas; these have a high potential to affect the project. Table 4 lists the most important risks for the OpenAIS project.

2.5.1 Mitigation Plan

As mentioned in Section Error! Reference source not found., we have done an early stage analysis based on the expectation of pilot system that would be designed for M33. The results and findings of the FMEA, HAZOP and risk analysis have been fed back to the OpenAIS architecture (WP2) team and the system design team (joint effort by WP2 and WP3), so that they can take appropriate measures at the architectural and design levels to reduce the risk.

2.5.1.1 First step - Mitigation action workshop

To address the main risks identified a second workshop has been conducted with OpenAIS architects (WP2 team). In addition to the WP2 team and WP4 team members facilitating the workshop, an external expert and WP1 team members (to represent the customers) were also invited. We began our analysis by considering one specific instantiation of the OpenAIS architecture that will be used in 2020. A summary of the recommendations identified are listed below. For a more detailed view, see deliverable D2.3 [OpenAIS_D2.3].

Mitigation recommendations

Concern 1: IT issues arising during commissioning

Note: IT network and IT support may become available only after lighting installations in the traditional workflow. Hence, if no contractual agreement can be made to ensure
IT availability at the commissioning stage, a preliminary network needs to be setup in the organization. This network can be made independent from standard IT network by using temporary devices (e.g. preliminary switches/routers, tablet with SIM that can directly connect to cloud, etc.)

(Architectural) Solutions:
- OpenAIS provides system diagnostics by giving system alarm messages such as error messages and status messages.
- OpenAIS standard for logging eases understanding and handling of logged messages and thereby eases troubleshooting.

Concern 2: Communication issues (RF interference, bandwidth, throughput loss, etc.)
Note: There is a contractual issue - use either a separate lighting network or ensure minimum BW available for lighting purpose by contract.

(Architectural) Solutions:
- Use IPv6 priority flags to receive higher priority for lighting packets (but it may not be available always).
- RF interference can be addressed by enabling channel agility in PHY or have a tool (in the devices) that can detect issues and change channels.
- Limit the hop counts to low numbers in wireless networks. This introduces the need for additional border routers. So it is a tradeoff between the infrastructural cost and debugging cost.
- Issues due to IT misconfiguration - triage is possible by using diagnostic tool.

Concern 3: Component/parts failure

(Architectural) Solutions:
- Limit spreading of failures and use fallback provisions
- Use redundant components and connections to circumvent failing parts

Concern 3: Wrong configuration due to human as well as design errors

(Architectural) Solutions:
- Architectural support for debugging, logging and error handling helps in diagnostics and triage
- Usage of diagnostic tools are supported
- Simulation helps to identify problems in design stage and also to find optimal configurations.

2.5.1.2 Next steps
We expect regular discussions with system design (joint effort by WP2 and WP3) and implementation teams (WP3) to discuss about the mitigation strategies and actions at design and implementation levels.

As part of the WP4 risk mitigation actions, WP4 team plan to adopt risk-based testing which will be explained in Section 3.1 Additionally, TNO-ESI is developing virtual prototypes for lighting system following a model-based lighting system development. This helps to validate system configuration and behaviour in the early design phases and avoids errors and issues in the actual system development. It eases design
changes for risk reduction purpose and ensures the correct functionality and desired performance of the system.
3. Risk-based testing

The risk levels mentioned in Section 2.4 will imply different test strategies. The risk register will be used to identify related components and their functionality affecting the identified risk event. Specific test strategies will be applied for this functionality. An inventory will be made during each feature pack delivery, to identify the components and functions, related to the risk events. Feature packs are described in OpenAIS deliverable D4.4 “Integration Approach/Process report”

In general the following test strategy [CODE][TST] can be applied to the components:

- Components related to very high risk events can use following test strategies:
  - Boundary Value Analysis
  - Branch coverage 90%
  - Use-cases: exceptional flow

- Components related to high risk events can use following test strategies:
  - Equivalence Partitioning
  - Statement coverage 90%
  - Use-cases: basic/alternative flow

- Components related to moderate risk events can use following test strategies:
  - Free format
  - Equivalence Partitioning
  - Statement coverage 70%
  - Use-cases: basic flow

3.1 The test plan for the identified high risks

Special test cases will be designed to test the risks identified in the risk register of Chapter 2.4.1. Table 5 indicates in the <Test Case> row which test case will be applied for each of the risks.

