Initial Design Report for
Mobile Conqueror

Prepared by Tolga Akın, Çiğdem Avcı, Güliz Seray Tuncay and Hüseyin Ulusoy

Middle East Technical University
Dept. of Computer Engineering

2009
Table of Contents

1. Introduction ......................................................................................................................... 1
   1.1 Purpose ............................................................................................................................. 1
   1.2 Scope .............................................................................................................................. 1
   1.3 Definitions, Acronyms and Abbreviations ........................................................................ 1
   1.4 References ...................................................................................................................... 1
   1.5 Team ............................................................................................................................... 1
      1.5.1 Organizational Structure .......................................................................................... 1
      1.5.2 Member Roles .......................................................................................................... 2
      1.5.3 Process Model ......................................................................................................... 2
   1.6 Overview .......................................................................................................................... 2
   1.7 Product Perspective ......................................................................................................... 3

2. System Design ...................................................................................................................... 3
   2.1 Architectural Design ........................................................................................................ 3
      2.1.1 Composition Decomposition .................................................................................... 3
      2.1.2 Module Decomposition ........................................................................................... 4
      2.1.3 Data Decomposition ............................................................................................... 15

3. Procedural Design ............................................................................................................... 20
   3.1 Install Function ............................................................................................................... 20
   3.2 Save Picture Function .................................................................................................... 20
   3.3 Send Picture Function .................................................................................................... 21
   3.4 Take Picture Function .................................................................................................... 21
   3.5 Get More Information Function ..................................................................................... 21

4. Interfaces ............................................................................................................................ 24
   4.1 Graphical User Interface Design .................................................................................... 24
      4.1.1 Application Entrance Point .................................................................................... 24
      4.1.2 Main Window .......................................................................................................... 25
      4.1.3 Result Window ....................................................................................................... 25
   4.2 Hardware Interfaces ....................................................................................................... 26
   4.3 Software Interfaces ....................................................................................................... 26

5. Libraries and Tools ............................................................................................................. 26
   5.1 Languages and platforms ............................................................................................... 26
      5.1.1 Java ......................................................................................................................... 26
      5.1.2 C++ ......................................................................................................................... 27
      5.1.3 Eclipse .................................................................................................................... 27
   5.2 Database Connectivity Systems ..................................................................................... 27
      5.2.1 JDBC (Java Database Connectivity) .................................................................... 27
   5.3 Web Service Technologies ............................................................................................. 28
      5.3.1 Apache Axis ............................................................................................................ 28
   5.4 Operating Systems and Relational Technologies .......................................................... 28
      5.4.1 Android Emulator .................................................................................................. 28
      5.4.2 DroidDraw ............................................................................................................. 28
   5.5 Additional APIs and Libraries ...................................................................................... 29
      5.5.1 MediaWiki API ...................................................................................................... 29
      5.5.2 Google Map API .................................................................................................. 29

6. Project Schedule ................................................................................................................. 30
   6.1 Gantt Chart .................................................................................................................... 30

7. Appendix A, the Glossary .................................................................................................. 31

8. Appendix B, References .................................................................................................... 31
1. Introduction

1.1 Purpose

This document includes initial design of Mobile Conqueror. The purpose of this document is to provide an initial design description of the Mobile Conqueror.

1.2 Scope

This initial design document contains design information about Mobile Conqueror which is a demonstration of architecture, modules, classes, use cases, functions, features, database model, graphical user interface, tools and special technologies. It describes in detail all that how Mobile Conqueror works properly and with safety. The architecture is intended as the basis for interesting versions in the future.

1.3 Definitions, Acronyms and Abbreviations

See Appendix A, the Glossary

1.4 References

See Appendix B, References

1.5 Team

1.5.1 Organizational Structure

Our team does not have a team leader to lead and control the flow of development. Instead, every member of the group has equal right to state his/her thoughts, put forward an opinion, make a recommendation or object to a decision. All decisions are made on an agreement in the judgment or opinion reached by the group as a whole.
1.5.2 Member Roles

- Çiğdem Avcı              Initiator, Optimist
- Hüseyin Ulusoy         Time Keeper
- Tolga AKIN              Devil’s Advocate
- Güliz Seray Tuncay     Devil’s Advocate, Time Keeper

1.5.3 Process Model

Our project combines different topics into one. We will need to use a lot of different technologies such as GPS, Image Processing, Mobile Communication and so on. This makes the project harder and more prone to errors. Because of that, we will need to identify requirements, do prototyping, coding and testing again and again. The process model that best fits our requirements is the “Spiral Model”. The spiral model starts with defining system requirements, creating a preliminary design and constructing a first prototype. The following prototypes are constructed by evaluating the previous prototype, defining the requirements of the next prototype, planning and designing the next prototype and building and testing the next prototype.

