|

Architecture and design Specification (ADS)

<PROJECT>

Journal: Architecture and Design Specification (ADS).docx

Date: <date>

Revision: <revision>

Pages: 20

Initials: <INI>

|  |
| --- |
| © Crisplant a/s 2016  This document and all data contained herein are the exclusive intellectual property rights of Crisplant a/s and are supplied in strictest confidence under the express condition that any reproduction, transmission, transcription etc., or storing in any retrieval system/form, or any kind of supply to others is strictly forbidden without express prior written consent. The receiver guarantees that any disclosed material shall not be used in any way detrimental to the interest of Crisplant a/s. |

**Contents**

1. Revision 4

2. Introduction 5

2.1. Identification 5

2.2. Scope and Purpose 5

2.3. References and Related design documents 5

2.4. Terms and Definitions 5

3. System Concept Overview 6

3.1. Overview Drawing 6

3.2. Overview Description 6

4. Requirements 7

4.1. Requirement - Implementation Mapping 7

4.2. Design Criteria 7

5. Hardware Overview 8

6. Software Architecture 9

6.1. Overview 9

6.2. Requirements 9

6.2.1. Service Requirements 9

6.3. Technology base 9

6.4. Subsystems 9

6.4.1. Subsystem AAA 9

6.4.2. Subsystem BBB 9

6.5. Reusable components 9

6.6. Risk Control 9

7. Domain/Data Model 10

8. Appendix A: Check list and Guidance 11

8.1. Identification 11

8.2. Scope and Purpose 11

8.3. References 11

8.4. Terms and Definitions 11

8.5. System Concept Overview 11

8.6. Requirement - Implementation Mapping 11

8.7. Design Criteria 12

8.8. Hardware Overview 12

8.9. Software Architecture 12

8.10. Overview 13

8.11. Requirements 13

8.12. Service Requirements 13

8.13. Technology 13

8.14. Subsystem description 14

8.15. Reusable components 14

8.16. Risk Control 14

8.17. Domain/Data Model 14

9. Appendix B: Examples 15

9.1. System Concept Overview 15

9.2. Software Architecture 16

# Revision

| Rev | Date | Init. | Description |
| --- | --- | --- | --- |
| 0.1 |  |  | Initial revision |

# Introduction

## Identification

## Scope and Purpose

## References and Related design documents

## Terms and Definitions

# System Concept Overview

## Overview Drawing

## Overview Description

# Requirements

## Requirement - Implementation Mapping

| No | Requirement | Implementation |
| --- | --- | --- |
|  | Host communication - implementation of xxx.p1. | Is handled by task yyy as described in section y.h. |

Table 1 **Functions to be handled by the project specific software.**

## Design Criteria

# Hardware Overview

# Software Architecture

## Overview

## Requirements

### Service Requirements

## Technology base

## Subsystems

### Subsystem AAA

### Subsystem BBB

## Reusable components

## Risk Control

# Domain/Data Model

# Appendix A: Check list and Guidance

The check lists are divided according to the parts of the document to which they are relevant.

## Identification

Identification of the project, customer and end user.

## Scope and Purpose

Provide an overview of the entire document:

1. Describe the purpose of this document
2. Describe the scope of this document
3. Describe this document's intended audience in terms of how they should use it.
4. Identify the document as a complete document, or as one of a group of documents.
5. Identify any information not covered by this document that could be reasonably expected to be covered and identify appropriate documents where such information is covered.
6. Summarize (or give an abstract for) the contents of this document.

## References

Provide references for any other pertinent documents such as:

1. Related and/or companion documents
2. Prerequisite documents (Requirement specifications related to the design)
3. Documents which provide background and/or context for this document
4. Documents that result from this document (e.g. test specification or integration plan)
5. Overview or plan for design documents in the project.

Examples:

1. system requirements
2. interface protocols
3. base system sw documentation

## Terms and Definitions

Define any new important terms, acronyms, or abbreviations. Be sure to use the same terms, acronyms and definitions that are used in referenced high level documents.

## System Concept Overview

A short description and overview of the system. Look at the system as an unopened black box and describe what is around it.

## Requirement - Implementation Mapping

The main design criteria are to ensure that all requirements/functions are included in the design. This is accomplished with a list of functions in this document. The list includes references to the sections in this document, in which the design/implementation of the function is described.

