ASHESI UNIVERSITY COLLEGE

SIMULATED AVATAR-BASED ENVIRONMENT

ALBERT KOFI MENSAH-ANSAH

2013
Applied Project

I hereby declare that this dissertation is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature: ………………………………………………………………

Candidate's Name: ……………………………………………………………..

Date: ……………………………………………………………..

I hereby declare that the preparation and presentation of the dissertation were supervised in accordance with the guidelines on supervision of dissertation laid down by Ashesi University College.

Supervisor's Signature: ……………………………………………………………………..

Supervisor's Name: ……………………………………………………………………..

Date: ………………………………………………………………………

ASHESI UNIVERSITY COLLEGE

Simulated Avatar-Based Environment with Chat and Resource Sharing Capabilities

By

Albert Kofi Mensah-Ansah

Dissertation submitted to the Department of

Computer Science,

Ashesi University College

In partial fulfillment of Science degree in Computer

Science

June 2012

Acknowledgement

I like to thank my supervisor for his help and suggestions during this project helping me accomplish the objective of the project. Also I thank Dr. Nathan Amanquah for his contribution in terms of technical advice and also lending me switch to use to the project experiment. I thank the IT office for help in providing me with network cables and other networking resources to be used for the project experiments. I also want to thank all the lecturers especially the ones that saw me programming. And lastly I thank my friends who encouraged me during the project and also offered to be test subjects during the development stages.

Abstract

In our world today, we see the evolution of the communication paradigms all with the aim of ensuring a quick and quality way of passing information from one subject to another in different locations. Currently, with the advent of the internet, many efforts are being made to make communication socially immersive; in a way that the person you are communicating with feels naturally close to you. With the inclusion of visual data like pictures, emoticons and videos, people are able to relive past events they missed and are able to have a more realistic sense of how that party, dance, wedding or funeral went. More over the addition of these extra data apart from the traditional text and voice, are able to convey the unspoken words; the emotions, after all a picture can say a 1000 words. This report presents a revolutionary communication product that is supposed to take communication to a whole new level; creating the opportunity for people to accomplish some task otherwise unimaginable on the traditional communication systems.

Contents

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 - Background Information:

Over the years, we have seen the progress and evolution of long distance communication over radio, cable etc. Now we are exposed to text based chat systems like Yahoo messenger[1], Facebook[2] chat etc. There are also some tools like Skype[3] that offer voice communication and even offer the ability to share files. These tools all aid in offering some form of communication from one space to another. However, it appears that the more functionality and diverse these existing platforms get, the harder it is to use. Using Skype as an example, it has the ability to make voice/video calls, text based chat, share files, add contacts etc. For a user who has been introduced to Skype the first time, he/she would have to learn where the options are for accessing all these features. In trying to find a system that would have as much features as there are in Skype but would also be intuitive to any kind of user, I started exploring Massively Multiplayer Online Role Playing (MMORPG) Systems. One kind of MMORPG is Final Fantasy[4] In this system, the communication features that exist in the traditional chat system like Facebook, Skype etc, are integrated and intuitively accessed by the user. For example, if a player wants to interact with some other player, he knows intuitively that if that other player is not geographically (in-game) close, that interaction would not be possible. So once a player's avatar is seen in other players' screen, it presupposes

[1] http://messenger.yahoo.com/
[2] https://www.facebook.com
[3] http://www.skype.com/en/
[4] http://www.techtree.com

that, that particular player is online and the users of the system would not have to think much to ascertain whether or not a player is online or not.

Figure 1 (Example of an MMORPG, Final Fantasy)

In my project, I intend to use the advantages of using an interactive virtual environment to improve the means of communication on the Ashesi University Campus.

