LGE Internal Use Only

LGE VS [OEM Name] [Project Name]

**SW Architectural Design**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **About This Template**   * Template Name: LGE\_VS\_SWAD\_T01\_SW Architecture Design(SAD) * Management Department: VS SW Process Team * Revision History  |  |  |  |  |  | | --- | --- | --- | --- | --- | | Version | Date | Comment | Author | Approver | | 2.5 | 2016-04-18 | Initial Release | EY.Koh  (VS SW Process Team) | VS SW Process Team Leader | | 2.6 | 2018-01-04 | 4. Interface Design’s options added. | SI.Jung  (VS SW Process Team) | VS SW Process Team Leader | | 2.7 | 2019-03-08 | Update due to annual organization restructuring (VC --> VS) | SH.Lee  (VS SW Process Team) | VS SW Process Team Leader | | 2.9 | 2021-08-24 | (\* actually updated in ’19.03.29 by TJ.Park)  - Updated ‘Read me first’, ‘About this template’  - Document structure | Soo.Yoon  (VS SW Process Unit) | SW Process Unit Leader | | Updated security notice of this template  (Before: LGE Confidential->After: LGE Internal Use Only) Security level related note (the last sentence in red color below) | |

|  |
| --- |
| **Read me first**   * This template is a basic form for a Software Architectural Design document and consists of guidelines (light green boxes) and examples. * Before writing the document, read and understand the General Guidelines. * After writing the document, all guidelines (light green boxes) should be deleted. * Since the content composition of this template may not be 100% suitable for all projects, it should be tailored according to the characteristics of the project. * The definitions of SW Element (or SW Module), SW Component, and SW Unit used in this document are as follows.   - SW Element: Among the System Elements defined in SysAD, Elements implemented in SW have a hierarchical level in the SW architecture.  ASPICE defines SW Element as follows.  “The software is decomposed into elements of the software architecture across appropriate hierarchical levels down to the software components (the lowest level elements of the software architecture).”  - SW Component: As the lowest level element defined in SAD (SW Architecture Design), it becomes the unit of SDD (SW Detailed Design) document.  - SW Unit: The smallest unit constituting SW (e.g. Function in C language)   * The notice “LGE Internal Use Only” is for this template itself. The document which use this template needs to be classified as suitable security level according to its content |

|  |
| --- |
| **General Guidelines!! You should not miss these.**   1. **Ground Rule on SAD Writing**     * In SAD, at least one or more SW components should be identified through architectural design. SDD will be documented by component for detatiled design.    * Rationales for the architectural desicisions should be fully represented.    * Describe quality requirement scenarios measurably in quantitative way. Also, these requirements should be applied when designing architecture.    * Any kind of design tools can be used for SAD. This template uses EA for diagrams. 2. **Seven Rules for Sound Documentation (Source: Documenting Software Architecture - Second edition)**    1. Documentation should be written from the point of view of the reader, not the writer.    2. Avoid unnecessary repetition    3. Avoid ambiguity. Define the meaning of notation used in diagram clearly.    4. Use the heirachical structure of this document.   If there is blank which cannot be filled yet, fill it with “to be determined” at once.   * 1. Record rationale. It will be useful when reviewing or changing the design later.   2. Do not change the document right after the design change, but change the document according to the version control and release schedule.   3. Review documentation for fitness of purpose.  1. **References**     * Automotive\_SPICE\_PAM\_v3.0 (SWE.2)    * Software Architecture in Practice, Third Edition    * Documenting Software Architectures : Views and Beyond, Second Edition    * IEEE 1471-2000, IEEE Recommended Practice for Architectural Description for Software-Intensive Systems    * Template that has been previously used in LGE VS Smart devision 2. **Document Format**   These are general rules for document format, which are applied to this template.   * + Figures and tables have captions to find and check them easily.     1. Creating Caption: Create Caption: [References] > [Insert Caption] > [Label] > Select [Figure] or [Table].     2. Put the caption under the figure. Figure 1 Title, Figure 2 Title,..     3. Put the table caption above the table. Table 1 Title, Table 2 Title,..  1. **Document Versioning, File Naming Rule**   Follow the “Work product Naming Rule” from “Configuration Management Plan” |

