|  |  |
| --- | --- |
| **Open Issue** | **Issue description** |
| - | - |

Table of Contents

[1 DOCUMENT INTRODUCTION 3](#_Toc488132997)

[1.1 Purpose 3](#_Toc488132998)

[1.2 CDP2 Overall Connectivity architecture 3](#_Toc488132999)

[1.3 Scope 3](#_Toc488133000)

[1.4 References 3](#_Toc488133001)

[1.5 Terminology & Abbreviations 3](#_Toc488133002)

[2 Test Method 5](#_Toc488133003)

[3 Sampling Method 5](#_Toc488133004)

[4 Test Set-up 5](#_Toc488133005)

[5 Test Conditions 5](#_Toc488133006)

[6 Test Cases 6](#_Toc488133007)

[6.1 Manual Test Cases 6](#_Toc488133008)

[6.2 Automated Test Cases 20](#_Toc488133009)

[7 Test Instructions 23](#_Toc488133010)

[8 Revision History 23](#_Toc488133011)

[9 Approval 24](#_Toc488133012)

# DOCUMENT INTRODUCTION

## Purpose

This document describes the test cases (test scenarios) to be used for the verification of CommLib, demonstrating it meets the requirements, see ref. [SwRS], of CommLib.

## CDP2 Overall Connectivity architecture

Below picture shows the overall connectivity architecture and components. The components inside the blue box are described in this document.



Figure 1 - CDP2 Overall Connectivity architecture

## Scope

This document applies to CommLib (Android and iOS), to be used in connected digital propositions.

## References

| **Reference** | **Identification** | **Title / additional remarks** |
| --- | --- | --- |
| [SwRS] | CML000001 | Requirements, CommLib |
| [DiComm] | JohSun-20160115-04V01 | SwIS DIComm Protocol |
| [SwTM] | CML000007 | RequirementsTraceability Matrix, CommLib |

## Terminology & Abbreviations

| **Terminology & Abbreviations** | **Description/Definition** |
| --- | --- |
| BLE | Bluetooth Low Energy |
| DiComm | Digital Interface Communication Protocol, see ref [DiComm] |
| Gherkin | Gherkin is a Business Readable, Domain Specific Language that lets you describe software’s behavior without detailing how that behavior is implemented. It is used to specify test scenarios that can be interpreted by tools like: cucumber and SpecFlow. Reference: The Cucumber Book (ISBN 978-1-93435-680-7) |
| TC | Test Case |

# Test Method

The test scenarios are written in the [Gherkin] design specification language. The [Gherkin] design specification language provides human-readable scenarios and steps (Given, When, Then) that could be executed either manually or automatically. The test scenarios shall verify that the requirements are met and shall be understandable by non-technical business owners.

A test scenario is part of a [Gherkin] feature file.  
A feature file contains one or more scenarios.

A requirement Id specified in the requirements document [SwRS] is tested on its key elements by one or more scenarios.

# Sampling Method

Not applicable. For the software tests no sampling method is applied.

# Test Set-up

Manual test cases require:

* Android and iOS phones
* Test Apps
* BLE Reference Node
* WiFI Reference Node
* Product Stub running on a Windows machine
* AirPurifier
* two WiFi networks with internet access.

The following Test Apps are used to verify CommLib:

* Commlib-ble-testapp (Android)
* Commlib-demoapp (Android)
* Commlib-explorer (Android)
* Commlib-demoapp (iOS)

Firmware revisions for Reference Nodes:

* BLE Reference Node firmware: 2017.5.0+ca8a385b
* WiFi Reference Node firmware: 2017.5.0+61ca348d

BLE test scenarios must be executed with multiple Android phones from multiple manufacturers and a variation in Android releases.

# Test Conditions

The test conditions are fully contained inside the [Gherkin] test scenarios themselves.

For BLE test scenarios it is important that the physical distance between the phone and the BLE Reference Node is 50 cm. Changing the physical distance may impact the test results.

# Test Cases

The test cases are specified by using the Gherkin specification language. Each test case is a Gherkin Scenario.

The test cases (scenarios) used to verify the requirements of CommLib can be found in TFS under Test Plan: **Platform: cml\_bll Test design (Id:14718)**.

The requirements traceability matrix [SwTM] shows what requirement is verified by what test case (scenario).