Table 5 Test plan for identified risks

<table>
<thead>
<tr>
<th>Risk No.</th>
<th>Risk Event</th>
<th>Concerns</th>
<th>Test Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cannot configure the device (e.g. floor controller) during installation and commissioning due to connectivity issues (e.g. discovery not working)</td>
<td>IT issues arising during commissioning</td>
<td>Apply test scenario with missing devices, missing or failing network devices, or disturbed connectivity</td>
</tr>
<tr>
<td>2</td>
<td>Wireless devices’ (luminaires, sensors etc.) communication interrupted by short interferences</td>
<td>Crowded band</td>
<td>Use of wireless network traffic pattern generator as hotspots in the network (see section 5.3.1)</td>
</tr>
<tr>
<td>3</td>
<td>Wrong placement of devices (e.g. sensors) during installations</td>
<td>Human errors</td>
<td>Mount a device at incorrect placement, and check whether wrong placement can be detected, using e.g. the BIM location information.</td>
</tr>
<tr>
<td>4</td>
<td>Wrong linking of devices (e.g. sensors) during commissioning</td>
<td>Human errors</td>
<td>Apply test case using a wrong linking of devices and check if commissioning tool can detect a wrong link.</td>
</tr>
<tr>
<td></td>
<td>Light not responding to UI Apps after getting authorized</td>
<td>Light is not reacting</td>
<td>Apply test scenarios to simulate connectivity problems, including the Wi-Fi connection, to check also the feedback provided by the UI App</td>
</tr>
<tr>
<td>---</td>
<td>------------------------------------------------------</td>
<td>----------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>PoE switch component damage/failure/power issues</td>
<td>No communication between connected device, Grouping does not work</td>
<td>Apply test cases, by removing or switching off a PoE switch, and verify the fall-back scenarios/recovery scenarios.</td>
</tr>
<tr>
<td></td>
<td>Communication with cloud services are broken and gaps in data received</td>
<td>Value of data analytics reduces</td>
<td>Apply test scenario with a broken external (cloud) connection, introduce gaps in local storage of data. Check connection recovery and missing data handling.</td>
</tr>
<tr>
<td></td>
<td>Sensor communication issues due to security errors (no keys, key expiry) affecting lighting control</td>
<td>Security vulnerability</td>
<td>Apply test scenario using prepared test devices having these specific security issues</td>
</tr>
<tr>
<td></td>
<td>Wrong configuration (e.g. border router, PoE switch) during installation</td>
<td>Human errors</td>
<td>Apply test cases using a wrong configurations</td>
</tr>
<tr>
<td></td>
<td>Border router not accessible due to broadcast storm</td>
<td>Human configuration/design error or intruder</td>
<td>Use of wireless network traffic pattern generator to induce a broadcast storm in the network (see section 5.3.1) and check recovery.</td>
</tr>
</tbody>
</table>
4. Required test setups

There will be one WP4 managed test setup, which resides at the HTC48 building of PHL partner. The first test setup built for integration will be simple. It is the intention to extend it during every integration increment, resulting in a system that is representative for the actual pilot system. The goal for the first test setup (also called the M16 setup) is to verify that:

- The common development platform is working, with development at different sites and devices from different vendors.
- The common hardware boards are working.
- The basic wired IP communication is working.
- The basic connectivity between OpenAIS objects is working, based on pre-configured relation between the object instances.
- The connection to vendor-specific device drivers is working.

The M16 test setup, will evolve into a bigger and final test setup, integrating and testing according the pilot needs; the final test setup is just a subset of the actual pilot system. At least the following aspects will be tested for the pilot system:

- The use-cases identified for the pilot system,
- The integration with IP infrastructure of the customer,
- The integration with existing Building Automation system, and existing Lighting systems.

4.1 The M16 test setup

This initial version of the test setup is for integration and testing of the basic infrastructural functions of the system (Error! Reference source not found.). The various contributing partners can easily reproduce this test setup to enable concurrent development of OpenAIS modules. All OpenAIS devices are implemented by means of a NXP FRDM-K64F board which interfaces to existing hardware.
The number of devices is scaled down to a minimal set of:

- 4 OpenAIS actuators connected to Philips and Zumtobel light points (luminaires)
- An OpenAIS light control object deployed on a dedicated device.
- An OpenAIS sensor: a push-button switch.