The idea of including graphic detail to add information to real-time communication between people in different spaces has been worked on by a number of companies like Microsoft with the introduction of Microsoft Comic chat[5]. This was actually an **Internet Relay Chat (IRC)** based chat system. Their idea was to generate images synonymous to what appear most comic books as the chat progresses. The images generated depict to some extent the emotions of the user at any point in time. This idea

[5] http://en.wikipedia.org/wiki/Microsoft_Comic_Chat

simulates the user experience and immersion gain from reading a physical comic book.

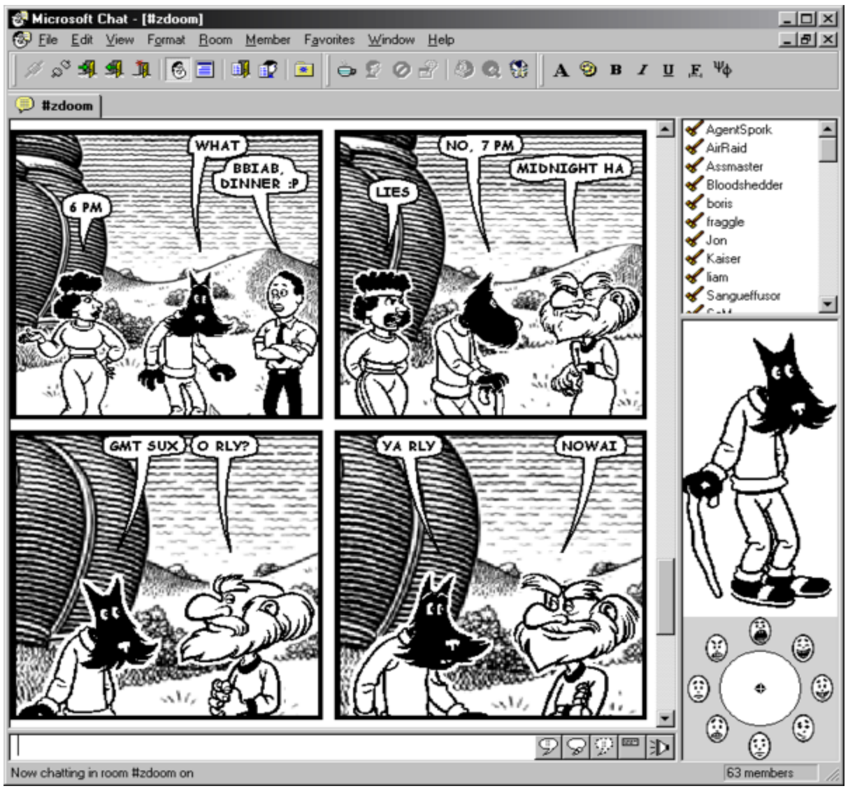


Figure 2(Microsoft Comic Chat, 1996)

Another communication system supporting multiuser usage in a virtual world is Active Worlds[6]. Active Worlds is an online 3D avatar based system that consists of multiple worlds and so depending on your preference, you choose where you would like to be. In addition to Active Worlds, SecondLife[7] also has an implementation of communication virtual worlds with the usage of avatars.

[6] http://www.activeworlds.com
[7] http://secondlife.com

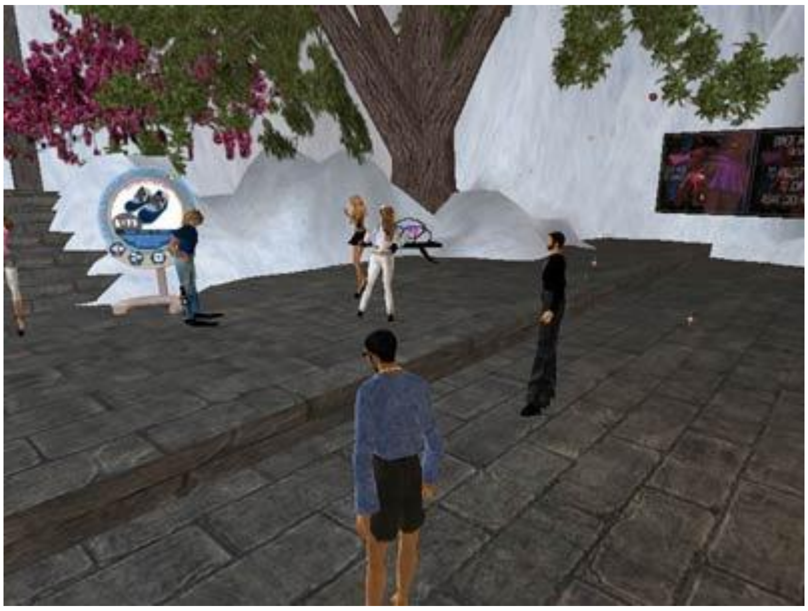
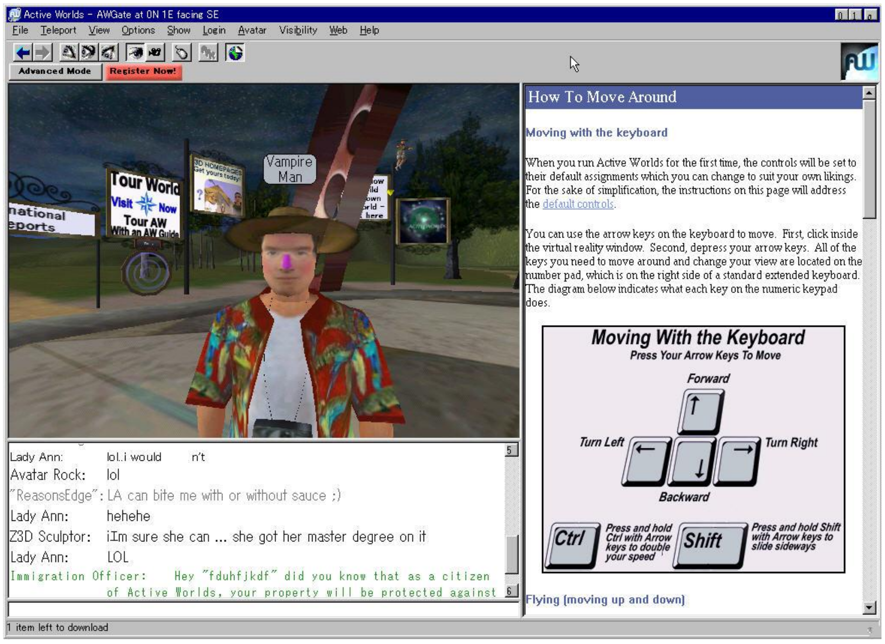


Figure 3(Second Life)

Other implementations of avatar-based chat systems are seen in video and computer games where players are to use their avatars to accomplish a mission collaboratively with other players connected by means of either a voice or text based chat.


Active Worlds - AWGate at 0N 1E facing SE
File Teleport View Options Show Login Avatar Visibility Web Help
Advanced Mode
Register Now!
Tour World Visit Now Tour AW With an AW Guide
national eports
3D HOMEPAGES Get yours today!
Vampire Man
Moving With the Keyboard
Press Your Arrow Keys To Move
Forward
Turn Left
Turn Right
Backward
Ctrl
Press and hold Ctrl with Arrow keys to double your speed
Shift
Press and hold Shift with Arrow keys to slide sideways
1 item left to download

```
Lady Ann:        lol..i would      n't
Avatar Rock:     lol
"ReasonsEdge": LA can bite me with or without sauce ;)
Lady Ann:        hehehe
Z3D Sculptor:    iIm sure she can ... she got her master degree on it
Lady Ann:        LOL
Immigration Officer:    Hey "fduhfjkdf" did you know that as a citizen
                 of Active Worlds, your property will be protected against
```

How To Move Around

Moving with the keyboard

When you run Active Worlds for the first time, the controls will be set to their default assignments which you can change to suit your own likings. For the sake of simplification, the instructions on this page will address the default controls.

