Karoshi

Software Requirements Specification

Context Aware Map Application (Sponsored by ASELSAN)

Fatma AKINCI – 1630540
İlker ARGIN – 1559897
İsmail Can COŞKUNER – 1560036
Meryem SAĞCAN -1560499

12/5/2010

1. Introduction

This Software Requirements Specification report is prepared by “Karoshi” group members to determine Map Application with Context Aware User Interface project requirements.

1.1 Problem Definition

Today most of the mobile devices have a standard user interface, which is a big problem for the users of these devices. They cannot respond to environmental changes or movement of the user. One may have difficulties in seeing the screen content and using the device when light conditions change or she/he moves.

Current map viewer applications used in military have static buttons, menus and texts whose fonts do not change. The colors of the map shown on the screen do not change, neither. This is problematic for the soldiers since they are often in motion and occur to be in places with different light conditions. This project will be a solution to non-dynamic user interfaced map applications used for military purposes.

1.2 Purpose

This SRS aims to provide a complete description of all the functions and specifications of Map Application with Context Aware User Interface project sponsored by ASELSAN. This document includes general information about the project, use cases, functions, features, special technologies used in the project, and what the application needs to work properly and with safety. In addition, it will include some literature research results regarding the project.

The intended audience for this document includes all users and prospective software developers associated with the project. Therefore, this document will function as a guide through each phase of the project.

1.3 Scope

When the user runs this map application, all the buttons and the map will be shown on the mobile device screen with the default sizes and default colors. After

that necessary computations will be done and screen content will change accordingly.

If the user starts to move, only the map will be shown, information bar and buttons will be hidden in the background. If the user clicks on the map, information bar and buttons that are modified according to their priority levels will appear again on the screen. However, their emergence on the screen will not have an effect on the map visibility. The modification includes changing size of the buttons, fonts of the texts and removing less important parts of the map and increasing the zoom level of the map. If the user slows down or stops, all the modification will be undone.

If the user goes to a place with a different illumination, the colors will be changed to increase the visibility of all the contents on the screen.

By these properties, the product will benefit users with easy usage of mobile devices.

1.4 User and Literature Survey

1.4.1 Users of the Project

This project was first intended to be used for military purposes. This software can be integrated to a large variety of mobile devices used in military. Soldiers will be able to use it in all areas where the environment conditions is not appropriate for comfortably using the device. The final product may be useful for personal purposes.

1.4.2 Literature Survey

Based on the literature research conducted, “Context is the set of environmental states and settings that either determines an application’s behavior or in which an application event occurs and is interesting to the user.”[1] According to the definition, the context can be discussed in three different approaches:

- **User Context:** Users’ movement, users’ location, people nearby the users, objects nearby the users or even the psychological situation of the users can be counted as user context.

- **Time Context:** Time of the day, week, month and season can all be discussed under time context umbrella.
- **Physical context:** Lighting of the environment, noise level in the environment, temperature of the weather and traffic density in the environment are all considered as physical context change.

All the worldwide studies to change the user interface of the mobile devices were conducted according to the context changes mentioned above. Some of the studies related with context aware user interface:

- ✓ **Nokia Research Center and Waseda University collaboration [2]**

  They have managed to implement text and image viewing application that responds to users' movement on a platform called Muffin.

- ✓ **Nokia [3]**

  The idea of Nokia focuses more on input manipulation rather than output manipulation. In other words, Nokia uses sensor technology to reduce input errors while moving, configuring system accordingly. What we are trying to is configuring output to improve readability.

- ✓ **Technology for Enabling Awareness [1]**

  Adaptive GSM phone and PDA: This product is designed to change the user interface of these devices by using user's activity, light level, pressure, and proximity of other people contexts. In the PDA scenario a notepad application was adapted to change by user's movement (large font when the user is walking) in the phone scenario, the profiles of the mobile phone are selected automatically. The phone chooses to ring, vibrate, adjust the ring volume, or keep silent, depending on whether the phone is in hand, on a table, in a suitcase, or outside. This project seems to have a lot of similarities with the project which is the subject of this SRS.

- ✓ **Paper with the title "Multi-sensor context-awareness" [4]**

The paper discusses the usage of single generic context-sensors such as camera and location sensors versus integration of multiple diverse sensors. Position sensors provide access to location's particular characteristics. Position is a static description of an environment and does not capture dynamic aspects of a situation. Because of this, its reliability depends on the previously obtained knowledge about locations. Cameras provide access to potentially rich information that can be derived by computer vision techniques. By usage of vision techniques can be employed to capture activity and other

dynamic aspects, but extraction of septic context is computationally expensive and problematic in mobile devices. On the other hand, paper suggests multi-sensor context-awareness as an alternative approach toward aware mobile devices. In this approach, the single powerful sensor is replaced by a collection of diverse simple sensors, and context is derived from multi-sensor data. This approach was first introduced in the European project TEA on mobile situation awareness, with the aim to provide comparatively powerful cheap technology both with respect to processing requirements and component cost. The research prototype integrated sensors for motion, orientation, light, and temperature, and relates to work with its emphasis on small, lightweight, low-power and cheap components. Separation of sensors also means that both sensors and feature extraction methods can be developed and replaced independently of each other.

1.5 Definitions and Abbreviations

SRS: Software Requirements Specification
Wi-Fi: Wireless Fidelity
CPU: Central Processing Unit
Java ME: Java Micro Edition
SQL: Structural Query Language
GPS: Global Positioning System
SDK: System Development Kit
JRE: Java Runtime Environment

CLDC: Connected Limited Device Configuration

MIDP: Mobile Information Device Profile

FP: Function Points

KLOC: Kilo Lines of Code

1.6 References

[1] Chen, G., & Kotz, D. (2000). A Survey of Context-Aware Mobile Computing Research. Dartmouth College, Department of Computer Science.

[2] Tetsuo, Y., & Takahashi, K. (2007, October). Experiments in Mobile User Interface Adaptation. Intelligent Pervasive Computing, 280-284.

[3] Nurmi, M. (2008). Adaptive user interface input device. U.S: Nokia Corporation.

[4] Gellersen, H. W., Schmidt, A., & Beigl, M. (2002, October). Multi-sensor context-awareness in mobile devices and smart artifacts. Mobile Networks and Applications, 7(5).

[5] IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications

1.7 Overview

This SRS document consists of seven parts which includes:

Section 1: *Introduction of SRS*

Section 2: *Overall description of the project. In this section, details are avoided and providing understanding of the overall project is aimed.*

Section 3: *In this section modules and interfaces are explained in detail.*

Section 4: *This section contains information about data models, relations with each other and data flow.*

Section 5: *Behavioral design which includes state transitions.*

Section 6: *Planning of the project, distribution of group members to specific parts of the project, estimation and process model.*

Section 7: *Conclusion of SRS.*

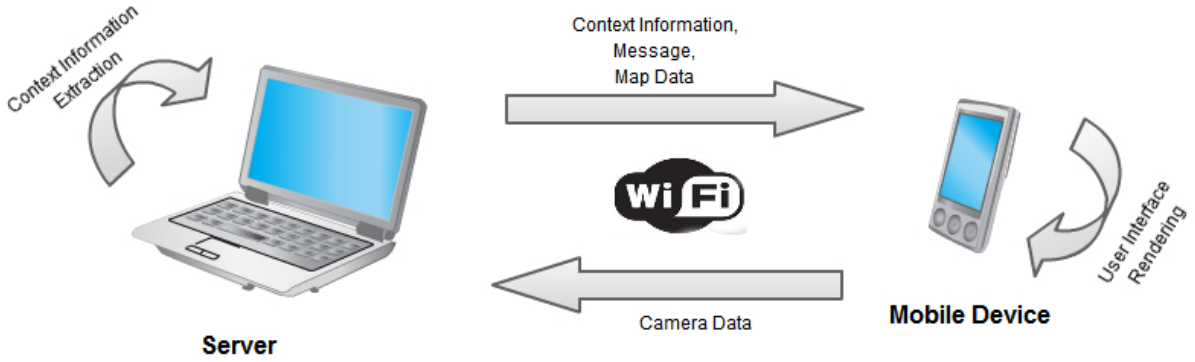
2 Overall Description

2.1 Product Perspective

This project has two main parts. In the first part of the project, a mobile phone will be used to run the map application and reveal the changes in the user interface according to the information coming from the server. The connection between the server and the mobile phone will be provided via wireless.