The infrastructural devices are:

- An off-the-shelf Ethernet router to create a small IPV6-based local network
- The Windows PC has a LWM2M server installed on a Virtual Linux environment, which has IPV6 Router Advertisement support. The LWM2M includes also a system dashboard, to monitor and control the object resources in this pre-configured system.

All OpenAIS devices are implemented by means of a NXP FRDM-K64F board (Error! Reference source not found.) which interfaces to existing hardware. This board uses the open source Mbed OS from ARM designed for IP-connectivity, low power and resource constrained devices. It can be extended with a wireless 802.15.4 radio.
4.2 The final test setup

The final test setup is based on the M16 Test setup and extended with OpenAIS devices and infrastructural devices, enabling testing of the agreed solution space requirements for the pilot locations (Figure 6). The number of devices is scaled up to a minimal sub-set of the actual pilot system:

- 8 wired and 8 wireless OpenAIS actuators connected to Philips and Zumtobel light points (luminaires)
- An OpenAIS group control object deployed on a dedicated wired device.
- An OpenAIS Area/Floor wireless floor controller
- An OpenAIS stand-alone wired light switch, representing a wall switch object.
- A Network emulator, emulation network nodes with OpenAIS objects.

The infrastructural devices are:

- An off-the-shelf Ethernet/Wifi router to create a small IPV6-based local network.
- A Low Power Radio Access Point (ARM) for the wireless network.
- The LWM2M server for OpenAIS dashboard, and providing a web-interface to OpenAIS devices.

4.3 System view of test setup

The OpenAIS system under test (SUT) is part of the IT infrastructure of the customer. The whole lighting system is broader than just the installation of the OpenAIS components. As an OpenAIS system needs an IP network as a connectivity backbone,
the required IP network equipment and devices are within the context of the test architecture. A view of such a system is shown in (Figure 7), with:

- The orange coloured blocks are part of the test management system, to allow for automatic configuration deployments and testing.
- The blue coloured blocks are considered to be part of the OpenAIS subsystem.
- The green coloured blocks are IT systems that are required for a proper operation of an OpenAIS-based system.
- The purple coloured blocks are needed for installation or configuration of an OpenAIS lighting system. These blocks are under control of the Test Management system.

4.3.1 Test Management System

The Test Management System is built for WP4 to support continuous integration and testing of a System Under Test (SUT). The SUT is an OpenAIS based system.

- The main tasks of the Test Management System is: Deploy test configurations into the SUT via an attached Lighting Configuration tool (or commissioning tool)
- Distribute localization data into the SUT, to automate the installation process.
- Set the time reference in the SUT, for correct time stamps in data loggings, using a time synchronisation master.
- Distribute new releases of firmware or 3rd party extensions.
- Execute test scripts, collect test results and verify the outcome of the results.
- Program the IP Network Monitor device to analyse the network traffic via connected network monitoring system.

Special attention is needed for configuration management, i.e. keeping track of firmware and hardware releases, versions and patches. In deliverable D4.4
(Integration Approach) version management will be described in more detail. It should be possible to deploy new configurations as well as a roll-back to older configurations. The build management system, which will be based on an open-source tool called Jenkins, will provide build scripts to manage the different configurations.

4.4 Test Architecture

The test architecture has to provide the means to test the intended system on different aspects:

- **Verification**: Checking and testing, whether the system is well engineered, and meeting the identified quality requirements. It does not check whether system is also useful.
- **Validation**: Checking whether the system meets the intended usage requirements. Related to use cases and related user and technical requirements, to also validate if the requirements are stated correctly.

The OpenAIS objects mentioned in this chapter are defined in the OpenAIS document D2.3 “Final architecture of OpenAIS system”.

4.4.1 Test execution and verification

In general a virtual COAP device [COAP] will execute test scripts by sending either a PUT operation (to set the value of a resource) or a POST operation (to execute an event or further processing of a value at a resource) to a selected node or group of nodes. The result of the PUT or POST operation will be verified by reading out the status of the resources from the affected nodes, via a GET operation (see Figure 8 Test execution).