You can use the arrow keys on the keyboard to move. First, click inside the virtual reality window. Second, depress your arrow keys. All of the keys you need to move around and change your view are located on the number pad, which is on the right side of a standard extended keyboard. The diagram below indicates what each key on the numeric keypad does.

Flying (moving up and down)

Figure 4(Active Worlds UI)

In Ashesi, emphasis is placed on being able to work in teams or groups and as part of academic activities, there are periods for group meetings. Currently, there are not enough venues where group meetings could be held without disturbance. Apart from the seminar room, there is no dedicated meeting venue where one can have a meeting undisturbed. Also, most group meetings involve the use of laptops which at some point need to be charged but the available outdoor spaces for meetings do not have power outlets and therefore the quality of the meeting is sub-standard not to include the disturbance from other disturbing people the meeting since it is an outdoor venue.

Also, Ashesi has learning lab facilities that help students with difficulties in writing, Mathematics and programming. Currently, these services run in the same venue and share the same resources (the white board, space etc) and I think that is not conducive enough for effective tutoring.

The usage of existing voice and text chat platform can be used to improve the situation; however these tools solely depend on internet connectivity.

1.2 - Project Overview:

In trying to remedy the problem of less conducive meeting spaces and others as discussed above, I thought of a chat system that would possess the features of having group meeting, appointments, casual conversations etc. However, just a text based chat system would be overly complex with respect to the user interface and what I want my users to be able to achieve. For example, in trying to establish communication between two people, a user may perhaps click a menu button like, "Chat" and inside that menu will show the various chat types like "Chat with everyone" or "Chat with a specific person". Then if the user clicks on "Chat with a specific person", another menu shows a search bar for the user to search for this specific person and from here the chat can begin. These steps for the user may be too much so making it Avatar-based would make certain actions that are supposed to be performed intuitive for the user to accomplish.

The Simulated Avatar-Based Chat environment is going to feature a virtualized model of the Ashesi Campus.

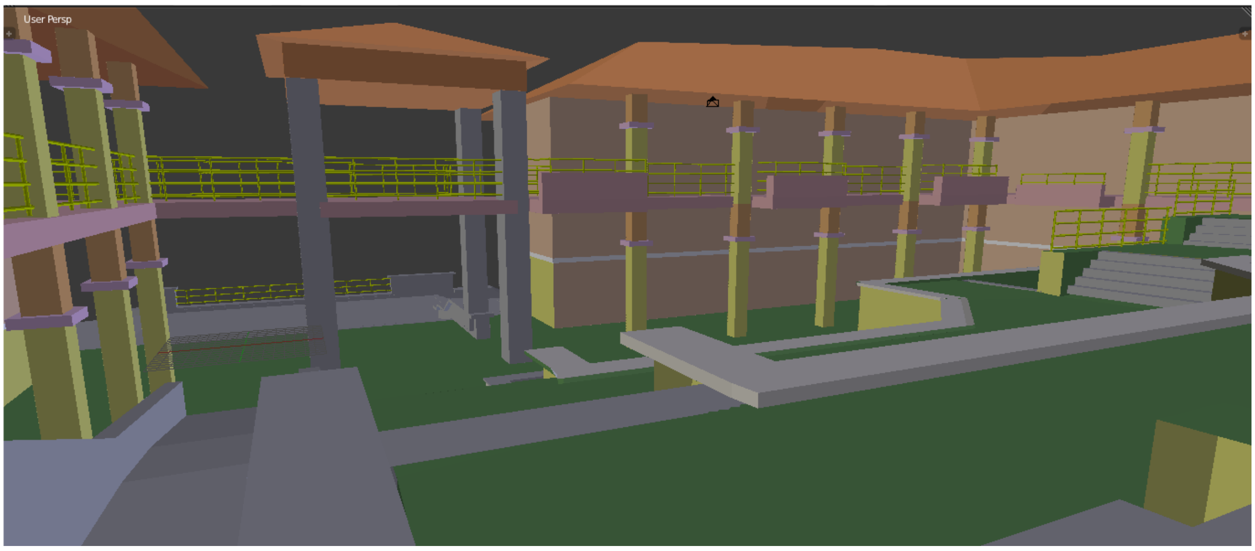


Figure 5 (Model of the Ashesi Campus)