About This Document

Document Information

|  |  |
| --- | --- |
| Issuing authority | VS OOO Team |
| Configuration ID | Configuration Item ID of CMBook |
| **Status of document** | In Progress / Approved / Released |

Revision History

|  |
| --- |
| The document history is organized in the order that the most recent history is at the top and the first history is at the bottom. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Date | Comment | Author | Approver |
| #.# | YYYY-MM-DD | OOOOOOOOOOOOOOO | OOO | OOO |
| 1.1 | 2016-02-20 | Modified xxx notation of C&C Diagram in 2.2 C&C View. | Gildong Hong | Sam Shin |
| 1.0 | 2016-01-20 | Initial Release | Gildong Hong | Sam Shin |

Table of Contents

|  |
| --- |
| The table of contents should be generated automatically. If you right-click on the table below and select [Update Field] menu, the changed table of Contents will be automatically reflected |

[1 Introduction 8](#_Toc4684775)

[1.1 Purpose 8](#_Toc4684776)

[1.2 Scope 8](#_Toc4684777)

[1.3 Audience 8](#_Toc4684778)

[1.4 Conventions 8](#_Toc4684779)

[1.5 Acronyms / Glossary 9](#_Toc4684780)

[1.6 Related Documents 9](#_Toc4684781)

[2 Architectural Drivers 10](#_Toc4684782)

[2.1 SW Main Features 10](#_Toc4684783)

[2.2 Quality Attributes 11](#_Toc4684784)

[2.3 Constraints 12](#_Toc4684785)

[2.3.1 Business Constraints 12](#_Toc4684786)

[2.3.2 Technical Constraints 12](#_Toc4684787)

[3 SW Architectural Representations 13](#_Toc4684788)

[3.1 Static View 13](#_Toc4684789)

[3.1.1 SW Component Descriptions 14](#_Toc4684790)

[3.2 Dynamic View 15](#_Toc4684791)

[3.2.1 Task Design 16](#_Toc4684792)

[3.2.1.1 Task Structure 16](#_Toc4684793)

[3.2.1.2 Task Scheduling 17](#_Toc4684794)

[3.3 Deployment View 18](#_Toc4684795)

[4 Resource Consumption Objectives 19](#_Toc4684796)

[5 Interface Design 20](#_Toc4684797)

[5.1 SW Component Interface Table 20](#_Toc4684798)

[6 Architectural Alternatives 22](#_Toc4684799)

[6.1 Architectural Design 1 [예: Native implementation of XXX core function] 22](#_Toc4684800)

[6.2 Architectural Design 2 22](#_Toc4684801)

[7 Quality Attribute Scenarios (Optional) 23](#_Toc4684802)

[7.1 Scenario N: Quality Attribute [예: Scenario #1: Performance] 23](#_Toc4684803)

[7.2 Scenario N: Quality Attribute 24](#_Toc4684804)

Figures

|  |
| --- |
| If you right-click on the table below and select [Update Field] menu, the changed table of Figures will be automatically reflected. |

[Figure 1 Module View 13](#_Toc4684805)

[Figure 2 C&C View 15](#_Toc4684806)

[Figure 3 C&C View for Task Scheduling 17](#_Toc4684807)

[Figure 4 Deployment View 18](#_Toc4684808)

Tables

|  |
| --- |
| If you right-click on the table below and select [Update Field] menu, the changed table of Figures will be automatically reflected. |

[Table 1 Software Main Features 1 10](#_Toc4684809)

[Table 2 Software Main Features 2 11](#_Toc4684810)

[Table 3 Quality Attribute Requirements 11](#_Toc4684811)

[Table 4 Business Constraints 12](#_Toc4684812)

[Table 5 Technical Constraints 12](#_Toc4684813)

[Table 6 SW Component Descriptions 14](#_Toc4684814)