Since mobile devices do not have a powerful CPU, they cannot handle complicated image manipulating processes. Therefore a server computer is needed to process the frames coming from mobile devices since the CPU of a computer is powerful enough. Hence, the second part of the project will be developed on a computer which will be used as a server doing some image processing operations and will have a database containing information about the places.

The overall system is shown in the diagram below:



2.1.1 Database Module

Map images, coordinates and type information about places and messages to assign a task to the soldier are kept in the database. Therefore, server will use this module to extract requested information from the database and send it to the mobile device.

2.1.2 Computer Vision Module

This module is again will be in the server part. It will extract light information and user's motion information from the frames that are sent by the mobile device.

2.1.3 Application

It will work on the mobile device. It determines what the buttons do and how information bar works just like in the static user interfaced map applications.

2.1.4 Context Manager

This part contains properties that apart this application from recently used map applications. It arranges the visibility of the buttons, the information bar, and objects on the map according to the context information from the server.

2.1.5 Server-Side Wireless Communication Module

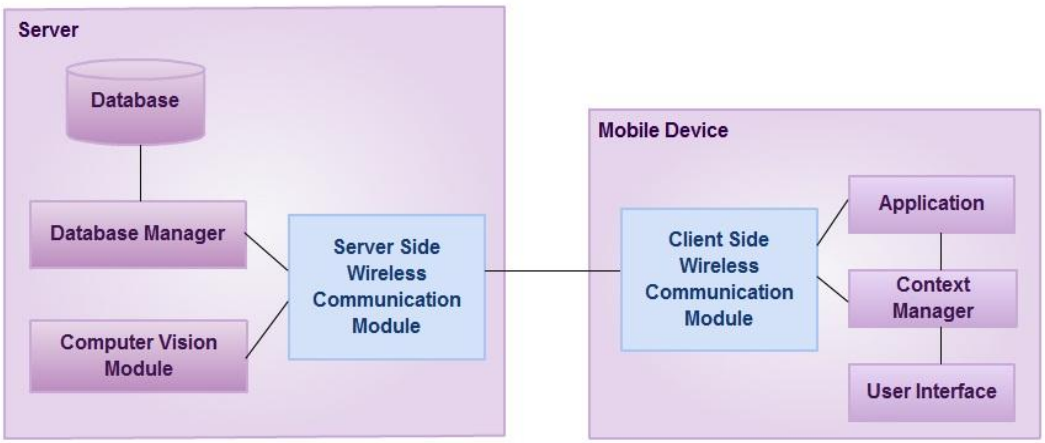
It provides the communication between server and client-side wireless communication module. The information extracted by computer vision module from the frames that come from the client, messages selected by the server-side user to assign a task to the client-side user, map images and information of the objects in the map will be sent to client-side wireless communication module. Another function of this module is to get the information from client-side wireless communication module and give it to the server.

2.1.6 Client-Side Wireless Communication Module

It supplies the connection between mobile device and the server-side wireless communication module. The frames taken from the camera of the mobile device will be sent to the server-side wireless communication module. In addition, this module will get the information from server-side wireless communication module.

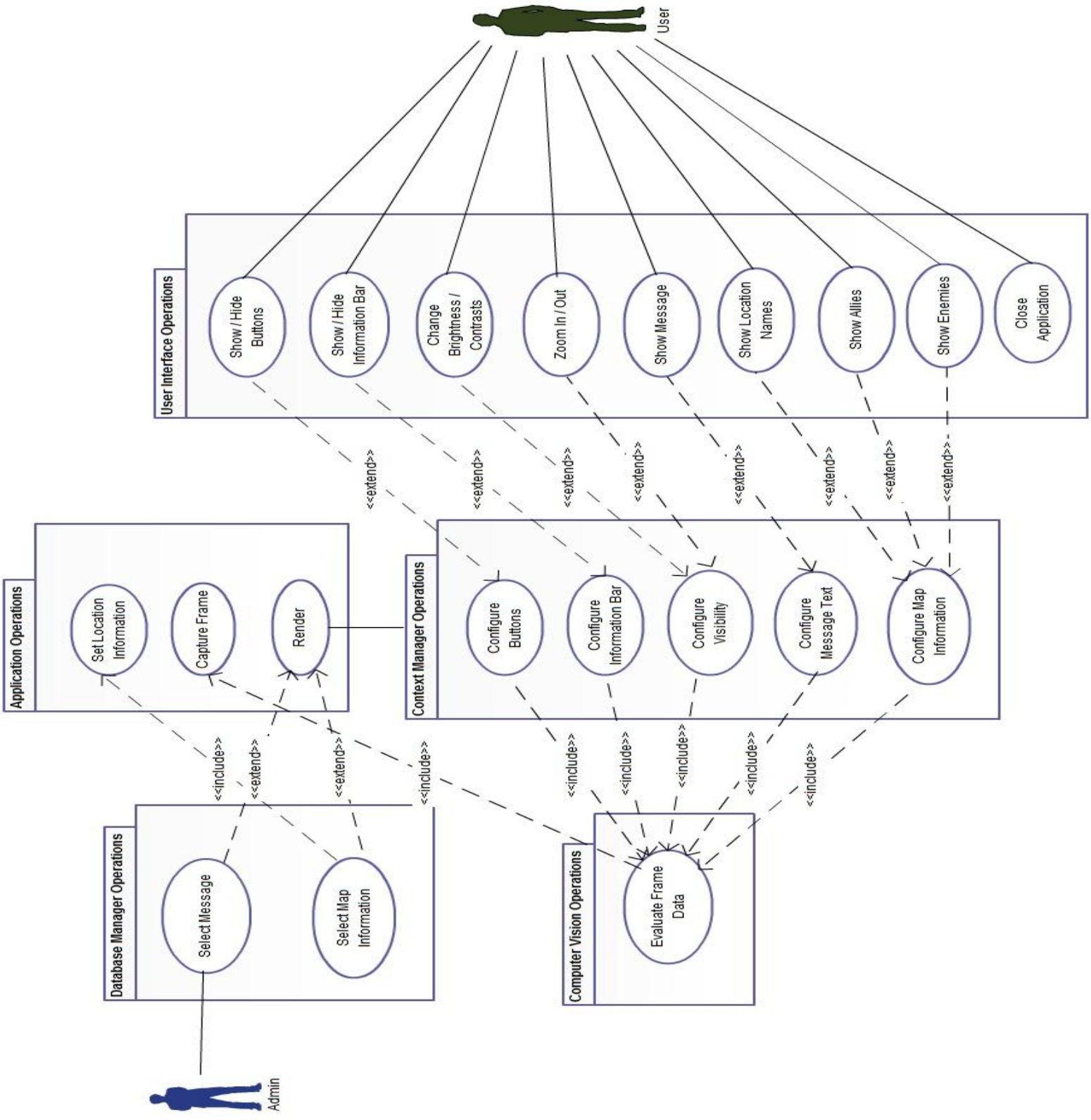
2.1.7 User Interface Module:

This module provides user interaction with core functions of the application. It transfers user preferences to context manager module.