## Manual Test Cases

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Test Case ID | Test Case Title | Steps | | 14726 | Scenario: PA001 Pairing happy path | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And a discovered Airpurifier appliance | |  |  | Step 5. And the app is "not paired" with the appliance | |  |  | Step 6. And the mobile device is using connection "local" | |  |  | Step 7. When app requests to pair with Airpurifier | |  |  | Step 8. Then the pairing request is successful within 30 seconds | | 14734 | Scenario Outline: PA002 Pairing already paired | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And a discovered Airpurifier appliance | |  |  | Step 5. And the app is "paired" with the appliance | |  |  | Step 6. And the mobile device is using connection '@connection' | |  |  | Step 7. When app requests to pair with Airpurifier | |  |  | Step 8. Then the pairing request is successful within 30 seconds | | 14743 | Scenario Outline: PA003 Cannot pair via wrong connection | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And an Airpurifier appliance | |  |  | Step 5. And the app is “not paired" with the appliance | |  |  | Step 6. And the mobile device is using connection '@connection' | |  |  | Step 7. When app requests to pair with Airpurifier | |  |  | Step 8. Then the pairing request is unsuccessful within 40 seconds | | 14802 | Scenario: PA004 Cannot re-pair when there is no connection | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And an Airpurifier appliance | |  |  | Step 5. And the app is “paired" with the appliance | |  |  | Step 6. And the mobile device is using connection 'none' | |  |  | Step 7. When app requests to pair with Airpurifier | |  |  | Step 8. Then the pairing request is unsuccessful within 40 seconds | | 14803 | Scenario Outline: PA005 Unpair happy path | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And an Airpurifier appliance | |  |  | Step 5. And the app is “paired" with the appliance | |  |  | Step 6. And the mobile device is using connection ‘@connection’ | |  |  | Step 7. When app requests to unpair from Airpurifier | |  |  | Step 8. Then the unpair request is successful within 40 seconds | | 14805 | Scenario: PA006 Cannot unpair when there is no connection | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And an Airpurifier appliance | |  |  | Step 5. And the app is “paired" with the appliance | |  |  | Step 6. And the mobile device is using connection ‘none’ | |  |  | Step 7. When app requests to unpair from Airpurifier | |  |  | Step 8. Then the unpair request is unsuccessful within 40 seconds | | 14813 | Scenario: PA007 Cannot unpair when appliance is not paired and there is no connection | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And an Airpurifier appliance | |  |  | Step 5. And the app is “not paired" with the appliance | |  |  | Step 6. And the mobile device is using connection ‘none’ | |  |  | Step 7. When app requests to unpair from Airpurifier | |  |  | Step 8. Then the unpair request is unusccessful within 40 seconds | | 14822 | Scenario Outline: PA008 Unpair when appliance is not paired | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And an Airpurifier appliance | |  |  | Step 5. And the app is “not paired" with the appliance | |  |  | Step 6. And the mobile device is using connection ‘@connection’ | |  |  | Step 7. When app requests to unpair from Airpurifier | |  |  | Step 8. Then the unpair request is successful within 40 seconds | | 14867 | Scenario: FW001 Perform a firmware upgrade for a valid upgrade package | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And firmware upgrade push timeout is set to 30 seconds | |  |  | Step 4. And application starts firmware upgrade push of Test.Latest.upg | |  |  | Step 5. And firmware upgrade push is successful | |  |  | Step 6. When firmware upgrade deployment timeout is set to 30 seconds | |  |  | Step 7. And application starts firmware upgrade deployment | |  |  | Step 8. Then firmware upgrade deployment is successful | | 14868 | Scenario: FW002 Cancel Firmware Upgrade push | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application starts firmware upgrade push of Test.Baseline.upg | |  |  | Step 4. When firmware upgrade cancel timeout is set to 30 seconds | |  |  | Step 5. And application cancels firmware upgrade push | |  |  | Step 6. Then firmware upgrade push is stopped | |  |  | Step 7. And application starts firmware upgrade push of Test.Latest.upg | |  |  | Step 8. And firmware upgrade push is successful | |  |  | Step 9. And application starts firmware upgrade deployment | |  |  | Step 10. And firmware upgrade deployment is successful | | 14869 | Scenario: FW003 Perform a firmware upgrade for an upgrade package with an invalid header version | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application starts firmware upload of Test.Product.InvalidHeaderVersion.upg | |  |  | Step 4. And firmware upload is successful | |  |  | Step 5. When application starts firmware deployment | |  |  | Step 6. Then firmware deployment is unsuccessful | |  |  | Step 7. And error message is displayed | | 14870 | Scenario: FW004 Perform a firmware upgrade for an upgrade package with an invalid product | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application starts firmware upload of Test.Product.InvalidProduct.upg | |  |  | Step 4. And firmware upload is successful | |  |  | Step 5. When application starts firmware deployment | |  |  | Step 6. Then firmware deployment is unsuccessful | |  |  | Step 7. And error message is displayed | | 14873 | Scenario: FW005 Perform a firmware upgrade for an upgrade package with an invalid signature | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application starts firmware upload of Test.Product.InvalidSignature.upg | |  |  | Step 4. And firmware upload is successful | |  |  | Step 5. When application starts firmware deployment | |  |  | Step 6. Then firmware deployment is unsuccessful | |  |  | Step 7. And error message is displayed | | 16043 | Non-discoverable peripheral is removed from list of discovered peripherals, when discovery is active | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started BLE discovery | |  |  | Step 4. And modelId filter is empty | |  |  | Step 5. And a peripheral is discovered and added to the list of discovered peripherals | |  |  | Step 6. When peripheral becomes non-discoverable (e.g. out of reach) | |  |  | Step 7. Then peripheral is removed from the list of discovered peripherals within 120 seconds | | 16044 | Lost peripheral is rediscovered | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started BLE discovery | |  |  | Step 4. And modelId filter is empty | |  |  | Step 5. And a peripheral is discovered and added to the list of discovered peripherals | |  |  | Step 6. And peripheral becomes non-discoverable (e.g. out of reach) | |  |  | Step 7. And peripheral is removed from the list of discovered peripherals | |  |  | Step 8. When peripheral becomes discoverable (e.g. within reach) | |  |  | Step 9. Then peripheral is added to the list of discoverable peripherals within 120 seconds | | 16098 | Discoverable peripheral is removed from list of discovered peripherals, when discovery is stopped | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started BLE discovery | |  |  | Step 4. And modelId filter is empty | |  |  | Step 5. And a peripheral is discovered and added to the list of discovered peripherals | |  |  | Step 6. When application stops BLE discovery | |  |  | Step 7. Then peripheral is removed from the list of discovered peripherals within 120 seconds | | 16629 | A peripheral which has been connected once is added to persistent storage | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a mobile device | |  |  | Step 4. And the discovered Airpurifier is not in the list of managed appliances | |  |  | Step 5. And the mobile device is using connection "local" | |  |  | Step 6. And the test application is used to toggle the Airpurifier light | |  |  | Step 7. And the test application is killed | |  |  | Step 8. When the test application is started | |  |  | Step 9. Then the discovered Airpurifier is in the list of managed appliances | | 16785 | Discoverable peripheral is added to list of discovered peripherals | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started BLE discovery | |  |  | Step 4. And modelId filter is empty | |  |  | Step 5. When a peripheral becomes discoverable | |  |  | Step 6. Then peripheral is added to the list of discovered peripherals within 120 seconds | | 16790 | Discoverable peripheral stays in list of discovered peripherals | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started BLE discovery | |  |  | Step 4. And modelId filter is empty | |  |  | Step 5. And a peripheral is discovered and added to the list of discovered peripherals | |  |  | Step 6. When peripheral remains discoverable for 2 minutes | |  |  | Step 7. Then peripheral stays in the list of discovered peripherals | | 16933 | Scenario: FW006 Firmware Upgrade push times out | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And firmware upgrade push timeout is set to 50 milliseconds | |  |  | Step 4. And application starts firmware upgrade push of Test.Baseline.upg | |  |  | Step 5. When firmware upgrade push timeout expires | |  |  | Step 6. Then firmware upgrade push is unsuccessful | |  |  | Step 7. And error message shows that 'push failed with timeout' | | 16934 | Scenario: FW007 Firmware Upgrade deployment times out | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And firmware upgrade push timeout is set to 30 seconds | |  |  | Step 4. And application start firmware upgrade push of Test.Latest.upg | |  |  | Step 5. And firmware upgrade push is successful | |  |  | Step 6. And firmware upgrade deployment timeout is set to 1 second | |  |  | Step 7. And application starts firmware upgrade deployment | |  |  | Step 8. When firmware upgrade deployment times out | |  |  | Step 9. Then firmware upgrade deployment is unsuccessful | |  |  | Step 10. And error message shows that 'deployment failed with timeout' | | 17177 | Scenario: FW008 Communication error during firmware upload | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application starts firmware upload of Test.Latest.upg | |  |  | Step 4. When BLE Reference Node is reset | |  |  | Step 5. Then firmware upload is unsuccessful | |  |  | Step 6. And error message is displayed | | 17187 | Get value from time port | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And distance between phone and BLE Reference Node is 50 cm | |  |  | Step 4. And a BLE Reference Node is discovered and selected | |  |  | Step 5. And "Stay connected" is disabled | |  |  | Step 6. And "Subscribe" is disabled | |  |  | Step 7. And "Continuously get Time" is disabled | |  |  | Step 8. When application requests time value from time port | |  |  | Step 9. Then time value is received without errors | | 17188 | Get value from time port continuously, disconnect after request | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And distance between phone and BLE Reference Node is 50 cm | |  |  | Step 4. And a BLE reference Node is discovered and selected | |  |  | Step 5. And "Stay connected" is disabled | |  |  | Step 6. And "Subscribe" is disabled | |  |  | Step 7. When application selects "continuously get time" | |  |  | Step 8. And the application requests the time 50 times | |  |  | Step 9. Then time value is received continuously without errors | | 17189 | Get value from time port continuously, stay connected | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And distance between phone and BLE Reference Node is 50 cm | |  |  | Step 4. And a BLE Reference Node is discovered and selected | |  |  | Step 5. And "Stay connected" is enabled | |  |  | Step 6. When application selects "continuously get time" | |  |  | Step 7. And the application requests the time 50 times | |  |  | Step 8. Then time value is received continuously without errors | | 17191 | Get value from time port continuously, via subscription/notification interface | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And distance between phone and BLE Reference Node is 50 cm | |  |  | Step 4. And BLE Reference Node is discovered and selected | |  |  | Step 5. And "Stay connected" is disabled | |  |  | Step 6. And "Continuously get Time" is disabled | |  |  | Step 7. When "Subscribe" is enabled | |  |  | Step 8. And the application receives the time 50 times | |  |  | Step 9. Then time value is received continuously without errors | | 17193 | Put value on device port | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And distance between phone and BLE Reference Node is 50 cm | |  |  | Step 4. And a BLE Reference Node is discovered and selected | |  |  | Step 5. When Device Name value on device port is changed | |  |  | Step 6. Then Device Name value is successfully changed | | 17194 | Get value from time port continuously, communication error | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And distance between phone and BLE Reference Node is 50 cm | |  |  | Step 4. And a BLE Reference Node is discovered and selected | |  |  | Step 5. And "Stay connected" is disabled | |  |  | Step 6. And "Subscribe" is disabled | |  |  | Step 7. And application selects "continuously get time" | |  |  | Step 8. When BLE Reference Node is reset | |  |  | Step 9. Then get time is unsuccessful | |  |  | Step 10. And error message is displayed | | 17383 | Discoverable peripheral with ModelId which matches ModelId filter is added to list of discovered peripherals | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And ModelId filter is set to "AB3456,PS1234,CD7890" | |  |  | Step 4. And application has started BLE discovery | |  |  | Step 5. When a peripheral with ModelId "PS1234" becomes discoverable | |  |  | Step 6. Then peripheral is added to list of discovered peripherals within 120 seconds | | 17752 | Discoverable peripheral with ModelId which does not match ModelId filter is not added to list of discovered peripherals | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And ModelId filter is set to "AB3456,CD7890,PS12" | |  |  | Step 4. And application has started BLE discovery | |  |  | Step 5. When a peripheral with ModelId "PS1234" becomes discoverable | |  |  | Step 6. Then peripheral is not added to list of discovered peripherals | | 17847 | Scenario: FW009 Firmware Upgrade cancel times out | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And firmware upgrade push timeout is set to 30 seconds | |  |  | Step 4. And application starts firmware upgrade push of Test.Latest.upg | |  |  | Step 5. And firmware upgrade cancel timeout is set to 10 milliseconds | |  |  | Step 6. And application cancels firmware upgrade push | |  |  | Step 7. When firmware upgrade cancel timeout expired | |  |  | Step 8. Then error message shows that 'cancel failed with timeout' | | 18489 | Non-discoverable peripheral is removed from list of discovered peripherals, when peripheral is switched off | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started LAN discovery | |  |  | Step 4. And a peripheral is discovered and added to the list of discovered peripherals | |  |  | Step 5. When peripheral becomes non-discoverable by switching it off | |  |  | Step 6. Then peripheral is removed from the list of discovered