[Table 7 Task Structure 17](#_Toc4684815)

[Table 8 Resource Consumption Objectives 19](#_Toc4684816)

[Table 9 SW Component Interface Table 20](#_Toc4684817)

[Table 10 SW Component 1 Interface Table 21](#_Toc4684818)

[Table 11 SW Component 2 Interface Table 21](#_Toc4684819)

[Table 10 Concerns for Design Alternatives 22](#_Toc4684820)

[Table 11 Scenario Refinement for Scenario #1 24](#_Toc4684821)

# Introduction

## Purpose

|  |
| --- |
| Describe the use of this document. |

This document specifies the software architecture design for XXX to support Software Requirement. This design document also serves as a guideline on how each software components in the system should be implemented and how the components should interact with each other.

## Scope

|  |
| --- |
| Describe the scope of contents covered in this document. |

* Architectural Drivers
* SW Architectural Representations
* Resource Consumption Objectives
* Interface Design
* Architectural Alternatives
* Quality Attribute Scenarios

## Audience

|  |
| --- |
| Describe the target audiences (main stakeholders) of this document. |

The target audience of this document is:

* Requirement engineer who will point out any contradiction between the design and the requirement
* Software architect who will evaluate the design of the software
* Component developer who will implement the design in actual code
* Application developers who need to interact with [SW Name]
* XXX project participants who want to understand the architecture of the [SW Name]
* Test engineers who verify [SW Name] and others related

## Conventions

|  |
| --- |
| Indicate what you want to emphasize or what is helpful. |

NOTE

useful notes

## Acronyms / Glossary

|  |
| --- |
| Describe the abbreviations and explanations used in this document.  The table should be written in alphabetical order of abbreviations / terms. In addition to the acronym, a description is required. |

|  |  |
| --- | --- |
| Acronym | Description |
| C&C View | Component-and-Connector View |
| IHU | IVI Head Unit |
| SAD | Software Architecture Design |
| SDD | Software Detailed Design |

|  |  |
| --- | --- |
| Glossary | Description |
|  |  |
|  |  |

## Related Documents

|  |
| --- |
| List the referenced documents when writing this document.  Describe the titles of the referenced documents. If there are multiple versions of the document, also describe the version name of the referenced document. |

Documents related to this document include:

* IHU\_MAIN SRS (Software Requirement Specification) v1.0
* IHU\_MAIN SDD (Software Detailed Design) v1.0

# Architectural Drivers

|  |
| --- |
| Architectural drivers are a set of requirements that have significant influence over your architecture and performance/quality. In this chapter, describe software main features, quality attributes, and constraints on the basis of SRS. |

## SW Main Features

|  |
| --- |
| * (Mandatory) Describe main features of the software to implement in following format. * (Optional) It is allowed to copy contents from SRS in order to keep consistency. |

**Example 1**

Table 1 Software Main Features 1

|  |  |  |  |
| --- | --- | --- | --- |
| Level 1 | Level 2 | Level 3 | Descriptions |
| IHU\_MAIN | Application Manager | Application Lifecycle Management | - Operates application such as resuming app, pausing app and launching app.  - Manages application state: applications can have states such as launched, paused, etc. |
| Application Information | … |
| … | … |
| Application Launcher | Boot Startup Management | - Decides the boot startup sequence based on various factors including the last used application.  - Delivers the LUC information to Application when starting the Application |
| Application Monitor | … |
| … | … |
| AppCommon API | Client API | - Registers application for communication  - Event Notification to and from applications |

**Example 2**

Table 2 Software Main Features 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Level 1 | Level 2 | Level 3 | Level 4 | Descriptions |
| Multimedia | Audio Player | Disc Audio Player |  | Plays music that is stored in disc (CD, MP3, DVD). |
| USB Audio Player |  | Plays music that is stored in USB. |
| Telematics | Telephony | Call | Voice Call Request | Requests voice call connection to the given number. |
| Voice Call Answer | Answers the incoming voice call. |
| … | … |
| Remote Vehicle Control | Remote Door Lock | Remote door lock. |
| … | … |