2.2 Product Functions

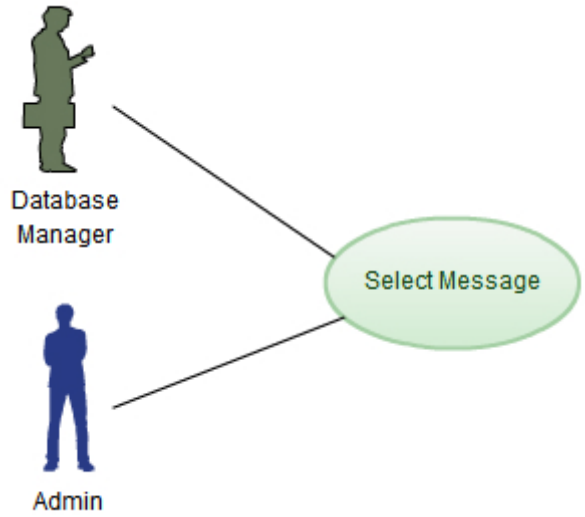
This section outlines the use cases for each of the actors separately and contains brief descriptions of these use cases. The use case diagram of the overall system is also shown below:



2.2.1 Database Manager Operations

2.2.1.1 Use Case: Select Message

Diagram:



Brief Description:

This is the function that works for sending the message specified by the user on the server side to the remote user who has the mobile device. It communicates with the database because messages are kept in the database.

2.2.1.2 Use Case: Select Map Information

Diagram:

Brief Description:

Since the maps of territories are kept in the database, with the help of this function and coordinate information of the user, related map image and information of the objects (coordinates of the enemy troops, ally troops in that territory, etc.) to be drawn on the map is taken from the database.

2.2.2 Computer Vision Operations

2.2.2.1 Use Case: Evaluate Frame Data

Diagram:

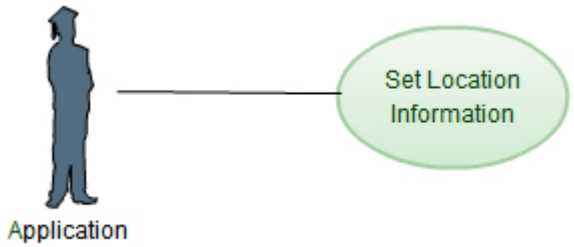
Brief Description:

This function determines the light level of the environment that user is in and whether the user is in motion or not. It does this by processing frames coming from the wireless communication modules. Those frames are captured with camera of the mobile device.

2.2.3 Application Operations

2.2.3.1 Use Case: Set Location Information

Diagram:

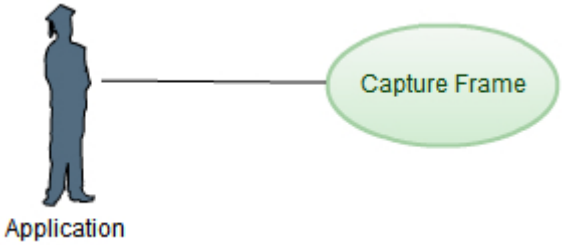


Brief Description:

This function takes the GPS information from the mobile device and necessary coordinates to send them to database manager.

2.2.3.2 Use Case: Capture Frame

Diagram:



Brief Description:

This function works to get frames from the video buffer which is constantly being recorded by the camera of the mobile device.

2.2.3.3 Use Case: Render

Diagram:

Brief Description:

When the application first starts, all the graphical user interface is drawn by this function. This user interface includes the map, objects on the map, buttons, if exists text messages and information bar. If the user get into the motion or the light conditions of the environment that user changes, this function becomes active and redraws the screen contents based on the values coming from the context manager module.

2.2.4 Context Manager Operations

2.2.4.1 Use Case: Configure Buttons

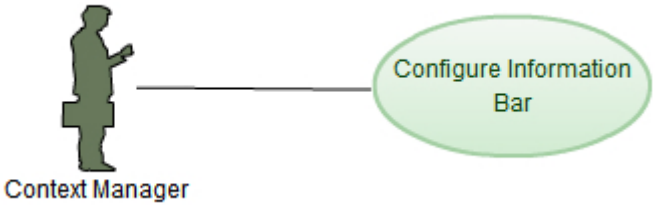
Diagram:

Brief Description:

According to the motion information, size information of the buttons and information about which buttons need to be shown are determined. These new results are used by the application to draw them accordingly.

2.2.4.2 Use Case: Configure Information Bar

Diagram:



Brief Description:

When the user moves, this function configures the size of the texts in the information bar. Amount of information is reduced considering their priority.

2.2.4.3 Use Case: Configure Visibility

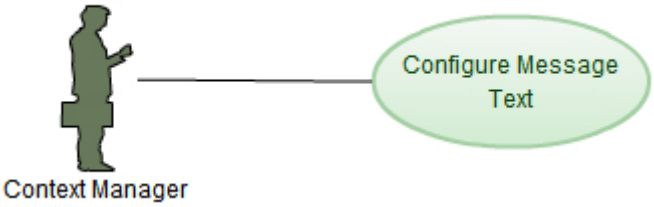
Diagram:

Brief Description:

For the user to gain maximum amount of information with minimum attention, the visibility of screen needs to be changed. This function arranges the information of the color of the map and brightness/contrast of the screen. In addition the zoom level of the map is changed via this function.

2.2.4.4 Use Case: Configure Message Text

Diagram:



Brief Description:

The server-side user is expected to type some part of the messages, which is considered to be emergent, in different color. According to the emergency of the parts this message text is configured if the user is in motion.

2.2.4.5 Use Case: Configure Map Information

Diagram:

Brief Description:

This function decides on which location names and what type of objects will be visible to the user.

2.2.5 User Interface Operations

2.2.5.1 Use Case: Show/Hide Buttons

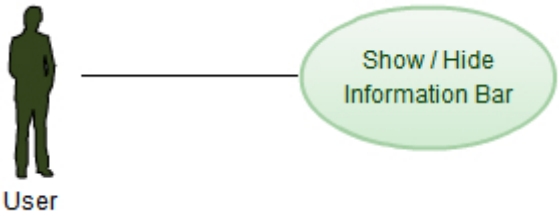
Diagram:

Brief Description:

In default screen the buttons are hidden in the background. This function works for showing the buttons when requested and then hiding them after a while.

2.2.5.2 Use Case: Show/Hide Information Bar

Diagram:



Brief Description:

This function shows the information bar which contains information about an object (places, enemies, etc.) in the map. The information bar is hidden if the user holds and drags it.

2.2.5.3 Use Case: Change Brightness and Contrast

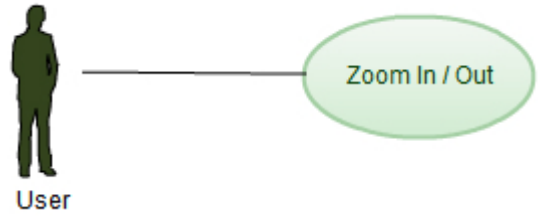
Diagram:

Brief Description:

The user is capable of changing these properties manually. In this case, this function modifies the value of these properties to apply the user's request.

2.2.5.4 Use Case: Zoom In/Out

Diagram:

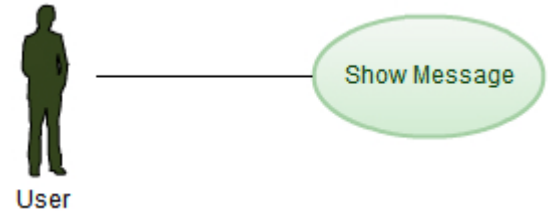


Brief Description:

This function is responsible for setting zoom level of the map based on the request from the user.

2.2.5.5 Use Case: Show Message

Diagram:

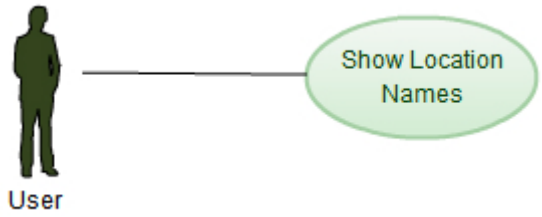


Brief Description:

This function, with the request of the user, shows the messages which is in the inbox of the mobile device.

