SOFTWARE REQUIREMENTS

SPECIFICATION

For

**CERES - Alljoyn Test Tool**

Version 0.1

Prepared by Tuan Ngo

Duy Phan

[**Revision History**](#h.gjdgxs)

[**Introduction**](#h.30j0zll)

[1. Purpose](#h.1fob9te)

[2. Scope](#h.3znysh7)

[3. Abbreviations](#h.2et92p0)

[4. References](#h.tyjcwt)

[5. Overview](#h.3dy6vkm)

[**Overall Description**](#h.4d34og8)

[1. Product Perspective](#h.2s8eyo1)

[*1.1.* *System interfaces*](#h.17dp8vu)

[*1.2.* *User Interfaces*](#h.3rdcrjn)

[*1.3.* *Hardware interfaces*](#h.26in1rg)

[*1.4.* *Software interfaces*](#h.lnxbz9)

[*1.5.* *Communications interfaces*](#h.35nkun2)

[*1.6.* *Memory*](#h.1ksv4uv)

[*1.7.* *Operations*](#h.44sinio)

[*1.8.* *Site adaptation requirements*](#h.2jxsxqh)

[2. Product Functions](#h.z337ya)

[3. User Characteristic](#h.3j2qqm3)

[4. General Constraints](#h.1y810tw)

[5. Assumption and Dependencies](#h.4i7ojhp)

[**Specific Requirements**](#h.1ci93xb)

[1. External interfaces](#h.3whwml4)

[2. Functions](#h.2bn6wsx)

[3. Performance requirements](#h.qsh70q)

[4. Logical database requirements](#h.3as4poj)

[5. Design constraints](#h.1pxezwc)

[*5.1.* *Standards compliance*](#h.49x2ik5)

[6. Software system attributes](#h.2p2csry)

[*6.1.* *Reliability*](#h.147n2zr)

[*6.2.* *Availability*](#h.3o7alnk)

[*6.3.* *Security*](#h.23ckvvd)

[*6.4.* *Maintainability*](#h.ihv636)

[*6.5.* *Portability*](#h.32hioqz)

[**Appendixes**](#h.1hmsyys)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Authors | Description |
| 0.1 | Oct 5, 2015 | Duy Phan, Tuan Ngo | The initial document |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

Purpose of this project is to create a software to verify functionalities Alljoyn on CERES development board.

Audience: manager, developer, tester related in CERES development.

## Scope

[This subsection should

a) Identify the software product(s) to be produced by name (e.g., Host DBMS, Report Generator, etc.);

b) Explain what the software product(s) will, and, if necessary, will not do;

c) Describe the application of the software being specified, including relevant benefits, objectives, and goals;

d) Be consistent with similar statements in higher-level specifications (e.g., the system requirements specification), if they exist.]

This software ...

## Abbreviations

|  |  |
| --- | --- |
| Abbreviations | |
| CLI | Command Line Interface |
| OS | Operating System |
|  |  |
|  |  |
|  |  |
|  |  |

## References

* Develop Alljoyn application guide: <https://allseenalliance.org/developers/develop>

## Overview

[This subsection should

a) Describe what the rest of the SRS contains;

b) Explain how the SRS is organized.]

# Overall Description

## Product Perspective

[This subsection of the SRS should put the product into perspective with other related products. If the product is independent and totally self-contained, it should be so stated here. If the SRS defines a product that is a component of a larger system, as frequently occurs, then this subsection should relate the requirements of that larger system to functionality of the software and should identify interfaces between that system and the software.

A block diagram showing the major components of the larger system, interconnections, and external inter-faces can be helpful.

This subsection should also describe how the software operates inside various constraints. For example, these constraints could include: System interfaces; User interfaces; Hardware interfaces; Software interfaces; Communications interfaces; Memory; Operations; Site adaptation requirements. ]

This application will be run on Linux OS independent to CERES OS. It communicates with CERES via bus interface is created by Alljoyn framework. Main function of this application is verify onboarding services, about services via test cases.

### System interfaces

[This should list each system interface and identify the functionality of the software to accomplish the system requirement and the interface description to match the system.]

