Software in Support of Hazardous Environments

Project Summaries
submitted by student groups in

CS 440 - Introduction to Software Engineering

at the

University of Illinois, Chicago

Fall 2017

John T. Bell, Instructor
Tanima Chatterjee, TA
Nooshin Mojab, TA

Instructor's Notes

During the Fall 2017 semester the students enrolled in CS 440, Introduction to Software Engineering, at the University of Illinois Chicago were asked to work together in groups on a software engineering project representing roughly the first half of a complete project, with the second half of the project deferred to a following course. The goal was to give the students experience working in groups on a (relatively) large software engineering project, for which they alone were responsible for the specification, design, implementation, and validation of the project. Splitting the complete software engineering experience into two half-projects meant that students were faced with the realistic challenge of specifying and designing their new original systems in sufficient detail that future students that they had never met would be able to implement their visions, while later they would work to implement the vision created by other students.

For the development project, their task was to conceive an original software product, and to develop it through the requirements, system design, and object design phases, using a more traditional waterfall approach, for eventual implementation by other students in future semesters of CS 440/442. This work represented the initial stages of a complete software engineering project, and gave the students the perspective of developing a complete software specification and design to be eventually implemented by other software engineers.

For Fall 2017, the development project was to develop software in support of hazardous environments, defined loosely as any kind of software that might relate to any kind of hazardous environment. These could include training simulators, support calculations, or software that would provide assistance in real time during hazardous operations. In developing the initial project descriptions, detailed system requirements, system and object designs and acceptance test plans for these software products, students were asked to not limit themselves to what a small group could accomplish in a few months, but rather to go ahead and plan a project as large as it needed to be to get the job done. Major reports were due every few weeks, and at the end of the semester, students submitted a final overall report of their project results.

Overall the students completed their projects admirably, producing some very thorough software requirements, designs, and test plans for their development projects. This document includes the two-page summaries of the development projects submitted by each of the groups. Anyone interested in learning more about the systems described here may either contact the course instructor or the students involved directly. In the meantime, I only hope that the learning experience turns out to be at least as valuable as the finished software designs.

John T. Bell
December 2017

Emergency Response Officer Assistant Final Summary

Group 1: Wilfried Christophe Bedu, Natasha Rice, Maribel Jaramillo, Anne Huang

Description

In this project tools for **Emergency Response Officers** - police and firefighters – are created to help identify their potential dangers. The goal is to protect them by introducing a smartwatch and a smart heads-up display (HUD). The wrist smartwatch, which an officer wears, detects the presence of hazardous gases; and the locations of dangerous neighborhoods, such as neighborhoods that have high recorded crime rates. In such neighborhoods, the watch alerts a Control Center Operator if the officer's heart rate has changed. The HUD can detect whether the car in front of it is speeding, photograph license plates and return driver records, and use GPS data to determine the best routes.

Overall, measurement of success would rely on comparing time spent on critical tasks, such as time to detect dangerous gases, while using the system; as well as money saved on expenses such as emergency health care bills due to faster detection of health risks while on call.

Requirements

Functional: The system shall wirelessly detect whether the wearer of the watch is in a dangerous neighborhood based on information in a database. The watch must be able to read a user's heart rate. The watch must be able to detect dangerous gases tracked by a database. The HUD must display textual and graphical data.
Reliability: Network connection failure that affects multiple devices in a geographical district must occur less than once every half year.
Performance: The smartwatch shall commit no errors when analyzing the toxic gas in the air. The system must support 60,000 simultaneous users.
Usability: The smartwatch and HUD shall be a valuable resource for the Emergency Response Officers. The HUD shall be easy to use one week after a one-day training session. The smartwatch and HUD shall present information in a way that an Emergency Response Officer will not be overwhelmed.
Look and Feel: The smartwatch shall not draw attention. The smartwatch shall be sleek and professional.
Environmental: The smartwatch and HUD must be bright enough to see in all sorts of weather and lighting conditions.

Design Goals

Reliability: The graphics should not be so complicated that the system makes strenuous demands on the network that cause connection failure more than once every half year. Because the goal of the project is to quicken the response time to officers' risks, and because failure will significantly increase that response time, reliability is a more important design goal than usability and look and feel.
High privacy and compliance*:* Watches may not transmit the location and vital statistics of a wearer without their consent. The product is for a government entity and contains personal health data. Privacy and compliance should be higher priorities than performance and space. Additional code that comes at the cost of the complexity, yet strengthens privacy guarantees, is appropriate to invest time on.
High adaptability*:* The product shall be adaptable to the custom hardware needs of different cities. Therefore, adaptability should be optimized. This lends its way to the *repository architectural style* for the Data Processing subsystem. Having one central repository for all data processing operations will keep the data processing loosely coupled from all hardware subsystems that rely on the data subsystem, so that specific cities can substitute alternative hardware subsystems and alternative databases. This goal also leads to the *facade design pattern*, so that if the system is extended to other hardware products the specific data operations do not have to be redesigned for every specific subsystem. Even though this layer of indirection between each subsystem and the data adds performance overhead, this design goal takes a higher priority than performance.

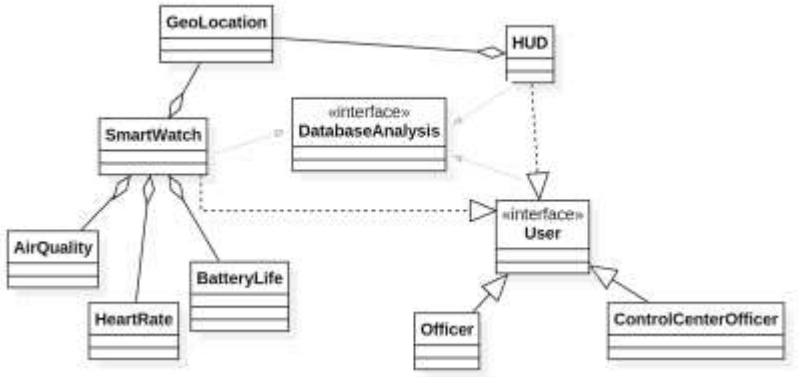
Subsystem Decomposition

The Data Storage subsystem features a unified relational database and can be directly accessed only by the Data Processing Analysis subsystem. The Data Processing Analysis subsystem is the central repository for all data processing operation. All hardware subsystems access the Data Processing Analysis subsystem using a uniform interface. These hardware subsystems are the HUD and the Watch. The Watch subsystem interfaces with the Heart Rate Monitor and Air Quality Analyzer. Both the HUD and Watch interface with Location Analyzer.

Global Software Control

Many of the initial requests to the software will be based off triggering external events, such as entering a high-crime neighborhood or pressing a button to request an air analysis. Thus, the global control flow is event-driven. Some subsystems depend on information provided by other subsystems, so they have to run synchronously opposed to asynchronously. The Heart Rate Monitor subsystem depends on the Location Analyzer subsystem because the heart rate isn't monitored until an Emergency Response Officer enters a high-crime neighborhood.

Class Diagram



Persistent Data Management

The persistent data management strategy is to use a relational database. This project defines four schemas for the database: Officer, ControlCenterOperator, ChemicalInfo, and TicketInformation. Both Officer and ControlCenterOperator have an “is-a” relationship with User schema, meaning that they can carry required unique information and share traits. The ChemicalInfo Schema will hold information for all known chemicals and the TicketInformation schema will hold data from any tickets given out.

User Interface

The software will prompt Emergency Response Officers to interact with the product interface to acknowledge when they need assistance. Emergency Response Officers also have the ability to start certain functionality, like finding the best route, via the user interface.

Test Plans

Unit:

GPSAccuracy: The test driver outputs a subject’s coordinates and compares this to the expected results.

LicensePlateTranslation: HUD scans license plate and compares returned driver results to oracle.

Acceptance:

Acceptance test plans include the accurate detection of 98% of toxic gases by the smartwatch.

Electromagnetic Tracking Visor:
Final Summary

Group 2
Members: Kevin Choi, Joseph Galante, Mark Hallenbeck, Vincent Pham

The Electromagnetic Tracking Visor (ETV) is a product that will be used by first responders in a building fire emergency. It comes in two parts, the visor that is worn by first responders, and the ETV sensor system that is implemented in the building. The system, when activated during a fire, will transmit electromagnetic waves and record events that happens in the building. When entering a building that is an alert state, the visors will receive data from the system and then process that to find civilians and keep track of other first responders. It will also generate an augmented reality display of directions to where the nearest safe entrance or exit is. Along with directional displays of personnel and places, the visors will also measure the heat and toxicity levels of the room, such as smoke and carbon monoxide.

With a product that is going to be used in high risk situations, there needs to be a guarantee that it will work when it is supposed to. The ETV will undergo extensive tests that will check for any faults in functionality. The visors will be used in scenario tests that will check its ability to properly calculate the heat and toxic levels and then display it on the HUD. The visors will be also be tested for its ability to generate an augmented reality display of the directions as well as the blueprints of the building. The visor and the system will need to test for its maintainability and dependency so it can reliably be used multiple times. The visors will need to be brought back to a technician to be unit tested again, and the system will need to investigated for any data it retrieves and recalibrated to be ready for another emergency. The ETV will undergo remote monitoring through many tests to make sure its requirements are fulfilled.

The hardware design of the ETV will look similar to the visors the firefighters have traditionally used, only the ETV will have a HUD to display information about the environment. The helmet will have a device that will retrieve the data sent by the system to process and display onto the HUD. The visor is to be as simple as possible so the first responder is not bothered by clunky or complicated augmented reality. The system will be a black box that will be hidden from view, and off-site to process data when an emergency happens. It will connect to sensors around the building which will collect data and then send it out to the visors.

The software of the ETV will involve a View class that will generate a display for the visor. In order for the display to even be created, the visor will take in information from the system, and the data will be sent to different classes within the visor's program. The classes the data will go into includes the Temperature class, Building class, and Direction class. The temperature class will take in data about the temperature and will return a value in fahrenheit back to the View class. The Building class will process data involving the layout of the building to create a blueprint of the building's structure, then returning a 3D image back to the View class. The Direction class will take in information about the location of entrances and exits, as well as the location of personnel, and return a path that will lead to the destination back to the View class.

The ETV System will have classes similar to the ETV, but it will take in information from sensors around the building process information from them. The System has an Information class that will send out information to the visors. The classes that will process the information going into the System will be the Heat class and Building class. The Heat class will process information regarding the heat in a certain part of the building. The Building class will have information about the building, such as its layout and the people inside of it. These may not be all the classes, but they are the most important ones.

MEAA – A Miner's HUD

Group 3 – Anthony Leon, Andy Cervantes, Meghana Sanjay, Emmanuel Escobar

Project Description and Overview

MEAA is a heads up display that will be available to miners to aid them during their daily mining activities. The main objective of the product is to inform the users about potential health risks brought upon from the hazardous environments that they currently work in. This product will use outside hardware such as a display, air sampler device, night vision camera, an on board computer and a storage medium. Each part will work together to give the miner a recommendation for their health. The product is essentially a device that can really help miners tracking their short-term and long-term health status. The HUD is designed to fit comfortably over the face and display live data being aggregated from the HUD's onboard sensors to inform them of hazardous chemicals within their working environment.

A typical day for a miner in a hazardous environment exposes them to many harmful chemicals. The amount of time they are exposed in these environment can greatly impact their health and possibly cause death. There are regulations that all mining companies have to follow and maintain this product will go more in depth. Rather than create a product that will be used by one worker and relayed to the other workers, each worker should have the proper information for themselves to make an educated decision regarding their work. The heads up display will give proper recommendations based on live data that will be collected in real time. Again, our product only does recommendation, it is up to the miner to take action. The last feature of the product is a night vision that will allow the user to better see their environment. This feature is recommended to help the miner vision-wise and also make the product more usable in the workday.

Features:

Key Functional Requirements

- The system must provide the user with the current ppm of the user's current environment.
- The system must retrieve the accurate substance ppm or hazardous PPM value from database
- The system must provide the user the proper comparison between the current and stored ppm and must calculate PEL formula for the user.
- The system must provide the user with an accurate recommendation of how to proceed in their current work day or the user's total workday.
- The system must provide the user ability to use the night vision sensor.

Key Non-Functional Requirements

- **Performance -** The startup time for the application will be under 10 seconds for all weather conditions
- **Dependability -** No air sampler data shall be lost or damaged in the event of a failure.
- **Maintenance -** All outside components should be operable and checked periodically.
- **Usability -** The product should be easy for users to access without the use of their hands.
- **Operational and Environmental -** The product shall be usable in low visibility conditions.

Testing plans:

Testing will take place for all current and future features of the product. Some include:

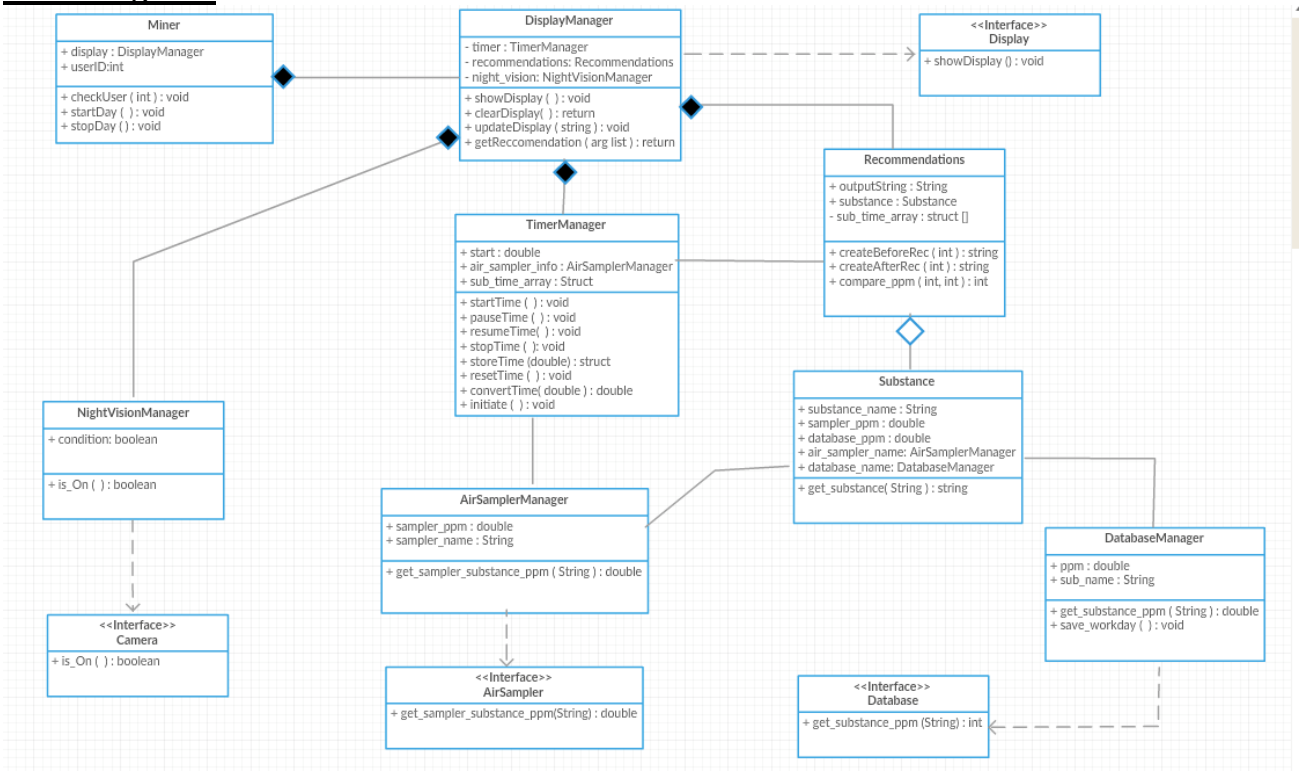
1. Air sampler is able to get environment reading

2. There is a proper connection to the database and that for each specific query the correct value is returned from the database.
3. Recommendations are displayed and visible for the user
4. Permissible Exposure Limit formula gets proper parameters and exports proper readings.
5. Night Vision feature when turned on does display and enhanced image

Design Goals:

- **Performance -** The system should be quick and responsive. When the user requests something the system should acknowledge and handle the request as quickly as possible. The calculations should be as accurate as possible even if speed must be sacrificed since we are dealing with health risks. The system can't rely on rough approximations and assumptions. However the speed at which the application starts up should be reasonable under all weather conditions.
- **Dependability -** The system should not crash, but in case it does it should be responsive as soon as possible to avoid affecting the health of the user. All operations should continue normally even under extreme conditions.
- **Maintenance -** The software should be able to adapt to many different hardware components.
- **End user criteria -** The system should always support the work of the user during the entire work shift.

Class Diagram:



HALS: Hazard Assistant for Land and Sea

Final Report Summary: Design Plans

Group 4: Asma Rashid, Hasan Iqbal, Mohammad A. Chaudhry, Saba Zikaria

Summary

HALS is an autonomous robot that relies on numerous sensors and data from external systems to monitor, detect, and analyze various entities such as, animals, chemicals, and natural disasters that can be potentially hazardous to a human being and alerts the user of the dangerous environment. HALS is designed to be the first non-human responder to search in hazardous situations without physically performing an action that would save anyone such as performing medical assistance.

Requirements

There are several requirements HALS should meet. The functional requirements would specify what HALS must do, and the non-functional requirements would specify what are the constraints it must meet and how should it operate. Some of the functional and non-functional requirements of HALS is briefly discussed below:

Functional

In the realm of functional requirements, HALS would need to be able to acquire data from various sources while operating in the environment. Different providers including GPS, guidance systems, navigational systems would be working together to provide HALS with the data it needs to operate. It would need to have visual sensors which would capture surrounding environment and perform statistical analysis on it to make decisions on what to do next. It also has alert capability based on its predictions. It would also be able to operate on a number of types of data including numerical, textual, visual and auditory per se. It would have wireless connection to the home base. It needs to have authentication and authorization capability to make sure no unwanted person is giving this system any commands. It would also have the ability to manipulate nearby objects with its 'hands'. This is necessary to rescue a trapped individual where something could be needed to move away.