This is the environment where all the avatars would exist once they are logged in. There is going be some form of database that keeps information of the avatar's information. This application is actually limited to users who have names on the school server. Any other person who wants to use must come in as a visitor like someone who wants to take a tour in Ashesi. The architecture of the system is going to involve an *authoritative*[8] server that would be responsible for the world logic and synchronization between the clients that contextually need to be updated about the events that occur in the world.

1.3 - Motivation:

The evolution of the way users are made to interact with their computers is all bent on making the system usable and equally useful. We are witnessing a revolution from the traditional keyboard-mouse way of computer interaction to the touches and taps and the now emerging eye tracking and brain wave sensitivity. The introduction of these ways of interaction is to make a user of such devices find the usability second nature; that every guess of how to use the system would be right without the need for external help. In that regard, this avatar based system falls in this category of intuitive systems. Also this system is expandable and in the future it could have more features that would increase the immersion of the user like ordering food from cafeteria, connecting to other third party APIs that would enable buying and selling etc. For example, if the user wants to buy airtime for his/her phone, all he/she has to do is to walk to the convenient shop virtually, and make the purchase and his/her airtime is topped up. This means in the future this system will be a world that the user can stay in and accomplish anything imaginable.

[8] For more information about authoritative servers check http://docs.unity.com/Components/net-HighLevelOverview.html

1.4 - Expected Results:

At the end of the project, it is expected that it would be possible for people in the Ashesi community to connect to the system and meet up in the virtual world for a meeting among other things. It would also have the ability to set appointments with other Avatars like a student and lecturer. Also for people who want to visit Ashesi University College and look around, they would have the opportunity to do so with this project.

1.5 - Potential Benefit:

In the future, Ashesi is going to expand and movement from one location to another would be tedious. If a meeting scheduled to be held at a certain venue that is quite inconvenient for you due to distance, this system can make you available in that time and space virtually. Also, you are sure the meeting minutes would be as detailed as possible because, it would be possible to save the transcript of the chat. Since this system would not be heavily dependent on the internet, communication is still ensured even if the internet is down. Moreover, people who do not live in Ghana would have the chance to see Ashesi and even get the chance to interact with some of the people in the community.

CHAPTER 2: DESIGN

Outlined in this chapter gives a high level overview of how the project SABE is going to end up. Listed here are the set of requirement of the project that would be accomplished after the project is completed.

2.1 - FUNCTIONAL REQUIREMENTS

2.1.1 - Virtual Meetings

1.) Users shall be able to create/edit virtual meeting spaces where they can add people to the meeting, specify dress-code, the venue, specify the time and date for meeting and the agenda for the meeting.

2.) Other users would be reminded of an impending meeting; that is if they have been invited.

3.) Users would be able to share Powerpoint files and actually see it as an image projection in the environment similar to using a projector.

4.) Users should be able to get access to the chat history of any form of meeting held.

2.1.2 - User Accounts

5.) Logging in for the first time would require an Ashesi username and password. After this, the user name and ID is copied to my database and the user is made to set a custom password for the system.

6.) The system should be able allow non-Ashesi members to login as guests or visitors with restricted access to what they are allowed to do.

2.1.3 - General Features

7.) The avatars in the world would have the ability to teleport to whichever part of the environment they want to be.

8.) Users should be able to customize their avatar in terms of appearance like clothes color, hair style etc.

