**Software Engineering**

**Semester 3**

**Cable Modelling Wizard**

**System Requirements Specification**

**Creator Leon Amtmann**

**ID 5156023**

Table of Contents

[1 Purpose 3](#_Toc87367182)

[2 Product Environment 3](#_Toc87367183)

[3 Usecases 4](#_Toc87367184)

[3.1 <UC.001> New Cable Registration 4](#_Toc87367185)

[3.2 <UC.002> Deprecated cable 5](#_Toc87367186)

[3.3 <UC.003> Edit Existing Cable Model 6](#_Toc87367187)

[3.4 <UC.004> Lookup/Search of Cable Information 7](#_Toc87367188)

[3.5 <UC.005> Download of Cable Information in AML format 8](#_Toc87367189)

[4 Product Requirements 9](#_Toc87367190)

[4.1 /LF10/ Get Folder & File Structure 9](#_Toc87367191)

[4.2 /LF20/ Generate Display List 9](#_Toc87367192)

[4.3 /LF30/ Refresh Display Page 9](#_Toc87367193)

[4.4 /LF40/ Delete File from File System 9](#_Toc87367194)

[4.5 /LF50/ Search Display List for String 9](#_Toc87367195)

[4.6 /LF60/ Navigation 10](#_Toc87367196)

[4.7 /LF70/ Cable Detail View 10](#_Toc87367197)

[4.8 /LF80/ Cable Creation Data Input 10](#_Toc87367198)

[4.9 /LF90/ Save New Cable Data 10](#_Toc87367199)

[4.10 /LF100/ Export Cable Data as AML 10](#_Toc87367200)

[5 Product Data 11](#_Toc87367201)

[6 Non-Functional Requirements 11](#_Toc87367202)

# Purpose

The ultimate end goal of this software shall be to create cable models though a web-based interface written in Angular. The software shall then be capable of exporting the created cable models in AutomationML-Format utilizing CAEX 2.0 and 3.15.

# Product Environment

The resulting application shall be able to run in a docker container, ensuring portability between systems and future-proofing for a cloud-native environment.

The product shall be designed to assist in managing cables, not as an inventory management system with counts and locations, instead as a repository of potentially available cables inside or from a company.

# Usecases

## <UC.001> New Cable Registration

|  |  |
| --- | --- |
| **Related Business Process:** | New Cable is registered in inventory system |
| **Use Cases Objective:** | User wants to store information about a new cable in a safe environment where it is easily accessible |
| **System Boundary:** | Inventory system |
| **Precondition:** | The cable must not be already registered, the program has to run without errors. |
| **Postcondition on success:** | The cable is successfully registered with all specifications |
| **Involved Users:** | User and inventory system |
| **Triggering Event:** | The user acquires a new cable which they want to be registered in the inventory system |

## <UC.004> Lookup/Search of Cable Information

|  |  |
| --- | --- |
| **Related Business Process:** | Customer Lookup of Cable Information |
| **Use Cases Objective:** | User wants to access the information and data of a specific cable with known identifier |
| **System Boundary:** | Application |
| **Precondition:** | The user must be aware of the exact identifier of the cable |
| **Postcondition on success:** | The correct cable is returned to the user |
| **Involved Users:** | User and Application |
| **Triggering Event:** | The user requires information on a specific cable for any reason |

## <UC.005> Download of Cable Information in AML format

|  |  |
| --- | --- |
| **Related Business Process:** | Customer Lookup of Cable Information |
| **Use Cases Objective:** | User wants to download the information of a cable in AML format |
| **System Boundary:** | Application |
| **Precondition:** | The user must be aware of the exact identifier of the cable the user wants to download |
| **Postcondition on success:** | The download of cable information is initiated by the application |
| **Involved Users:** | User and Application |
| **Triggering Event:** | The user requires the portability of cable data for any reason, such as compatibility or use-case-analysis on the user side |

# Product Requirements

The following functions shall be implemented in the applications.

## /LF10/ Get Folder & File Structure

To ensure a pleasant user experience, the application must be able to present the user with a listing of currently available cable connectors. To this end, the application must gather all the files inside a folder and be able to display them to the user. This part of the function shall ensure that the application can be pointed to a folder location and report back all the files it encounters.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| Folder path | File containing the file and folder names of the provided folder path |

## /LF20/ Generate Display List

The application shall be able to interpret a description file containing file locations and display them as a listing of cable connectors with accompanying information.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| file describing files in a folder | List of cable connectors located inside the folder described by the file |

## /LF30/ Refresh Display Page

The application shall be able to refresh the list of available cable connectors.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| Current display list | Updated display list containing newly added cable connectors |

## /LF40/ Delete File from File System

The application must be able to delete a cable connector from the file system to remove it from the list of available cable connectors.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| Path to file to delete | Acknowledgement of successful file delection |

## /LF50/ Search Display List for String

The application shall be capable of receiving a search string by a user and then filter the current display list for that string. This way, the listed cable connectors can be narrowed down.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| Search string | New display list containing only listings matching the search string |

## /LF60/ Navigation

The user shall be able to navigate through the cable connectors. To this end, the application must be able to accept clicks on the current display list of cables and open up the selected cable connector for viewing in a separate details page.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| Click on specific cable | Clearing of current display list and entering of connector detail view |

## /LF70/ Cable Detail View

The application shall be able to display the details to any cable model in a list of cables.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| Path to cable model | Cable detail view containing the cable model and a way to initiate data download as well as a way to edit cable details |

## /LF80/ Save New Cable Data

The application shall be able to take the user input data as file and save it in a suitable location on disk.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| Path to unused file | File at the expected location |

## /LF90/ Export Cable Data as AML

If not yet so, the application shall be able to convert a cable model to AML compliant format and export it for user download.

|  |  |
| --- | --- |
| Input Description | Expected Output |
| File path to cable model | AML-compliant file |

# Non-functional Requirements

## Security

The system shall not need authentication by password, because the application shall only be run internally and should be accessible by all personnel dealing with the creation and management of cables inside the company. Storage of proprietary information in the application will neither be supported nor recommended.

## Reliability

The application must be designed for optimal uptime. In the event of a fatal crash, the application container shall signal failure using an integrated health check. Outside action shall be necessary to determine cause of failure and to rectify any bugs.

## Performance

The application shall be as performant as possible. Angular is known for its relatively high resource consumption in comparison to native applications, but any sufficiently powerful server that can run company operations shall also be capable of running this application.

## Maintainability

The application container shall be easy to maintain by employing a code version control system that makes code changes over time obvious to any future maintainer.

## Scalability

Each instance of the application shall be able to support one concurrent user. If more users are desired to work simultaneously with different cable models, more application instances shall be able to be spun up using docker or some other container orchestration tool.

## Future-Proofing

The application shall be delivered packaged inside a docker container, ensuring compatibility with the emerging industry trend of running on premise applications in cloud-like container orchestration software.