1.6 Overview

This initial design description document contains design information about Mobile Conqueror and The rest of the document is divided into chapters for better understanding.

- In chapter 2 the architectural design is discussed.
- In chapter 3 the procedural design is analyzed.
- In chapter 4 graphical user, hardware and software interfaces are discussed.
- In chapter 5 libraries and tools are specified.
- In chapter 6 schedule of the project is presented.

This document is intended for

Developers: in order to be sure they are developing the right project that implements the design described in this documentation.

Testers: in order to have an exact list of the features, functions, models, diagrams and tools that have to respond according to design and provided diagrams.
1.7 Product Perspective

Mobile Conqueror is a self-contained product that is, it is not a component of a larger system. Apart from using GPS information to determine the location of the user, it will use information extracted from the DB which makes it a unique product in its way of supplying the user detailed information by using image processing techniques within the DB provided by the administrator of the system.

2. System Design

2.1 Architectural Design

2.1.1 Composition Decomposition

*The Mobile Conqueror* consists of three main components namely Web Server, Client and Database Server. The application connects to J2EE Server and this web server connects to the Oracle Database via JDBC interface.
2.1.2 Module Decomposition

The Mobile Conqueror consists of several packages namely *Server*, *Client*, *UserData*, *DatabaseModel*, *Users*, *Installer* Packages.

- **Server Package** is responsible for handling the operations taken by the client.
- **Client Package** is responsible for extracting related user information, sending them to the server and supplying the user of the application related information sent by the server.
- **UserData Package** holds the information of GPS location, compass and the picture itself.
- **DatabaseModel Package** holds the information of the database tables as objects.
- **Users Package** handles the user actions and passes them to the server.
- **Installer Package** is responsible for the install and uninstall actions taken by the user.

*Figure 2.2 - Architecture of Mobile Conqueror*
Figure 2.3 - Client Package and its subpackages

Figure 2.4 - Server Package and its subpackages
**Client Package**

**MachineCommunication Class**

This class is responsible for obtaining GPS location and compass information from the smartphone itself.

**ClientPosition Class**

This is the data class for keeping GPS information and compass information.

**MainController Class**

This is the class for enabling the application to return the GPS information and compass information. (Later on, MachineCommunication Class helps to get the related information from the phone.)

**SCommunication Class**

The responsibility of this class is to connect to the database and sending user information to the server.

**ServerResponse Class**

This is the data class for saving the resulting landmark information.

**DisplayMessage Class**

The message that will be displayed to the user will be kept in this class.

**Parser**

The Wikimedia API returs wikicode format as result and this result must be parsed. This class is responsible for parsing the wikicode formatted result.

**MediaWikiConnector**

This class enables the application to connect to the WikimediaAPI.

**Server Package**

**S2GMCommunication Class**

This class is the main class which communicates with the Google Maps API.

**MapInfo Class**

This is the data class for representing the Map information returned by Google Maps API.
**S2CCommunication class**

This is the main class which handles the user action. The user information is sent to this class.

**LandMarkInfo**

This class is a data class representing the information about the landmark.

**ImageProcessor Class**

This is the main class handling the image processing process. The class is mainly responsible for finding the correct match in the database, given the user picture.

**RangedImage Class**

Features of images in the range of given GPS location are kept in this class.

**ImageProcessorInfo Class**

ImageProcessorInfo class is created to hold the data such as image, GPS and Compass information which are obtained from the Database.

**Users Package**

**User Class**

The user actor will call the operation of that class.

**Admin Class**

Admin class is capable of handling the database operations.

**Installer Package**

**InstallationWizard**

This class is responsible for installation process.

**UninstallationWizard**

This class is responsible for uninstallation process.

**Phone Package**

**PhoneController**

This class is responsible for accessing the features of phone like sending image as a short message and saving pictures to the file directory.
Figure 2.5 Client Package Classes
Figure 2.6 Server Package Classes
Initial Design Report for Mobile Conqueror

Figure 2.7 Users Package Classes (Top View)

Figure 2.8 Users Package Classes and Database Model Package Classes (Part 1)
Figure 2.9 Users Package Classes and Database Model Package Classes (Part 2)
Figure 2.10 - Data Flow Diagram - Level 0

Figure 2.11 - Data Flow Diagram - Level 1
Figure 2.12 - Data Flow Diagram - Level 2 (Top View)

Figure 2.13 - Data Flow Diagram - Level 2 (Part 1)
Figure 2.14 - Data Flow Diagram - Level 2 (Part 2)
2.1.3 Data Decomposition

Data Design

**UserData Package**

**ClientData Class**

This is the data class for keeping user location information and picture itself.

**GPSInfo Class**

This is the class keeping the longitude, latitude and altitude.

**Compass Class**

This class saves the compass information.