2.2.5.6 Use Case: Show Location Names

Diagram:



Brief Description:

The user sometimes wants to see the names of the locations on the map. This function reveal the names of the locations when the user requests.

2.2.5.7 Use Case: Show Allies

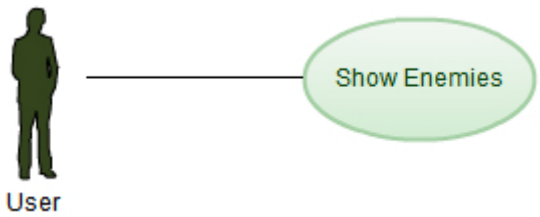
Diagram:

Brief Description:

This function shows the ally troop objects on the map according to the user demand.

2.2.5.8 Use Case: Show Enemies

Diagram:

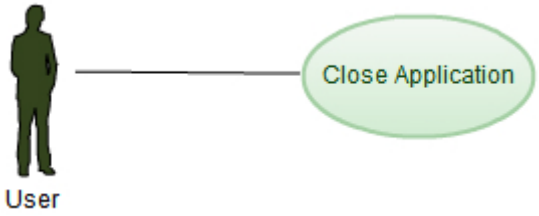


Brief Description:

This function shows the enemy troop objects on the map according to the user demand.

2.2.5.9 Use Case: Close Application

Diagram:



Brief Description:

User closes the application.

2.3 Constraints, Assumptions and Dependencies

- Mobile device has GPS module and camera.
- Java platform exists in the mobile device.
- Maximum velocity is achieved when there is no relation among consecutive frame data.
- Minimum velocity is achieved when there is at most 5% difference among consecutive frame data.
- Network connection should handle frame acquisition in reasonable time.
- Camera is active and recording video all the time.

3 Specific Requirements

3.1 Interface Requirements

3.1.1 User Interface

There are buttons in the user interface to communicate with the user. These buttons are hidden in the background. When the user wants to use the buttons,

he/she needs to click on the arrowhead at the bottom of the screen. The buttons will disappear again after a while if there is no action to be done.

The buttons and related functions:

- **Change Brightness/Contrast:** When the user clicks on this button, product function 2.2.5.3 **Change Brightness/Contrast** function is called.
- **Message Box Button:** When the user clicks on this button, product function 2.2.5.5 **Show Message function** is called.
- **Location Names On/Off:** When the user clicks on this button, location names on the map are shown. In the second click, location names disappear. The related function is **Show Location Names** from product function 2.2.5.6.
- **Allies:** When the user clicks on this button, ally troop objects on the map are shown. In the second click, location names disappear. The related function is **Show Allies** from product function 2.2.5.7.
- **Enemies:** When the user clicks on this button, enemy troop objects on the map are shown. In the second click, location names disappear. The related function is **Show Enemies** from product function 2.2.5.8.
- **Exit button**, when clicked, **Close Application** function is called.

If user clicks on a place on the map, information bar appears and shows the information about that place. Related function is **Show/Hide Information Bar** from product function 2.2.5.2.

In addition, there is up/down arrowhead which helps the user with zooming in/out the map. These arrowheads do not interfere with the visibility of the map.

Buttons will be arranged according to their assigned priorities if the user is moving. Some buttons will be grouped under a single button. The others, which have higher priorities, will grow in size to be easily seen and used by the user. This task is done by 2.2.4.3. **Configure Visibility** function.

Colors of the screen contents are changed if the light of the environment changes. According to the illumination of the environment, appropriate colors are chosen for the map to be more visible. The function responsible for this action is again **Configure Visibility**.

3.1.2 Hardware Interface

There will be a communication with wireless network internet card both in the server part and mobile device part. There are 6 ports to send/get all necessary data for the system to properly work. The functions in sections 3.2.6 and 3.2.7 explain how the system communicates with wireless network.

Mobile device is required to have a camera.

3.1.3 Software Interface

As a database management system, MySQL 5.1 is going to be used on server side. Map images, location information of places, messages, and object information are kept in this database. SQL is going to be used to interact with database.

On the server side, for image processing tasks MATLAB R2010b will be used. Operating system of the server computer is Ubuntu 10.10.

On the mobile device side, Java ME SDK 3.0 is required to develop the application. As for the operating system, any operating system supporting Java ME is acceptable.

3.2 Functional Requirements

This section includes each major software function along with data flow and requirements of the function.

3.2.1 Database Manager Module

3.2.1.1 Select Message

Use Case Name	Select Message

Trigger	Admin selects message from database.
Precondition	Admin is on *Message Sending Screen* linked from Server Side Application Main Page.
Basic Path	1. Admin selects message template from Database. 2. Admin clicks send button.
Alternative Paths	If a connection error occurs, admin will be notified.
Postcondition	The message is forwarded to Server Communication Module.
Exception Paths	Admin can abandon the operation at any time.
Other	This use case extends 3.2.3.3, Render.

3.2.1.2 Select Map Information

Use Case Name	Select Map Information
Trigger	New map information is needed by client.
Precondition	Server has got the current GPS information of client.
Basic Path	1. Current GPS information is processed. 2. Corresponding map partition is prepared. 3. Information corresponding to this map partition is filtered.
Alternative Paths	1. Current GPS information is processed. 2. If client map information is up to date, do nothing.
Postcondition	The map information is forwarded to Server Communication Module.
Exception Paths	If GPS information is not valid, do nothing.
Other	Map information includes field image, information about locations and critical points. This use case extends 3.2.3.3, Render and includes 3.2.3.1, Show Map.

3.2.2 Computer Vision Module

3.2.2.1 Evaluate Frame Data

Use Case Name	Evaluate Frame Data
Trigger	Corresponding timer triggers.
Precondition	Frame data set is gathered from client.
Basic Path	1. Process frame data 2. Extract luminosity value 3. Extract movement value
Alternative Paths	None.
Postcondition	Luminosity and movement values are forwarded to Server Communication Module.
Exception Paths	If frame data is not valid, do nothing.
Other	Frame data is consecutive images taken from camera of the mobile device in a fixed interval. This use case includes 3.2.3.2, Capture Frame.

3.2.3 Application Module

3.2.3.1 Set Location Information

Use Case Name	Set Location Information
Trigger	Corresponding timer triggers.
Precondition	Mobile device connected to network.
Basic Path	Get location data using GPS.
Alternative Paths	None.
Postcondition	Location data is forwarded to Server Communication Module.
Exception Paths	If mobile device can not get GPS information, application exits with a warning message.
Other	None.

3.2.3.2 Capture Frame

Use Case Name	Capture Frame
Trigger	Correspondent timer triggers.
Precondition	Camera of the mobile device should be ready for image

	capturing.
Basic Path	Get image from camera.
Alternative Paths	None.
Postcondition	Image will be captured and forwarded to Client Communication Module.
Exception Paths	If camera is not available, application exists with a message.
Other	None.

3.2.3.3 Render

Use Case Name	Render
Trigger	Corresponding timer triggers.
Precondition	• There exists a map information on mobile device downloaded from server. • There exists consistent data supported from Context Manager Module.
Basic Path	1. Render map image. 2. Put locations on map image. 3. Put allies on map image. 4. Put enemies on map image. 5. Draw buttons. 6. Set colors according to schema. 7. Adjust brightness and contrast.
Alternative Paths	1. Follow basic path 2. If there is an incoming message, adjust it accordingly. 3. Show message on the map transparently.
Postcondition	Render executes until termination of the application.
Exception Paths	• If a map can not be found, exit from application with a message. • If data supported by Context Manager Module is not consistent, exit from application with a message.
Other	Information about locations, enemies, allies, buttons, color

	schema, brightness and contrast is supported by Context Manager Module. This module processes all data using context variables supported by server, and synthesize the information that will be used.

