CS211 Group 15
Design Specification

Authors: lpd1; jcm14; jap;
Date: 02-12-2013
Version: 0.1
Config Ref: DOC-Des-DesSpec-001
Date: 03-12-13

Department Of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB

Contents

1 INTRODUCTION

1.1 Purpose of this document

The purpose of this document is to show the design specification for the Walking Tour Creator.

1.2 Scope

This document will include a decomposition description which will include programs in the system, significant classes in each program, modules shared between programmes and mapping from requirements to classes; a dependency description which will include component diagrams for all programs and inheritance relationships; an interface description; and a detailed design of the Walking Tour Creator which will include sequence diagrams, significant algorithms and significant data structures.

1.3 Objectives

- Describe each of the programs and the relationship between programs

- Provide a short description of the purpose of each class

- Describe the relationships and dependencies between modules

- Provide component diagrams for each program, showing the method links between modules

- Produce an interface description which includes:

    - The name and type of the class or interface
    - Classes or interfaces which it extends
    - Public methods implemented by the class or interface, including parameter names and types for each method

- Provide sequence diagrams which shows how the classes work together for the major operations of the program

- Provide a textual description of difficult parts of the system that need to be implemented (algorithms)

- Outline significant data structures using class diagrams to show entity relationships between classes, along with object diagrams which show how static relationships in class diagrams work types for each method

2 DECOMPOSITION DESCRIPTION

2.1 Programs In System

We can split the overall system down into two programs, which allow the system to be used, and the database to be accessed, both on the internet and on any Android based mobile system.

- The mobile application written in Java for Android

- Web-based application to allow a user to view submitted tours

While these two programs will be handled separately from each other, they will share certain properties, such as similar data structures to handle the same type of data, and both will use the same database to store and retrieve data
The Server is PHP based, and handles data to and from both of the programs and the SQL Database.
We will be using a PHP server to process incoming and outgoing database requests from both the web application and the mobile application.

2.2 Significant Classes In Each Program

2.2.1 Significant classes in Program 1

2.2.2 Significant classes in Program 2

2.3 Modules Shared Between Programs

## 2.4 Mapping From Requirements To Classes

The Classes that actively translate to fulfilling the Functional Requirements are mainly located within the Android application, although the WTD will be required to complete a few of the requirements not completed by the Android application.

| FR1	This will be fulfilled by the Android application. This is the most basic functional requirement, and will most likely be partly completed by [Main Class] as opposed to any specific class. Creating a new walk will be processed by the [Walk Creater] and will have the information sent to the server for later recall by either the App or the Website.
FR2	The Title and the descriptions of the walk will be stored as properties of the Walk, to be displayed upon access by either the Application or the Website. This will be done by displaying the appropriate properties when the file is accessed through the [Opening Class]. The Application will use the [Walk Creater] to save the properties when the walk is initially created by the user.
FR3	Adding the Process will be done through very similar methods as used through saving the Title and Descriptions, except that the Waypoint will be saved as an object with the GPS Co-Ordinates, Name, Description and Time Stamp saved as properties, and will be saved itself to the walk through the [Walk Creater]
FR4	This functional requirement will be completed using the same class as FR3, as the photo will also be saved as an object with different properties. The properties of the image will be the image file path, GPS Co-Ordinates, Name, Description and Time Stamp.
FR5	Within the Walk Creator their will be a button to Delete the currently in progress walk, which will link to [Deletion Class?] which will permanently delete that walk, along with all of it's properties and objects associated uniquely with that walk. Due to the fact that this could be disastrous for the user their will be a confirmation message displayed as part of the running of this class.
FR6	This functional requirement will be completed by the [Server Upload Class]. The information will be uploaded to the server using PHP, specifically using a HTTP POST method to upload the required information to a predefined URL, which will then be translated into the correct formatting by the server, and then uploaded into the database.
FR7	Should the user switch away from the application it will use a predefined Android local storage method to keep the user's data until such point that the user either closes the program or switches back to it.
FR8	This will be completed by the WTD. It will use the information loaded onto the server (see FR6 above) as well as a Google Maps API in order to correctly display the information required by the user on the map, correct to the corresponding GPS Co-Ordinates. This will be done by the client side web application's main displaying class, [Will the Web Site even have classes?].