peripherals within 120 seconds | | 18492 | Non-discoverable peripheral is removed from list of discovered peripherals, when network is switched | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started LAN discovery | |  |  | Step 4. And a peripheral is discovered and added to the list of discovered peripherals | |  |  | Step 5. When peripheral becomes non-discoverable by switching the network on the phone | |  |  | Step 6. Then peripheral is removed from the list of discovered peripherals within 120 seconds | | 18508 | Lost peripheral is rediscovered | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started LAN discovery | |  |  | Step 4. And a peripheral is discovered and added to the list of discoverable peripherals | |  |  | Step 5. And peripheral becomes non-discoverable by switching it off | |  |  | Step 6. And peripheral is removed from the list of discovered peripherals | |  |  | Step 7. When peripheral becomes discoverable by switching it on | |  |  | Step 8. Then peripheral is added to the list of discoverable peripherals within 120 seconds | | 18510 | Discoverable peripheral is added to list of discovered peripherals | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started LAN discovery | |  |  | Step 4. When a peripheral becomes available | |  |  | Step 5. Then peripheral is added to the list of discovered peripherals within 120 seconds | | 18511 | Discoverable peripheral stays in list of discovered peripherals | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started LAN discovery | |  |  | Step 4. And a peripheral is discovered and added to the list of discovered peripherals | |  |  | Step 5. When peripheral remains discoverable for 2 minutes | |  |  | Step 6. Then peripheral stays in the list of discovered peripherals | | 22751 | CppId of discovered peripheral is available | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started BLE discovery | |  |  | Step 4. And modelId filter is empty | |  |  | Step 5. When a peripheral becomes discoverable | |  |  | Step 6. Then CppId of discovered peripheral is shown in application | | 22760 | ModelId of discovered peripheral, with a ModelId, is available | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started BLE discovery | |  |  | Step 4. And modelId filter is empty | |  |  | Step 5. When a peripheral with a ModelId becomes discoverable | |  |  | Step 6. Then ModelId of discovered peripheral is shown in application | | 22764 | ModelId of discovered peripheral, without a ModelId, is not available | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And application has started BLE discovery | |  |  | Step 4. And modelId filter is empty | |  |  | Step 5. When a peripheral without a ModelId becomes discoverable | |  |  | Step 6. Then ModelId of discovered peripheral is not shown in application | | 23990 | User pairing | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to one Wifi network | |  |  | Step 3. And commLib-demoapp is started | |  |  | Step 4. And LAN scanning strategy is selected | |  |  | Step 5. And Wifi Reference Node is discovered | |  |  | Step 6. And Wifi Reference Node is selected | |  |  | Step 7. When appliance-side pairing is requested with default values (as provided in the App) | |  |  | Step 8. Then commLib-demoapp shows that pairing has "succeeded" | |  |  | Step 9. And logging in device portal shows that AddRelationShipRequest results in '{ "RelationshipStatus": "pending" }' | | 31700 | CommLib upgrade from previous Platform release | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp (from previous Platform release) is installed and started on a mobile device | |  |  | Step 3. And a discovered Airpurifier appliance | |  |  | Step 4. And the mobile device is using connection "local" | |  |  | Step 5. And app requests to pair with Airpurifier | |  |  | Step 6. And the pairing request is successfull | |  |  | Step 7. When commlib-demoapp is upgraded to the new release | |  |  | Step 8. And the mobile device is using connection "remote" | |  |  | Step 9. Then the app can access the Airpurifier over the remote interface | | 32880 | Firmware Port canUpgrade property is provided | Step 1. Given Test Environment is logged | |  |  | Step 2. And commlib-demoapp is started | |  |  | Step 3. And a BLE Reference Node is discovered and selected | |  |  | Step 4. And Firmware Upgrade screen is open | |  |  | Step 5. When commlib-demoapp has received Firmware Port properties | |  |  | Step 6. Then commlib-demoapp shows that value of "canUpgrade" property is "Yes" | | 40231 | Use HTTPS to communicate over LAN with connectivity node which supports HTTPS | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to one Wifi network | |  |  | Step 3. And CommLib DemoApp is started | |  |  | Step 4. And LAN scanning strategy is selected | |  |  | Step 5. And Wifi Reference Node is discovered | |  |  | Step 6. And Wifi Reference Node is selected | |  |  | Step 7. When CommLib DemoApp reads port properties of device port | |  |  | Step 8. Then logging shows that HTTPS is used as transfer protocol | |  |  | Step 9. And logging shows that properties are successfully retrieved | | 40541 | Use HTTP to communicate over LAN with connectivity node which only supports HTTP | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Airpurifier are connected to one Wifi network | |  |  | Step 3. And CommLib DemoApp is started | |  |  | Step 4. And LAN scanning strategy is selected | |  |  | Step 5. And Airpurifier is discovered | |  |  | Step 6. And Airpurifier is selected | |  |  | Step 7. When Light state is modified | |  |  | Step 8. Then logging shows that HTTP is used as transfer protocol | |  |  | Step 9. And logging indicates that property was successfully set | | 47366 | Use HTTPS to communicate over LAN with connectivity node which supports HTTPS | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to one Wifi network | |  |  | Step 3. And commlib-demoapp is started | |  |  | Step 4. And LAN scanning strategy is selected | |  |  | Step 5. And Wifi Reference Node is discovered | |  |  | Step 6. And Wifi Reference Node is selected | |  |  | Step 7. When commlib-demoapp reads port properties of device port | |  |  | Step 8. Then logging shows that HTTPS is used as transfer protocol | | 47904 | Use HTTPS to communicate over LAN with connectivity node which is not configured for a specific Hypertext Transfer Protocol | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Vacuum Cleaner are connected to one Wifi network | |  |  | Step 3. And commlib-demoapp is started | |  |  | Step 4. And Vacuum Cleaner is discovered | |  |  | Step 5. And Vacuum Cleaner is selected | |  |  | Step 6. When commlib-demoapp reads port properties of device port | |  |  | Step 7. Then logging shows that HTTPS is used as transfer protocol | |  |  | Step 8. And commlib-demoapp shows an error | | 48062 | Use HTTP to communicate over LAN with connectivity node which only supports HTTP | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Airpurifier are connected to one Wifi network | |  |  | Step 3. And commlib-demoapp is started | |  |  | Step 4. And Airpurifier is discovered | |  |  | Step 5. And Airpurifier is selected | |  |  | Step 6. When Light state is modified | |  |  | Step 7. Then logging shows that HTTP is used as transfer protocol | | 49881 | CommLib listens for, receives and decrypts DIComm ChangeIndication UDP packets from the connectivity node when subscription is active | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to one Wifi network | |  |  | Step 3. And commLib-demoapp is started | |  |  | Step 4. And Wifi Reference Node is discovered and selected | |  |  | Step 5. When "Subscribe to time port" is enabled | |  |  | Step 6. And "Enable Communication" is enabled | |  |  | Step 7. Then time value is received continuously without errors | | 50928 | CommLib listens for, receives and decrypts DIComm ChangeIndication UDP packets from a legacy connectivity node when subscription is active | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and an Air Purifier are connected to one Wifi network | |  |  | Step 3. And commlib-demoapp is started | |  |  | Step 4. And the Air Purifier is discovered and selected | |  |  | Step 5. And "Subscribe to air port" is enabled | |  |  | Step 6. When pressing the light switch on the Air Purifier | |  |  | Step 7. Then the light state is reflected in the app without errors | | 54008 | Trust certificate on first use | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to the same Wifi network | |  |  | Step 3. And a clean install of commlib-demoapp is performed (no pin is stored) | |  |  | Step 4. And commlib-demoapp is started | |  |  | Step 5. And WiFi Reference Node is discovered | |  |  | Step 6. And WiFi Reference Node is selected | |  |  | Step 7. When Device Name value is requested | |  |  | Step 8. Then certificate is received | |  |  | Step 9. And logging shows that pin is added for appliance | |  |  | Step 10. And Device Name value is displayed | | 54051 | Pin must be persisted for a connected managed appliance | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to the same Wifi network | |  |  | Step 3. And commlib-demoapp is installed and a pin has already been stored for Wifi Reference Node | |  |  | Step 4. And commlib-demoapp is started | |  |  | Step 5. And WiFi Reference Node is discovered | |  |  | Step 6. And WiFi Reference Node is selected | |  |  | Step 7. When Device Name value is requested | |  |  | Step 8. And Device Name value is displayed | |  |  | Step 9. And logging does not show that pin is added for appliance | | 54418 | Pins must be persisted for multiple connected managed appliances | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to the same Wifi network | |  |  | Step 3. And commlib-demoapp is installed and pins have already been stored for Wifi Reference Node 1 and Wifi Reference Node 2 | |  |  | Step 4. And commlib-demoapp is started | |  |  | Step 5. And Wifi Reference Node 1 is discovered | |  |  | Step 6. And Wifi Reference Node 1 is selected | |  |  | Step 7. When Device Name value is requested | |  |  | Step 8. And Device Name value is displayed | |  |  | Step 9. And logging does not show that pin is stored | |  |  | Step 10. And Wifi Reference Node 2 is discovered | |  |  | Step 11. And Wifi Reference Node 2 is selected | |  |  | Step 12. When Device Name value is requested | |  |  | Step 13. And Device Name value is displayed | |  |  | Step 14. And logging does not show that pin is added for appliance | | 58371 | Allow pin mismatch to be resolved | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to the same Wifi network | |  |  | Step 3. And commlib-demoapp is installed and a pin has already been stored for Wifi Reference Node | |  |  | Step 4. And commlib-demoapp is started | |  |  | Step 5. And WiFi Reference Node is discovered | |  |  | Step 6. And WiFi Reference Node is selected | |  |  | Step 7. And factory reset procedure of Wifi Reference Node is performed (as described in attachment of this shared step) | |  |  | Step 8. And Device Name value is requested | |  |  | Step 9. And certificate is received | |  |  | Step 10. And logging shows that pin mismatch is detected | |  |  | Step 11. And commlib-demoapp requests user to accept or to reject the new pin | |  |  | Step 12. When user accepts new pin | |  |  | Step 13. And Device Name value is requested | |  |  | Step 14. Then Device name value is displayed | |  |  | Step 15. And logging shows that re-pin occurred for appliance | | 59923 | Don't allow pin mismatch to be resolved | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to the same Wifi network | |  |  | Step 3. And commlib-demoapp is installed and a pin has already been stored for Wifi Reference Node | |  |  | Step 4. And commlib-demoapp is started | |  |  | Step 5. And WiFi Reference Node is discovered | |  |  | Step 6. And WiFi Reference Node is selected | |  |  | Step 7. And factory reset procedure of Wifi Reference Node is performed (as described in attachment of this shared step) | |  |  | Step 8. And Device Name value is requested | |  |  | Step 9. And certificate is received | |  |  | Step 10. And logging shows that pin mismatch is detected | |  |  | Step 11. And commlib-demoapp requests user to accept or to reject the new pin | |  |  | Step 12. And user rejects new pin | |  |  | Step 13. And appliance does not appear in "PIN MISMATCH" list | |  |  | Step 14. And WiFi Reference Node is selected | |  |  | Step 15. And Device Name value is requested | |  |  | Step 16. And certificate is received | |  |  | Step 17. And logging shows that pin mismatch is detected | |  |  | Step 18. And commlib-demoapp requests user to accept or to reject the new pin | |  |  | Step 19. When user cancels new pin | |  |  | Step 20. Then appliance appears in "PIN MISMATCH" list | |  |  | Step 21. And WiFi Reference Node is selected | |  |  | Step 22. And Device Name value is requested | |  |  | Step 23. When Device Name value is requested | |  |  | Step 24. Then certificate is received | |  |  | Step 25. And logging shows that pin mismatch is detected | |  |  | Step 26. And commlib-demoapp requests user to accept or to reject the new pin | |  |  | Step 27. When user accepts pin | |  |  | Step 28. And Device Name value is requested | |  |  | Step 29. And certificate is received | |  |  | Step 30. And Device name value is received | |  |  | Step 31. Then logging shows that pin is re-pinned | | 69961 | Use HTTPS to communicate over LAN with connectivity node which is not configured for a specific Hypertext Transfer Protocol | Step 1. Given Test Environment is logged | |  |  | Step 2. Given Mobile phone and Vacuum Cleaner are connected to one Wifi network | |  |  | Step 3. And commlib-demoapp is started | |  |  | Step 4. And Vacuum Cleaner is discovered | |  |  | Step 5. And Vacuum Cleaner is selected | |  |  | Step 6. When commlib-demoapp reads port properties of device port | |  |  | Step 7. Then logging shows that HTTPS is used as transfer protocol | |  |  | Step 8. And logging shows that get properties failed | | 78605 | Upgrade CommLib DemoApp when appliances are stored in persistent data | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone, Airpurifier and Wifi Reference Node are connected to one Wifi network | |  |  | Step 3. And commlib-demoapp from previous Platform release is installed and started | |  |  | Step 4. And Wifi Reference Node and Airpurifier are discovered | |  |  | Step 5. And Wifi Reference Node and Airpurifier are selected once and are available in list of managed devices | |  |  | Step 6. When commlib-demoapp is upgraded to new release and started | |  |  | Step 7. Then Wifi Reference Node and Airpurifier are in list of managed devices | |  |  | Step 8. And commlib-demoapp can read Device name of Wifi Reference Node and Airpurifier | | 97193 | Automatic resubscribe when subscription expires | Step 1. Given Test Environment is logged | |  |  | Step 2. And Mobile phone and Wifi Reference Node are connected to the same Wifi network | |  |  | Step 3. And commlib-demoapp is started | |  |  | Step 4. And WiFi Reference Node is discovered | |  |  | Step 5. And WiFi Reference Node is selected | |  |  | Step 6. When Subscribe on Time Port is selected | |  |  | Step 7. And User waits for 6 minutes | |  |  | Step 8. Then time values are received continuously without errors | |  |  |
|  |  |  |
|  |  |  |