3.2.4 Context Manager Module

3.2.4.1 Configure Buttons

Use Case Name	Configure Buttons
Trigger	Luminosity and movement values are gathered from server.
Precondition	Luminosity and movement values are valid.
Basic Path	1. Resize buttons. 2. Change order of the buttons according to priority.
Alternative Paths	If buttons will be hidden, no operation takes place.
Postcondition	Button arrangments will be transferred to Application Module to be rendered.
Exception Paths	If luminosity and movement values are not valid, do not change current arrangement.
Other	This use case includes 3.2.2.1, Evaluate Frame Data.

3.2.4.2 Configure Information Bar

Use Case Name	Configure Information Bar
Trigger	• Luminosity and movement values are gathered from server. • User selects a legend on the map in order to view information about it.
Precondition	• Luminosity and movement values are valid. • There exists information about selected legend.

Basic Path	1. Get corresponding information about legend. 2. Decide which information about legend will be shown using movement value. 3. Decide how information will be shown using movement and luminosity values. 4. Show information in information box.
Alternative Paths	If hide information is selected, no operation takes place.
Postcondition	Information box will be transferred to Application Module to be rendered.
Exception Paths	• If luminosity and movement values are not valid, do not change current arrangement. • If no information exists about selected legend, do nothing.
Other	This use case includes 3.2.2.1, Evaluate Frame Data.

3.2.4.3 Configure Visibility

Use Case Name	Configure Visibility
Trigger	• Luminosity and movement values are gathered from server. • User changes brightness, contrast or zoom level manually.
Precondition	Luminosity and movement values are valid.
Basic Path	1. Change color schema using luminosity value. 2. Adjust brightness and contrast according to luminosity value. 3. Decide on zoom level using movement value.

Alternative Paths	User may adjust brightness, contrast or zoom level manually.
Postcondition	Color, brightness, contrast and zoom level values will be transferred to Application Module to be using in rendering.
Exception Paths	If luminosity and movement values are not valid, do not change current arrangement.
Other	This use case includes 3.2.2.1, Evaluate Frame Data.

3.2.4.4 Configure Message Text

Use Case Name	Configure Message Text
Trigger	• Luminosity and movement values are gathered from server. • User selects a message from inbox in order to view it.
Precondition	Luminosity and movement values are valid.
Basic Path	1. Get selected message from message list. 2. Decide how message will be shown using movement and luminosity values. 3. Show message in message box.
Alternative Paths	None.
Postcondition	Message box will be transferred to Application Module to be rendered.
Exception Paths	If luminosity and movement values are not valid, do not change current arrangement.
Other	This use case includes 3.2.2.1, Evaluate Frame Data.

3.2.4.5 Configure Map Information

Use Case Name	Configure Map Information
Trigger	• Movement value is gathered from server. • User changes a filter preference.

Precondition	Movement value is valid.
Basic Path	Filter general map information into specific map information using movement value and filter preferences.
Alternative Paths	None.
Postcondition	This specific map information will be transferred to Application Module to be rendered.
Exception Paths	If movement value is not valid, do not change previous evaluation.
Other	Filter preferences include show allies, show enemies and show locations. This use case includes 3.2.2.1, Evaluate Frame Data.

3.2.5 User Interface Module

3.2.5.1 Show / Hide Buttons

Use Case Name	Show / Hide Buttons
Trigger	• User uses arrowhead button. (Show) • Idle timer. (Hide)
Precondition	• To hide buttons, buttons should be on the screen. • To show buttons, buttons should be hidden except arrowhead button.
Basic Path	Change decision variable about showing buttons.
Alternative Paths	None.
Postcondition	Corresponding decision variable changes and Context Manager Module uses it while arranging buttons.
Exception Paths	None.
Other	This use case extends 3.2.4.1, Configure Buttons.

3.2.5.2 Show / Hide Information Bar

Use Case Name	Show / Hide Information Bar
Trigger	• User selects a legend on the map. (Show) • User holds and drags information box. (Hide)

Precondition	To hide information bar, information bar should be on the screen.
Basic Path	1. User selects a legend on the map. 2. Change decision variable. 3. Legend coordinates are sent to Context Manager Module.
Alternative Paths	When user holds and drags information box, it will become hidden changing corresponding decision variable.
Postcondition	Corresponding decision variable changes and Context Manager Module uses it while arranging map information.
Exception Paths	None.
Other	This use case extends 3.2.4.2, Configure Information Bar.

3.2.5.3 Change Brightness / Contrast

Use Case Name	Change Brightness / Contrast
Trigger	User uses adjust brightness / contrast button.
Precondition	Currently brightness and contrast values exist.
Basic Path	1. User selects adjust brightness / contrast button. 2. Adjust brightness / contrast screen appears. 3. User makes changes. 4. User clicks OK button. 5. New brightness and contrast values are sent to Context Manager Module.
Alternative Paths	User may cancel operation on any step.
Postcondition	Context Manager Module uses new brightness and contrast values while configuring visibility.
Exception Paths	None.
Other	This use case extends 3.2.4.3, Configure Visibility.

3.2.5.4 Zoom In / Out

Use Case Name	Zoom In / Out

Trigger	User interacts with zoom bar.
Precondition	Currently zoom value exists.
Basic Path	1. Get new user defined zoom value. 2. Zoom value is sent to Context Manager Module.
Alternative Paths	None.
Postcondition	Context Manager Module uses new zoom value while configuring visibility.
Exception Paths	None.
Other	This use case extends 3.2.4.3, Configure Visibility.

3.2.5.5 Show Message

Use Case Name	Show Message
Trigger	• User selects a message from inbox in order to view it. • A new message comes from server.
Precondition	User is on the main screen.
Basic Path	1. Select inbox button. 2. Select a message from inbox. 3. Forward selected message to Context Manager Module.
Alternative Paths	When a new message comes from server, it will be directed to the Context Manager Module.
Postcondition	Context Manager Module views message using configure message text operation.
Exception Paths	None.
Other	This use case extends 3.2.4.4, Configure Message Text.

3.2.5.6 Show / Hide Location Names

Use Case Name	Show / Hide Location Names
Trigger	User selects show / hide location name button.
Precondition	• To hide locations, locations should be on the screen. • To show locations, locations should be hidden.
Basic Path	Change decision variable about showing location names.

Alternative Paths	None.
Postcondition	Corresponding decision variable changes and Context Manager Module uses it while configuring map information.
Exception Paths	None.
Other	This use case extends 3.2.4.5, Configure Map Information.

3.2.5.7 Show / Hide Allies

Use Case Name	Show / Hide Allies
Trigger	User selects show / hide allies button.
Precondition	• To hide allies, allies should be on the screen. • To show allies, allies should be hidden.
Basic Path	Change decision variable about showing allies.
Alternative Paths	None.
Postcondition	Corresponding decision variable changes and Context Manager Module uses it while configuring map information.
Exception Paths	None.
Other	This use case extends 3.2.4.5, Configure Map Information.

3.2.5.8 Show / Hide Enemies

Use Case Name	Show / Hide Enemies
Trigger	User selects show / hide enemies button.
Precondition	• To hide enemies, enemies should be on the screen. • To show enemies, enemies should be hidden.
Basic Path	Change decision variable about showing enemies.
Alternative Paths	None.
Postcondition	Corresponding decision variable changes and Context Manager Module uses it while configuring map information.
Exception Paths	None.
Other	This use case extends 3.2.4.5, Configure Map Information.

3.2.5.9 Close Application

Use Case Name	Close Application
Trigger	User selects close application button.
Precondition	Application is running.
Basic Path	Quit application.
Alternative Paths	None.
Postcondition	Application terminates.
Exception Paths	None.
Other	None.