**Picture Class**

The class keeps the pictures.

![Diagram of UserData Package Classes (Top View)](image-url)
Figure 2.16 - UserData Package Classes (Part 1)
### Figure 2.17 - UserData Package Classes (Part 2)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Compass</strong></td>
<td>direction: float</td>
<td>+ setCompass(direction: float): void</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+ getCompass(): Compass</td>
</tr>
<tr>
<td><strong>Picture</strong></td>
<td>image: IPLImage</td>
<td>+ setPicture(image: IPLImage): void</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+ getPicture(): Picture</td>
</tr>
<tr>
<td><strong>PhoneController</strong></td>
<td></td>
<td>+ saveToPhone(): void</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+ sendAsMessage(): void</td>
</tr>
<tr>
<td><strong>Client.Position</strong></td>
<td></td>
<td>- GPS: GPSInfo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- compass: Compass</td>
</tr>
</tbody>
</table>

---
Database Design

Entities

Admin
Admin entity holds the name, id, last name and password information of the administrator of the server.

Image_info
Image_info entity holds the images to be processed during picture matching with the picture taken by the mobile phone together with the metadata and location info of the images.

Users
Users entity holds the information of the user on the client side such as name, lastname and IMEI of the client.

Features
Features hold the important features which are going to be used during image processing part of the application such as region based features, edge, histogram, invariant_features, gabor_features, weight_parameter, interest_points.

Location
Location holds the location information of each image such as latitude, longitude, altitude and compass.

Relations

Added_by
Added_by relation relates images added by administrators

Sent_by
Sent by relation relates users with images sent by users

Has_features
Has_features relation relates images with features it has

Has_location
Has_location relation relates the location information with images.
Figure 2.18 - Entity Relationship Diagram
3. Procedural Design

3.1 Install Function

In order to be able to use the application, the user must be first purchased and installed to the smart phone. This process is handled by the InstallationWizard.

![Diagram](image1.png)

3.2 Save Picture Function

This function helps the user to save the tagged image into the phone.

![Diagram](image2.png)
3.3 Send Picture Function

This option enables user to send the tagged image to other mobile phone users by short messaging.

![Sequence Diagram](image)

*Figure 3.3 - Send Picture Sequence Diagram*

3.4 Take Picture Function

By this option, the users takes the photograph and it is sent to the server to be processed for obtaining more information. At the end of this processes user will be supplied with more information related to the landmark.

3.5 Get More Information Function

If the user desires to get more information related to the landmark, a browser will be opened and she will be directed to the Wikipedia web page.
Figure 3.5 - Take Picture & Get More Information Sequence Diagram Part 2

**PSEUDOCODE**

```
takeImage()
{
    boolean result
    GPSCompassInfo <- getLocation()
    result <- processInfo()
    if(result != "unknown place")
        then return result
    else
        result <- matchImage() // using image processing aspects
        return result
}
```
4. Interfaces

4.1 Graphical User Interface Design

During graphical user interface design, simplicity should be strictly taken into account. Since functionality is the main purpose of the GUI design, developing simple functions for user is what has been done so far.

An easy to use and clear GUI should be:

- **Consistent**
  
  A consistent color scheme should be used together with consistently placed tools like buttons.

- **Easy to navigate between windows**
  
  Not to bother user with solving the structure of how pages are connected, the flow between screens should be coinciding with the work which user is trying to accomplish.

- **Effectively labeled**
  
  Since the main information sources are the texts that are going to be presented on the screen to the user, they shouldn't be poorly worded and they should be easily understood.

4.1.1 Application Entrance Point

The application will start by double clicking the icon presented to the user on the main menu. And “Conqi” will be the user friendly abbreviation of the application name “Conqueror”.

*Figure 3.6 - Application Entrance Point*
4.1.2 Main Window

As it can be seen in figure 3.6, main window of the application consists of the view taken from the camera of the application together with a button which allows the user to request to take a picture through application.

*Figure 3.6 - Application Entrance Point*

4.1.3 Result Window

After the user requested to take a picture of his/her surroundings, the information is presented to the user as a label on the picture taken.

*Figure 3.7 - Application Entrance Point*
4.2 Hardware Interfaces

Machine interfacing is normally a very specific thing. The manufacturer normally provides an API (Application programmer interface) dll for interfacing with the hardware. Simply, main hardware interfaces of mobile conqueror will be listed as touch interface of the phone and ports of the server.

4.3 Software Interfaces

Firstly, JDBC is going to be used as the software interface used to reach the database. In addition to that, WSDL is and XML format for describing network services as a set of endpoints operating on messages containing either document oriented or procedure oriented information. The operations and messages are described abstractly, end then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate, however, the only bindings described in this document describe how to use WSDL in conjunction with SOAP 1.1, HTTP GET/POST, and MIME.