~~Internet socket (STREAM, DATAGRAM socket)~~

~~Unix domain socket~~

### User Interfaces

[This should specify the following:

a) The logical characteristics of each interface between the software product and its users. This includes those configuration characteristics (e.g., required screen formats, page or window layouts, content of any reports or menus, or availability of programmable function keys) necessary to accomplish the software requirements.

b) All the aspects of optimizing the interface with the person who must use the system. This may simply comprise a list of do’s and don’ts on how the system will appear to the user. One example may be a

requirement for the option of long or short error messages. Like all others, these requirements should be verifiable, e.g., “a clerk typist grade 4 can do function X in Z min after 1 h of training” rather than “a typist can do function X.” (This may also be specified in the Software System Attributes under a section titled Ease of Use.)]

This software does not have user interfaces. It would be interactive via command line interface (CLI) on Linux OS. It would receive input via configuration input file.

Configuration input file includes: configuration, test cases, necessary settings uses to communication with Alljoyn Router, ...

### Hardware interfaces

[This should specify the logical characteristics of each interface between the software product and the hardware components of the system. This includes configuration characteristics (number of ports, instruction sets, etc.). It also covers such matters as what devices are to be supported, how they are to be supported, and protocols. For example, terminal support may specify full-screen support as opposed to line-by-line support.]

Network communication over NIC (wire and wireless)

### Software interfaces

[This should specify the use of other required software products (e.g., a data management system, an operating system, or a mathematical package), and interfaces with other application systems (e.g., the linkage between an accounts receivable system and a general ledger system). For each required software product, the following should be provided:

- Name;

- Mnemonic;

- Specification number;

- Version number;

- Source.

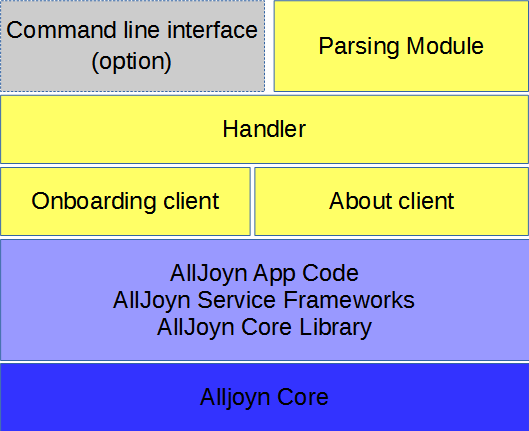
For each interface, the following should be provided:

- Discussion of the purpose of the interfacing software as related to this software product.

- Definition of the interface in terms of message content and format. It is not necessary to detail any well-documented interface, but a reference to the document defining the interface is required.]

Alljoyn framework software is required for develop this application:

* <https://allseenalliance.org/releases/alljoyn/15.04/alljoyn-15.04.00b-src.tar.gz>
* <https://allseenalliance.org/releases/alljoyn/15.04/alljoyn-services-15.04.00-src.tar.gz>



### Communications interfaces

This application would use wireless interface communicate CERES.

Onboarding service runs only on Wi-Fi.

### Memory

There is no specific requirement memory for this application.

### Operations

[This should specify the normal and special operations required by the user such as

a) The various modes of operations in the user organization (e.g., user-initiated operations);

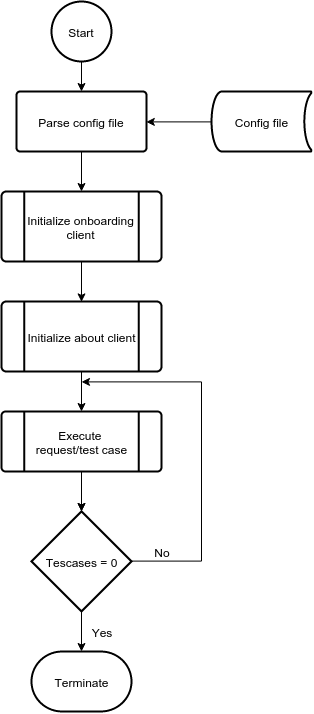
b) Periods of interactive operations and periods of unattended operations;

c) Data processing support functions;

d) Backup and recovery operations.