## Quality Attributes

|  |
| --- |
| * (Mandatory) Describe scenarios for each quality requirement specified in SRS. * (Mandatory) Scenarios should be described measurably in quantitative way. * (Optional) In Chapter 7 Quality Attribute Scenarios, refine high-priority scenarios. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Scenario #** | Scenario | Quality Attribute | SRS ID | Priority |
| Scenario number: It will be used to identify scenarios in Chapter 6. | Describe scenarios for quality attributes measurably in quantitative way. If quality attributes are described in SRS, you can copy them here. | Related quality attribute | Related SRS ID | Priority specified in SRS |

Table 3 Quality Attribute Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Scenario # | Scenario | Quality Attribute | SRS ID | Priority |
| 1 | The IVS shall generate an eCall within 1 second after the confirmed triggering signal is received. | Performance | QA1\_1 | High |
| 2 | … | … |  | … |
| 3 | … | … |  | … |
| 4 | … | … |  | … |

## Constraints

|  |
| --- |
| (Mandatory) Describe constraints from business and technical perspectives. |

### Business Constraints

|  |  |  |
| --- | --- | --- |
| **ID** | Business Constraint | SRS ID |
| Business constraint ID | Describe constraints from business perspective.  If business constraints are described in SRS, you can copy them here. | Related SRS ID |

Table 4 Business Constraints

|  |  |  |
| --- | --- | --- |
| ID | Business Constraint | SRS ID |
| BC1 | Deploy the architecture system in 10 weeks. |  |
| BC2 | ... |  |

### Technical Constraints

|  |  |  |
| --- | --- | --- |
| **ID** | Technical Constraint | SRS ID |
| Techincal constraint ID | Describe constraints from technical perspective. If technical constraints are described in SRS, you can copy them here. | Related SRS ID |

Table 5 Technical Constraints

|  |  |  |
| --- | --- | --- |
| ID | Scenario | SRS ID |
| TC1 | The system consists of a head office server located at the head office, and the POS terminals placed at store cashiers |  |
| TC2 | ... |  |

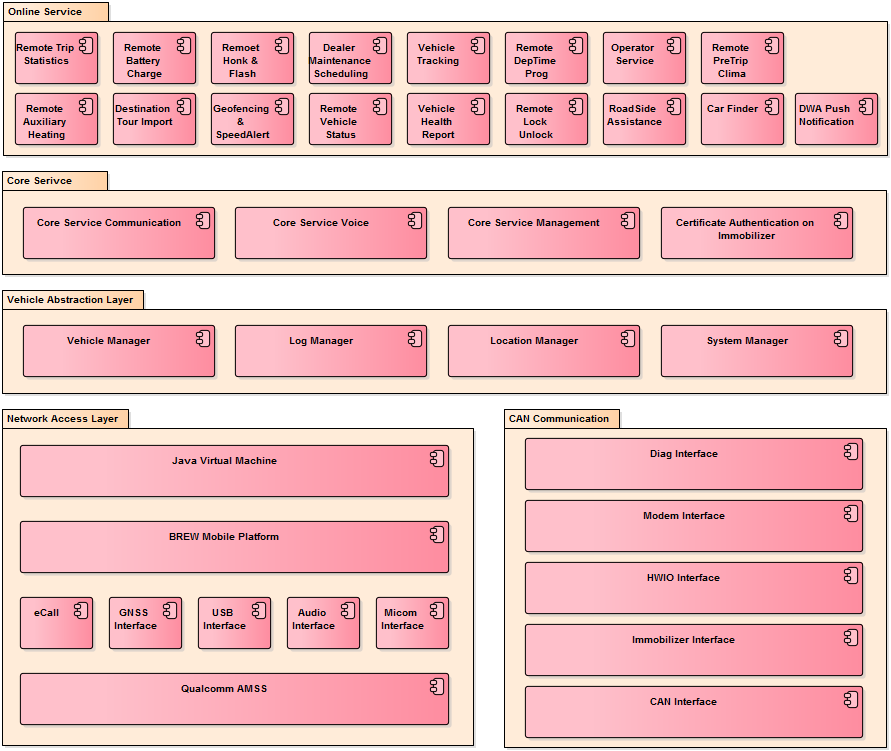
# SW Architectural Representations

|  |
| --- |
| * Define the software architecture in module, C&C, and deployment viewtypes required for different perspectives of various stakeholders. * The main purpose of software architectural representations is to identify components consisting the the software. |