5. Libraries and Tools

5.1 Languages and platforms

5.1.1 Java

Java is a programming language originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems’ Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to bytecode (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture.
5.1.2 C++

C++ is a statically typed, free-form, multi-paradigm, compiled, general-purpose programming language. It is regarded as a middle-level language, as it comprises a combination of both high-level and low-level language features. It was developed by Bjarne Stroustrup starting in 1979 at Bell Labs as an enhancement to the C programming language and originally named "C with Classes". It was renamed to C++ in 1983.

Some of its application domains include systems software, application software, device drivers, embedded software, high-performance server and client applications, and entertainment software such as video games. Therefore, C++ is used for image processing part of mobile conquerer because of the speed constraints.

5.1.3 Eclipse

Eclipse is a multi-language software development environment comprising an IDE and a plug-in system to extend it. It is written primarily in Java and can be used to develop applications in Java and, by means of the various plug-ins, in other languages as well, including C, C++, COBOL, Python, Perl, PHP, and others. The IDE is often called Eclipse ADT for Ada, Eclipse CDT for C, Eclipse JDT for Java and Eclipse PDT for PHP.

5.2 Database Connectivity Systems

5.2.1 JDBC (Java Database Connectivity)

JDBC is an API for the Java programming language that defines how a client may access a database. It provides methods for querying and updating data in a database. JDBC is oriented towards relational databases.

JDBC allows multiple implementations to exist and be used by the same application. The API provides a mechanism for dynamically loading the correct Java packages and registering them with the JDBC Driver Manager. The Driver Manager is used as a connection factory for creating JDBC connections.

JDBC connections support creating and executing statements. These may be update statements such as SQL's CREATE, INSERT, UPDATE and DELETE, or they may be query statements such as SELECT. Additionally, stored procedures may be invoked through a JDBC connection.
5.3 Web Service Technologies

5.3.1 Apache Axis

Apache Axis is an open source, XML based Web service framework. It consists of a Java and a C++ implementation of the SOAP server, and various utilities and APIs for generating and deploying Web service applications. Using Apache Axis, developers can create interoperable, distributed computing applications. Axis is developed under the auspices of the Apache Software Foundation.

Besides, as a new version Apache Axis2 is a core engine for Web services. It is a complete redesign and re-write of the widely used Apache Axis SOAP stack. Implementations of Axis2 are available in Java and C.

Axis2 not only provides the capability to add Web services interfaces to Web applications, but can also function as a standalone server application.

Axis2 has support for REST by just removing the SOAP headers both on the client and on the server.

5.4 Operating Systems and Relational Technologies

5.4.1 Android Emulator

The Android SDK includes a mobile device emulator -- a virtual mobile device that runs on your computer. The emulator lets you prototype, develop, and test Android applications without using a physical device.

The Android emulator mimics all of the typical hardware and software features of a typical mobile device, except that it can not receive or place actual phone calls. It provides a variety of navigation and control keys, which you can "press" using your mouse or keyboard to generate events for your application. It also provides a screen in which your application is displayed, together with any other Android applications running.

5.4.2 DroidDraw

DroidDraw is a graphical user interface (GUI) builder for the Android platform. DroidDraw has licensed under the GNU General Public License v2.
5.5 Additional APIs and Libraries

5.5.1 MediaWiki API
The goal of MediaWiki API (Application Programming Interface) is to provide direct, high-level access to the data contained in the MediaWiki databases. Client programs can use the API to login, get data, and post changes. The API supports thin web-based JavaScript clients, such as Navigation popups or LiveRC, end-user applications (such as vandal fighter), and can be accessed by another web site (tool server's utilities).

5.5.2 Google Map API
Google created the Google Maps API to allow developers to integrate Google Maps into their websites with their own data points. It is a free service, and currently does not contain ads, but Google states in their terms of use that they reserve the right to display ads in the future.
By using the Google Maps API, it is possible to embed the full Google Maps site into an external website. Developers are required to request an API key, which is bound to the website and directory entered when creating the key. The Google Maps API key is no longer required for API version 3. Creating a customized map interface requires adding the Google JavaScript code to a page, and then using Javascript functions to add points to the map.
When the API first launched, it lacked the ability to geocode addresses, requiring users to manually add points in (latitude, longitude) format. This feature has since been added for premier.
6. Project Schedule

6.1 Gantt Chart
7. Appendix A, the Glossary

DB: Database

GPS: Global Positioning System

GUI: Graphical User Interface

OS: Operating Systems

API: Application Programming Interface

Admin: Administrator

8. Appendix B, References

[1] Web Services Description Language (WSDL) 1.1
http://www.w3.org/TR/wsd1

http://en.wikipedia.org/wiki/Apache_Axis2

http://en.wikipedia.org/wiki/Apache_Axis
