S.A.R.C.I

Search And Rescue Coastal Intelligence

Requirements Specification

12/07/2021

Version 1

Team Mentor: Han Peng

Group Members: Dylan Woolley, Vidal Martinez, Randy Duerinck, Jabril Gray

Team Sponsor: General Dynamics Mission Systems

Accepted as baseline requirements for the project:

Client *sign/date:*__________________ Team *sign/date:*___________________

Table of Contents

1. Introduction

The United States has one of the largest exclusive economic zones (EEZ) in the world. An EEZ is an area of ocean where the owning territory has exclusive rights to the exploration and resources of the area. The United States has 4,383,000 square miles of EEZ which means the United States has 4,383,000 square miles of EEZ to patrol and protect. 4,383,000 square miles is about 1.14 times larger than Canada and about 1.155 times larger than the United States. The United States Coast Guard (USCG) is responsible for watching and protecting our coasts and this is a major job. Our client, General Dynamics Mission Systems (GDMS), is responsible for working with the USCG on the Rescue21 system for assistance with search and rescue (SAR) missions. Whenever there is a problem at sea and you need assistance the USCG is the one to pick up the call but how does the USCG even begin searching about 4.4 million square miles off our coast? The answer lies in the Rescue21 system which is a system of antenna towers, called remote fixed facilities (RFF), that are placed strategically off our coasts with directional finding very high frequency radios that are used to pick up distress signals, the direction of the signal, the strength of the signal, and whether or not it is a hoax call. This information can be used to dramatically reduce the area that needs to be searched saving time, money, and increasing the likelihood of those in distress being rescued.

GDMS is content with the Rescue21 system however they want some additional features that will help them with maintaining their system. There are two main features that they need: the ability to record the power levels of the antennas at an RFF and the ability to record the weather information at the RFF sites. Knowing this information will help GDMS with determining the cause of RF interference. RF interference causes static while communicating over the VHF signal and it can be caused by a few different factors. One of those factors is that the radio equipment is broken and needs to be replaced. The other factor is rain. Rain is conductive so if there is rain in the air then the signal will be absorbed by the rain and the signal will fade over distance. Knowing the cause of RF interference will help GDMS determine if they need to send a technician to an RFF site or not. The other reason that the weather information is helpful is when predicting damage. Knowing if there is a severe storm at an RFF site will help GDMS with predicting equipment outages before they happen so they can have someone ready to fix the site before an outage happens. This will help reduce outage times ensuring that the USCG can always help those in need. Knowing the weather also helps with scheduling maintenance on a RFF site. It is risky for GDMS to send someone to climb the antenna towers in the middle of high winds or storms. Knowing the weather will help GDMS reduce the amount of times they send a technician to a site when they cannot perform maintenance.

GDMS contacted Northern Arizona University (NAU) in order to build the device that would record the antenna power and weather data last year. The project went very well as they were able to build a site weather and power recorder (SWAPR) which worked properly. The only shortcomings to the SWAPR device was that the data being given out is not human readable. That is why GDMS reached out to NAU again this year to build the software for the SWAPR network. GDMS is looking to have the data be displayed in a human readable way on a secure website. In the coming document we will outline exactly what GDMS is expecting out of this project.

2. Problem Statement

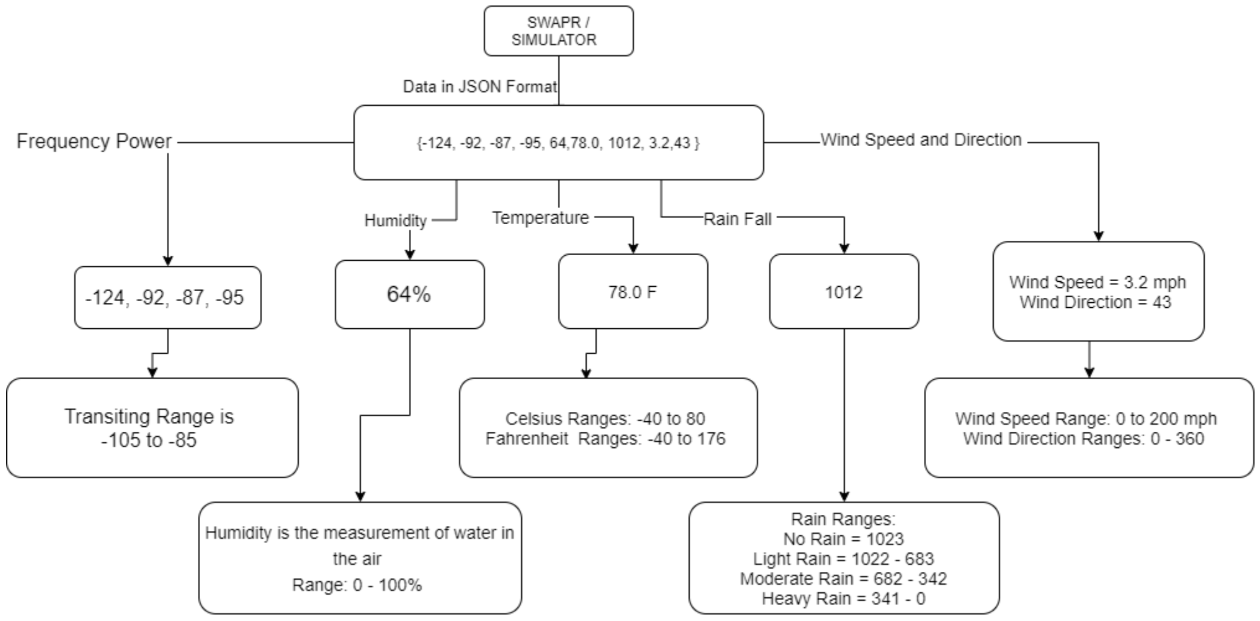

Figure 2.A: SWAPR Data Diagram outlining expected data values for each attribute.

We are trying to solve the problem with the SWAPR device which is that it only collects the power and weather data from an RFF site but it doesn't do anything to make the output human readable. Referencing *Figure 2.A*, we can see that a SWAPR device will output the antenna power and weather data in a list of numbers like [-124, -92, -87, -95, 64, 78.0, 1012, 3.2, 43] where the numbers represent [frequency one power (dBm), frequency two power, frequency three power, frequency four power, humidity (%), temperature (F), wind speed (mph), wind direction (degrees)]. There is a print out every five seconds and there are almost 250 RFF that would have one of these devices. As you can see that is a lot of information to go through as a human and make sense of. Therefore, we are building a secure website that will collect, store, analyze, and build and display a couple of different views that can very quickly be used by GDMS to make decisions that reduce the risk of personal injury, and save on labor costs.

1. SWAPR device output is unreadable to humans
    a. [-123, -92, -87, -85, 64, 78.0, 1012, 3.2, 43]
2. SWAPR device output is hard to analyze by humans
    a. The first value of -123 is telling us that there is a potential problem with the first frequency because -123 is not in the range of acceptable values. However, the other three frequencies (-92, -87, -85) are all working as they should since they are in the range of acceptable values.
3. There is a large amount of SWAPR data

   a. SWAPR devices give out data every five seconds
   b. Around 250 SWAPRs in the system
4. The data needs to be easily accessible
   a. SWAPR devices do not store any data
   b. If SWAPR data is stored it is only stored on the device
5. SWAPR devices are very simple and cannot send their data
   a. SWAPR devices are built to only record information and have no way of sending or analyzing data
6. SWAPR data needs to remain secure meaning it needs to be stored and accessed in a secure environment
   a. As the SWAPR data is only on the device, it is not being stored or accessed

3. Solution Vision

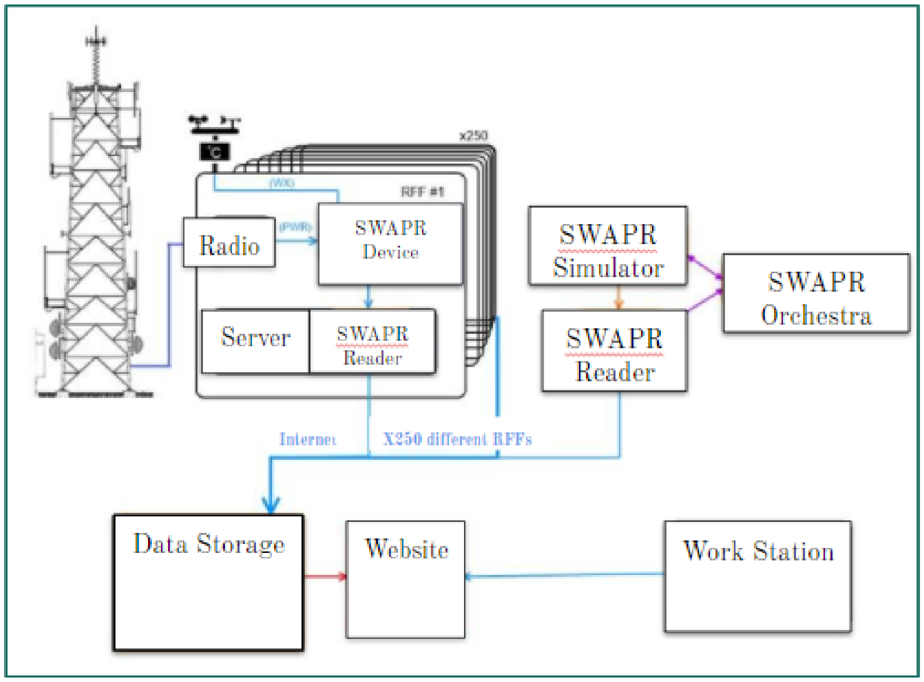

Figure 3.A: Diagram showing how the 5 subsystems will work together.

Our solution to building the additions that our client would like added to the Rescue21 system will expand on last year's capstone project by creating the software to collect, store, analyze, and build and display a couple of different views in a secure manner. As shown in *Figure 3.A,* we envision our project being broken up into five different subsystems which all work together to create a solution for our client. Those five subsystems are called the simulator, reader, database, website, and orchestra. The simulator will act like a SWAPR, generating data and sending it to the reader subsystem. The reader will receive SWAPR data and send it to the database system. The orchestra system will be an extra requirement that connects any number of reader and simulator systems. Our database will store and send SWAPR data to our website system. Our website subsystem is the biggest subsystem and is responsible for displaying the SWAPR data in a few ways. The first and second graphical displays are a list view and map view, which both display a summary of the newest entry from all SWAPR devices in the network. The third displays historical SWAPR data through different graphs such a line graph and wind rose compass. We will have these displays in a secure website environment that has functionality to retrieve, analyze, and build the graphics.

In order to do these views we will need a way to first get the data so we need a way to securely access the database and query it. There are two query methods, one that gets the latest entry from each device and the other grabs all data from one SWAPR device over a date range.

We will also need a way to get the SWAPR statuses for color coding the map view. This goes hand in hand with our notification system since if the status is anything other than green or in other words functioning properly then we need to create a notification to be shown on the website. If we do create a notification then we will store that in a queue until the next admin user logs in so we can display it to them until they clear them. We also need a way to export SWAPR data used to create graphs for the historical view and put it in a .csv file that can be given to the user. Lastly, we need to wrap all of this functionality into a secure website environment that has user accounts and role-based permissions. We will only need to create two roles which are the user and administrator. The user will have access to everything except the notification system. The administrator will have access to everything including the notifications.

- **Simulator Subsystem**
  - C# Random Class will randomly generate SWAPR data for valid and invalid data
  - We will be using System.IO.Ports.SerialPort class to, Send data over virtualized com port on RS-232 protocol
- **Reader Subsystem**
  - We will be using System.IO.Ports.SerialPort class to read data over virtualized com port on RS-232 protocol
  - We will be using System.Net.Sockets.TCPClient class to establish a TCP Client connection with the database queue
- **Database Subsystem**
  - We be using TCPListener class to establish a TCP Host connection with the reader subsystems
  - Queue TCP data collected in AWS SQS
  - Ability to create database entries from data in AWS SQS queue
  - Hosting the database server off AWS RDS
- **Website Subsystem**
  - Secure authentication between subsystems as well as “faking” user accounts and role based permissions

    - Ability to query the database server in a secure manner
    - Notification system that detects when there is a problem with an RFF and creates a modal for a specific user group
    - When in the graphical view of historical SWAPR data the user should be able to export the selected data to a comma separated file
    - Ability to create a list-view summary of the latest data from all SWAPR devices in the network
    - Ability to create a map-view summary of the latest data from all SWAPR devices in the network
    - Ability to create a static graphic view of historical data of one SWAPR device
- **Orchestra Subsystem (Stretch Goal)**
    - Connects any number of simulator and reader subsystems over virtualized com ports for lab environment stress testing

Next we will go over our project requirements in detail. Our subsystems above are listed with their key functionalities as sub-bullets. The following section will cover what is expected of our overall system followed by the functional, performance, and environmental requirements defined for our project.

4. Project Requirements

As we have affirmed in our previous section, our system consists of five main subsystem components that en masse form our project. Our aim is for this project to provide functionality for a secure website with retrievable weather data for creating informative visuals. The performance requirements specify our need for speed, and ease of use. The Simulator, Reader, Database, Website, and Orchestra subsystems are responsible for meeting our project requirements.

- Create data imitating the SWAPR device's output
- Store and serve SWAPR data in a secure manner
- Take data from a SWAPR device and send it securely off-site
- Establish a secure website environment with authentication
- Create a list view summary of the SWAPR devices in the network
- Create a map view summary of the SWAPR devices in the network
- Create graphs using historical SWAPR data
- Create a way to notify operator when there is a problem with a RFF site
- Export data from the database as a .csv file
- Emulate the entire SWAPR device network for stress testing

The Simulator Subsystem has a direct connection to our Reader Subsystem and together they drive our real environment emulation by providing simulated SWAPR output and secure data transfer. The Reader Subsystem must read data, then package the received data over a TCP connection to our Database Subsystem. The data from our database is then retrievable by our Website Subsystem. Authorization is required for access to our database by user-based role authentication from the Website Subsystem. The retrieved data can then be viewed in different historical and summary views on our website through the graphical user interface. The historical view includes graphs or charts generated using data from the database. The summary view includes the list and map view; both show details of each SWAPR device and give the user an understanding of summary information such as their location, current temperature, windspeed, and more. Notifications are needed for our system so users are up-to-date on the status of the devices. Data should also be able to be stored locally from the website so users can save data when necessary. The Orchestra Subsystem is a stretch goal for our project. It utilizes the simulator, reader and database subsystem for the reason that the orchestrator utilizes the simulated SWAPR device output and data transmission in the reader software to establish an instantiatable emulation of the communication. The orchestrator subsystem will allow us to stress test various aspects of the sending and receiving of data from device to database.

4.1 Simulator Subsystem Requirements

The simulator subsystem considers random generation of SWAPR data and data transmission over a virtualized com port on the RS-232 protocol as the prominent functionalities. Random generation of data gives our environment a reproducible, ostensibly non deterministic model to provide concise and effective test cases for assessment as we develop our project. An example of this is wind direction. Our simulator, to simulate a real-world scenario, will need to have altering wind speeds and distinct wind speeds for a given SWAPR devices produced output. To achieve this, we will generate random values ranging from 0 to 360 degrees to test handling and viewing of this data output from a given SWAPR device. Additionally to this, we will look to have this value not be the same across multiple devices to emulate a real environment.

Sending this generated data from our simulated SWAPR devices over the virtualized com port allows additional testing conditions to evaluate. Altogether we are building a subsystem capable of evaluating the expected circumstances the Rescue 21 system will operate under in a controlled environment without needing real SWAPR device hardware or a real RFF site. This is a vital functionality in our system. Without a virtualized serial port in our simulation we would have no way of sending data between our applications, the reader software and the database. Not only would this complicate simulating the SWAPR devices mode of data transmission, but would also leave us incapable of testing this part of our system and potentially developing an incompatible system to work with the actual hardware version of SWAPR devices.

When considering these functional requirements we also evaluate the performance requirements. The SWAPR device simulation must be capable of generating output data in 5 second intervals or even a faster frequency if possible. This data must be generated at such a high frequency, because this data from the SWAPR devices is the basis for all analysis and visualization the users will execute and produce respectively. If data generated to simulate a SWAPR device is being produced on an interval different from reality, then we may face a skew in our systems expectations of the SWAPR devices performance which will likely impact the development of our system.

The output data generated from the simulator must be transmitted by the virtual com port to the reader software in the Reader Subsystem at a 9600 baud (Bd) rate - a rate of 9600 bits per second. It is important for the rate of bits transmitted from a SWAPR device simulation to the reader software through the virtual com port to imitate real standards for bit rates. This rate of data transfer, as a maximum bit rate in the serial port, guarantees our simulation imitates a real and common standard. A 9600 Bd rate is an adequate speed capable of streaming over 10 lines per second.

4.2 Reader Subsystem Requirements

In the Reader Subsystem the data read in over the virtual com port using the RS-232 protocol and the TCP client connection established with the database are the two functional requirements our team needs. The received read data transmitted over the com port comes from the Simulator Subsystem. The necessity of this functionality is the same for the Simulator Subsystem's functionality for sending data over the virtual com port. Using the virtual com port imitates a real-world activity a SWAPR device will have, allowing our team to test this subsystem with realistic cases.

Establishment of a TCP connection with the database queue is a process that will be used in a real implementation of our system. This connection is necessary in transporting all data from a SWAPR device to our database queue for usage in the Website Subsystem. The Reader Subsystem is expected to handle data transportation with the assertion that our database always receives generated SWAPR device data from our Simulated Subsystem when a TCP connection with the database queue is acknowledged.

The functional requirements we are concerned with also impose a couple performance requirements for our subsystem. The virtual com port connection between the Simulator Subsystem and Reader Subsystem must have a 9600 Bd rate - identical to the Simulator Subsystem performance requirement. A standard rate for communication between both subsystems is a standard for com port data transmission and should be met as any other adequate system requiring data transmission would.

The data received from the Simulator Subsystem is to be transported over the TCP connection to our database in at least 5 second intervals. This means that our connection must have a round-trip time (RTT) of 5 seconds where every 5 seconds a data entry from the Reader Subsystem is fully received by our database queue. Our time restraints are calculated based on the importance of storing data in our database within a feasible time to avoid delays in other subsystems down the pipeline such as the Database Subsystem or Website Subsystem.

4.3 Database Subsystem Requirements

The functional requirements for the Database Subsystem are primarily concerned with TCP connection and building the database. The TCP host connection needs to be established with the Reader Subsystem to allow communication between the two subsystems.

As the receiving end of this connection, the data incoming from the Reader Subsystem must be placed in a queue using the Amazon Web Services Simple Queue Service (AWS SQS). Storing the data from the TCP connection in a queue using the AWS SQS is the first step in storing SWAPR device data in the database and making it usable in the Website Subsystem.

Once in queue the SQS can create database entries to be entered into our database. The database entries are required for building our database on the AWS server. With a database populated with entries acquired from the TCP connection, we will have all SWAPR device data accessible by users in the database.

The hosting for the database server will use the Amazon Web Services Relational Database Service (AWS RDS). The RDS will be necessary for hosting our Database Subsystem which can be used with our Website Subsystem. With a secure TCP connection between the server host and Reader Subsystem client, an established database, and hosted by our AWS server, our Website Subsystem can access any available data generated by the Simulator Subsystem.

The Database Subsystem performance requirements deal with the specifics for the data queue in our database and the total storage expectation of our database. The received data from the TCP connection established with the Reader Subsystem must be received and placed in a queue in 5 second intervals. This time interval is necessary to guarantee adequate time for this subsystem to acquire data and place this data in a queue without underestimating the time it will take.

As mentioned with the Reader Subsystem, we want to guarantee a sufficient turn around time for generated data from the SWAPR devices. The database needs to be capable of communicating with more than 200 facilities each with their own SWAPR device. Our database should be capable of handling inputs from every connected device in an uninterrupted, real-time stream with each SWAPR devices data being stored in their own data entry within the database.

We need the data entries to be built within 5 seconds for the sake of maintaining accessibility to our database's data. This is important to maintain a consistent speed from the time of data generation in the Simulator Subsystem to entry creation in the Database Subsystem. With every interval of data entry creation and insertion into the database taking approximately 20 seconds, the data stored in the database should be accessible in a reasonable and realistic amount of time. We believe this length of time may be generous, but once our team completes our prototypes, we should have a clearer understanding of exactly how long this process can take.

Not only is the speed of our data generation and storing important in performance, but the capacity of data that can be stored is important as well in our Database Subsystem. The total amount of data stored in our database should be capable of reaching at least a year. This would mean that with a 24 hour run-time with data generated and stored from a single SWAPR device in five seconds we should expect to store approximately six million data entries with each entry being generated from a single SWAPR devices output. This will be done for 250 SWAPR devices which creates about 1.5 billion data entries. Our Estimate of a given file is about 50 bytes, therefore the total amount of data stored on the server after a year will reach 75 gigabytes.

4.4 Website Subsystem Requirements

The functional requirements for the Website Subsystem Backend are hosting the Blazor Server on AWS, generating a graphical view of historical SWAPR data, generating notifications with a notification system, and establishing secure authentication between subsystems as well as “faking” user accounts and role-based permissions.

We need to create a Blazor Server application that will take and display information retrieved from our database. The Blazor Server can be hosted on AWS to allow features covered in the following functional requirements to be achievable.

Once we can query data, we will need to use the historical data from the database to create graphs for the user's view. Graphical views of the historical data will allow users to view the data in organized and summarizable ways that will improve the user experience. For example a line graph, wind rose compass, or radar chart.

Our notification system will be responsible for notifying users when there is problematic information or a response signifying a critical event in a SWAPR devices output data once received by the backend of the Website Subsystem. The notifications must be displayed when the next admin user logs in to the website. Upon detection of a critical event in the backend of the Website Subsystem a notification will be created instantly in collaboration with the frontend interface.

There are four types of statuses for a SWAPR device. There is green which means that everything is working correctly and the data is in acceptable ranges. Yellow means that there is an issue with the weather data which is detected from data outside of acceptable ranges. Orange means that there is a problem with the antenna power that was recorded. Red means that there was no response received from the SWAPR device so the site is offline and the problem is unknown. This is important for informing technicians that are not on site of issue and allow the technicians to visit a site with the knowledge of its operational status beforehand.

The Coast Guard and GDMS need to ensure that only authorized users are able to access SWAPR devices and their data. To do this, we need a secure authentication system that consists of two roles: user and administrator. We want authentication that works well with the Windows operating system, because this maintains consistency with General Dynamics' requirements. Currently, we will fake the user and administrator roles because of budget and time constraints, the software needed is available and applicable to the purpose. The software will be recessed at the end of the project for implementation.

For our Website Subsystem Backend, we have a number of performance based requirements. The Website Subsystem needs to retrieve data for summary and historical views in under five seconds, then render the graphical map view of the SWAPR device data taking no more than six seconds for retrieval and ten seconds for rendering. Our team is looking for a wait time of 30 seconds or less to view the most recent incoming data from our database in the desired format.

Our backend also needs to be able to take in historical SWAPR database information and extract it to a csv file in under three seconds. The creation of the csv file in this time frame is helpful in guaranteeing the data can be accessible in a downloadable, text-based format for a given user's usage.

The functional requirements for the frontend side of the Website Subsystem are focused on the ability to create a list-view summary of the latest data from every SWAPR device in the conceptual network of RFFs. The same is needed for our Website Subsystems map visuals; we need to create a map-view summary of the latest data from every SWAPR device. Having this functionality provides users with alternative views to text-based displays.

Our team needs our website to be able to create a static, graphic view of historical data for any given SWAPR device and as an additional stretch goal we want to have the graphical view be interactive. On the other end of the Website Subsystem, the frontend portion of functionality is specifically relevant to the interface the user sees and interacts with. The historical data in the database should be viewable as a graph whether that is a wind compass, line graph, or bar graph for example. The interactivity we look to add expands beyond the static images that can be generated to include adjustable graphs. This for example could be an adjustment of the range of dates or of the range of values affecting either axis on a graph.

The performance requirements are necessary for defining the speeds to which data is created and updated in our website's frontend. Without consideration and efficiency our website may suffer poor refresh times and load times. The time to process and create our list-view summary widgets should be under 6 seconds and under 3 seconds for the map-view summary.

The live widgets in the list-view should be refreshed approximately every minute. This is the amount of time approximated based on the generation of data and retrieval of data from our database.

The map view will need to take into consideration the Area of Responsibility (AOR). The AOR describes the expected capacity of RFFs to view in an instance. The AOR for the SWAPR devices is currently set to 18. Our team will need to have a system to view these devices as their correct set sizes for clarity in their relative responsible sectors.

With generating data for visualizations on the frontend of our Website Subsystem, we need to create requested graphs within 5 seconds for users.

Displaying critical event notifications generated in the backend of the Website Subsystem need to occur quickly due to their importance in notifying the user of critical events. The event will need to be visible in under two seconds from the point of creation and the notification should not be cleared until the appropriate user chooses to clear it. We consider this to go hand-in-hand with the backend side of the Website Subsystem.

4.5 Orchestra Subsystem Requirements

The Orchestra Subsystem has several functional requirements necessary to work properly. This Subsystem needs to be able to connect to any number of simulator and reader subsystems over virtualized com ports for lab environment stress testing.

There are two non-functional requirements which are essentially simpler tests which will prove our functional requirements will work. The first non functional requirement is to be able to connect to at least one simulator subsystem over RS-232 protocol on a virtual com port via the Null-modem emulator mentioned above. At least one established connection is the minimum requirement by our clients request.

Our second non-functional requirement is for the simulator to be able to connect to the reader subsystem in no more than five seconds per connection. The purpose of this requirement is to make sure our system works as an individual SWAPR device and also to stress test our system before adapting the reader software to a large-scale implementation with more SWAPR devices.

5. Environmental Requirements

We will be working in a Microsoft Windows environment to avoid any complication with our other tools in our environment. The Windows Operating System (OS) is used by our client and necessary for implementation with their systems as well. Using this OS will allow easy implementation of Microsoft Visual Studio, the Integrated Development Environment (IDE) and interface we have been required to use by our client for creating our code base.

Visual Studio is needed, because it has a range of features useful to our team. Without Visual Studio we would face more complexity in finding and using dependencies to meet the specifications of our client. Two of these features are the .NET 5 and Blazor Server products. .NET 5 and Blazor Server are two Microsoft products that are needed by our team for our systems development and easily integratable with Visual Studio in a Windows environment. The purpose of .NET and Blazor Server in our development is to provide our team with the capability to create our systems website and database.

The usage of Visual Studio with .NET and Blazor server defines an ecosystem of development tools with a built-in run-time environment for debugging and running tests. Our client has also requested our team use .NET 5 with Visual Studio. .NET 5 includes an extensive library of C# functions that can be used in our websites backend and frontend creation, database entry creation and data transmission through TCP client connection in communication with the reader software.

The Microsoft Blazor Server is the best tool for building our website's architecture because the Blazor Server is useful for handling live data dashboards. The Blazor Server provides faster load times compared to Blazor Web Assembly, because the Blazor server allows users to download files from the site as needed as opposed to all at once.

The C# programming language is an available development language for Visual Studio. We need to use C# not only because it is available to use in Visual Studio, but because C# is a main language supported and Visual Studio provides an expansive library for C#. C# is a high-level, easy to implement, and dependable language we can use to create all of the programmatic functionality in our system. Our team has been able to establish a C# based approach to all of our programmatic problems through researching Visual Studio, .NET 5, and Blazor Server documentation.

6. Potential Risks

Our project will save GDMS time and money when it comes to monitoring and maintenance of RFFs and SWAPR devices. It will help identify RF interference from equipment failure, prevent them from accidentally sending a technician to a location when there is bad weather, and help GDMS schedule tower maintenance. This means lower labor costs for those false calls, and less time wasted. Though it is not life-threatening without our solution in place, there are a few potential risks of our project if everything does not go as planned. We will discuss the severity of these risks and their relevance.

The first risk we will face is if our device reports inaccurate weather information caused by malfunction in weather recording hardware. If our code improperly reports that it is sunny outside, but the RFF tower is in the midst of a rainstorm it means there is an issue with the SWAPR device. In the context of maintenance of RFFs and SWAPRs, a technician needs to know what parts they need to fix on the SWAPR device, and inaccurate data in acceptable ranges would make that more difficult to diagnose. However, if the data is inaccurate and in unacceptable ranges then we can determine exactly what part is malfunctioning. Our system will detect large storms which may cause damage to the site. On the other hand, if the information reports there is no storm while a hurricane is on top of the RFF site then GDMS will have a longer down time since they didn't schedule a technician to visit the site sooner.

The second risk we could encounter is inaccurate antenna power data. This would be a minor inconvenience to the GDMS, as they already have another way of getting the operational status of the antennas on an RFF site. This is not a huge risk to GDMS since they have been doing repairs without prior knowledge since their inception.

Our last risk that may be encountered is if the SWAPR device does not send weather data to our database at all. The network might get so congested that TCP is waiting to send, or something could have happened to a SWAPR device or the Reader Subsystem. The power could have gone out or a cable could have gotten loose. We have decided to use TCP for reliable data transmission from our database to our web server to ensure that packets are not lost due unforeseen circumstances like corrupted or lost data, but a problem with the equipment at the RFF site. If a hardware malfunction occurs, then GDMS will have to go back to repairing the tower where they may encounter bad weather. This is how they have operated up to this point in time, therefore it is not a huge risk for them to face.

Overall, our project has several risks associated with it, but only within the scope of our product and as a working prototype. The US Coast Guard will not implement our system until they go through fully exhaustive testing and complete the development phase. If our prototype works perfectly as planned, then GDMS will save both time and money. If our system does not meet the requirements our client needs, then GDMS will proceed with their business unchanged by our project complications.. The Coast Guard has been using the Rescue21 system without SWAPR devices for years with great success, so there is no doubt it is performing adequately. We want to help them save on labor costs, time spent, and overall, make their lives easier.

- (above) Risk 1: falsely reported weather status
    - Relevance:
        - Maintenance sent in bad conditions
        - Maintenance unprepared
        - Unexpected expenditures due to weather damage since it wouldn't be known
- Risk 2: Falsely reported antenna power
    - Relevance:
        - Kinda the same stuff
        - Guy gets sent out/same thing not that big of risk
- Risk 3: Database access to unauthorized parties
    - Relevance:
        - Access to secret information
        - Malicious intent/ possibly
- Risk 4: SWAPR doesn't send the data to database
    - Relevance:
        - Network bogged down and can't send
        - No response
        - Someone has to be sent out(once again)
- Overall if it doesn't work/wrapping it up
    - Goes back to how it was
        - Same man hours etc. money spent
        - EXCEPT for malicious entry of Database

7. Project Plan

In our Gantt Chart, we listed all of our assignments from now to the end of the semester and the start and end times for each of them. To provide a brief overview of the chart, we first need to complete our Requirements Document Draft by November 15th. Our Design Review Presentation is also due on November 15th. After that, we will put together a Project Info Mini Video, due December 3rd. After that, we plan on creating a Tech Demo “Flight Plan” to begin planning our Tech Demo and later develop this with our team and mentor. This Flight Plan will be complete by late November. While completing these other tasks, we will continue to complete and update our Team Website. By the time of the Final Exam at the end of the semester, around December 3rd, we plan to complete the final draft of the Requirements Document. Our client requested for us to have our project planned out in pseudo code so they can understand how we intend to complete the project and give us feedback. Our pseudo code is due to our client by December 3rd. This plan will help us complete our tasks within our timeframe, and set a precedent for consistent progress throughout the end of this semester. Our client will be having a meeting with the Coast Guard sometime after this semester or next semester which is why we must have our pseudo code completed. If the Coast Guard comes at the end of this semester then General Dynamics will use our presentations and or our pseudo code to inform the Coast Guard of the project, our progress, and to try and get a contract to implement our project into the Rescue21 system

Insert Gantt Chart Here

8. Conclusion

To wrap everything up, the Coast Guard already knows how to detect if a RFF tower is operational but have no way to detect weather on individual sites. This leads to the problem that The Coast Guard does not have a way to tell the cause of RF signal interference. They may assume hardware malfunction, when the signal interference is actually caused by rain. Tower climb scheduling has been somewhat unpredictable since bad weather that would inhibit a climb is not recorded. A technician may run into bad weather when they are on a repair job which means they would not be able to climb the site's tower. The tower would have longer service outages because they have no way of determining weather, since a technician has to be sent out while the tower is down. This leads to more time being wasted and time being spent. Our solution will supply the Coast Guard with weather information which will allow them to schedule repairs in advance when potential storm damage might occur and reduce possible outage time. Our solution will help with detecting weather caused RF signal interference, and will help with scheduling tower climbs for days with clear skies.

Previously in this document, we discussed the problems we currently face, and our five subsystem solution comprised of the: Simulator Subsystem, Reader Subsystem, Database Subsystem, Website Subsystem, and Orchestra Subsystem. This allows understanding of the individual subsystems and possible ways of implementing them. We listed the functional and non-functional requirements for each of these subsystems, and have a clear idea of the essential traits for each subsystem as well as traits that are nice, but unnecessary for the project to function. We completed demonstrations of a few of the technologies that will be used in our Tech Feasibility Analysis. Our team has a much firmer grasp on our project's potential risks and how to possibly for multiple SWAPR devices to be simulated.

Our progression on this project is reflected in the project plan. We gained valuable insight on our project and learned that our project is making sure that GDMS reduces the downtime of the Rescue21 System. If RF signal interference is detected and it's raining on the site, then the technician isn't sent. If it is due to a hardware problem, they are sent. It does not drastically change their current process, but it does simplify their ability to schedule tower climbs, and know the weather at each of these locations. Though no pseudo code or demos specific to our project have been completed at this time, we are confident that we will be able to use our research and have a plan for pseudo code by the end of the semester and a fully functional prototype by the end of the year. Having completed these prior steps and documentation, we know our project requirements, and will be able to tell when our project is finished and be able to test to ensure our project is fully complete.

8. Glossaries and Appendices

M.V.P.- Minimum Viable Product

RFF- Remote-Fixed-Facility. These are facilities equipped with towers and direction finding antennas for search and rescue and related missions for the US Coast Guard

RF - Radio Frequency

Baud- a unit of transmission speed equal to the number of times a signal changes state per second. For one baud is equivalent to one bit per second.

SWAPR- Site Weather and Power Recorder. This device was created by last semester's capstone group, Oscillator 3000 to record weather and power information.

VHF- Very high frequency (VHF) is the ITU designation for the range of radio frequency electromagnetic waves (radio waves) from 30 to 300 megahertz