## Static View

|  |
| --- |
| * (Mandatory) Define the software architecture of target software in module viewtype. * (Mandatory) In Section 3.1.1 SW Component Descriptions, you need to describe components identified below. |

This is the software architectural design of OOO. OOO consists of Remote Trip Statistics, Remote Battery Charge, Remote Honk&Flash.



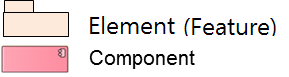


Figure 1 Module View

### SW Component Descriptions

|  |
| --- |
| * (Mandatory) Describe all components which have been identified in Section 3.1 Static View. For each component, related SRS IDs should be specified. * Depending on the software size and characteristic, components may not be classified as features. * If SRS has a large amount of requirements, it is more efficient to manage traceability in a separate excel file. * (Optional) If you will reuse or buy a component, add a column and indicate whether the component is reusable or buyable. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | SW Component ID | **SW Component Name** | **Description** | **SRS ID** |
| Feature name | Component ID | Component name | Describe the component. | Related SRS ID |

Below table describes the software components which are specified in the chapter 3.1.

Table 6 SW Component Descriptions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Feature | SW Component ID | SW Component Name | Description | SRS ID |
| Online Service | XXX\_SWC-001 | Remote Trip Statistics | Vehicle’s trip information (mileage, average consumption) | SRS-XXX\_100 |
| XXX\_SWC-002 | Remote Battery Charge | To control e-vehicle’s battery charging function remotely | SRS-XXX\_101 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Dynamic View

|  |
| --- |
| * (Mandatory) Define the software architecture in Component & Connector viewtype to describe connection relations and communication mechanisms between components. Component in C&C view means principal processing units and data stores. * In C&C view, component types are client, server, filter, object, and DB. * (Mandatory) In Chapter 5 Interface Design, you need to describe all interfaces identified below. |

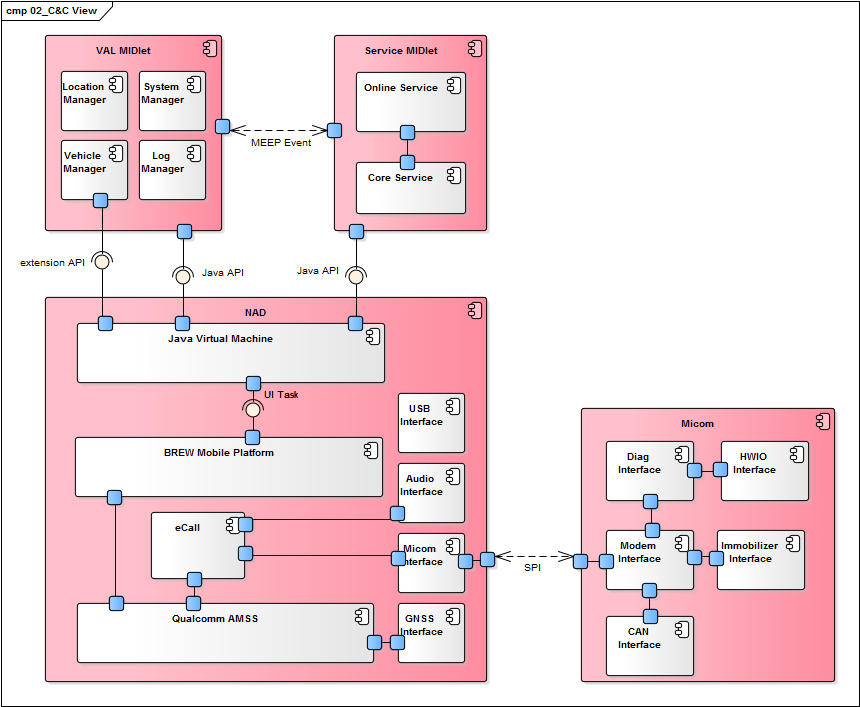


Figure 2 C&C View

This diagram mainly represents components consisting the OCU architecture and interfaces connecting between components.