3.2.6 Server-side Communication Module

3.2.6.1 Get Location Data

Use Case Name	Get Location Data
Trigger	Mobile device sends location data.
Precondition	Location data is ready on the port.
Basic Path	1. Get location data from port. 2. Forward location data to select map information. 3. Set current location data as reliable.
Alternative Paths	None.
Postcondition	Location data is used for selecting map information operation.
Exception Paths	If data is not on the port or marked as unreliable by the sender, ignore it and set current data as unreliable.
Other	None.

3.2.6.2 Send Message

Use Case Name	Send Message
Trigger	Admin selects message from database and uses send button.

Precondition	Network connection is suitable.
Basic Path	Put message data on the port.
Alternative Paths	None.
Postcondition	Data is send to port of the mobile device to be handled.
Exception Paths	If a suitable network connection is not available, give an error message.
Other	None.

3.2.6.3 Send Map Information

Use Case Name	Send Map Information
Trigger	Map information is selected by database manager module.
Precondition	• Network connection is suitable. • Map information exists and it is valid.
Basic Path	1. Put map information on the port. 2. Mark data as reliable.
Alternative Paths	None.
Postcondition	Data is send to port of the mobile device to be handled.
Exception Paths	• If a suitable network connection is not available, give an error message. • If there is a problem about map information, give a warning and mark data as unreliable.
Other	None.

3.2.6.4 Send Luminosity Data

Use Case Name	Send Luminosity Data
Trigger	New luminosity data evaluated using evaluate frame data.
Precondition	• Network connection is suitable. • Luminosity data exists and it is valid.
Basic Path	1. Put luminosity data on the port. 2. Mark data as reliable.

Alternative Paths	None.
Postcondition	Data is send to port of the mobile device to be handled.
Exception Paths	• If a suitable network connection is not available, give an error message. • If there is a problem about luminosity data, give a warning and mark data as unreliable.
Other	None.

3.2.6.5 Send Movement Data

Use Case Name	Send Movement Data
Trigger	New movement data evaluated using evaluate frame data.
Precondition	• Network connection is suitable. • Movement data exists and it is valid.
Basic Path	1. Put map information on the port. 2. Mark data as reliable.
Alternative Paths	None.
Postcondition	Data is send to port of the mobile device to be handled.
Exception Paths	• If a suitable network connection is not available, give an error message. • If there is a problem about movement data, give a warning and mark data as unreliable.
Other	None.

3.2.6.6 Get Frame Data

Use Case Name	Get Frame Data
Trigger	Mobile device sends frame data.
Precondition	Frame data is ready on the port.
Basic Path	1. Get frame data from port. 2. Forward frame data to evaluate frame data. 3. Set current frame data as reliable.
Alternative Paths	None.

Postcondition	Frame data is used for extracting movement and luminosity values.
Exception Paths	If data is not on the port or marked as unreliable by the sender, ignore it and set current data as unreliable.
Other	None.

3.2.7 Client-side Communication Module

3.2.7.1 Send Location Data

Use Case Name	Send Location Data
Trigger	Location information is set by set location information operation.
Precondition	• Network connection is suitable. • Location data exists and it is valid.
Basic Path	1. Put location data on the port. 2. Mark data as reliable.
Alternative Paths	None.
Postcondition	Data is send to port of the server to be handled.
Exception Paths	• If a suitable network connection is not available, give an error message. • If there is a problem about location data, give a warning and mark data as unreliable.
Other	None.

3.2.7.2 Get Message

Use Case Name	Get Message
Trigger	Server sends a new message.
Precondition	Message is ready on the port.
Basic Path	1. Get message from port. 2. Forward message to Context Manager Module to prepare message for rendering. 3. Save message to inbox.

Alternative Paths	None.
Postcondition	Message is rendered on the screen and placed into inbox.
Exception Paths	If message is not on the port, give information.
Other	None.

3.2.7.3 Get Map Information

Use Case Name	Get Map Information
Trigger	Map information is sent by server.
Precondition	Map information is ready on the port.
Basic Path	1. Get map information from port. 2. Forward map information to render and configure map information operations. 3. Set current map information as reliable.
Alternative Paths	None.
Postcondition	Map information is processed and rendered on screen.
Exception Paths	If data is not on the port or marked as unreliable by the sender, ignore it and set current data as unreliable.
Other	None.

3.2.7.4 Get Luminosity Data

Use Case Name	Get Luminosity Data
Trigger	Luminosity data is sent by server.
Precondition	Luminosity data is ready on the port.
Basic Path	1. Get luminosity data from port. 2. Forward luminosity data to Context Manager Module. 3. Set current luminosity data as reliable.
Alternative Paths	None.
Postcondition	Luminosity data is used with user interface arrangement operations.
Exception Paths	If data is not on the port or marked as unreliable by the sender, ignore it and set current data as unreliable.

Other	None.

3.2.7.5 Get Movement Data

Use Case Name	Get Movement Data
Trigger	Movement data is sent by server.
Precondition	Movement data is ready on the port.
Basic Path	1. Get movement data from port. 2. Forward movement data to Context Manager Module. 3. Set current movement data as reliable.
Alternative Paths	None.
Postcondition	Movement data is used with user interface arrangement operations.
Exception Paths	If data is not on the port or marked as unreliable by the sender, ignore it and set current data as unreliable.
Other	None.

3.2.7.6 Send Frame Data

Use Case Name	Send Frame Data
Trigger	Capture frame operation triggers.
Precondition	• Network connection is suitable. • Frame data exists and it is valid.
Basic Path	1. Put frame data on the port. 2. Mark data as reliable.
Alternative Paths	None.
Postcondition	Data is send to port of the server to be handled.
Exception Paths	• If a suitable network connection is not available, give an error message. • If there is a problem about frame data, give a warning and mark data as unreliable.
Other	None.

3.3 Non-functional Requirements

3.3.1 Performance requirements

There are three important performance requirements namely image processing, communication, and user interface generation. Considering these three requirements

- Since image processing operations are heavy and mobile devices do not have a CPU powerful enough, these operations are to be done on the server side.
- Communication with the wireless shall be fast enough to get/send required data from server/client to client/server. The reason to use 6 parallel ports rather than only one serial port is to satisfy this fast communication.
- Instead of communication with the server for each change (zooming etc.), rendering task shall be completed on the mobile device. This prevents the communication module from being overloaded.

3.3.2 Design constraints

3.3.2.1 Standards

- Java ME configuration should be at least CLDC-1.1.
- Java ME profile shall be at least MIDP-2.1.

3.3.2.2 Programming Languages

- As mentioned in section 3.1.3, Java ME is used for client-side application.
- As mentioned in section 3.1.3, MATLAB is used for image processing tasks.
- The main application on the server side will be coded in C++.

3.3.2.3 Hardware Constraints

- Mobile device is supposed to have a camera and support Java applications.

3.3.2.4 Software system attributes

3.3.2.4.1 Reliability

Since the system is too complex, under some circumstances unreliable data may occur. For these conditions, communication between server and client is supported with a validation mechanism to keep overall system reliable.

3.3.2.4.2 Security

This project is a military software project, therefore security is very important. In this application, critical information which flows through network (such as location information of user, location information of allied troops etc.) will be encrypted to maintain security.

3.3.2.4.3 Maintainability

This software has a consistent and maintainable structure inside. However, it uses wireless technologies that can be problematic for maintaining the software.

3.3.2.4.4 Portability

Since we use Java ME on client-side, this software can be ported to large variety of mobile devices which include JRE.

4 Data Model and Description

This section describes information domain for the software.

4.1 Data Description

In the overall system;

- “Button” objects

- A “Map” object
- “Message” objects
- An “Information Bar” object
- “ItemOnMap” objects
- “Place” objects

will be managed/manipulated according to the user’s motion and the light condition of the environment.

4.1.1 Data objects

- size, color, and coordinate attributes of the “Button” objects will be manipulated. Other attributes of the Button class are priority, and name attributes.

- rgbArray, and zoomLevel attributes of the “Map” object will be managed. There will be also a Place array and ItemOnMap array in the Map class.

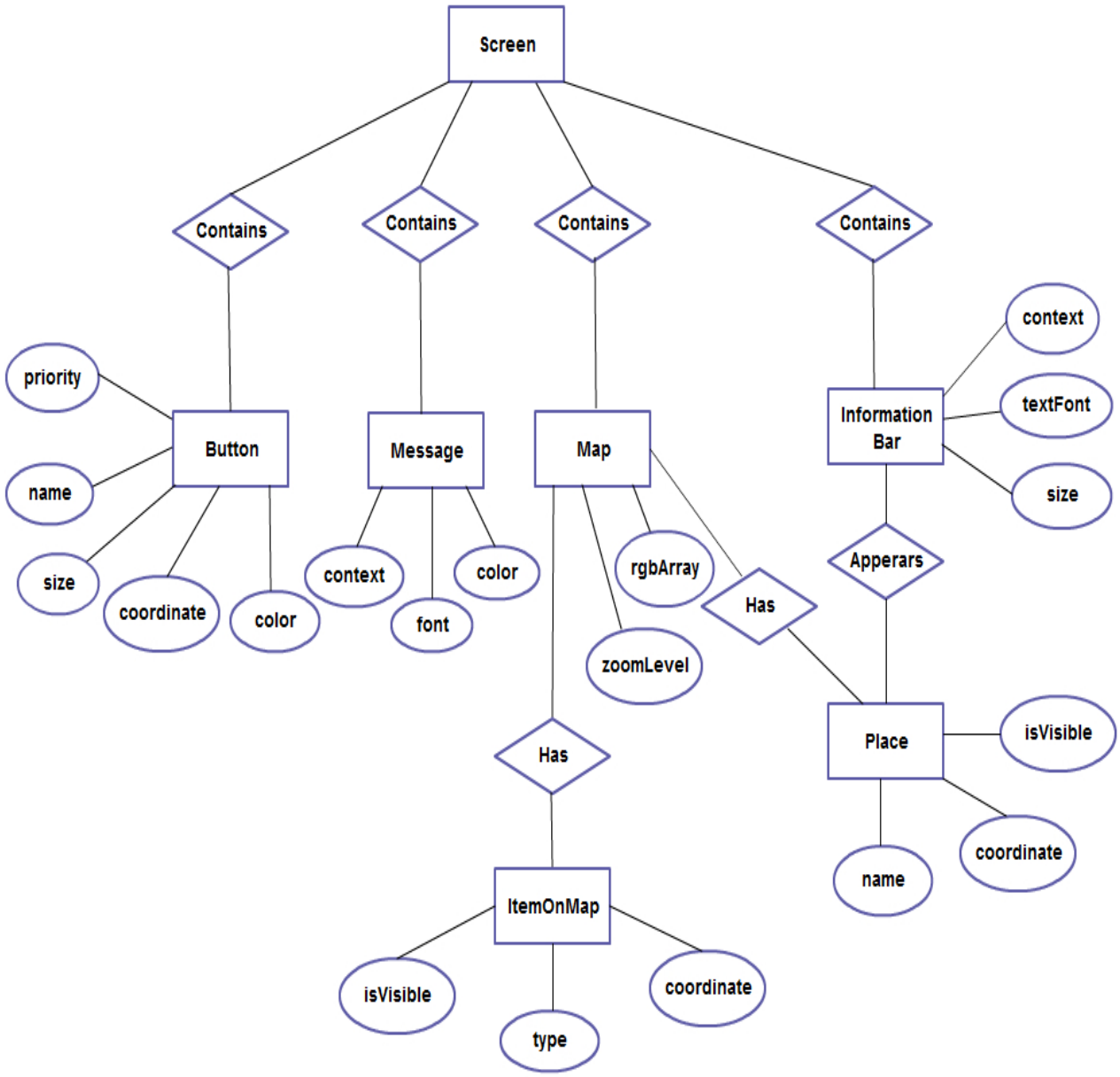
- font and color attributes of the “Message” objects will be changed. The other attribute is a context string which holds the context of the message.

- size and textFont attributes of the “InformationBar” object is manipulated. The other attribute in this class is context attribute.

- isVisible attribute in the “ItemOnMap” objects will be managed. Other attributes are type and coordinate attributes.

- isVisible attribute in the “Place” objects will be managed. Other attributes are coordinate and name.

4.1.2 Complete data model



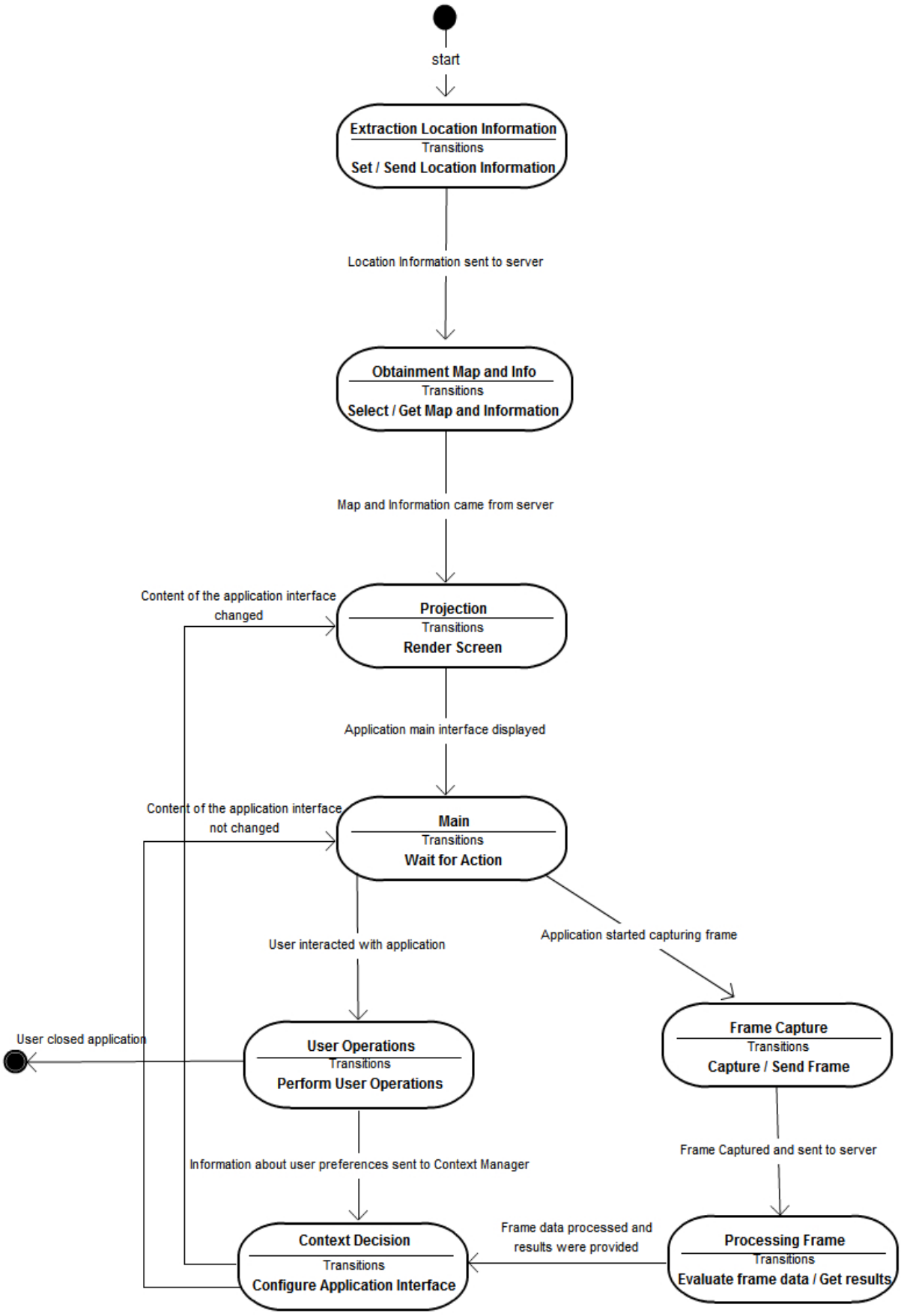
5 Behavioral Model and Description

This section presents a description of the behavior of the software.

5.1 Description for software behavior

When the application starts, location information of the user is sent automatically to server. Server selects the map and information related to location and sends to the application. System projects the default application interface and map coming from the server. Then the application passes to main state that waits the user interaction and / or context change. Meanwhile, the application keeps capturing frames of the user's environment and sends these frames to server. Server extracts information about user's context such as velocity and luminosity values. These values are provided to application by the server. System decides application interface by considering context information and user's preferences about the interface. If decided interface different from current one, application goes back to projection state to render new application interface. Otherwise system stays main state to listen user interactions and context changes. After newly decided interface projected, system passes to main state again. System works like this way until user closes the application.

5.2 State Transition Diagram


start
Extraction Location Information
Transitions
Set / Send Location Information
Location Information sent to server
Obtainment Map and Info
Transitions
Select / Get Map and Information
Map and Information came from server
Content of the application interface changed
Projection
Transitions
Render Screen
Application main interface displayed
Content of the application interface not changed
Main
Transitions
Wait for Action
User interacted with application
Application started capturing frame
User closed application
User Operations
Transitions
Perform User Operations
Frame Capture
Transitions
Capture / Send Frame
Information about user preferences sent to Context Manager
Frame Captured and sent to server
Context Decision
Transitions
Configure Application Interface
Frame data processed and results were provided
Processing Frame
Transitions
Evaluate frame data / Get results

6 Planning

6.1 Team Structure

We divided what will be done into two parts. Meryem Sağcan and İsmail Can Coşkuner will be dealing mostly with the communication and computer vision modules. Fatma Akıncı and İlker Argın will mostly work on the applications on the client-side. The group leader is Meryem Sağcan and she is responsible for arranging the meetings and controlling whether the work is completed in the determined time in weekly periods.

6.2 Estimation (Basic Schedule)

Function Points

Parameters		Simple	Average	Complex	Count
User Inputs	Brightness/Contrast Arrangement	3			22
	Zoom Arrangement		4		
	Map Content Arrangement			6	
	Screen Content Arrangement			6	
	Closing Application	3			
User Outputs	Buttons		5		31
	Information Bar			7	
	Map			7	
	Message Box		5		
	Screen Visibility			7	
User Inquiries	Map Images		4		17
	Location Information		4		
	Object Information	3			
	Messages			6	

Files	Messages	7			7
External Interfaces	Server-Side Wireless Communication			10	27
	Client-Side Wireless Communication			10	
	Database Communication		7		
Count total					104

Complexity Adjustment Factor

1 Reliable backup and recovery	5
2 Data communications	4
3 Distributed processing	3
4 Critical performance	5
5 Heavily utilized operational environment	1
6 On-line data entry	4
7 Input transactions over multiple screens (on-line)	2
8 Master file updates on-line	4
9 Complex input/output/file/inquiries	4
10 Complex internal processing	4
11 Reusable code design	4
12 Conversion and installation included in design	1
13 Multiple installations for different organizations	0
14 Design for facilitating change and ease of use	4
Total F_i	**45**

FP = CountTotal x [0.65 + 0.01 x $\sum Fi$]

= 104 x [0.65 + 0.01 x 45]

= 114.4

We decided to use an Object Oriented Programming Language. Therefore,

FP / KLOC = 30

Project Size Estimation:

KLOC = FP / 30

= 114.4 / 30

= 3.81

Project Effort Estimation:

Since we are a small group of people with good experience, our project is organic.

Effort = a_b x KLOC $^{b}_{b}$

= 2.4 x $3.81^{1.05}$

= 9.78 people / month

Project Schedule Estimation:

Time = c_b x Effort $^{d}_{b}$

= 2.5 x $9.78^{0.38}$

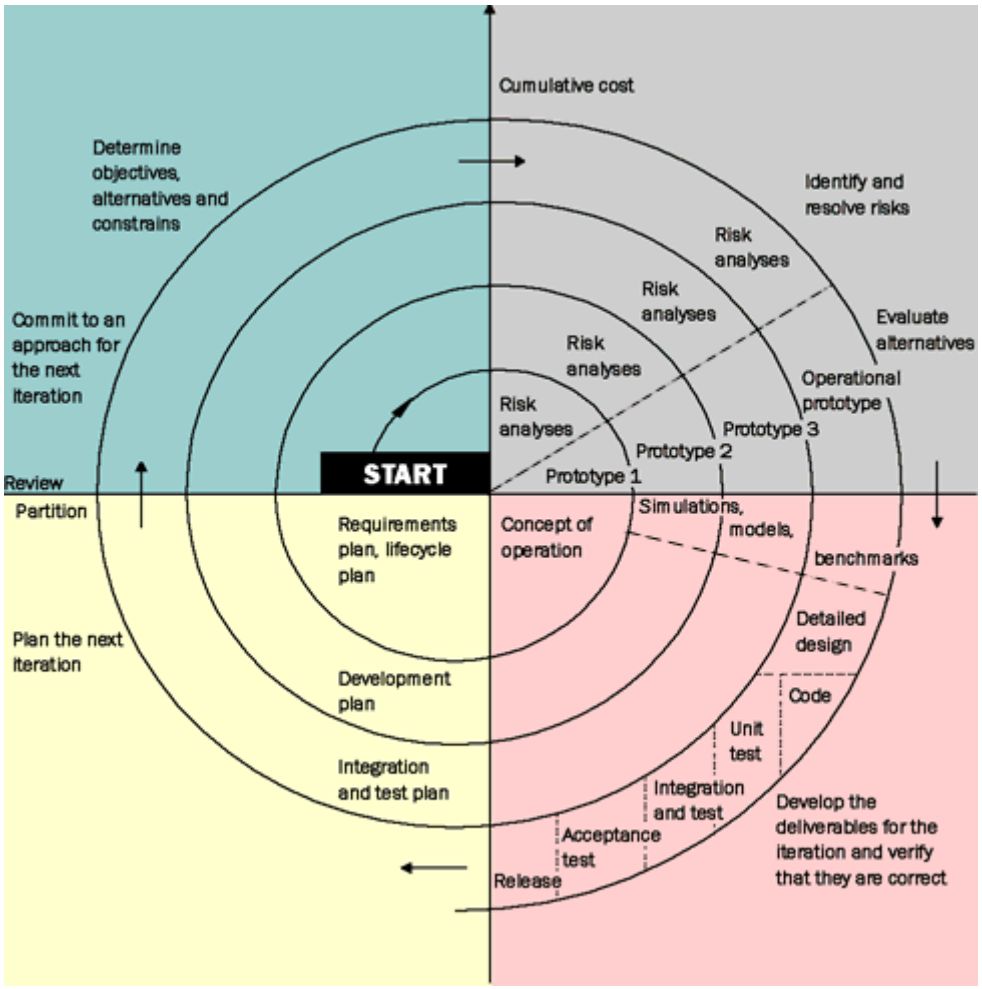
= 5.95 months ~ 6 months

Project Staffing Levels Estimation:

A small group of 4 people.

6.3 Process Model

In this project spiral model will be used. The spiral model is a software development process combining elements of both design and prototyping-in-stages, in an effort to combine advantages of top-down and bottom-up concepts.



7 Conclusion

This SRS has been intended to show a brief description of the system. The requirements of the project has been explained clearly. Developers interested in the project and users of the system can benefit from this document. Design details of the project will be explained in the Software Design Description document.