## Automated Test Cases

To read the steps of a scenario, open its feature file in any text editor.

The feature files are listed in the table below:

|  |  |  |
| --- | --- | --- |
| Test Case ID | Test Case Title | Steps |
| 98571 | Scenario Outline: Accepts initial certificate | Step 1. Given an appliance with id 'x' |
|  |  | Step 2. And appliance 'x' is configured with the certificate @certificate |
|  |  | Step 3. When reloading the device port for appliance 'x' |
|  |  | Step 4. Then the certificate is accepted |
| 98572 | Scenario: Communicates when identity can be confirmed | Step 1. Given an appliance with id 'x' |
|  |  | Step 2. And appliance 'x' is configured with the certificate FirstCertificate |
|  |  | Step 3. And reloading the device port for appliance 'x' |
|  |  | Step 4. When reloading the device port for appliance 'x' |
|  |  | Step 5. Then the certificate is accepted |
| 98573 | Scenario: Mismatch occurs when different certificate is received | Step 1. Given an appliance with id 'x' |
|  |  | Step 2. And appliance 'x' is configured with the certificate FirstCertificate |
|  |  | Step 3. And reloading the device port for appliance 'x' |
|  |  | Step 4. And appliance 'x' is configured with the certificate SecondCertificate |
|  |  | Step 5. When reloading the device port for appliance 'x' |
|  |  | Step 6. Then a pin mismatch is detected |
| 98574 | Scenario: Accepts communication when different certificate is received | Step 1. Given an appliance with id 'x' |
|  |  | Step 2. And appliance 'x' is configured with the certificate FirstCertificate |
|  |  | Step 3. And reloading the device port for appliance 'x' |
|  |  | Step 4. And appliance 'x' is configured with the certificate SecondCertificate |
|  |  | Step 5. And reloading the device port for appliance 'x' |
|  |  | Step 6. And a pin mismatch is detected |
|  |  | Step 7. When the new certificate is accepted for appliance 'x' |
|  |  | Step 8. And reloading the device port for appliance 'x' |
|  |  | Step 9. Then the certificate is accepted |
| 98575 | Scenario: Rejects communication when different certificate is received | Step 1. Given an appliance with id 'x' |
|  |  | Step 2. And appliance 'x' is configured with the certificate FirstCertificate |
|  |  | Step 3. And reloading the device port for appliance 'x' |
|  |  | Step 4. And appliance 'x' is configured with the certificate SecondCertificate |
|  |  | Step 5. And reloading the device port for appliance 'x' |
|  |  | Step 6. And a pin mismatch is detected |
|  |  | Step 7. When the new certificate is rejected for appliance 'x' |
|  |  | Step 8. And reloading the device port for appliance 'x' |
|  |  | Step 9. Then a pin mismatch is detected |
| 98576 | Scenario: Accepts communication when original certificate is received again | Step 1. Given an appliance with id 'x' |
|  |  | Step 2. And appliance 'x' is configured with the certificate FirstCertificate |
|  |  | Step 3. And reloading the device port for appliance 'x' |
|  |  | Step 4. And appliance 'x' is configured with the certificate SecondCertificate |
|  |  | Step 5. And reloading the device port for appliance 'x' |
|  |  | Step 6. And a pin mismatch is detected |
|  |  | Step 7. When the new certificate is rejected for appliance 'x' |
|  |  | Step 8. And appliance 'x' is configured with the certificate FirstCertificate |
|  |  | Step 9. And reloading the device port for appliance 'x' |
|  |  | Step 10. Then the certificate is accepted |
| 99047 | Scenario: Strategy is not available | Step 1. Given the BLE communication strategy is initialized with id 'x' |
|  |  | Step 2. Then the BLE communication strategy is not available |
| 99048 | Scenario: Strategy is available | Step 1. Given a mock device is found with id 'x' |
|  |  | Step 2. And the BLE communication strategy is initialized with id 'x' |
|  |  | Step 3. Then the BLE communication strategy is available |
| 99049 | Scenario: Strategy is available when device is found later | Step 1. Given the BLE communication strategy is initialized with id 'x' |
|  |  | Step 2. And a mock device is found with id 'x' |
|  |  | Step 3. Then the BLE communication strategy is available |
| 99050 | Scenario: Strategy is not available, different device id | Step 1. Given a mock device is found with id 'y' |
|  |  | Step 2. And the BLE communication strategy is initialized with id 'x' |
|  |  | Step 3. Then the BLE communication strategy is not available |
| 99064 | Scenario: CommLib starts discovery | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. When starting discovery for BLE appliances |
|  |  | Step 4. Then startScanning is called 1 time on BlueLib |
| 99065 | Scenario: CommLib stops discovery | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. And starting discovery for BLE appliances |
|  |  | Step 4. When stopping discovery for BLE appliances |
|  |  | Step 5. Then stopScanning is called on BlueLib |
| 99066 | Scenario: Commlib stops discovery when not started, silent ignore | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. When stopping discovery for BLE appliances |
|  |  | Step 4. Then stopScanning is called on BlueLib |
| 99067 | Scenario: CommLib discovers a supported appliance | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliances: mixer toothbrush |
|  |  | Step 3. When starting discovery for BLE appliances |
|  |  | Step 4. And toothbrush1 is discovered 1 time by BlueLib |
|  |  | Step 5. Then the following appliance is in the list of available appliances: toothbrush1 |
| 99068 | Scenario: CommLib discovers multiple supported appliances of different type | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliances: mixer shaver |
|  |  | Step 3. When starting discovery for BLE appliances |
|  |  | Step 4. And mixer1 is discovered 1 time by BlueLib |
|  |  | Step 5. And shaver1 is discovered 1 time by BlueLib |
|  |  | Step 6. Then the following appliances are in the list of available appliances: mixer1 shaver1 |
| 99069 | Scenario: CommLib discovers multiple supported appliances of same type | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliances: mixer shaver |
|  |  | Step 3. When starting discovery for BLE appliances |
|  |  | Step 4. And shaver1 is discovered 1 time by BlueLib |
|  |  | Step 5. And shaver2 is discovered 1 time by BlueLib |
|  |  | Step 6. Then the following appliances are in the list of available appliances: shaver1 shaver2 |
| 99070 | Scenario: CommLib discovers only supported appliances | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. When starting discovery for BLE appliances |
|  |  | Step 4. And mixer1 is discovered 1 time by BlueLib |
|  |  | Step 5. And shaver1 is discovered 1 time by BlueLib |
|  |  | Step 6. Then the following appliance is in the list of available appliances: shaver1 |
| 99071 | Scenario: CommLib discovers no supported appliances | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. When starting discovery for BLE appliances |
|  |  | Step 4. And mixer1 is discovered 1 time by BlueLib |
|  |  | Step 5. Then the length of the list of available appliances is 0 |
| 99072 | Scenario: CommLib discovers appliance once | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for BLE appliances |
|  |  | Step 4. When shaver1 is discovered 2 times by BlueLib |
|  |  | Step 5. Then the following appliance is in the list of available appliances: shaver1 |
|  |  | Step 6. And the length of the list of available appliances is 1 |
| 99073 | Scenario: CommLib discovers supported appliance after discovery was stopped | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. And starting discovery for BLE appliances |
|  |  | Step 4. And stopping discovery for BLE appliances |
|  |  | Step 5. And stopScanning is called on BlueLib |
|  |  | Step 6. When mixer1 is discovered 1 time by BlueLib |
|  |  | Step 7. Then the following appliance is in the list of available appliances: mixer1 |
| 99074 | Scenario: CommLib discovers a supported appliance with a specific model id | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliances: mixer toothbrush |
|  |  | Step 3. When starting discovery for BLE appliances with model ids: MX0001 TB0001 |
|  |  | Step 4. And toothbrush1 is discovered 1 time by BlueLib, matching model id TB0001 |
|  |  | Step 5. Then the following appliance is in the list of available appliances: toothbrush1 |
| 99075 | Scenario: CommLib does not discover supported appliance with a specific model id | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliances: mixer toothbrush |
|  |  | Step 3. When starting discovery for BLE appliances with model ids: MX0001 TB0001 |
|  |  | Step 4. And toothbrush1 is discovered 1 time by BlueLib, matching model id TB0000 |
|  |  | Step 5. Then the length of the list of available appliances is 0 |
| 99076 | Scenario: CommLib discovers multiple supported appliances with a specific model id | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliances: mixer |
|  |  | Step 3. When starting discovery for BLE appliances with model ids: MX0001 MX0002 MX0003 |
|  |  | Step 4. And mixer1 is discovered 1 time by BlueLib, matching model id MX0001 |
|  |  | Step 5. And mixer2 is discovered 1 time by BlueLib, matching model id MX0002 |
|  |  | Step 6. And mixer3 is discovered 1 time by BlueLib, matching model id MX0003 |
|  |  | Step 7. And mixer4 is discovered 1 time by BlueLib, matching model id MX0004 |
|  |  | Step 8. Then the following appliances are in the list of available appliances: mixer1 mixer2 mixer3 |
|  |  | Step 9. And the following appliance is not in the list of available appliances: mixer4 |
| 99077 | Scenario: CommLib loses an appliance which was discovered, while discovery is active | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for BLE appliances |
|  |  | Step 4. When shaver1 is discovered 1 time by BlueLib |
|  |  | Step 5. And the cached data expires for the following appliance: shaver1 |
|  |  | Step 6. Then the following appliance is not in the list of available appliances: shaver1 |
| 99078 | Scenario: CommLib loses an appliance which was discovered, while discovery is inactive | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for BLE appliances |
|  |  | Step 4. When shaver1 is discovered 1 time by BlueLib |
|  |  | Step 5. And stopping discovery for BLE appliances |
|  |  | Step 6. And the cached data expires for the following appliance: shaver1 |
|  |  | Step 7. Then the following appliance is not in the list of available appliances: shaver1 |
| 99079 | Scenario: CommLib loses an appliance and rediscovers it | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for BLE appliances |
|  |  | Step 4. And shaver1 is discovered 1 time by BlueLib |
|  |  | Step 5. And stopping discovery for BLE appliances |
|  |  | Step 6. And the cached data expires for the following appliance: shaver1 |
|  |  | Step 7. And the following appliance is not in the list of available appliances: shaver1 |
|  |  | Step 8. And starting discovery for BLE appliances |
|  |  | Step 9. When shaver1 is discovered 1 time by BlueLib |
|  |  | Step 10. Then the following appliance is in the list of available appliances: shaver1 |
| 99080 | Scenario: CommLib loses multiple appliances that were discovered | Step 1. Background: Given a BlueLib mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. When starting discovery for BLE appliances with model ids: MX0001 MX0002 MX0003 |
|  |  | Step 4. And mixer1 is discovered 1 time by BlueLib, matching model id MX0001 |
|  |  | Step 5. And mixer2 is discovered 1 time by BlueLib, matching model id MX0002 |
|  |  | Step 6. And mixer3 is discovered 1 time by BlueLib, matching model id MX0003 |
|  |  | Step 7. And stopping discovery for BLE appliances |
|  |  | Step 8. And the cached data expires for the following appliances: mixer1 mixer2 mixer3 |
|  |  | Step 9. Then the following appliances are not in the list of available appliances: mixer1 mixer2 mixer3 |
| 99088 | Scenario: CommLib starts discovery | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. When starting discovery for LAN appliances |
|  |  | Step 4. Then startScanning is called 1 time on SSDP |
| 99089 | Scenario: CommLib stops discovery | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. And starting discovery for LAN appliances |
|  |  | Step 4. When stopping discovery for LAN appliances |
|  |  | Step 5. Then stopScanning is called on SSDP |
| 99090 | Scenario: Commlib stops discovery when not started, silent ignore | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. When stopping discovery for LAN appliances |
|  |  | Step 4. Then stopScanning is called on SSDP |
| 99091 | Scenario: CommLib discovers a supported appliance | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliances: mixer toothbrush |
|  |  | Step 3. When starting discovery for LAN appliances |
|  |  | Step 4. And toothbrush1 is discovered by SSDP |
|  |  | Step 5. Then the following appliance is in the list of available appliances: toothbrush1 |
| 99092 | Scenario: CommLib discovers multiple supported appliances of different type | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliances: mixer shaver |
|  |  | Step 3. When starting discovery for LAN appliances |