![Figure 8 Test execution](image-url)
4.4.2 Test Architecture Context

Within the context of the test architecture, the following physical devices can be identified (see Figure 9 Test Architecture):

- Wired OpenAIS devices in a wired network (Ethernet), in which a device can deploy a sensor, an actuator, a control, or a datacollect object instance, or combination of them.
- Wireless OpenAIS devices in a wireless network (Thread/6loWPAN), in which a device can deploy a sensor, an actuator, a control, or a datacollect object instance, or combination of them.
- A Border Router, responsible for creating a wireless IP network for the wireless OpenAIS devices. It will route information packets between the wired IP domain and the wireless IP domain.

The following virtual devices can be identified running in a Linux or Windows environment:

- A virtual OpenAIS device, supporting a wired connection
- A virtual OpenAIS device, supporting a wireless connection
- A network emulator

The test automation framework is part of the test architecture, as it is required to initiate and execute test scripts, and check outcome by verifying the state of devices. It runs on a Linux or Windows environment.

4.4.3 Object types

In this section a detailed description of the supported object types, their purpose, and how they will be deployed on the devices. There should be a wired and a wireless version of each device type, and preferably from the different vendors, being Zumtobel and Philips to test interoperability aspects between the devices.

Some basic requirements for each of the devices:

- Each of the object instances should have resources available to read out or report the actual status of the object instance to verify the outcome of a test.
- Each of the object instances should have resources available for binding purposes.
The test architecture should allow OpenAIS devices to be integrated as end-point devices (single IP address), covering either a single instance of an object (a single URL), or multiple instances of an object or instances of different objects (more than one URL involved). The following devices need to be at least supported:

- **Devices with a single instance of an object, being:**
  - Occupancy sensor object instances,
  - Temperature sensor object instances,
  - Actuator object instances,
  - Control object instances,
  - Datacollect object instances.
  - Digital Input object instances (e.g. switches)
- **Devices with multiple instances of an object, being:**
  - Devices with two sensor object instances, e.g. an occupancy sensor and a temperature sensor,
  - Devices with two actuator object instances, e.g. task light and background light,
  - Devices with a stacked control object, e.g. a local ON/OFF control with an override control.
- **Devices with a single instance of different objects, being:**
  - Devices with a control object instance and a sensor object instance,
  - Devices with a control object instance and an actuator object instance,
  - Devices with a control object instance and a datacollect object instance.

More complex device deployments are allowed, but not necessary.

### 4.4.4 Actuator types

The test architecture should support testing of the following typical actuator-based devices:

- A white light type of actuator,
- A coloured light type of actuator.

### 4.4.5 Sensor types

The test architecture should support testing of the following typical sensor-based devices:

- An occupancy detection sensor,
- A daylight measurement sensor.

A manual input device, such as a switch, may also be considered as a sensor, sensing the state of the switch.

### 4.5 Connections to BMS domain

The test setup provides an adapter to the BMS domain, to test the interaction with existing BMS interfaces, such as BACnet. The BMS adapter (Figure 10) need to map the OpenAIS object model to BACnet object types, and act as a communication gateway supporting the protocols at both sides.
As the adapter is part of the OpenAIS system, it needs to be commissioned, to get the right authorizations to access OpenAIS devices, either to control them, or to collect data from them. The adapter and related software, may be a device by itself or running on a PC environment.

The testing will mainly focus on two use cases:

- The BMS has a dashboard to report the real-time energy consumption of the test setup, with the granularity of consumption of each individual device.
- The BMS wants to deploy a load shedding operation into the test setup, e.g. as a scene to lower the energy consumption.

In addition to these two main use cases, following BMS integration aspects should be supported:

- Read/Write testing of single points, e.g. on/off of a single device
- Command scheduling
- BACnet auto-discovery

Some aspects related to data collection and aggregation depend on the OpenAIS data model [OPENAIS_D_D2.4], as the datacollect object could provide this. As an indication following functions are requested:

- Trend collection
- Accumulators (minimum, maximum, sum, average)
- Reporting (failures, errors, etc.)
5. Test Framework

The test frameworks are configured and programmed using Python [PYTH] scripting. Python is selected as it widely used within test automation (Figure 9 Test Architecture). There are several test automation frameworks available using Python; one is selected as central system and two others as test execution engines. Next to the test automation framework there will be Python-based virtual devices which interact with the OpenAIS system using COAP operations. A network emulation, also using Python, is added to extend the network, or to work with different network configurations using software-defined networking.

