Homework 1&2

Team 8

Andrew Ross: [anross@fullerton.edu](mailto:anross@fullerton.edu)

Mohammed Hassnain: [Mohammed-17@csu.fullerton.edu](mailto:Mohammed-17@csu.fullerton.edu)

Hemanth Sivakumar: [hemanth@csu.fullerton.edu](mailto:hemanth@csu.fullerton.edu)

Jay Kim: [Jkim128@csu.fullerton.edu](mailto:Jkim128@csu.fullerton.edu)

Submission date: Dec 6, 2017

CPSC 462-03: Software Design

Dr. Chang Hyun Jo

**Table of Contents**

1.Title

2.Table of Contents

3.Revision History

4.Project Plan

9. Development Cases

11. Use Case Diagram

12-13. Inception Artifacts

14-16. Supplementary Specification

17-18. Glossary

19. Elaboration Artifacts

20-26. Use Cases

27. Traceability Matrix

28. Traceability Diagram

29-37. Low level SSD

38. Domain Model

39. UP Disciplines

40. Package Diagram

41. Arch Layer

42-47. Interaction Diagram

48-50. GRASP Patterns

51. Design Class Diagram

52-55. Communication Diagram

56-60. Operation Contracts

61-62. Lessons Learned

63. Team Evaluation Template

64-67. Team Charter Template

68-79. Main C++ Code/Output

80. References

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author |
| Elaboration draft | 12/06/2017 | Final draft. Refined in elaboration. | Mohammed, Jay, Rishi, Andrew |

**Project Plan**

**Vision**

**Introduction**

We envision to build a multi-purpose wristband device with internet capabilities to perform basic everyday functions as well as to provide health and convenience features.

**Positioning**

**Business Opportunity**

Current wristband devices on the market are not meeting the demands that customers are requesting for. Most watches lack in ease-of-use, maneuverability, and multi-functions. In addition, they do not provide other benefits with regards to human health other than displaying the time. There is a marketplace for this dissatisfaction which has led to a lack of innovative products.

**Problem Statement**

Traditional wristband devices are bulky, not suitable for long term use and are difficult to use. They also do not provide customization options for the user. These products limit the user’s ability to adapt a product into every part of their daily lives and therefore is obsolete.

**Product Position Statement**

Our Fit Band device is for user’s looking for people who are on the go and want to make a change to their health lifestyle. When it comes to reaching your fitness goals, steps are just the beginning. Our fitness watch tracks every part of your day to help you find your fit, stay motivated, and see how small steps make a big impact. Sync wirelessly and automatically from your device to your phone or computer. We have developed a product that adapts/learns/grows with your needs to help you succeed.

**Stakeholder Descriptions**

Much has been written about Fit Band’s potential to transform an everyday experience, regulations governing wristband devices, particularly the regulation of mobile applications, and regulatory effects on technology development. We conducted a overview of the stakeholder analysis based on the framework of the primary target audience for this product line.

**Market Demographic:**

All ages: Teens and Adults ranging from 15-45 years old

**Stakeholder (Non-User) Summary**

***Stakeholders in this group include:***

Researchers: Researchers may use Fit Band to generate more and potentially better data for use in clinical trials.

Nutrition Focused Retail stores: Vendors, suppliers, distributors, small-to-medium enterprise app developers and consultants could potentially develop business selling Fit Bands, and major platform providers also benefit from our product.

Friends and Families: Families and others responsible for care seek improvements in care delivery and care coordination, and more efficient management of their loved one’s care.

**User Summary**

Our wristband device is intended for users who want to live a healthy life style and track their personal health status daily. Additional features include: being able to send texts to contacts (or 911), find location via GPS, detect a heart attack, check pulse, check glucose level, provide clock functionality.

**Key High-Level Goals and Problems of the Stakeholders**

Through analysis with subject matter experts and other stakeholders, and surveys at several retail outlets led to identification of the following key goals and problems:

|  |  |  |  |
| --- | --- | --- | --- |
| **High-Level Goal** | **Priority** | **Problems and Concerns** | **Current Solutions** |
| Sending texts to contacts/911 | High | Lack of signal in rural areas or response from mobile network | Sending messages in range of a cellular tower. Device enters offline mode |
| Find location via GPS | High | Lack of update response time from mobile network | Using existing product specifications to process location request. Device enters offline mode |
| Detect a heart attack or pulse/glucose levels | High | Sensor misreading/Equipment failure | Product return or exchange is an option within the return policy time frame (180 days) |
| Provide clock/alarm functionality | Medium | None | Existing product provides basic time features for a clock. |
| Provide nightlight functionality | Medium | None | Existing product provides basic light feature. |
| Allow network connectivity | High | Connection loss | Download updates and information when connected to a network |
| Make/receive calls | High | Connection loss | Device enters offline mode |
| Step tracker/Calorie counter | High | None | Existing product provides basic features for each requirement. |

**User-Level Goals**

The users (and external systems) need a system to fulfill these goals:

System administrator: handles text messages sent, manages pulse and glucose level activities for alert notifications.

Wristband system: analyze health data (glucose, pulse, heart rate), display time, provide GPS location, send text messages over a network.

**User Environment**

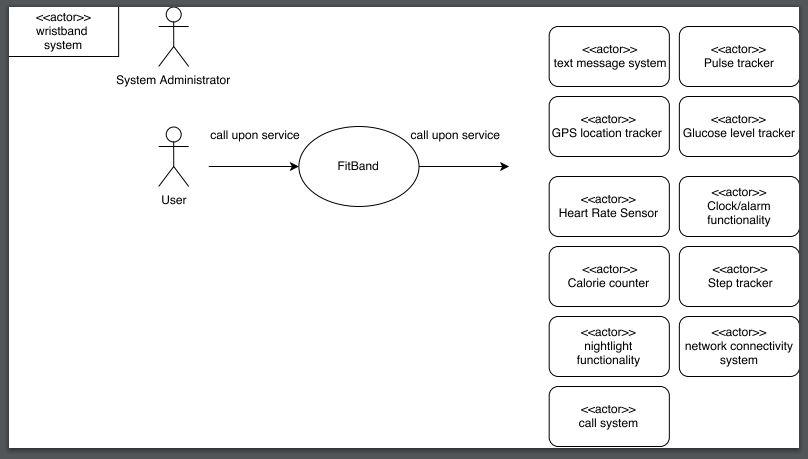
Users can wear the device wherever they go bringing ease and convenience with them as they take on each day. Whether at the gym getting in a intense workout or at work in a corporate office our device is designed for any environment your life entails.

**Product Overview**

***Product Perspective***

The FitBand device will reside in retail stores and nutrition shops. The device can communicate with a mobile network to provide further services to users, and collaborate with other systems.

**Figure Vision: System Context Diagram**



***Summary of Benefits***

|  |  |
| --- | --- |
| **Supporting Feature** | **Stakeholder Benefits** |
| Functionally, the system will provide all the common services a consumer requires, including step counter, calorie tracker, ability to make calls, nightlight functionality, clock/alarm features etc. | More features that allow for greater user convenience. |
| Automatic detection of internal sensor failures, providing feedback on device battery status. | Other components continue to work as designed |
| Real time updates over mobile network | Notified with updates immediately |

***Assumptions and Dependencies:***

User capable of navigating device to utilize features listed in the use cases. Certain features will be dependent on external factors for example making calls or sending texts will rely on a mobile network.

***Cost and Pricing:***

Cost: Fit-Band device flat rate: $150.00

Pricing: Subject to change based on retail store location and future price adjustments

***Licensing and Installation:***

Product is a registered trademark by Fit-Band Inc.

Installation: Place wristband device on wrist and tighten at your convenience.

**Summary of System Features:**

\*Glucose level tracker

\*Heart rate pulse tracker

\*Ability to make calls over a mobile network

\*Ability to send text to contacts and 911 in case of an emergency

\*Heart rate alert monitor to provide a notification to the user when the heart rate enters a high-risk level.

\*GPS location information provided to the user when the device is connected to a mobile network.

\*Clock/alarm functionality

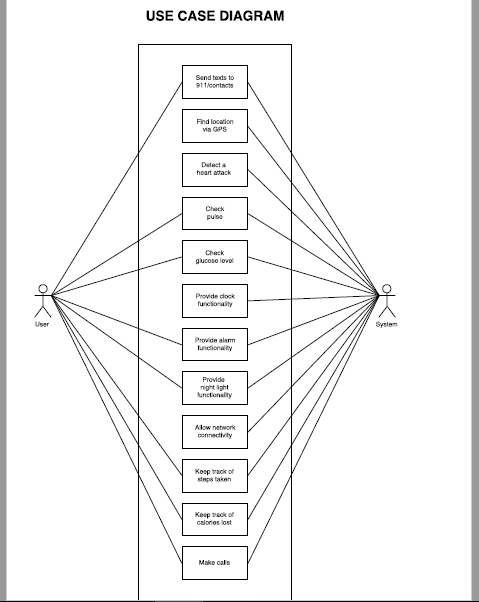
\*Nightlight functionality embedded to allow user to see in low visibility.

**Other Requirements and Constraints**

Including design constraints, usability, reliability, performance, supportability, documentation, packaging, refer to: supplementary specifications and use cases.

**Development Cases**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Discipline | Practice | Artifact | Incep. | Elab. | Const. | Trans. |
|  |  | Iteration-> | I1 | E1...En | C1...Cn | T1...T2 |
| Business Modeling | Agile modeling req. workshop | Domain Model |  | s | -- | -- |
| Requirements | Req. workshop vision box exercise dot voting | Use-Case Model  \_\_\_\_\_\_\_\_\_  Use case diagram  \_\_\_\_\_\_\_\_\_  Vision  \_\_\_\_\_\_\_\_\_  Supplementary Specification  \_\_\_\_\_\_\_\_\_  Glossary  \_\_\_\_\_\_\_\_\_ | s  \_\_\_\_\_\_\_\_\_  s  \_\_\_\_\_\_\_\_\_  s  \_\_\_\_\_\_\_\_\_  s  \_\_\_\_\_\_\_\_\_  s  \_\_\_\_\_\_\_\_\_ | r  \_\_\_\_\_\_\_\_\_  r  \_\_\_\_\_\_\_\_\_  r  \_\_\_\_\_\_\_\_\_  r  \_\_\_\_\_\_\_\_\_  r  \_\_\_\_\_\_\_\_\_ | -- | -- |
| Design | Agile Modeling Test-driven development | Design Model  \_\_\_\_\_\_\_\_\_  SW Architecture  Document  \_\_\_\_\_\_\_\_\_  Data Model | \_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_ | s  \_\_\_\_\_\_\_\_  s  \_\_\_\_\_\_\_\_  s | r  \_\_\_\_\_\_\_\_  --  \_\_\_\_\_\_\_\_  r | -- |
| Implementation | Test-driven dev. Pair programming continuous integration coding standards | Implementation Mode  ...l |  | s | r | -- |
| Project Management | agile PM | Proj/Proc Mgmt & Support... | s | r | -- | -- |
|  |  |  |  |  |  |  |



**Inception Artifacts**

|  |  |
| --- | --- |
| **Artifacts** | **Comments** |
| Vision and Business case | **Goal**: Build a multi-purpose wristband device with network connectivity capabilities to perform basic everyday functions as well as to provide health and convenience features.  **Business Case**: System must run on a mobile network. User interface must be easy to navigate. Software customization should be available for any future updates to the device.  **Exec. Summary**: To initiate a project that builds a mobile application, running on the mobile environment. Particularly a wrist-band wearable on the human body.In order to develop this device we will be using the Object-Oriented Analysis/Design (OOAD) and Unified Process (UP) in order to build software for the target system. |
| Use-Case Model | Functional requirements description: Ability to send/receive texts to/from 911 and their favorite contacts, share location via GPS, detect heart attack(send emergency message to authorities), check pulse rate/glucose level, check time/set alarm, access apps through voice-recognition, connect to a network, keep track of steps/calories and make/receive calls.  Details for use case #9: network connectivity-When the device is powered on and user enables network connectivity they can utilize features for the device that rely on a network connection for example: sending text/making calls etc. (a network service provider is needed for this) |
| Supplementary Specification | Non-functional requirements description:  **User Friendly:**  Speed, ease, and task processing will allow for a pleasant with regards to navigating the device as well as completing any tasks. For example, the device will be able to tell whether the user is walking or not through a step count notification as well as GPS location coordinates.  **Accessibility:**  The device must be connected to a mobile network to send/receive phone calls/text messages and any type of software update. The mobile network will also provide GPS capability to the device for exact coordinate locations.  **Availability:**  The device will have a battery life capable of staying powered on for exactly a week. This will give the user plenty of time to utilize the device before it needs to be charged again.  **Customization:**  User will be able to customize the device through the settings tool. The user will be able to install/remove apps when connected to a network. The device can receive updates for apps/software patches over a network as well. The device can be configurable to work with the user’s needs for any task mentioned in the functionality field above. |
| Glossary | Key terminology includes: display,fitness tracker,heart rate, notifications, wearable technology, syncing. |
| Prototype and proof-of-concepts | The vision will give a detailed description of the high level requirement goals mentioned in the supplementary specifications.It is an important priority in building a wristband device with the features that have been requested for in the uses cases. Technical ideas will be provided once program development starts in the next iteration. |
| Iteration plan | Refining business models and requirements such as use case diagram, vision and supplementary specifications for further updates. |
| Phase Plan & Software Development Plan | Will be determined in next phase/iteration. |
| Development Case | UP disciplines, architectural layers, use cases, use case diagram, UP artifact influence, supplementary specification, glossary, vision, compare-contrast sys. features with uses cases, quality attributes,domain model, association, attributes, system sequence diagram, operation contracts. (subject to change) |

**Supplementary Specification**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author |
| Inception Draft | 9/27/2017 | Working on refining during the elaboration phase. | Mohammed, Rishi, Jay, Andrew |

**Introduction**:

This document is the repository of all wristband device requirements not captured in the use cases.

**Functionality:**

The device can send/receive texts to/from 911 and their favorite contacts. Text messages are stored in the internal device memory. It can share location via GPS. It can detect the symptoms of a heart attack. The device will send an emergency text message to 911. The user can check their pulse. History of pulse checks are stored in the internal device memory. The user can check their glucose (blood sugar) level via skin. The device displays the time in 12-hour or 24-hour format. The user can set alarm(s). The user can use the nightlight feature. Device can connect to a network. The user can keep track of steps taken. The steps can be counted within a timeframe: minutes, hours, and days. The user can keep track of calories lost. The calories lost can be counted within a timeframe: minutes, hours, and days. User can make and receive calls.

**Non-functional requirements with detail:**

**User Friendly:**

Speed, ease, and task processing will allow for a pleasant with regards to navigating the device as well as completing any tasks. For example, the device will be able to tell whether the user is walking or not through a step count notification as well as GPS location coordinates.

**Accessibility:**

The device must be connected to a mobile network to send/receive phone calls/text messages and any type of software update. The mobile network will also provide GPS capability to the device for exact coordinate locations.

**Availability:**

The device will have a battery life capable of staying powered on for exactly a week. This will give the user plenty of time to utilize the device before it needs to be charged again.

**Customization:**

User will be able to customize the device through the settings tool. The user will be able to install/remove apps when connected to a network. The device can receive updates for apps/software patches over a network as well. The device can be configurable to work with the user’s needs for any task mentioned in the functionality field above.

**Logging and Error Handling:**

Log all errors to persistent storage.

**Pluggable Roles:**

At various scenario points of several uses cases support the ability to customize the functionality of the system with a set of arbitrary rules that execute at that point or event.

**Security:**

User will need to log in using security authentication in or to use any feature of the device.

**Usability:**

***Human Factors***

The user will be able to view the display screen on the device with ease. Therefore:

Text font will be visible from near arms-length range.

**Reliability:**

If the device experiences any internal/hardware issues an error message will display a warning message with context to what needs to be fixed.

**Performance:**

As mentioned under human factors, the user will want a fast experience with working with tasks on the device and therefore that is why the device will prioritize resources based on the user’s preference.

**Supportability:**

***Adaptability***

As mentioned in performance, the device will be able to adapt to new circumstances based on how the user configures the device settings for processing resources. The device battery power settings can also be adjusted for a longer battery life extension.

**Implementation Constraints:**

There are limitations currently resting on the battery performance for the device which is being worked on and perfected. The navigation and user interface is also being enhance for ease of use.

**Purchased Components:**

wearable device, battery charger

**Interfaces:**

***Hardware and Interfaces***

\*Touch screen display

\*Glucose sensor reader

\*watch strap

**Software Interfaces:**

The device will come with an integrated software setup that can be customized over a mobile network. The user will have the ability to further enhance the device to their needs through the settings option.

Application-Specific Domain Business Rules

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Rule** | **Changeability** | **Source** |
| Rule: 1 | Purchaser discount rules, Examples: Employee 20% off,  Preferred Customer 10% off | High,  Each retailer uses different rules | Retailer Policy |
| Rule: 2 | Sale transaction-level discount rules:  Applies to pre-tax total.  Examples: 10% off if total greater than $100 USD. | High,  Each retailer uses different rules | Retailer Policy |
| Rule: 3 | Product line item level discount:  10% off watches this week | High,  Each retailer uses different rules | Retailer Policy |

**Legal Issues:**

User is responsible for activities done while wearing the device. We recommend not to use the device while drive/operating machinery or any activity that requires full attention. Please use the device responsibly.

**Information in Domains of Interest**

***Pricing***

In addition to the pricing rules mentioned in the domain rules section, note that our product does have a manufacturer retail price and will receive markdown prices which will be given over certain time periods.

**Glossary**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author |
| Inception draft | 9/30/2017 | First draft. To be refined primarily during elaboration. | Mohammed, Rishi, Jay, Andrew |

**Definitions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Term | Definition and Information | Format | Validation Rules | Aliases |
| Display | The display is the section on your fitness tracker which will show you the data you are monitoring during exercise on a small screen. | n/a | n/a | n/a |
| Fitness Tracker | A fitness tracker tracks and monitors physical activity and other physiological functions like breathing, sleep, calorie burn, and heart rate. | n/a | n/a | n/a |
| Heart Rate | The heart rate is how many beats per minute your heart takes. On average, a resting heart rate (meaning you are not being physically active) is around 72 beats/min. | Beats per minute format for reading heart rate | n/a | BPM |
| Notifications | When you wear your fitness tracker, you can set it to alert you of your progress, remind you to get up and move, or message you when you've reached your peak heart rate. | SMS message | n/a | SMS |
| Wearable Technology | A fitness tracker is an example of wearable technology, or more simply put, an electronic device you affix somewhere on the body. | n/a | n/a | n/a |
| Syncing | Syncing will transfer this data from the tracker to the smartphone or other device you'll read from. | n/a | Must be connected to a mobile network for it to occur. | n/a |

Elaboration Artifacts

|  |  |
| --- | --- |
| Artifact | Comment |
| Domain Model | The domain model is a visual representation of the conceptual classes, and their relevant attributes and associations which are related to your current iteration of development. |
| Design Model | The design model is a set of diagrams that allows for the understanding of the systems logical design of the current iteration. |
| Software Architecture Model | The software architecture model is a summary of the key architectural issues and how they are resolved in the current iteration |
| Data Model | The data model is the way data is mapped between object and non-object models and database schemas |
| Implementation Model | The implementation model is an area were test development takes place for coding. Continuous integration is also checked for consistency. |

**Use Cases**

1. Send texts to 911 and favorite contacts

2. Find location via GPS

3. Detect a heart attack

4. Check pulse

5. Check glucose (blood sugar) level

6. Provide clock functionality

7. Provide alarm functionality

8. Provide night light functionality

9. Allow network connectivity

10. Keep track of steps taken

11. Keep track of calories lost

12. Make calls

Use Case ID: UC – 001

Title: Send texts to 911 and favorite contacts

Description: User is able to send/receive texts to/from 911 and their favorite contacts. Text messages

are stored in the internal device memory.

Primary Actor: User

Preconditions: Device is powered on and has network access.

Postconditions: User is able to send and receive texts.

Main Success Scenario:

1. Device is powered on.

2. User accesses texting interface.

3. User sends/receives texts to/from

a. 911

b. favorite contacts

Extensions: None

Frequency of use: Above average

Owner: Team 8

Use Case ID: UC – 002

Title: Find location via GPS

Description: User has the ability to share their location via GPS.

Primary Actor: User

Preconditions: Device is powered on and has network access. GPS location sharing enabled.

Postconditions: User is able to share their GPS location.

Main Success Scenario:

1. Device is powered on.

2. User shares their GPS location with their favorite contacts.

Extensions: None

Frequency of use: Below average

Owner: Team 8

Use Case ID: UC – 003

Title: Detect a heart attack

Description: The device is able to detect the symptoms of a heart attack. The device will send an

emergency text message to 911.

Primary Actor: User

Preconditions: Device is powered on and has network access. Device is strapped to the user’s wrist.

Postconditions: Heart attack is detected and an emergency text message is sent to 911.

Main Success Scenario:

1. Device is powered on.

2. Device detects the symptoms of a heart attack.

3. Device sends an emergency text message to 911.

Extensions: None

Frequency of use: Below average

Owner: Team 8

Use Case ID: UC – 004

Title: Check pulse

Description: The user is able to check their pulse. History of pulse checks are stored in the internal

device memory.

Primary Actor: User

Preconditions: Device is powered on and strapped to the user’s wrist.

Postconditions: The user checks their pulse.

Main Success Scenario:

1. Device is powered on.

2. User accesses “check pulse” interface.

3. User checks their pulse.

Extensions: None

Frequency of use: Average

Owner: Team 8

Use Case ID: UC – 005

Title: Check glucose (blood sugar) level

Description: The user is able to check their glucose (blood sugar) level via skin.

Primary Actor: User

Preconditions: Device is powered on and strapped to the user’s wrist.

Postconditions: The user checks their glucose (blood sugar) level

Main Success Scenario:

1. Device is powered on.

2. User accesses “check glucose (blood sugar) level” interface.

3. User checks their glucose (blood sugar) level.

Extensions: None

Frequency of use: Average

Owner: Team 8

Use Case ID: UC – 006

Title: Provide clock functionality

Description: The device displays the time in 12-hour or 24-hour format.

Primary Actor: System

Preconditions: Device is powered on. User chooses 12-hour or 24-hour format.

Postconditions: Device displays the time to the user.

Main Success Scenario:

1. Device is powered on.

2. Device displays the time.

Extensions: None

Frequency of use: Above average

Owner: Team 8

Use Case ID: UC – 007

Title: Provide alarm functionality

Description: The user is able to set alarm(s).

Primary Actor: User

Preconditions: Device is powered on.

Postconditions: The user sets alarm(s).

Main Success Scenario:

1. Device is powered on.

2. User accesses alarm interface.

3. User sets alarm(s).

Extensions: None

Frequency of use: Average

Owner: Team 8

Use Case ID: UC – 008

Title: Provide voice-recognition functionality

Description: The user is able to access applications through voice recognition.

Primary Actor: User

Preconditions: Device is powered on. User has performed voice recognition initialization.

Postconditions: The user accesses applications by voice.

Main Success Scenario:

1. Device is powered on.

2. User enables voice-recognition by touch or voice.

3. User says a command.

4. User accesses applications.

Extensions: None

Owner: Team 8

Use Case ID: UC – 009

Title: Allow network connectivity

Description: Device is able to connect to a network.

Primary Actor: System

Preconditions: Device is powered on. User enables network connectivity. User has a network service

provider.

Postconditions: Device is connected to a network.

Main Success Scenario:

1. Device is powered on.

2. Device is connected to a network.

Extensions: None

Frequency of use: Above average

Owner: Team 8

Use Case ID: UC – 0010

Title: Keep track of steps taken

Description: The user is able to keep track of steps taken. The steps can be counted within a timeframe:

minutes, hours, and days.

Primary Actor: User

Preconditions: Device is powered on and strapped to the user’s wrist.

Postconditions: The user keeps track of steps taken.

Main Success Scenario:

1. Device is powered on.

2. User accesses the “step counter” interface.

3. User initiates the step counter application.

4. User keeps track of steps taken.

Extensions: None

Frequency of use: Average

Owner: Team 8

Use Case ID: UC – 0011

Title: Keep track of calories lost

Description: The user is able to keep track of calories lost. The calories lost can be counted within a

timeframe: minutes, hours, and days.

Primary Actor: User

Preconditions: Device is powered on and strapped to the user’s wrist.

Postconditions: The user keeps track of calories lost.

Main Success Scenario:

1. Device is powered on.

2. User accesses the “calorie counter” interface.

3. User initiates the calorie counter application.

4. User keeps track of calories lost.

Extensions: None

Frequency of use: Average

Owner: Team 8

Use Case ID: UC – 0012

Title: Make/receive calls

Description: User can make and receive calls.

Primary Actor: User

Preconditions: Device is powered on and has network access. User has a network service provider.

Postconditions: The user makes/receives calls.

Main Success Scenario:

Make calls:

1. Device is powered on.

2. User accesses the call interface.

3. User inputs a phone number or a contact.

4. User makes a call.

Receive calls:

1. Device is powered on.

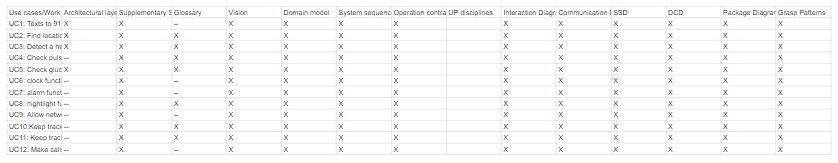
2. Incoming call is shown on the main display.

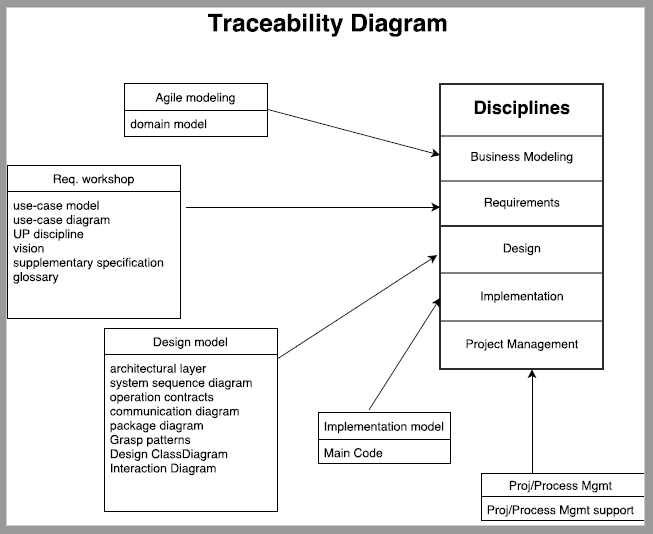
3. User receives a call.

Extensions: None

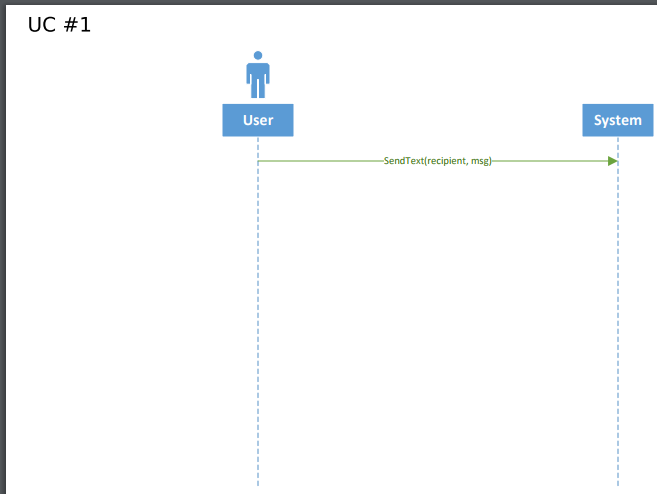
Frequency of use: Above average

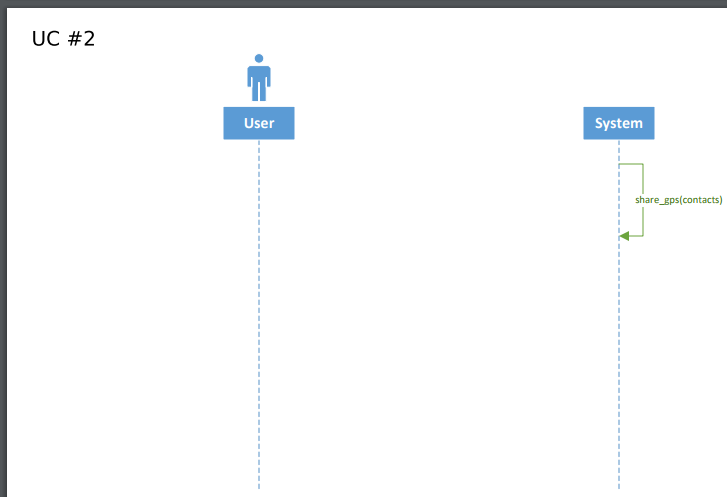
Owner: Team 8

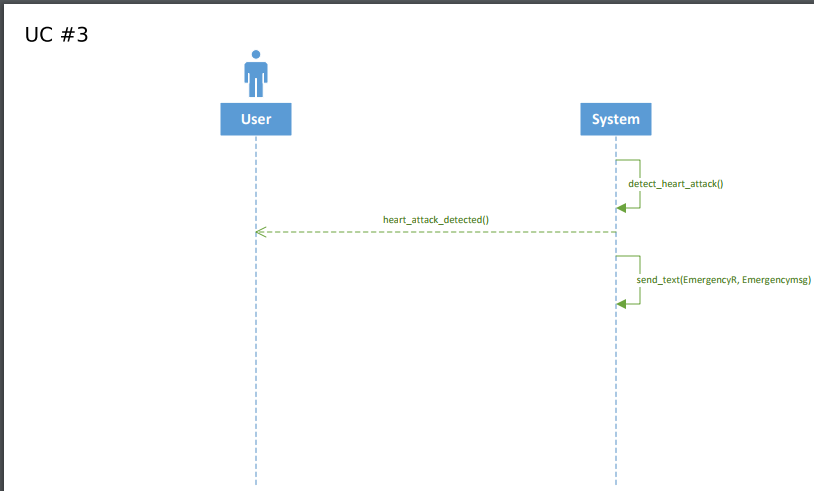
**Traceability Matrix**

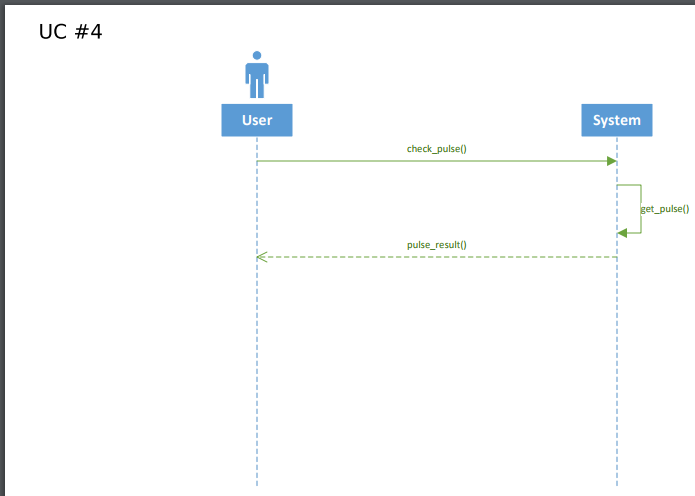


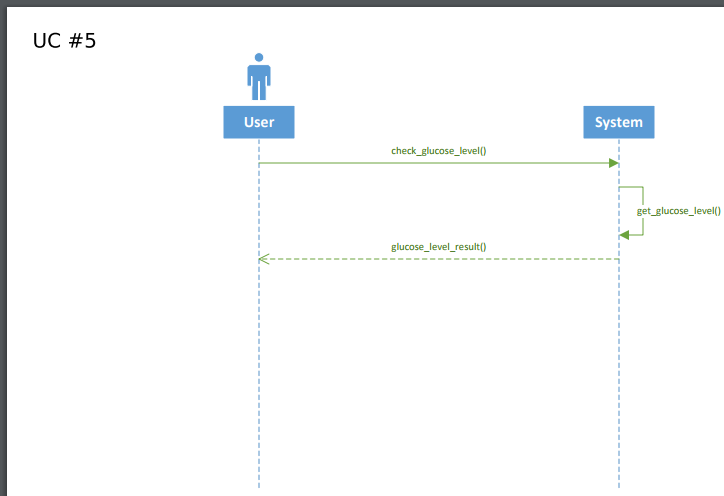
**Low Level SSD**

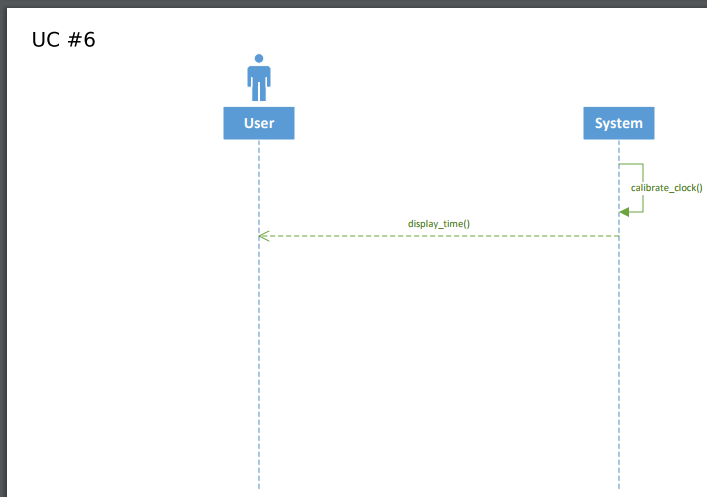


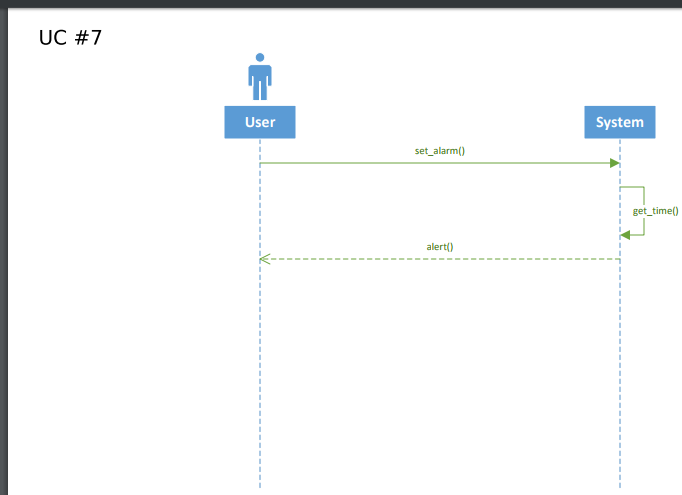


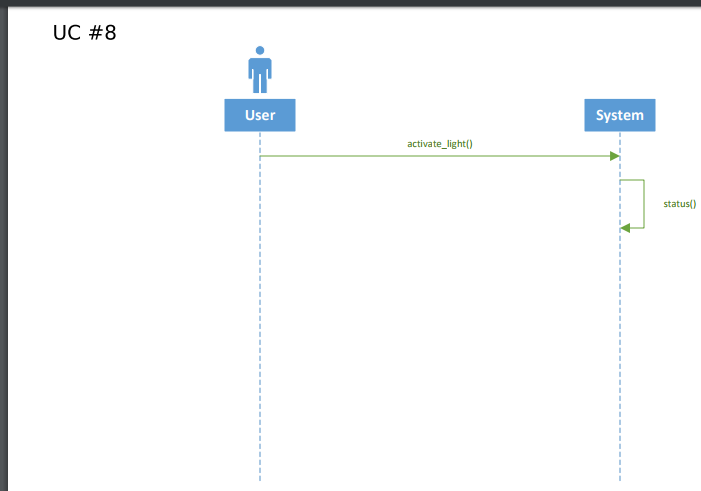


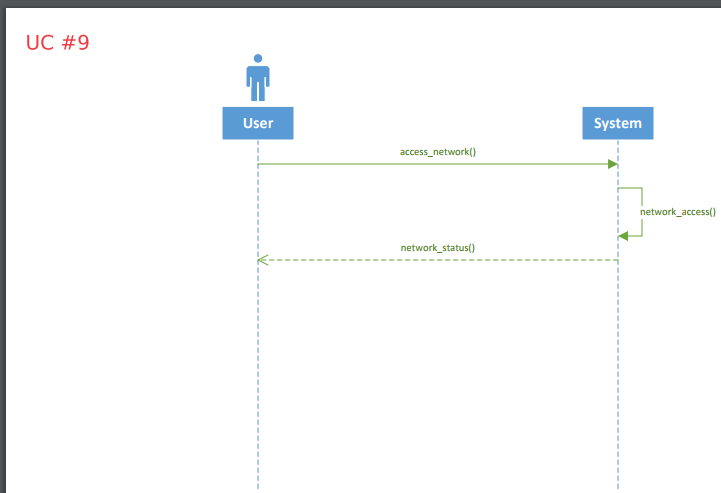


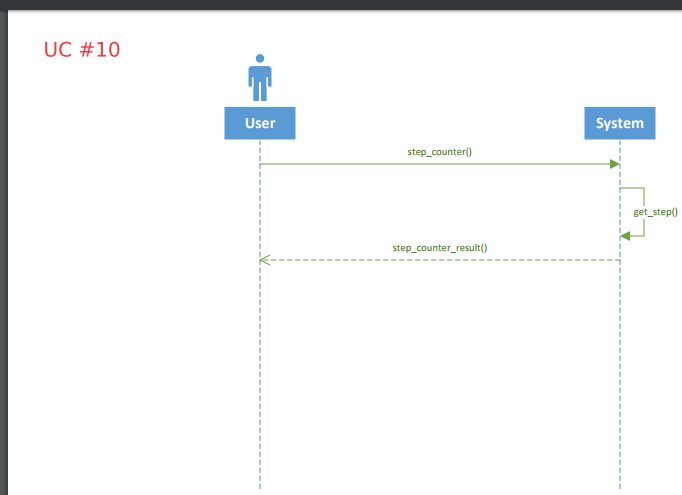


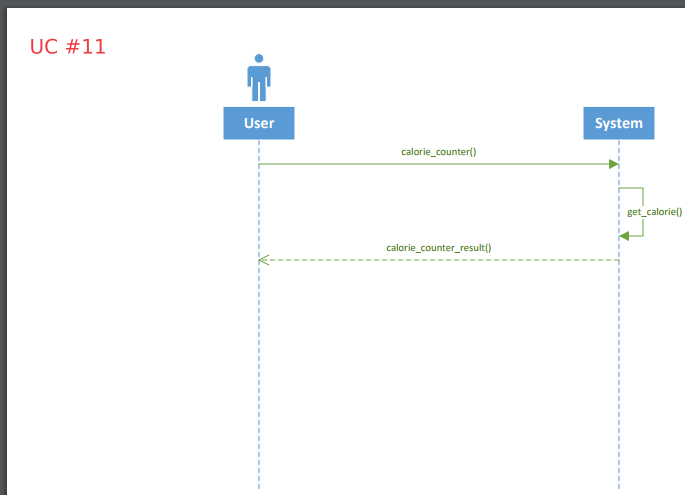


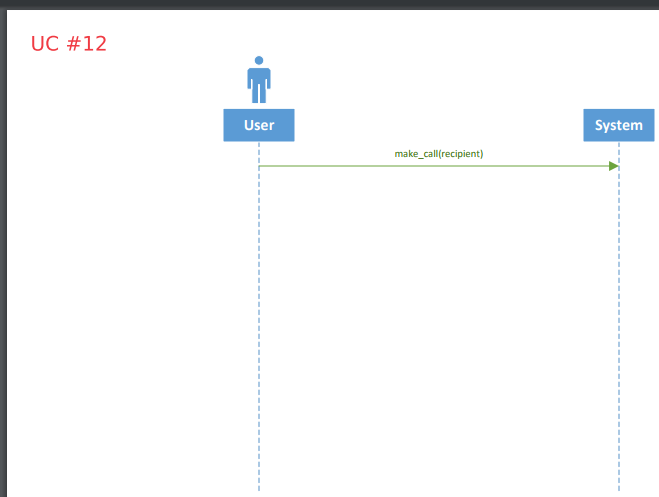


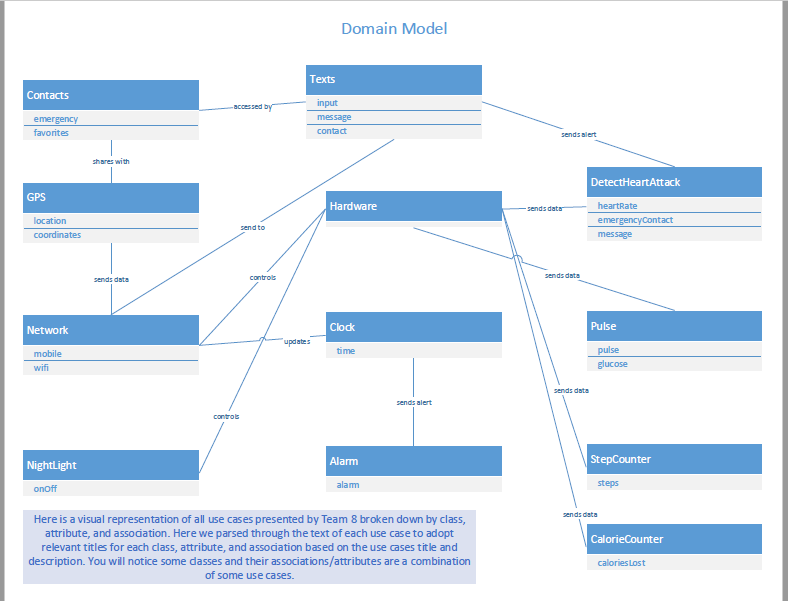




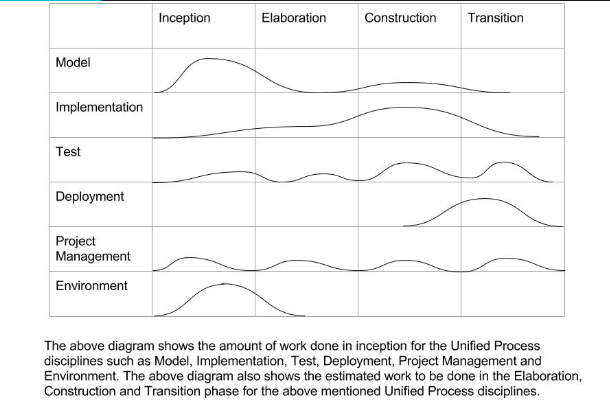




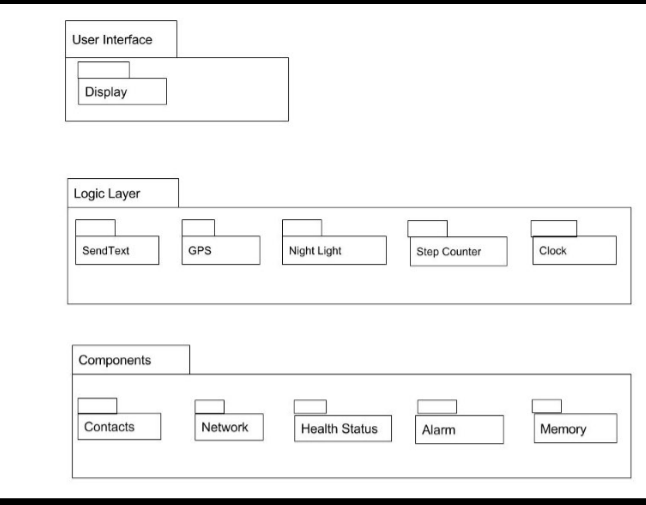




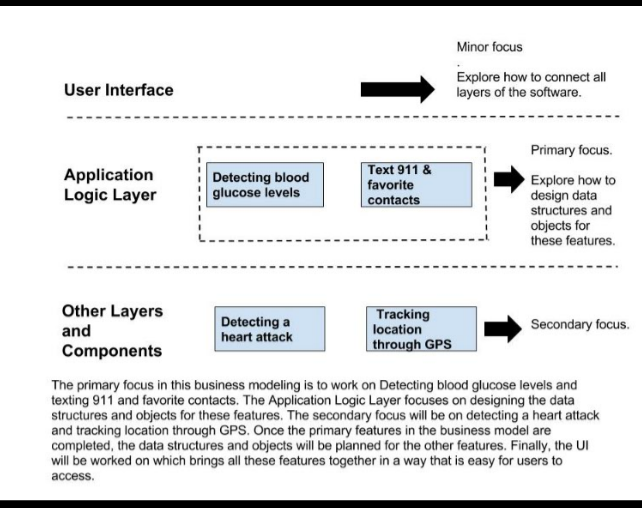
**UP Disciplines**



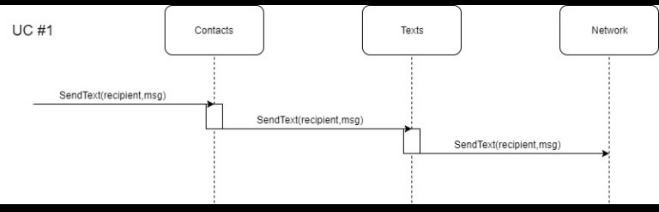
**Package Diagram**

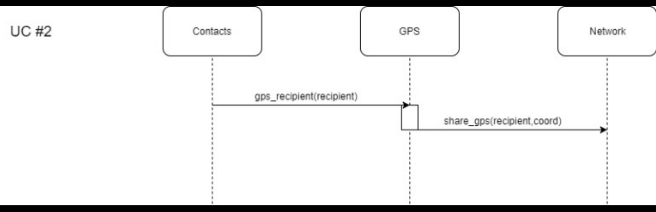


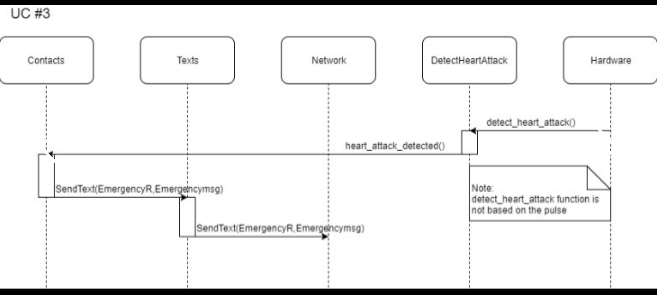
**Architectural Layer**

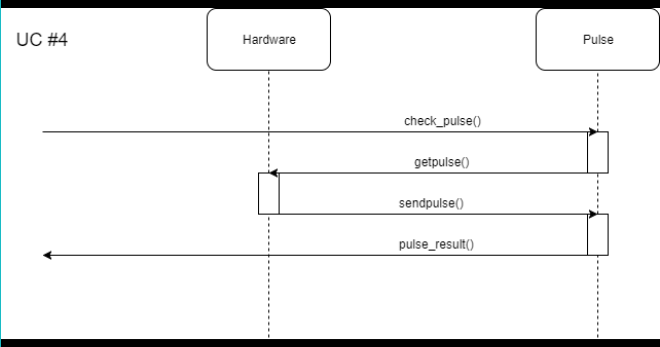


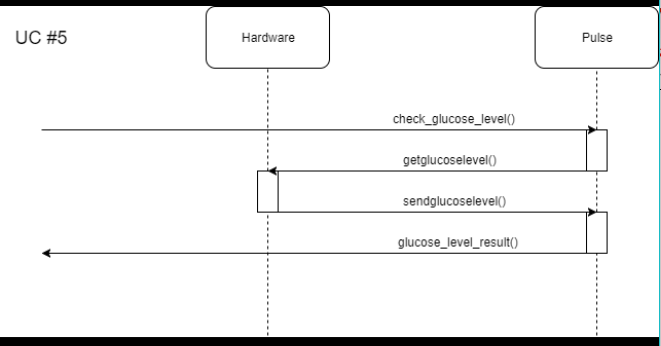
**Interaction Diagram**

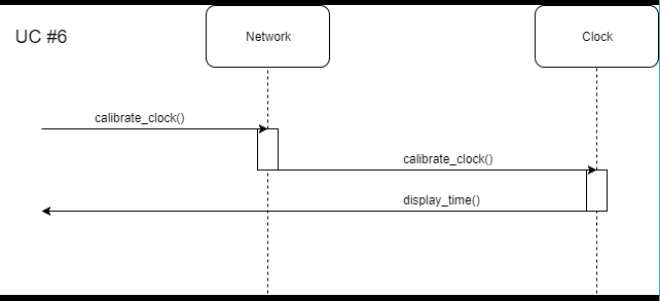


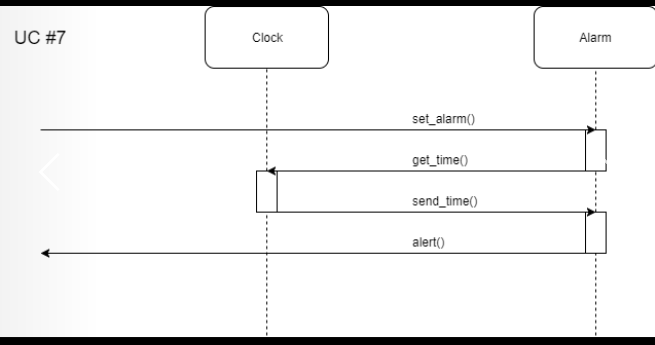


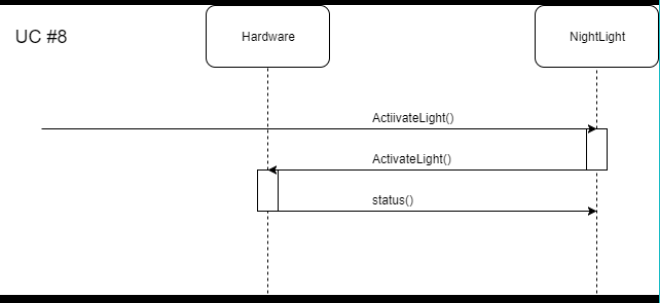


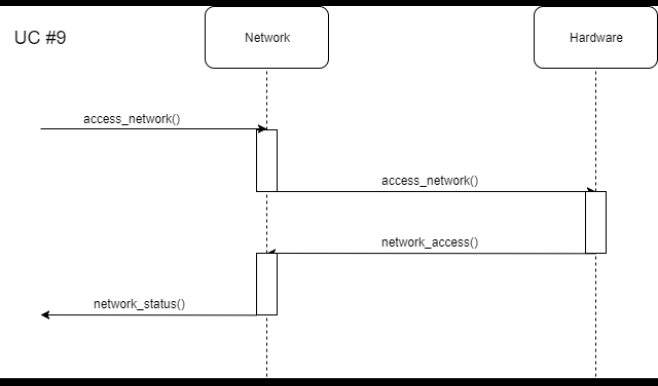


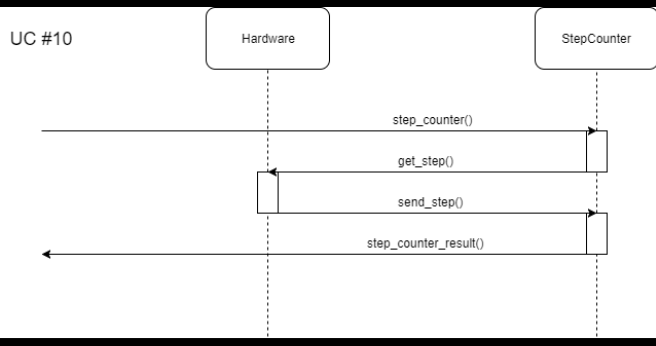


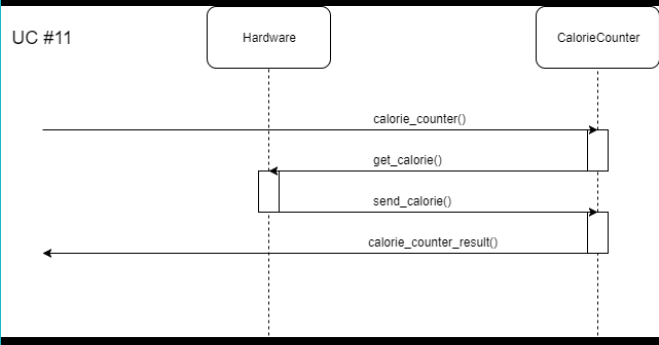


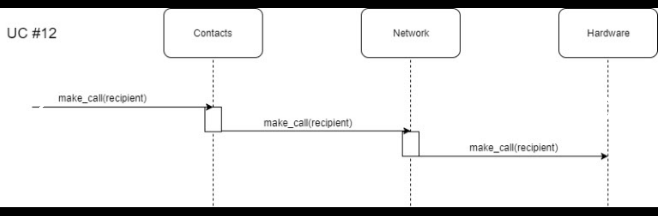






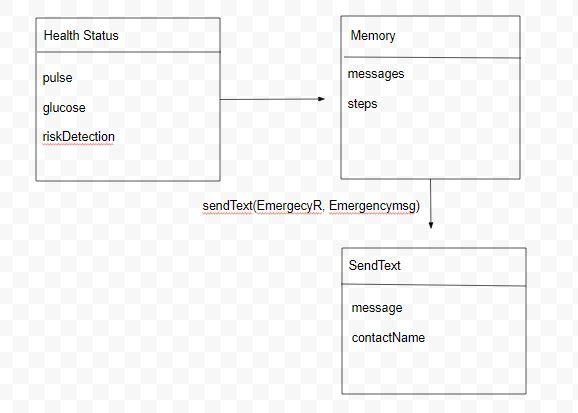






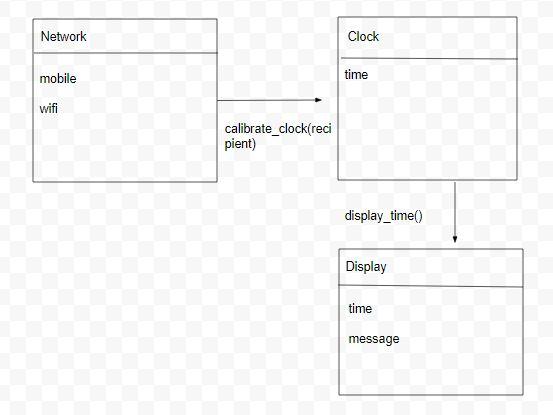
**GRASP Patterns**

1. **Expert Pattern**



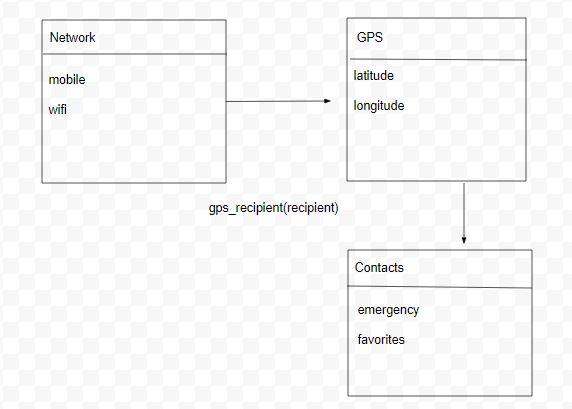
This is an example of the GRASP Expert pattern. In this example we can see that the class HealthStatus has all the necessary information needed to send an emergency text. An emergency text can only be sent if the Memory class is fed any risks which are detected by the HealthStatus class. The Memory class then communicates with the SendText class to send the emergency text. Thus in this case, the HealthStatus class is the expert as it contains all the vital health information needed for the emergency texts.

1. Low Coupling:

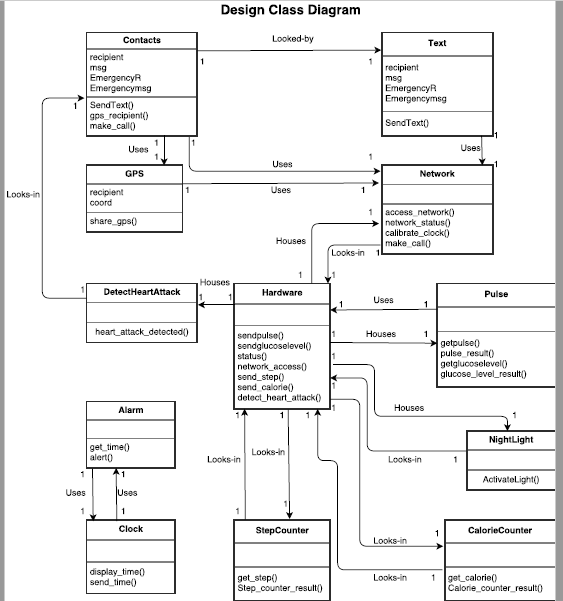


Low coupling is used to reduce the functionalities of the classes and simplify them. By using low coupling, the classes have less functions individually and are instead distributed among additional classes. In this example, for the user to view the time, the Network class sends the time to a different class called Clock which in turn displays it to the user. In this case, the Network class does not directly share the time with the Display class.

1. Controller Pattern:

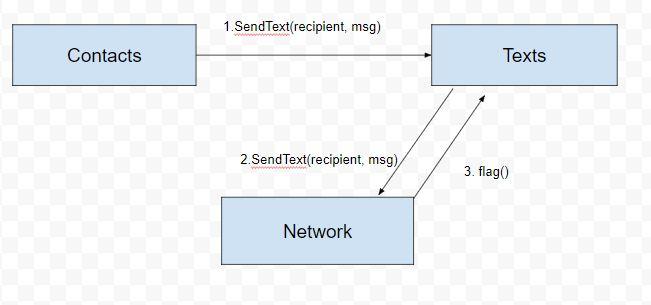


Controller Pattern is an example where a class controls the functionalities of another class. In this case, the class GPS will not be able to function without the Network class. Without an active network, the device will not be able to find its location.

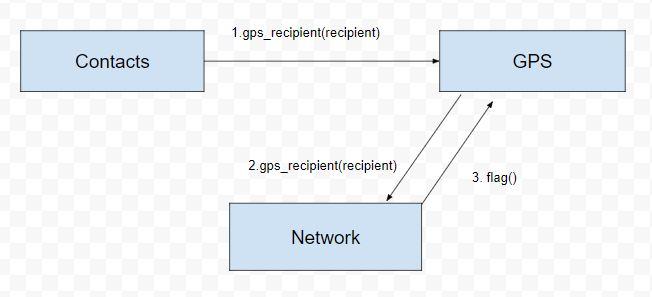


**Communication Diagrams**

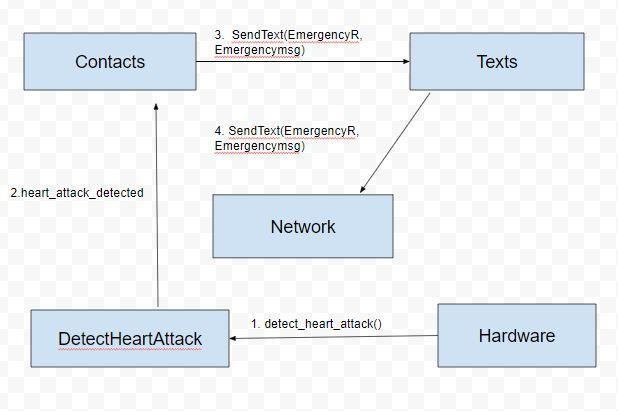
1. UC: 001 Send texts to 911 and favorite contacts



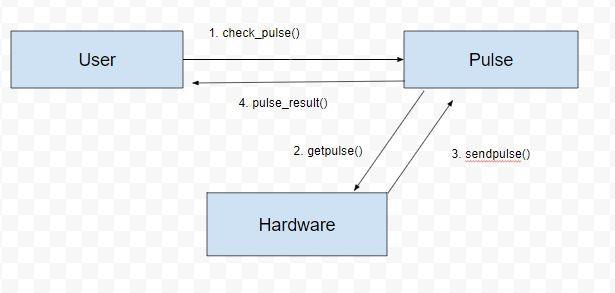
1. UC:002 Find location via GPS



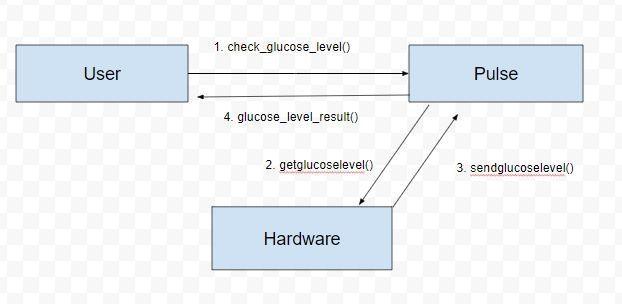
1. UC:003 Detect a heart attack



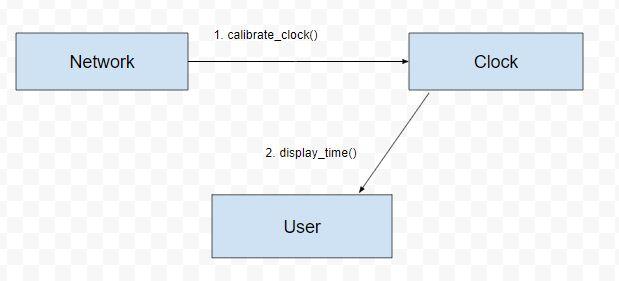
1. UC:004 Check pulse



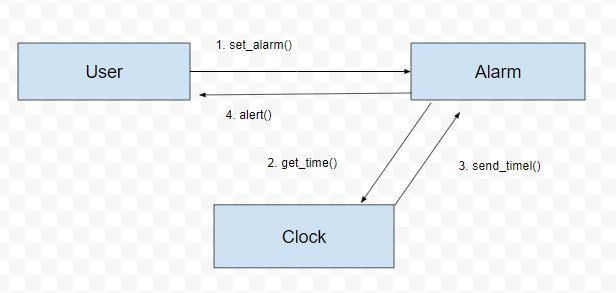
1. UC:005 Check glucose (blood sugar) level



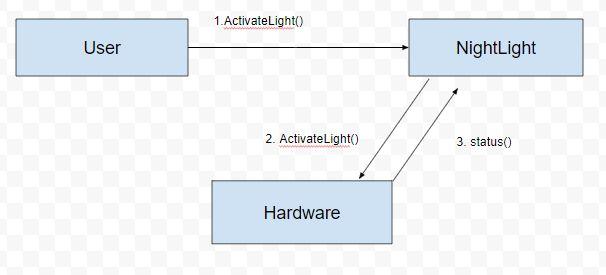
1. UC:006 Provide clock functionality



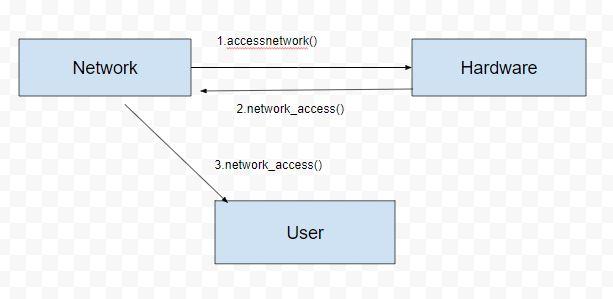
1. UC:007 Provide alarm functionality



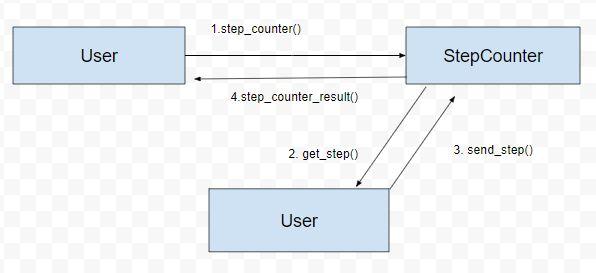
1. UC:008 Provide night light functionality



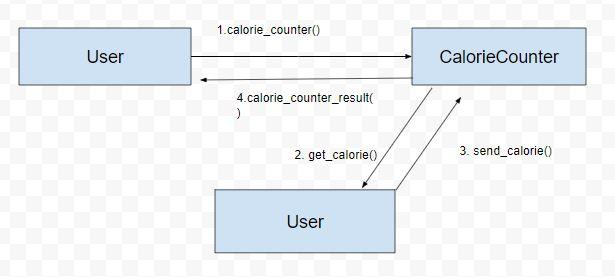
1. UC:009 Allow network connectivity



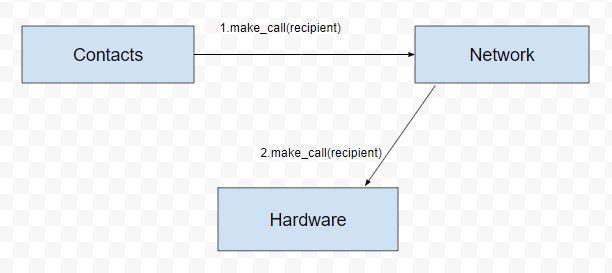
1. UC:010 Keep track of steps taken



1. UC:011 Keep track of calories lost



1. UC:012 Make calls



**Operation Contracts**

**Contract CO1: send\_text**

**Operation:** SendText(recipient,mgs)

**Cross References:** Use Cases: Send texts to 911/contacts (UC-001)

Detect a heart attack (UC-003)

**Preconditions:** Device has network access.

**Postconditions:** A SendText instance sText was created.

sText was associated with Network.

**Contract CO2: share\_gps**

**Operation:** share\_gps(recipient, coord)

**Cross References:** Use Cases: Find location via GPS (UC-002)

**Preconditions:** Device has network access. GPS location sharing enabled.

**Postconditions:** A GPS instance gps was created.

Gps was associated with Network.

**Contract CO3: detect\_heart\_attack**

**Operation:** detect\_heart\_attack()

**Cross References:** Use Cases: Detect a heart attack (UC-003)

**Preconditions:** Device has network access. Device is strapped to the user’s wrist.

**Postconditions:** A DetectHeartAttack instance dha was created.

dha was associated with HealthMonitor

**Contract CO4: check\_pulse**

**Operation:** check\_pulse()

**Cross References:** Use Cases: Check pulse (UC-004)

**Preconditions:** Device is strapped to the user’s wrist.

**Postconditions:** A HealthMonitor instance hm was created.

hm was associated with heartRate.

**Contract CO5: pulse\_result**

**Operation:** pulse\_result()

**Cross References:** Use Cases: Check pulse (UC-004)

**Preconditions:** Device is strapped to the user’s wrist.

**Postconditions:** A HealthMonitor instance hm was created.

hm was associated with heartRate.

**Contract CO6: check\_glucose\_level**

**Operation:** check\_glucose\_level()

**Cross References:** Use Cases: Check glucose level (UC-005)

**Preconditions:** Device is strapped to the user’s wrist

**Postconditions:** A HealthMonitor instance hm was created.

hm was associated with heartRate.

**Contract CO7: glucose\_level\_result**

**Operation:** glucose\_level\_result()

**Cross References:** Use Cases: Check glucose level (UC-005)

**Preconditions:** Device is strapped to the user’s wrist.

**Postconditions:** A HealthMonitor instance hm was created.

hm was associated with heartRate.

**Contract CO8: display\_time**

**Operation:** display\_time()

**Cross References:** Use Cases: Provide clock functionality (UC-006)

**Preconditions:** None.

**Postconditions:** a Clock instance clock was created

**Contract CO9: set\_alarm**

**Operation:** set\_alarm()

**Cross References:** Use Cases: Provide alarm functionality (UC-007)

**Preconditions:** None.

**Postconditions:** a Clock instance clock was created.

**Contract CO10: ActivateLight**

**Operation:** ActivateLight()

**Cross References:** Use Cases: provides night light functionality (UC-0008)

**Preconditions:** Device is powered on.

**Postconditions:** none.

**Contract CO11: access network**

**Operation:** access\_network()

**Cross References:** Use Cases: Allow network connectivity (UC-009)

**Preconditions:** User has performed initialization

**Postconditions:** None.

**Contract CO12: connect\_to\_network**

**Operation:** network\_access()

**Cross References:** Use Cases: Allow network connectivity (UC-009)

**Preconditions:** Device has network access.

**Postconditions:** A Network instance net was created.

**Contract CO13: step\_counter**

**Operation:** step\_counter()

**Cross References:** Use Cases: Keep track of steps taken (UC--0010)

**Preconditions:** Device is strapped to the user’s wrist.

**Postconditions:** A CalorieTracker instance ct was created

**Contract CO14: step\_counter\_result**

**Operation:** step\_counter\_result()

**Cross References:** Use Cases: Keep track of steps taken (UC--0010)

**Preconditions:** Device is strapped to the user’s wrist.

**Postconditions:** A CalorieTracker instance ct was created

**Contract CO15: calorie\_counter**

**Operation:** calorie\_counter()

**Cross References:** Use Cases: Keep track of calories lost (UC--0011)

**Preconditions:** Device is strapped to the user’s wrist.

**Postconditions:** A CalorieTracker instance ct was created

ct was associated with HealthMonitor.

**Contract CO16: calorie\_counter\_result**

**Operation:** calorie\_counter\_result()

**Cross References:** Use Cases: Keep track of calories lost (UC--0011)

**Preconditions:** Device is strapped to the user’s wrist.

**Postconditions:** A CalorieTracker instance ct was created

ct was associated with HealthMonitor.

**Contract CO17: make\_call**

**Operation:** make\_call(recipient)

**Cross References:** Use Cases: Make calls (UC-0012)

**Preconditions:** Device has network access.

**Postconditions:** A VoiceCall instance vc was created.

vc was associated with Network.

**Contract CO18: connect\_call**

**Operation:** make\_call(recipient)

**Cross References:** Use Cases: Make calls (UC-0012)

**Preconditions:** Device has network access.

**Postconditions:** A VoiceCall instance vc was created.

vc was associated with Network.

**Lesson Learned**

**Mohammed Hassnain:**

Our team has continued this project by adding the business model/requirements portion of the disciplines which will align with the development case requirements that were necessary for the Fit Band wearable device to be able to complete the task features listed in the project initiation. We created each of the inception artifacts to essentially show the target system features with all of its external systems. It also shows the potential components inside of the system and their interactions. Using this we created various models for the first iteration level. Using this information, we developed a detailed requirement business model that details the progress of development for each step of the process for the wearable device. Our application is a combination of work products to build a wearable device called the Fit Band device or Fit Band for short. The device will be built to automatically behave as preprogrammed. It will have several features listed in the use cases. We worked together to develop each artifact and accomplish a singular goal for a product. This product still needs refinements which will be further dealt with in the elaboration iteration phase. Each member of our group has contributed to the iteration phase with ideas and suggestions that will ultimately shape how our product is seen in its next stage.

**Andrew Ross:**

Going into this project, I had no idea what to expect. I have never particularly enjoyed the software design process, so I was expecting a lot of information to enter through one ear and out of the other. After we began our first iteration of the Inception phase I realized I was wrong. Initially, my method of development has always resembled the waterfall model and I would work extremely hard to complete all requirements immediately to complete my assignment in the required amount of time. After learning about the unified process and implementing it within our project allowed for clear goals to be set and met in an effective time frame. I learned the difference between the different phases (inception, elaboration, etc.), disciplines (requirements, analysis, design, etc.), and the key to taking quality progress the iterations that take place across the phases and disciplines. On top of learning more about UP, I have learned the importance of readability in terms of diagrams, notation, and summaries. This project taught me that even though we are all intelligent people, it does not mean everyone is able to understand our thought process and in order the project to thrive we must adapt and provide guidelines in the decisions we make. Getting deeper into the elaboration phase I began to realize how important synchronization is between each artifact. After this course, I see value in UP when before it was just a buzzword that I heard thrown around.

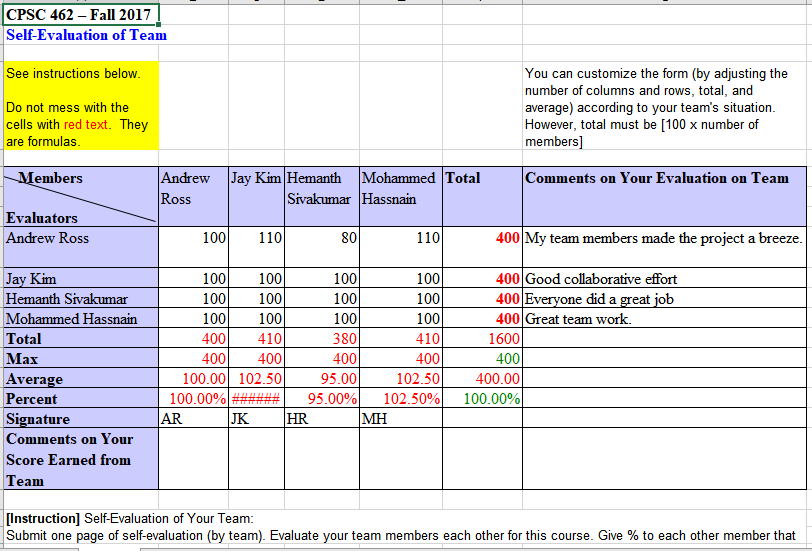
**Jay Kim:**

Throughout the course of the first homework, there were many lessons learned. One of great importance was frequent discussion. This helped the team maintain the same train of thought and focus. It also allowed for greater productivity and efficiency. The frequent discussions were usually followed by good collaboration. A team member would create a draft of a work product, and as a group, the team would discuss and make necessary changes. Although refining a work product is unavoidable, good collaboration amongst the team members certainly decreased the amount of extra work performed. Another lesson learned is producing quality work for each and every work product. Although many of the work products are similar and related, each work product depicts a significant aspect. These work products should not be rushed. Careful thinking must be put forth to ensure that these artifacts are traceable. In addition, these work products should not be divided only to be created individually. This can disrupt the flow of traceability as well as hinder good collaboration. The lesson learned here is that each work product must have the entire team’s involvement. Only then can the progress of the project flow smoothly and make work products meaningful. The first homework was a great experience in the UP and iterative development.

**Hemanth Sivakumar:**

This was a challenging task in the beginning for the fact that the requirements were a bit too few and we had to be organized as team. Communication was key as we all had other classes and exams and projects to be done. We did a very good job at delegating and meeting soft deadlines set by the group. In our group texts there was constant communication and we were always ready to help each other out when needed. The project seemed less intimidating as the weeks went by and as our group started to break down the process into smaller steps which made it easier on all of us. The main thing I learnt from this project are the very many documents that go along creating a software. I always thought that documentation was boring and pointless but after reading the book and listening to lectures, I understood the value of documentation and how it helps to organize the creation of a software. Another thing I learnt was that in any group, communication is key. Had our group not communicated, we would not have been able to keep up with the assignment. Even when we had some questions on certain parts of the documents, we always asked the other members in the group for help and they stepped in. Last but not least, we over did a few use cases and other requirements, as not all of them were required for this submission. It went against the iterative model. This was because once we started on a task, we wanted to ensure we were all the way done. Overall, this was a great project to learn about software design and to work as a team.

**Team Evaluation Template**



**Team Charter Template**

|  |  |  |  |
| --- | --- | --- | --- |
| **Course Title** | CPSC 462 (Sect03) Software Design |  | *All team members participated in the creation of this charter and agree with its content.* ***Date*** *08/30/2017cpsc462\_Team\_Charter\_Template* |
| **Instructor** | Dr. Chang-Hyun Jo |  |
| **Course Dates** | 08/21/2017 – 12/08/2017 |  |

**Team Members** (Contact Information)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Address (city, state, country) | Phone | Cell | Email |
| Andrew Ross | Fullerton, CA | xxx-xxx-xxxx | 657-253-0321 | [anross@fullerton.edu](mailto:anross@fullerton.edu) |
| Mohammed Hassnain | Corona, CA | xxx-xxx-xxxx | 213-810-3473 | [Mohammed-17@csu.fullerton.edu](mailto:Mohammed-17@csu.fullerton.edu) |
| Hemanth Sivakumar | Fullerton, CA | xxx-xxx-xxxx | 714-600-0114 | [hemanth@csu.fullerton.edu](mailto:hemanth@csu.fullerton.edu) |
| Jay Kim | Irvine, CA | xxx-xxx-xxxx | 949-351-2286 | [Jkim128@csu.fullerton.edu](mailto:Jkim128@csu.fullerton.edu) |

**Team Member Skill Inventory** (Areas individual members can contribute)

|  |  |
| --- | --- |
| Andrew Ross | * MS Office + Visio, etc * C++, Python, Powershell, etc * MS Server/Client, Linux * Server Administration (IIS, File, etc) |
| Mohammed Hassnain | * C++, C * MS Office * Linux |
| Hemanth Sivakumar | * MS Office * C++, Python |
| Jay Kim | * MS Office * C, C++ * Linux |

**Team Goals** (Project goals, team process goals, quality goals, etc.)

|  |
| --- |
| * Learn about software processes and software design and architecture. * Develop a strong, collaborative team to produce quality results in the amount of time specified. * Acquire practical knowledge from our teammates * Maintain great relationship between teammates. * Produce and deliver a good final paper to the professor. * Develop skills to facilitate future career goals. |

**Team Roles** (Define roles of each member to achieve goals)

|  |  |
| --- | --- |
| Andrew Ross  (Team Member) | * Relate the vision to the other artifacts * Define conceptual classes, attributes, and associations * Define elaboration artifacts |
| Mohammed Hassnain (Team Member) | * Build and update development case * Define inception artifacts * Refine inception artifacts * Write supplementary specification |
| Hemanth Sivakumar (Team Member) | * Initialize UP disciplines * Define architectural layers * Define use cases * Draw UP artifact influence |
| Jay Kim  (Team Member) | * Define use cases * Define system sequence diagram * Define use case diagram |

**Ground Rules** (Meeting schedule/locations, attendance expectations, agenda, assignment completion, communication methods, etc.)

|  |
| --- |
| * Show up to class on time * Keep up to date with readings * Work cohesively as a unit * Finish requirements within the required time. * Have fun |

**Time Commitments/Availability** (Pacific Time)

|  |  |
| --- | --- |
| Andrew Ross | * Monday – Friday, 5:00 PM – 8:00 PM (After work/class) * Saturday – Sunday, Anytime (given no emergencies) * Note: Due to my job, I am required to constantly monitor my email/phone throughout the day so that is the best way to contact me. If our schedules conflict, we could always do webinar style with screensharing/remote control for our group meetings in the future. |
| Mohammed Hassnain | * Monday – Friday, 5:00 PM – 10:00 PM * Saturday – Sunday, 3:00 PM - 8:00 PM |
| Hemanth Sivakumar |  |
| Jay Kim | * Monday, Wednesday: After 5:00 PM * Tuesday, Thursday: After 8:15 PM * Friday - Sunday: Anytime with advanced notice |

**Conflict Management** (What are potential conflicts that might arise among or between team members during this course? How will team members deal with these and other conflicts?)

|  |
| --- |
| * In order to avoid conflict clear roles and responsibilities must be assigned, so that there is no confusion. * If a team member is not performing, we will speak as a group and resolve the issue. * If conflicts arise, we will speak as a group and resolve the issue. * All team members must set pride aside for the good of the group. |

**Risk Management** (What are potential barriers to the achievement of these goals?)

|  |
| --- |
| * Time constraint in the sense that we are students juggling multiple high level classes or our career * Unable to meet effectively in person * Different perspective (but can also be used as a strength) |

**Team Evaluation Criteria** (List evaluation criteria that will be used to evaluate team members objectively.)

|  |
| --- |
| * Our group’s sole objective is to create the best project possible and we will objectively grade each other based on the individual and group effort we put forth |

**Main C++ Code**

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

#include "Contacts.h”

#include "Texts.h”

#include "GPS.h”

#include "Network.h”

#include "DetectHeartAttack.h”

#include "Hardware.h”

#include "Pulse.h”

#include "Clock.h”

#include "Alarm.h”

#include "NightLight.h”

#include "StepCounter.h”

#include "CalorieCounter.h”

int main()

{

Contacts contacts;

Texts texts;

GPS gps;

Hardware hardware;

DetectHeartAttack heartAttack;

Pulse pulse;

Network network;

Clock clock;

Alarm alarm;

NightLight light;

StepCounter stepCounter;

bool watchOn = 1;

int choice;

string recipient;

string msg;

string EmergencyR;

string Emergencymsg;

while (watchOn)

{

// Display menu

cout << "----Fullerton Wear---- \n\n";

cout << "Choice: ";

cin >> choice;

if(choice=9)

{

// UC #9: allow network connectivity

// method to establish network connection via hardware

On = network.access\_network();

// method to display the status of the network connection

network.network\_status(On);

// Request processed now exiting

watchOn=0;

}

}

cout << "Request Completed. \n\n" << endl;

return 0;

}

**///////////// Header Files ///////////////**

**//Contacts.h**

#ifndef contact\_class

#define contact\_class

class Contacts

{

private:

string recipient;

string msg;

string EmergencyR;

string Emergencymsg;

public:

void SendText(string recipient, string msg)

{

// ...

}

void make\_call(string recipient)

{

// …

}

void gps\_recipient(string recipient)

{

//…

}

};

#endif

**//Texts.h**

#ifndef Texts\_class

#define Texts\_class

class Texts

{

private:

string recipient;

string msg;

string EmergencyR;

string Emergencymsg;

public:

void SendText(string recipient, string msg)

{

// ...

}

};

#endif

**//GPS.h**

#ifndef GPS\_class

#define GPS\_class

class GPS

{

private:

string recipient;

string coord;

public:

void share\_gps(string recipient, string coord)

{

// …

}

};

#endif

**//Network.h**

#ifndef Network\_class

#define Network\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

#include "Hardware.h"

using namespace std;

class Network

{

private:

string recipient;

bool On = 0;

public:

void calibrate\_clock()

{

// …

}

int access\_network()

{

Hardware hd;

cout << " Connecting to Mobile network...\n\n" << endl;

On=hd.network\_access();

return On;

}

void network\_status(bool On)

{

if(On)

{ cout << " Successfully Connected. \n\n" << endl; }

else

{ cout << "Connection Failure. \n\n" << endl;}

}

void make\_call(string recipient)

{

// …

}

};

#endif

**//DetectHeartAttack.h**

#ifndef Detect\_Heart\_Attack\_class

#define Detect\_Heart\_Attack\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

class DetectHeartAttack

{

public:

void heart\_attack\_detected()

{

// …

}

};

#endif

**//Hardware.h**

#ifndef Hardware\_class

#define Hardware\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

using namespace std;

class Hardware

{

public:

bool detect\_heart\_attack()

{

// …

}

void sendpulse()

{

// …

}

void sendglucoselevel()

{

// …

}

void status()

{

// …

}

void send\_step()

{

// …

}

void send\_calorie()

{

// …

}

int network\_access()

{

// Network connection is assumed to be always active

return 1;

}

};

#endif

**//Pulse.h**

#ifndef Pulse\_class

#define Pulse\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

class Pulse

{

public:

void getpulse()

{

// …

}

void pulse\_result()

{

// …

}

void getglucoselevel()

{

// …

}

void glucose\_level\_result()

{

// …

}

};

#endif

**//Clock.h**

#ifndef Clock\_class

#define Clock\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

class Clock

{

public:

void display\_time()

{

// …

}

void send\_time()

{

// …

}

};

#endif

**//Alarm.h**

#ifndef Alarm\_class

#define Alarm\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

Class Alarm

{

public:

void get\_time()

{

// …

}

void alert()

{

// …

}

};

#endif

**// NightLight.h**

#ifndef NightLight\_class

#define NightLight\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

class Nightlight

{

public:

void ActivateLight()

{

// …

}

};

#endif

**//StepCounter.h**

#ifndef StepCounter\_class

#define StepCounter\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

class StepCounter

{

public:

void get\_step()

{

// …

}

void step\_counter\_result()

{

// …

}

};

#endif

**//CalorieCounter.h**

#ifndef CalorieCounter\_class

#define CalorieCounter\_class

#include <iostream>

#include <string>

#include <fstream>

#include <mutex>

class CalorieCounter

{

public:

void get\_calorie()

{

// …

}

void calorie\_counter\_result()

{

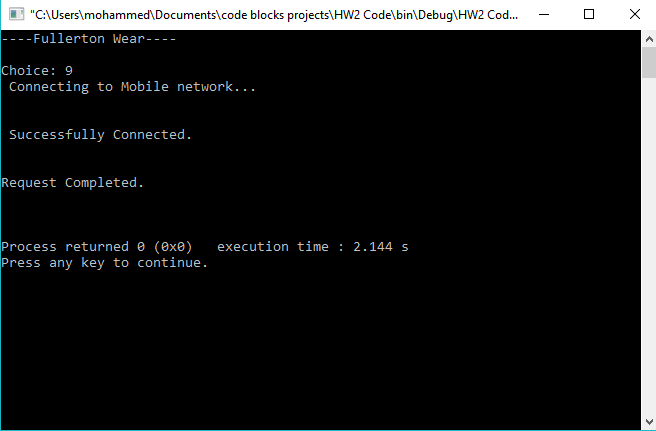
// …

}

};

#endif

**Output of Test Run for UC#:9**



**References**

1.Larman, Craig. Applying UML and patterns: an introduction to object-Oriented analysis and design and iterative development. Pearson, 2016.

2.“Free flowchart maker and diagrams online.” *Flowchart Maker & Online Diagram Software*, www.draw.io/.