**© 2015 – Foxes Team**

[http://process.ws](http://foxes.info.vn)

**Software Architecture Document**

**Foxes Project**

**Abstract**

…

**Table of Contents**

[1. Introduction 3](#_Toc428220560)

[1.1. Purpose 3](#_Toc428220561)

[1.2. Scope 3](#_Toc428220562)

[1.3. Definitions and Abbreviations/Acronyms 3](#_Toc428220563)

[1.4. Users of the Document 4](#_Toc428220564)

[2. Architectural Representation 4](#_Toc428220565)

[3. Architectural Goals and Constraints 4](#_Toc428220566)

[3.1. General Constraints 4](#_Toc428220567)

[3.2. Goals and Guidelines 4](#_Toc428220568)

[3.3. Architecturally Significant Driving Requirements 4](#_Toc428220569)

[3.4. Solution Background 5](#_Toc428220570)

[3.5. Technologies Used 5](#_Toc428220571)

[4. Use-Case View 5](#_Toc428220572)

[5. Logical View 5](#_Toc428220573)

[5.1. Overview 5](#_Toc428220574)

[5.2. Architecturally Significant Design Packages 6](#_Toc428220575)

[5.3. Sequence Diagram 6](#_Toc428220576)

[5.4. Collaboration Diagram 6](#_Toc428220577)

[5.5. Class Diagram 6](#_Toc428220578)

[6. Process View 6](#_Toc428220579)

[7. Deployment View 7](#_Toc428220580)

[8. Implementation View 7](#_Toc428220581)

[8.1. Overview 8](#_Toc428220582)

[8.2. Layers 8](#_Toc428220583)

[9. Data View 8](#_Toc428220584)

[10. Size and Performance 8](#_Toc428220585)

[11. Quality 8](#_Toc428220586)

[12. Flow of Events – Design 8](#_Toc428220587)

[12.1. Main Flow 9](#_Toc428220588)

[12.2. Alternative Flow 9](#_Toc428220589)

[12.3. Derived Requirements 9](#_Toc428220590)

[13. Business Rules 9](#_Toc428220591)

[14. Interface Design 9](#_Toc428220592)

[15. Database Design Description 9](#_Toc428220593)

[16. Reference 9](#_Toc428220594)

[17. Appendix 10](#_Toc428220595)

[Revision 11](#_Toc428220596)

**Table of Figures**

# Introduction

## Purpose

*[This section defines the purpose of the* ***Software Architecture Document****, in the overall project documentation, and briefly describes the structure of the document. The specific audiences for the document should be identified, with an indication of how they are expected to use the document.]*

## Scope

*[A brief description of what the Software Architecture Document applies to; what is affected or influenced by this document.]*

## Definitions and Abbreviations/Acronyms

*[This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the* ***Software Architecture Document****. This information may be provided by reference to the project Glossary.]*

### Definitions

|  |  |
| --- | --- |
| **Terms** | **Definition** |
|  |  |
|  |  |

### Abbreviations/Acronyms

|  |  |
| --- | --- |
| **Abbreviations/Acronyms** | **Definition** |
|  |  |
|  |  |

## Users of the Document

*[All concerned who are responsible for preparing any kind of documentation needs to be included in this section.]*

# Architectural Representation

*[This section describes what software architecture is for the current system, and how it is represented. Of the* ***Use-Case****,* ***Logical****,* ***Process****,* ***Deployment****, and* ***Implementation Views****, it enumerates the views that are necessary, and for each view, explains what types of model elements it contains.]*

# Architectural Goals and Constraints

## General Constraints

*[This section describes any global limitations or constraints that have a significant impact on the architecture. The section also describes the associated impact.]*

## Goals and Guidelines

*[This section describes any goals, guidelines or principles which control or embody the system architecture.]*

## Architecturally Significant Driving Requirements

*[This section describes the software requirements and objectives that have some significant impact on the architecture, for example, safety, security, privacy, use of an off-the-shelf product, portability, distribution, and reuse. It also captures the special constraints that may apply: design and implementation strategy, development tools, team structure, schedule, legacy code, and so on.]*

## Solution Background

*[This section describes why the architecture is the way it is and argument to convince that the architecture is the right one to satisfy the requirements, goals.]*

## Technologies Used

*[Briefly describe the technologies selected (selected architecture/technical solutions.)*

*If the selected architecture/technical solution comes from a DAR, provide the reference to the DAR report*

*If the architecture/technical solution is selected not using DAR, put comments why the solution is selected, for ex: solution is required by customer.]*

# Use-Case View

*[This section lists use cases or scenarios from the use-case model if they represent some significant, central functionality of the final system, or if they have a large architectural coverage - they exercise many architectural elements, or if they stress or illustrate a specific, delicate point of the architecture.]*