Non-Functional

In the non-functional requirements area, HALS must meet several requirements. It needs to have a solid performance capability so that it doesn't break down while operating. It can exacerbate a hazardous environment if it itself breaks down. It needs to be very reliable. As in a hazardous environment, there are not much support for proper electronics and electrical systems, it must be maintainable and supportable with minimal cost. Most of the residents where it would operate would not be too familiar with such hi-tech system, so the look and feel of HALS needs to be congruent with people's usual perception. It must not look too outlandish or the residents may not want it to operate. Also, while operating under water, it needs to blend in with the underwater environment so that it itself does not become a target for hostile life-forms. It should be able to withstand a substantial temperature or lack thereof. It should not be too light so as to lose stability in a windy environment or should not be too heavy so that it becomes immobile in a

muddy environment. The legal and cultural restrictions should also be taken into consideration as it's non-functional requirement. For example, it cannot be deployed in an Amish community as they would not be as welcoming to it as other neighborhoods. If it breaks down while operating, the down time should not be too long either. It needs to be recoverable quickly and with minimal resource. Because in a Hazardous environment, the value of a split-second time is more than in a normal situation.

Test Plans

HALS is to be tested repeatedly and continuously throughout its development process. Each individual component must be tested on a standalone basis and in conjunction with other components. Some of the features to be tested are animal and chemical detection, communication systems, and terrain mobility. In order for any of these systems to pass, they must be able to maximize it's expected result. The expected results are illustrated in some of the following cases:

Test Scenario	*Test Step*	*Expected Result*
Verify all sensors work	Use sensor manuals to verify sensors work	Sensors work based on manual specifications
Verify battery sustains work load	Run system in all conditions measuring the battery usage	System should be able to operate fully without a recharge for 3 hours
Verify animals are being detected by system	Place life size animal balloons/toys in front of bot to evaluate	System detects and alerts the user of potential threat
Verify humans are being detected by system	Place manikins or living humans in front of bot to evaluate	System detects and alerts the user if human is in danger

Design

The system will be designed to make it easily maintainable and still leave room for improvements. Each sensor and or major action that HALS will be able to perform will have its own subset of classes. For example, the terrains that HALS will be able to traverse will maintain a parent Terrain class with multiple instances of the Terrains as it child classes. Which terrains to implement first and what to add will be done easily by modifying the class definitions. HALS's user will communicate through the accompanied tablet. All control functionality and communication transmissions will be performed through this means. The UI design will allow the user to log in and customize the interface to their liking. It can be rearrangement of the app and widget locations to color schemas and even installing add on or extra packages. The systems first concern will be to make it as easily usable and maintainable to maximize the efficient usage of the robot and the developers. If in any case, the user requests an update to the system, the developers should be able to modify changes because the system is broken into multiple well organized subsystems.

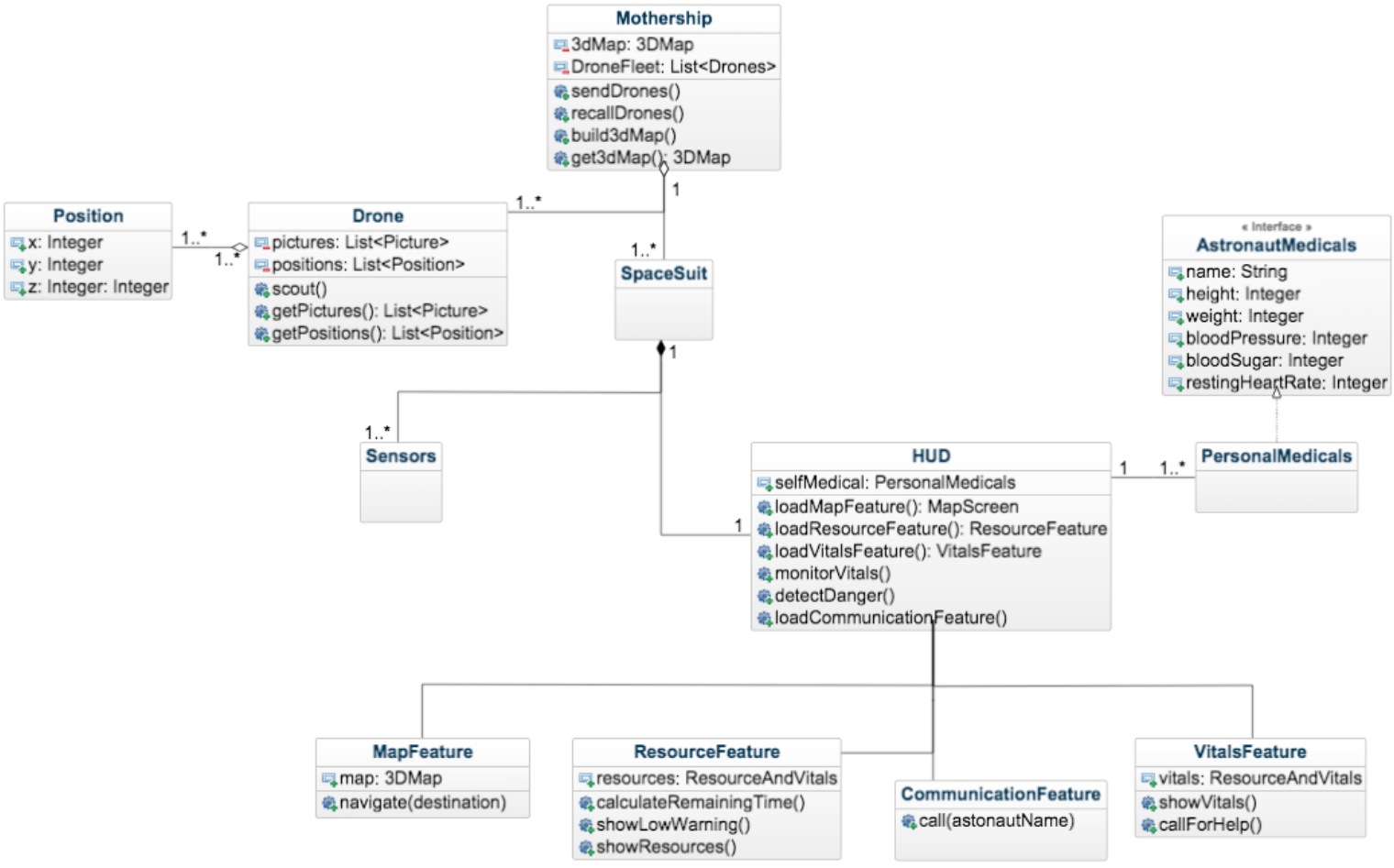
For more information, refer to the HALS documentation.

SIIMS: Final Summary

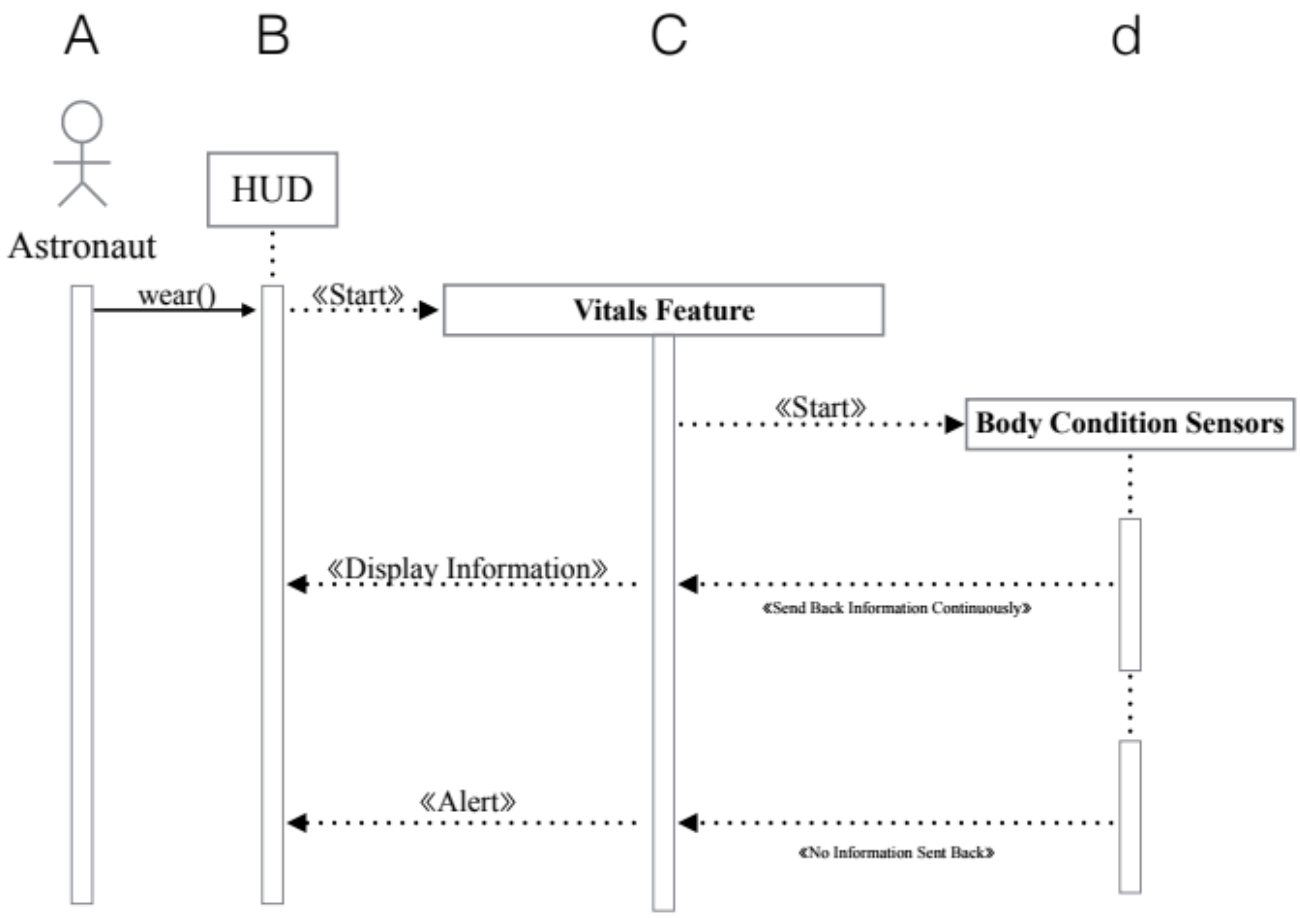
Group 5 –Ahmed Metwally, Royden Onishi, Lingfei Zhang, Wallace Zhen

The design goals for SIIMS are to be fast, accurate, and easy to use while security is not as important. Speed is very important for the astronauts in an alien environment since they may be running away from an alien attack or a volcano explosion. During these life or death situations, having system lag may result in loss of life. The accuracy of the 3D map is very important since one wrong step could result in being swallowed by quicksand. Ease of use is important because when one astronaut is calling for help, he needs to make the call in preferably one step or zero steps. Having a five step procedure to make a help call is just not acceptable in such an environment. Security is not as important because the hackers on Earth will not be able to connect to the SIIMS software that is on a different planet. It is also highly unlikely that an alien specie will be able to program in the same language that could hack SIIMS.

An overview of our class diagram can be seen from the following picture.



The sequence diagram for the sensor detecting the vitals is shown in the following picture.



There are 3 core features that we will have fully implemented 2 years prior to the 2024 launch. The 3D map feature, the resource and vital integration, and the communication feature. The nonfunctional requirements will be developed in conjunction to the core features. The development effort will begin in 2018. This gives 4 years to launch the first iteration for the astronauts to use to complete their training. More features will be added after the first launch. There will also be continuous support to fix bugs. The following cost will be for the initial launch of SIIMS.

IsoCare – Final Project Summary

Software in defence of public health

Team 6 - Allison Channic, Jesse Tapia, Sundeep Kiran Anne, Vladimir Stremoukhov

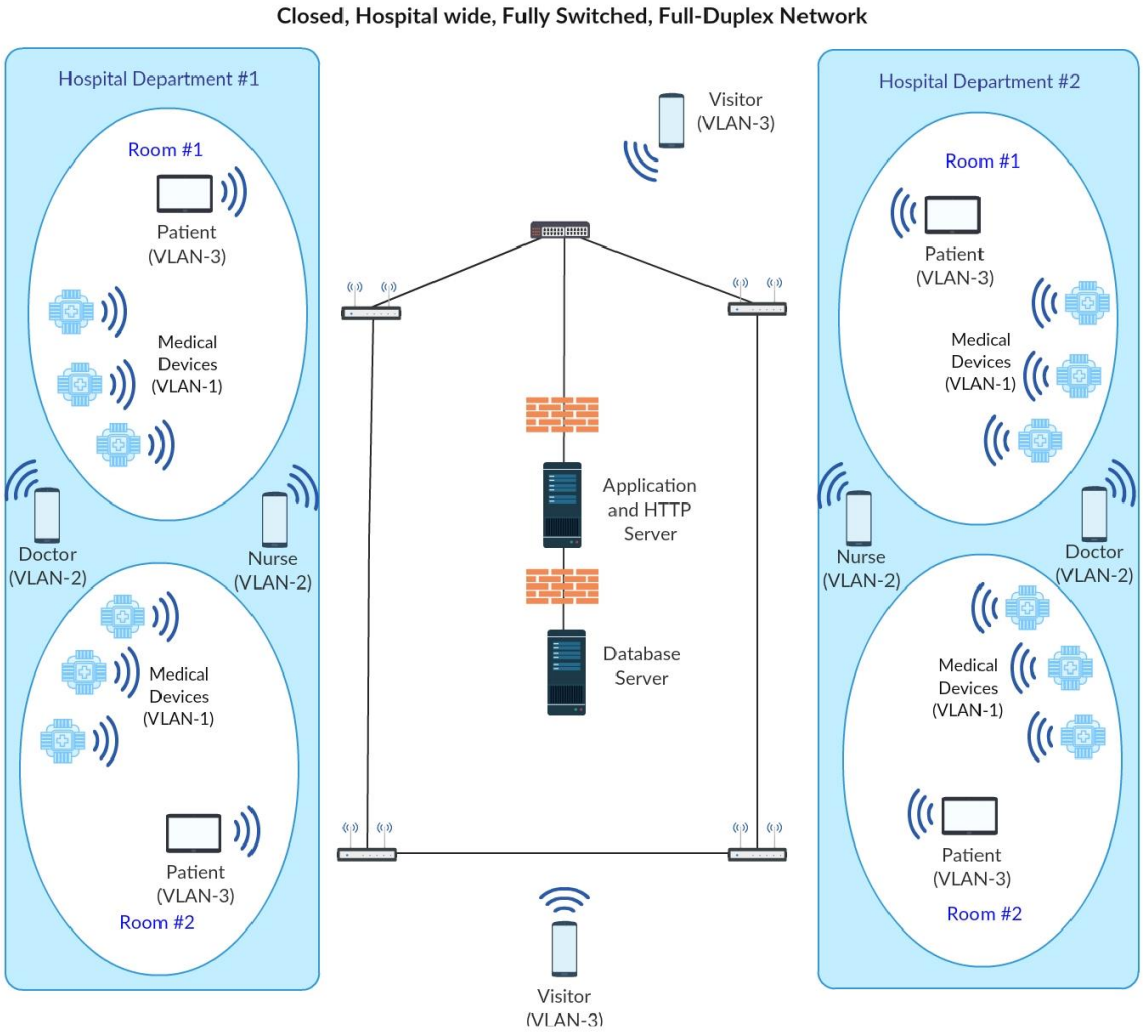
Introduction

The idea for this software was inspired by the desire to help improve current situation with care for highly contagious patients in a hospital environment. The occupational hazards of medical staff (nurses in particular) working conditions, oftentimes play a significant role in overall dissatisfaction and high turnover rate of nursing staff in hospitals across United States. Furthermore, there are often cases of outbreaks of diseases which start on the hospital grounds and spread into outside world when someone from nursing staff gets infected and carries it outside. According to a 2004 report, published by Centers for Disease Control and Prevention, 88% of surveyed nurses reported safety and health concerns related to their working conditions. 55% of responders said they would not recommend nursing as a profession.

In light of these findings, we have tasked ourselves with creating a software that would help reduce the risk of infections for nursing staff and conversely, help protect our society from infectious outbreaks. IsoCare, is the product of realisation that we could reduce the chances of infections to nurses if we were to find a way to automate at least some of the nurse's functions and responsibilities. The two main functions which are performed by our software and which will enable us to achieve our goal are, Remote Patient Monitoring and Remote Operation of Medical Devices.

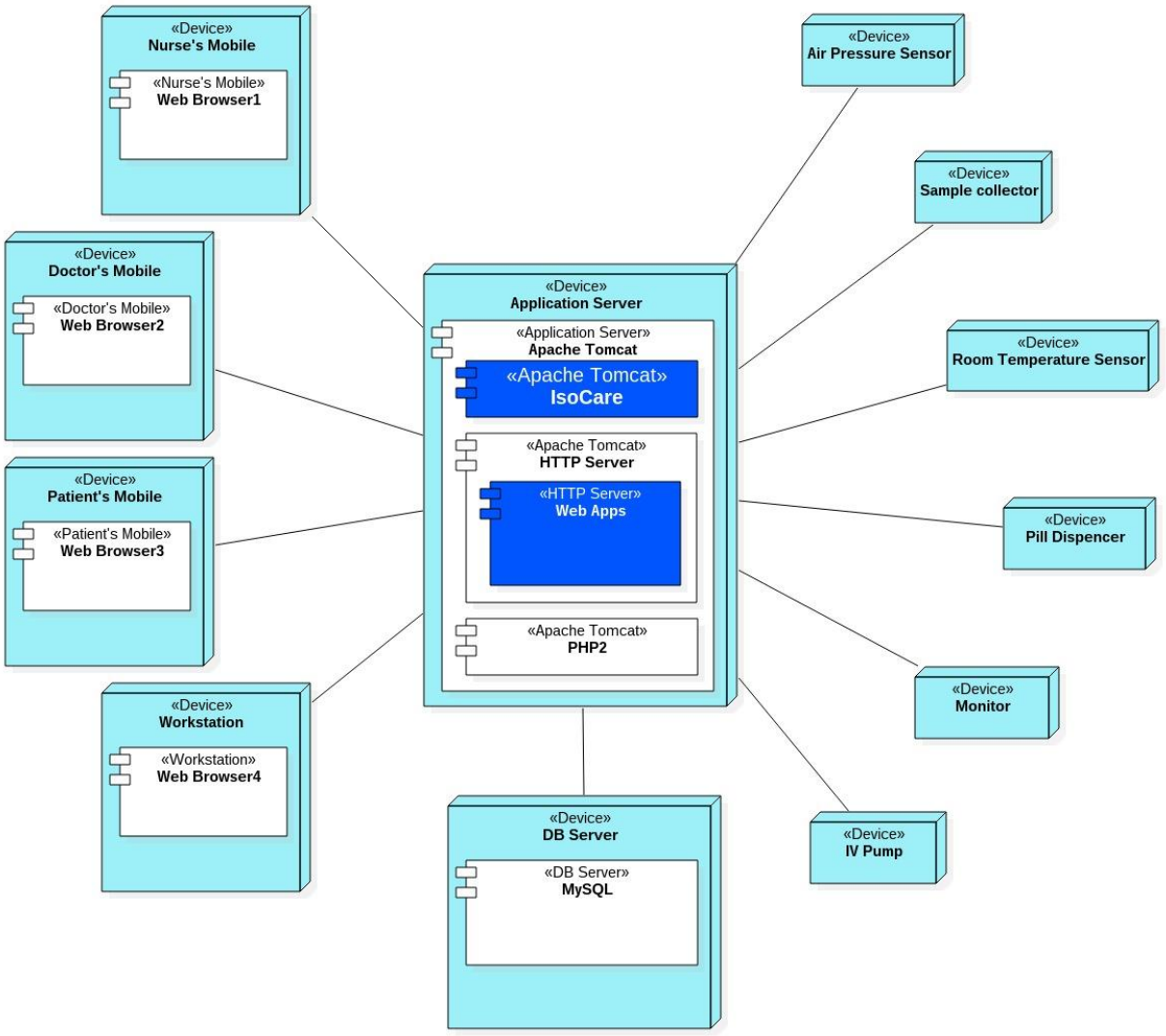
Project Overview

One of the main requirements for our software was the ease of deployment and use. For this reason, our product was designed using the most available technologies and hardware that is widely available and not too costly. At a present time, there is a wide choice of medical devices, produced by numerous manufacturers. Many of these devices have wireless connectivity built into them but more often than not, it's potential is not realized fully. The devices are being used mainly for monitoring a patient. Our system, expands the range of wireless capabilities by including support for additional medical devices such as, remotely operated IV pump, Pill dispenser, Air pressure regulator and Urine sample collector. Naturally, a p patient monitor is a part of our system too. Since the main function of our software is to manage communication between various medical devices and user's mobile devices, our choice had naturally fallen onto the MVC Architectural design style for our product. The role of a View is played by all of the mobile Doctors, Nurse's , Patients and Guests devices. The Model is implemented in our software, in the form of a collection of multiple runnable Java classes which can be invoked concurrently, therefore making the whole system fast, reliable, dependable and scalable. (Direct your attention to the following diagram for a "bird eye view" of our system)



The above diagram depicts a simplified view of one of many possible hardware setups On which our software can be deployed. Blue boxes on the left and right signify two of many possible hospital departments, integrated into our system. The white ovals represent two of many possible rooms in each of the departments. In each of the rooms, are individual patient's medical devices. In the center of the diagram is a central hospital database and in front of it is the Application and HTTP server. All devices are connected wirelessly and are grouped and separated into several Virtual Networks. This is done for the purpose of security and ease of management of communications. The HTTP Server runs Java Web Services, which in our design plays the role of Controller in the MVC.

JWC is configured so as to direct messages from different Virtual Networks (directed to HTTP Server by a Network Switch) to our software which resides alongside the JWS on the same Apache Tomcat Server, and in the opposite direction. The next diagram shows a symbolic view of a software to hardware mapping of our system.



The two components in blue show the the bulk of or software to be residing on a single machine, where one part is a collection of runnable Java classes (labeled IsoCare) and the other is a collection of Web Apps tailored to be used by all of our users and granted with different level of access rights. The users will be accessing these web apps from a web browsers on their mobile devices and through them will be able to perform different actions. The actions being, receiving and delivering alerts and regular updates from and to patient's medical devices and medical personnel devices. Additionally, all of communicatiaon are logged into central hospital database.

Requirements

Some of the requirements, such as speed, reliability, robustness, scalability, ease of deployment and implementation have been addressed in the project overview. Other requirements for our system that have not been mentioned yet are, Look and Feel, Operational, Cultural and Political, and Legal requirements.

Keeping in mind that our software's intent is to help the current situation, it should do so in a simplest way possible, so as to not make the lives of our users more complicated. For that reason, our software should present an appealing look and feel of the user interface, which should provide a simple and clear view of data to our user.

Our software should conform to the industry standards established for protection of sensitive data, so as to provide our users with security and safety of their information.

Our software should include support for other languages and provide help documentation on how it can be used.

Testing Plans

Due to the fact that our software will be used in the situation where people's lives will depend on its functionality, the testing will have to be done in a simulated environment. JTest technology seems to be a natural choice to go along with Java which we chose for our implementation. Other technologies, such as C++ can be used instead with just as much success. A software testing suite will be required to run a series of automated tests to put our software through stress testing in order to measure and hone it's speed and reliability performance.

Issues

Some of the issues that may arise in the course of implementation of our design is most likely to happen when interfacing medical hardware with our software library. Depending on a manufacturer of the device, the protocol used by a device vendor may slightly differ from other devices protocol's. It will be necessary then to modify our code to tailor it to the specific needs of such devices.

Conclusion

Implementation of our system design might prove to be a daunting task if not approached correctly. An extensive research into available medical devices has to be done before the choices are made for buying hardware, so as to eliminate a s much as possible the pains of having to develop additional communication protocols and interfacing software.

Another consideration for those who will be interested in our product design, is to try to use Amazon Web Services in place of a local HTTP/Application Server. This would help save money which would otherwise be spent on buying and maintaining hardware.

Sources

Stone, P. W., Clarke, S., Cimiotti, J., & Correa-de-Araujo, R. (2004). Nurses' Working Conditions: Implications for Infectious Disease. *Emerging Infectious Diseases*, *10*(11), 1984-1989. https://dx.doi.org/10.3201/eid1011.040253.

Auto-Astro: Safer Space Exploration

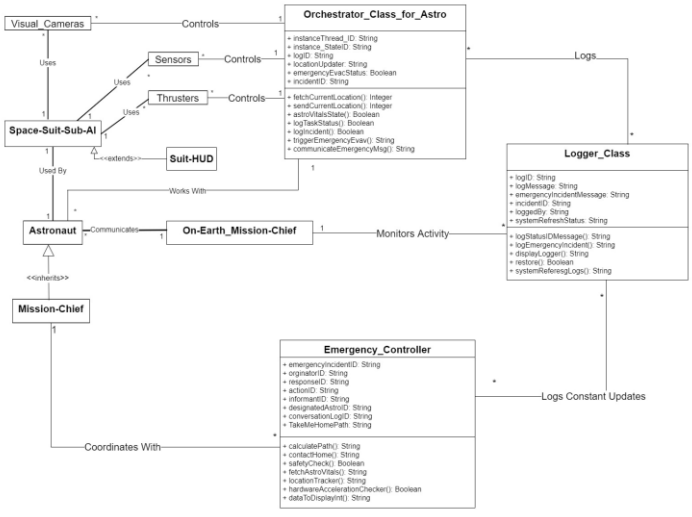
Project Summary II

Group 7: Jeet Patel, Shubadra Govindan, Hassan Pasha, Anas Ahmed

Auto-Astro is an intelligent software system that works on making an astronaut's journey into space safer. This software makes navigation and survival in space easier. Auto-Astro uses sensors and a HUD to do just that for the efficient astronaut. Primary use cases of the product would be to monitor for anomalies in the astronaut by keeping a check on the vitals. Further, in case of an unforeseen explosion of equipment within the radius of the astronaut's contact, Auto-Astro steps in, steers the astro to safety. As this is a software that would be integrated with the current systems guiding the space missions, the transition to use this system would be seamless.

The use cases cover the primary focus of the Auto-Astro system - to provide safety to the astronaut during their exploratory mission. A few of the requirements include:

Functional Requirements	- The sensors on the space suit will be activated at the instant the astronaut leaves the safe area - The system shall display the immediate dangers to the user on the HUD. - The system must be able to capture the exact coordinates of the space suits during an exploratory mission - The system shall not allow manual entering of any values other than those gathered during a space walk
Non - Functional Requirements	- The system shall have a back-up temporary system in case of failures of the main system. - Data collected during the course of the mission is constantly backed-up in a cloud based storage. Hence if the system does fail, the data will not be affected / lost.



The diagram to the left presents the overview of the system.

One issue we're facing is dealing with the weight of the thrusters, since not all agencies will

have the same type of pre-existing suits. Because of this, we are not able to generalize certain calculations regarding thruster force and such, since depending on the weight and installation place of the thruster, different configurations will apply to different suits.

Some of the reusable components we have in the suit are, all of the code essentially, which can never die down. The HUD display and the AVS system are reusable as well, we have built this suit with many reusable parts so that our future cost of materials can come down.

A potential problem, at the very start, would be difficulty for astronauts, and even technicians, to get a hold of this new and different product. At the start, the space agencies may even be hesitant to try out our product, but eventually they will because of the latest technologies we would have used, and the safety measures that are there for this product.

Our suit will mainly use the IPA to get the suit up and ready. So the classes that are required for the interface are purposely made simple. The classes are listed below, Username(),Location(),Health() and Helpline(). The location class can be customized for every user. The location class hold the current location plus updating the location. The health class monitors the health of the user. The helpline class is list of all the designated helpline for the user.

We have decided that the user might need to customize the suit for their needs. The packages are as follows, gender, font and language. The gender package is designed to give the user to switch between a male or female IPA voice if wanted. The language package lets the user change between language while communicating with other foreign users. The font package is designed to be changed for better read in the darkness of space.

Project VARGAS: Final Design Summary

Group 8: Joshua Castor, Joseph Borowicz, Bartosz Kupiec, Isabel Lindmae

Project Overview

Project VARGAS (Visual Analytics, Reconnaissance, and Guidance System) is a software and hardware system in support of firemen in hazardous environments. The system is meant to provide advanced tactical/logistical support, situational analysis, and logging of events for future analysis. It aims to make the job of a firefighter safer and more efficient by providing a system that makes firefighters more aware of potential risks and reduces the time they are exposed to those risks.

System Design

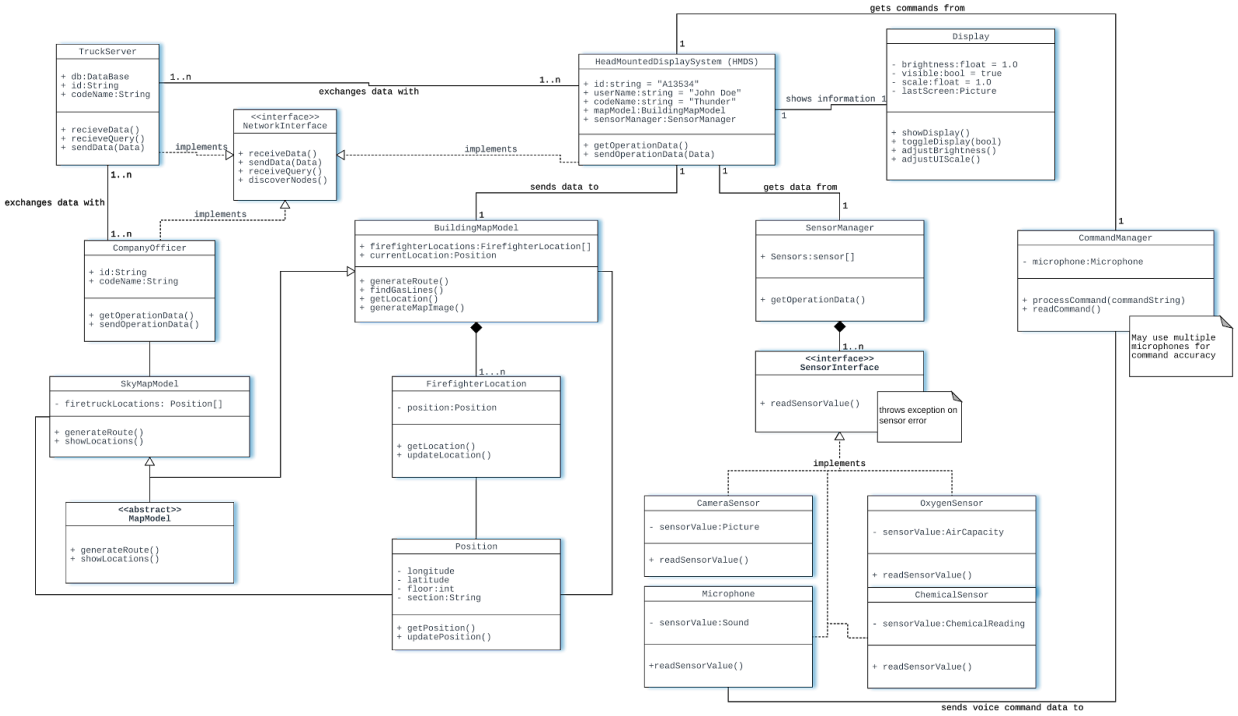
The system will be secure, stable, accurate, not compromising on speed, and be easy to use by the firefighters (and their company officers) during deployment. The most important design quality will be accuracy that does not compromise speed. As firefighters work in life or death situations, the equipment needs to be accurate and display any important messages/alerts quickly to reduce the amount of time firefighters spend in hazardous conditions.

Current Software Architecture

The system will be using currently existing systems for networking, gps location, and available database information (which comes from the current city that the system is deployed). The databases in question, which we would to access for each city, may contain building plans, hydrant locations, and utility lines/mains which will aid in the rendering of the map model overlay for each operation.

Proposed Software Architecture

The system will use it's own dedicated server which will travel within the firefighting trucks. Both versions of the system will have different methods on how to interact between the server and other systems. There is a primary focus on the interaction of the multiple systems (along with their subsystems), to ensure security and reliability of the overall system.



Subsystem Services

The subsystems, as shown in the diagram above,will only have access that is required for that subsystem to function properly. The universal network interface will provide seamless mesh networking capability handling redundant sensor object information and forwarding as necessary.

User Interface

We will ensure that both, the head mounted display (HMD), and the Company Officer System (COS) will be easy to use and have quick access to its features.

The HMD will display information in an non-invasive way, and format the information to an easily understandable format. That information includes sensor information, location/tracking, and alerts/notifications. The information presented includes oxygen levels, camera, microphone, chemical sensors, structural integrity analytics, routing, and more. For locations or tracking, the UI will show floor plans, gas lines, and fellow firefighters. Alerts and notifications shown by the UI can be commands issued by the company officers or alerts about equipment. There is focus on using voice commands to operate, with a keyword that triggers the ability to communicate directly over the microphone to minimize complications between voice controls and radio communication between firefighters.

The COS will use a graphical user interface, which will be accessed via a tablet or laptop, and gives the officer an overview of the operation, ability to issue commands to multiple firefighters, look over the current operation statistics, browse databases, review building plans, and communicate with firefighters.

Object Design

Separating VARGAS into three major types of components (Truck Server, Head Mounted Display System, and Company Officer System) increases overhead when sending/receiving data, but it allows for the creation of a dynamic mesh network to maximize the uptime of connections between nodes; this allows data exchange to be more robust against network issues due to the nature of our mesh network.

Possible Problems

Our proposed system will require: additional equipment be installed on the current firetrucks, cloud database hosting, and ensuring IT infrastructure is solid.

The hardware required to satisfy the requirements is currently partially unavailable. The technology currently available does not provide the require field of view for the AR display, the processing power to analyze the sensor input, the modular reusable sensor units, all while having have the capabilities to survive an environment as hazardous as a fire.

Cost/Planning of the Development Phases

The system will cost about $14 million for developers, testing will be about $1 million, and the hardware is an estimated to be about $2 million, rounding our cost to about ~$17 million for the project overall. The project will have a planning phase, software building phase, hardware incorporation phase, and then at least 3 different releases (alpha/beta/final). After release there will be support/updates to the system for 10 years.

Retrospective

- Spending more time researching requirements may have resulted with a better system.
- Having multiple deadlines kept the project on track and always made it move forward.
- Scheduling weekly meetings, with clearly outlined the work for each of the members.

The Inferno Suit

Group 9 - Aditya Sinha, Jignesh Patel, Bushra Baqui, Nishant Patel

Project Description:

The inferno suit concept aims to change the way firefighting suits are currently used. The current suits are mainly focused on preventing the firefighter from getting burns. This serves a very limited purpose and can have a lot of additional features that can be added. The inferno suit aims to do this by revolutionizing the firefighting suit by adding three main components. To reiterate, these three components are the external frame of the suit which is the exoskeleton. There is also an inner suit that will help the firefighter keep cool by using nitrogen and pumping them through the tubes of the inner suit. The last component that is crucial for this project and contains most of the software is the helmet that will possess the heads-up display that will display valuable information for the firefighter in real time. This information ranges from oxygen levels, battery levels, location of the firefighters, and any temperature changes. To make the suit more efficient, several sensors are going to be placed on the external frame of the suit and it is going to be receiving data from the environment. This useful as it provides critical information and also helps the firefighter gain more information about the emergency that they are working with. The data collected from this changing environment is in real time and it is processed and sent to the HUD for the firefighter to display. The Inferno suit also comes with a cooling vest which will help firefighters using water nitrogen. The exoskeleton also assists in the firefighter in lifting heavy objects and provides them with a boost in performance.

System Design:

In terms of system design, thee were several components that had to be taken into consideration and the tradeoffs that had to be made required justification. Some of the tradeoffs are that the helmet is designed in a way so that it can fit most people. In trying to make the helmet accommodate a wider amount of people, we weren't able to keep the visor at a fixed position. To better customize the display for each firefighter, we decided to include a larger display so that we can better adjust the information that is going to be displayed for the firefighter. Another design element that we decided to include is the use of OLED technology for the visor's display. The helmet will be given priority in terms of battery usage so that the CPU and visor is able to function efficiently and has minimal lag.

We are going to have three main classes which is user, inferno suit and the fire unit. These three classes will be for each new user which will have one inferno suit, and which will communicate to one fire unit. The inferno suit will contain three additional sub classes that is the AR helmet, the exoskeleton suit and the inner suit. The user class will contain the getter and setter functions that will retrieve information from the inner suit. During startup or an initial use, the first information that will be gathered is whether the HUD is functional and whether the exoskeleton is connect, and the cooling suit is functional. In terms of the inferno suit communicating with the fire unit, we have functions for the radio communication, the server method that will communicate with the inferno suit. The AR helmet will contain functions to check whether the display is active, and functioning and it will perform the same checks for the radio and the camera that is available on board. The exoskeleton suit class will perform a similar

check for the sensors that are on board such as the temperature sensor. In addition, it will also check the actuators and battery by calling on the respective functions. The inner suit will call the body temperature function and the heart rate function that will monitor the firefighter at all times and report to the fire unit and other fire fighters in case there is an abnormality.

The AR Helmet will also contain additional subclasses for the other components that are going to be included in the helmet. This includes the display, the radio, the camera and the sensors. The display will call the function isDisplay to ensure that the display is connected and active and it will retrieve data such as blueprints or additional data from the sensors and print them on to the visor. The sensors subclass calls on the retrieve data function and this is the function that will retrieve the data directly from the sensors and also perform any calculations that are necessary before sending this information back to the user. The radio and camera are also going to be storing their information such as any recordings onto the internal storage that is going to be built into the helmet itself. We decided to choose these design elements because not only are they efficient, they improve functionality. This is also why the exoskeleton design as appealing when first looked at but it does add functionality for the firefighter and will hopefully assist them in the future.

User Interface:

We want the user interface that is going to be displayed onto the HUD to be easily accessible and easy to use for anyone that is using the suit for the first time. Firefighters that have been serving for long time should also be able to use the suit easily. One of the main components that will make the user interface easy to use is the eye tracking of the user and how the firefighter is going to select each menu and sub-options. This will not only make the experience engaging, it will also provide a fully hands-off experience. Another ease of use that the inferno suit's dash has is that it lets the user customize the menu screens according to their own preferences. This enables the user to place the individual items that is going to be displayed on the screen according to their own liking. Not only does this add a personal touch for the firefighters, it also provides as an easy of use so that each firefighter is ready to go and knows exactly where each menu and option is before getting started. Throughout the user's time within the inferno suit, we want the majority of the experience to look beautiful. The main part of this suit is the HUD and it is the one component that the firefighter will be interacting with constantly. At the same time, we don't want to block the firefighters vision during an emergency.

Object Design:

The different object design trade-offs that we had to face while designing this project are mostly the ones that are related to the exoskeleton. For the exterior of the suit, we didn't design the exoskeleton based on design but instead we focused on functionality. We allowed this trade-off as we want anybody with a reasonable operating system to be able to use the Inferno Suit. Therefore, we will be incorporating processor optimization that will be available on each inferno suit to ensure maximum performance. Doing so, also enables each suit to perform at its peak level and ensures there is minimal lag and creates a seamless experience for the user.

Guide Me - Final Project Description Summary
Group 10 – Harsha Dodda, Rohit Nambiar, Harsh Patel, Krunal Patel

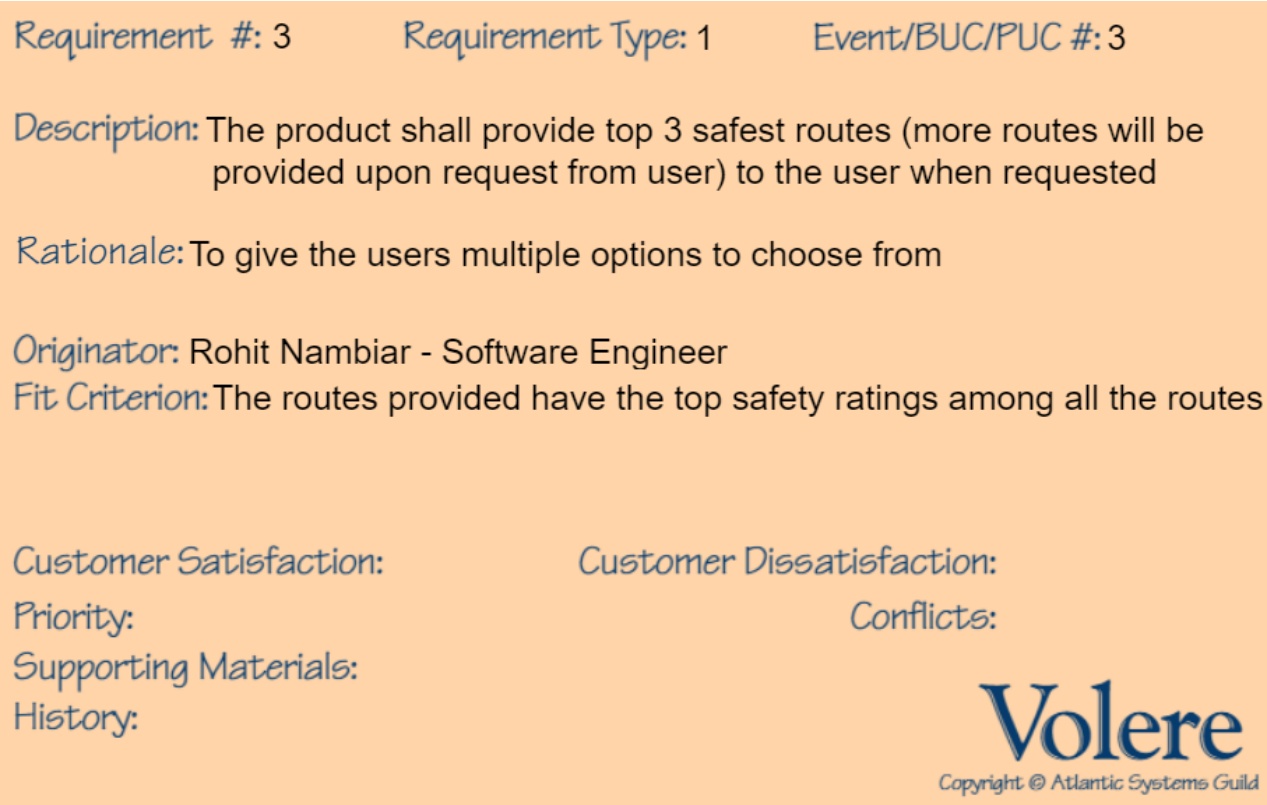
Project Overview

This project is an application that will show the user where the hotbeds for crime in the city are and which routes the user can take for the safest route to their destination. The application can also show the general hotbeds of crime within an area so the user can be informed of the safest areas around them as well as which areas to avoid when roaming about in a city that may be unfamiliar. This software would project a heat map that shows areas of high crime within your area as well as between you and your destination. Then the application would show you the safest route to take based on how much crime occurs in areas between you and the location you are traveling to. This application would also show other information about crimes and safety.

Sample Use Case

Use case name	Start Navigation(5)
Participating actors	• Initiated by **User** • Communicates with **System**
Flow of events	1. **User** can activate this case by either opening a route from saved results(case 4) or search routes(case 3).
Entry condition	The **User** opens a route option.
Exit condition	• **User** presses start navigation button and exits this case.**OR** • **User** can just exit the application
Quality requirements	• Navigation must be precise of 300 ft of the User.

Functional Requirements



Subsystem Services

- One subsystem will get and display Google maps routing data
- One subsystem will get data from the City of Chicago data portal and store the data in a local database in real time
- Another subsystem will take the information in the local database and feed the data into the Google charts and Google heat maps APIs and then display the information in a series of easy to understand graphs, maps, and charts
- The last subsystem will allow the user to create, login to, and manage their profile and settings

Features to test

- Acquire the routing information from the starting point to the destination
- Dynamic rerouting when the user veers off course from the given route
- Voice guidance as the user travels through the route
- Getting data from the City of Chicago data portal

Food Allergen Detector - Final Summary

Group 11: Sohum Mehrotra, Danny Chen, Yoonha Kang, Jun Lee

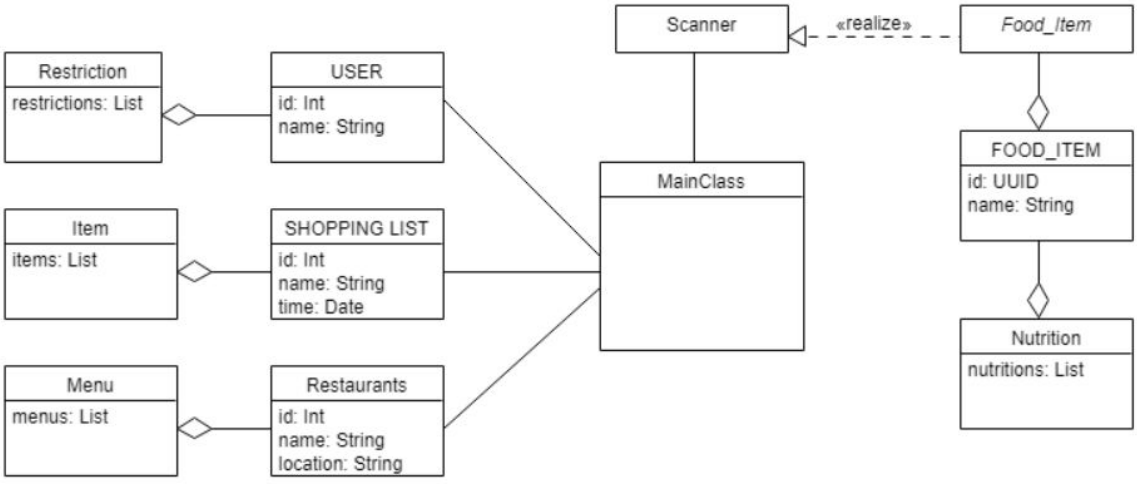
Project Description

The product that is being produced is a mobile application called the Food Allergen Detector. There are several key features to this application. First, the user will be able to input all the dietary restrictions they have. If the user scans a food item and it violates a dietary restriction, then the user will be alerted of this violation. Second, the app will also serve as an inventory projector. It will analyze all the food items that are scanned and purchased and determine trends for how frequently the user buys a certain item. The user will then be alerted if they are due to purchase that item again. Third, the app will also integrate a related food recommender that will provide suggestions of related food items under each scanned item (this is especially important when the item is scanned is restricted to the user). Fourth, the user can go into restaurants and can see all items that do not violate their restrictions. Fifth, the user can physically scan the food item and determine whether that food item is in violation of the user's restrictions without the use of a barcode.

Requirements

Data

The data(food item) should be organized to eliminate as much repetition as possible while maintaining functionality.



Security:

- Only database admin and developers can update the database.
- Only registered users can access their profile and set restrictions.
- Only system admin and user can remove a registered account.
- Any user can use the scan feature.

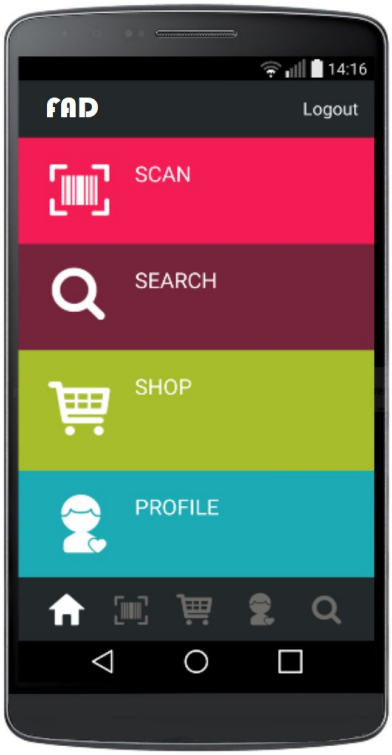
Productization and Release:

- The product shall be distributed as a free application in mobile platform's respective application store
- The product shall be categorized in these stores as a lifestyle/health product
- The product shall be integrated with facebook for login
- The initial release of the product will be subject to the discretion of the release team.
- Updates will be pushed through on an as need basis, as more functionality is created or if the mobile platform demands more native compatibility
- A team of staff will be required to monitor complaints and reports generated by users of the app and make appropriate hotfixes

Testing

Development should be test-driven, each feature/requirement should pass developed test cases before moving on to the next step to ensure quality.Once product has been launched, surveys and reviews can be used to collect user feedback for future updates and fixes. Existing features will be regularly tested and maintained with regression tests through the lifetime of the product. Testing should take place during the complete development of the product.Testing will be done extensively before the initial release of the product, during the duration of the product's release, and when users report bugs.

Design



Food Allergen Detector (FAD) is designed for a mobile device. Users will perform actions such as setting the allergy restrictions, scanning barcodes, and the server will respond based on requested actions. The approach to our main design goal is to offer users to get a correct warning for food items with their particular allergens(s). The scanning feature should correctly identify allergens in the scanned food item as close to 100% of the time as possible.

The product design should be simple to get allergic information of a food product. Any user must be able to get allergy information with single scanning of barcode. And the app will have user profile system in order to sync with a server and users don't need to input the restriction everytime.

The software development will be divided into several layers following model-view-controller architectural pattern. Models will be responsible for managing the business logic and handling network and database API. The view will be modified from the fragments with hosting activities. Controller gets notified of the user's action and updates the model as needed.

Envisor: Project Summary

Group 12: Suleiman Suleiman, Daniel Pulley, Dana Dolat, Ahmad Atra

Overview

The Envisor is a hardware solution that utilizes both a visor and gloves that contain multiple sensors. The overall purpose of the project is to provide environmentalists tools that can assist them with their day to day work, with the overall goal of the project being to quickly identify issues in an ecosystem so proper action can be taken for environmental remediation.

Functional Requirements

The functional requirements for the Envisor are relatively simple. They include, but are not limited to: the system must provide a means for collecting sample data, the system must be able to send/receive data over a TCP/IP network connection, the system must be able to decontaminate itself, and much more. We found that most of the functional requirements needed pertained to the actual hardware itself, rather than the software end of things. However, through software and the use of good quality components, many of the requirements listed in our document (including the ones stated above) can be satisfied and tested to a great extent.

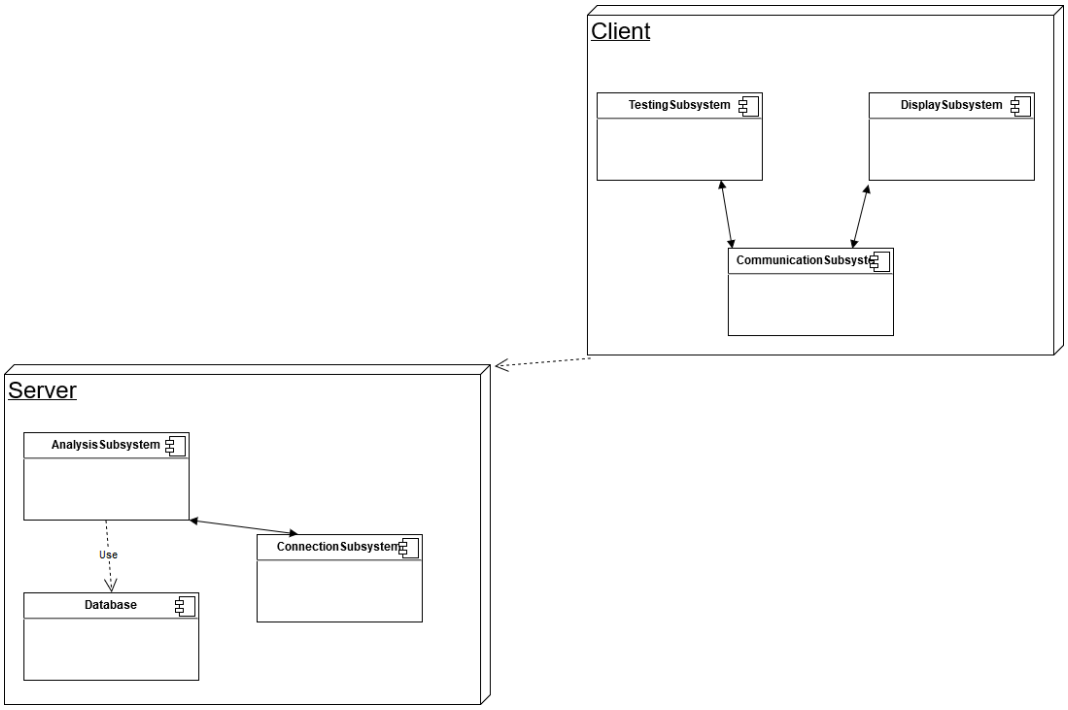
Non-functional Requirements

The non-functional requirements for this project were relatively complex. In fact, I'd go as far and say that these requirements. Some of them include: ensuring results produced by the product are accurate, the product must have a user friendly interface with on-screen guidance, materials used for the product must meet the EN-374 standard. Many of these non-functional requirements that we've written ensured to the

stakeholders that this product we are developing for them will ensure quality of life, and won't require updated releases in the near future.

Design

Simply put, the design of our system can be simplified by viewing this high-level diagram of the proposed software architecture:



Envisor utilizes a client-server software architecture, where the client is the physical components of the system (the gloves and visor) that have subsystems responsible for: the testing of collected samples, communication between server/client, displaying view objects, analyzing collected sample data in relation to the database, etc etc. All the subsystems that we've developed for this project have been aimed to use well known design patterns to ensure consistency throughout the software level of the system, as well as make it easier for developers to generate unit tests (much easier to make a unit test when the class organization is predictable/understandable).

VISIONARY

Group 13 - Ankita Tank, Jacqueline Tapia, Kunal Shah, Pierce Zajac

Final Project Summary - Fall 2017

Project Overview

Visionary is a software and hardware product which aims to enhance the experience and safety of Scuba Diving through an integration with pre-existing Dive Computer and addition of a HUD.

Project Purpose

Scuba diving technology is used by both the casual recreationalist and professionals in many industries. While rewarding, scuba diving is laden with risk. To some this might be part of the thrill, but divers undergo intense training and practice to ensure their safety underwater. In the same way industries like automotive and aeronautic have turned to technology to lessen the risk of the end user, there is the same potential in scuba diving that has yet to be fully tapped.

The Scope

The product consists of two main components which are the HUD and the dive computer. The dive computer gets information from the outside world through a variety of sensors, sonar and GPS. A scuba suit encapsulates this technology to provide vital situational and safety information to the Designated Diver through the HUD. The work consists of using the necessary gear (mask, fins, regulators, tanks, dive computers, and more) as well as, enhancing or adding materials such as: HUD helmet, software to dive computers, and integrating dive computers with the HUD helmet. It ranges from how to use sensors to collect gas level data, sonar mapping for night vision, integrate GPS chip into the product, and integrate the existing dive computers with our software.

Product Use Cases

Our use cases consist of two systems, the External Facing System(EFS) and Internal Facing System(IFS). EFS use cases consist of: Enabling/Disabling Night Vision, Enabling/Disabling Sonar Mapping, Enabling/Disabling HUD, Enabling/Disabling GPS Tracking, Displaying Vitals, Low Gas Warnings, and a Depth Warning. Subsequently, the IFS sue cases consist of: Reading Nitrogen, Oxygen, Carbon Dioxide, Nitrogen, and Pressure from each respective sensor, Calculating Decompression Sickness, and Oxygen Remaining.

Requirements

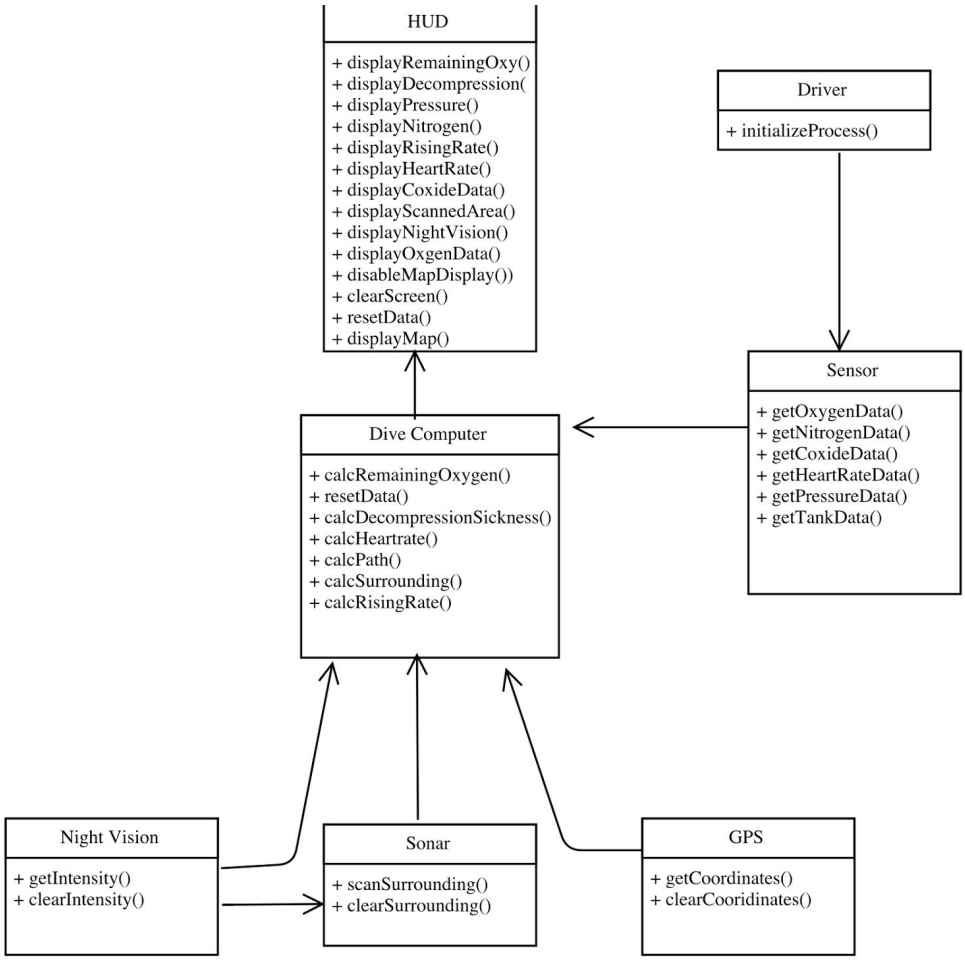
Functional requirements are divided into EFS and IFS. The dive computer functionality includes, Enable/Disable the HUD / Night Vision / Sonar Mapping / GPS. The dive computer will display vital calculation and warnings. The data objects involved are: HUD, Driver, Night Vision, Sonar Mapping, Dive Computer, Sensors, Risks, and Vitals. Sensor Readings- Real time || HUD update- every 2 seconds based on diver's vitals and location.The product shall not fail unless drained by battery. Maintenance after every 200 dives or once every year.The HUD glass should be changed once every 2 years to provide clear display. We expect the HUD to be used under water, in areas of low visibility, at depth levels with pressure of up to 70 psi, and shall be used by professional and technical divers.The product must not persist vitals data for longer than necessary to compute calculations in order to abide with HIPAA. Location data shall not persist in order to protect diver's privacy as per compliance with the Fourth Amendment as well as the GPS Privacy Act.

Testing

We are testing the software calls which obtain data from each sensor and testing the result of the each procedure call. We will also test the HUD in order cross check the display values with the calculated values in order to test functionality of the HUD. Lastly, the GPS, Night Vision, and Sonar Mapping must also be tested. In order to test the HUD Helmet, we already realized what to test. Next, we will create

initial test cases for each testable item, and then proceed to run the tests. Based upon the results of our tests, we will refactor our code, test cases, as well as add new test cases for newly found bugs. Stress tests will be done under various pressures to ensure usability remains constant up to the max rated depth. Users testing will also be done to ensure usability for divers of all shapes, sizes and ages.

Proposed Software Architecture



Boundary conditions

Startup- Data not initially collected nor displayed. Additional functionality must be turned on manually by the diver. Shut down- data must be flushed out in order to preserve security and privacy of the diver. Sensors failure- Consider all invalid data and no longer perform calculations in concern for the safety of the diver. A notification would inform the diver of the sensor error, its consequences, and recommend them to resurface. Dive Computer failure- the HUD will warn diver of critical error and go transparent.

Object Design trade-offs

Accuracy over performance in concern for the safety of the diver since there are critical calculations being performed. Performance over security since any data which would require security is not being persisted due to compliance with regulatory laws.

Noah's Ark: Final Project Summary

Group 14: Wynn Drahorad, Shruthi Kumar, Gary Mei, James Chou

I. Project Description

Our application will be used in natural disasters and will help ease the relief and rescue during these events. We hope one of our main stakeholders, FEMA who already assists in these events, will help fund this application that will help refugees find rescue shelters suitable for their needs. Our application must be usable on mobile platforms as well as keep a local cache for offline use due to connectivity issues. With a budget of $600,000, we would have a database of accurately inventoried supplies and be able to see locations of high demands. Our timeframe of one year, with 3 months for a viable product and 6 months for improvements, will allow us to have a program in beta and be able to create our inventory database from scratch. This timeframe will also allow us to test 3000 concurrent users without decreasing system functionality.

II. Requirements

Our system functional requirements include important features such as the ability for a refugee to receive login credentials through our application. The program must be able to keep all refugee information private and secure. Among the main features our our system, the application will provide a list of shelters, and can choose shelters that fit the requirements/filters that a user requires such as location, supplies, and capacity.

Performance requirements include the application loading within 3 seconds whether offline or online. Also, a user should be able to see a rescue shelter within 5 seconds of opening the application. While offline, the application should not crash more than 5% of the time. Finally, data should be updated in real-time when available.

Accuracy requirements include rescue shelters' locations being accurate to within 5 feet, and the refugee's location being accurate to within 10 feet. Basic supply quantities should be accurate 95% of the time, with medical supplies being accurate 99% of the time (due to their sensitive nature).

III. Design

The software application is divided into three distinct pieces. A server coordinates the information of shelters and refugees to their respective parties. The client apps are split depending on the type of user. A refugee sees a client that allows them to search and find shelters nearby that fulfill their needs best, while a shelter sees a client that allows them to manage inventory and refugees. A refugee selects a shelter, and can notify the shelter that they are on their way along with their needs. A shelter can check-in refugees and assign medical supplies to specific persons.

The application's main purpose, delivering actionable information to both the refugee and the shelter, would be hampered by inaccurate or outdated data. For that reason, there is emphasis on the design to utilize caching. The client applications focus heavily on caching data as often as possible tempered by reasonable battery usage, while using as little data as possible. The server reflects the latest updates pushed to it from the client applications, making stale existing

information. To avoid excessive data usage and processing on the client side, the server calculates delta updates whenever possible and sends those to the client application.

IV. Test Plans

Features that must be tested include:

- Location tracking of user
- Application returns listings of relief shelters
- Application returns listings depending on additional filters from user
- Application provides directions to the relief shelter chosen
- Application functions with intermittent data signal

The product shall be tested on all mobile operating systems (Android, iOS), all desktop operating systems (Windows 7, Linux, macOS), all operating-system web browsers (Safari, Firefox, Edge), and on a network with 1 Mbps/500kbps of download/upload bandwidth.

Development and implementation of features should also be completed in the order given:

V. Project Issues

Creating a way to authenticate the shelters to avoid fake shelters being created. Real world vetting, registration with the proper authorities, and working with existing organizations is the solution. Testing our system under load with 3000 users connected needs to be further tested and potentially readjusted by minimizing the amount of data used per session.

FEMA has an application for locating nearby shelters. The application comes from a trusted source but the results do not indicate how full the shelter is, the current status of the shelter, navigation, and medical compatibility.

The estimated cost is $600,000. This would include software engineers, data centers, auditing the inventory.

Final Summary - Sonic Phision

Group 15 - Jim Mei, Michael Mei, David Fulmer, Ahmad Zaghloul

Project Description

Sonic Phision is a hardware and software system used by shipwreck, cave, and worker divers which works as both a dive computer display and an underwater mapping application. It can communicate with a map database to store and reuse maps of various dive sites. This product will allow divers to safely traverse the ocean and keep track of their locations. Along with the features of the underwater mapping capabilities, the visor will be able to monitor the pressure, depth, water quality, oxygen levels, heart rate, and battery levels. This product will be able to provide the best user experience diving underwater.

Project Requirements

Some main aspects of our visor is its capabilities in monitoring several aspects necessary for the safety of the user. Our visor will monitor the oxygen level in the tank, the water quality surrounding the diver, the pressure currently being exerted on the diver, and the heart rate of the user.

For the oxygen level, the visor will have monitors and measuring the oxygen tanks on the user. The data gathered from this measure will be analyzed and converted into information for the diver to see such as time remaining left in the oxygen tank and the actual percentage of oxygen inside the tank.

For the water quality sensor, there will be multiple detectors on the visor and user to take in and sample the water quality the diver is currently in. After the quality of the water has been analyzed, a blip or a spot on the heat map will be created or updated to reflect the current condition of the water the user is currently diving overall. The heat map will be displayed on the corner of the HUD so that it will not block the vision of the diver.

There will be a pressure sensors on the visor to detect and measure the current water pressure being exerted on the diver. With this information, the visor will calculate and determine the current depth of the diver and display the information on the HUD.

The heart rate monitor will be included in the visor to monitor the heart rate of the diver but its main purpose is to be used as an emergency signal should the user's heart rate stop. If the diver's heart rate stop, the visor will send a signal from its location so that other divers can come and retrieve the diver and potentially save the diver's life.

Our main selling point of our visor is the sonar wire mapping feature. This is an entire wire map of the area the diver has traveled through. It is created through the use of sonar and map will create the surfaces surround the diver so that the diver can travel through dark areas without the fear of getting lost or running into any walls. On the map will also be a trail of the diver's travel so that if any reason to occur, the diver and back track their course. This feature is one of our main selling points and it is essential that this feature is optimized and is as polished as possible.

As with all of our features, it is impossible to include if they aren't accurate and precise. The data gathered from the sensors is essential in keeping our users safe. Just as accuracy and precision is crucial, the speed and latency of our visor is just as important. The response and speed of the visor is necessary so that the diver can get the information as quick as possible so that they can make the most educated decision. So it is crucial that our data be not just accurate but also precise in providing the most accurate data to diver for their safety as this information

and the information will be sent as fast as possible so that it can be used to notify the diver whether or not they should continue their exploration.

After all these requirements, the final requirement is that the visor should be durable so that it can withstand the every possible condition underwater. There is no point in a fancy and advanced visor if it breaks. A broken visor will only endanger the diver, so having the visor be as durable as possible is important. It should be able to withstand pressure at 70m below sea level, withstand various water currents, erosion, impacts up to 2,000 lb of force.

Test Plans

To begin with our test plans, we must test the hardware of our product. Our visor requires various detectors and measuring tools so having these work properly must be done first and foremost. Each of the measuring functions: oxygen, pressure, heart rate, will each have around 2 weeks each of testing to make sure that these features will meet our requirements and will be satisfactory to our team. We will also need to test the sonar and the wire mapping capabilities. We will need about 4 weeks to test these functions so that the wire map created by the sonar will be satisfactory. We will also be testing the software needed to have all these features working correctly together so another 3 weeks to test and debug. We will also test the durability and waterproofness of our visor. We will have to conduct these tests over a 9 week span, 4 weeks for the waterproofness and 5 weeks for durability. Next we will need to test the battery, water quality monitoring, and bluetooth capabilities of our product. These tests will take 1 week each. Overall, our product should have at least 22 weeks of testing.

Design

The way our software works is that all the HUD class should contain all the other classes. What this means is that every time the classes that processes any information detects a new value, the new information will be sent to the HUD class so that the HUD class can make the necessary changes to the display and update the information so that the information currently on the visor is the most recent info. The HUD class will then send the information it currently has to the AlertManagement and UIManagement classes so that they can check the data for any irregularities or any possible values that may be dangerous for the diver. After analyzing this data, these two classes will send a message back to the HUD class (displayManagement) and will let that class know whether a warning needs to be issued or any information on the HUD needs to be changed.

Current Environment

Since our visor uses new information and technology, there is little to no previous information. This means that it is not necessary for us to migrate information from databases. But this does mean that we would have to start collecting information from scratch.

We wish for our product to make a big impact on the market as well as other fields. Since our visor centralizes a lot of tools into one, other products such as dive computers and deep sea navigators would not be necessary but may pose as competitors. The impact we wish to have on fields that are related to diving such as diving teachers/instructors and professional divers. With our visor we wish for more people to dive more as it may make the experience in learning to dive easier and provide a more enjoyable environment. But this means that there may be less diving instructors/teachers as it will be easier to learn how to dive. And if there are more divers , there will be an increase in the amount of professional divers and will create more competition in those fields.

Project DIVE: Final Summary

Group 16: Pedro Cardenas, Arbaaz Meghani, Emiliano Velazquez, Jacob Waller

I. Overview

Project DIVE is a scuba diving apparatus that assists professional scuba divers by providing them with vital information such as oxygen level, estimated time left of oxygen, nitrogen level, and the current time. It also allows the diver to see the path they have taken in the water through augmented reality by plotting their location that is acquired through communication with the home boat using radio frequency data transfer as well as enabling the diver to use this same technology to communicate with other people.

II. Functional Requirements

Project DIVE enhances the safety of scuba divers underwater. The system is designed so that it must display to the diver vital information that helps the diver be aware and be given control of their situation at all times. The vital information that is displayed to the user is the oxygen levels, oxygen estimated time remaining, nitrogen levels, and time. These requirements allow the user to know how long they have been in the water and allow them to estimate how much longer they will be able to stay in the water. The system must also provide the user with a general warning. The general warning system is tied to various hazards that may occur in an underwater situation. A couple examples include running low on oxygen, or going to a very high depth level where the water pressure is too high for a diver to handle. Moreover, the system will warn the diver if there is a large object near them.

Additionally, the system provides an augmented reality view of the path that diver has followed since they have been in the water. In the event that a general emergency warning occurs, the diver may panic or be in a serious situation where they must leave the water. The augmented reality view of the path that is tracked accurately onto the water, using environmental data provided by the camera, the diver should be able to follow that path back up the home boat which is at the surface of the water. The location of the diver is received through data transfer between the home boat and the radio frequency module that the diver has on his diving apparatus. For example, oceanographers who may explore unknown areas underwater and could end up getting lost and running out of oxygen. The augmented reality path system allows them to safely leave the unknown areas without running out of oxygen.

III. Non-Functional Requirements

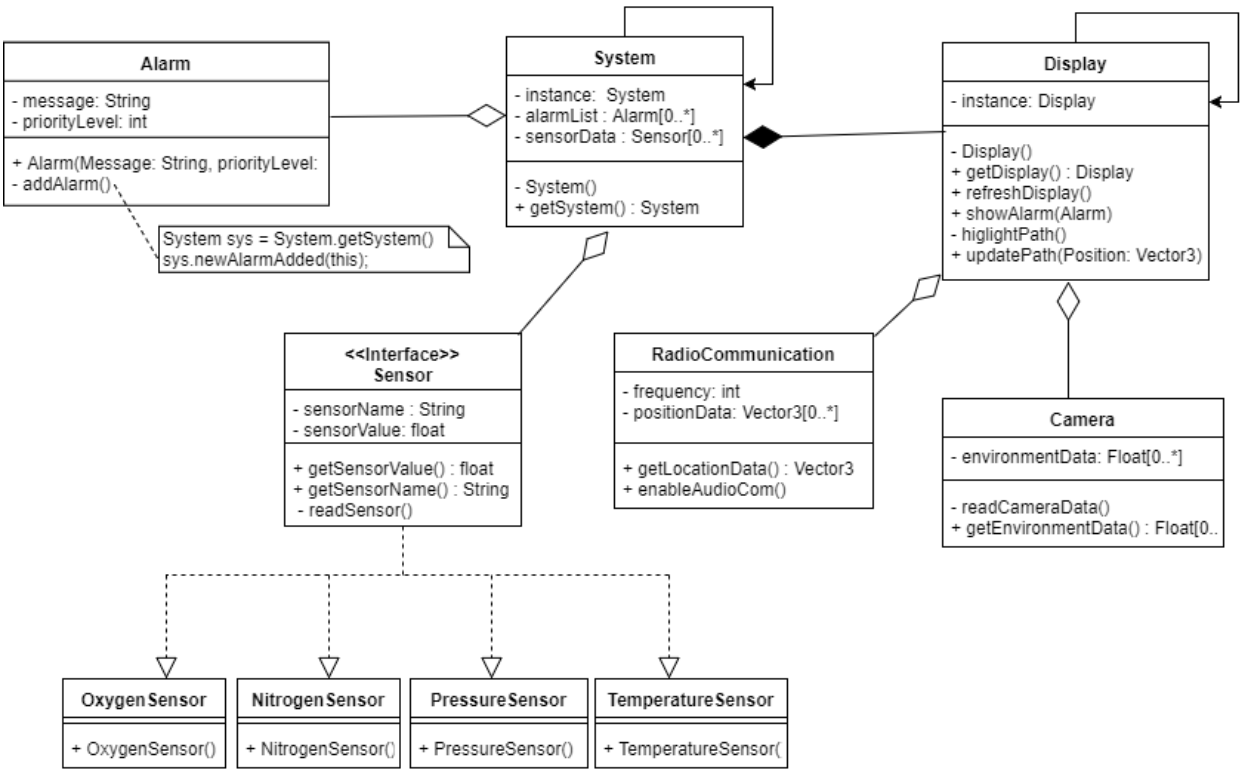
Project DIVE makes use of various sensors and provides that information to the diver through the display. The system should be dependable in the case that a sensor fails. The scuba diving apparatus should have multiple sensors of each of the sensors required. Not only does it account for failure of sensors, but it also enhances the accuracy of measurements that the sensors provide. For example, using multiple oxygen sensors would allow the system to make sure each sensor has the same exact value so that the user receives the most accurate information. When a sensor does fail, the system must be made so that the diver can replace it on their own by following a modular design. The diver would be able to take out different sensors and put in new sensors without any trouble. The sensors must be authorized sensors that can be interfaced with the system so that unauthorized sensors that try to exploit the system are not installed.

IV. Testing Plan

Each of the sensors should be individually tested in a controlled environment. For example, when testing the oxygen sensor for accuracy, we can get an oxygen tank with a known, specific level of oxygen and use the oxygen sensor to measure the level of oxygen in the tank. We can then compare these values to make sure that they are the same. We apply the same method of testing for each of the sensors that is part of the system. Any feature that interferes with a feature being tested should be disabled temporarily to test a single feature at a time in an ideal environment. The effects of interference can be handled after each feature passes their individual testing requirement. To effectively test each feature, an underwater environment will be required. Any feature that does not meet the requirements should be fixed to meet the requirements or different hardware should be used. Each sensor measurement should be 95% - 100% accurate 100% of the time. Anything below that is considered to be a failed test for the sensor.

V. System Design

The sensors used as part of the scuba diving apparatus should be as accurate as possible. Since this is a piece of equipment that will be used in a hazardous situation, the sensors should be accurate so that it can provide the most valuable and trustworthy information to the diver. When the information is made available, the display should refresh as soon as it can so that that diver can be kept up to date with the most recent information. The system should require as little human interaction from the diver as possible. The user should not have to consistently put in input. They should be able to move around freely.



Mold Detection: Project Summary

Group #17 Members: Matthew Gizzo, Deon Zoss, Mac Squibb, Casey Charlesworth

Project Description:

Our project is a system that collects and measures data about the atmosphere in rooms and buildings after a major flood or hurricane to predict the likelihood of mold being present. The system consists of two physical components, a sensor and a headpiece. The sensor will be something that can fit into the average adult's hand and will consist of a touch screen User Interface and a thin needle that can penetrate normal wall surfaces to collect and measure the atmospheric data. The headpiece consists of a visor that acts as the User Interface, a breathing apparatus that may or may not be worn, and an external flashlight. There is also a smartphone application for this product that is similar to the main product with the hand-held sensor and head piece for home users who wish to check for mold in their own buildings. The overall product will work in such a way that the user will utilize the sensor to gather data about the environment and such data will be transmitted to the headpiece for the user to see and analyze.

Requirements:

Listed below are the functional and nonfunctional requirements that we deemed most important for this summary. Note however, there are many more in the project document and all requirements in the document have the specific details for each requirement.

Functional:

- The system shall allow for multiple basic calibration settings
- The system shall insure the transmission of data from the sensor to the head piece
- The system shall communicate with external devices to access necessary information
- The system shall display a graphical selection of options for user to select
- The system shall measure aspects of the environment related to mold growth, and perform calculations on such measures

Nonfunctional:

- Learnability of the system
- Light and Compact
- Waterproof
- Quick performance
- Smooth performance while transitioning from states of internet connectivity

Testing the Product:

For how testing is done, there are different routines for the software and hardware requirements. When it comes to testing the software requirements each requirement will go

through 2 phases of testing. The first phase of testing will be done on a computer that will run all the software requirement tests and to be able to move onto phase 2 of testing the software requirement must pass this phase of testing. The second phase of testing for software requirements will be done with the sensor component and headpiece component to assure that the tests also pass when utilizing the software on the products equipment. When testing any hardware requirements there will be special test rooms dedicated to representing the wide variety of spaces that have been affected by major water damage. These test rooms will be able to change and adjust certain conditions about the environment in the room for maximum testing accuracy. These environmental conditions include things such as gas levels, humidity levels, amount of light available, physical structure, and overall hazardous intensity. All hardware tests will be conducted by the sensor component and headpiece component with a user; no computer testing phase is needed.

System Design and System Design Goals:

The type of system that we designed utilizes the classic model-view-controller schema. We designed 4 main subsystems for this project and they are: Back End, Data Collection, Database and 3rd Party Applications, and User Interface and Application. For this system the Models holding and gathering the data are the Data Collection and Database and 3rd Party Applications subsystems. The view will be represented by the User Interface and Application subsystem, while the Back End system acts as the controller. Overall we want the models to collect and store the necessary data for the system for the controller to manipulate and transfer to the view that would then format and display the data for the user to see and use as necessary.

In terms of system design goals we would like to highlight what we see as the 3 most important design goals for the system. There are a few more design goals for the system and they can be found within the project document if you wish to observe them.

System Design Goals:

1. Modifiability
    a. Modifying existing code should be very easy and fast
2. Ease of Use
    a. Classes and functionality should be fairly simple
    b. The UI and data layout should be understandable for any user
3. Robustness
    a. The system should be able to handle any runtime errors without crashing
    b. The system should be able to handle erroneous inputs form the user

The Dolphin - Project Requirements Summary

Group 18: Dieu Do, Luis Hernandez, Brent Yurek, Kandyce Burks

Diving is a dangerous activity to engage in. Divers are always faced with dangers such as water pressure and air consumption. A main function our system has is displaying diving time elapsed, dive time remaining, and dive depth. The body mounted sonar would be able to create an image of the surroundings giving divers an idea of their surroundings allowing them to be able to perceive possible hazards.

Functional Requirements:

- The product shall change the rotational sensor being used based on the diver's position. The system will use the head rotational sonar when the diver is swimming vertically in the water, and the back rotational sonar when the diver is swimming horizontally.

Speed and Latency Requirements:

- System retrieves data that the sonar picked every half a second for real time sonar map.
- System alerts diver 20 minutes before oxygen tank depletes to allow the diver time to surface.

Precision Requirements:

- The system shall calculate the remaining oxygen left in the tank with a precision of $\pm$ 1 minute, in order to avoid killing someone because of a miscalculation.

Safety-Critical Requirements:

- The HUD shall not block more than 25% of the mask since covering too much of the mask may lead to blocking the view of the diver, and causing the diver to get injured.

Operational and Environmental Requirements:

- The system will be used in areas with high water pressure, so it will have to withstand waters of up to 100 meters.

Features to be tested / not to be tested:

- Ability to withstand ~100+ meters depth.

Approach:

- Test mask functionality in approximate water pressure level at 100 meters depth

Suspensions and resumption:

- Suspension: When >=70% of device functionality begins to malfunction
- Resumption: When <40% of device is malfunctioning

Test Cases:

- Device sonar shall produce accurate images up to ~100+ meters depth.

Design Goals:

- Sonar mapping should be computed and displayed as fast as possible.
- Vital calculations should be computed and displayed as fast as possible.

Software Architecture:

- multiple classes interacting and communicating with each other to create a proper task flow.

Subsystem Decomposition:

- Application
- User Interface
- Persistent Data Storage
- Peripherals

Object Design:

- Cost vs. Durability: high quality material used; costly.

Open Issues:

- Hardware Support: There is currently no hardware to support our product.
- Efficient software: The calculations that are required by the product is very intensive

Ready-Made Products:

- Full dive mask: Diving mask with a dive computer attached.
- Console diving computer: Handheld dive computer
- Wrist dive computer: A small dive computer that is wearable on the wrist.

Risks:

- Inaccurate metrics
- Slow computations
- High power consumption
- Inaccurate mapping

Costs:

Type	Cost
Requirements and use cases	$16,200
Project Design	$16,000
Project Development FP cost	$648,000
Project Development man-hour	$1,296,000
Hardware R&D	$2,000,000
Total	$3,976,200

Natural Disaster Help Application: Final Summary

Group 19
Briana Crockett, David Kay, Deep Patel

Introduction: We want to provide users with survival guide before, during and after a disaster by creating a general purpose application.

Description: The application consists of navigation and tracking subsystems which will perform several key tasks at different times during the disaster.

Along with the live navigation, we want to provide the users with offline navigation by downloading necessary data to the user's device before a disaster. This data should contain possible safe spots near the user's daily route. The user also has the ability to turn on tracking subsystem right before a disaster begins which will ping their location to the server every x minutes, which can later be for rescues.

During a disaster, the server will gather the user's pings (if user's device can still send data to our server) and calculate the velocity and acceleration to predict the user's position. The navigation subsystem will make use of the data on the server, offline data that it downloaded previously if no internet access is available, to guide user to the safest location.

After a disaster, the users will be prompted with a wellness check. On a failure or reply of no the server will deploy rescuers to the predicted location for rescue. The server will continue to predict user location until the user has been rescued or reply of yes in the wellness check.

Requirements: The system has a significant amount of requirements the most important being those dealing with the speed, efficiency and accuracy of the data transfer between the servers and the users devices. Because of the need for access to the data before the almost inevitable loss of internet. However closely following that in importance are the requirements detailing the need for the safety of the users. Such as not directing them into dangerous areas and facilitating their rescue after the fact. There are a number of things which are not requirements for this project. For instance as the system will not store users medical data we need not follow any guidelines in that respect. And in order to keep the number of security requirements low after a disaster is sufficiently in the past tracked location data shall be deleted.

Design: There are several design goals for our product but the most important one is accuracy vs speed. We should prioritize accuracy for the weather data due to the life/death situations. While also providing speedy navigation system so that the user can

get to safe location as fast as possible. The key thing here is to get the user away from the danger swiftly. Also supply users with information about how best to survive local natural disasters.

Test Plans: We plan to use many different APIs in order to decide which APIs to use we shall need to test them. However the maintenance of these APIs is up those supplying them. The testing of our software shall completed mostly before the app is published. However our most important test is that if the trend of people dying in a natural disaster goes down after the publishing of this app. There are certain test that will have to be constantly maintained so as to make sure that the quality of the system is kept up. For instance servers need to constantly be checked to make sure there is enough room for all the users at once.

Project Issues: There are several important project issues that will need to be address in the implementation phase. The first being that it is rather difficult to get a perfectly accurate model of the current weather. The system must employe some system to deal with this. Of similar note there are some big complications in getting safe directions during a natural disaster. A possible solution to this is that the users should have a well enough idea of the area around their home that given a location of a safe place they should probably be able to get there. However this should not dissuade us from implementing a direction system.

Final Project Summary:
Water Quality Monitoring and Flood Detection System
Group 20: Manasa Bandapalle, Xiaotong He, Zixuan Zeng, Snigdha Sultana

Description:
Our project is a software system that will maintain water quality for supply management from natural water sources as well as overhead detection of floods, and provides control operations to issue flood warnings to affected persons. The system is intended to be used by two major user groups: pharmaceutical companies and state water agencies.

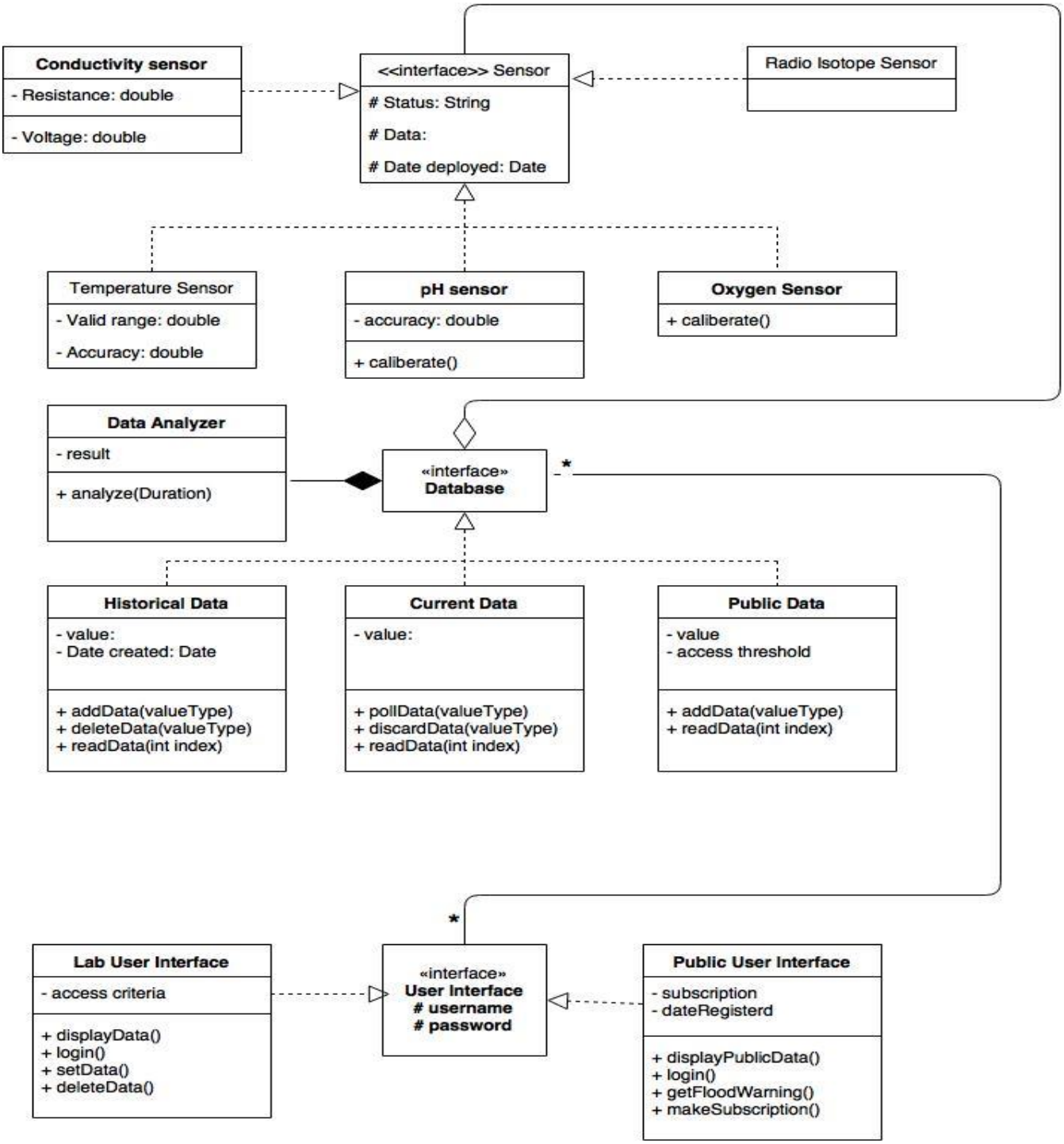
Requirements:
- System must collect the data from local weather station.
- System must provide water quality report according to the data collection.
- System must detect the flow speed/state of current water stream(rivers, creeks)
- System shall poll the sensors every 10 seconds.
- The water quality sensors must be polled frequently enough to prevent the contaminated water from entering the client factory. Since bad water quality can severely damage the production line, the client company.
- System must maintain the accuracy of the temperature within ± 0.2°C.
- The product database shall cater for 20,000 simultaneous public users with read access to the flood warning and water quality summary.
- The system is expected to have over 1000 sensors of different types. Only a small portion of them can fail at the same time.
- The system should be available at all times.
- The sensors and its connected circuit outdoors should be completely waterproof and weatherproof.

Test Plans:
Features that needed to be tested are sensors, temperature, pH, data delivery, database running, system robustness, changing to new sensors, working on existing OS system.

Design:
Our design goals can summarized as follows:
- The sensors should be durable to withstand deployment underwater and extreme weather conditions.
- Sensor readings should be as accurate as possible in order to analyze water quality and to determine if the water is suitable and meets the parameters for production use.
- The system should be tolerant to some extreme error readings.
- The database should cater to as much public users as possible, and at the same time, it should also be secure.

Our system architecture will use the Repository and Model View Controller patterns.

Class diagram:



Augmented Reality for Firefighting

Group 21: Amey Barapatre, Adarsh Janapareddy Venkata, Yumeng Yang

Summary

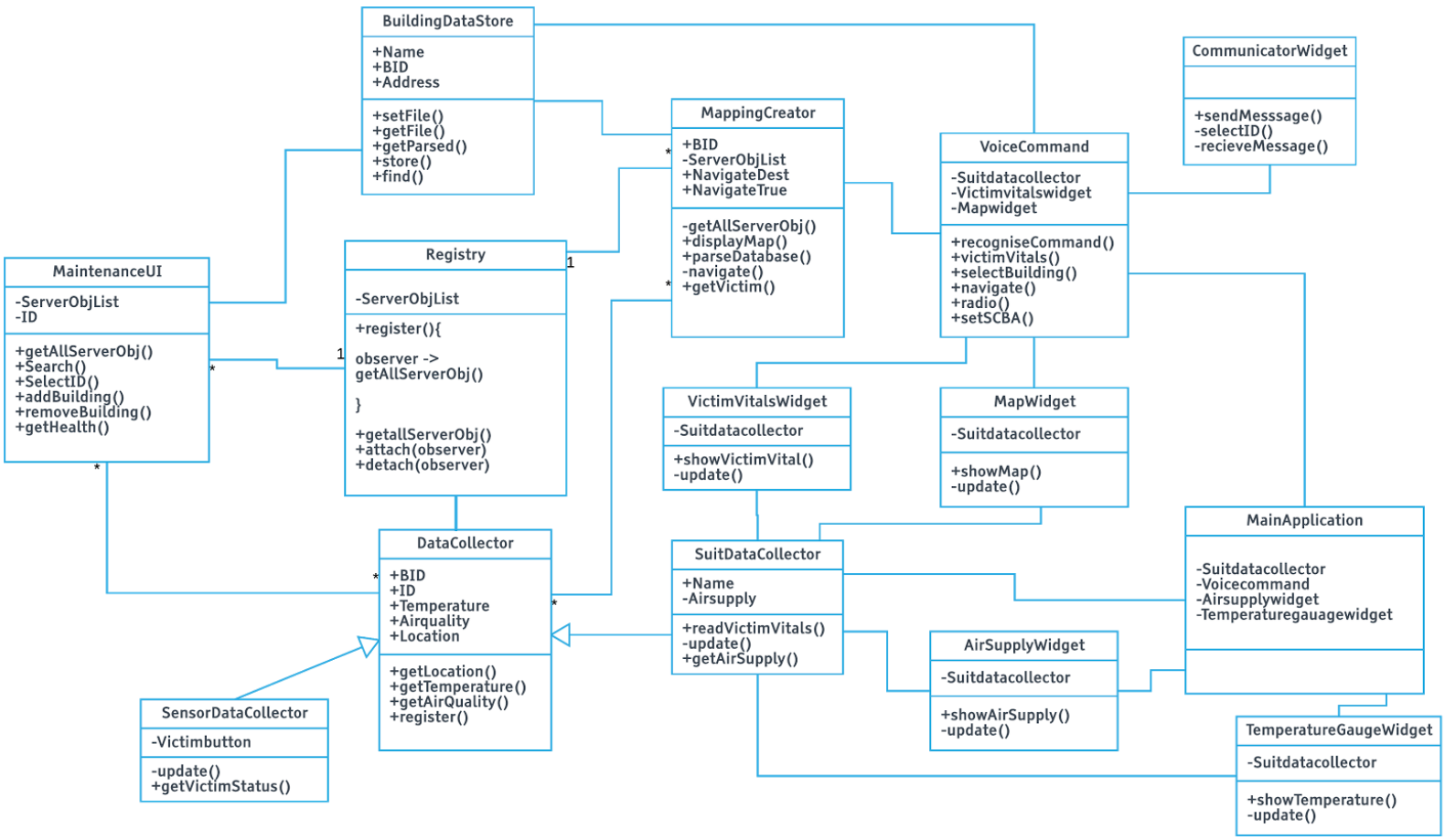
Description:

Augmented Reality for Firefighting is an Augmented Reality System integrated within the helmet for providing real time information of the environment and assisting the fire fighter. Some core functionalities of ARF are hands free operation, displaying information like temperature, air quality, SCBA backup. Also, it would display a real-time map of locations of victims, exits and other fire fighters. The data collected from sensors within building and from the suits is used to provide navigation with minimum risk and time to the requested destination.

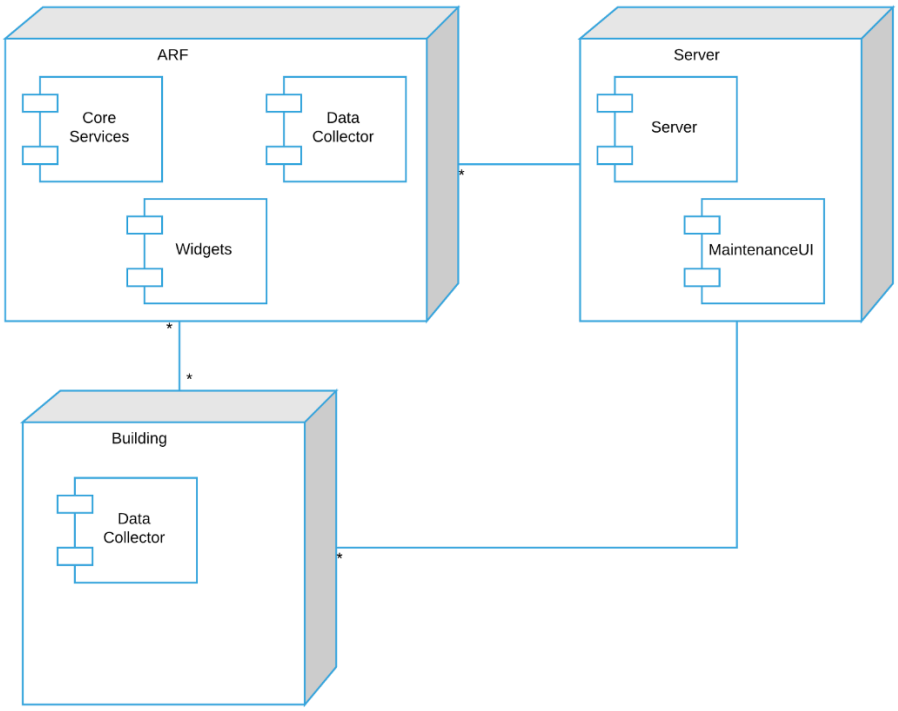
Requirements: Some key requirements are discussed below:

- **Functional:** System must help firefighter navigate through the structure in efficient fashion. It must inform the firefighter about the current situation in her/his surroundings. It must automatically handle the SCBA. It must help in assessing the system health. It must provide a way to modify the building database. It must keep a record of all the collected data. It must provide a way to visualize the current situation in the structure. It must provide a way read victim's vitals when prompted. It must be hands free operable.
- **Reliability:** The system must be available at least 10 hours one time. The material used to make suits, sensors, HUD, SCBAs and thermal cameras must be thermostable and water-proof.
- **Usability:** The quality of display on HUD shouldn't be influenced by strong light. The display on HUD shouldn't block eyesight. The colour of suits or parts of suits must be easily recognizable.
- **Performance:** The navigation algorithm must be efficient enough to respond quickly to the constantly changing environment.
- **Security:** The data available on the system must be securely communicated and stored.

Design:



Class Diagram



Hardware Software Mapping

Subsystem Decomposition:

1. **Core Services Subsystem:** MainApplication, MappingCreator, VoiceCommand.
2. **MaintenanceUI Subsystem:** MaintenanceUI.
3. **Server Subsystem:** BuildingDataStore, Registry.
4. **Data Collector Subsystem:** SuitDataCollector, SensorDataCollector, DataCollector.
5. **Widgets Subsystem:** VictimVitalsWidget, MapWidget, TemperatureGaugeWidget, AirsupplyWidget, CommunicatorWidget.

Testing:

The testing process is carried out by carrying out Black Box Testing which tests all functional requirements of the product. Key test cases are as follows:

- Verify that the system can collect firefighters' and Victim's position.
- Verify that the system can collect environment data.
- Verify the system can display environment data on HUD.
- Verify the system can display position data on a 3D model on HUD.
- Verify the system can predict accessibility to an area.
- Verify the system can compute path to victim.
- Verify the system can show directions on HUD.
- Verify the system can show current statistic of air supply remaining in the SCBA and numbers of victims to be rescued.
- Verify the system can provide technology for checking vitals of victims.
- Verify the system can establish communication between firefighters.
- Verify the system can provide hands free operation.

Conclusion:

Although technologies like SCBAs, thermal imaging cameras and digital pre-planning have been incorporated in modern firefighting efforts, the number of fire ground injuries per 1000 fires has remained relatively constant for the past 20 years according to report NFPA's *U.S. Firefighter Injuries-2015.* Thus, it is necessary for us to upgrade existing technology and adopt advanced technology especially firefighting is a hazardous work related to people's lives. Compared to traditional firefighting system, Augmented Reality for Firefighting, integrating HUD and sensors, strengthens the timelessness and integrity of information thereby helps firefighters improve rescue efficiency and minimize exposure to danger. The application of this system may also help in working in other kinds of hazardous environment.

SOFTWARE ENGINEERING PROJECT SUMMARY(CS 440)

SURVIVOR FINDER

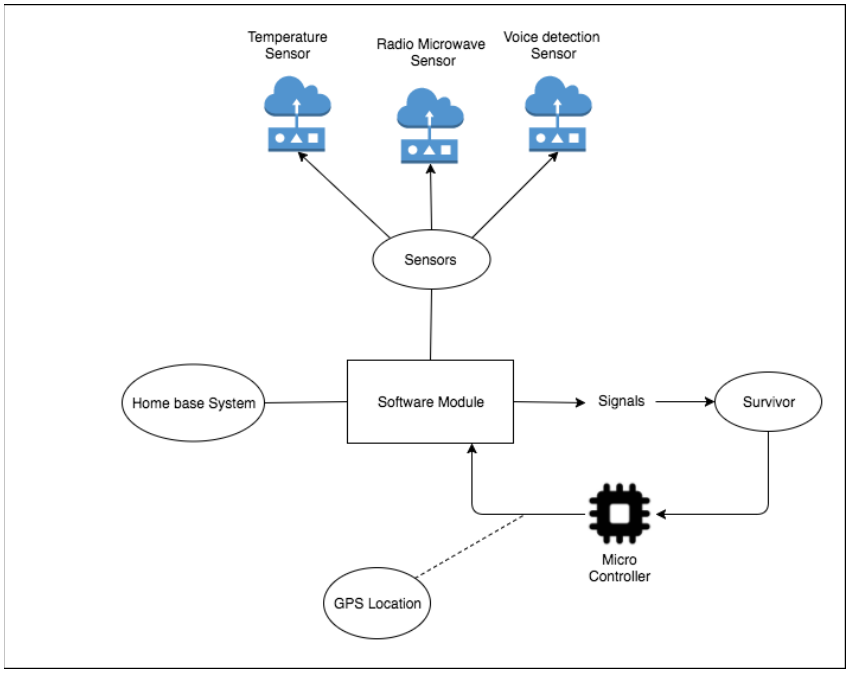
Malavika Srinivas
Sanjana Channabasappa Mallik
Nive Parthiban
Kaeyan Jones

INTRODUCTION

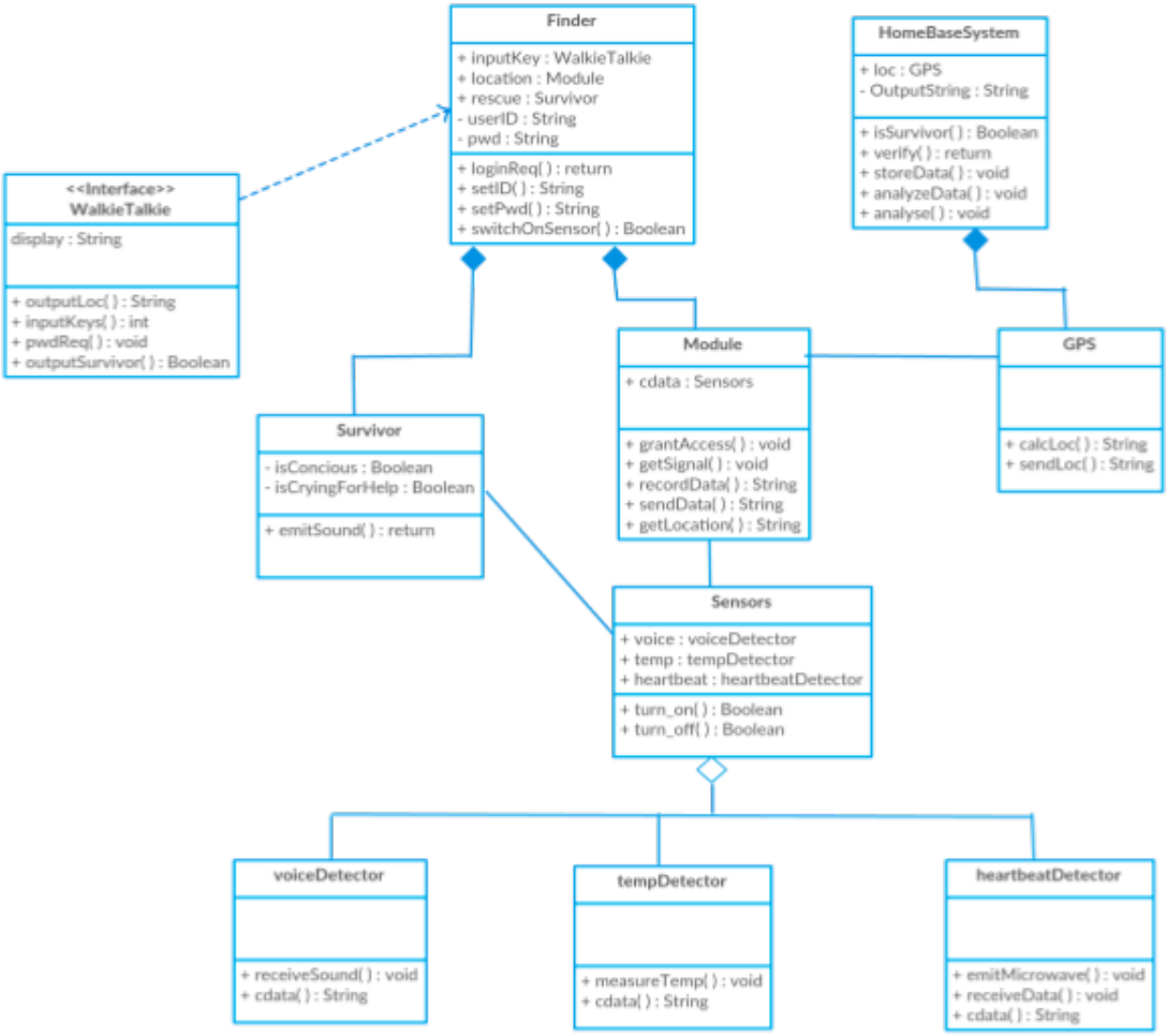
We have been witnessing the natural disasters that happen very often and also see the loss of life that is incurred because of that. The effects of this usually depends upon the affected population's resilience or ability to recover from it. During the times of such disasters, the survivors face threats such as death, physical injury, or loss of possessions and homes. In order to help such survivors, we aim to generate a comprehensive computing toolset to assist in search-and-rescue missions during or after natural calamities. The main intention is to devise a software that sends out signals at the location of the calamity. Depending on the varied frequency signals and voice signals that are received, classification is done to differentiate between humans and others beings. Microwaves are used to detect human heartbeat and breathing. Thermo sensors are also used to help in easy and faster detection of humans. Once a human being is detected, the location of the survivor is found. The rescue team reaches the exact point and takes necessary actions to rescue the survivor.

DESIGN

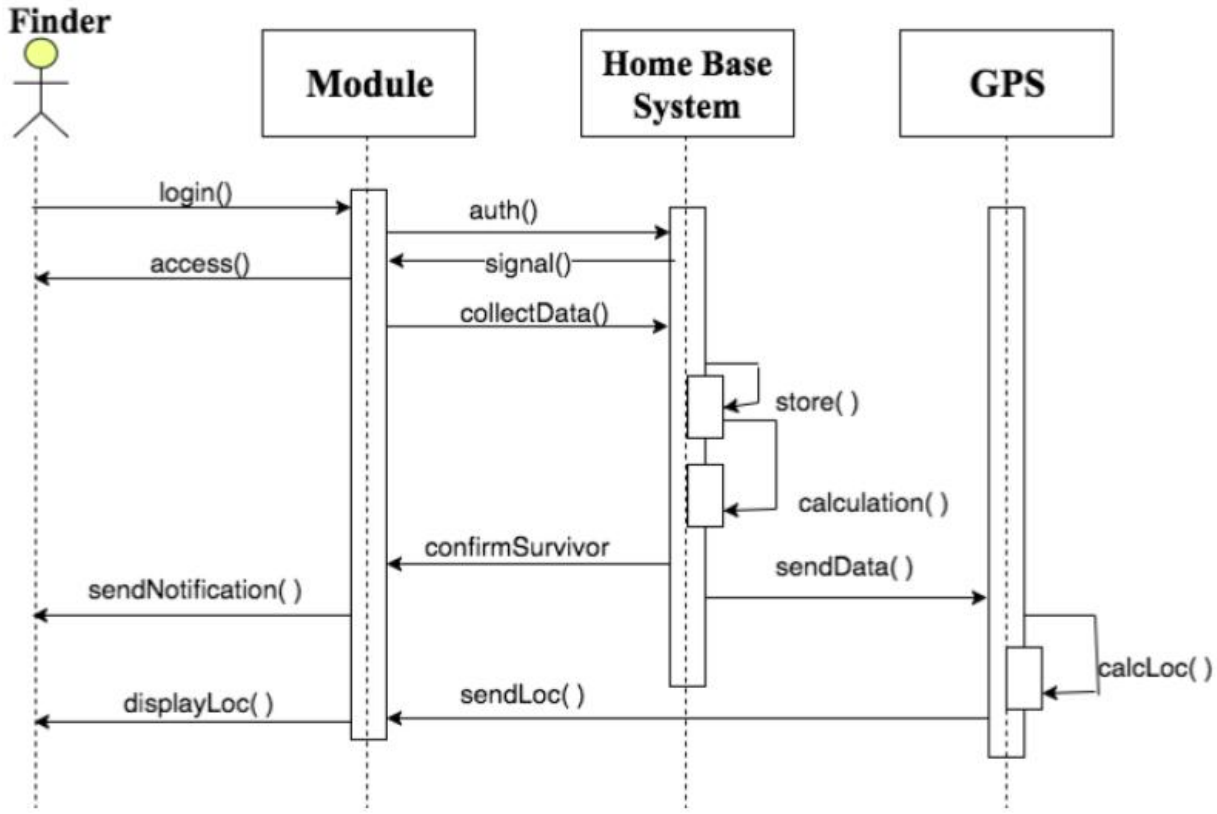
- System Design



- Class Diagram



- Dynamic Model for analysis and calculation of the exact location



PROJECT ISSUES

- Open Issues

Whether to use semi-custom or off-the-shelf hardware for the Walkie Talkies.

Hardware andOperating System basis for off-the-shelf Walkie Talkies.

- Off-the-Shelf Solutions

FINDER (Finding Individuals for Disaster and Emergency Response) is as close as a ready-made product comes but is more of a library solution.

- New Problems

Effects on the Current Environment:

The system should be designed in such a way that the consequences of doing so should not affect the current design and environment in a negative way

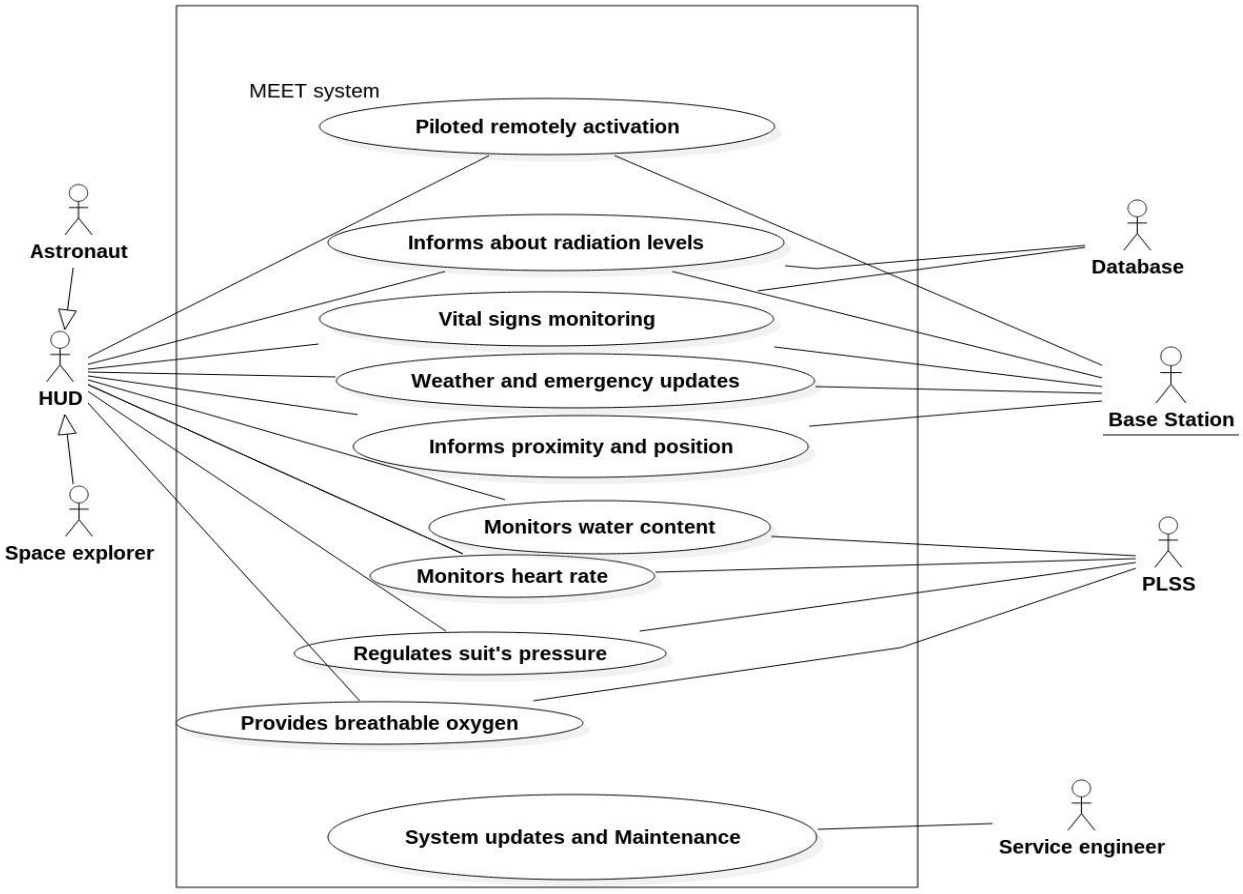
CONCLUSION

This project is designed to develop a software which has a number of modules that are connected to each other and to a home base system. These modules have various types of sensors within them that perform detections, which help in identifying a human being from other living and non living things. Along with this, the location of the survivor is found so that the users (search and rescue team etc) who are using the module can reach the site of action as soon as possible to rescue the survivor. This system helps in rescuing the survivors who would have been injured physically due to the effects of natural hazardous environments. With the help of this software, it will be possible to help minimize the loss of human life in disastrous environments.

Group 23
Final Summary
Manned Extraterrestrial Exploration Tool (MEET)
Cheng Chang | Suraj Gholap | Ibrahiem Mohammad | Yash Dharmamer

Overview:

MEET is an embedded software designed for exploratory missions to extraterrestrial planetary systems. It is in essence, the software that controls a smart spacesuit and protects the astronaut inside. The application software serves as support to the explorer via a HUD to keep them safe during their missions. This scenario diagram will present the MEET system in detail.



Requirements:

1)Functional Requirements

The various immediate functionalities available to the product users is as follows:
Radiation alert: MEET will alert and navigate the explorer to move to a safer location away from dangerous radiation
Remote pilot activation:, MEET will activate the remote pilot mode when the pilot is unconscious or unable to respond.
Weather and emergency updates: Precautionary warnings about the weather conditions will be relayed to the explorer through MEET.
Informs proximity and position: To enhance safety of the expedition team members, MEET provides proximity analysis system for detection of both above and below ground situations.
3D Rendering: MEET will provide an exploration aid in form of a 3D survey map on the HUD.
Vital signs monitoring: The software will maintain an information and telemetry systems which are configured

to provide real-time and immediate access to the astronaut's vital signs data.
Monitor water content in the suit: MEET with the help of PLSS will monitor the water content in the explorer's spacesuit.
Monitor suit's pressure: Suit's internal pressure will be regularly monitored with the help of the PLSS and pressure values will be displayed on the HUD.
Monitor suit's temperature: MEET will provide feedback if the suit does not insulate the user from extreme temperatures outside the suit.
Monitor breathable oxygen and poisonous gases: MEET will notify user if poisonous gases are present in the suit and about the status of the remaining oxygen in the suit.
Monitor Explorer's sleep cycle: Explorers/astronauts often experience chronic insomnia. MEET will keep a track of explorer's sleep cycle and notify the explorer in case of disruption.

2)Non-Functional Requirements

For Dependability, the system must be highly reliable as it will be used in a life-threatening outer space environment.
For Safety-Critical requirements, the project organizational structure should take care of the authority and responsibility of each Safety and Mission Assurance concern that may arise in accordance to policies set by the NASA Engineering and Safety Center (NESC).
For Maintenance requirements, the system must be able to be maintained by companies that are part of the project, such space companies as NASA, SpaceX, Boeing etc.
For Supportability requirements, the members of help desk should be among the astronauts and it should be located in space station or spaceships in order to provide timely support.
For Security requirements, MEET system should be highly secure. It can be accessed by Astronauts and respective engineers only.

Test Plans:

The types of testing that will be conducted on the system are: Unit Testing, Component Testing, Operational Readiness Testing, Load Testing Test, User Acceptance Testing, Integration Testing. Rigorous testing is needed to make sure there are no bugs in the software. For the system to be acceptable, system faults and errors detected during the testing stages must be resolved completely. There will be a managerial team consisting of Project Manager, Test Manager and Software architect overseeing the testing for the project.

System Design and Subsystems:

The design/subsystem are all based on the HUD, base station, and the database. The design itself revolves around the HUD having multiple interactive views with which the explorer can navigate, explore, and collect data critical to the mission. It then connects with the base station for updates and data transfer. The database is used least often, used mostly when the HUD is offline. The system uses blackboard architecture--solving new issues by breaking them down.

Mining Safety Management System - Final Project Summary

Group 24: Dimitar Kirilov, Abhishek Ranjan, Piyush Chandra, Ajinkya Upasani

Project Overview:

Mining Safety Management System is a project which incorporates both software and hardware in order to lower the risks posed to the miners. The project incorporates an overarching alert system for the mining industry which triggers alarms based off of previously collected data which has been analyzed for potential patterns and causes which may trigger hazardous situations.In this document, we are providing both functional and non-functional requirements along with the test plan summary of the project as well.

Functional Requirements:

The functional requirements for our project were primarily focused on the interaction between the sensors, the central system, and the users. The requirements derived from the sensors describe the data flow, its format, and how it reaches our central system. The requirements derived from the central system dealt with data transfer and issuing alerts throughout the mines as well as analyzing data and predicting possible hazardous situations. Finally, the requirements derived from the users dealt with how the gear they use receives data and how the workers are able to communicate with others.

Non-Functional Requirements:

Usability Requirements: The usability requirements for our project heavily focused on creating a easy to use interface and tools for the users of our product. To achieve this, we require hardware and software which is simple to use and demands minimal technical experience.

Reliability Requirements: The reliability requirements focused on creating a fail safe system and minimizing the potential risks which could be caused by making our system as reliable as possible. The product has strict up time requirements and is always expected to fail in a safe manner.

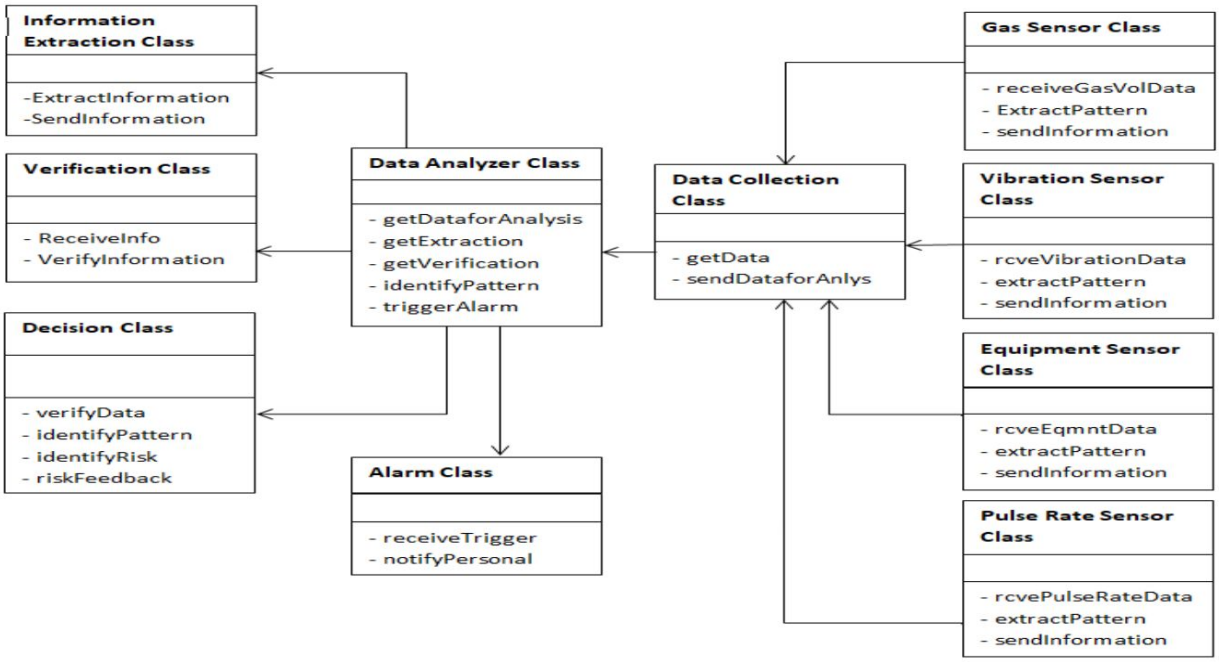
Performance Requirements: The performance requirements focused on setting specific constraints on the speed of operations and functionality through our system. This includes the speed of which data is sent from sensors to the central system to how fast information must be processed and calculated.

Supportability Requirements: The supportability requirements focused on achieving proper support for our users. The requirements involve having a well maintained support system which includes proper documentation, manuals, and a customer support team.

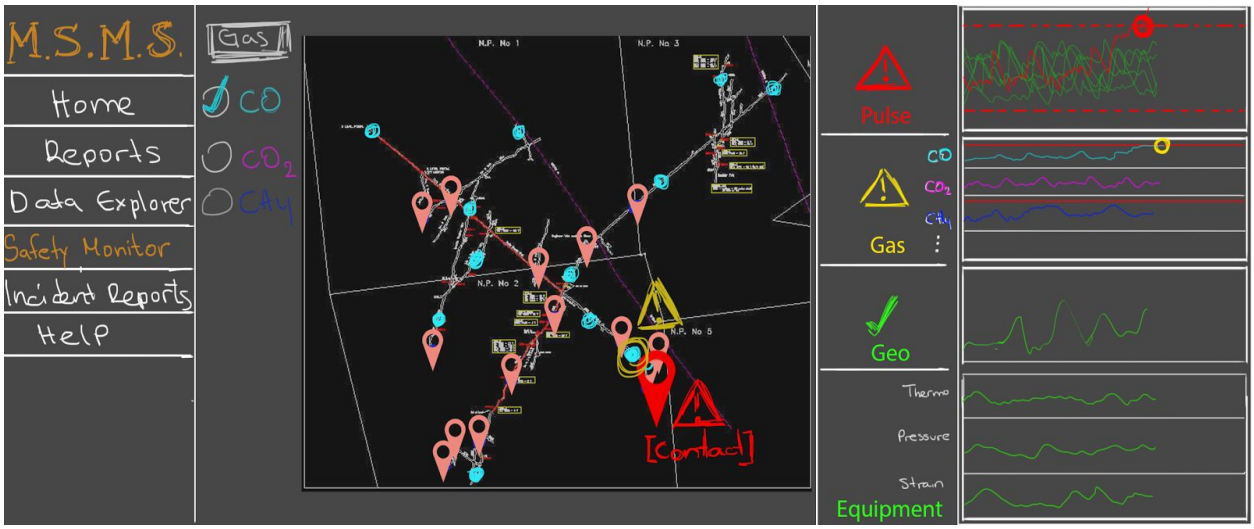
Test Plan:

For the hardware components,e will be testing all sensors, radio communication, network coverage, and alarms. This will be done by exposing the technology to a known input and determining whether the output is precise and accurate up to a defined threshold. For the software components, we will be testing our data analyzing algorithms, and our report generation tool. The tests for these two components will consist of unit, component integration, and acceptance level tests.

Class Diagram:



User Interface:



Conclusion

The project aims to improve the safety compliance of mining industries. Small scale mining industries occupy substantial portion of the overall mining industry. Safety compliance in such industries is often neglected. The project aims to build a product that would help improve the overall safety of mining industries in cost effective manner

SOFTWARE IN SUPPORT OF HAZARDOUS ENVIRONMENT

GROUP 25: Sarah Kazi, Vignan Thmmu, Vipul Vivek Haralkar, Dinesh Anand Sivasubramanian Surianarayanan

WASTEPRO: THINK WASTE, THINK SAFE.
PROJECT FINAL SUMMARY

Description:

The idea of this project is to help the garbage collectors to cautiously collect and dispose garbage without exposure to harmful materials (bio-waste, e-waste, metal waste, Volatile Organic Compounds etc.) and to help reduce the accidents caused by traffic.

Functional Requirements:

From the use-case flow of events we can get the functional requirements. The main functionality of the system is to reduce the number of deaths caused to garbage collectors by making their work ease and safe. Some of the functional requirements include,

F1. Metal detectors must detect all amount metals.
F2. VOC meters must record even tiny traces of VOC present in the dumpster.
F3. Overflow sensors must detect overflow.
F4. System must indicate garbage collectors through application interface that there might be an overflow.
F5. Weight meters must record the weight of the dumpster.

Non-Functional Requirements:

Some of non-functional requirements include

NF1. The system should be easy to use by garbage collectors and should be organized in such a way that user errors must be minimized.
NF2. The priority list calculated must be accurate.
NF3. Multiple priority lists should be generated at the same time (parallel processing)
NF4. Depending upon the usage server capacity can be increased.

Design and Implementation:

The main classes identified include Dumpster, User Report, Garbage Controller, Priority List and Garbage Controller. The dumpster uses VOC detector and metal detector to capture information about harmful materials and Garbage controller uses belt which captures the parking sensor information and notifies garbage collector the incoming traffic. The web interface acts as a central repository to store and process information. The important thing to note is priority list contains a

private method of generate priority list which is not accessible to other classes and not editable. This is because the central theme of the project involves the generation of priority list. Once the garbage controller clears the garbage the web application interface notifies the Priority list class to reset the priorities. And then the process repeats.

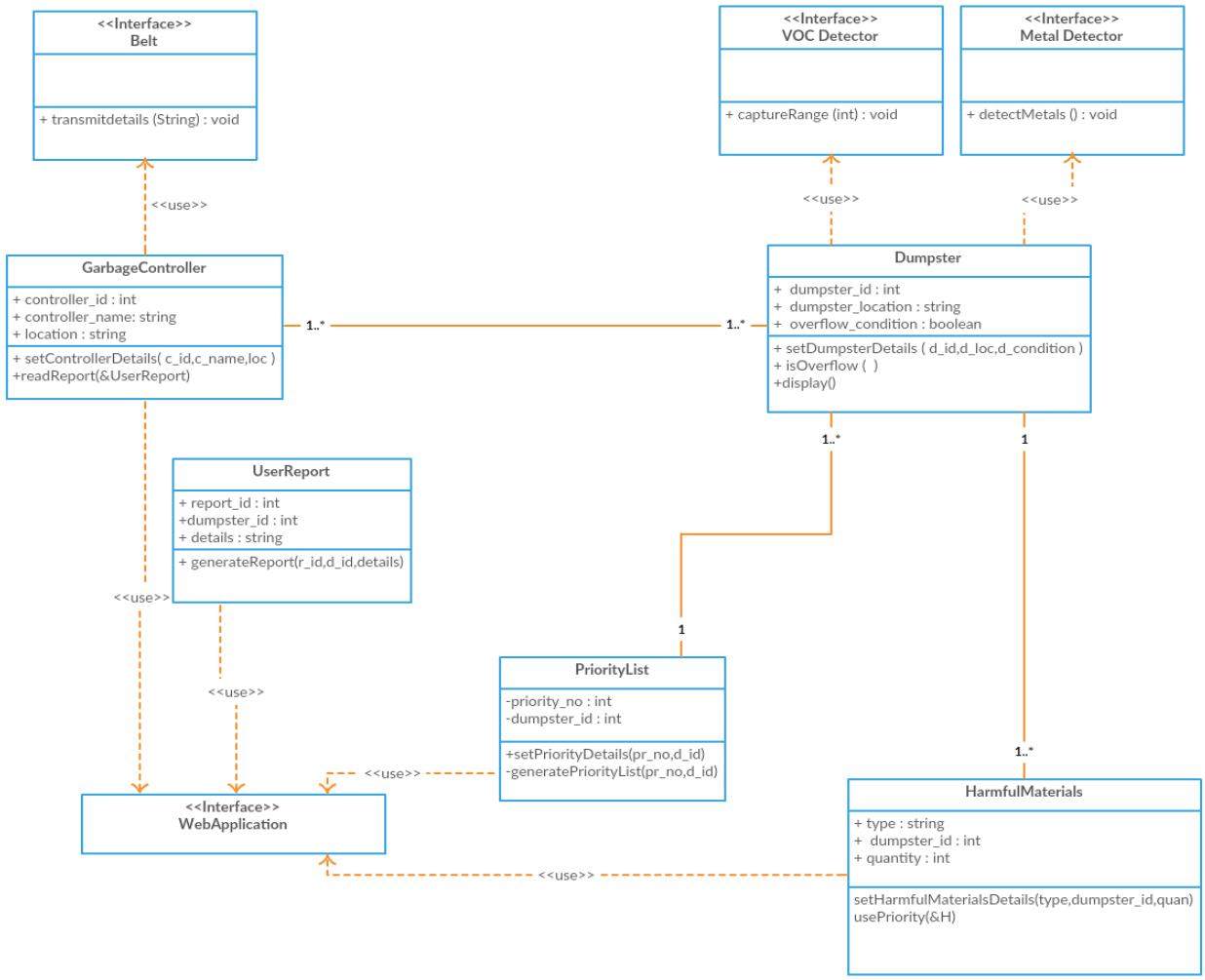


Figure 1: Class Diagram of Waste Pro

Testing:

For each requirement, there is a fit criterion and tests must be carried out to check whether the fit criterion is satisfied. The testing can be done using selenium web testing tool which tests the capacity and fault tolerance of the web-server. The most important part to be tested is the priority list generated whether it conforms to the theoretical one and whether it uses the harmful material information stored. The garbage collectors must follow the priority list irrespective of their schedule.