5.1 Test Automation Framework

One centralized test automation framework (Twister [TWIST]) is used to manage the test execution. The Twister framework uses Python as scripting language, which:

- Picks tests from repositories and delivers them to correct test execution engine.
- Loads functional modules and delivers them to the test execution engine.
- Collects the output from the test execution engines.

Two test execution engines will be used for different test purposes:

- The Robot Framework [ROBOT] is a test automation framework based on keyword-driven test approach. Several useful test libraries and tools are available. Robot Framework will be mainly used to test the basic functionality of the OpenAIS system, such a basic actuator functionality (ON/OFF).
- The Cucumber Framework [CUCUM] is a test automation framework for acceptance testing based on a behaviour-driven test approach. This framework will be mainly used to test use-cases related to device and user interactions.

5.2 Virtual devices

Virtual devices will embed instances of OpenAIS objects mainly to send out specific operations to initiate events or state changes in the system. These virtual devices will use PUT or POST operations, based on COAP messages. The virtual devices provide an API which is used by the test automation framework to select and initiate events. An example of an event, is “Occupancy detected” at a sensor. The virtual device is part of the OpenAIS system, and should be a commissioned, as such, to use the correct groups and access levels. The virtual device will be developed in Python, as there are open source COAP libraries available in Python.

5.3 Controlled OpenAIS devices

ARM’s mbed platform [MBED] offers a test environment called Greentea [GTEA] that allows for automated testing of (ARM) mbed enabled devices, using the actual device code. The test framework will make use of this provision to setup a controlled OpenAIS device.
Via a duplex serial port connection, the test automation system communicates with an mbed device (Figure 11). The intended tests run on the device under the supervision of the test automation system. The device code and related test scripts are built and flashed into the device under supervision of Greentea. Via the serial interface, test scripts can be initiated and results will be exchanged. This offers the possibility to initiate certain events on the OpenAIS network, which are controlled by the test script in the device, or to monitor for activity in the OpenAIS network. Benefit of this approach is the realistic interaction of a device with the OpenAIS system. The controlled OpenAIS device is based on real hardware and the actual software code of the device, so verification can be done using real-time processing. This Greentea test approach will be an integrated part of the test automation framework.

5.3.1 Wireless network traffic generator

A wireless network traffic generator will be used to create certain wireless network patterns, to verify the performance of the OpenAIS system. E.g. heavy network traffic to interfere with the OpenAIS device communication. The FRDM-K64F board with RF extension can be used to act as a wireless network pattern generator using the Greentea interface to control the device.
5.4 Network emulation

As network emulation Mininet [MINI] will be used. It can create and run a collection of end-hosts, switches, routers and links on a single Linux Kernel. It can run next to the application as it uses a lightweight virtualization (Figure 12).

- Mininet can send packets with a given link speed and delay like a real Ethernet interface.
- Packets get processed by what looks like a real Ethernet switch or router.
- Applications can run on a real hardware network in combination with the emulated network.

The test automation framework is able to interface with Mininet to deploy a certain network configuration, required for certain test cases.

5.5 Remote Testing

The Twister test automation framework (Figure 13 Twister System Architecture) allows for distributed test execution. The central engine provides a XML-RPC interface which allows test configuration to be managed and initiated from remote sites. The central system will be based in Eindhoven. The SUT's (System Under Test) are OpenAIS test system that resides at different locations (e.g. Eindhoven and Dornbirn)
Figure 13: Twister System Architecture
6. REFERENCES


[CUCUM] Cucumber test framework for behaviour-based testing: https://cucumber.io/


[OPENAIS_D1.4] OpenAIS deliverable - D1.4 Solution Specification report (due in M12)
[OPENAIS_D2.3] OpenAIS deliverable - D2.3 Final architecture of OpenAIS system (due in M12)
[OPENAIS_D2.5] OpenAIS deliverable - D2.5 Prototype of the object structure report (due in M24)
[OPENAIS_D4.2] OpenAIS deliverable - D4.2 FMEA and Hazard analysis report (due in M12)

[PYTH] Python programming language: https://www.python.org/


[RAUSAND] Marvin Rausand, Risk Assessment: Theory, Methods, and Applications
