Department of Computer Science and Engineering
The University of Texas at Arlington

Personal ECG Monitor
eCardiac

Team Members:
Michael Flanagan
Jose Miranda
Nicholas Skrobe
Marc Streeter
Minh Tran

Late Updated: 2 October 2012 @ 2:01:00 PM
Table of Contents

1 Introduction ................................................................................................................................. 8
   1.1 Overview ................................................................................................................................. 8
   1.2 Project Scope ............................................................................................................................ 8
   1.3 Meta Architecture .................................................................................................................... 8
       1.3.1 Guiding Principles ........................................................................................................... 8
       1.3.2 Guiding Assumptions ....................................................................................................... 9
   1.4 Requirements Mapping .......................................................................................................... 9
2 Architectural Layer Description ................................................................................................. 10
   2.1 Input Layer ................................................................................................................................ 10
       2.1.1 Overview ....................................................................................................................... 10
       2.1.2 Functions ....................................................................................................................... 10
       2.1.3 Interfaces ....................................................................................................................... 11
       2.1.4 Interactions ..................................................................................................................... 11
       2.1.5 Dependencies .................................................................................................................. 11
   2.2 Logic Layer ................................................................................................................................ 11
       2.2.1 Overview ....................................................................................................................... 11
       2.2.2 Functions ....................................................................................................................... 11
       2.2.3 Interfaces ....................................................................................................................... 11
       2.2.4 Interactions ..................................................................................................................... 11
       2.2.5 Dependencies .................................................................................................................. 11
   2.3 Mobile Application Layer ....................................................................................................... 11
       2.3.1 Overview ....................................................................................................................... 11
       2.3.2 Functions ....................................................................................................................... 11
       2.3.3 Interfaces ....................................................................................................................... 11
       2.3.4 Interactions ..................................................................................................................... 12
       2.3.5 Dependencies .................................................................................................................. 12
3 Inter-Subsystem Data Flow ......................................................................................................... 13
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Overview</td>
<td>13</td>
</tr>
<tr>
<td>3.2 Architectural Data Flow</td>
<td>14</td>
</tr>
<tr>
<td>3.3 Producer Consumer Relationships</td>
<td>15</td>
</tr>
<tr>
<td>4 Subsystem Description Section</td>
<td>16</td>
</tr>
<tr>
<td>4.1 Input Layer</td>
<td>16</td>
</tr>
<tr>
<td>4.1.1 Overview</td>
<td>16</td>
</tr>
<tr>
<td>4.1.2 Components</td>
<td>16</td>
</tr>
<tr>
<td>4.1.3 Desired Output</td>
<td>16</td>
</tr>
<tr>
<td>4.1.4 Required Input</td>
<td>16</td>
</tr>
<tr>
<td>4.1.5 Assumptions</td>
<td>17</td>
</tr>
<tr>
<td>4.1.6 Tradeoffs</td>
<td>17</td>
</tr>
<tr>
<td>4.2 Logic Layer</td>
<td>17</td>
</tr>
<tr>
<td>4.2.1 Overview</td>
<td>17</td>
</tr>
<tr>
<td>4.2.2 Components</td>
<td>17</td>
</tr>
<tr>
<td>4.2.3 Desired Output</td>
<td>18</td>
</tr>
<tr>
<td>4.2.4 Required Input</td>
<td>18</td>
</tr>
<tr>
<td>4.2.5 Assumptions</td>
<td>18</td>
</tr>
<tr>
<td>4.2.6 Tradeoffs</td>
<td>18</td>
</tr>
<tr>
<td>4.3 Mobile Application Layer</td>
<td>19</td>
</tr>
<tr>
<td>4.3.1 Overview</td>
<td>19</td>
</tr>
<tr>
<td>4.3.2 Components</td>
<td>19</td>
</tr>
<tr>
<td>4.3.3 Desired Output</td>
<td>20</td>
</tr>
<tr>
<td>4.3.4 Required Input</td>
<td>20</td>
</tr>
<tr>
<td>4.3.5 Assumptions</td>
<td>20</td>
</tr>
<tr>
<td>4.3.6 Tradeoffs</td>
<td>20</td>
</tr>
<tr>
<td>5 Operating System Dependencies Section</td>
<td>21</td>
</tr>
<tr>
<td>5.1 Operating System Dependencies Section</td>
<td>21</td>
</tr>
<tr>
<td>5.1.1 Input Layer</td>
<td>21</td>
</tr>
<tr>
<td>5.1.2 Logic Layer</td>
<td>21</td>
</tr>
<tr>
<td>5.1.3 Mobile Application Layer</td>
<td>21</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Testing Considerations Section</td>
<td>22</td>
</tr>
<tr>
<td>6.1 Testing Overview</td>
<td>22</td>
</tr>
<tr>
<td>6.2 Input Layer</td>
<td>22</td>
</tr>
<tr>
<td>6.3 Logic Layer</td>
<td>22</td>
</tr>
<tr>
<td>6.4 Mobile Application Layer</td>
<td>22</td>
</tr>
<tr>
<td>6.5 Integration</td>
<td>22</td>
</tr>
</tbody>
</table>
Figures