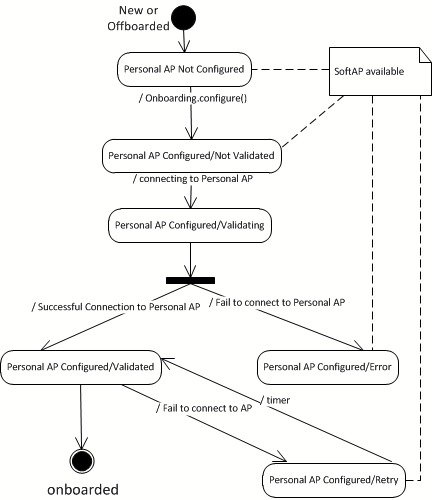
NOTE: This is sometimes specified as part of the User Interfaces section.]

Software operation flow is as below

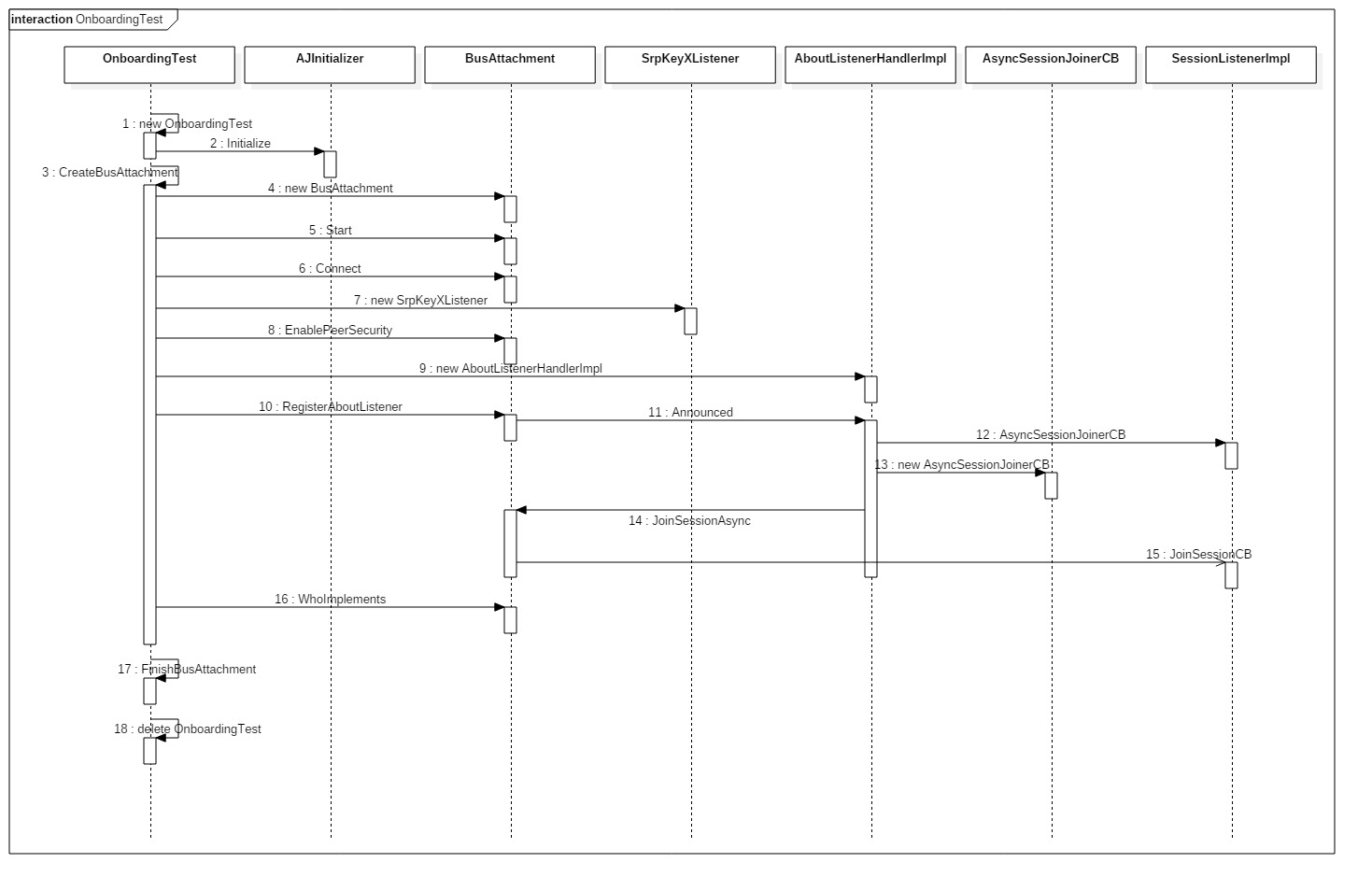


The Onboarding service provides a common and simple way for new device to be brought onto the Wi-Fi network. There’re 2 roles:

* Onboardee: the device (eg: CERES) that is unconfigured and needs to be brought onto the Wi-Fi network.
* Onboarder: the device is configuring the Onboardee device, typically a mobile application or PC.

An Onboardee application is responsible for starting up as an Access Point. An application that provides the Onboarder side of the service is responsible for using the Onboarding interface to transmit Access Point credentials that the Onboardee application should connect to.

Onboarding test tool sequence diagram



### Site adaptation requirements

## Product Functions

[This subsection of the SRS should provide a summary of the major functions that the software will perform.

For example, an SRS for an accounting program may use this part to address customer account maintenance, customer statement, and invoice preparation without mentioning the vast amount of detail that each of those functions requires.

Sometimes the function summary that is necessary for this part can be taken directly from the section of the higher-level specification (if one exists) that allocates particular functions to the software product. Note that for the sake of clarity

a) The functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document for the first time.

b) Textual or graphical methods can be used to show the different functions and their relationships.

Such a diagram is not intended to show a design of a product, but simply shows the logical relationships among variables.]

Testing CERES alljoy service run on CERES:

Alljoyn about anouncement

Alljoyn onboarding

## User Characteristic

Developer who want to test their alljoyn service function run on CERES. Tester who will test product features

[This subsection of the SRS should describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise. It should not be used to state specific requirements, but rather should provide the reasons why certain specific requirements are later specified in Section “*Specific Requirements*” of the SRS.]

## General Constraints

[This subsection of the SRS should provide a general description of any other items that will limit the developer’s options. These include