1. Service MIDlet consists of Java Online service and Java Core service, and it uses Java standard API.
2. …
3. …
4. …

### Task Design

|  |
| --- |
| * (Mandatory) Identify tasks to be executed during runtime, define the structure of tasks, and describe related components and functions. Run units such as processes and threads are called task. * As run unit depends on the environment such as OS and development languages, tasks should be designed according the execution environment. Depending on the development environment like BSP, OS, or AUTOSAR platform, task types and constraints may change. |

This chapter describes the tasks that the IHU\_MAIN module requires. The task is the runtime unit that executes a job periodically or event-basis. The task is represented as process, thread, etc.

#### Task Structure

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task ID** | **Task Name** | **Description** | **SW Component Name** | **Constraints** |
| Task ID | Task name | Describe the responsibility of the task | Component name | Generally, following items are described for constraints.  - Cycle time  - Processing time  - Memory usage (heap, stack size)  Properties defined for a task in code or configuration file are described here usually. e.g. priority, cycle time, and memory size |

The task may have constrains that the maximum cycle time, etc. that needs to be described in the below table.

Table 7 Task Structure

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task ID | Task Name | Description | SW Component Name | Constraints |
| IHU\_MAIN\_SWT\_001 | AppManager Process | This task runs as a process, and receives requests for resuming and pausing Applications. | AppManager | Cycle time: n/a (request basis processing)  Processing time: Maximum startup time to start last audio/video source is 10 seconds. |
| IHU\_MAIN\_SWT\_001 | AppLauncher Process | … | … | ... |

#### Task Scheduling

|  |
| --- |
| Describe the task scheduling result by considering the task execution time, order, and cycle. Check that if there is no problem for synchronizing tasks and executing a task within max processing time. |

When the IHU\_MAIN module is executed at the startup of IHU System, AppManager and AppLauncher are executed as separate processes. They load AppCommon library to communicate with other entities (e.g., AppManager and AppLauncher use AppCommon library to communicate).

AppManager manages the lifecycle of Applications such as resuming and pausing them. Application may be the native Application written with QT, or the Web Application. The runtime view for AppManageris shown in the following figure:



Figure 3 C&C View for Task Scheduling

## Deployment View

|  |
| --- |
| (Mandatory) Define the software architecture representing relations between softwares and hardwares consisting the system, on the basis of C&C view defined in Section 3.2 Dynamic View. Describe main considerations if needed. |