Figure 1-1 eCardiac Product Concept ........................................................................................................8
Figure 2-1 Major Architectural Layers .......................................................................................................10
Figure 3-1 Architecture Design Diagram ..................................................................................................13
Figure 4-1 Diagram of Input Layer ..........................................................................................................16
Figure 4-2 Diagram of Logic Layer ..........................................................................................................17
Figure 4-3 Diagram of Mobile Application Layer ......................................................................................19
Tables

Table 1-1 Requirements Mapping ..................................................................................................................9
Table 3-1 Data Flows .....................................................................................................................................14
Table 3-2 Producer Consumer Matrix ........................................................................................................15
## Document Revision History

<table>
<thead>
<tr>
<th>Revision Number</th>
<th>Revision Date</th>
<th>Description</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>9/10/2012</td>
<td>Gate Review</td>
<td>Compile all subsections</td>
</tr>
<tr>
<td>1.1</td>
<td>9/22/2012</td>
<td>Design Change</td>
<td>New Architecture</td>
</tr>
<tr>
<td>1.3</td>
<td>9/23/2012</td>
<td>Update Content</td>
<td>Refine Content</td>
</tr>
<tr>
<td>1.5</td>
<td>9/24/2012</td>
<td>Formatting</td>
<td>Update Formatting</td>
</tr>
<tr>
<td>1.9</td>
<td>9/24/2012</td>
<td>Baseline</td>
<td>Finalize Updates</td>
</tr>
</tbody>
</table>
# 1 Introduction

## 1.1 Overview

The purpose of the Architecture Design Specification document is to provide a high level overview of the entire system. The ADS will define the layers and subsystems that makeup the personal ECG monitor and detail the data flow between those layers and subsystems. Each layer has its own functionality that will be individually analyzed to better understand the design of the system. It also describes any operating system dependencies used by the system and any test considerations for validating the system architecture.

## 1.2 Project Scope

The Personal ECG Monitor is to be worn by a client and will regularly take ECG readings. The product will monitor readings and make note of noticeable changes. If these changes indicate that the patient is experiencing heart attack like symptoms it will contact emergency personnel by interfacing with the patient’s phone.

Our product is intended to offer effective constant monitoring for high risk heart attack patients. The product is intended to replace or alleviate the need for on premises personnel. This product is ideal for stay at home or assisted living patients. The ideal patient is in need of constant monitoring and has limited mobility.

![Figure 1-1 eCardiac Product Concept](image)

## 1.3 Meta Architecture

### 1.3.1 Guiding Principles

- Data should be transferred securely
- The system will be built as an independent unit capable of functioning with any mobile system that meets the requirements.
- The system will have a modular design so that each component can be tested independently of other components and the system could easily be improved on in the future.
- The user can easily change mobile phones from the system and transfer relevant data to the new phone.
- The system will be unobtrusive to the user wearing it.

1.3.2 Guiding Assumptions
- The user’s mobile phone will have the necessary connections to interface with the system.
- The user’s mobile phone is running an operation system that is supported by our software.
- The user will not perform strenuous activity while wearing the device.