a) Regulatory policies;

b) Hardware limitations (e.g., signal timing requirements);

c) Interfaces to other applications;

d) Parallel operation;

e) Audit functions;

f) Control functions;

g) Higher-order language requirements;

h) Signal handshake protocols (e.g., XON-XOFF, ACK-NACK);

i) Reliability requirements;

j) Criticality of the application;

k) Safety and security considerations. ]

## Assumption and Dependencies

The application depend on an alljoyn-daemon for running normally on host personal computer.

[This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS.

These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption may be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.]

# Specific Requirements

[This section of the SRS should contain all of the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. Throughout this section, every stated requirement should be externally perceivable by users, operators, or other external systems. These requirements should include at a minimum a description of every input (stimulus) into the system, every output (response) from the system, and all functions performed by the system in response to an input or in support of an output. As this is often the largest and most important part of the SRS, the following principles apply:

a) Specific requirements should be stated in conformance with all the characteristics described in section “*User Characteristic*.”

b) Specific requirements should be cross-referenced to earlier documents that relate.

c) All requirements should be uniquely identifiable.

d) Careful attention should be given to organizing the requirements to maximize readability.

Before examining specific ways of organizing the requirements it is helpful to understand the various items that comprise requirements as described below]

## External interfaces

[This should be a detailed description of all inputs into and outputs from the software system. It should complement the interface descriptions in *“Overall Description”* and should not repeat information there.

It should include both content and format as follows:

a) Name of item;

b) Description of purpose;

c) Source of input or destination of output;

d) Valid range, accuracy, and/or tolerance;

e) Units of measure;

f) Timing;

g) Relationships to other inputs/outputs;

h) Screen formats/organization;

i) Window formats/organization;

j) Data formats;

k) Command formats;

l) End messages.]

