| | |
|---|---|
| Acronym | Acamo |
| Project | **Active Aircraft Monitor** |
| Doctype | **Requirements** |
| Author | **Hans-Gerhard Gross, Kai Warendorf** |
| Contact | hggross@hs-esslingen.de;<br>Kai.Warendorf@hs-esslingen.de |
| Client | Esslingen University |
| Contact | Faculty of Information Technology |
| Version | 1.0 |
| Date | March 2, 2015 |

# Contents

# Chapter 1

# Project Drivers

## 1.1 Purpose of the Project

### 1.1.1 Vision Statement

This project aims at developing an application that shows the active aircraft in range of the ADS-B receiver.

### 1.1.2 Project Outcomes

The Java application reads ADS-B messages.

The Java application decodes ADS-B messages.

The Java application transforms ADS-B message data into aircraft data.

The Java application displays decoded message data and aircraft data.

### 1.1.3 Learning Objectives

After having completed this project, as student, you can ...

- develop and integrate Java classes and interfaces.
- identify and solve domain problems through advanced Java programming.
- develop rudimentary graphical user interfaces with Java.

## 1.2 Stakeholders

### 1.2.1 Project Team

Various members and roles.

### 1.2.2 Product Users

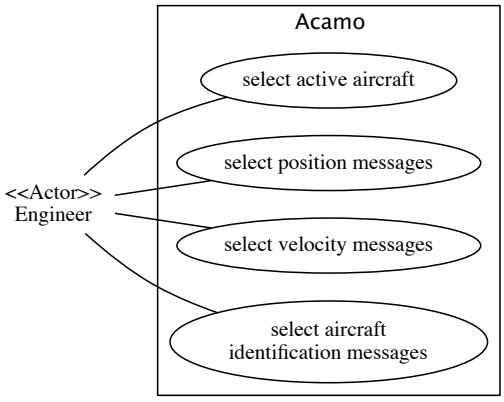
**Local Flight Control Engineer, User.** Priority: **Key User**.

# Chapter 2

# Functional Requirements

## 2.1 Data Model and Data Dictionary

### 2.1.1 Use Case Diagram



## 2.2 Acamo Functional Requirements

### Acamo.F.10 Select Active Aircraft essential

**Feature** In order to get an overview of the local flight traffic, as a flight control engineer, I want to be able to observe the aircraft that are currently active.

*Scenario*

*Given* the application does not show the active aircraft

*When* I select to observe the active aircraft

*Then* the application should show the active aircraft

Active aircraft send any arbitrary new ADS-B message within a time span of 4 minutes.

*Scenario*

*Given* an ADS-B message for aircraft ICAO 02A1A2 at time $t$

*When* an ADS-B message is received from aircraft 02A1A2, and the current time is less than or equal to $t + 240$ sec

*Then* aircraft 02A1A2 is an active aircraft, and shall be shown in the list of active aircraft

*Scenario*

*Given* an ADS-B message for aircraft ICAO 02A1A2 at time $t$

*When* no more ADS-B messages are received for ICAO 02A1A2 after $t + 240$ sec

*Then* aircraft 02A1A2 is an inactive aircraft, and shall be removed from the list of active aircraft

**Feature** Each active aircraft shall be shown with the following information:

- ICAO of the aircraft
- Timestamp of the last activity
- Call Sign of the aircraft if available
- Most recent 3-dimensional position with latitude, longitude and altitude
- Most recent velocity with horizontal and vertical speeds, and heading

**Feature** The application shall show the active aircraft upon application startup.

*Scenario*

*Given* the application is off

*When* I start the application

*Then* the application should show the active aircraft

### Acamo.F.20 Select Position Messages essential

**Feature** In order to get an overview of the local flight traffic, as a flight control engineer, I want to be able to observe the decoded incoming position messages.

*Scenario*

*Given* the application does not show the position messages

*When* I select to observe the position messages

*Then* the application should show the position messages

**Feature** Each position message shall be shown with the following decoded information:

- ICAO of the aircraft
- CPR format
- CPR encoded latitude
- CPR encoded longitude
- Altitude

### Acamo.F.30 Select Velocity Messages essential

**Feature** In order to get an overview of the local flight traffic, as a flight control engineer, I want to be able to observe the decoded incoming velocity messages.

*Scenario*

*Given* the application does not show the velocity messages

*When* I select to observe the velocity messages

*Then* the application should show the velocity messages

**Feature** Each velocity message shall be shown with the following decoded information:

- ICAO of the aircraft
- Horizontal speed
- Vertical speed
- Heading

### Acamo.F.40 Select Aircraft Identification Messages essential

**Feature** In order to get an overview of the local flight traffic, as a flight control engineer, I want to be able to observe the decoded incoming aircraft identification messages.

*Scenario*

*Given* the application does not show the aircraft identification messages

*When* I select to observe the aircraft identification messages

*Then* the application should show the aircraft identification messages

**Feature** Each aircraft identification message shall be shown with the following decoded information

- ICAO of the aircraft
- Call Sign of the aircraft

# Chapter 3

# Non-Functional Requirements

## 3.1 Look and Feel Requirements

**Acamo.NF.10 Graphical User Interface (GUI)** essential

**Feature** The application user interface shall be realized as graphical user interface.

**Feature** The GUI window shall be organized in terms of four tabbed panes, one for each of the previously defined use cases.

## 3.2 Performance Requirements

**Acamo.NF.20 Timing** essential

**Feature** The list of active aircraft shall be updated at least once per second.

## 3.3 Implementation-Specific Requirements

### 3.3.1 Process

**Acamo.NF.50 Test Driven Development** **essential**

In order to ascertain sufficient testing of the product, the implementation must be carried out following a test-driven development approach.

## 3.4 Maintainability Requirements

**Acamo.NF.70 Documentation** **essential**

In order to ascertain high understandability, the source code must be self-explanatory.

**Acamo.NF.80 Cohesion and Coupling** **essential**

In order to support high maintainability, the modules of the system must be realized with high-cohesion and low coupling.

**Acamo.NF.90 OO Design Principles** **essential**

In order to support high maintainability, the other well-known principles of good object-oriented design must also be applied.