3 DEPENDENCY DESCRIPTION
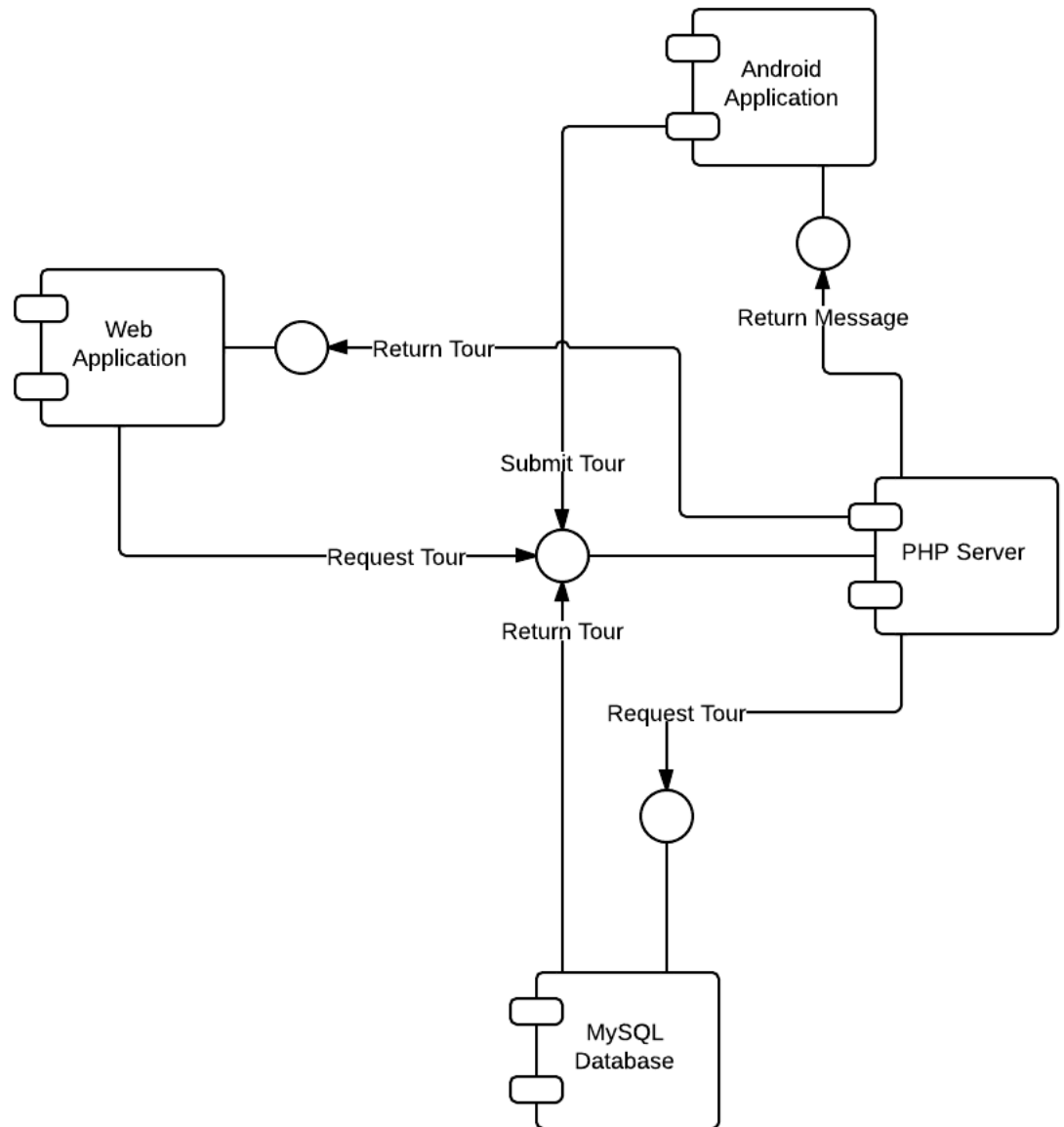
3.1 Component Diagrams

3.1.1 Deployment Diagram
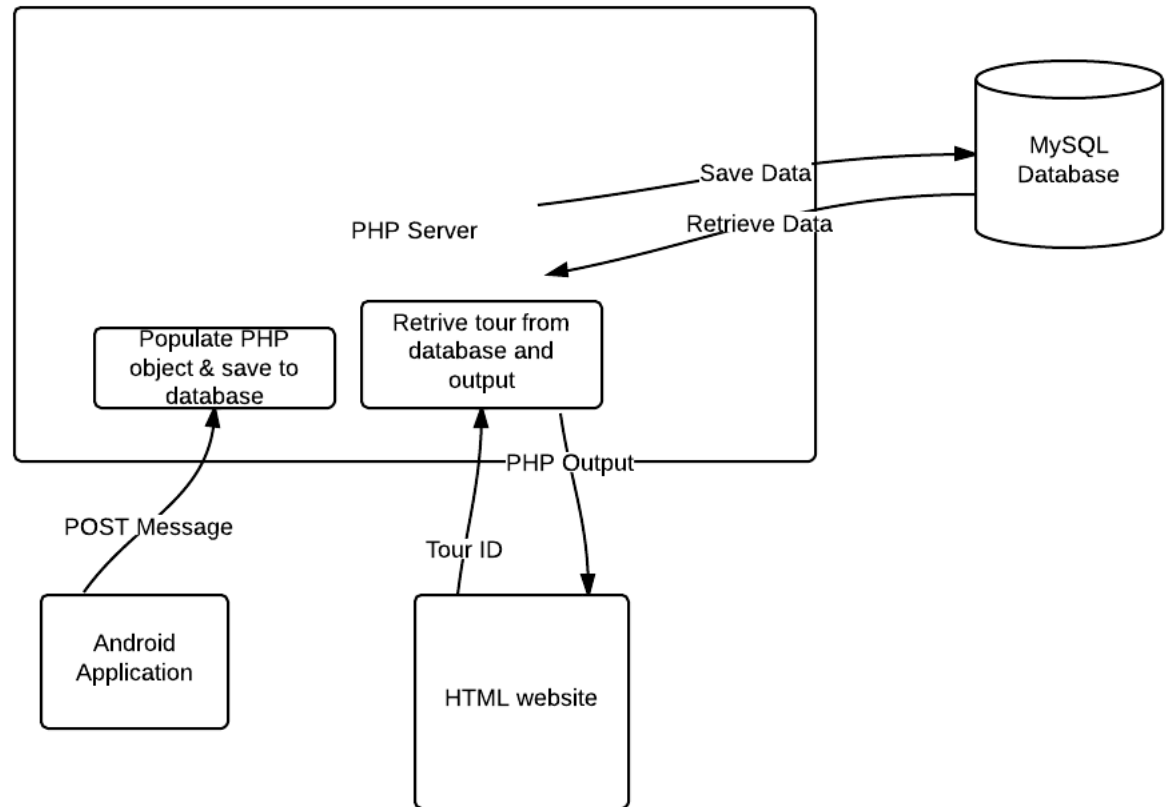
3.1.2 Component diagram for program 1
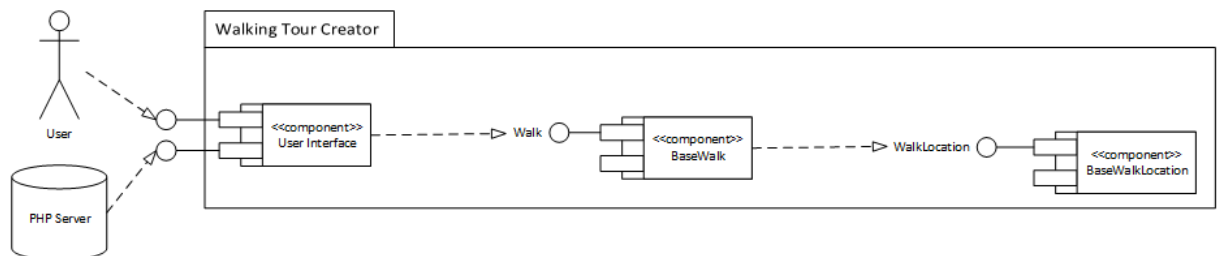
3.1.3 Component diagram for program 2