|  |  | Step 4. And mixer1 is discovered by SSDP |
|  |  | Step 5. And shaver1 is discovered by SSDP |
|  |  | Step 6. Then the following appliances are in the list of available appliances: mixer1 shaver1 |
| 99093 | Scenario: CommLib discovers multiple supported appliances of same type | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliances: mixer shaver |
|  |  | Step 3. When starting discovery for LAN appliances |
|  |  | Step 4. And shaver1 is discovered by SSDP |
|  |  | Step 5. And shaver2 is discovered by SSDP |
|  |  | Step 6. Then the following appliances are in the list of available appliances: shaver1 shaver2 |
| 99094 | Scenario: CommLib discovers only supported appliances | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. When starting discovery for LAN appliances |
|  |  | Step 4. And mixer1 is discovered by SSDP |
|  |  | Step 5. And shaver1 is discovered by SSDP |
|  |  | Step 6. Then the following appliance is in the list of available appliances: shaver1 |
| 99095 | Scenario: CommLib discovers no supported appliances | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. When starting discovery for LAN appliances |
|  |  | Step 4. And mixer1 is discovered by SSDP |
|  |  | Step 5. Then the length of the list of available appliances is 0 |
| 99096 | Scenario: CommLib discovers appliance once | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for LAN appliances |
|  |  | Step 4. When shaver1 is discovered by SSDP |
|  |  | Step 5. And shaver1 is discovered by SSDP |
|  |  | Step 6. Then the following appliance is in the list of available appliances: shaver1 |
| 99097 | Scenario: CommLib discovers supported appliance after discovery was stopped | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: mixer |
|  |  | Step 3. And starting discovery for LAN appliances |
|  |  | Step 4. And stopping discovery for LAN appliances |
|  |  | Step 5. And stopScanning is called on SSDP |
|  |  | Step 6. When mixer1 is discovered by SSDP |
|  |  | Step 7. Then the following appliance is in the list of available appliances: mixer1 |
| 99098 | Scenario: CommLib loses an appliance which was discovered, data in cache expires | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for LAN appliances |
|  |  | Step 4. And shaver1 is discovered by SSDP |
|  |  | Step 5. And stopping discovery for LAN appliances |
|  |  | Step 6. When the cached data expires for the following appliance: shaver1 |
|  |  | Step 7. Then the following appliance is not in the list of available appliances: shaver1 |
| 99099 | Scenario: CommLib loses an appliance which was discovered, SSDP reports lost appliance | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for LAN appliances |
|  |  | Step 4. And shaver1 is discovered by SSDP |
|  |  | Step 5. When shaver1 is lost by SSDP |
|  |  | Step 6. Then the following appliance is not in the list of available appliances: shaver1 |
| 99100 | Scenario: CommLib loses an appliance which was discovered, network switch | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for LAN appliances |
|  |  | Step 4. And shaver1 is discovered by SSDP |
|  |  | Step 5. When network is switched on mobile phone |
|  |  | Step 6. Then the following appliance is not in the list of available appliances: shaver1 |
| 99101 | Scenario: CommLib loses an appliance and rediscovers it | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given application has support for appliance: shaver |
|  |  | Step 3. And starting discovery for LAN appliances |
|  |  | Step 4. And shaver1 is discovered by SSDP |
|  |  | Step 5. And shaver1 is lost by SSDP |
|  |  | Step 6. And the following appliance is not in the list of available appliances: shaver1 |
|  |  | Step 7. When shaver1 is discovered by SSDP |
|  |  | Step 8. Then the following appliance is in the list of available appliances: shaver1 |
| 99108 | Scenario: CommBle00101-get-properties-req-happy-path-connection-node | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '0' and port 'wifi' |
|  |  | Step 4. Then write occurred to mock device with id 'p' with data 'FEFF04000730007769666900' |
| 99109 | Scenario: CommBle00102-get-properties-req-happy-path-non-connection-node | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. Then write occurred to mock device with id 'p' with data 'FEFF0400083100616972636F00' |
| 99110 | Scenario: CommBle00103-get-properties-req-productid-with-multiple-digits | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '12345' and port 'airco' |
|  |  | Step 4. Then write occurred to mock device with id 'p' with data 'FEFF04000C313233343500616972636F00' |
| 99111 | Scenario: CommBle00104-get-properties-req-empty-port-identifier | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port '' |
|  |  | Step 4. Then the result is an error |
|  |  | Step 5. And no write occurred to mock device with id 'p' |
| 99112 | Scenario: CommBle00105-get-properties-req-nested-port-identifier | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'port1/subport1' |
|  |  | Step 4. Then write occurred to mock device with id 'p' with data 'FEFF0400113100706F7274312F737562706F72743100' |
| 99113 | Scenario: CommBle00106-get-properties-resp-happy-path-non-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070011007B22747572626F223A66616C73657D00' |
|  |  | Step 6. Then the result is success with data {"turbo":false} |
| 99114 | Scenario: CommBle00107-get-properties-resp-happy-path-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070004007B7D00' |
|  |  | Step 6. Then the result is success with data {} |
| 99115 | Scenario: CommBle00108-get-properties-resp-leading-garbage-even | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data '01020304FEFF070011007B22747572626F223A66616C73657D00' |
|  |  | Step 6. Then the result is success with data {"turbo":false} |
| 99116 | Scenario: CommBle00109-get-properties-resp-leading-garbage-odd | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data '010203FEFF070011007B22747572626F223A66616C73657D00' |
|  |  | Step 6. Then the result is success with data {"turbo":false} |
| 99117 | Scenario: CommBle00110-get-properties-resp-non-null-terminated-packet | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A747275657D01' |
|  |  | Step 6. Then the result is an error |
| 99118 | Scenario: CommBle00111-get-properties-resp-too-short-odd | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A7472756500' |
|  |  | Step 6. And the request times out |
|  |  | Step 7. Then the result is an error |
| 99119 | Scenario: CommBle00112-get-properties-resp-too-short-even | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A74727500' |
|  |  | Step 6. And the request times out |
|  |  | Step 7. Then the result is an error |
| 99120 | Scenario: CommBle00113-get-properties-resp-excess-data-odd | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A747275657D00BABEEF' |
|  |  | Step 6. Then the result is success with data {"turbo":true} |
| 99121 | Scenario: CommBle00114-get-properties-resp-excess-data-even | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A747275657D00BABE' |
|  |  | Step 6. Then the result is success with data {"turbo":true} |
| 99122 | Scenario: CommBle00115-get-properties-resp-multiple-packets-2-chunks | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B2274757262' |
|  |  | Step 6. And the mock device with id 'p' receives data '6F223A747275657D00' |
|  |  | Step 7. Then the result is success with data {"turbo":true} |
| 99123 | Scenario: CommBle00116-get-properties-resp-multiple-packets-3-chunks | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF' |
|  |  | Step 6. And the mock device with id 'p' receives data '070010007B2274757262' |
|  |  | Step 7. And the mock device with id 'p' receives data '6F223A747275657D00' |
|  |  | Step 8. Then the result is success with data {"turbo":true} |
| 99124 | Scenario: CommBle00117-get-properties-resp-multiple-packets-5-chunks | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FE' |
|  |  | Step 6. And the mock device with id 'p' receives data 'FF07' |
|  |  | Step 7. And the mock device with id 'p' receives data '' |
|  |  | Step 8. And the mock device with id 'p' receives data '0010007B2274757262' |
|  |  | Step 9. And the mock device with id 'p' receives data '6F223A747275657D00' |
|  |  | Step 10. Then the result is success with data {"turbo":true} |
| 99125 | Scenario: CommBle00118-get-properties-resp-server-error-no-such-port-and-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070004037B7D00' |
|  |  | Step 6. Then the result is an error 'NO\_SUCH\_PORT' with data '{}' |
| 99126 | Scenario: CommBle00119-get-properties-resp-server-error-invalid-parameter-and-non-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF0700280C7  B227265706F7274223A22436F6D707574657220736179732  06E6F222C22636F6465223A307D00' |
|  |  | Step 6. Then the result is an error 'INVALID\_PARAMETER' with data '{"report":"Computer says no","code":0}' |
| 99127 | Scenario: CommBle00120-get-properties-resp-server-error-unknown-and-non-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07000CFF7B22636F6465223A307D00' |
|  |  | Step 6. Then the result is an error 'UNKNOWN' with data '{"code":0}' |
| 99128 | Scenario: CommBle00121-get-properties-resp-server-error-out-of-memory-with-only-json-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07001C027B226572726F726D7367223A22446176652C2  073746F702E227D00' |
|  |  | Step 6. Then the result is an error 'OUT\_OF\_MEMORY' with data '{"errormsg":"Dave, stop."}' |
| 99129 | Scenario: CommBle00122-get-properties-resp-server-error-out-of-memory-with-empty-json-object | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070004027B7D00' |
|  |  | Step 6. Then the result is an error 'OUT\_OF\_MEMORY' with data '{}' |
| 99130 | Scenario: CommBle00123-get-properties-resp-max-size-payload | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07FFFF007B22626967747874223A2261616161' |
|  |  | Step 6. And the mock device with id 'p' receives data '6161616161616161616161616161616161616161616161616161  616161616161' repeated '2047' times |
|  |  | Step 7. And the mock device with id 'p' receives data '616161616161616161616161227D00' |
|  |  | Step 8. Then the result is success |
|  |  | Step 9. And the json result contains the key 'bigtxt' |
|  |  | Step 10. And the json value for key 'bigtxt' has length '65520' |
| 99131 | Scenario: CommBle00124-get-properties-resp-max-size-payload-plus-1 | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07FFFF007B22626967747874223A2261616161' |
|  |  | Step 6. And the mock device with id 'p' receives data '6161616161616161616161616161616161616161616  161616161616161616161' repeated '2047' times |
|  |  | Step 7. And the mock device with id 'p' receives data '61616161616161616161616161227D00' |
|  |  | Step 8. Then the result is an error |
| 99132 | Scenario: CommBle00125-get-properties-resp-valid-json-parsing | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07013D007B22737472696E67223A227374  72696E67222C226E756D626572696E74223A36392C226  E756D6265726672  6163223A36392E36392C226E656761746976656E756D626572  223A2D36392C226578706E756D626572223A3639652D31322C  226172726179223A5B312C322C332C342C355D2C22617272617  96D697865647479706573223A5B22737472696E67222C312C322  E322C2D322C34452B342C66616C73652C747275652C6E756C6C  5D2C22626F6F6C74727565223A747275652C22626F6F6C66616C7  365223A66616C73652C226E6F7468696E676E657373223A6E756  C6C2C22657363617065737472696E67223A225C225C5C5C2F5C62  5C665C6E5C725C745C7531323334222C22223A2276616C75652D  666F722D656D7074796B6579222C226B65792D666F722D656D707  4792D737472696E67223A22227D00' |
|  |  | Step 6. Then the result is success with data { "string":"string", "numberint":69, "numberfrac":69.69, "negativenumber":-69, "expnumber":69e-12, "array":[1,2,3,4,5], "arraymixedtypes":["string",1,2.2,-2,4E+4,false,true,null], "booltrue":true, "boolfalse":false, "nothingness":null, "escapestring":"\"\\\/\b\f\n\r\t\u1234", "":"value-for-emptykey", "key-for-empty-string":"" } |
| 99133 | Scenario: CommBle00126-get-properties-resp-valid-json-parsing-nested-objects | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07006A007B226B6579666F72737472696E67223A22736  F6D65737472696E67222C226E6573746C6576656C31223A7  B226E6573746C6576656C32223A7B226E6573746C6576656C  33223A7B22736F6D65626F6F6C223A66616C73657D7D2C227  36F6D65696E74223A31317D7D00' |
|  |  | Step 6. Then the result is success with data { "keyforstring":"somestring", "nestlevel1":{ "nestlevel2":{ "nestlevel3":{"somebool":false} }, "someint":11 } } |
| 99134 | Scenario: CommBle00127-get-properties-resp-no-json | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF0700020300' |
|  |  | Step 6. Then the result is an error 'NO\_SUCH\_PORT' without data |
| 99135 | Scenario: CommBle00128-get-properties-resp-invalid-json | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A66616C736500' |
|  |  | Step 6. Then the result is an error |
| 99136 | Scenario Outline: CommBle00129-get-properties-req-same-port-on-different-productids-is-allowed | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '@productid ' and port 'device' |
|  |  | Step 4. Then write occurred to mock device with id 'p' with data @bytedata |