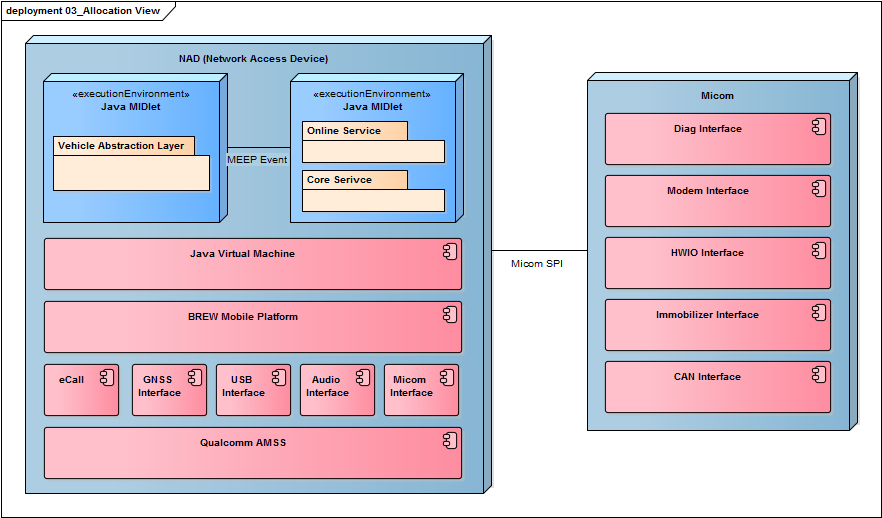


Figure 4 Deployment View

# Resource Consumption Objectives

|  |
| --- |
| (Mandatory) Describe resource consumption for resources like CPU, runtime memory, and ROM memory for all software components which have been identified in Section 3.1.1. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Feature** | SW Component Name | **CPU Load** | **CPU Load** | **Runtime Memory** | ROM Size | Remarks |
| Feature name | Component name | CPU load of the component (%) | CPU load of the element (%) | Runtime memory size of the element | ROM size of the element | Remarks |

Below table shows the estimated CPU load, runtime memory and rom memory for each software components.

Table 8 Resource Consumption Objectives

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Feature | SW Component ID | SW Component Name | CPU Load | CPU Load | Runtime Memory | ROM Size | Remarks |
| Online Service | XXX\_SWC-001 | Remote Trip Statistics | 15% | 30% | OO MB | OO MB |  |
| XXX\_SWC-002 | Remote Battery Charge | … |  |
|  | … | … |  |
|  |  |  | 3% | 40% | OO MB | OO MB |  |
|  |  | … |  |
|  |  | … |  |
|  |  | … |  |
|  |  | … |  |
|  |  | … |  |
|  |  |  | 10% | 70% | OO KB | OO KB |  |
|  |  | … |  |
|  |  | … |  |
|  | … | … |  |
|  | … | … |  |

# Interface Design

|  |
| --- |
| Describe all interfaces which have been idenfified in [Section 3.2 Dynamic View](#_C&C_View). |

## SW Component Interface Table

|  |
| --- |
| (Mandatory) Define interfaces for each SW component.  (Option 1)  If you are supposed to make 1 table to represent interface list, Write SW Component Name at the first columm. |

|  |  |  |  |
| --- | --- | --- | --- |
| **SW Component Name** | Interface Name | **Type** | **Parameters** |
| Component name | Interface name | Choose among types below. (Can be added)  - call: Call local function.  - dbus: Communicate with dbus.  - LIN: | Input/output parameters |

This section specifies the interface related to Application Launcher.

Table 9 SW Component Interface Table

|  |  |  |  |
| --- | --- | --- | --- |
| SW Component Name | Interface Name | Type | Parameters |
| Application Loader | MakeLaunchAppList | call | In:   * context data: LUC, country information, car variant   Out:   * Boot sequence list: list in which order the Application shall be started |
| … | … | … |
| … | … | … |
| AppProcess | launch | call | N/A |
| checkRestart | call | In:   * Restart count: the restart number until now |
| … | … | … |
| Logger | log | call | In:   * Data: data to be logged |

|  |
| --- |
| (Option 2)  If you are supposed to make tables separately to represent interface list for the SW component, create multiple tables using below. |

|  |  |  |
| --- | --- | --- |
| Interface Name | **Type** | **Parameters** |
| Interface name | Choose among types below. (Can be added)  - call: Call local function.  - dbus: Communicate with dbus.  - LIN: | Input/output parameters |

This section specifies the interface related to Application Launcher.

Table 10 SW Component 1 Interface Table

|  |  |  |
| --- | --- | --- |
| Interface Name | Type | Parameters |
| MakeLaunchAppList | call | In:   * context data: LUC, country information, car variant   Out:   * Boot sequence list: list in which order the Application shall be started |
| … | … | … |
| … | … | … |
| log | call | In:   * Data: data to be logged |

Table 11 SW Component 2 Interface Table

|  |  |  |
| --- | --- | --- |
| Interface Name | Type | Parameters |
| checkRestart | call | In:   * Restart count: the restart number until now |
| … | … | … |
| … | … | … |
| … | … | … |
| launch | call | N/A |

# Architectural Alternatives

|  |
| --- |
| (Mandatory) Evaluate alternative software architectures and describe rationales for the chosen software architecture. |

## Architectural Design 1 [예: Native implementation of XXX core function]

The following core functions of XXX need to be implemented using native because starting XXX should not be effected by any architecture related to JavaVM or abstraction. (from customer requirement ID XX, XX)

- …

- …

- ….

For this, design alternatives are considered with the following concerns:

Table 10 Concerns for Design Alternatives

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Concerns | Alternative #1:  using Java XXX | Alternative #2:  using native XXX |
| AD\_1 | Reduce time to market for future enhancements | **Yes** | No |
| AD\_2 | Reduce costs | **Yes** | No |
| AD\_3 | Reduce risks | No | **Yes** |
| AD\_4 | System qualities | No | **Yes** |
| AD\_5 | Reuse existing infrastructure | **Yes** | Unknown |
| AD\_6 | Use proven technologies | Unknown | **Yes** |
| AD\_7 | Performance | No | **Yes** |

In terms of reuse or maintenance, it is advantageous to use Java, however, the core function of XXX is XXXX. Therefore, we decided to implement XXX in native application.

## Architectural Design 2

# Quality Attribute Scenarios (Optional)

|  |
| --- |
| (Optional) Select high-priority scenarios from Section 2.2 Quality Attributes and refine them.  By refinement, requirements can be changed into measurable quality attribute goals so that you can set strategies more easily when designing software architecture. |

## Scenario N: Quality Attribute [예: Scenario #1: Performance]

|  |
| --- |
| Refine each selected scenario according to the following format. |

|  |  |  |
| --- | --- | --- |
| Scenario Refinement for Scenario N | | |
| **Scenario** | | Quality requirement scenario |
| **Scenario Components** | **Stimulus** | A condition that requires a response when it arrives at a system |
| **Source of stimulus** | Some entity (a human, a computer system, or any other actuator) that generated the stimulus |
| **Environment** | Certain condition that occurs the stimulus |
| **Artifact (If known)** | Some artifact (a collection of systems, whole system, or some piece) that is stimulated |
| **Response** | Activity undertaken as the result of the arrival of the stimuls |
| **Response measure** | Measurable response to make it testable |
| **Architectural Decisions & Reasoning** | | Architectual decisions affecting quality attribute responses, rationales, and discussions |
| **Risks** | | Anaylzed risks |
| **Trade-off** | | Anaylzed trade-off as a result of architectural decision. |
| **Allocated SW Component** | | Allocated SW component name |

Table 11 Scenario Refinement for Scenario #1

|  |  |  |
| --- | --- | --- |
| **Scenario Refinement for Scenario #1** | | |
| **Scenario** | | The IVS shall generate an eCall within 1 sec after the confirmed triggering signal is received. |
| **Scenario Components** | **Stimulus** | Airbag deployment signal / Emergency call button pressed |
| **Source of Stimulus** | Airbag or Button |
| **Environment** | Car accident or emergency situation |
| **Artifact (If Known)** |  |
| **Response** | Trigger eCall after receiving the airbag or button signal. |
| **Response Measure** | Generate eCall within 1 sec after receiving the signal. |
| **Architectural Elements** | | eCall / CAN Interface / HWIO Interface / Modem Interface / Micom Interface / Audio Interface |
| **Architectural Decisions & Reasoning** | | Time required for communicating between Micom and NAD using SPI 🡪 Airbag crash signal is set to high priority.  Time required for communicating with MIB to secure audio path 🡪 If there is no response from MIB within OOO seconds, set HW mute and change to use local loud speaker. |
| **Risks** | | 1) Low priority services may be blocked  2) If converting to local loud speaker gets delayed, eCall sound may not be heard properly. |
| **Trade-off** | | None |
| **Allocated SW Component** | | OOO |

## Scenario N: Quality Attribute