4 INTERFACE DESCRIPTION

4.1 Class 1 Interface Specification
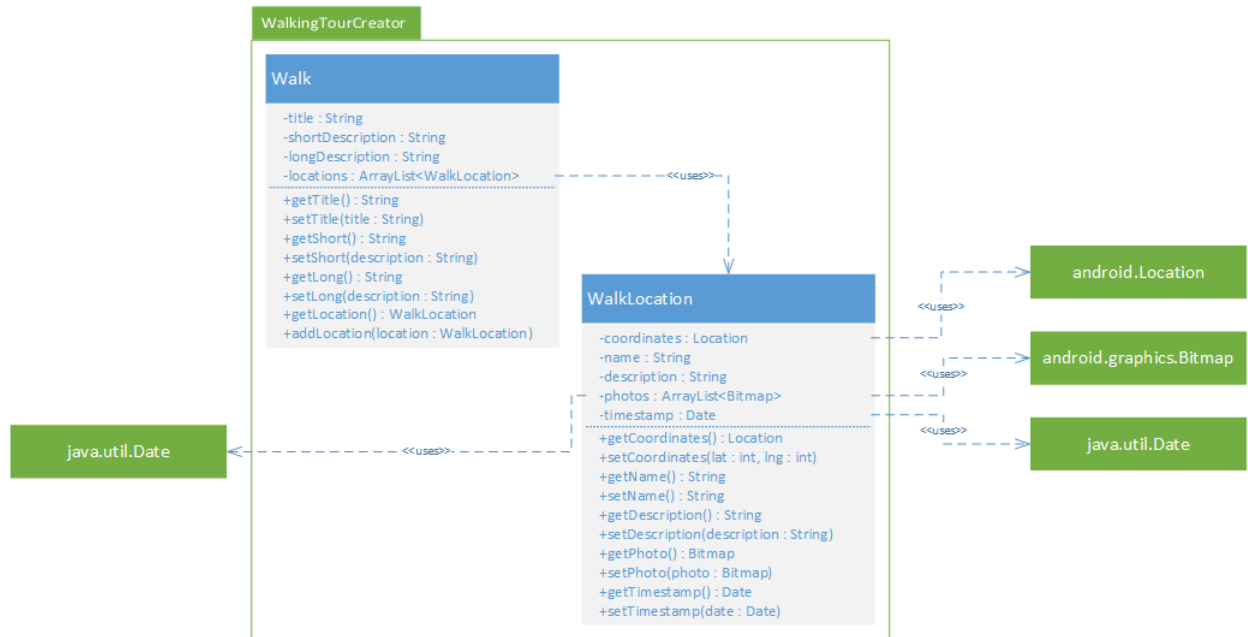
Compilation/inheritance dependencies:

    User interface depends on Walk.

    Walk depends on WalkLocation.

BaseWalk:

    + getTitle() : String

    + setTitle(title : String)

    + getShort() : String

    + setShort(description : String)

    + getLong() : String

    + setLong(description : String)

    + getLocation() : WalkLocation

    + addLocation(location : WalkLocation)

BaseWalkLocation:

+ getCoordinates() : Location*

+ setCoordinates(lat : int, lng : int)

+ getName() : String

+ setName(name : String)

+ getDescription() : String

+ setDescription(description : String)

+ getPhoto() : Bitmap**

+ setPhoto(photo : Bitmap**)

+ getTimestamp() : Date***

+ setTimestamp(date : Date***)

* imported from android.Location
* imported from android.graphics.Bitmap
** imported from java.util.Date

5 DETAILED DESIGN

5.1 Sequence Diagrams
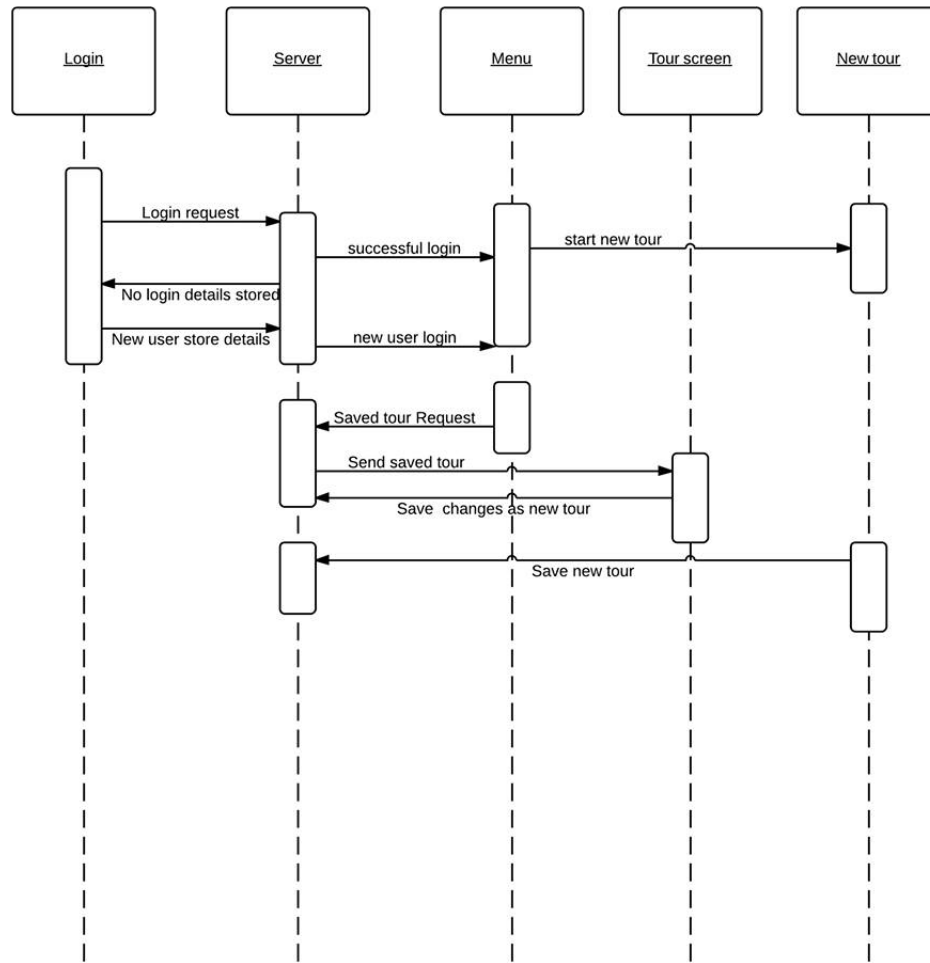
Above is the sequence diagram showing how the mobile application works and the messages passed through the mobile application and the server. The five objects show the application screens (the login, menu, saved tour screen and the new tour screen) and the server and the arrows between them show the messages sent between the objects. This chart is useful for showing the processes of the application and the ordered way the classes interact with each other.

5.2 Significant Algorithms

In this section we have designed pseudo code outlining each of the major algorithms in our system.

5.2.1 Algorithm 1 - Creating A New Walk

while App is open **do**
    if User presses "Create new walk screen" **then**
        if User has connection to internet **and** User has GPS signal **then**
            Show Walking Tour Creation screen
        else
            Show error message informing the user that they must be connected to create a walking tour
        end if
    end if
end while

This algorithm allows the user to create a new walk on the Walking Tour Creator application. It checks whether the user is connected to the internet and has a GPS signal before allowing them to begin the process.

5.2.2 Algorithm 2 - Adding Waypoints To The Walk

while App is open **do**
    if User is in walking tour creation mode **and** User has internet and GPS connection **and** User presses "Add Waypoint" **then**
        Take current GPS Location
        Take details of waypoint
        Save Waypoint information to local database
    end if
end while

5.2.3 Algorithm 3 - Sending The Walk To The Server

if User presses submit button **and** User has internet and GPS connection **then**
    Get tour information from local Android database
    Create instance of post structure as defined in Section $5.3.x$
    for all Waypoints in Database **do**
        Add waypoint information to POST
    end for
    Send POST to server
    Wait for server response
    if No response message received **or** Message shows that an error has occurred **then**
        Show error message to user
    else

Show success message to user
     end if
  end if

5.2.4   Algorithm 4 - Processing The Walk On The Server

Receive POST from mobile application
Check structure of POST
if POST is structured correctly **then**
   Create database entry for tour in tours table
   return Tour ID
   if Tour ID returned **then**
      for all Waypoints in tour **do**
         Create database entry for waypoint in tours table, referencing tour ID
      end for
      Return success message to mobile application.
   else
      Return error message to mobile application.
   end if
else
   Return error message to mobile application.
end if

5.3   Significant Data Structures

The data structures that will mainly be used on the Android application are
ArrayLists. The reason ArrayList has been chosen over an Array is because, in
Java an Array is a fixed length data structure whereas an ArrayList is a vari-
able length Collection class. This means the ArrayList will re-size itself when
it reaches its capacity. This is required because the size of the arrays we will be
using will not be known until after elements have been entered into it.

5.3.1   WalkLocation

An object of the WalkLocation class will hold variables which contain data
about the current location the user is at, such as:

- Coordinates

    – The latitude and longitude value of the location.

- Name

    – The name given to the location by the user.

- Description

    – A description of the location given by the user.

11

- Photos

  - An ArrayList of images for the current location.

- Timestamp

  - The date of the current moment when the creates the current Walk-Location object.

5.3.2 Walk

An object of the Walk class will hold variables which contain data about the walk tour the user created such as:

- Title

  - The name given to the walk tour.

- Short Description

  - A short summary of the walk tour.

- Long Description

  - A full description of the walk tour.

- Locations

  - An ArrayList of WalkLocation objects.

6 DOCUMENT HISTORY

Version	Date	Description
0.1	03-12-13	Initial document creation.
1.0	04-12-13	First draft complete.