Fedora+

by Fedora Tech

Team: Team Zipfile
Members:
Team Leader: Himanshu Patel | himanshup@csu.fullerton.edu
Kim-Lan Hoang | khoangl@csu.fullerton.edu
Jayro Alvarez | jayroalvarez@csu.fullerton.edu
Submission Date: 5/6/2019 | **Course**: CPSC 462- Software Design
Instructor: Dr. Swayam Pati

Revision History

Date	Status	Author
2/2/19	Created	All
2/2/19	Business Requirement and Vision	Jayro Alvarez
2/5/19	Functional and Non-F Requirements Created	Himanshu Patel
2/7/19	Use Case and Use Case Descriptions Created	Kim-Lan Hoang
2/8/19	**Revised** Use Cases to better fit entire project scope	All
2/9/19	Use Case Diagram	Jayro Alvarez
2/10/19	Start Elaboration Diagrams	All
2/15/19	**Revised** Elaboration Diagrams to better fit project scope	All
2/27/19	**Revised** Elaboration Iteration II Diagrams	All
3/11/19	**Revised** Business Related Documentation	All
3/13/19	SSDs, Operation Contracts, Glossary	Jayro Alvarez
3/13/19	Traceability Diagram	Kim-Lan Hoang
3/13/19	**Revised** Sequence and activity diagrams	Himanshu Patel
3/25/19	Package Diagrams, CRC Cards, Visibility	Jayro Alvarez/Kim-Lan Hoang
4/7/19	Apply GRASP, GoF, Rake to Diagrams	All
4/15/19	Forward/Backward Matrices	Himanshu Patel
4/25/19	**Gen. Revision Day, Review Whole Doc.**	All
4/30/19	Deployment Diagram & Activity Diagrams	Himanshu Patel
5/5/19	**Finalized** Charter and Review All	Jayro Alvarez

Meeting Minutes

Date of Meeting: 2/6/19	Time: 7:00-8:50
Meeting Notes: Finalized group and built team charter. We also reviewed chapter 1 and discussed the goals for the class and talked about our previous experiences with UML and the software engineering process. Finished meeting by planning how to work collaboratively as a team.	

Date of Meeting: 2/13/19	Time: 7:00-8:50
Meeting Notes: Went over what was going to be the functional and non-functional requirements of the target system. We discussed the Unified Process and what specific AGILE methods we would be using in our project.	

Date of Meeting: 2/20/19	Time: 7:00-8:50
Meeting Notes: Went over chapters 4-7 and what artifacts would be helpful for our project. Started discussing the actors, scenarios, and use cases that could be used. Constructed use cases based on most important requirements.	

Date of Meeting: 2/27/19	Time: 7:00-8:50
Meeting Notes: Presented our current work up to this date to our senior supervisor (Prof. Pati). Discussed our progress in the inception phase and decided we were ready to start elaboration phase. Ended meeting by discussing what artifacts would be helpful in created a domain model.	

Date of Meeting: 3/6/19	Time: 7:00-8:50
Meeting Notes: Went over chapters 10 and 11 and discussed the importance of traceability between all of our documented artifacts. Had our final discussion regarding if we needed any more use cases and if we had too many.	

Meeting Minutes

Date of Meeting: 3/20/19	Time: 7:00-8:50
Meeting Notes: Group started going over the initial plans for the next part of our project.	

Date of Meeting: 3/27/19	Time: 7:00-8:50
Meeting Notes: Started discussing the logical architecture that would be incorporated into the design. Also started discussing the following artifacts: Package Diagrams, CRC Cards, and Interaction Diagrams.	

Date of Meeting: 4/10/19	Time: 7:00-8:50
Meeting Notes: Went over chapter 14 and 15 and discussed what other artifacts would be useful in incorporating into the final project. Decided on converting the class diagrams into code and incorporating the GRASP patterns into our existing class diagram.	

Date of Meeting: 4/17/19	Time: 7:00-8:50
Meeting Notes: Went over chapters 17 and 18 and discussed what artifacts would be important to furthur show the important of the design patterns being incorporated into already-existing diagrams. We decided that GRASP patterns from the previous week would be sufficient to realize our use cases.	

Date of Meeting: 4/24/19	Time: 7:00-8:50
Meeting Notes: As the deadline was approaching we decided to focus and the finishing touches to our project. The following artifacts were chosen to be part of the project: Activity Diagrams w/Rake, forward/backwards compatibility matrices, applying GoF patterns, and applying stereotype concepts.	

Date of Meeting: 5/1/19	Time: 7:00-8:50
Meeting Notes: Started going over final details and individual assignments prior to presentation the following class day.	

Project Charter

Project Charter

Project Name: 462 Project

Date Charted: 5/5/19

Fedora +

Project Statement - A brief description of the problem or business case	**Goal Statement** - Define the mission and the target of the project
The proposal of this new product aims to reach new users and gain user data by providing a new smart-wearable which syncs with both iOS and Android devices.	Our mission is to create a new smart-wearable device that will aid users with monitering a variety of their health and fitness statistics through breakthrough advances in smart sensors, access and manipulate data on their smart phone, and interact seemlessly with the device's own apps.
Project Leader	**Project Approach**
Jayro Alvarez	Agile Approach

Project Team

Name	Role/Responsibilites
Jayro Alvarez	Project Leader/Project Planning/Project Architecture & Design
Himanshu Patel	Project Planning/Project Architecture & Design
Kim-Lan Hoang	Project Planning/Project Architecture & Design

Support Personnel

Name	Role
Swayam Pati	Senior Advisor

Project Approach	**Project Difficulties**
Our team decided on the agile approach as it is a very effective managing methodology that gives alot of flexity.	Difficulties we came across were not anything major. Just some small issues with communicating the handling of certain sections in the documentation.

Project Artifacts		**Timelines**
Supplementary Specification	Collaboration Diagram	Project Start: 2/2/19
Business Case	CRC Cards	Project Planning Review: 2/4/19
Non-Functional Requirements	Class Diagrams to Code	Planning Complete: 2/17/19
Use Case Diagram	Grasp & GoF Patterns	Elaboration of Requirements: 3/1/19
Class Diagrams	Visibility	Continue Elaboration: 4/5/19
Project Vision	Activity Diagrams	End of Elaboration Cycle: 5/4/19
Functional Requirements	Backwards/Forwards Matrices	
Use Case Descriptions	Package Diagrams	
Domain Models	Deployment Diagram	

Project Scope	**Other Project Information** - e.g. project description, constraints, etc.	
Our project plan will efficiently describe the planning and architecture of our product.	**Project Description**: Complete planing of new smart-wearable-Fedora+.	**Constraints**: Limited name recognition (first product), getting integration approval from major smart phone makers
Start Date	**Estimate Completion Date**	
2/2/19	5/4/19	

Project Milestones	1	2	3	4	5	6
Description:	Inception I	Elaboration I	Inception II	Elaboration II	Elab. III	Elab. IV
Date:	2/2/19	2/27/19	3/1/19	3/11/19	4/5/19	5/4/19

Signatures - The signatures of the people below document approval of the formal Project Charter

	Signature	Date
Project Leader:	*Jayro Alvarez*	5/5/19
Project partner:	*Himanshu Patel*	5/5/19
Project partner:	*Kim-Lan Hoang*	5/5/19

The Project Leader is empowered by this charter to proceed with the project as outlined above

Table of Contents

Revision History	1
Meeting Minutes	2-3
Project Charter	4
Table of Contents	5
Project Plan	6
Iteration I	7-22
a. Inception I	7-9
b. Elaboration I	9-22
Iteration II	23-37
a. Inception II	23
b. Elaboration II	23-37
Glossary	38
Elaboration III	39
Elaboration IV	49
Lessons Learnt	59

Project Plan

Task	Assigned To	Start Date	Due Date	Completed Date	Dependencies	Status	Comments
Project Cover Page	ALL	2/2/19	3/11/19	2/15/19	N/A	Completed	
Revision History	ALL	2/2/19	3/11/19	3/12/19	N/A	Completed	
Meeting Minutes	Himanshu Patel	2/4/19	3/11/19	3/11/19	N/A	Completed	
Project Charter	ALL	2/4/19	3/11/19	3/11/19	N/A	Completed	Attach from seperate doc.
Document Table of Contents	Himanshu Patel	2/2/19	3/11/19	3/11/19	N/A	Completed	
Project Vision	Jayro Alvarez	2/2/19	2/13/19	2/10/19	N/A	Completed	
Business Case	Jayro Alvarez	2/2/19	2/13/19	2/10/19	N/A	Completed	
Business Requirements	Jayro Alvarez	2/2/19	2/13/19	2/10/19	N/A	Completed	
Functional and Non-Functional Req's	Himanshu Patel	2/5/19	2/20/19	2/12/19	Inception 1	Completed	
Use Cases for Requirements	Kim-Lan Hoang	2/5/19	2/20/19	2/12/19	Inception 1	Completed	
Use Case Diagram	Jayro Alvarez	2/5/19	2/20/19	2/12/19	Inception 1	Completed	
Black Box Use Cases	Kim-Lan Hoang	2/5/19	2/20/19	2/12/19	Inception 1	Completed	
Black Box Use Case Diagram	Jayro Alvarez	2/7/19	2/27/19	2/12/19	Inception 1	Completed	
Fully Dressed Use Case	Jayro Alvarez	2/7/19	2/27/19	2/12/19	Black Box Use Cases	Completed	
Casual Use Cases	Himanshu Patel	2/7/19	2/27/19	2/14/19	Black Box Use Cases	Completed	
Brief Use Cases	Kim-Lan Hoang	2/7/19	2/27/19	2/14/19	Black Box Use Cases	Completed	
Voice Assistant Diagram	Jayro Alvarez	2/10/19	3/6/19	2/14/19	Use Case Descript.	Completed	
Count Heart Pulse Diagram	Kim-Lan Hoang	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Check Pulse for Irregularites Diagram	Himanshu Patel	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Fedora Component Diagram	Kim-Lan Hoang	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Domain Model	Jayro Alvarez	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Class Diagram	Jayro Alvarez	2/15/19	3/6/19	3/4/19	Domain Model	Completed	
Collaboration Diagram	Kim-Lan Hoang	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Sequence Diagram	Himanshu Patel	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Component Diagram	Kim-Lan Hoang	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Activity Diagram	Kim-Lan Hoang	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Domain Model (Iteration II)	Jayro Alvarez	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Class Diagram (Iteration II)	Jayro Alvarez	2/15/19	3/6/19	3/4/19	Domain Model	Completed	
Collaboration Diagram (Iteration II)	Kim-Lan Hoang	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Sequence Diagram (Iteration II)	Himanshu Patel	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Component Diagram (Iteration II)	Kim-Lan Hoang	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Activity Diagram (Iteration II)	Kim-Lan Hoang	2/15/19	3/6/19	3/4/19	Use Case Descript.	Completed	
Package Diagrams	Kim-Lan Hoang	3/11/19	3/30/19	3/25/19	Iteration II	Completed	
CRC Cards	Jayro Alvarez	3/11/19	3/30/19	3/25/19	Iteration II	Completed	
Interaction Diagrams	Himanshu Patel	3/11/19	3/30/19	3/25/19	Iteration II	Completed	
Class Diagram to Code	Jayro Alvarez	3/11/19	3/30/19	3/25/19	Iteration II	Completed	
Sequence & Communication Diagrams	Himanshu Patel	3/11/19	3/30/19	3/25/19	Iteration II	Completed	
GRASP Concepts Applied	Jayro Alvarez	3/30/19	4/15/19	4/14/19	Iteration II	Completed	
GoF Concepts Applied	Himanshu Patel	3/30/19	4/15/19	4/14/19	Iteration II	Completed	
Visibility Concepts Applied	Kim-Lan Hoang	3/30/19	4/15/19	4/14/19	Iteration II	Completed	
Activity Diagrams	Himanshu Patel	4/15/19	5/5/19	5/1/19	Iteration II	Completed	
Rake Concepts on Activity Diagrams	Himanshu Patel	4/15/19	5/5/19	5/1/19	Iteration II	Completed	
Backward Tracibility Matrix	Himanshu Patel	4/15/19	5/5/19	5/1/19	Iteration II	Completed	
Forward Tracibility Matrix	Himanshu Patel	4/15/19	5/5/19	5/1/19	Iteration II	Completed	
Deployment Diagrams	Himanshu Patel	4/15/19	5/5/19	5/1/19	Iteration II	Completed	
Updated Meeting Minutes	Kim-Lan Hoang	4/15/19	5/5/19	5/1/19	Iteration II	Completed	
Updated Project Charter	Jayro Alvarez	4/15/19	5/5/19	5/1/19	Iteration II	Completed	
Updated Project Plan	All	4/15/19	5/5/19	5/1/19	Iteration II	Completed	
Final Group Revision Process	Kim-Lan Hoang	4/27/19	5/5/19	5/4/19	Project Completion	Completed	
Finalizing Final Document	Jayro Alvarez	4/27/19	5/5/19	5/4/19	Project Completion	Completed	

Iteration 1:

Inception 1: "Establish a Common Vision and Basic Project Scope"

Project Vision:

Our company envisions the next generation of smart wearables, the smart fedora. Fedora+ aims to use a seamless combination of software, hardware, and great fashion to give users access to applications and greater ease of performing tasks.

Business Case: "Benefits of the Proposed Business Change"

The proposal of this new product aims to reach new users and gain user data by providing a new smart-wearable which syncs with both iOS and Android devices.

Business Requirements:

1. Users will want to use Fedora+ device due to ease of use out of the box through link with pre-existing smartphone.
2. Multitude of new user data captured through device's sensors will be recorded and used to improve future features and individual user experience.
3. Users will have many options to choose from. Product will possibly be offered in different:
    a. Shapes, Colors, Designs, Storage Capacities, etc.

Business Rules:

ID	Rule	Changeability	Source
RULE1	Need a smartphone to use device	Low	Company no where near to having a stand-alone device
RULE2	User must be able to give permission for what data is used by company	Low	Law
RULE3	User must be notified prior to any software update	Low	Company policy
RULE4	Lifetime warranty included with every purchase	Mid-Level	Company policy
RULE5	Fedora+ app store open to all software developers	Low	Company policy, aims to continue growth

Functional and Non-Functional Requirements

Functional Requirements:

1. Send/Receive Texts (911, Favorites, etc)
2. Make Phone Calls
3. Location Finding via GPS
4. Detect Heart Attack
5. Count Pulse
6. Check Glucose
7. Voice Assistant

Non-Functional Requirements:

1. System Must Run on Mobile Environment
2. Availability 99%
3. Modifiability
4. Connection to User's Smartphone
5. Long-Lasting Battery

Black-Box Use Cases

Use Case 1	Use Case 2	Use Case 3	Use Case 4	Use Case 5
Send and Receive Text	Turn Talk into Text	Make and Receive Phone Calls	Check Phonebook	Use GPS
Use Case 6	Use Case 7	Use Case 8	Use Case 9	Use Case 10
Examine Heart Pulse	Send Emergency Text and Alarm	Count Heart Pulses	Examine Blood Sugar Level	Talk to Virtual Assistant

Use Cases for Functional Requirements

1. Send/Receive texts
    a. UC 1: Send and receive text messages
    b. UC 2: Speech-to-text
2. Make phone calls
    a. UC 3: Make and receive phone calls with voice trigger

    b. UC 4: Access phonebook from smartphone
3. Location finding via GPS
    a. UC 5: Get location with user permission
4. Detect heart attack
    a. UC 6: Check pulse to detect heart irregularities
    b. UC 7: Send text/activate alarm when issue is detected
5. Count pulse
    a. UC 8: Track heart pulse via skin touch
6. Check glucose
    a. UC 9: Examine blood sugar level
7. Virtual assistant
    a. UC 10: Answer questions and open apps with voice trigger

Elaboration 1: "Address Known Risk Factors and Establish and Validate the System Architecture"

Brief Use-Cases (6):

Actor	Use Case	Brief
User	Make and receive phone calls with voice trigger	User makes a phone call via voice recognition. Fedora will access the smartphone's phonebook via Bluetooth to find and call person requested.
System	Speech-to-Text	System will analyze user's speech and type into the text message as accurate as possible.
User	Get location with user permission	On user's permission, the system will tracks its own location and will respond when the user ask for direction.

System	Track heart pulse via skin touch	System check the pulse of the user by skin touch on the head. System will then record the pulse and will display pulse on smartphone via Bluetooth.
System	Access phonebook from smartphone	System access' the user's phonebook from smartphone when needed.
System	Answer questions and open apps with voice trigger	System will respond to user's request either answering questions or opening apps by using voice trigger.

Casual Use-Cases (3):

Use Case: Check pulse to detect heart irregularities

Actor: System

The user must have the sensors for detecting heart irregularity enabled and a user account.

The system will monitor the user continuously without requiring the user to do anything. The system will monitor for irregular heartbeats, record the data, and send it to the user's doctor with the user's consent.

If the system ever goes down it will return an error and any next steps the user should take. The system won't be able to check the user's pulse and record any data. The system will notify the user when it is back up. The system will only update with the user's consent.

If the system detects heart irregularities but the user is not actually having any, the user tells the system and it will relay that to the user's doctor.

Use Case: Examine blood sugar levels using vibration

Actor: System

The user must have the sensors for checking glucose enabled.

The system tracks the user's glucose throughout the day and notifies them of any highs and lows so they can take control of their blood sugar levels. The system checks the user's blood sugar level every 5 minutes and displays the results on the user's phone/other device or uses the voice assistant to tell the user the results.

If the system goes down or stops tracking the user's glucose, it will return an error message and notify any contacts if the user has the option enabled. The user can't track their glucose or blood sugar while the system is down. The system will notify the user when it can start monitoring the user again.

Use Case: Send text/activate alarm when an issue is detected

Actor: System

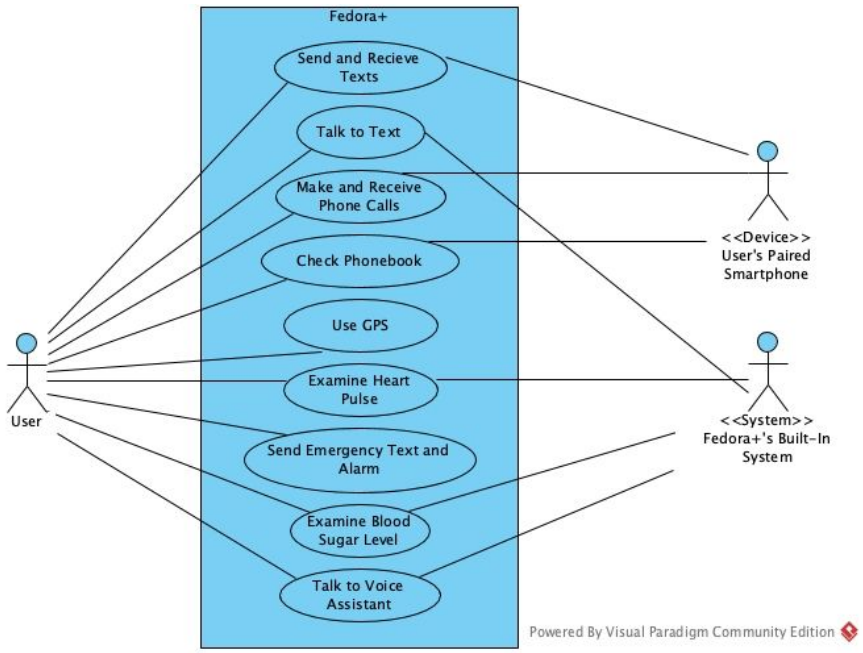
The user must have the option to send text and activate an alarm enabled in settings.

The system detects heart irregularities and sends a text message to 911 and any other emergency contacts. The system activates an alarm if it detects an heart attack.

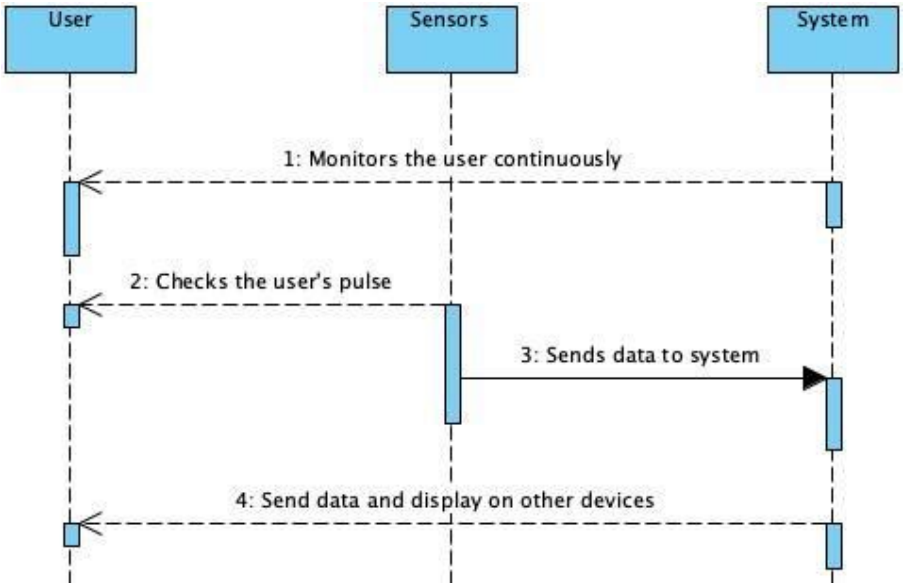
If the system is down, it will not be able to send text messages or activate an alarm.

The system must respond to the user within 2 seconds and it should take no longer than 4 seconds for the system to respond. If the system detects an issue but the user is not experiencing anything, the user tells the system it was a false alarm and it will relay that message to any emergency contacts.

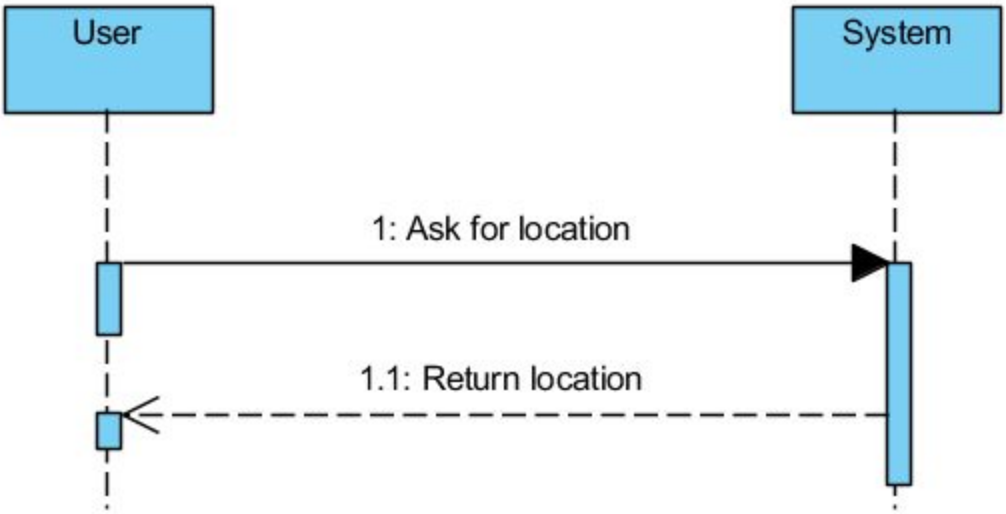
Initial Use Case Diagram:



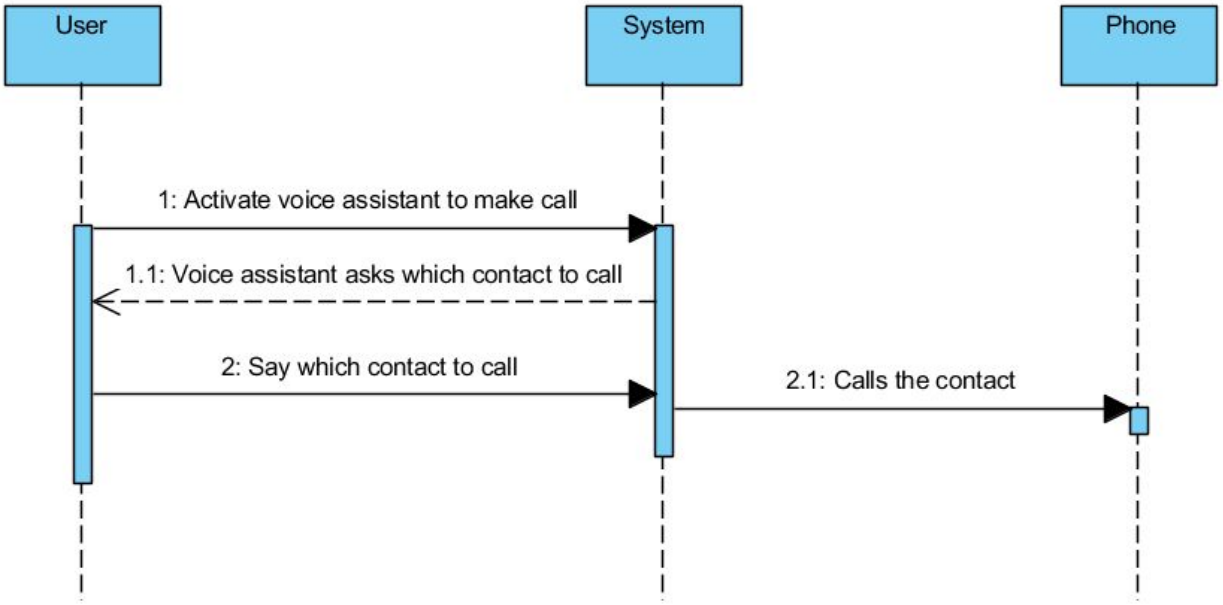
Sequence Diagram 1 - Check pulse to detect heart irregularities:



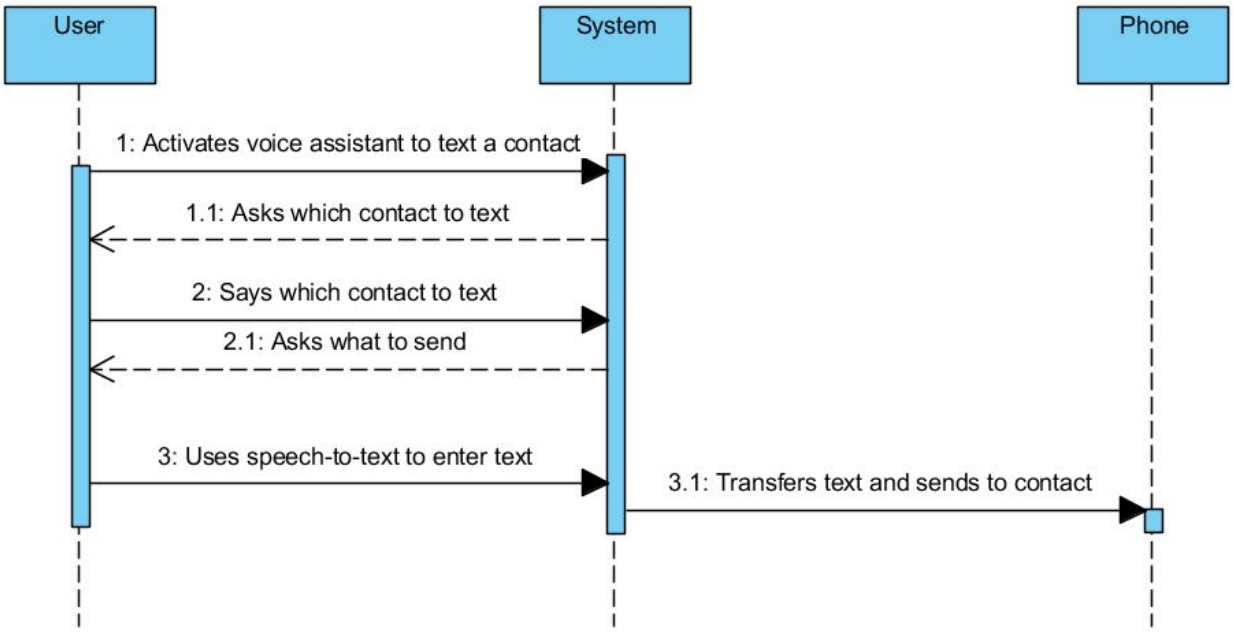
Sequence Diagram 2 - Get location:



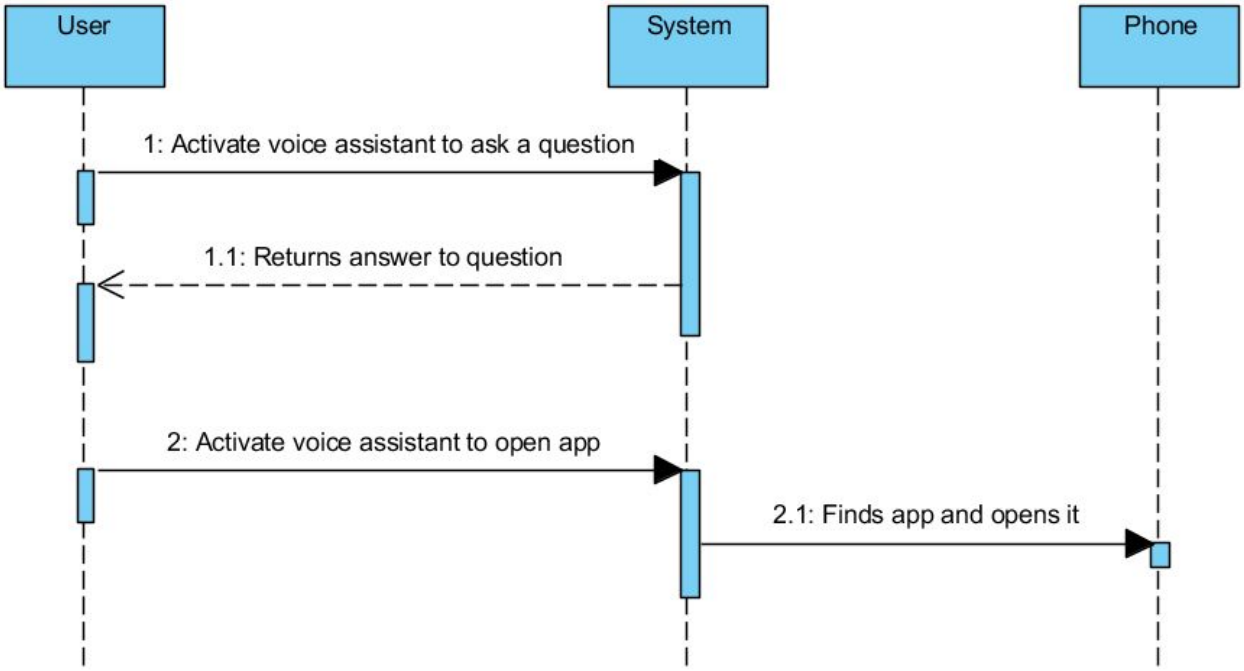
Sequence Diagram 3 - Make and receive phone calls with voice trigger:



Sequence Diagram 4 - Send and receive text messages



Sequence Diagram 5 - Answer questions and open apps with voice trigger:

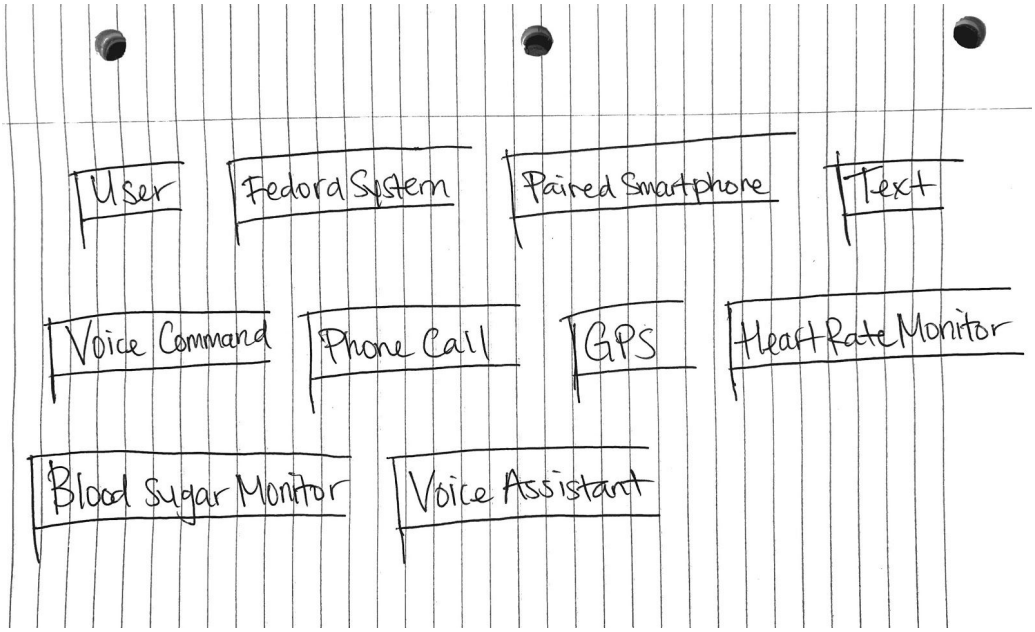


Fully Dressed Use-Case (1):

Use Case ID:	UC-01		
Use Case Name:	Send and receive text messages		
Created By:	Jayro Alvarez	Last Updated By:	Jayro Alvarez
Date Created:	Feb 14, 2019	Last Revision Date:	Feb 14, 2019
Actor(s):	1. User 2. Smartphone BlueTooth Connection		
Description:	The user is able to send and receive texts. Done through		
Trigger(s):	- Trigger voice assistant and instruct it to send a text to a specific		

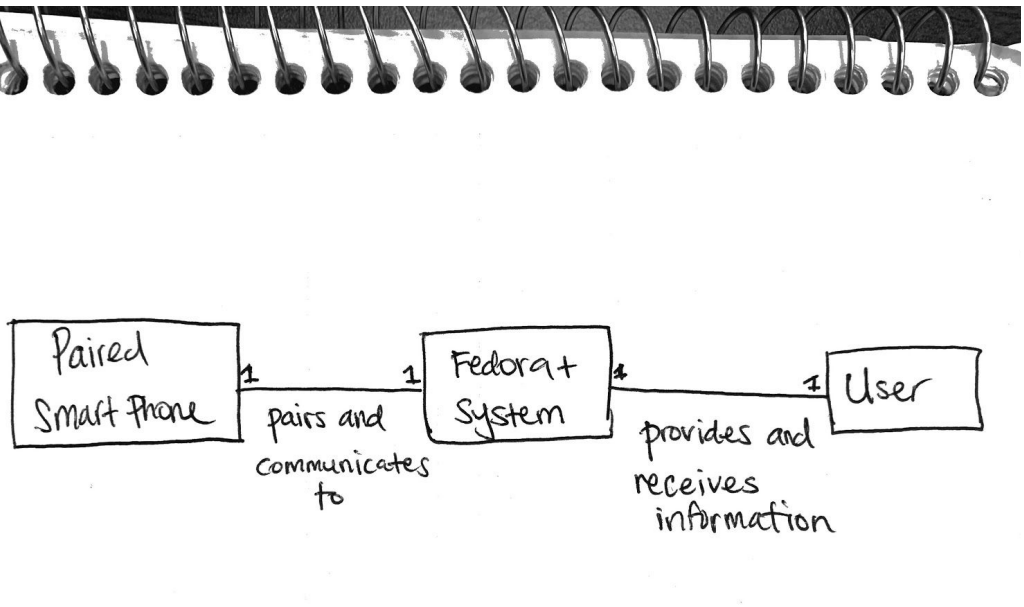
	person in contacts or read inbox of received text messages.
Preconditions:	- Fedora+ device must be on and have sufficient battery life. - Fedora+ device must be connected to Wifi or cellular data.
Postconditions:	- Feddy will continue listening after completing text task. Assistant will go back to standby if no further vocal input is heard from user.
Normal Flow:	1. User triggers voice assistant: verbally or manually. 2. Voice assistant software will trigger a sound to let user know it is ready for voice input request 3. After completing task and communicating back to user about chosen decision, input sound is triggered again 4a. Will continue getting more input if user responds 4b. Will go back to stand-by if user is silent
Frequency of Use:	- Used frequently by user to send and listen to received messages.

Initial Domain Model:



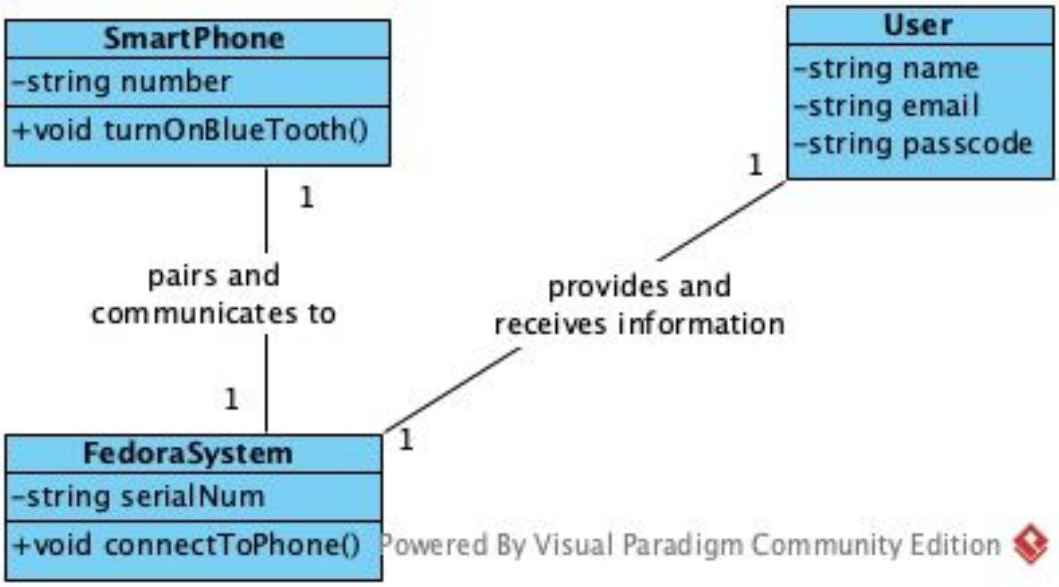
*Initial Conceptual Classes

Domain Model:



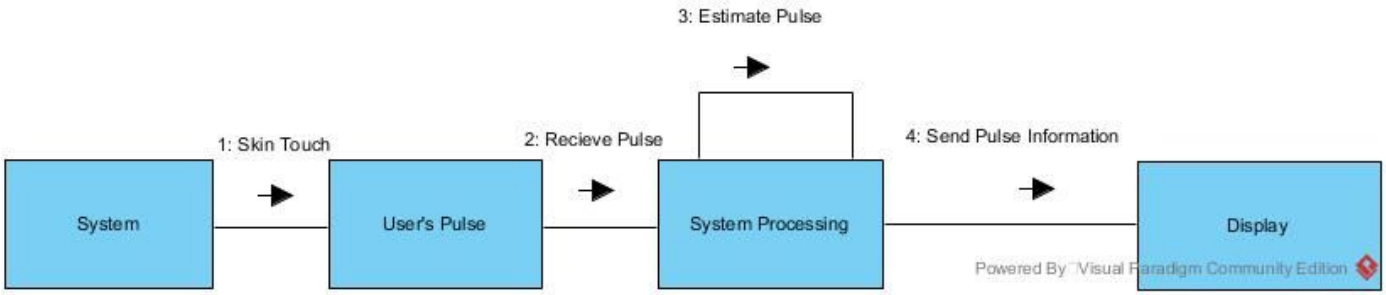
*Simple Domain Model w/Main Conceptual Classes/Nouns

Class Diagram:

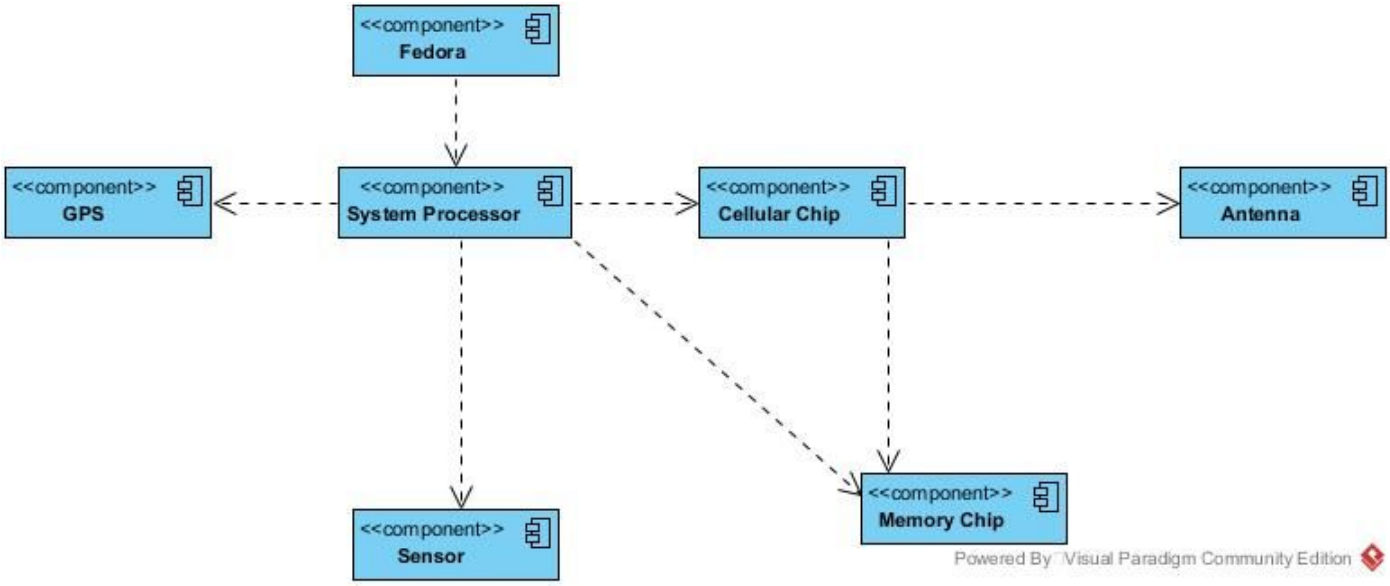


*Based on Initial Domain Model

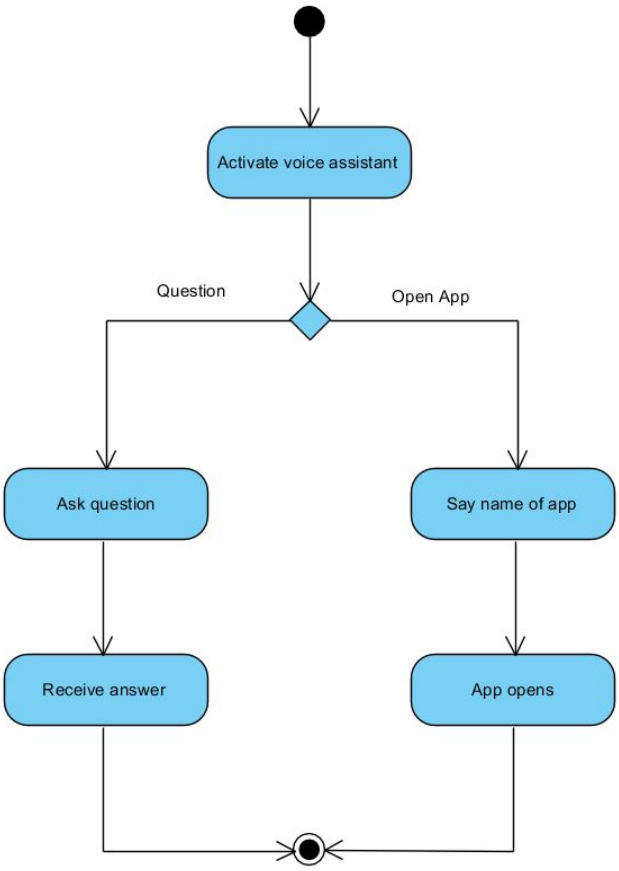
Collaboration Diagram - Track heart pulse via skin touch:



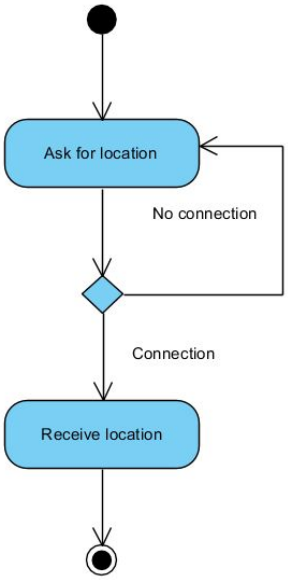
Fedora+ Component Diagram:



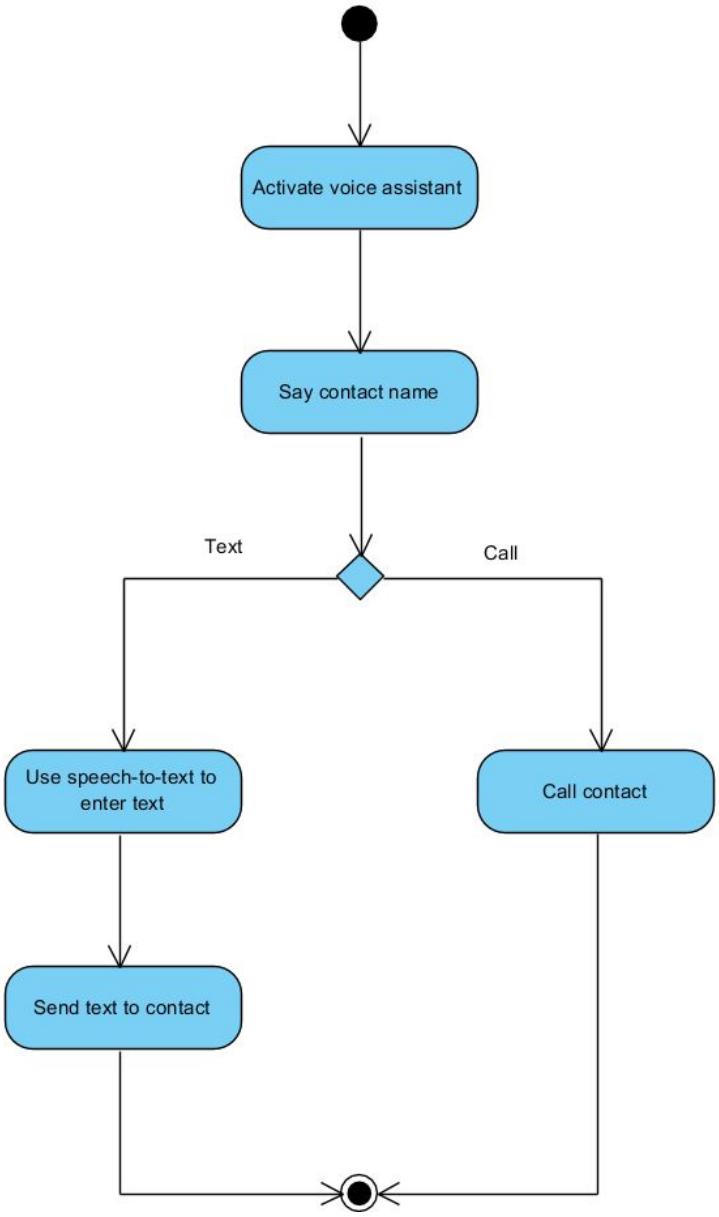
Activity Diagram 1 - Answer questions and open apps with voice trigger:



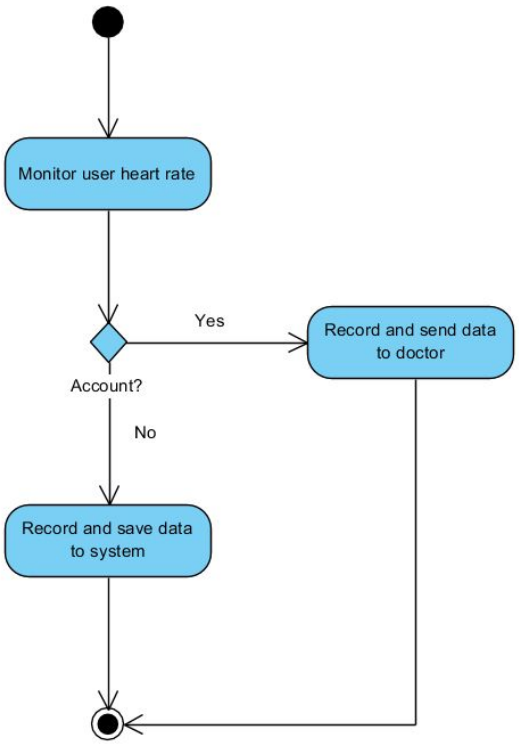
Activity Diagram 2 - Get location



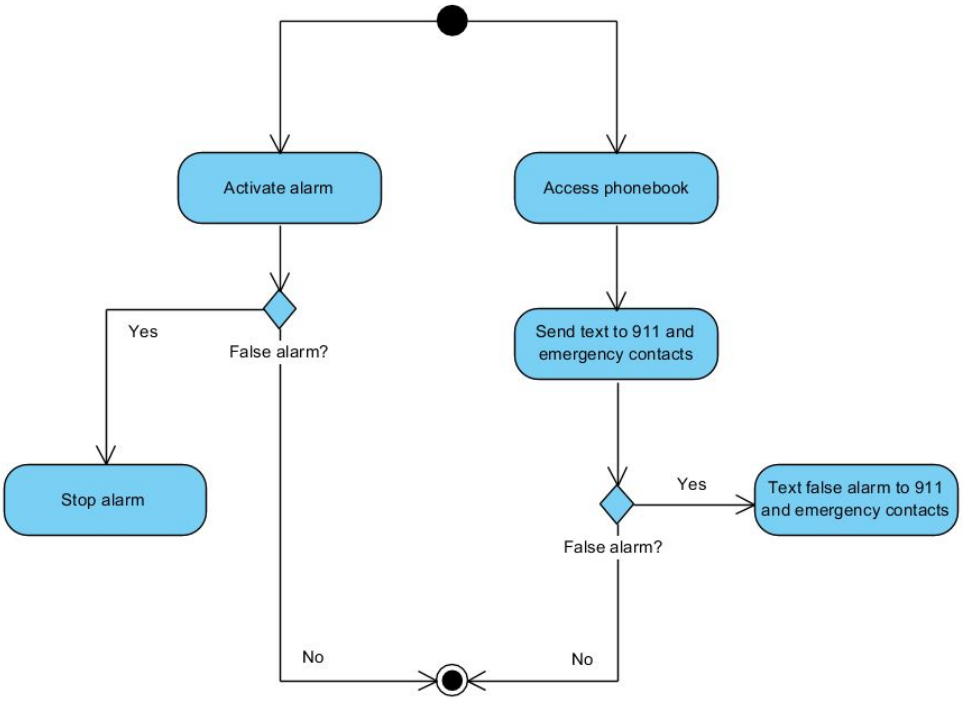
Activity Diagram 3 - Send and receive texts/Make and receive phone calls with voice trigger:



Activity Diagram 4 - Check pulse to detect heart irregularities:



Activity Diagram 5 - Send text/activate alarm when issue is detected:



Supplementary Specification

Introduction
The document is the repository of all Fedora+ requirements not captured in the use cases

Functionality
(Functionality common across many use cases)

Logging and Error Handling
All error in systems will be logged into internal memory

Human Factors
The user will be wearing the fedora and speak toward the systems. Therefore:

- System will recognized a few different native languages
- System will differentiate the user's voice from the other people to avoid confusion.
- Fedora should be light and comfortable enough for the user to wear

Reliability
One of fedora's main purpose is to prevent the sun from becoming too unbearable to user, while being able to access the user's smartphone. Therefore:

- The system will be able to remain cool under hot weather to avoid overheat
- The system should be compatible with most of the smartphone

More analysis is needed here.

Performance
System response to voice activation should be done quickly and efficiently. Time goal for response should be within 2 seconds, and should not take more than 4 seconds. Speed may vary depending on smartphone's performance and access.

Configurability
There will be various users who have difference preference, most noticeably native language for voice activation. Smartphone will have access to Fedora+ to change and make configuration when needed. More work is needed, which includes finding other native speaker make the voice recognition more compatible to the user.

Hardware/Software Implementation Constraints
Chips and devices must be powerful enough to ensure that Fedora+ response to the user's request in a timely manner, but not be heavy to make it uncomfortable to wear.

Purchased Component

- Touch Sensor – receive information, such as user's pulse, by skin touch.
- Speaker – receive user's voice and transfer into the system
- GPS Module – Get location of user's position

Documentation
Most installation should be done and ready to use. User manual will be given along side with the device to explain the function and component of Fedora+. Other installation instruction such as installing SD Cards, application, etc., will be included in the manual.

Licensing
All licensing for the fedora itself, along with any device or hardware purchased, will be processed in a legal manner before the product is developed and shipped to avoid and legal issue.

Physical Environment Concerns
Due to nature of the device, Fedora+ can be damaged by various circumstances which includes, but not limited to:

- Heat
- Water
- Being pressed down (Ex: Sitting/Placing heavy object on top)

User manual will contain instructions to prevent such accidents.

Qualities:

Usability:
- High
- Very user intuitive to use

Reliability:
- High
- Device very reliable under conditions where device has sufficient battery life and sufficient internet connectivity

Supportability:
- High
- Device supports major smartphone operating systems (iOS and Android)

Iteration 2:

Inception 2: "Establish a Common Vision and Basic Project Scope"

Project Vision (Updated):

The **updated vision** of the company remains with the same goals, attracting users with a never-before-seen smart-wearable device, Fedora+. User connectivity just seamlessly works with the wearing of the device. Communication with user and device is as easy as speaking to your device's voice assistant to access a multitude of different applications and perform different tasks.

Business Case (Updated): "Benefits of the Proposed Business Change"

The proposal of this new product aims to reach new users and gain user data by providing a new smart-wearable which syncs with both iOS and Android devices. Users of this new product are automatically potential customers to any software, applications, or future device iterations in a new market of unique smart devices.

Business Requirements:

1. Users will want to use Fedora+ device due to ease of use out of the box through link with pre-existing smartphone.
2. Multitude of new user data captured through device's sensors will be recorded and used to improve future features and individual user experience.
3. Users will have many options to choose from. Product will possibly be offered in different:
    a. Shapes, Colors, Designs, Storage Capacities, etc.
4. User-to-device communication is seamless.
5. Easy-to-use product for all ages.
6. Increase customer loyalty with company through future device iterations/models.

Elaboration 2: "Address Known Risk Factors and Establish and Validate the System Architecture"

Additional Fully Dressed Use Cases (2):

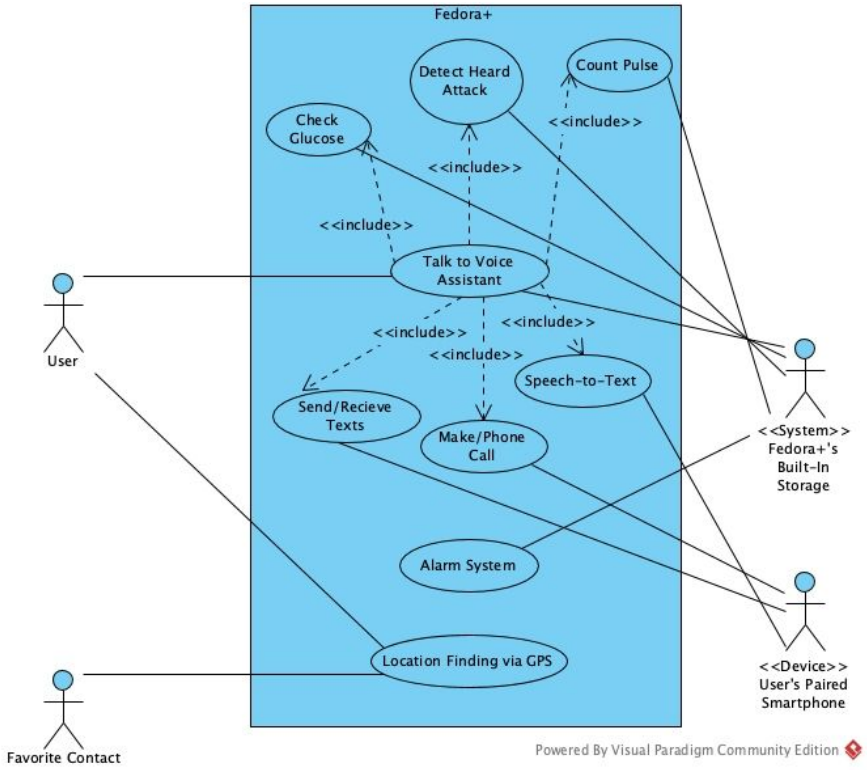
Use Case ID:	UC-10		
Use Case Name:	Talk to Voice Assistant		
Created By:	Jayro Alvarez	Last Updated By:	Jayro Alvarez

Date Created:	Mar 4, 2019	Last Revision Date:	Mar 11, 2019
Actor(s):	1. User 2. Fedora+ System		
Description:	The user is able to naturally speak with the Fedora's built-in voice assistant.		
Trigger(s):	- Can trigger assistant by holding button on device or saying the keyphrase, "Hello, <name>", depending on what user has named their assistant.		
Preconditions:	- Fedora+ device must be on and have sufficient battery life. - Fedora+ device must be connected to Wifi or cellular data.		
Postconditions:	- Feddy will continue listening after completing user's task. Assistant will go back to standby if no further vocal input is heard from user.		
Normal Flow:	1. User triggers voice assistant: verbally or manually. 2. Voice assistant software will trigger a sound to let user know it is ready for voice input request 3. After completing task and communicating back to user about chosen decision, input sound is triggered again 4a. Will continue getting more input if user responds 4b. Will go back to stand-by if user is silent		
Frequency of Use:	- Used frequently by user to perform many tasks, launch apps, etc.		

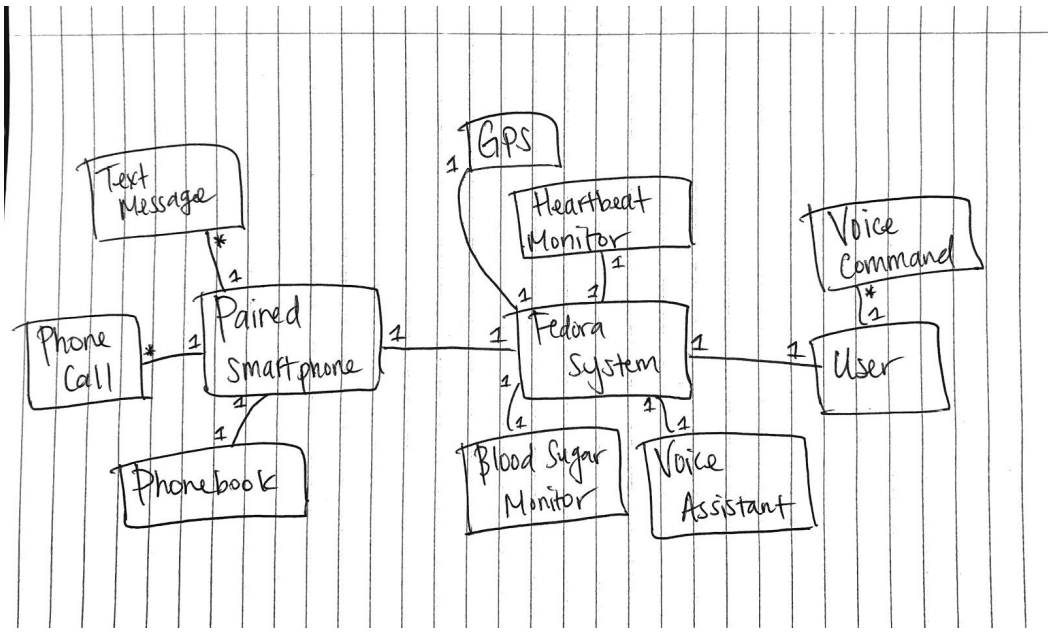
Use Case ID:	UC-07		
Use Case Name:	Send Emergency Text and Alarm		
Created By:	Jayro Alvarez	Last Updated By:	Jayro Alvarez
Date Created:	Mar 4, 2019	Last Revision Date:	Mar 11, 2019
Actor(s):	1. Fedora+ System		

Description:	The system is able to detect an emergency situation and trigger a loud alarm while simultaneously alerting the right emergency officials depending on GPS location.
Trigger(s):	- Can be triggered due to many situations. Fedora uses machine learning and a combination of built-in sensors to check on the user. Example: Fedora detects that user is having a heart attack.
Preconditions:	- Fedora+ device must be on and have sufficient battery life. - Fedora+ device must be connected to Wifi or cellular data.
Postconditions:	- Device will continue trying to contact the correct emergency personnel until a response is detected.
Normal Flow:	1. Fedora detects emergency situation. 2. Will prompt user to see if they actually are in trouble. 2a. If user voids the emergency notification, device will go back into Idle. 2b. Decides emergency is required and starts attempting to contact the right emergency personnel.
Frequency of Use:	- Not used frequently, only in cases of emergency where the user cannot react.

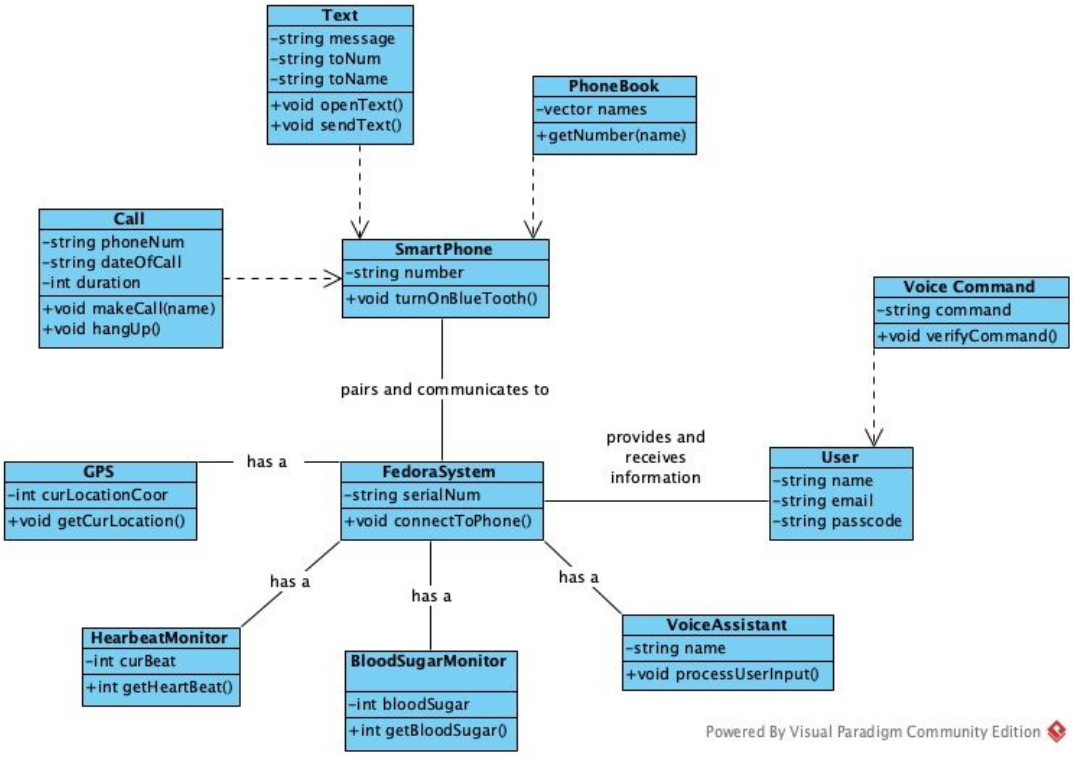
Use Case Diagram:



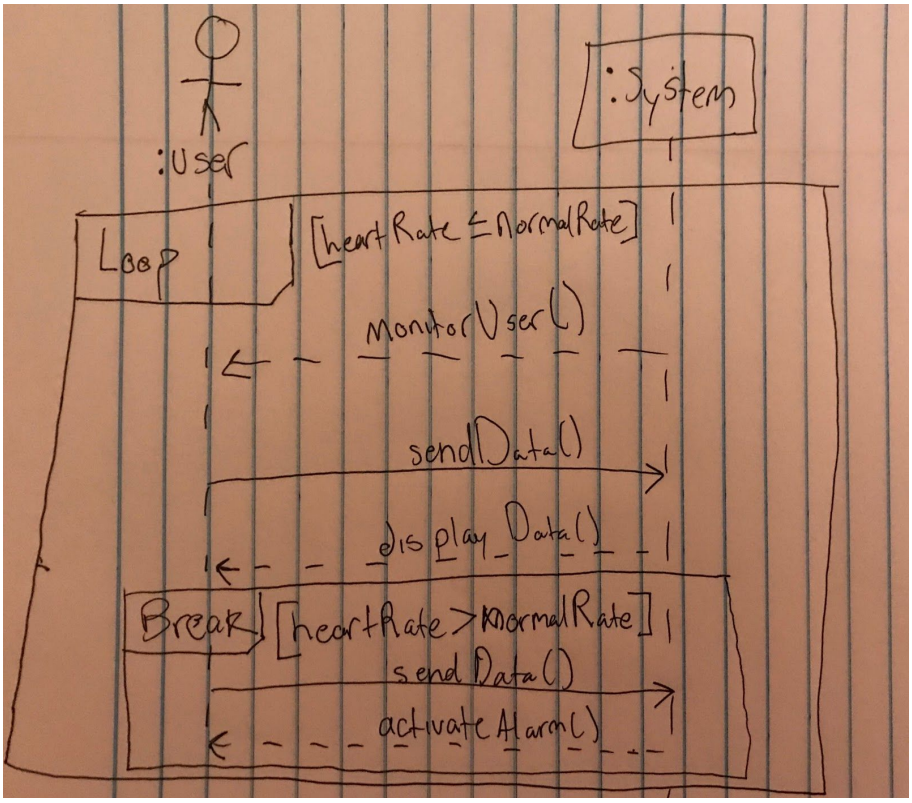
Domain Model:



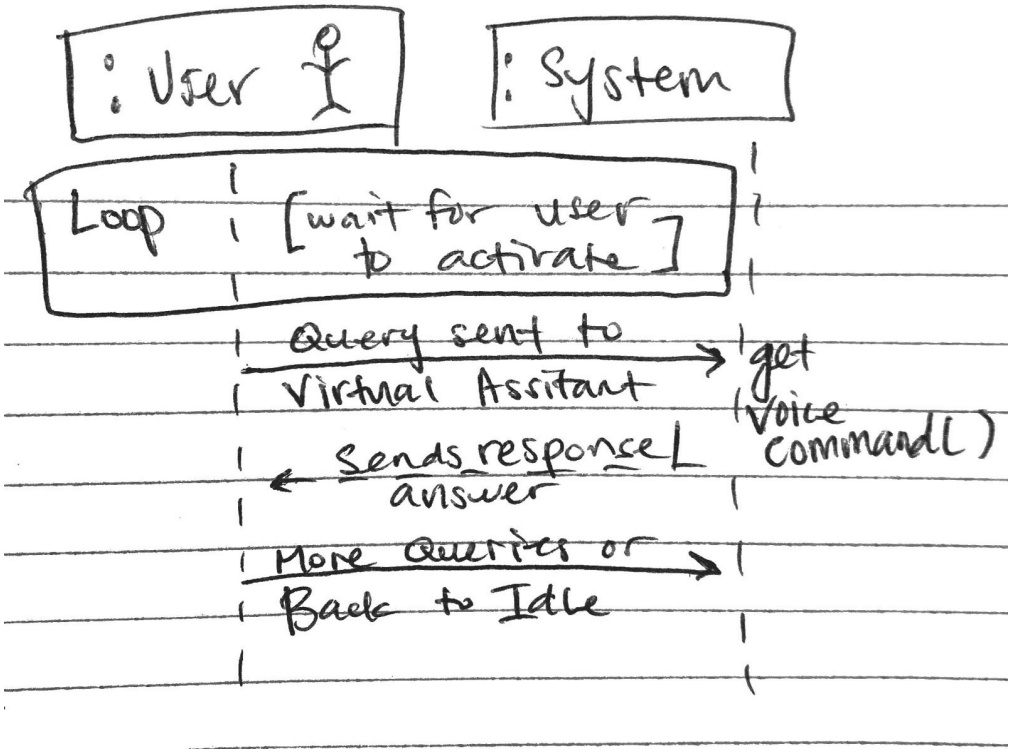
Class Diagram:



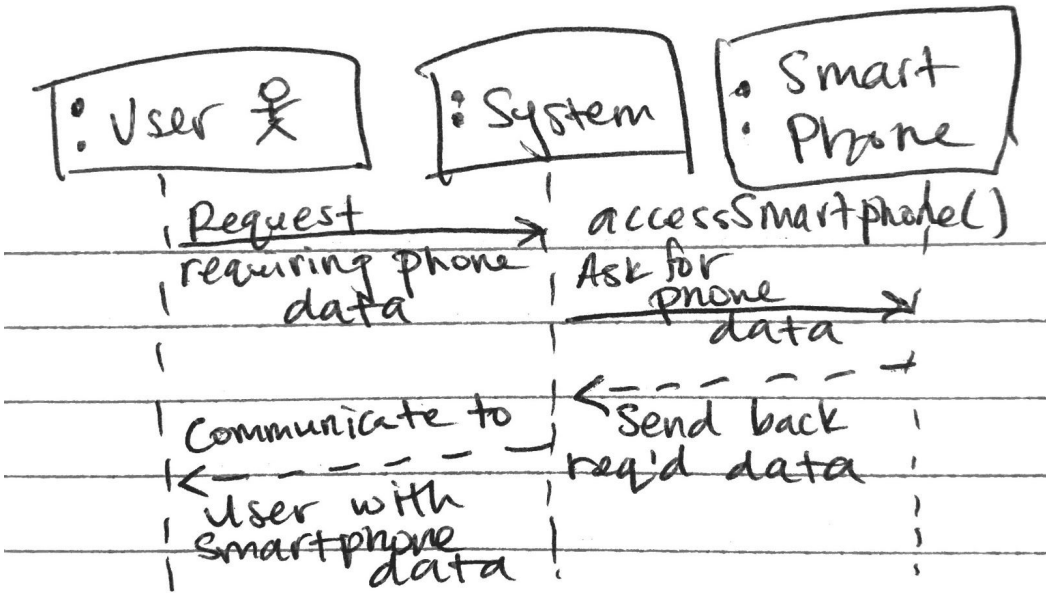
System Sequence Diagram 1:



System Sequence Diagram 2:



System Sequence Diagram 3:



Use-Case Contracts:

Contract 1: monitorUser

Operation: monitorUser (normalHeartRate: integer)

Cross References: Use Cases: 6 and 7

Constraints/Preconditions:
- The system is not down and the sensors work correctly.
- The device has sufficient battery life.
- The device must be connected to the user's account.

Postconditions:
- A userHeartRate variable was created
- A boolean named activateAlarm was set to true because there was a break in the loop

Contract 2: getVoiceCommand

Operation: getVoiceCommand (voiceCommand: voiceCommand)

Cross References: Use Cases: 2 and 10

Constraints/Preconditions:
- The system is not down and the sensors work correctly.
- The device has sufficient battery life.
- The device must be connected to the user's account.

Postconditions:
- System results back in Idle mode

Contract 3: accessSmartphone

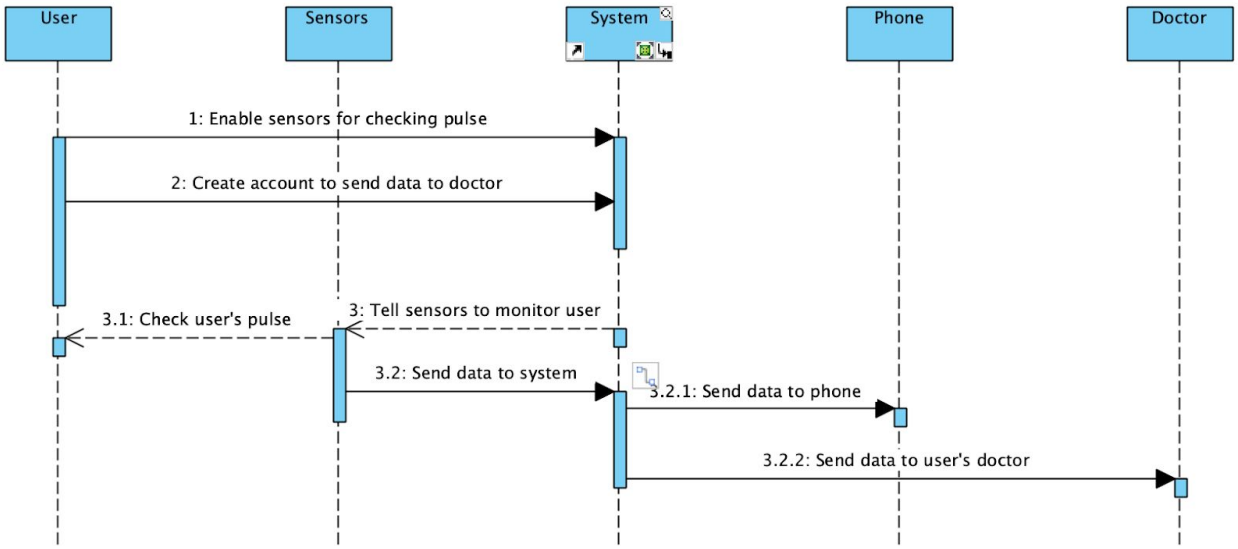
Operation: accessSmartphone (smartphoneConnection: smartphoneConnection)

Cross References: Use Cases: 1, 3, and 4

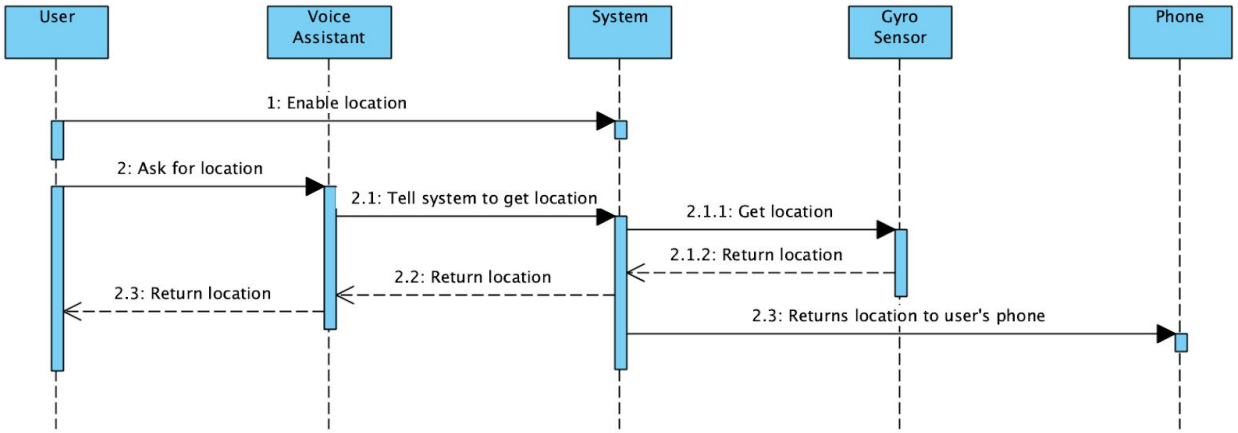
Constraints/Preconditions:
- The system has successfully connected to user's smartphone
- The device has sufficient battery life.
- The device must be connected to the user's account.

Postconditions:
- System results back in Idle mode

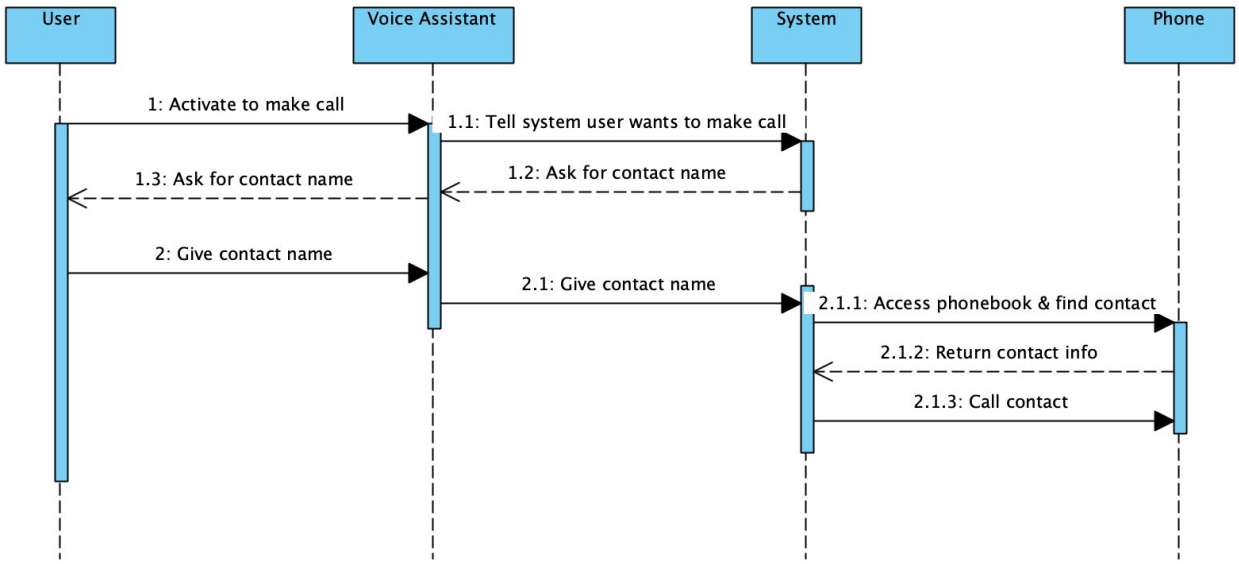
Sequence Diagram 1 - Check user's pulse for heart irregularities:



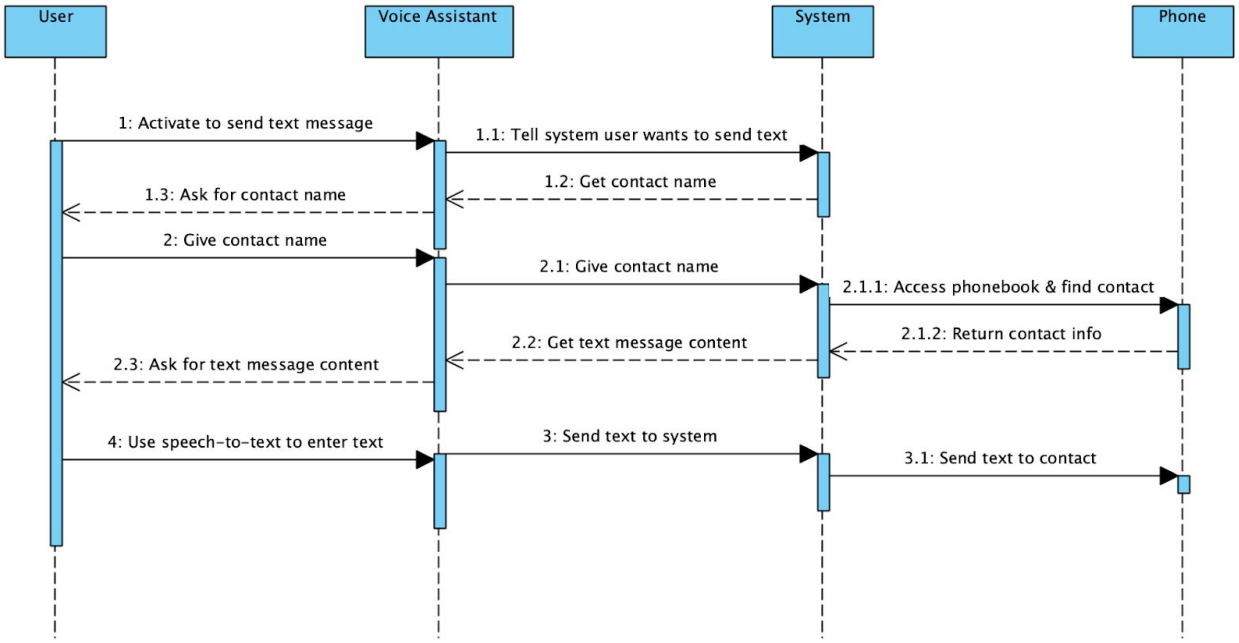
Sequence Diagram 2 - Get location with user's permission:



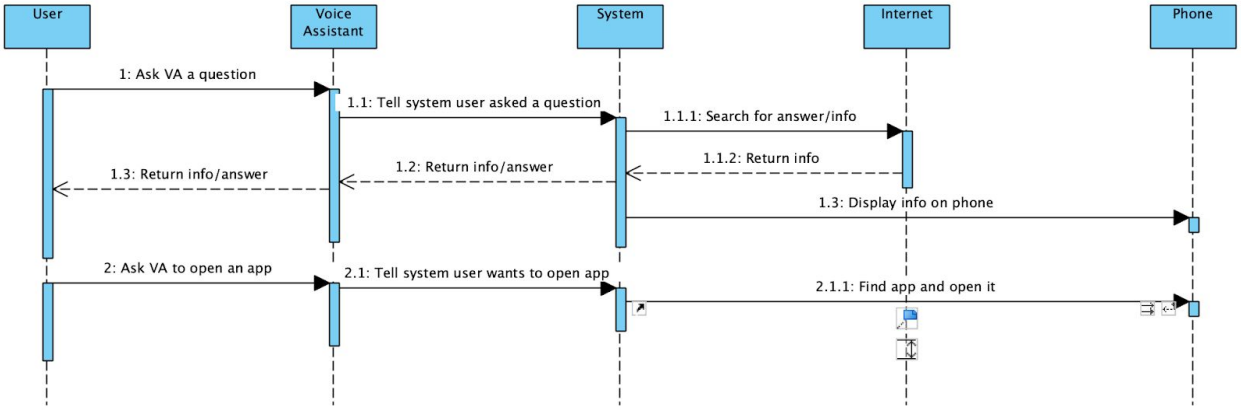
Sequence Diagram 3 - Make/receive phone calls with voice trigger:



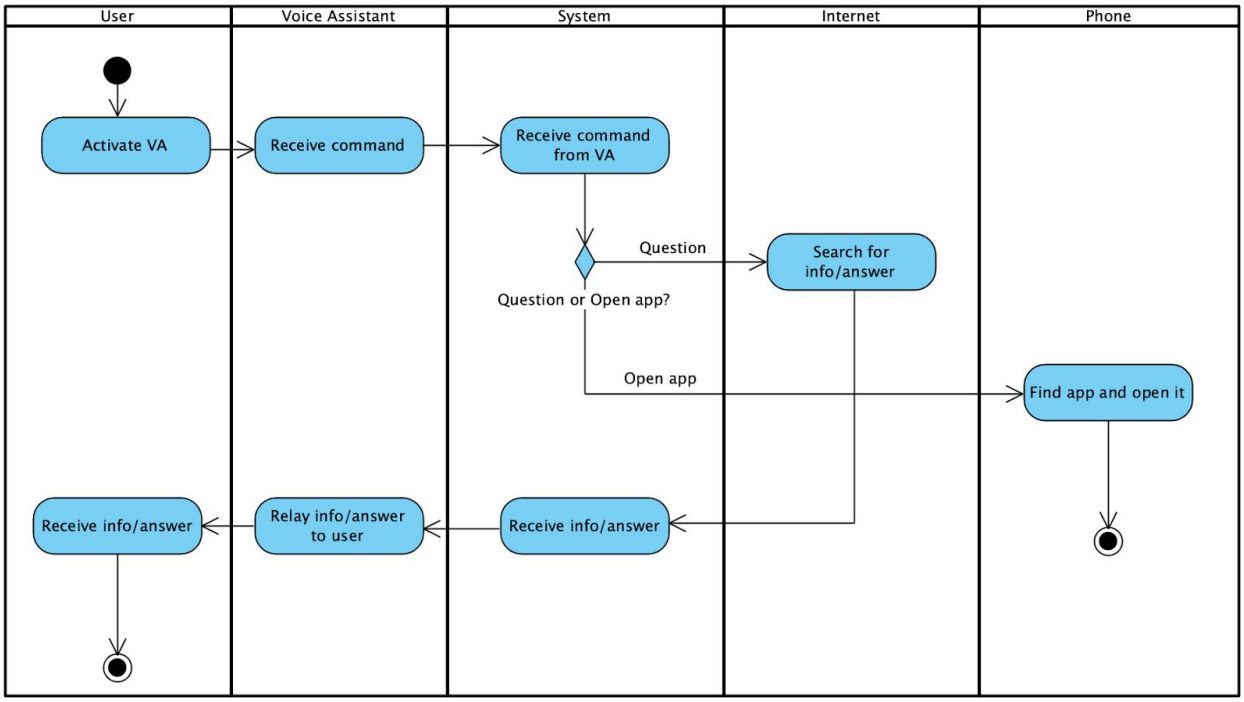
Sequence Diagram 4 - Send/receive text messages with voice trigger:



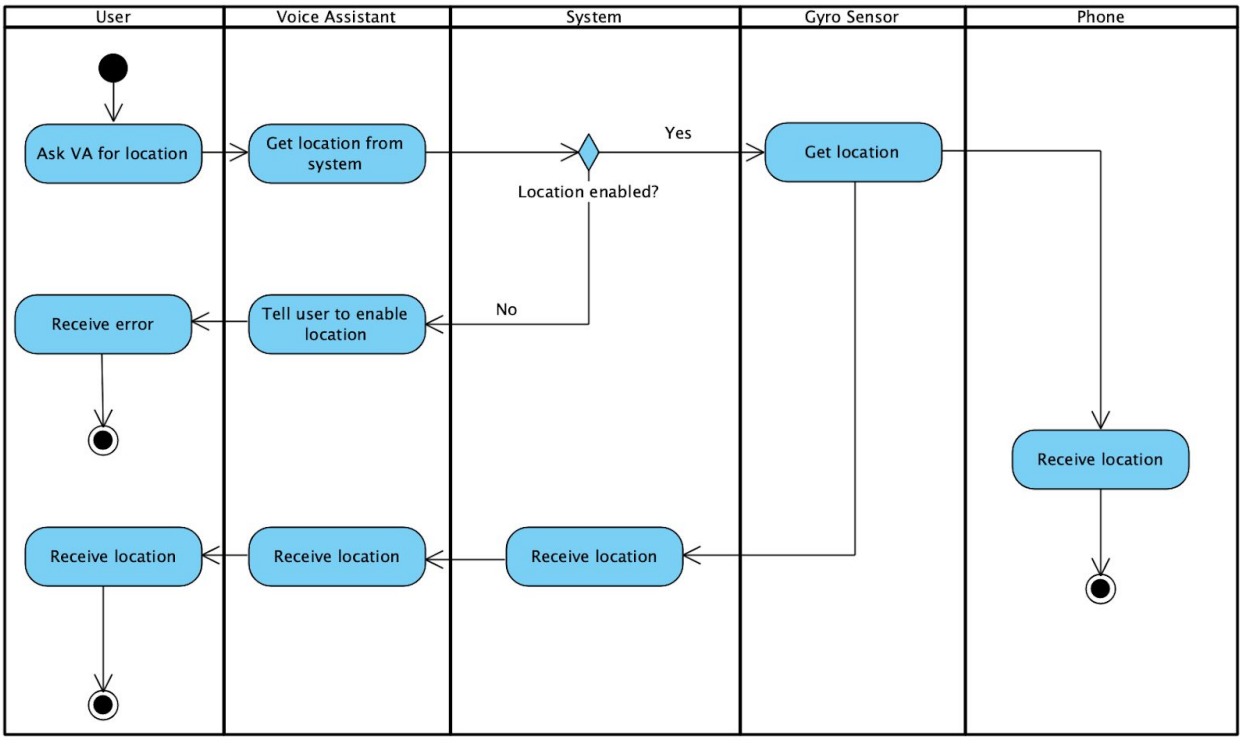
Sequence Diagram 5 - Ask questions and open apps with voice trigger:



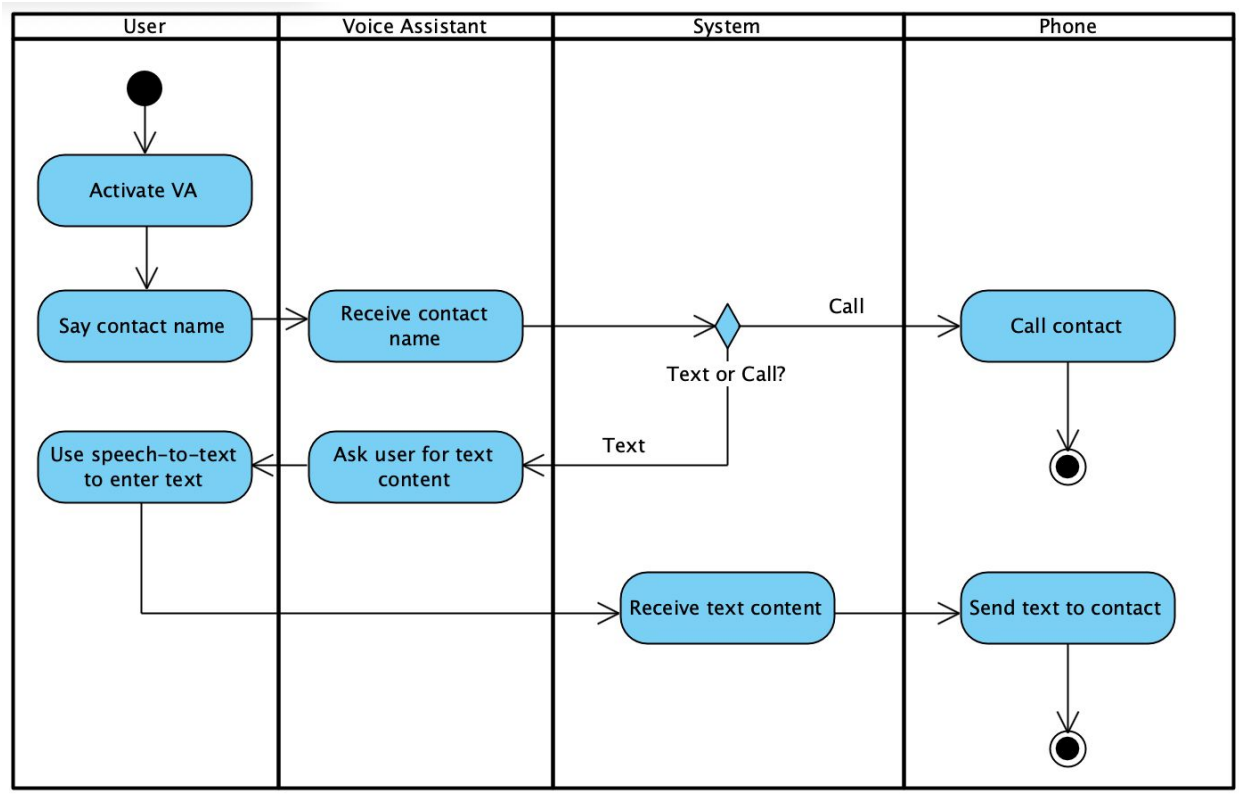
Activity Diagram 1 - Ask questions and open apps with voice trigger:



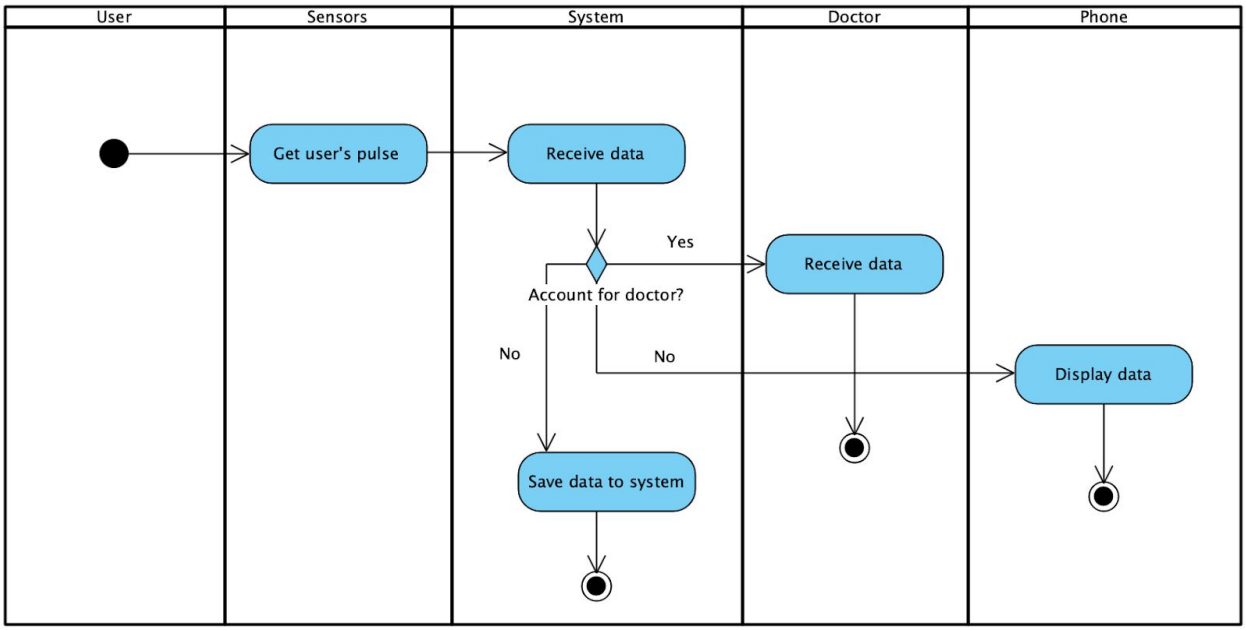
Activity Diagram 2 - Get location with user's permission:



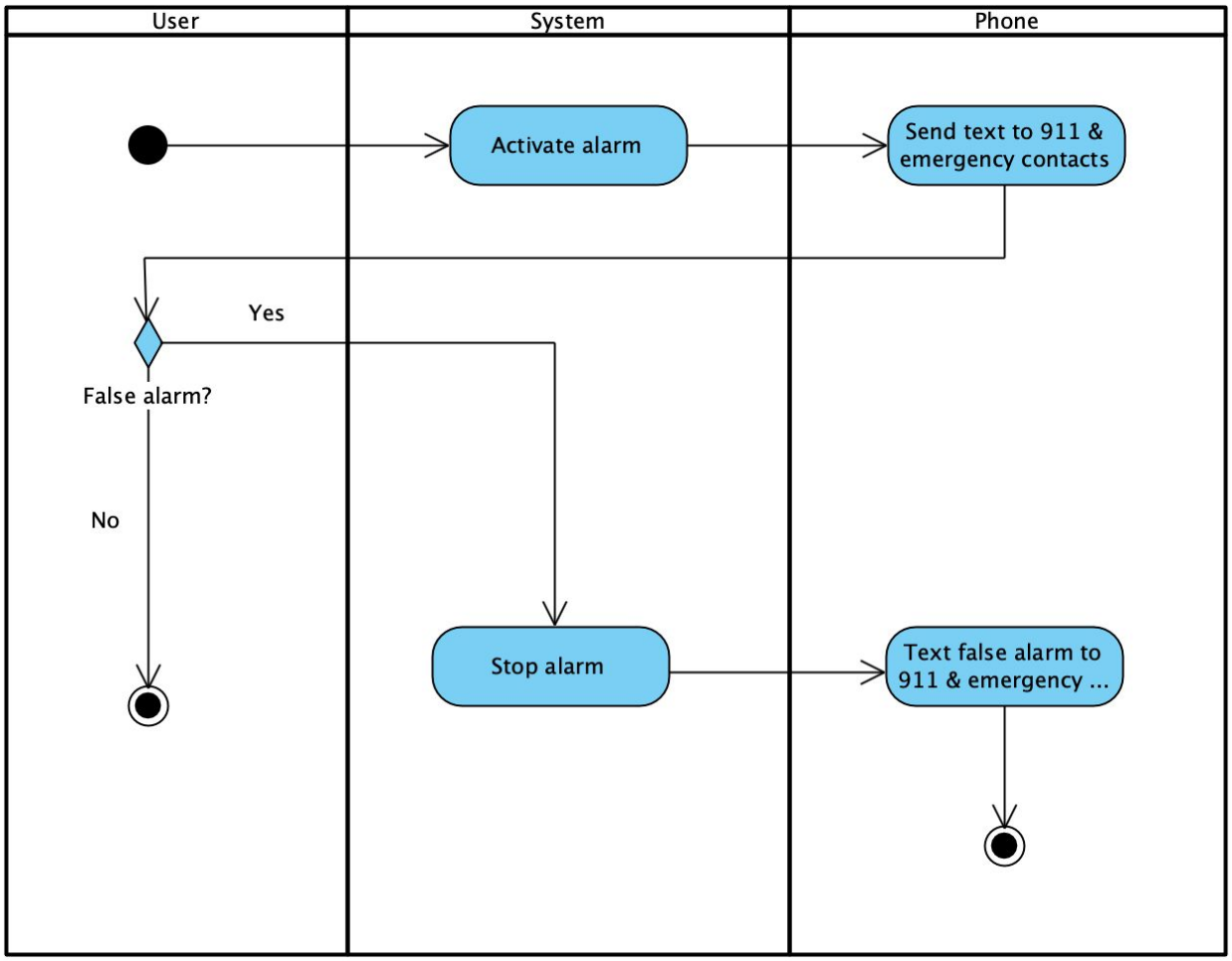
Activity Diagram 3 - Make/receive phone calls and send/receive texts with voice trigger:



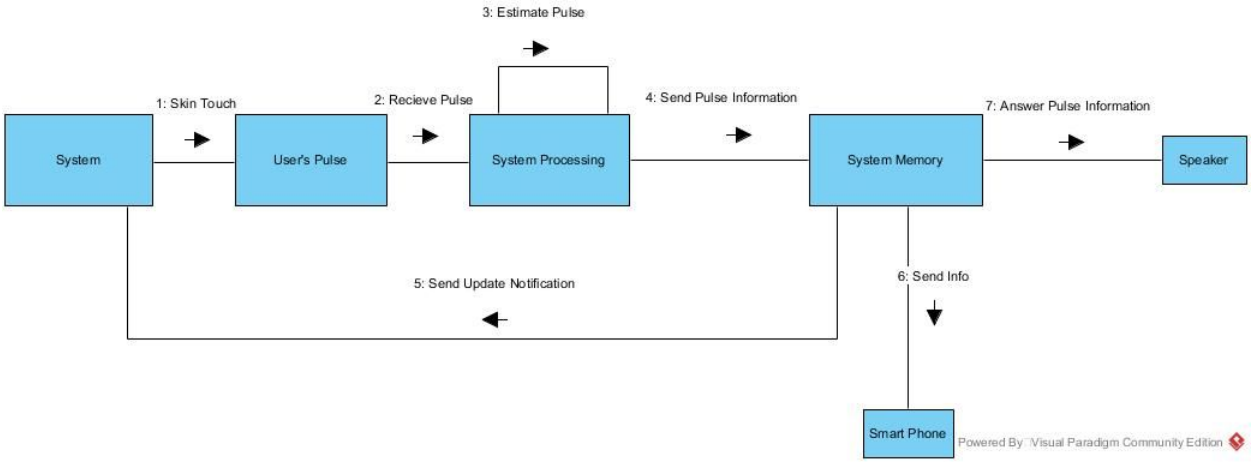
Activity Diagram 4 - Check pulse to detect heart irregularities:



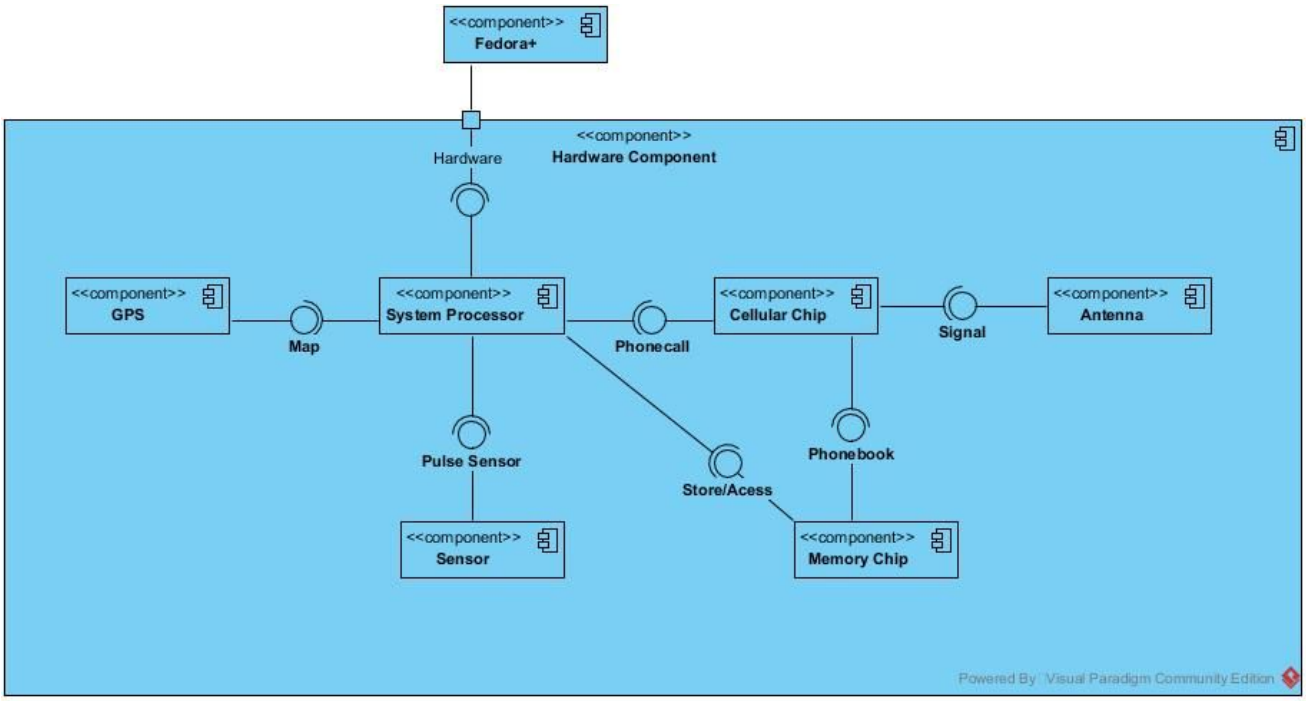
Activity Diagram 5 - Send text/activate alarm when issue is detected:



Pulse Check Collaboration Diagram:



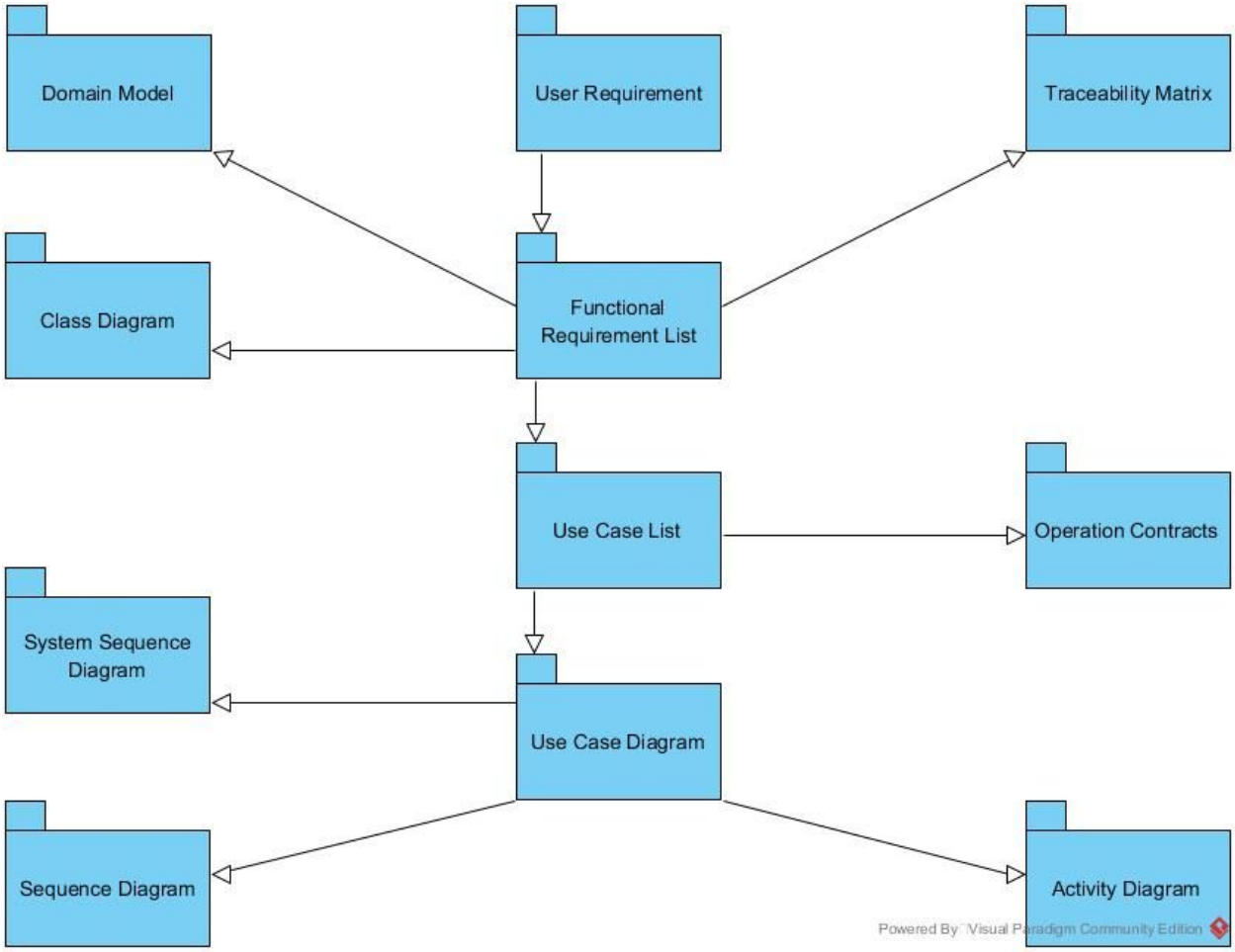
Fedora+ Component Diagram:



Traceability Matrix:

	FR 1	FR 2	FR 3	FR 4	FR 5	FR 6	FR 7
Use Cases							
1	X						
2	X						
3		X					
4		X					
5			X				
6				X			
7				X			
8					X		
9						X	
10							X

Traceability Diagram:



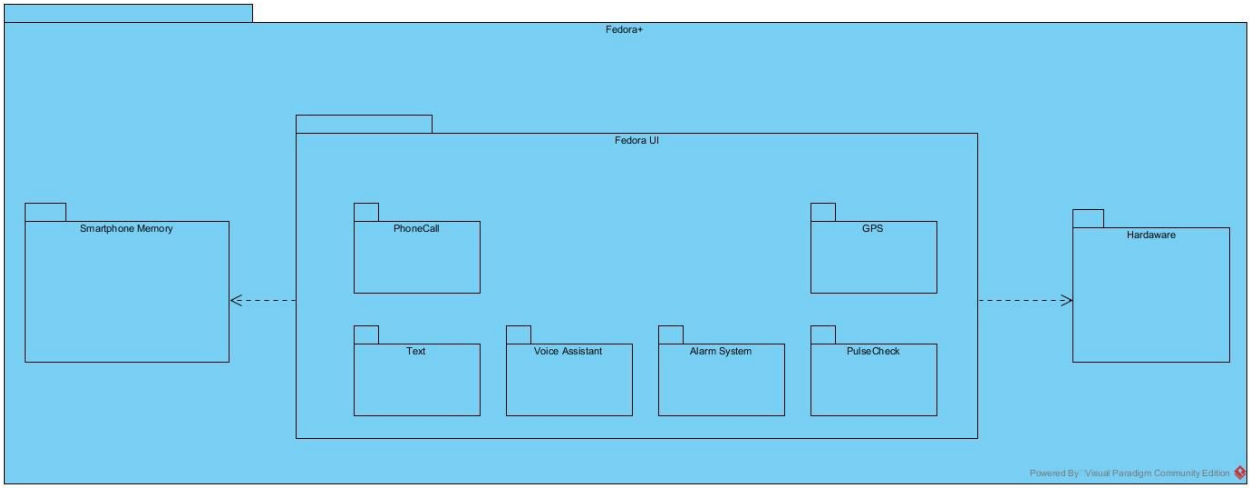
Glossary:

Term	Definition and Information	Format	Validation Rules	Aliases
Smart Wearable	smart electronic devices that can be incorporated into clothing or worn	X	X	Wearable
Fedora	a low, soft felt hat with a curled brim and creased crown	Physical Device	X	Smart Hat
Virtual Assistant	intelligent virtual assistant that emulates human interaction	Virtual/ Electronic	X	Voice Assistant
Phonebook	refers to electronic phonebook on user's smartphone	Virtual/ Electronic	Validated after phone connection	X
GPS	built in global positioning system	Virtual/ Electronic & Physical Component	X	X
Glucose	simple sugar used by living organisms for energy	X	X	
Voice Trigger	user spoken phrase which is captured by system	X	X	Voice Shortcut
Talk-to-Text	algorithm in system which translates user's spoken words to understandable electronic phrases	Virtual/ Electronic	X	X
Sensor	device that detects and responds to some sort of input from physical environment	Various	X	X
Notification	automated message sent by an application	Virtual/ Electronic	Immediately shown to user	Pop-up

Iteration 3:

Elaboration 3: “Address Known Risk Factors and Establish and Validate the System Architecture”

Package Diagram (1):



CRC Cards:

Blood Sugar Monitor	
-Knows of current blood sugar of user -Can return current sugar level of user to the Fedora System -Knows healthy blood range -Can push alert if detects unhealthy blood sugar range	-Fedora System

Call	
-Knows how to establish connection -Knows recipient number -Knows recipient name -Returns and receives steady line of connectivity with smartphone class object	-Smartphone

Fedora System	
-Knows its unique serial# -Can establish connection to smartphone -Can process user calls to its inherited methods from connnected components	-Smartphone -GPS -Heartbeat Monitor -Blood Sugar Monitor -Voice Assistant -User

GPS	
-Knows current location of user -Knows of user's current -Can return a current location object to Fedora System	-Fedora System

Heartbeat Monitor	
-Knows of current beat of user -Can return the current beat of the user to the Fedora System -Knows the healthy heart-beat range for a user	-Fedora System

Phonebook	
-Can return a contact and their information -Can search for a contact -Knows all contacts located in Smartphone	-Smartphone

Smartphone	
-Gives Bluetooth Conectivitiy to Fedora -Knows phone number -Knows how to pair with Fedora -Knows network carrier	-Text -Phonebook -Phone call -Fedora System

Text	
-Knows message contents -Knows to numbers -Knows to name -Returns and receives text objects	-Smartphone

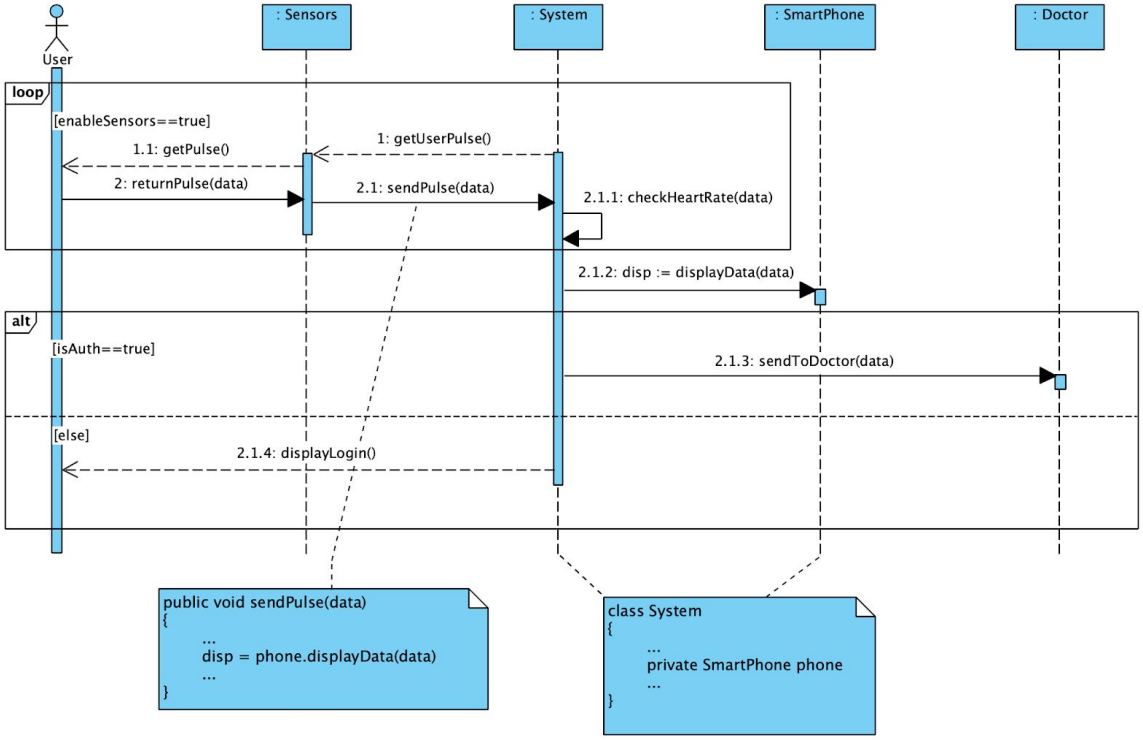
User	
-Knows user name -Knows user email -Knows user passcode -Provides information to Fedora System through Voice Commands -Knows how to activate Voice assistant	-Fedora System -Voice Command

Voice Assistant	
-Knows keyphrase trigger to be activated by user -Can process user's verbal input into action functions sent to Fedora System	-Fedora System

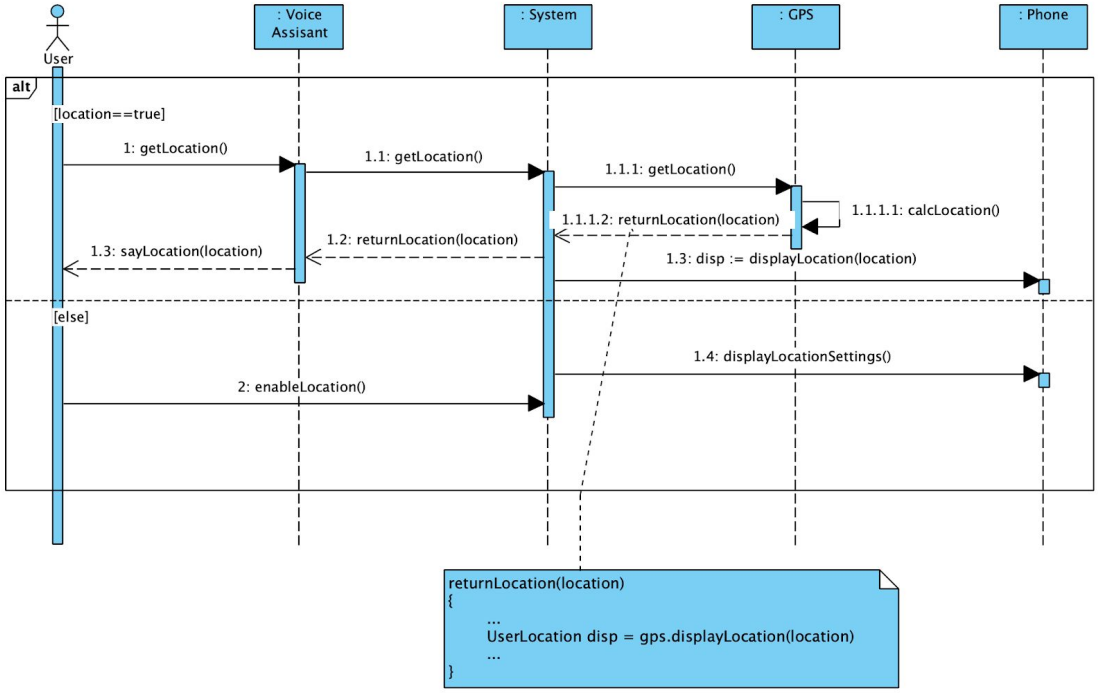
Voice Command	
-Knows the contents of user command -Verifies command received is valid	-User

Sequence Diagrams:

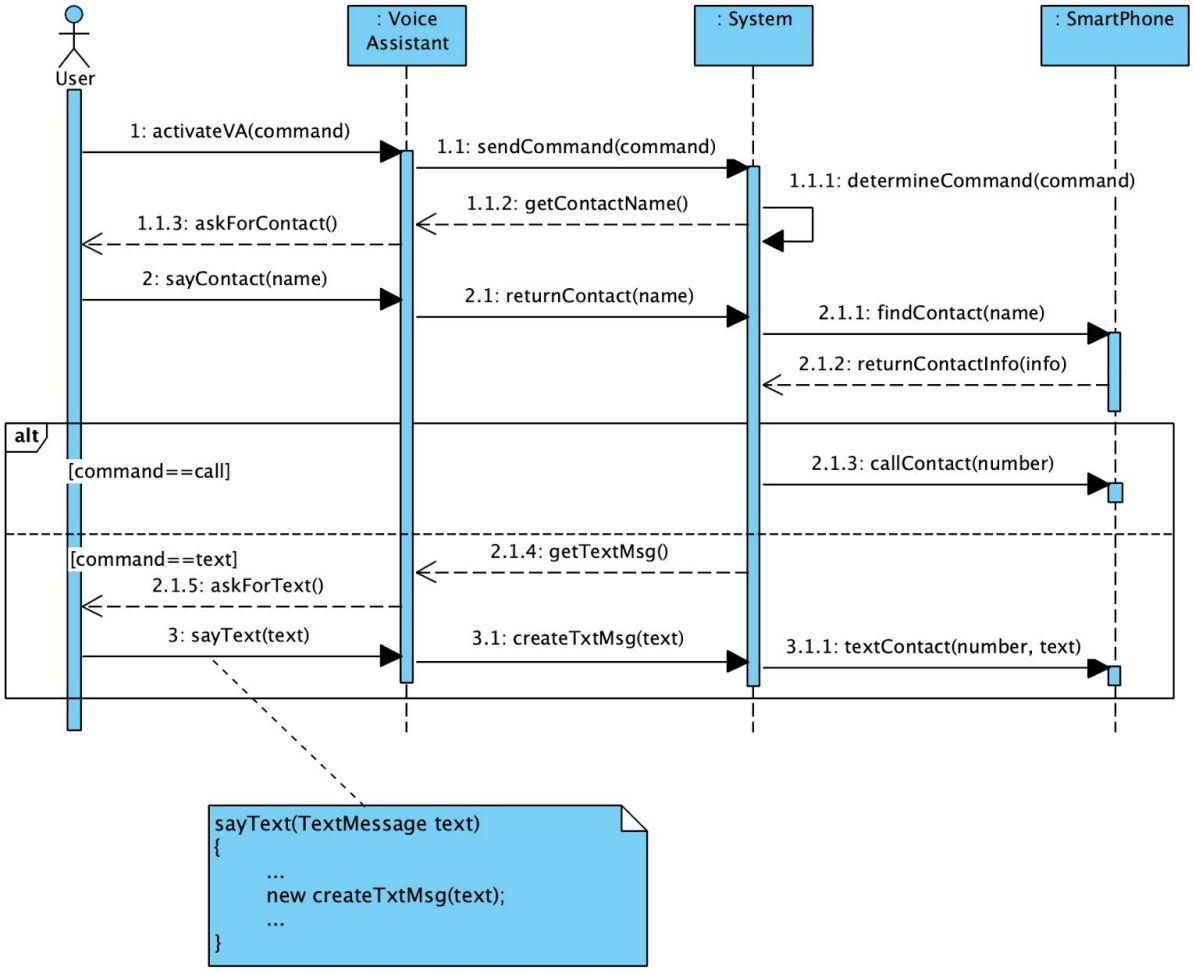
Sequence Diagram 1 - Check pulse to detect heart irregularities (Attribute Visibility)



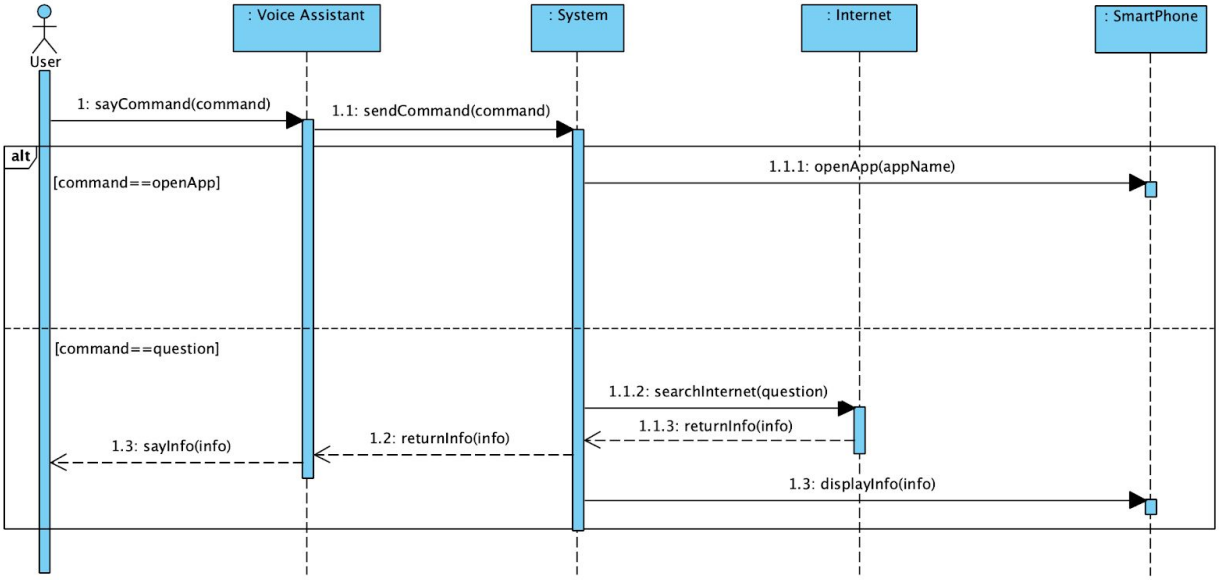
Sequence Diagram 2 - Get location with user's permission (Local Visibility)



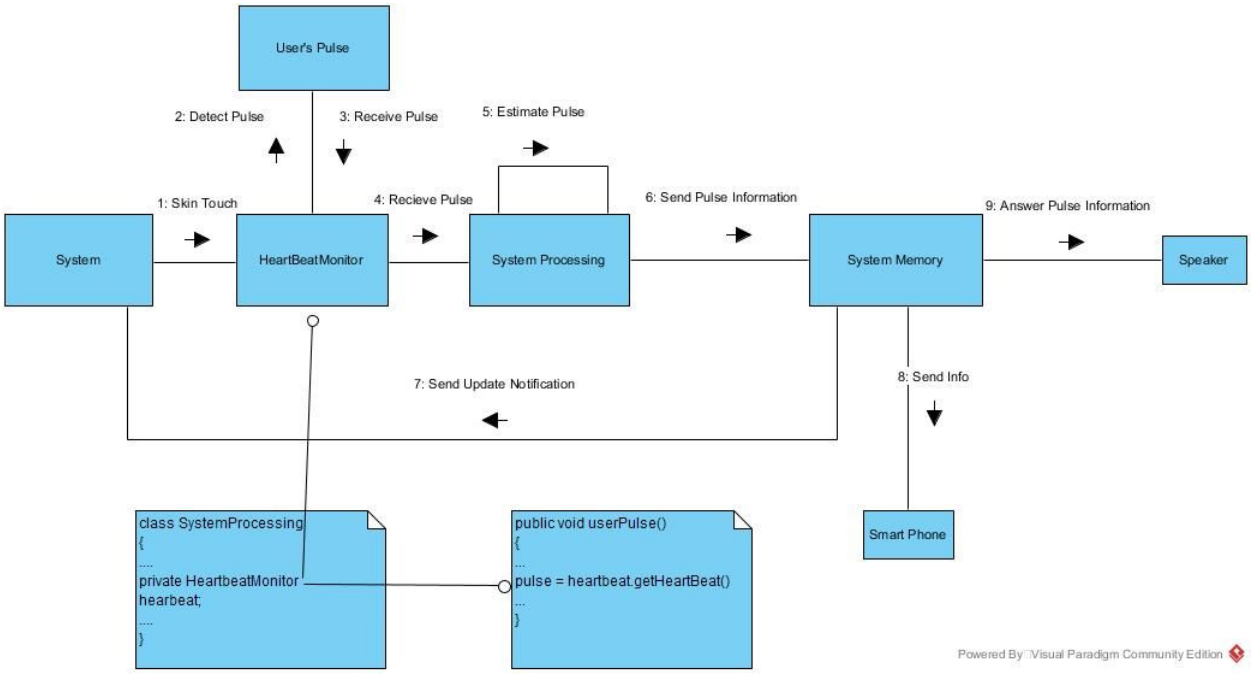
Sequence Diagram 3 - Make/receive phone calls and send/receive texts with voice trigger (Parameter Visibility)



Sequence Diagram 4 - Answer questions and open apps with voice trigger



Communication Diagram:



Mapping Designs to Code (C++):

```
//  Fedora+ Classes.cpp
//  Created by TeamZipFile on 4/24/19.

#include <stdio.h>

class Call{
private:
    string phoneNum;
    string dateOfCall;
    int callDuration;

public:
    void makeCall(string phoneNum){
        //action to make phone call with parameter passed in
    }
    void hangUp(){
        //end current call on smartphone...
    }
}

class Text{
private:
    int textId;
    string message;
    string toNum;
    string toName;

public:
    void openText(){
        //action to open the text
    }
    void sendText(string toName){
        this->toName = toName;
        //sets text class's name equal to toName parameter passed in
    }

}

class PhoneBook{
private:
    map<string, name> contacts;

public:
    string getContact(string findName){
        //find contact name in contacts map where key is equal to findName parameter
    }
}

class Smartphone{
private:
    string number;
    vector<Call> calls;
    vector<Text> texts;
    PhoneBook PhoneBook;
public:
    void turnOnBlueTooth(){
        //turns on BlueTooth via built-in smartphone function call...
    }
```

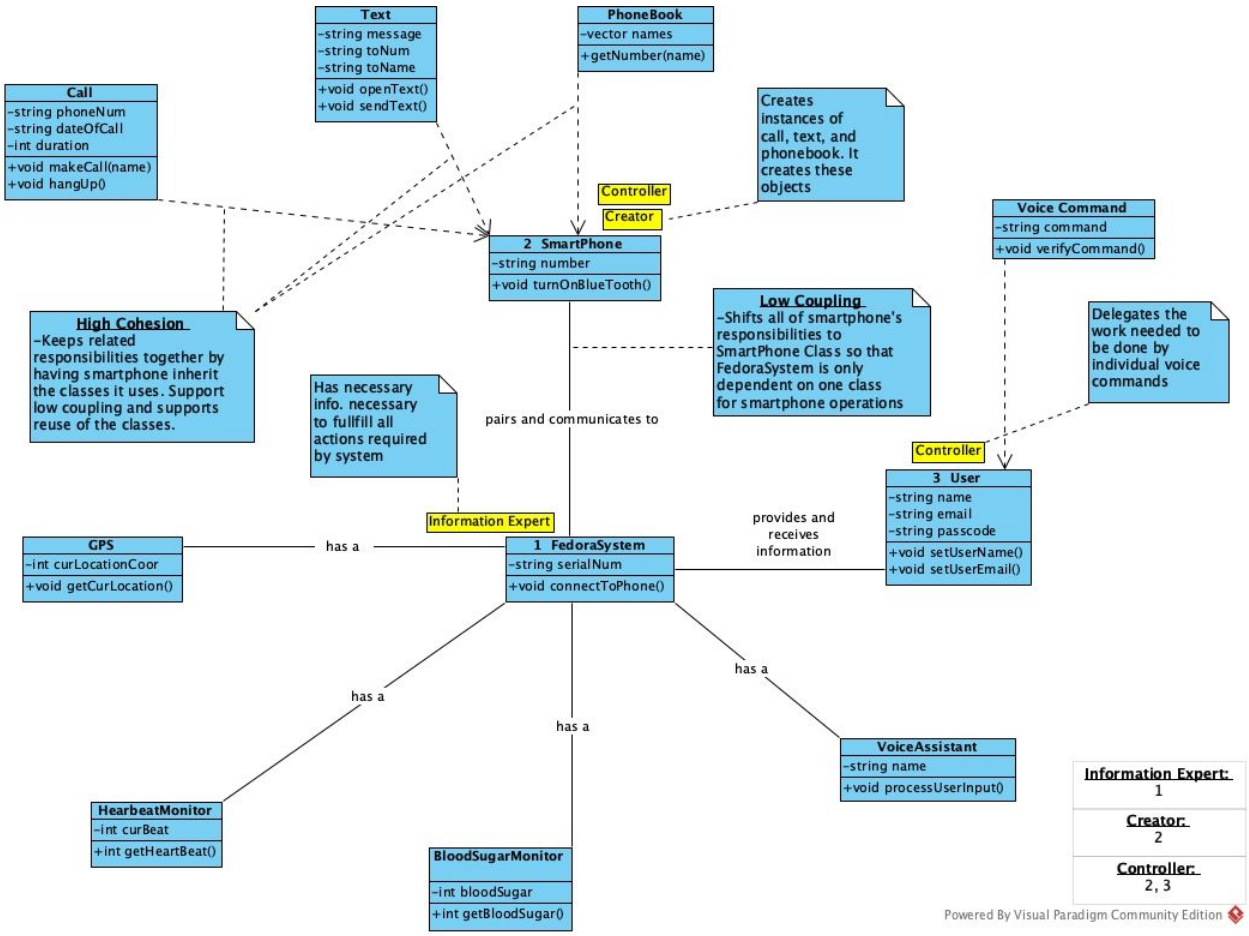
```
}

class VoiceCommand{
private:
    string command;
public:
    void verifyCommand(){
        //checks local command variable for verification
    }
}

class User{
private:
    string name;
    string email;
    string passcode;
public:
    void verifyPasscode(){
        //called to verify user password
    }
    void setUserName(){
        //sets username
    }
    void setEmail(){
        //sets email address
    }
}

class GPS{
private:
    int currentLocation;
public:
    void setCurrentLocation(){
        //will use sensors to find user's current location
    }
}

class HeartbeatMonitor{
private:
    int currentBeat;
public:
    void getCurrentBeat(){
        //will use sensors to get user's current heartbeat and save it locally
    }
}

class BloodSugarMonitor(){
private:
    int bloodSugar;
public:
    void getCurrentBeat(){
        //will use sensors to get user's current blood sugar and save it locally
    }
}

class VoiceAssistant(){
private:
    //name of voice assistant
    string name;
public:
    void processUserInput(){
```

```
        //processes user input to see what actions should be performed
    }
}

class FedoraSystem{
private:
    //fedora system has access to other classes
    GPS fedoraGPS;
    HeartbeatMonitor fedoraHeartMonitor;
    BloodSugarMonitor fedoraBloodSugarMonitor;
    VoiceAssistant fedoraVoiceAssistant;
    User fedoraUser;
    Smartphone fedoraSmartphone;

    string serialNumber;
public:
    FedoraSystem::FedoraSystem(Smartphone phone){
        //constructor calls connectToPhone to accept the phone parameter being passed
in
        connectToPhone(phone);
    }
    void connectToPhone(){
        //connects to user's phone and sets up the user's phone
    }
}
```

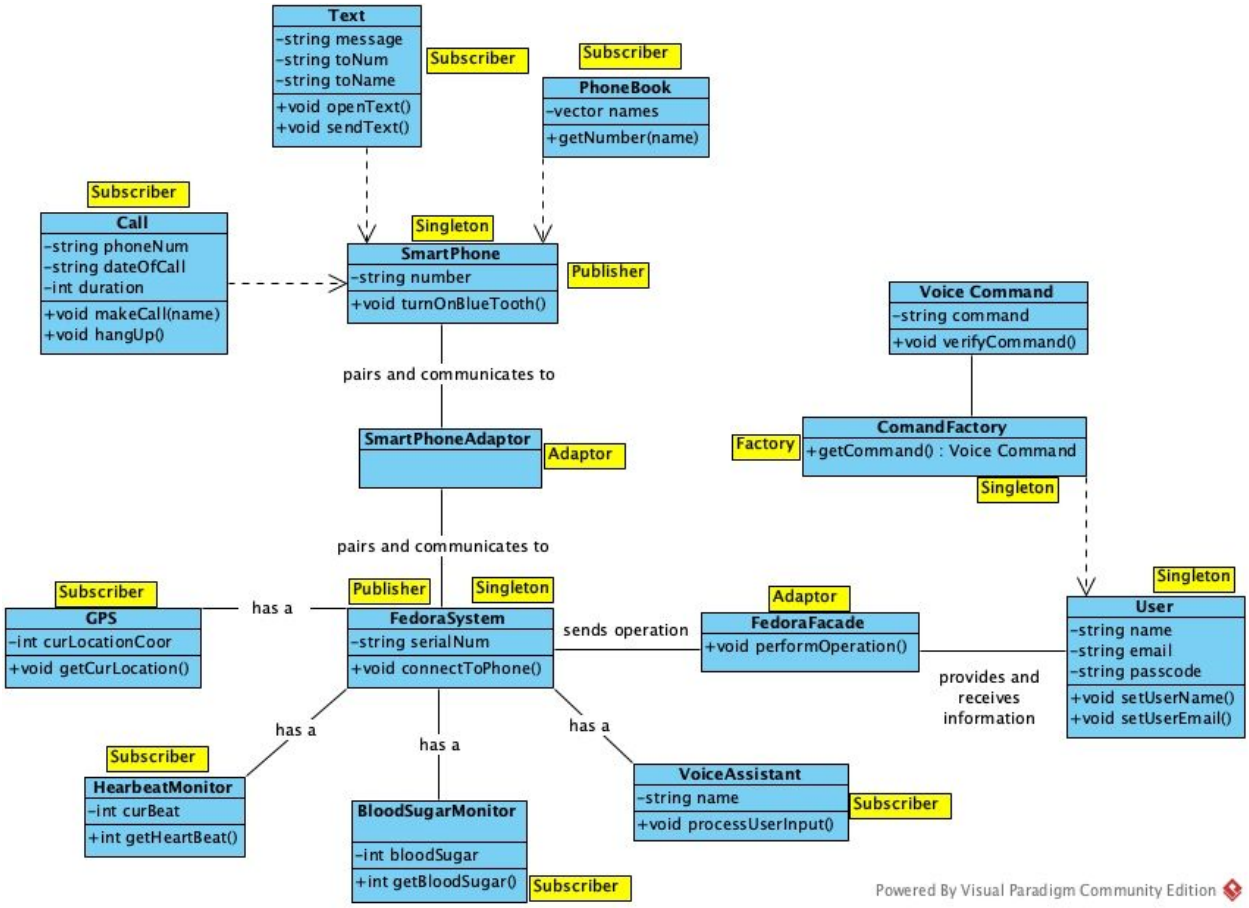
Iteration 4:

Elaboration 4: "Address Known Risk Factors and Establish and Validate the System Architecture"

GRASP Patterns displayed in Class Diagram:

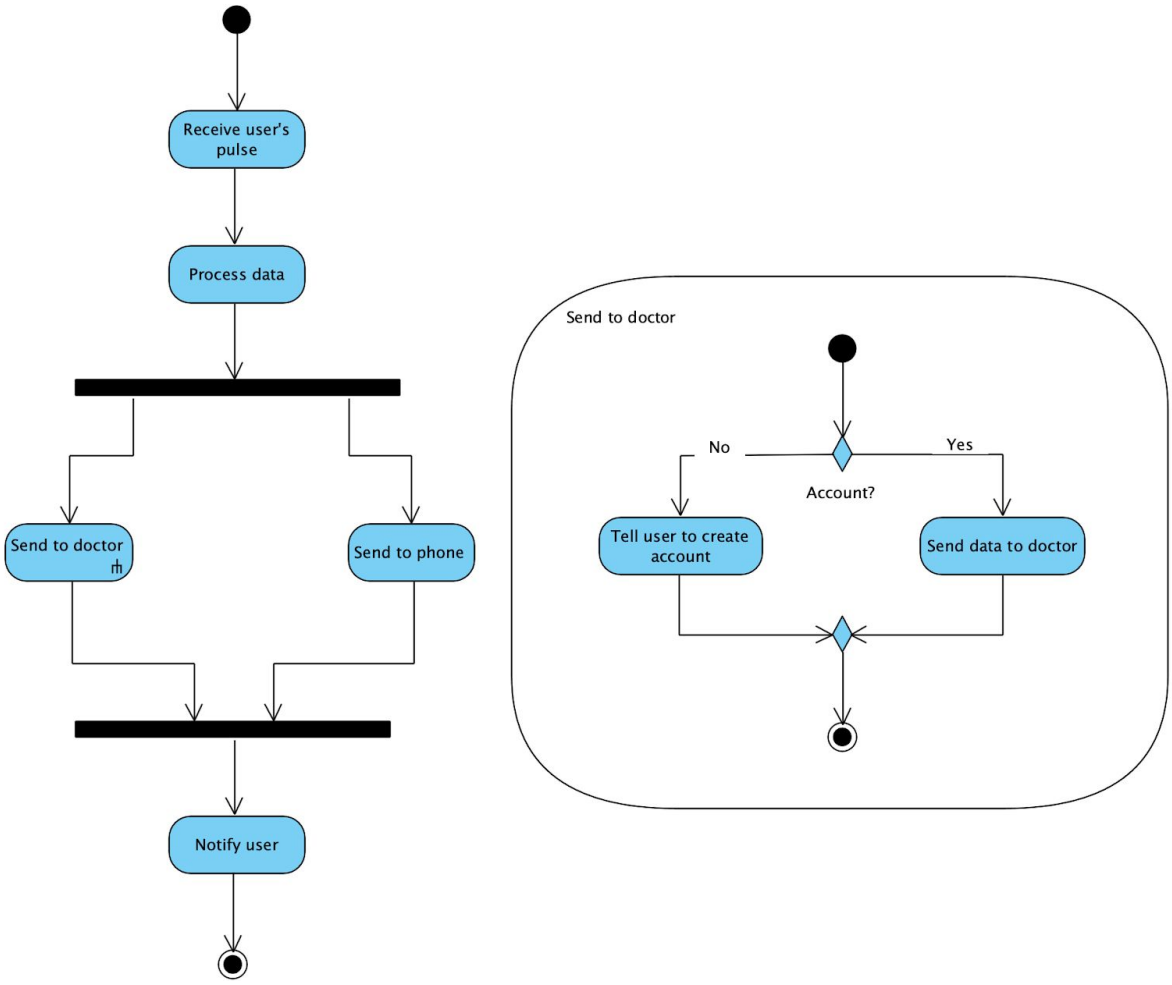


GoF Patterns displayed in Class Diagram:

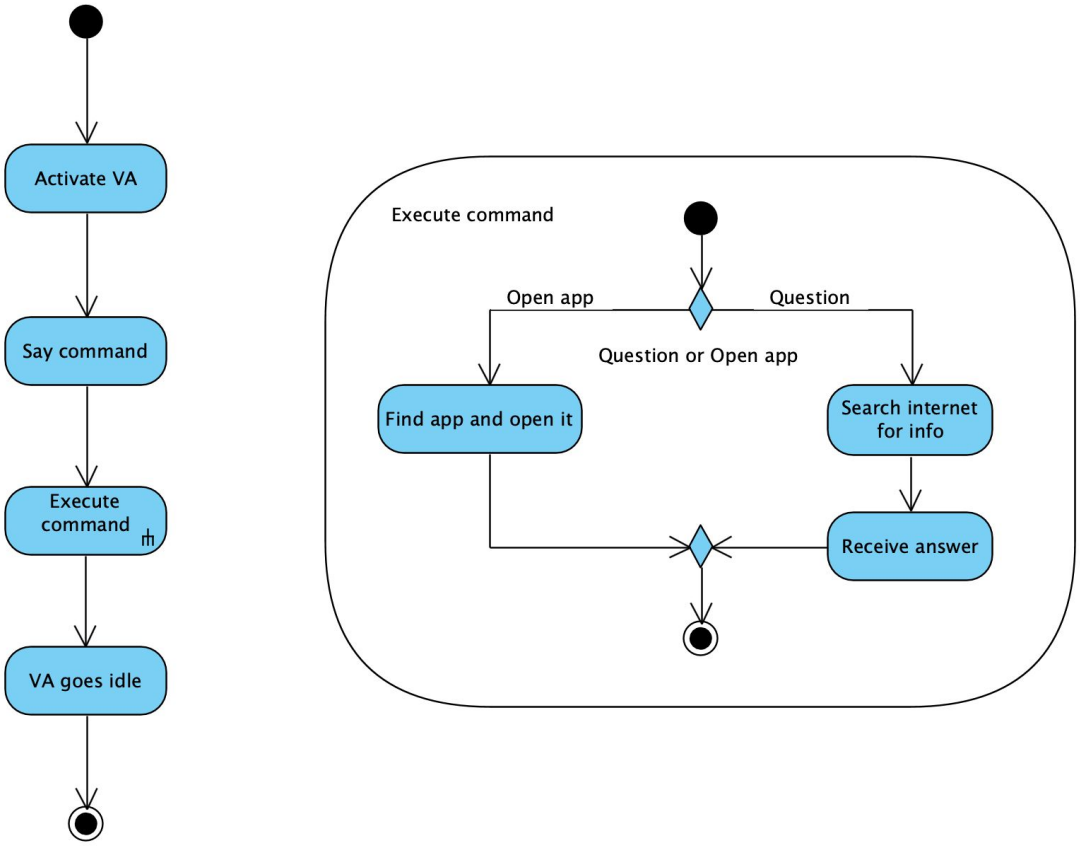


Activity Diagrams with Rake:

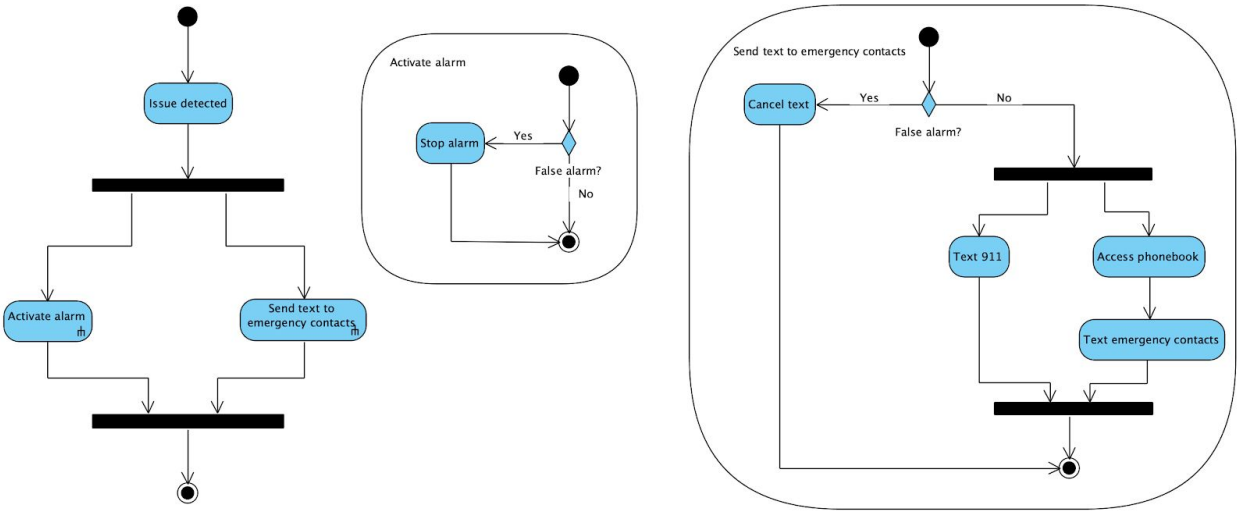
Activity Diagram 1 - Check pulse to detect heart irregularities



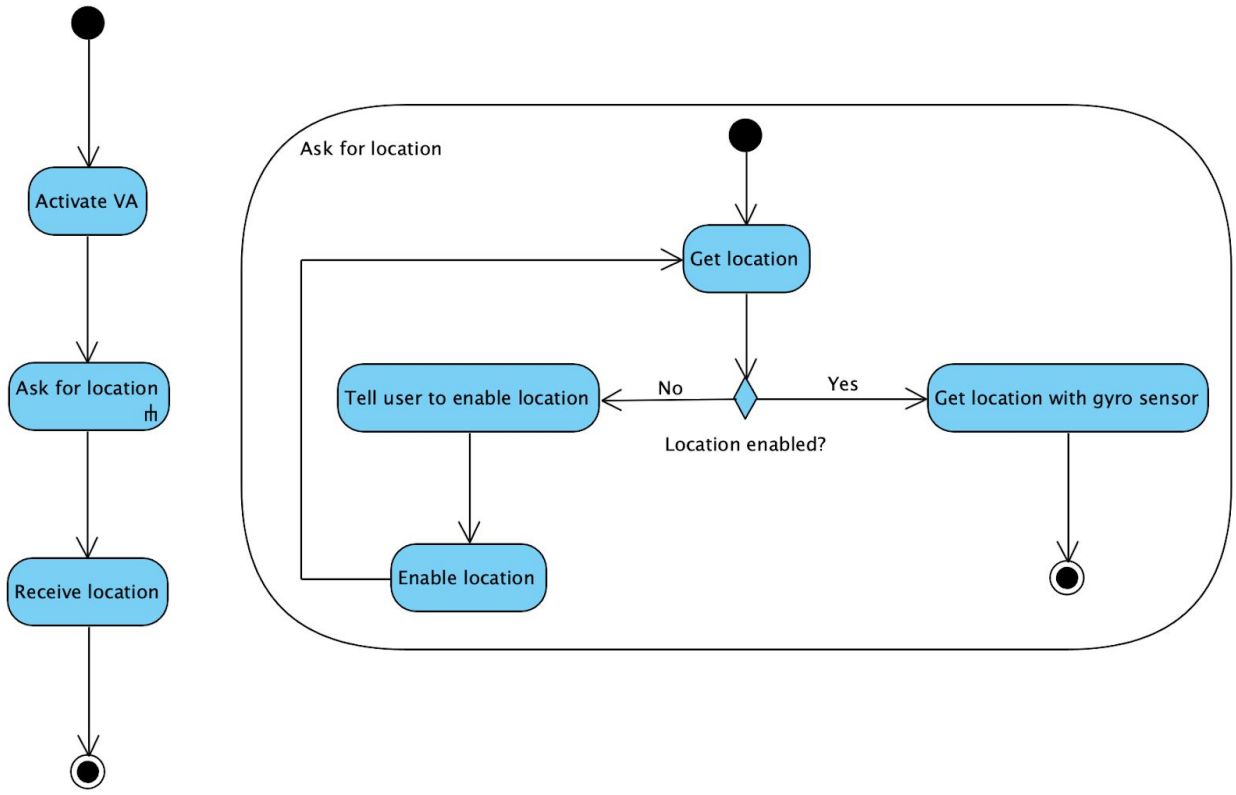
Activity Diagram 2 - Answer questions and open apps with voice trigger



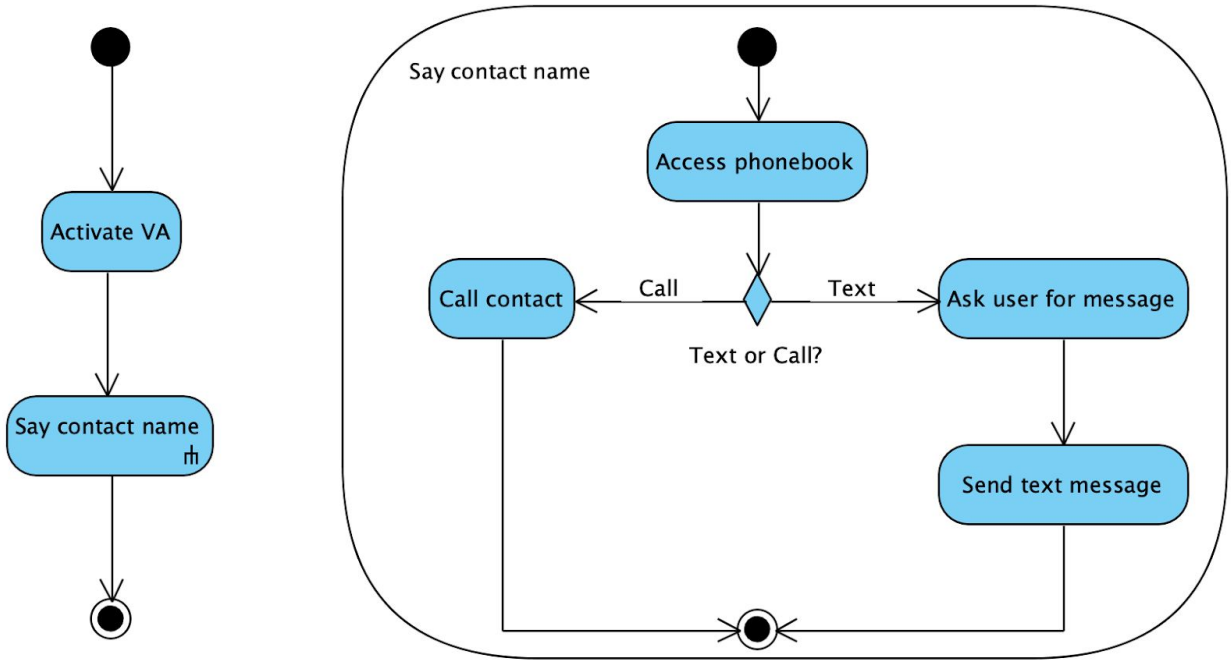
Activity Diagram 3 - Send text/activate alarm when issue is detected



Activity Diagram 4 - Get location with user's permission

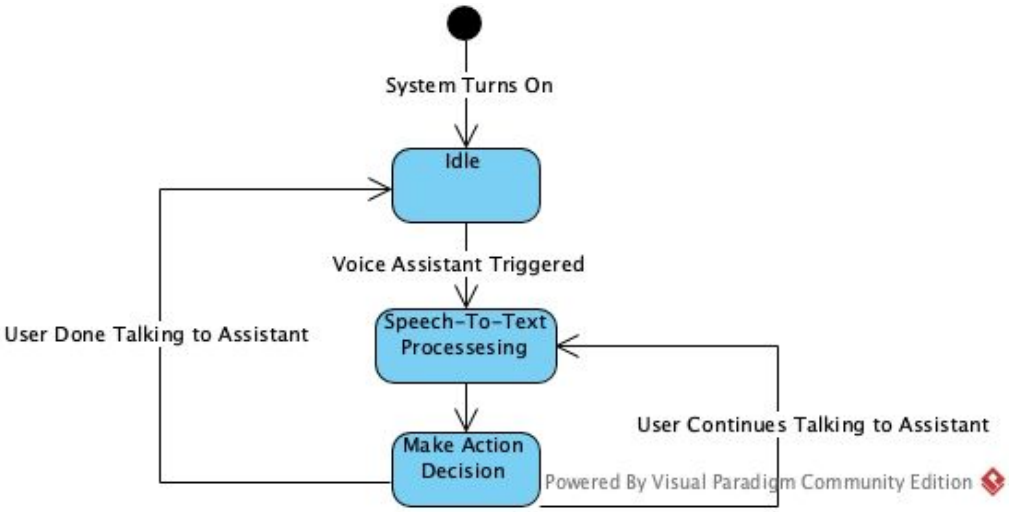


Activity Diagram 5 - Make/receive phone calls and send/receive texts with voice trigger

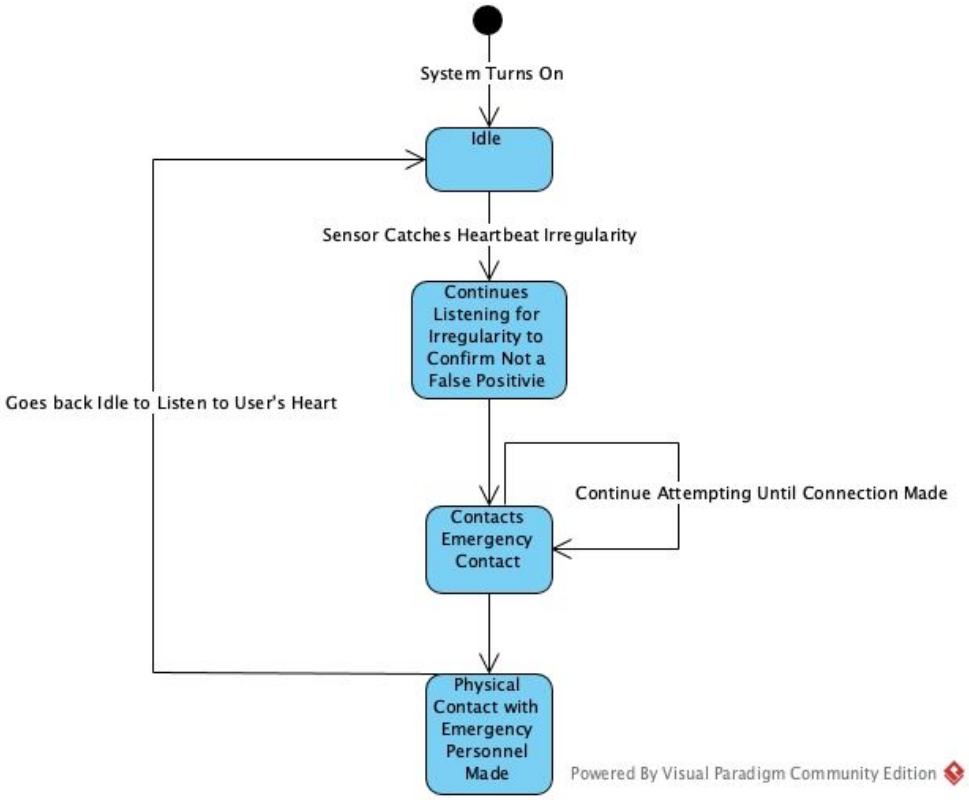


State Machine Diagram:

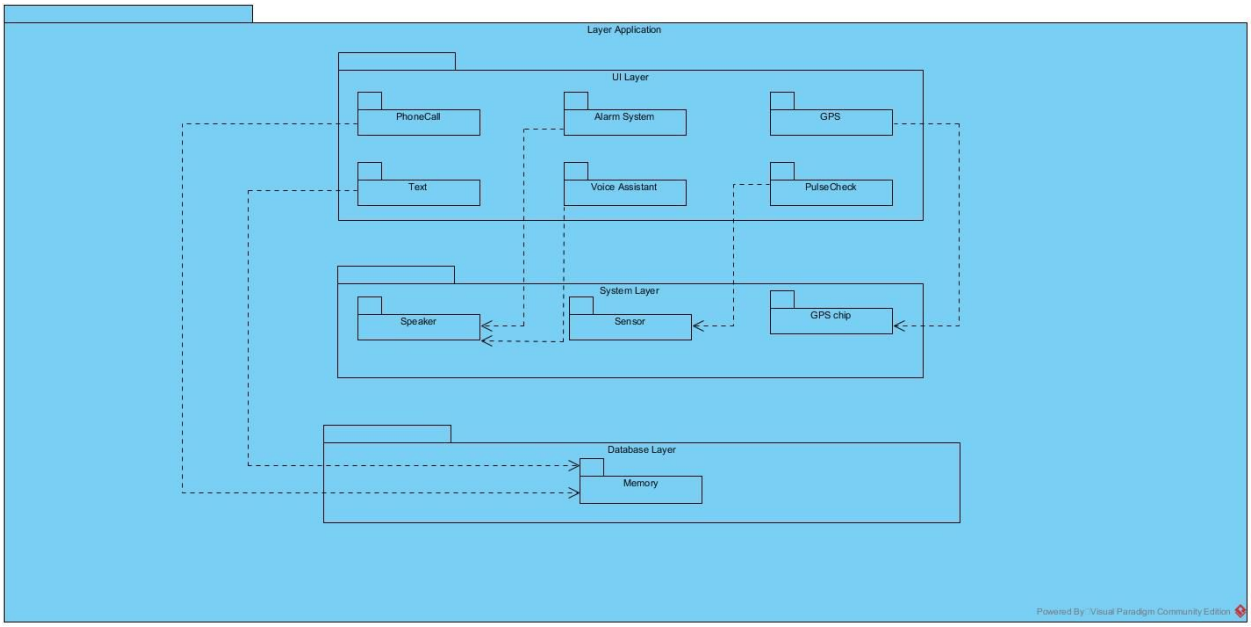
1 - Voice Assistant



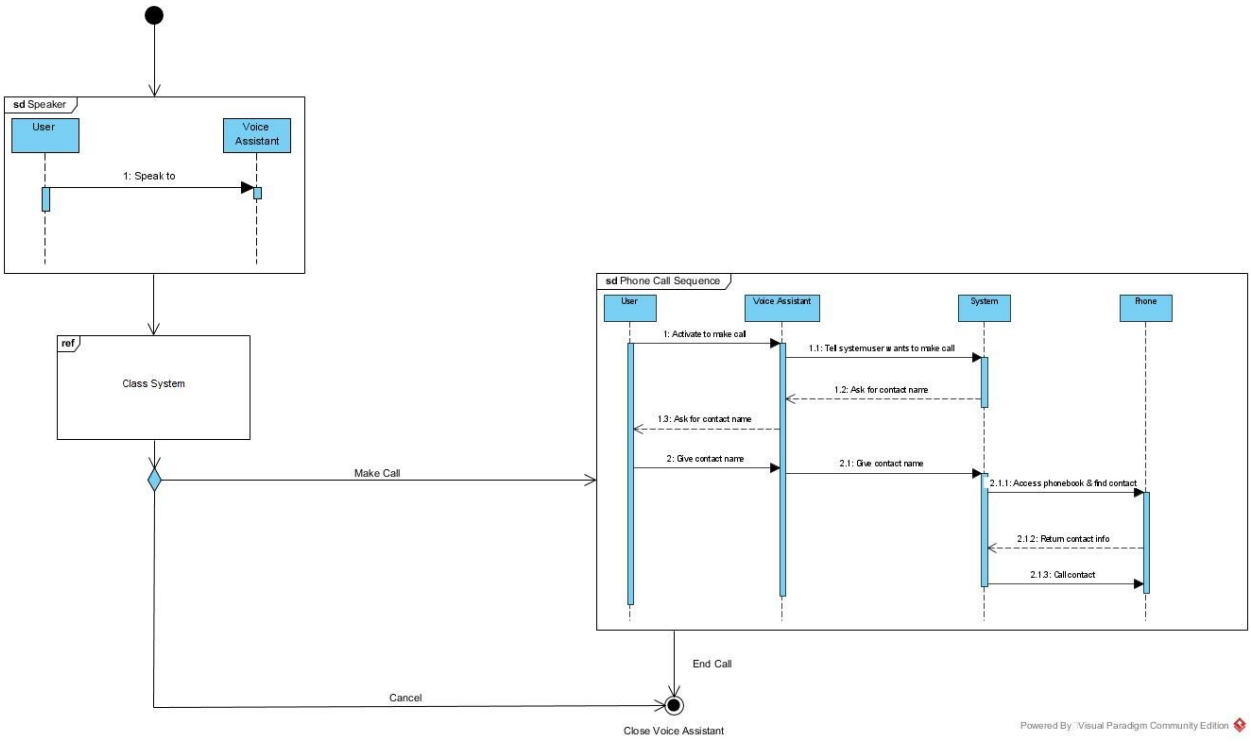
2 - Detecting HeartBeat Irregularity



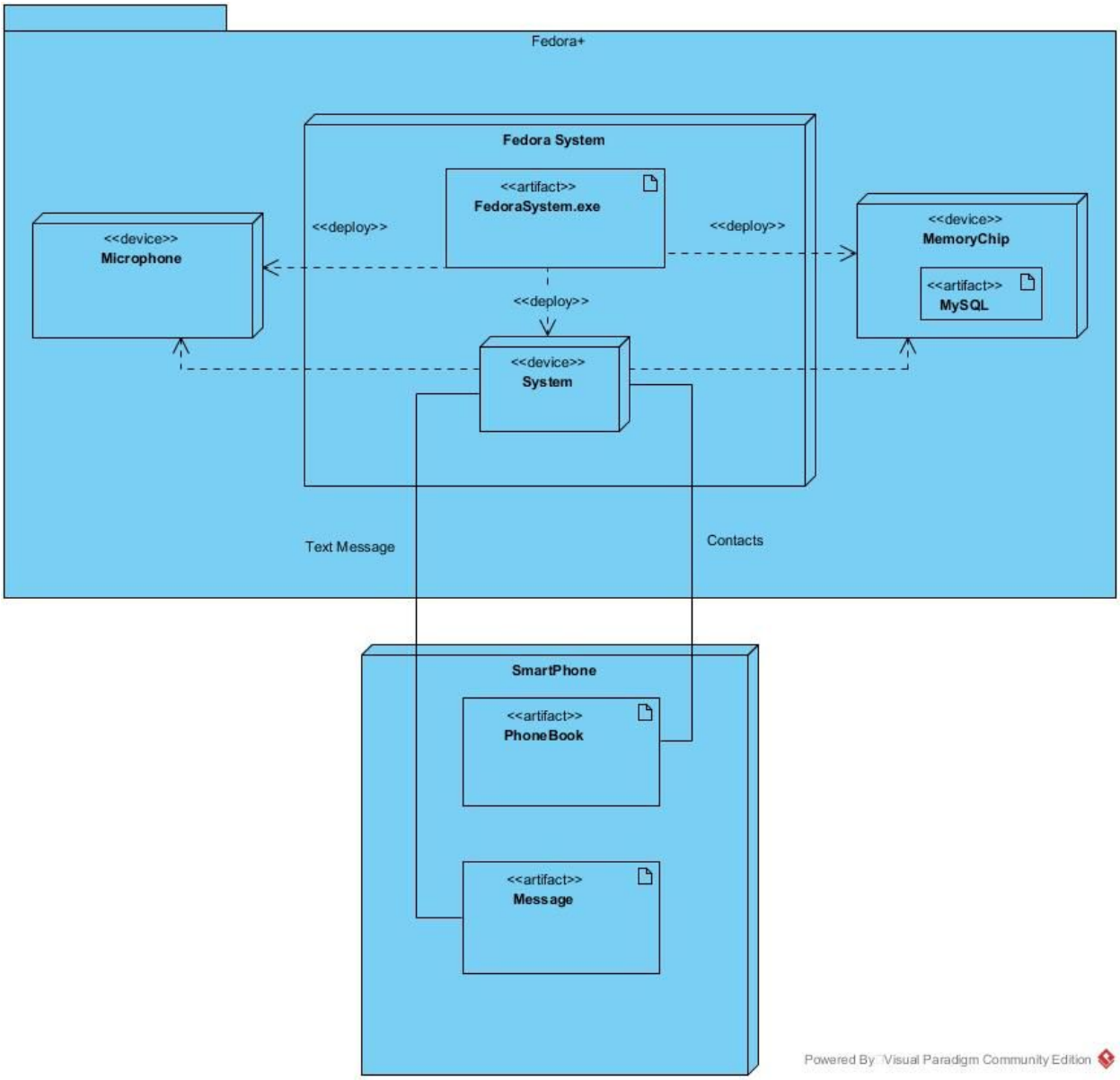
Package Diagram (2):



Use Case Realization:



Deployment Diagram:



Forward Traceability Matrix:

Functional Requirements (FR)

1. Send/Receive Texts (911, Favorites, etc)
2. Make Phone Calls
3. Location Finding via GPS
4. Detect Heart Attack
5. Count Pulse
6. Check Glucose
7. Voice Assistant

Diagrams (D)

1. Sequence diagram
2. Communication/collaboration diagram
3. Activity diagram
4. State machine diagram
5. UC Realization

	D 1	D 2	D 3	D 4	D 5
FR 1	x		x		
FR 2	x				x
FR 3	x		x		
FR 4	x		x	x	
FR 5		x			
FR 6		x			
FR 7	x		x	x	x

Backward Traceability Matrix:

Functional Requirements (FR)

1. Send/Receive Texts (911, Favorites, etc)
2. Make Phone Calls
3. Location Finding via GPS
4. Detect Heart Attack
5. Count Pulse
6. Check Glucose
7. Voice Assistant

Diagrams (D)

1. Sequence diagram
2. Communication/collaboration diagram
3. Activity diagram
4. State machine diagram
5. UC Realization

	FR 1	FR 2	FR 3	FR 4	FR 5	FR 6	FR 7
D 1	x	x	x	x			x
D 2					x	x	
D 3	x		x	x			x
D 4				x			x
D 5		x					x

Lessons Learnt:

Himanshu Patel:

After working on project 1, I learned a lot more about the software development process and increased my knowledge of artifacts. Before this project, I only knew of the basic UML diagrams, now I know that an actual project has an incredible amount of artifacts. I also got better at creating UML diagrams, creating a bunch of the same ones definitely helped. Creating all these artifacts really helped me understand how important planning is and that you can't just start coding first.

Kim-Lan Hoang:

Working on this project, I realized how much efforts and steps are needed to make software development a reality. With so many different potential use cases, diagrams, and other important information included, software engineering must look at as many angles as possible to ensure that the meeting, development, and so on, will go as smoothly and efficiently as possible.

The second part of the project became much more code and object heavy than the first part of the project. Due to this, learning and understanding how to implement from objects and codes into diagrams became very challenging. I cannot say that I have fully grasps the concept of this, but I know how serious and hardworking software engineering are.

Jayro Alvarez:

Throughout the development of this software planning project I learned many things crucial to the AGILE development process. Prior to this project, all previous projects I have worked on have followed waterfall method developments processes so this is a major change in processes. The concept of constant iterations opened my eyes to a much more flexible way to develop software. I also learned a great deal about how to efficiently apply UML diagrams and other artifacts to make further development much more simple.

After completing the second portion of this project, I can now see the importance of going in to far greater depth with all the diagrams. This includes the smaller details such as the ones presented in the GRASP and GoF patterns of design. An example of a learning experience with the GRASP patterns was when I was trying to apply it to our previously made class diagram; I found myself trying to figure out how other classes such as an adapter were needed in order to efficiently transfer information in between different interfaces.

Group Experience:

We all learned a great deal about crucial AGILE development processes typically used in the industry today. We discussed major components of this iterative process such as the essence of customer collaboration, efficiently responding to change, and the importance of having such comprehensive and iterative documentation.