1.4 Requirements Mapping

<table>
<thead>
<tr>
<th>Req.</th>
<th>Description</th>
<th>DAcS</th>
<th>DAnS</th>
<th>DSS</th>
<th>VS</th>
<th>MS</th>
<th>CS</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3</td>
<td>Automatic Emergency Notification</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>3.7</td>
<td>Language</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>3.8</td>
<td>Intermittent heart readings</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.9</td>
<td>Limited heart readings stored on device</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.12</td>
<td>Simple User Interface</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4.4</td>
<td>Ready to Use Processing Unit</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.1</td>
<td>Battery Life Minimum</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5.3</td>
<td>Status indicator</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5.4</td>
<td>Quick Startup</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>5.5</td>
<td>Low Battery Warning</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>5.6</td>
<td>Processing Interval</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>8.2</td>
<td>Alert Sound When Call out Occurs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>8.3</td>
<td>Able to Cancel Call</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>8.4</td>
<td>Forced Emergency Call out</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

**DAcS** (Data Acquisition Subsystem), **DAnS** (Data Analysis Subsystem), **DSS** (Data Storage Subsystem), **VS** (View Subsystem), **MS** (Model Subsystem), **CS** (Controller Subsystem)

Table 1-1 Requirements Mapping
2 Architectural Layer Description

This section describes the overall structure and strategy for how the system will be built. Each layer is described in detail of its interactions, functions, interfaces, and requirements.

The eCardiac system consists of 3 major layers:
- **Input Layer**: Retrieves data in the form of voltage readings from the user’s body and converts the readings to a digital ECG signal to be read by the Logic layer.
- **Logic Layer**: Analyzes and, potentially, may store digital representation of user’s heart activity.
- **Mobile Application Layer**: Handles all the software on the mobile system and manages the GUI display for the user as well as polling for ECG readings when needed and syncing the model and data storage between the mobile system and Logic Layer.

![Figure 2-1 Major Architectural Layers](image)

2.1 Input Layer

2.1.1 Overview

The Input Layer reads in the data through electrodes attached to the user’s body to measure electrical activity of the heart. It then prepares the data for analysis and sends it to the logic layer.

2.1.2 Functions

Acquire data in the form of ECG reading and perform an analog to digital conversion on the data. A waveform is constructed from the data that is sent to the Logic Layer.
2.1.3 Interfaces

The Input Layer will interface directly with the user’s body.

2.1.4 Interactions

The Input Layer will send data to the Logic Layer.

2.1.5 Dependencies

The Input Layer is dependent on a secure connection of electrodes to the user’s body for accurate readings.

2.2 Logic Layer

2.2.1 Overview

The Logic Layer handles ECG data analysis for the best course of action and data storage for both ECG readings taken and user data that could be necessary for emergency services.

2.2.2 Functions

The Logic Layer is responsible for storing ECG readings and user information. It can analyze the data and apply an algorithm to determine if the user is having a heart attack.

2.2.3 Interfaces

The Logic Layer will interface with the mobile system provided by the user.

2.2.4 Interactions

The Logic Layer will interact with the Input Layer to receive ECG waveform data. The Logic Layer will also interact with the mobile application that is installed on the mobile phone’s operating system (Android).

2.2.5 Dependencies

This layer depends on a connection with the mobile application to upload ECG readings to the phone and download user data.

2.3 Mobile Application Layer

2.3.1 Overview

The Mobile Application Layer handles the application on the mobile phone. This displays a GUI for the user and handles any control needed for the ECG unit over the mobile system. This layer is using a model-view-controller design to manage the data flow in the layer.

2.3.2 Functions

The Mobile Application Layer displays the user status based off of the last ECG reading. It also has an interface for the user to control the application. This layer has a listener to accept emergency notification from the Logic Layer and is able to retrieve or send data to the Logic Layer.

2.3.3 Interfaces

The Mobile Application Layer will interface with the operating system on the mobile phone (Android).
2.3.4 **Interactions**

This layer will have interactions with the Logic Layer to sync data between the model and data storage, as well as listen for push emergency notifications. It will have access to the mobile systems components to transmit or receive audio, as well as access the phone’s mobile network.

2.3.5 **Dependencies**

This layer is dependent on the current operation system distribution and version installed on the mobile phone. Not all operating systems and versions will be supported by the phone due to the large amount of production cost associated with developing applications for each operating system.
3 Inter-Subsystem Data Flow

3.1 Overview

This section introduces yet another level of abstraction. Here we graphically represent the logical subsystems of the architectural design by showing interactions between the different subsystems both internally and externally to their layer. Each subsystem represented herein can be thought of as a major programming unit that provides a particular service to the rest of the layer it resides in. Focus is given to the particular data that is consumed and produced by the different subsystems in order to greater define where responsibility lies from a functional perspective. The following is the architectural diagram that shows the locations of various data flows described more in detail later on in this section.

Figure 3-1 Architecture Design Diagram
3.2 Architectural Data Flow

Here we detail the various data flows between subsystems and the basics of how they’re used, from whom and by whom.

<table>
<thead>
<tr>
<th>Data Element</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Heart Voltages</td>
<td>An X number of variable heart voltage readings coming in from the body set electrodes to the data acquisition subsystem for processing/refinement</td>
</tr>
<tr>
<td>2. ECG Waveform</td>
<td>A Waveform object comprised of voltage readings and pulse indicators sent from Data Acquisition to Data Analysis for evaluating the healthy state of the Waveform.</td>
</tr>
<tr>
<td>3. Analyzed Waveform</td>
<td>A Waveform object, indicating its status of being healthy or being an emergency event. Sent from Data Analysis to Data Storage for storage and to Model for display.</td>
</tr>
<tr>
<td>4. Standard Waveforms</td>
<td>Waveform objects to indicate ‘safe’ heart conditions. Sent from Data Storage to Data Analysis for comparison.</td>
</tr>
<tr>
<td>5. User Information Packet</td>
<td>A packet of information identifying the location name and status of the user sent from Data Storage to Model for eventual display.</td>
</tr>
<tr>
<td>6. Display Data</td>
<td>A packet of information indicating what screens to display, and what data to populate the fields with.</td>
</tr>
<tr>
<td>7. Touch Activity Triggers</td>
<td>A packet of information indicating what buttons have been pressed, and what values the user has put into the phone through the touch-screen.</td>
</tr>
<tr>
<td>8. Controller Updates</td>
<td>A packet of information indicating what changes need to be made in model so updated information can be updated in view.</td>
</tr>
<tr>
<td>9. Pressure Signals</td>
<td>Signals from the touch sensitive screen to view</td>
</tr>
<tr>
<td>10. Visual Bits</td>
<td>The video draw from View to the screen/user</td>
</tr>
<tr>
<td>11. Audio In</td>
<td>Speaker output, either emergency center voice, or sounds from our app.</td>
</tr>
<tr>
<td>12. Audio In</td>
<td>The user’s voice, used during calls.</td>
</tr>
<tr>
<td>13. Receiving Radio Signal</td>
<td>The Cell phone signal from the call center to the phone.</td>
</tr>
<tr>
<td>14. Sending Radio Signal</td>
<td>The Cell phone signal from our phone to the call center.</td>
</tr>
</tbody>
</table>

Table 3-1 Data Flows
### 3.3 Producer Consumer Relationships

The following table is a more direct visual representation showing consumer and producers of the various data flows listed above, and using their individual identification numbers to represent them.

<table>
<thead>
<tr>
<th>Producer</th>
<th>ECG Pads</th>
<th>DAcS</th>
<th>DAnS</th>
<th>DSS</th>
<th>VS</th>
<th>CS</th>
<th>MS</th>
<th>Cell Network</th>
<th>Cell Screen</th>
<th>Mic</th>
<th>Speaker</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECG Pads</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Acquisition</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Analysis</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Storage</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>View</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>Controller</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cell Network</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cell Screen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microphone</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Speaker</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DAcS** (Data Acquisition Subsystem), **DAnS** (Data Analysis Subsystem), **DSS** (Data Storage Subsystem), **VS** (View Subsystem), **MS** (Model Subsystem), **CS** (Controller Subsystem)

Table 3-2 Producer Consumer Matrix
4 Subsystem Description Section

Here is a detailed analysis of the different subsystems that comprise the whole project. Primarily their purpose and function are described.

4.1 Input Layer

![Figure 4-1 Diagram of Input Layer](image)

4.1.1 Overview

The Input Layer is responsible for taking in the raw cardiac data generated from the user's body. It makes that data available in a format that is usable in the algorithm run by the Logic Layer.

4.1.2 Components

4.1.2.1 Data Acquisition Subsystem

**Summary:**
This subsystem must produce ECG waveforms that will be passed to the Data Analysis Subsystem present in the Logic Layer. Part of this process includes converting the analog voltage signals from the electrodes to a digital waveform.

**External Dependencies:**
In order to take meaningful readings the electrodes must be properly connected to the user at certain points on the body; otherwise the readings attained would either generate a false alarm or, worse yet, fail to alarm the user of an emergency.

**Internal Data Flow:**
It sends its resulting waveform to the Data Acquisition Subsystem.

4.1.3 Desired Output

This layer will produce a waveform from the data acquired from the electrodes connected to the user's body.

4.1.4 Required Input

In order to take meaningful readings the electrodes must be properly connected to the user at certain points on the body (as mentioned in the External Dependencies listed above).
4.1.5 Assumptions
Electrodes are connected to patient properly and only reasonable interference is being experienced. Interference that would typically throw off the readings of any non-portable/portable ECG reader will be expected to cause undesired malfunction to our device as well.

4.1.6 Tradeoffs
Having more leads makes the signal generated from the user’s body more accurate. With more leads, however, also comes the raised probability that at least one will fall off of the patient but still take in erroneous information. Also, the cost of ECG’s available on the market raises with more leads is higher than our project expenses provide. With these criteria in mind, we chose to reduce the number of leads in order to reduce the cost and complexity of this project.

4.2 Logic Layer

4.2.1 Overview
The Logic Layer is critical to every other system in the unit. Its critical nature stems from the fact that every other system depends on the data it generates. It analyzes the data harvested from the user in the Input Layer. The results from analysis are then stored periodically in the Data Storage Subsystem.

Note: Though all readings it acquires are analyzed live, only emergency and near emergency events are stored for later retrieval.

4.2.2 Components

4.2.2.1 Data Analysis Subsystem

Summary:
This subsystem must take waveform generated by Data Acquisition Subsystem and analyze it for heart issues (heart attack, cardiac arrest, pulse, etc.).

External Dependencies:
This subsystem depends on data from the Input Layer.
Internal Data Flow:
Takes live data from Data Acquisition Subsystem and analyzes the data with regard to stored data (heuristic) in the Data Storage Subsystem.

4.2.2.2 Data Storage Subsystem

Summary:
This subsystem stores all relevant user data, system status data, and serves as a live repository of the most recent cardiac activity. It can be polled by the Mobile Application Layer, but typically will be the instigator of interlayer communication (due to power saving issues).

External Dependencies:
It gets user name and location from the Mobile Application Layer. Also it must have a hook to a listener in the Mobile Application Layer when heart attack has been sensed within acceptable probability ranges. This way it can push an alert.

Internal Data Flow:
It sends heuristic to the Data Analysis Layer for use in the heart analysis algorithm.

4.2.3 Desired Output
- All data required for Mobile Application Layer (User Location, Name, Status, etc.)
- Push notifications for emergency heart conditions.

4.2.4 Required Input
In order to analyze the heart activity a proper waveform must be provided by the Input Layer.

4.2.5 Assumptions
ECG waveform construction signal processing is a sufficiently accurate representation of the internal heart activity.

4.2.6 Tradeoffs
The Data Storage Layer was moved from the Mobile Application to the Logic Layer. A tradeoff had to be made between the desires to store larger quantities of data (on the Mobile Application Layer) versus battery life. If the data were to be stored on the mobile device (where the Mobile Application Layer would reside) then data would need to flow constantly between it and the micro controller. Even if the heart activity analysis was done on the mobile device there would need to be a constant battery drain to maintain communication between the Data Acquisition Subsystem and itself since we are transmitting data wirelessly.
4.3 Mobile Application Layer

![Diagram of Mobile Application Layer](image)

**Figure 4-3 Diagram of Mobile Application Layer**

4.3.1 Overview
This layer deals major function alerting the user of his current status and that of the eCardiac device he has paired his phone with. This layer in this way is an intermediary and visual representation of the internal status of the eCardiac device.

4.3.2 Components

4.3.2.1 Model Subsystem
**Summary:**
This subsystem links the Mobile Application Layer with the Logic Layer. It holds the functions necessary to sync the other subsystems with the current data held. When a view may want to poll for continuous live ECG readings it would need to pass through here.

**External Dependencies:**
This subsystem depends on a hook to the Data Storage Subsystem of the Logic Layer. It communicates directly for information. It also is a listener for the emergency notification sent out by the Logic Layer.

**Internal Data Flow:**
The Model Subsystem sends updates of its local storage to the View Subsystem for drawing out to the screen the proper data. This subsystem also receives updates to user data (like name and address) from input received in the Controller Subsystem.

4.3.2.2 View Subsystem
**Summary:**
This subsystem lays out the user interface elements and status elements of the Mobile Application Layer. All relevant data will be arranged here for display. Additionally user interactions that are caught on these view elements will be sent to their corresponding view controllers within the Controller Subsystem.

**External Dependencies:**
None
Internal Data Flow:
Receives updates for the views from the Model Subsystem and sends events notifications to the Controller Subsystem.

4.3.2.3 Controller Subsystem
Summary:
This subsystem is our application’s link to the underlying mobile framework. Many different functions referenced here link into lower level functions that we are not otherwise given access to for performance and/or security issues. This subsystem also receives events from our views if they are interacted with by the user. If the user decides to change the screen they are on by, for example, entering a screen to enter in their current location or personal name then this subsystem would switch what view were actually activated to show the user. This would also be the system that would send notifications to the phone’s internal operating system when, for instance, our application was in the background and needed attention from the user in the event of an emergency.

External Dependencies:
It depends on the operating system making available a framework that can handle intercommunicating user tactile input events. Also depends on the framework’s ability to interpret the views generated via the coupling of the view controllers of the Controller Subsystem and the view components of the View Subsystem into the video image that will be displayed on the mobile screen.

Internal Data Flow:
This subsystem sends updates received from the user and the phone (a dropped call perhaps) to the Model Layer. It also receives events captured by the individual components as interacted with by the user on the screen (assuming the screen can also be an input device).

4.3.3 Desired Output
The correct framework interactions with the mobile operating system which will do the low level heavy lifting. We will not be writing code that is already available and so will be depending on the output from our Controller Subsystem to this external framework to be correct. Also we depend on this layer communicating any information changes received from the View Subsystem back to the Model Subsystem. In this way the View Subsystem can be updated.

4.3.4 Required Input
Any interactions from a keyboard, call status change (call engaged, call lost, location data - where available, etc.), etc.

4.3.5 Assumptions
The framework will handle how to create the video image displayed on the device, the framework will allow access to the audio transmitted over the Mobile Network, and the Android OS will allow our application to send and receive status updates and perform outbound calls even while performing as a background process.

4.3.6 Tradeoffs
None.
5 Operating System Dependencies Section

5.1 Operating System Dependencies Section

5.1.1 Input Layer

The Input Layer will be developed in C or PIC Assembly using a PC. Using a development environment loaded on the PC, we must be able to both load and create binary code from and to the microcontroller.

5.1.2 Logic Layer

The Logic Layer will be developed in C or PIC Assembly using a PC. Using a development environment loaded on the PC, we must be able to both load and create binary code from and to the microcontroller.

5.1.3 Mobile Application Layer

The application inside Mobile Application Layer will be developed on a mobile operating system platform. The platform must provide API (Application Programming Interface) libraries and development tools for building, testing and debugging. We will use the GUI, Action Handler and Update Listener subsystems to build our mobile Application Layer, process incoming data and produce outgoing data.
6 Testing Considerations Section

6.1 Testing Overview

Team eCardiac will begin by testing each function or class independently. Next we will test the function or methods to insure correct results. We will proceed by testing the subsystem that the code belongs to. We will then test the interactions between the subsystems. Upon success we will proceed by testing the layer as a whole. Once we have tested the layer we will test the architecture by integrating the layers and checking for correctness. At each stage we will make note of the expected results and the actual results to insure that they align. At each step we will also confirm that the expected data flow was passed.

6.2 Input Layer

The Input Layer is responsible for obtaining from the user’s body heart readings that are clear enough to be read and analyzed by the Logic Layer. We will test that it is correctly retrieving a waveform comparable to other proven ECG devices that are used on the market.

6.3 Logic Layer

The Logic Layer is responsible for analyzing and storing the users ECG readings. We will insure that it is correctly analyzing those readings. The Logic Layer also must be able to store user information so we will make sure that information can be correctly stored and received. We will test that correct ECG readings are going into the layer and that proper data flows and information is being obtain by the Mobile Application Layer.

6.4 Mobile Application Layer

The Mobile Application Layer is responsible for querying user information and displaying a UI. This layer also displays status depending on the user’s ECG readings. Team eCardiac will insure the proper displays are prompted for expected status updates. The Mobile Application Layer must also display interactive user options that will be checked against expected results. This layer must obtain a correct waveform from the Logic Layer and properly display it. It must also correctly initiate a call out when an emergency test case has been performed.

6.5 Integration

The Personal ECG monitor is expected to take an ECG reading and analyze whether it is indicative of a heart attack. If a positive heart attack reading is detected a call to 911 would be made. In order to test the integrated layers we will feed in different ECG readings into our system. We will make certain that a call to emergency services results from a simulated heart attack ECG reading and that in the event the user doesn’t wish to call emergency services he is given the opportunity to cancel the call.