## Functions

## Alljoyn onboarding client

* + - 1. **Get scan-wifi info**
      2. **Configure target wifi info**
      3. **Request connect to target wifi**
      4. **Verify onboardee device fully onboarded**
      5. **………….**
    1. **Alljoyn about announcement client**
       1. **Do bus attachment which specific interface**
       2. **Get service about data**
       3. **List all signal, method which service is providing**
       4. **call signal/method**
       5. **Get response data**
       6. **……………...**

[Functional requirements should define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs. These are generally listed as “shall” statements starting with “The system shall…”

These include

a) Validity checks on the inputs

b) Exact sequence of operations

c) Responses to abnormal situations, including

1) Overflow

2) Communication facilities

3) Error handling and recovery

d) Effect of parameters

e) Relationship of outputs to inputs, including

1) Input/output sequences

2) Formulas for input to output conversion

It may be appropriate to partition the functional requirements into subfunctions or subprocesses. This does not imply that the software design will also be partitioned that way.]

## Performance requirements

[This subsection should specify both the static and the dynamic numerical requirements placed on the software or on human interaction with the software as a whole. Static numerical requirements may include the following:

a) The number of terminals to be supported;

b) The number of simultaneous users to be supported;

c) Amount and type of information to be handled.

Static numerical requirements are sometimes identified under a separate section entitled Capacity.

Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions.

All of these requirements should be stated in measurable terms.

For example,

*95% of the transactions shall be processed in less than 1 s.*

rather than,

*An operator shall not have to wait for the transaction to complete.*

NOTE: Numerical limits applied to one specific function are normally specified as part of the processing subparagraph description of that function.]

## Logical database requirements

[This should specify the logical requirements for any information that is to be placed into a database. This may include the following:

a) Types of information used by various functions;

b) Frequency of use;

c) Accessing capabilities;

d) Data entities and their relationships;

e) Integrity constraints;

f) Data retention requirements. ]

## Design constraints

[This should specify design constraints that can be imposed by other standards, hardware limitations, etc.]

### Standards compliance

[This subsection should specify the requirements derived from existing standards or regulations. They may include the following:

a) Report format;

b) Data naming;

c) Accounting procedures;

d) Audit tracing.

For example, this could specify the requirement for software to trace processing activity. Such traces are needed for some applications to meet minimum regulatory or financial standards. An audit trace requirement may, for example, state that all changes to a payroll database must be recorded in a trace file with before and after values.]

## Software system attributes

[There are a number of attributes of software that can serve as requirements. It is important that required

attributes be specified so that their achievement can be objectively verified.]

### Reliability

[This should specify the factors required to establish the required reliability of the software system at time of delivery.]

### Availability

[This should specify the factors required to guarantee a defined availability level for the entire system such as checkpoint, recovery, and restart.]

### Security

[This should specify the factors that protect the software from accidental or malicious access, use, modification, destruction, or disclosure. Specific requirements in this area could include the need to

a) Utilize certain cryptographical techniques;

b) Keep specific log or history data sets;

c) Assign certain functions to different modules;

d) Restrict communications between some areas of the program;

e) Check data integrity for critical variables.]

### Maintainability

[This should specify attributes of software that relate to the ease of maintenance of the software itself. There may be some requirement for certain modularity, interfaces, complexity, etc. Requirements should not be placed here just because they are thought to be good design practices.]

### Portability

[This should specify attributes of software that relate to the ease of porting the software to other host machines and/or operating systems. This may include the following:

a) Percentage of components with host-dependent code;

b) Percentage of code that is host dependent;

c) Use of a proven portable language;

d) Use of a particular compiler or language subset;

e) Use of a particular operating system. ]

# Appendixes

[The appendixes are not always considered part of the actual SRS and are not always necessary. They may include

a) Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;

b) Supporting or background information that can help the readers of the SRS;

c) A description of the problems to be solved by the software;

d) Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When appendixes are included, the SRS should explicitly state whether or not the appendixes are to be considered part of the requirements.]