Do not forget:

1. new requirements.
2. test and debug demands

## Design Criteria

In addition to the list of functions, the design must ensure that certain criteria are fulfilled. List in order of importance. Example:

1. Capacity (ex. number of items, number of database transactions)
2. Responsive, ie. response times (time from input to output, ex. code conversion time)
3. Deadlocks
4. Reliability
5. Robust to requirement changes
6. Encapsulation of hardware dependency
7. Modular testability
8. Simplicity
9. Parallel development
10. Binary configurable, ie. stable interfaces and encapsulation of functionality
11. Write the sw for reuse later on in a component
12. Minimizing disk space
13. Minimizing memory use

## Hardware Overview

Describe the hardware and check sw demands for the hw

1. Check that product standards exists for all used hardware.
2. Does the hardware configuration support the required level of fault tolerance and redundancy ?

## Software Architecture

Describe the tasks/processes in the system.

This chapter gives a short survey through the modules and tasks of the project specific sw, including:

1. High level diagrams showing the threads, control flow and data flow in the system
2. High level diagrams showing the packages in the system and their main dependencies.

Do not forget to take the following into consideration:

1. race conditions, unexpected results because of thread execution times.
2. task priority, how are the tasks prioritized?
3. deadlocks, prevent deadlocks by examining the way the threads are executed.
4. process/thread control, control of start/stop/execution of threads
5. task/thread allocation, ie. Who executes the code?
6. internal communication concept, how do the threads communicate?
7. allocation of threads to Java Virtual Machines
8. UPS, monitoring power supply
9. Load monitoring, How will Memory use, CPU load and Disk space use be monitored ?
10. module structure
11. directory structure, Describe the project specific implementation of the directory standard. How will the “src” directory be structured ?
12. the use of database
13. data cleanup strategy
14. clear definition of interfaces and responsibilities between the modules
15. critical sections/semaphores, which parts of the software are used by multiple threads and are not thread safe?
16. synchronization between different data copies
17. consider if normal control logic for this type of plant has been taken into consideration, ie compare the existing designs of this type of plant.
18. implementation strategy for associations.
19. strategy for startup, shutdown, self-test, backup, crash recovery and error handling.

## Overview

Draw an overview of the software.

## Requirements

Identify requirements to the software that are not present in the system requirement specification.

1. restart of part of the system, how should it be handled?
2. performance/capacity/response time, requirements for system performance.
3. bottlenecks, which kind of bottlenecks are not permitted ?
4. parallel computing, should independent tasks be handled in parallel?
5. error tolerance/error handling, how should errors be handled?
6. flexibility, what are the obvious extensions in the future ?
7. data persistence, how should data be stored?
8. RAM space, what is the limit, should the usage be calculated ?
9. hard disk space, what is the limit, should the usage be calculated ?
10. security, security levels and users.
11. load on internal communication interfaces
12. log and debug tools, how should it be debugged runtime?
13. handling of language, what languages should be handled?
14. data handling performance

## Service Requirements

1. Trace plan, Which traces should exist and what information should be included

## Technology

Which technology is used in the project. Ex:

1. chosen components (JDK, DAWOS, DIO, DTOOLS etc)
2. database engine
3. backup concept
4. sw development language
5. WinDAP version

## Subsystem description

List all the subsystems including GUI and System Interfaces (ex. protocols). Write a short description of each part (module/task/GUI/interface) of the software, including relationships and dependencies between the parts.

For each part that is described the description might include the following:

1. identification
2. type
3. purpose and responsibility
4. function
5. sub-elements
6. dependencies
7. interface with interface method, interface data, return value and distribution of responsibility (sequence model of interface use)
8. resources
9. how the job is done?
10. internal data
11. internal states (state/event table)
12. description of its responsibilities
13. interface to the rest of the system (request codes and data sent)
14. is it possible to test the sub-elements?
15. new software requirements (review requirements in sec.8.11 Requirements)
16. is it possible to fulfill the requirements.

Add the following items to the description of each external interface:

1. Description and type of interface
2. Transport interface protocol
3. Application interface protocol (definition af telegrams)
4. Data

## Reusable components

Description of:

1. reuse of existing components
2. use of design patterns
3. including modified base tasks.
4. what can be reused from this project in future projects

## Risk Control

List of sw risks ranged according to priority. What is new and what might go wrong.

## Domain/Data Model

When designing according to SPU this chapter contains data types, according to OO it contains domain object model. Describe the types and their purpose.

1. Is the object model able to implement all use cases?
2. Is the entry point(s) clearly identified?
3. Are unnecessary details and application classes left out?
4. Database layout

# Appendix B: Examples

## System Concept Overview

Example from disney.s1:



Example from parcv.d5.



## Software Architecture

Example from billund.d6:



Example from tatf.d1.



Example from parcv.d6:



Example from parcv.d8:

