Feature List

The following is a list of the features which we believe are either necessary, or which we would like to have within our application. Below is a justification as to why we chose these features and their necessity.

1. The server must process incoming page requests and provide correct responses. Users must be routed to different pages in our simulation based on what they intend to view.
2. Create and manage multiple virtual networks with:
    a. Ability to add, move, or remove any virtual devices from any networks
    b. Ability to allow or deny communication between members of different networks.
3. The server is able to store, update, and handle unique devices within this simulation. Each device is able to be manipulated by the user which owns it and the administrator.
    .
    a. Can connect and disconnect from pre-existing virtual networks within its simulation
    b. Can create, and manage virtual networks which it has created.
    c. Can run HTML5 applications which can communicate with other virtual devices running the same application
4. Possesses a robust token system in order to selectively allow devices to participate in a simulation.
5. Token propagation to different users via Email.
6. Be able to view a visual representation of the network topology, which can be interacted with in order to create partitions between networks, and move devices between networks.
7. Keeps track of history of events which have occurred in the simulation, and is able to display the history at any given time-stamp. This includes displaying the topology of the simulation at that time, and all events which have occurred up to this time.
8. Be able to upload replicated data types to the server which users can access using HTML5 applications.
9. Register HTML5 applications with a simulation session.
10. Propagate the HTML5 applications to specific devices within the simulation.
11. Automatic execution of test scripts.
12. Our application must be able to handle and manage multiple different simulations, which can be created by users of our application.
13. Compatible with all browsers

14. Token propagation via multiple methods, which the administrator may select from.

Feature List Justification

We present justification behind the ordering of the features in the feature list. This includes explanations of why these features are risky and their applications to the three Q's or architecture.

1. Server features such as handling incoming page requests and providing correct responses is a very risky feature to our architecture. It is the foundation of our program, as it handles channelling data to and from users, therefore it is essential to our system. Only a single member of our team is familiar with node-js which we will be using to implement the server, therefore we are unsure of how to implement it. Finally, we are not entirely sure what the server should do and how to handle it. Therefore this satisfies the three Q's of architecture, and this feature is very risky and should be handled early by architecture.

2. Create and manage multiple virtual networks. This is a very essential feature to our system, as it is the part of the core of our program and most of the other features would be nonexistent or useless without it. We believe that we do not understand this feature in its entirety. We do not know how the networks will have to behave in the future, and what sort of communication will be required. As we do not entirely understand this feature, we do not understand how to implement it. Therefore this feature satisfies the 3 Q's of architecture and is very risky. Thus, we handle it with architecture early on and place it high on the priority of the feature list.

3. Create virtual mobile devices and allow connection of real mobile devices to system. This is essential to our system, but we believe we understand how to handle the implementation and understand how it should work. This satisfies one of the three Q's of architecture, and therefore is not as risky as the ones above but should be still handled by architecture and as early as possible, as it is one of the most essential parts of our system.

4. Having a robust token system is critical to our system, it allows for devices to access a simulation, and without it none of the other required features for the system would be possible to implement. Therefore this is an essential feature to our system. This is a challenging feature, as actual devices as well as simulated devices must be able to have tokens, the tokens must be unique, and can only be used once. Therefore, we do not entirely understand what this means and how this should be implemented. Thus, this feature satisfies the 3 Q's of architecture and is at high risk to our design, so we should tackle

it with architecture as early as possible.

5. It is necessary for our tokens to be propagated to devices which should be added to our system. The most straight-forward, sought after, and secure way we believe to do this is with email. As token propagation is required for any device to join a network, it is essential to our system. We understand this feature, but are unsure of entirely how to implement this feature. Therefore this satisfies 2 out of 3 of the three Q's of architecture, but as it is such a central feature to our system, we place it high on our features list.

6. Viewing network topology. The administrator of the simulation should be able to interact with and manipulate a visual representation of the devices and networks within a simulation. The changes done in this should affect the topology of the simulation as a whole. This is a feature which the client specifically requested, and thus is very important to handle. As well, it is a very complicated feature, which we do not understand. Thus, it is very risky, and should be handled with architecture early on.

7. Event History Viewing. This is a very complicated feature. It involves keeping track of all events which have occurred on our simulation thus far. Not only this, but it requires data management, as this could potentially require storing a lot of data. Thus, this is a very risky feature. On top of this, it is essential to our system, as it is a feature which the client specifically asked for. Therefore this is very important to handle with architecture.

8. Test the performance of given HTML5 applications. This is essential to our system as it is in the given vision statement of the project. We are unsure of what it means; as we do not know how the system should interact with HTML5 applications and what should be done. As well, we do not know how to implement this, as we do not know how these applications are to be sent or run. Therefore, this satisfies the three Q's of architecture, and this feature is very risky and should be handled early by architecture.

9. Uploading and interacting with replicated data types. This is a crucial feature, which the client specified that they wanted since iteration 1. Therefore, this is a very risky feature, as it is necessary. As well, replicated data types are not a feature which we have encountered before and so we must understand them in order to implement them. Thus, it satisfies all three Q's of architecture and must be handled early.

10. Register HTML5 applications with a simulation session which any participating virtual device may run. This feature is essential to our system as it is stated in the vision document of the project and therefore must be included in the program. We are unsure of exactly how to implement this and how this should be done. Therefore this satisfies the three Q's or architecture,

and is quite risky. Therefore we handle this with architecture.

11. The person who uploaded the application should be able to decide which devices are able to access it. This is not as essential as the ability to have HTML 5 applications, but is still very important. Thus, it is still risky and should be handled with design.

12. Automatic execution of test scripts. This is essential to our system as it is given in the vision statement of the project. We are not exactly sure what this means, as we do not know what scripts the customer would want run and how to execute them. Therefore we are also unsure of how to implement. This satisfies the three Q's of architecture and is therefore a very risky feature and should be handled by architecture early on.

13. Track and manage multiple network simulations is essential to our system, as it is one of the required features to the system, although we do not believe that it is of utmost importance and therefore it is significantly low on the features list. We believe that we understand what this concept means, but we are unsure of how this should be handled and how to implement it, especially along the lines of how various simulations should be accessed. Therefore it satisfies two of the 3 Q's of architecture, and should be handled by architecture and design. For this feature, our belief is that it would be simplest to have a website which allows you to select the simulation you wish to connect to and then route you to that specific simulation.

14. Token propagation via multiple methods this is not essential to our system, although it would provide added flexibility and allow certain users not willing to supply their email addresses, or without an email address to access our system. It would as well allow for administrators to handle different kinds of simulations with more or less security (for instance, a user being assigned a token when accessing the website is much less secure than email authentication). We are unsure of how to handle this, but we understand the feature. Therefore this satisfies 1 of the three Q's of architecture, and so is low on our list of features, although we still believe it is an important feature.

15. Compatible with all browsers. This feature is not very risky. It is not "essential to our system", we know how to handle it, and we know how to implement it. Therefore it is not pertinent to include architecture for this feature.