# Logical View

*[This section describes the architecturally significant parts of the design model, such as its decomposition into subsystems and packages. And for each significant package, its decomposition into classes and class utilities. You should introduce architecturally significant classes and describe their responsibilities, as well as a few very important relationships, operations, and attributes.]*

## Overview

*[This subsection describes the overall decomposition of the design model in terms of its package hierarchy and layers.]*

## Architecturally Significant Design Packages

*[For each significant package, include a subsection with its name, its brief description, and a diagram with all significant classes and packages contained within the package.*

*For each significant class in the package, include its name, brief description, and, optionally a description of some of its major responsibilities, operations and attributes.]*

## Sequence Diagram

*[Behavioural specification that comprises a sequence of message exchanges among a set of objects within a collaboration to accomplish a specific purpose.*

*This section is OPTIONAL because it could be described in Low Level Design documents.]*

## Collaboration Diagram

*[They provide an alternative notation for interaction that also shows links between the collaborating objects.*

*This section is OPTIONAL because it could be described in Low Level Design document.]*

## Class Diagram

*[A diagram that shows classes with their attributes and operations together with the association between classes*

*This section is OPTIONAL because it could be described in Low Level Design documents.]*

# Process View

*[This section describes the system's decomposition into lightweight processes (single threads of control) and heavyweight processes (groupings of lightweight processes). Organize the section by groups of processes that communicate or interact. Describe the main modes of communication between processes, such as message passing, interrupts, and rendezvous.*

*This view is OPTIONAL. Use this view only if the system has more than one thread of control, and the separate threads interact or are dependent upon one another]*

# Deployment View

*[This section describes one or more physical network (hardware) configurations on which the software is deployed and run. It is a view of the Deployment Model. At a minimum for each configuration it should indicate the physical nodes (computers, CPUs) that execute the software, and their interconnections (bus, LAN, point-to-point, and so on.) Also include a mapping of the processes of the* ***Process View*** *onto the physical nodes.*

*This view is OPTIONAL. Use this view only if the system is distributed across more than one node. Even in these cases, only use the deployment view where the distribution has architectural implications. For example, in cases where there is a single server and many clients, a deployment view only needed to delineate the responsibilities of the server and the clients as a class of nodes; there is no need to show every client node if they all have the same capabilities.]*

# Implementation View

*[This section describes the overall structure of the implementation model, the decomposition of the software into layers and subsystems in the implementation model, and any architecturally significant components.*

*This view is OPTIONAL. Use this view only in cases where the implementation is not strictly driven from the design, i.e. where there is a different distribution of responsibilities between corresponding packages in the Design and Implementation models. If the packaging of the design and implementation models are identical, this view can be omitted.]*

## Overview

*[This subsection names and defines the various layers and their contents, the rules that govern the inclusion to a given layer, and the boundaries between layers. Include a component diagram that shows the relations between layers. ]*

## Layers

*[For each layer, include a subsection with its name, an enumeration of the subsystems located in the layer, and a component diagram.]*

# Data View

*[A description of the persistent data storage perspective of the system*

*This view is OPTIONAL if there is little or no persistent data, or the translation between the Design Model and the Data Model is trivial.]*

# Size and Performance

*[A description of the major dimensioning characteristics of the software that impact the architecture, as well as the target performance constraints.]*

# Quality

*[A description of how the software architecture contributes to all capabilities (other than functionality) of the system: extensibility, reliability, portability, and so on. If these characteristics have special significance, for example safety, security or privacy implications, they should be clearly delineated.]*

# Flow of Events – Design

*[A textual description of how the use case is realized in terms of collaborating objects. Its main purpose is to summarize the diagrams connected to the use case and to explain how they are related.*

*This section could and should be described in Use-Case Realization Specification document. Refer to TEMP-Use Case Realization.doc.]*

## Main Flow

*[The sequence of normal events that take place in the execution of events.]*

## Alternative Flow

*[The description of any exceptional flow.]*

## Derived Requirements

*[A textual description that collects all requirements, such as non-functional requirements, on the use-case realizations that are not considered in the design model, but that need to be taken care of during implementation.]*

# Business Rules

*[This section is used to define terminology specific to the problem domain, explaining terms which may be unfamiliar to the reader of the use-case descriptions or other project documents. Often, this section can be used as an informal data dictionary, capturing data definitions so that use-case descriptions and other project documents can focus on what the system must do with the information.]*

# Interface Design

*[This section could and should be described in another document such as User Interface Design or Use Case.]*

# 15. Database Design Description

# 16. Reference

*[This subsection should provide a complete list of all documents referenced elsewhere in the* ***Software Architecture Document****. Each document should be identified by title, report number (if applicable), date, and publishing organization. Specify the sources from which the references can be obtained. This information may be provided by reference to an appendix or to another document.]*

# 17. Appendix

*[List out the referenced information in the document.]*

# Revision

|  |  |  |  |
| --- | --- | --- | --- |
| # | Date | Editor | Description |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |