TADP 2-2009

Lucky Imaging CCD Camera Control Software Development Case

Version 1.0

Revision History

Date	Version	Description	Author
26/09/09	1.0	Initial Draft	TAPD 2 - 2009
27/09/09	1.01	Understanding Document	TAPD 2 - 2009
28/09/09	1.02	Fill the documentation	TAPD 2 - 2009

Table of Contents

Development Case

1. Introduction

Lucky Imaging CCD Camera Control Software is a project under TADP. The main purposes of the project are generating an implementation or interface for the camera and his integration with ACS as communication framework for sending data from the camera (images) to a certain client with scientist's ends.

1. Purpose

The purpose of the document is to describe the development process for the Lucky Imaging CCD Camera Control Software.

This document will cover the whole development case for the project, as the disciplines with his artifacts and the definitions of them; also the document applies to a sub-planning of the development plan as a kind of guideline, later the project execution will be follow the "statements" of the Development Case project document..

More information how the Development Case Document works and his functionalities can be found in the Rational Unified Process, Development Case Guidelines [1].

2. Scope

The scope of this document involves the "Lucky Imaging CCD Camera Control Software" only, during its initial and future development. The detail of each individual iteration of the project will be provided as an independent document, also will be cover only the phases with the disciplines and his artifacts associated.

3. Definitions, Acronyms, and Abbreviations

Some words and abbreviations in this document are expressed to refer technologies and frameworks.

- **CCD** : Charge-coupled device [2], refers to the way that the image signal is read out from the chip, mostly used in digital cameras.

- **ACS** : ALMA Common Software [3], provides a software infrastructure based on Distributed Objects common to all partners and consists of a documented collection of common patterns in and of components, which implement those patterns.

- **TADP** : Course "Topicos Avanzados de Diseño y Programación" (Advanced Topics of Desing and Programming).

4. References

[1] http://www.ts.mah.se/RUP/RationalUnifiedProcess/webtmpl/templates/environ/wb_dvlcs.htm
[2] http://en.wikipedia.org/wiki/Charge-coupled_device
[3] http://www.eso.org/~almamgr/AlmaAcs
[4] http://www.ast.cam.ac.uk/~optics/Lucky_Web_Site/
[5] http://www.ambysoft.com/unifiedprocess/aup11/html/disciplines.html

5. Overview

The document is organized by disciplines, and each discipline has his own artifacts with the description and the review for the project phases according to the Rational Unified Process.

Also this documents contains the roles that each person involved in this project are, such as developers, external clients and internal clients, lifecycle project and project reports.

2. Overview of the Development Case

1. Lifecycle Model

The development of the Project will be based on phases with one or more iterations in each one of them. These phases are Inception, Elaboration, Construction and Transition, each represents a milestone.

Phase	**Description Estas descripciones huelen a modelo en cascada! -Jaime Pavlich-Mariscal 10/6/09 9:34 AM**
Inception Phase	In this phase will be specify the project requirements, a draft of the iterations and a guideline to the development planning.
Elaboration Phase	Once the requirements are specified, a prototype of the software will be drafted. At the end of this phase

Construction Phase	Analisys and design it's has to be finished to construct the software.
Transition Phase	Documentation project and releases end this phase.

2. Disciplines

Each phase, has his own disciplines. The explanation and definition of each discipline used in the development case can be found in the Agile Unified Process Disciplines [5].

3. Discipline Configuration

The purpose of this section is to explain how the discipline configuration works. This includes an explanation of the purpose for the various tables and for each of the sections that describe the various disciplines listed in the section titled Disciplines.
In the tables of the section, each discipline have one or more artifacts that have a different uses in each phase, in example the artifact Functional Test Cases it is only used in Construction phase and reviewed by a external stakeholders. Also the project reports configuration and the artifacts that are not used with the reason why.

1. *Workflow*

The project workflow it is defined by the disciplines with his artifacts, in some cases represented with one or more iterations. To know what are the artifacts in each disciplines, are specified in the next point of this section.

2. *Artifacts No usar términos ambiguos, como "Should". Utilizar "C" para artefactos que son Creados en una etapa y "R" para artefactos que son refinados en una etapa. Dejar un espacio en las etapas donde el artefacto no es utilizado -Jaime Pavlich-Mariscal 10/6/09 9:34 AM*

Artifacts	How to Use				Review
	Incep	Elab	Const	Trans	Details
Requirements					
Glossary / Requirement	Must	Could	Could	Could	Internal-Formal
Design / Analysis					
Design	Won't	Must	Could	Could	Internal-Formal
Data Model	Won't	Must	Could	Could	Internal-Formal
Implementation					
User prototypes interface	Won't	Should	Must	Could	External-Formal
Configuration and Change Managment	Must	Must	Must	Must	Internal-Formal

Project Management					
Software Development Case	Must	Must	Must	Must	Internal-Formal
Software Requirements Specification	Must	Must	Could	Could	Internal-Formal
Software Development Plan	Must	Must	Must	Must	Internal-Formal
Test					
Functional Test Cases	Won't	Could	Should	Could	Internal-Formal
Enviroment	Must	Could	Could	Could	Informal

1. Explanation of the table

Column Name	Purpose	Contents/Comments
Artifacts	The name of the artifact	
How to Use	Qualify how the artifact is used across the lifecycle	Decide for each phase: • Must have • Should have • Could have • Won't have These are defined in artifacts classification section.
Review Details	Define the review level, and review procedures to be applied to the artifact.	Decide on the review level: • Formal-External • Formal-Internal • Informal • None For **details see** review procedures section.

3. *Reports*

Report planning for the project:

Reports	How to Use	Templates/Examples
Development project report	Must	Meeting - Week

4. *Notes on the Reports*

Table with the reports that will be not use in the projects:

Reports	How to Use	Reason
Development	Won't	Small project
Testing	Won't	Small project

4. Artifact Classification

An artifact is a deliverable of the process. It is often developed within one core workflow, although there are exceptions. The artifacts are organized in the workflow where they are created. To describe how an artifact will be used, we use the following classification scheme .

- **Must** : You must use this artifact. It is a key artifact and may cause problems later in development if it's not produced.
- **Should** : You should have this artifact, if at all possible, but it is negotiable. If you do not produce this artifact, you should be able to justify why not.
- **Could** : Could have means that this artifact doesn't have to be produced. It's only produced if it adds value and if there's enough time.
- **Won't** : This means you won't use this artifact. This may occur where a Rational Unified Process artifact is replaced by a local artifact.

5. Review Procedures

The project uses the following review levels:

- **Formal-External** : This artifact is part of the delivery at a specific milestone and requires some form of approval by the customer, the sponsor or some other external stakeholder.
- **Formal-Internal** : This artifact is formally reviewed internally by the project developers.
- **Informal** : This artifact is reviewed, but not formally approved.

For details see Guidelines: Review Levels.

6. Sample Iteration Plans

Phase	Iterations
Inception	2
Elaboration	1
Construction	3
Transition	1

5. Roles

Person	Rational Unified Process Role
Jaime Pavlich	Project Manager Deployment Manager Requirements Reviewer Architecture Reviewer

Miguel Ortiz	Configuration Manager Change Control Manager System Analyst Requirements Specifier User Interface Designer Designer Implementer Code Reviewer Integrator Test Designer Tester Technical Writer
Mario Aguilera	System Analyst Requirements Specifier User Interface Designer Designer Implementer Code Reviewer Integrator Test Designer Tester Technical Writer
Luis Gordillo	System Analyst Requirements Specifier User Interface Designer Designer Implementer Code Reviewer Integrator Test Designer Tester Technical Writer
Alexis Tejeda	System Analyst Requirements Specifier User Interface Designer Designer Implementer Code Reviewer Integrator Test Designer Tester Technical Writer System Analyst Requirements Specifier User Interface Designer

Francisco Ramirez	Designer Implementer Code Reviewer Integrator Test Designer Tester Technical Writer