| 99139 | Scenario: CommBle00001-put-properties-req-happy-path | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. Then write occurred to mock device with id 'p' with data 'FEFF0300183100616972636F007B22747572626F223  A66616C73657D00' |
| 99140 | Scenario: CommBle00002-put-properties-req-productid-with-multiple-digits | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '12345' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. Then write occurred to mock device with id 'p' with data 'FEFF03001C313233343500616972636F007B22747572626F2  23A66616C73657D00' |
| 99141 | Scenario: CommBle00003-put-properties-req-empty-port-identifier | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port '' with data {"turbo":false} |
|  |  | Step 4. Then the result is an error |
|  |  | Step 5. And no write occurred to mock device with id 'p' |
| 99142 | Scenario: CommBle00004-put-properties-req-null-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with null data |
|  |  | Step 4. Then the result is an error 'INVALID\_PARAMETER' with any data |
|  |  | Step 5. And no write occurred to mock device with id 'p' |
| 99143 | Scenario: CommBle00004a-put-properties-req-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with empty data |
|  |  | Step 4. Then the result is an error 'NO\_REQUEST\_DATA' with any data |
|  |  | Step 5. And no write occurred to mock device with id 'p' |
| 99144 | Scenario: CommBle00005-put-properties-req-empty-port-identifier-and-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port '' with empty data |
|  |  | Step 4. Then the result is an error 'NO\_SUCH\_PORT' with any data |
|  |  | Step 5. And no write occurred to mock device with id 'p' |
| 99145 | Scenario Outline: CommBle00006-put-properties-req-nested-port-identifier | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'port1/subport1' with data '@payload ' |
|  |  | Step 4. Then write occurred to mock device with id 'p' with packet 'FEFF03[0-9A-F]{4}3100706F7274312F737562706F72743100([0-9A-F]\*)00' and payload equivalent to '@payload ' |
| 99146 | Scenario: CommBle00007-put-properties-req-valid-json-types | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data { "string":"string", "numberint":69, "numberfrac":69.69, "negativenumber":-69, "expnumber":22e-6, "array":[1,2,3,4,5], "arraymixedtypes":["string",1,2.2,-2,4E+4,false,true,null], "booltrue":true, "boolfalse":false, "nothingness":null, "escapestring":"\"\\\/\b\f\n\r\t\u1234", "":"value-for-emptykey", "key-for-empty-string":"" } |
|  |  | Step 4. Then write occurred to mock device with id 'p' with packet 'FEFF03[0-9A-F]{4}3100616972636F00([0-9A-F]\*)00' and payload equivalent to { "string":"string", "numberint":69, "numberfrac":69.69, "negativenumber":-69, "expnumber":22e-6, "array":[1,2,3,4,5], "arraymixedtypes":["string",1,2.2,-2,4E+4,false,true,null], "booltrue":true, "boolfalse":false, "nothingness":null, "escapestring":"\"\\\/\b\f\n\r\t\u1234", "":"value-for-emptykey", "key-for-empty-string":"" } |
| 99147 | Scenario: CommBle00008-put-properties-req-valid-json-constructs | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {  "nestlevel1":{ "nestlevel2":{ "nestlevel3":{"somebool":false} } } } |
|  |  | Step 4. Then write occurred to mock device with id 'p' with data 'FEFF0300483100616972636F007B226E6573746C65  76656C31223A7B226E6573746C6576656C32223A7B2  26E6573746C6576656C33223A7B22736F6D65626F6F  6C223A66616C73657D7D7D7D00' |
| 99148 | Scenario: CommBle00009-put-properties-resp-happy-path-non-empty-json | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070011007B22747572626F223A66616C73657D00' |
|  |  | Step 6. Then the result is success with data {"turbo":false} |
| 99149 | Scenario: CommBle00010-put-properties-resp-happy-path-empty-json | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070004007B7D00' |
|  |  | Step 6. Then the result is success with data {} |
| 99150 | Scenario: CommBle00011-put-properties-resp-not-allowed-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF0700020000' |
|  |  | Step 6. Then the result is success without data |
| 99151 | Scenario: CommBle00012-put-properties-resp-error-no-such-port-and-not-allowed-empty-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF0700020300' |
|  |  | Step 6. Then the result is an error |
| 99152 | Scenario: CommBle00012-put-properties-resp-error-no-such-property-and-json-object | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070021067B226572726F726D7367223A224  E6F20737563682070726F7065727479227D00' |
|  |  | Step 6. Then the result is an error 'NO\_SUCH\_PROPERTY' with data '{"errormsg":"No such property"}' |
| 99153 | Scenario: CommBle00013-put-properties-resp-error-unknown-and-json-object | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070036FF7B226572726F726D7367223A2255  6E6B6E6F776E206572726F72222C226F74686572  6B6579223A226F7468657276616C7565227D00' |
|  |  | Step 6. Then the result is an error 'UNKNOWN' with data '{"errormsg":"Unknown error","otherkey":"othervalue"}' |
| 99154 | Scenario: CommBle00014-put-properties-resp-error-out-of-memory-with-empty-json-object | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070004027B7D00' |
|  |  | Step 6. Then the result is an error 'OUT\_OF\_MEMORY' with data '{}' |
| 99155 | Scenario: CommBle00015-put-properties-resp-error-out-of-memory-with-non-json-but-still-ascii-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07001B024920646F206E6F74207265  6D656D62657220616E796D6F726500' |
|  |  | Step 6. Then the result is an error |
| 99156 | Scenario: CommBle00016-put-properties-resp-error-out-of-memory-with-non-json-and-non-ascii-data | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07000702000102030400' |
|  |  | Step 6. Then the result is an error |
| 99157 | Scenario: CommBle00017-put-properties-resp-multiple-packets-7-chunks | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FE' |
|  |  | Step 6. And the mock device with id 'p' receives data 'FF07' |
|  |  | Step 7. And the mock device with id 'p' receives data '0011007B2274' |
|  |  | Step 8. And the mock device with id 'p' receives data '7572626F223A66616C' |
|  |  | Step 9. And the mock device with id 'p' receives data '' |
|  |  | Step 10. And the mock device with id 'p' receives data '73657D' |
|  |  | Step 11. And the mock device with id 'p' receives data '00' |
|  |  | Step 12. Then the result is success with data {"turbo":false} |
| 99158 | Scenario: CommBle00018-put-properties-resp-multiple-packets-3-chunks | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070011007B22' |
|  |  | Step 6. And the mock device with id 'p' receives data '747572626F223A66' |
|  |  | Step 7. And the mock device with id 'p' receives data '616C73657D00' |
|  |  | Step 8. Then the result is success with data {"turbo":false} |
| 99159 | Scenario: CommBle00019-put-properties-resp-multiple-packets-2-chunks | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070011007B22747572' |
|  |  | Step 6. And the mock device with id 'p' receives data '626F223A66616C73657D00' |
|  |  | Step 7. Then the result is success with data {"turbo":false} |
| 99160 | Scenario: CommBle00020-put-properties-resp-non-null-terminated-packet | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A747275657D01' |
|  |  | Step 6. Then the result is an error |
| 99161 | Scenario: CommBle00021-put-properties-resp-too-short-odd | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A7472756500' |
|  |  | Step 6. And the request times out |
|  |  | Step 7. Then the result is an error |
| 99162 | Scenario: CommBle00022-put-properties-resp-too-short-even | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":false} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A74727500' |
|  |  | Step 6. And the request times out |
|  |  | Step 7. Then the result is an error |
| 99163 | Scenario: CommBle00023-put-properties-resp-excess-data-odd | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":true} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A747275657D00BABEEF' |
|  |  | Step 6. Then the result is success with data {"turbo":true} |
| 99164 | Scenario: CommBle00024-put-properties-resp-excess-data-even | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":true} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF070010007B22747572626F223A747275657D00BABE' |
|  |  | Step 6. Then the result is success with data {"turbo":true} |
| 99165 | Scenario: CommBle00025-put-properties-req-max-size | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with any data with size '65526' |
|  |  | Step 4. Then write occurred to mock device with id 'p' with pattern 'FEFF03FFFF3100616972636F00[0-9A-F]{131052}00' |
| 99166 | Scenario: CommBle00026-put-properties-req-max-size-plus-one | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with any data with size '65527' |
|  |  | Step 4. Then the result is an error |
|  |  | Step 5. And no write occurred to mock device with id 'p' |
| 99167 | Scenario: CommBle00027-put-properties-resp-max-size-payload | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":true} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07FFFF007B22626967747874223A2261616161' |
|  |  | Step 6. And the mock device with id 'p' receives data '616161616161616161616161616161616161616161  6161616161616161616161' repeated '2047' times |
|  |  | Step 7. And the mock device with id 'p' receives data '616161616161616161616161227D00' |
|  |  | Step 8. Then the result is success |
|  |  | Step 9. And the json result contains the key 'bigtxt' |
|  |  | Step 10. And the json value for key 'bigtxt' has length '65520' |
| 99168 | Scenario: CommBle00028-put-properties-resp-max-size-payload-plus-1 | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'airco' with data {"turbo":true} |
|  |  | Step 4. And write occurred to mock device with id 'p' with any data |
|  |  | Step 5. And the mock device with id 'p' receives data 'FEFF07FFFF007B22626967747874223A2261616161' |
|  |  | Step 6. And the mock device with id 'p' receives data '6161616161616161616161616161616161616161616  161616161616161616161' repeated '2047' times |
|  |  | Step 7. And the mock device with id 'p' receives data '61616161616161616161616161227D00' |
|  |  | Step 8. Then the result is an error |
| 99180 | Scenario: Uses HTTPS by default | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given an appliance with id 'x' |
|  |  | Step 3. When communicating with appliance 'x' |
|  |  | Step 4. Then communication uses HTTPS |
| 99181 | Scenario: Can be configured to use HTTP | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given an appliance with id 'x' that has been configured to use HTTP |
|  |  | Step 3. When communicating with appliance 'x' |
|  |  | Step 4. Then communication uses HTTP |
| 99182 | Scenario: Can be configured to not use HTTP | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given an appliance with id 'x' that has been configured to not use HTTP |
|  |  | Step 3. When communicating with appliance 'x' |
|  |  | Step 4. Then communication uses HTTPS |
| 99183 | Scenario: Discovery does not reconfigure to use HTTP | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given an appliance with id 'x' |
|  |  | Step 3. And the appliance with id 'x' has been discovered as requiring HTTP |
|  |  | Step 4. When communicating with appliance 'x' |
|  |  | Step 5. Then communication uses HTTPS |
| 99184 | Scenario: Discovery does not reconfigure to use HTTPS | Step 1. Background: Given an SSDP mock |
|  |  | Step 2. Given an appliance with id 'x' that has been configured to use HTTP |
|  |  | Step 3. And the appliance with id 'x' has been discovered as requiring HTTPS |
|  |  | Step 4. When communicating with appliance 'x' |
|  |  | Step 5. Then communication uses HTTP |
| 99192 | Scenario: CommBle00201-strategy-queue-depth1-happy-flow-1 | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '0' and port 'wifi' |
|  |  | Step 4. And doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 5. And write occurred to mock device with id 'p' with data 'FEFF04000730007769666900' |
|  |  | Step 6. And the mock device with id 'p' receives data 'FEFF07001E007B2270617373776F7264223A22667  26970706574696566726170227D00' |
|  |  | Step 7. Then the result is success with data {"password":"frippetiefrap"} |
|  |  | Step 8. And write occurred to mock device with id 'p' with data 'FEFF0400083100616972636F00' |
| 99193 | Scenario: CommBle00202-strategy-queue-depth1-happy-flow-2 | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '0' and port 'wifi' |
|  |  | Step 4. And write occurred to mock device with id 'p' with data 'FEFF04000730007769666900' |
|  |  | Step 5. And doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 6. And the mock device with id 'p' receives data 'FEFF07001E007B2270617373776F  7264223A2266726970706574696566726170227D00' |
|  |  | Step 7. Then the result is success with data {"password":"frippetiefrap"} |
|  |  | Step 8. And write occurred to mock device with id 'p' with data 'FEFF0400083100616972636F00' |
| 99194 | Scenario: CommBle00203-strategy-queue-depth1-timeout-of-first-req | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '0' and port 'wifi' |
|  |  | Step 4. And doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 5. And write occurred to mock device with id 'p' with data 'FEFF04000730007769666900' |
|  |  | Step 6. And the request times out |
|  |  | Step 7. Then the result is an error |
|  |  | Step 8. And write occurred to mock device with id 'p' with data 'FEFF0400083100616972636F00' |
| 99195 | Scenario: CommBle00204-strategy-queue-depth1-unavailability | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '0' and port 'wifi' |
|  |  | Step 4. And doing a get-properties for productid '1' and port 'airco' |
|  |  | Step 5. And write occurred to mock device with id 'p' with data 'FEFF04000730007769666900' |
|  |  | Step 6. And total bytes written to id 'p' is '12' |
|  |  | Step 7. And the BLE communication strategy becomes unavailable |
|  |  | Step 8. And the request times out |
|  |  | Step 9. Then the result for request nr '0' is an error 'TIMED\_OUT' |
|  |  | Step 10. And total bytes written to id 'p' is '12' |
|  |  | Step 11. Then the result for request nr '1' is an error 'NOT\_AVAILABLE' |
| 99196 | Scenario: CommBle00205-strategy-queue-depth2-same-port | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '1' and port 'nr01' |
|  |  | Step 4. And doing a get-properties for productid '1' and port 'nr01' |
|  |  | Step 5. And doing a get-properties for productid '1' and port 'nr01' |
|  |  | Step 6. And write occurred to mock device with id 'p' with data 'FEFF04000731006E72303100' |
|  |  | Step 7. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223031227D00' |
|  |  | Step 8. Then the result is success with data {"portnr":"01"} |
|  |  | Step 9. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303100' |
|  |  | Step 10. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223031227D00' |
|  |  | Step 11. Then the result is success with data {"portnr":"01"} |
|  |  | Step 12. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303100' |
|  |  | Step 13. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223031227D00' |
|  |  | Step 14. Then the result is success with data {"portnr":"01"} |
| 99197 | Scenario: CommBle00206-strategy-queue-depth20-unique-ports | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a get-properties for productid '0' and port 'wifi' |
|  |  | Step 4. And doing a get-properties for productid '1' and port 'nr02' |
|  |  | Step 5. And doing a get-properties for productid '1' and port 'nr03' |
|  |  | Step 6. And doing a get-properties for productid '1' and port 'nr04' |
|  |  | Step 7. And doing a get-properties for productid '1' and port 'nr05' |
|  |  | Step 8. And doing a get-properties for productid '1' and port 'nr06' |
|  |  | Step 9. And doing a get-properties for productid '1' and port 'nr07' |
|  |  | Step 10. And doing a get-properties for productid '1' and port 'nr08' |
|  |  | Step 11. And doing a get-properties for productid '1' and port 'nr09' |
|  |  | Step 12. And doing a get-properties for productid '1' and port 'nr10' |
|  |  | Step 13. And doing a get-properties for productid '1' and port 'nr11' |
|  |  | Step 14. And doing a get-properties for productid '1' and port 'nr12' |
|  |  | Step 15. And doing a get-properties for productid '1' and port 'nr13' |
|  |  | Step 16. And doing a get-properties for productid '1' and port 'nr14' |
|  |  | Step 17. And doing a get-properties for productid '1' and port 'nr15' |
|  |  | Step 18. And doing a get-properties for productid '1' and port 'nr16' |
|  |  | Step 19. And doing a get-properties for productid '1' and port 'nr17' |
|  |  | Step 20. And doing a get-properties for productid '1' and port 'nr18' |
|  |  | Step 21. And doing a get-properties for productid '1' and port 'nr19' |
|  |  | Step 22. And doing a get-properties for productid '1' and port 'nr20' |
|  |  | Step 23. And doing a get-properties for productid '1' and port 'nr21' |
|  |  | Step 24. And write occurred to mock device with id 'p' with data 'FEFF04000730007769666900' |
|  |  | Step 25. And the mock device with id 'p' receives data 'FEFF07001E007B2270617373776F7264223A2266726970706  574696566726170227D00' |
|  |  | Step 26. Then the result is success with data {"password":"frippetiefrap"} |
|  |  | Step 27. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303200' |
|  |  | Step 28. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223032227D00' |
|  |  | Step 29. Then the result is success with data {"portnr":"02"} |
|  |  | Step 30. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303300' |
|  |  | Step 31. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223033227D00' |
|  |  | Step 32. Then the result is success with data {"portnr":"03"} |
|  |  | Step 33. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303400' |
|  |  | Step 34. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223034227D00' |
|  |  | Step 35. Then the result is success with data {"portnr":"04"} |
|  |  | Step 36. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303500' |
|  |  | Step 37. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223035227D00' |
|  |  | Step 38. Then the result is success with data {"portnr":"05"} |
|  |  | Step 39. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303600' |
|  |  | Step 40. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223036227D00' |
|  |  | Step 41. Then the result is success with data {"portnr":"06"} |
|  |  | Step 42. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303700' |
|  |  | Step 43. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223037227D00' |
|  |  | Step 44. Then the result is success with data {"portnr":"07"} |
|  |  | Step 45. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303800' |
|  |  | Step 46. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223038227D00' |
|  |  | Step 47. Then the result is success with data {"portnr":"08"} |
|  |  | Step 48. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303900' |
|  |  | Step 49. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223039227D00' |
|  |  | Step 50. Then the result is success with data {"portnr":"09"} |
|  |  | Step 51. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313000' |
|  |  | Step 52. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223130227D00' |
|  |  | Step 53. Then the result is success with data {"portnr":"10"} |
|  |  | Step 54. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313100' |
|  |  | Step 55. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223131227D00' |
|  |  | Step 56. Then the result is success with data {"portnr":"11"} |
|  |  | Step 57. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313200' |
|  |  | Step 58. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223132227D00' |
|  |  | Step 59. Then the result is success with data {"portnr":"12"} |
|  |  | Step 60. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313300' |
|  |  | Step 61. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223133227D00' |
|  |  | Step 62. Then the result is success with data {"portnr":"13"} |
|  |  | Step 63. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313400' |
|  |  | Step 64. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223134227D00' |
|  |  | Step 65. Then the result is success with data {"portnr":"14"} |
|  |  | Step 66. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313500' |
|  |  | Step 67. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223135227D00' |
|  |  | Step 68. Then the result is success with data {"portnr":"15"} |
|  |  | Step 69. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313600' |
|  |  | Step 70. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223136227D00' |
|  |  | Step 71. Then the result is success with data {"portnr":"16"} |
|  |  | Step 72. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313700' |
|  |  | Step 73. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223137227D00' |
|  |  | Step 74. Then the result is success with data {"portnr":"17"} |
|  |  | Step 75. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313800' |
|  |  | Step 76. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223138227D00' |
|  |  | Step 77. Then the result is success with data {"portnr":"18"} |
|  |  | Step 78. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72313900' |
|  |  | Step 79. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223139227D00' |
|  |  | Step 80. Then the result is success with data {"portnr":"19"} |
|  |  | Step 81. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72323000' |
|  |  | Step 82. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223230227D00' |
|  |  | Step 83. Then the result is success with data {"portnr":"20"} |
|  |  | Step 84. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72323100' |
|  |  | Step 85. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223231227D00' |
|  |  | Step 86. Then the result is success with data {"portnr":"21"} |
| 99198 | Scenario: CommBle00207-strategy-queue-mixed-operations | Step 1. Background: Given a mock device is found with id 'p' |
|  |  | Step 2. Background: And the BLE communication strategy is initialized with id 'p' |
|  |  | Step 3. When doing a put-properties for productid '1' and port 'nr01' with data {"portnr":"01"} |
|  |  | Step 4. And doing a get-properties for productid '0' and port 'wifi' |
|  |  | Step 5. And doing a get-properties for productid '1' and port 'nr03' |
|  |  | Step 6. And doing a put-properties for productid '1' and port 'nr04' with data {"portnr":"04"} |
|  |  | Step 7. And doing a put-properties for productid '1' and port 'nr05' with data {"portnr":"05"} |
|  |  | Step 8. And doing a get-properties for productid '1' and port 'nr06' |
|  |  | Step 9. And doing a get-properties for productid '1' and port 'nr07' |
|  |  | Step 10. And write occurred to mock device with id 'p' with data 'FEFF03001731006E723031007B22706F72746E7222  3A223031227D00' |
|  |  | Step 11. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223031227D00' |
|  |  | Step 12. Then the result is success with data {"portnr":"01"} |
|  |  | Step 13. When write occurred to mock device with id 'p' with data 'FEFF04000730007769666900' |
|  |  | Step 14. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223032227D00' |
|  |  | Step 15. Then the result is success with data {"portnr":"02"} |
|  |  | Step 16. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303300' |
|  |  | Step 17. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223033227D00' |
|  |  | Step 18. Then the result is success with data {"portnr":"03"} |
|  |  | Step 19. When write occurred to mock device with id 'p' with data 'FEFF03001731006E723034007B22706F72746E7222  3A223034227D00' |
|  |  | Step 20. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223034227D00' |
|  |  | Step 21. Then the result is success with data {"portnr":"04"} |
|  |  | Step 22. When write occurred to mock device with id 'p' with data 'FEFF03001731006E723035007B22706F72746E72223A223035227D00' |
|  |  | Step 23. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223035227D00' |
|  |  | Step 24. Then the result is success with data {"portnr":"05"} |
|  |  | Step 25. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303600' |
|  |  | Step 26. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223036227D00' |
|  |  | Step 27. Then the result is success with data {"portnr":"06"} |
|  |  | Step 28. When write occurred to mock device with id 'p' with data 'FEFF04000731006E72303700' |
|  |  | Step 29. And the mock device with id 'p' receives data 'FEFF070011007B22706F72746E72223A223037227D00' |
|  |  | Step 30. Then the result is success with data {"portnr":"07"} |
| 99204 | Scenario: Application subscribes to notifications on a port | Step 1. Background: Given there is an appliance |
|  |  | Step 2. Background: And a port has been added with name 'x' and productID '0' |
|  |  | Step 3. When application subscribes to port 'x' with productID '0' |
|  |  | Step 4. Then subscription is successful |
| 99205 | Scenario: Application unsubscribes from notifications on a port | Step 1. Background: Given there is an appliance |
|  |  | Step 2. Background: And a port has been added with name 'x' and productID '0' |
|  |  | Step 3. Given application subscribes to port 'x' with productID '0' |
|  |  | Step 4. When application unsubscribes from port 'x' with productID '0' |
|  |  | Step 5. Then cancelling subscription is successful |
| 99206 | Scenario: Application receives a notification from a port | Step 1. Background: Given there is an appliance |
|  |  | Step 2. Background: And a port has been added with name 'x' and productID '0' |
|  |  | Step 3. Given application subscribes to port 'x' with productID '0' |
|  |  | Step 4. When port 'x' with productID '0' sends '1' notification |
|  |  | Step 5. Then application receives a notification from port 'x' with productID '0' |
| 99207 | Scenario: Application receives a notification from a port with an expired key | Step 1. Background: Given there is an appliance |
|  |  | Step 2. Background: And a port has been added with name 'x' and productID '0' |
|  |  | Step 3. Given application subscribes to port 'x' with productID '0' with an expired key |
|  |  | Step 4. When port 'x' with productID '0' sends '2' notification |
|  |  | Step 5. Then application receives a notification from port 'x' with productID '0' |

# Test Instructions

Not applicable. The test instructions are part of the steps of the test scenarios themselves.

# Revision History

| **Version** | **Date** | **Author** | **Description of Change** | **Reason for Change** |
| --- | --- | --- | --- | --- |
| 0.1 | 2017-May-01 | Gerard Arts | Initial draft | Creation |
| 1.0 | 2017-May-03 | Gerard Arts | Approver updated | Ready for Review |
| 1.1 | 2017-May-05 | Gerard Arts | Review remarks implemented | Review remarks implemented |
| 1.2 | 2017-May-10 | Bas Flaton | Removed test definitions that were meant for next release | Improvement |
| 2.0 | 2017-May-10 | Gerard Arts | Version changed | For approval |
| 2.1 | 2017-July-10 | Gerard Arts | Test cases added for PI 17.3. New test cases are related to HTTPS support . | Ready for Review |
| 2.2 | 2017-July-13 | Gerard Arts | Review remarks implemented | Review remarks implemented |
| 3.0 | 2017-July-18 | Gerard Arts | Version changed | For approval |
| 3.1 | 2017-Nov-20 | Nicky van Meir | Updated manual and automated test cases | Ready for Review |

# Approval

| **Name** | **Role / Function** | **Date** (YYYY-MON-DD) | **Signature** |
| --- | --- | --- | --- |
| Thijs Winter | Domain lead Connectivity |  |  |