9.) The environment would have intuitive features like; moving closer to an avatar would start a chat with that person. If there are more than two people chatting, moving further closer to one of the avatars would make the user *whisper* to the corresponding person (that is the message sent will only be sent to that one person even though there are other people involved in the chat).

10.) There would different chat modes,

   a. **Shout**: This is where whatever a user says is broadcasted across everyone connected the SABE World Server

   b. **Conversation**: This is where a user would be talking to someone he/she is geographically (virtually) close to.

   c. **Whisper**: This is the state where one’s avatar is very close to another avatar. In this case, the chat will only be between these two parties.

11.) The world would have a mini map that would give the user an idea of his/her current location and the location of other people in the world

2.2 - NON - FUNCTIONAL REQUIREMENTS

1.) Any form of Meeting can hold up to 32 attendants.

2.) The virtual world can hold up to only 32 connected computers.

3.) Before a user can interact with people in the world, he/she must first belong to the Ashesi Community and also have a SABE user account.

4.) A user cannot connect to two meetings at the same time.

5.) In order to be able to run project SABE your web browser must have the Unity3D Webplayer installed.

6.) The user's screen should support a screen resolution of at least 800 x 600 for optimum performance

7.) The system should be robust.

8.) The system must be easy and intuitive to use.

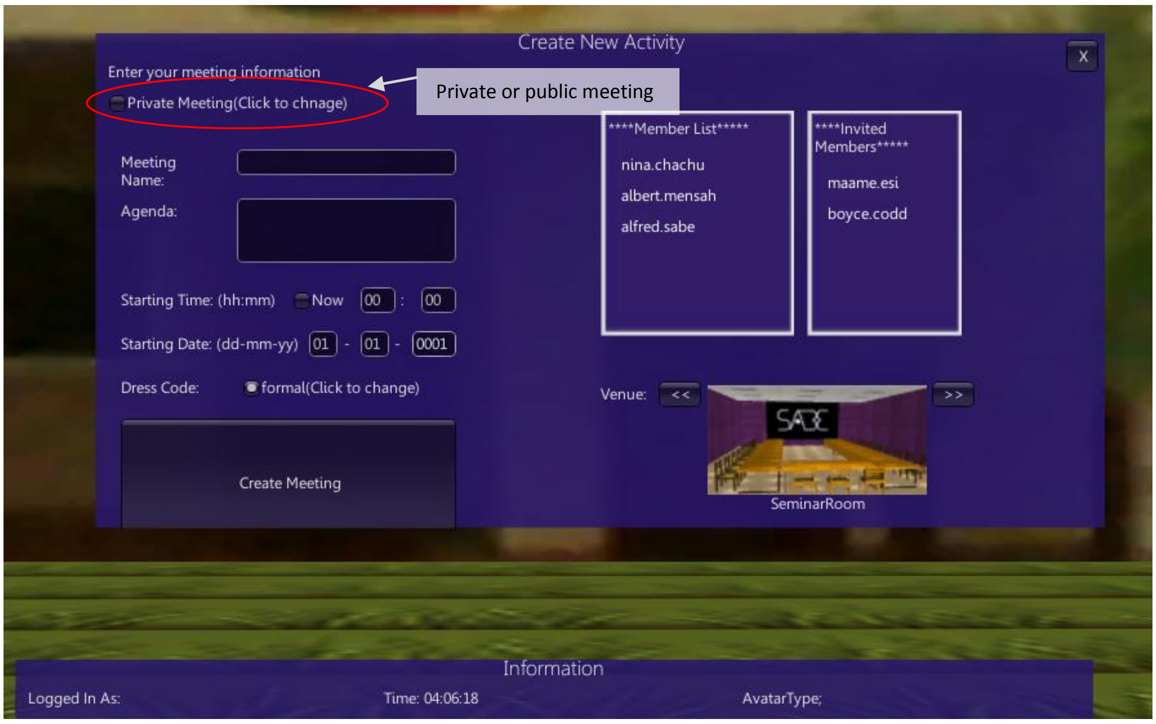
9.) The system must make sure the database is fed with up to date information about what is going on in the world.

2.3 - Scenario

2.3.1 - Setting Up a meeting

Adwoa is a sophomore student at Ashesi University College. She has been given a design project work to do with group members. At the same time, Adwoa, is boiling water and she has to be there to keep watch. She happens to be connected to the school's network. These are the steps she goes through to setup a SABE meeting.

1.) She immediately connects to the SABE world server and asks her friends to also login to SABE world server.
2.) She wants to create a group meeting now so she goes on to click "Create Activity" leaving the now option checked.
3.) The create activity window gives Adwoa the option to choose a meeting or a public event. Since this is a private group meeting, she chooses the meeting option and starts filling in the details
4.) She clicks on the Create Meeting button. She is immediately sent to the venue she chose to have the meeting. Her friends who have been invited are notified using the Activity window.
5.) This marks successfully setting up a meeting.



Create New Activity

X

Enter your meeting information

Private Meeting(Click to chnage)

Private or public meeting

Meeting Name:

Agenda:

Starting Time: (hh:mm) Now 00 : 00

Starting Date: (dd-mm-yy) 01 - 01 - 0001

Dress Code: formal(Click to change)

Create Meeting

****Member List*****

nina.chachu

albert.mensah

alfred.sabe

****Invited Members*****

maame.esi

boyce.codd

Venue: << >>

SeminarRoom

Information

Logged In As:

Time: 04:06:18

AvatarType;

Figure 6(Setting Up a Meeting)

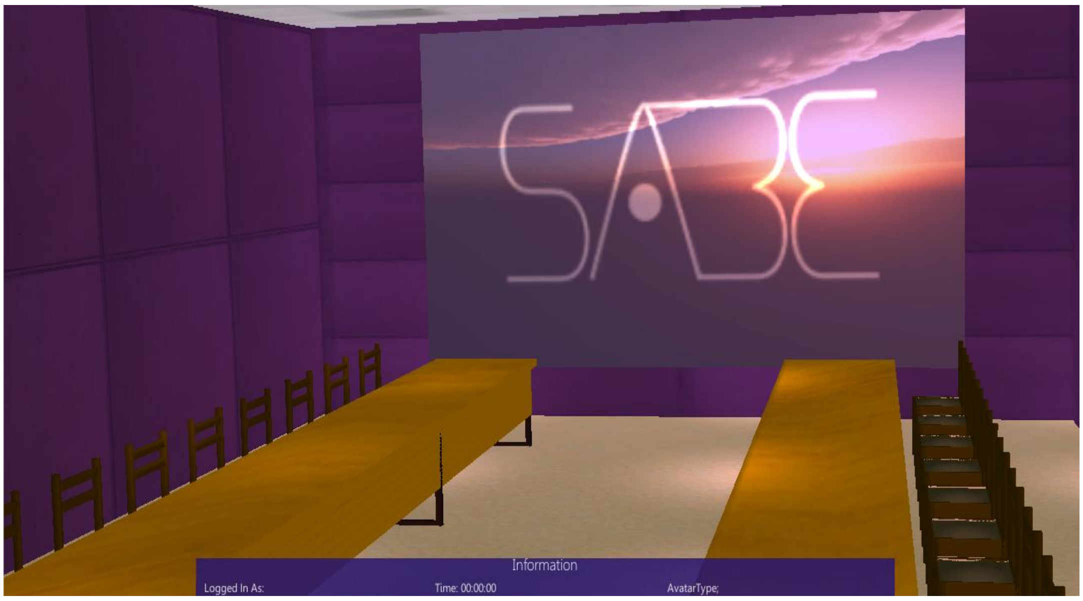


Figure 7(Screen Shot of Meeting, Seminar room)

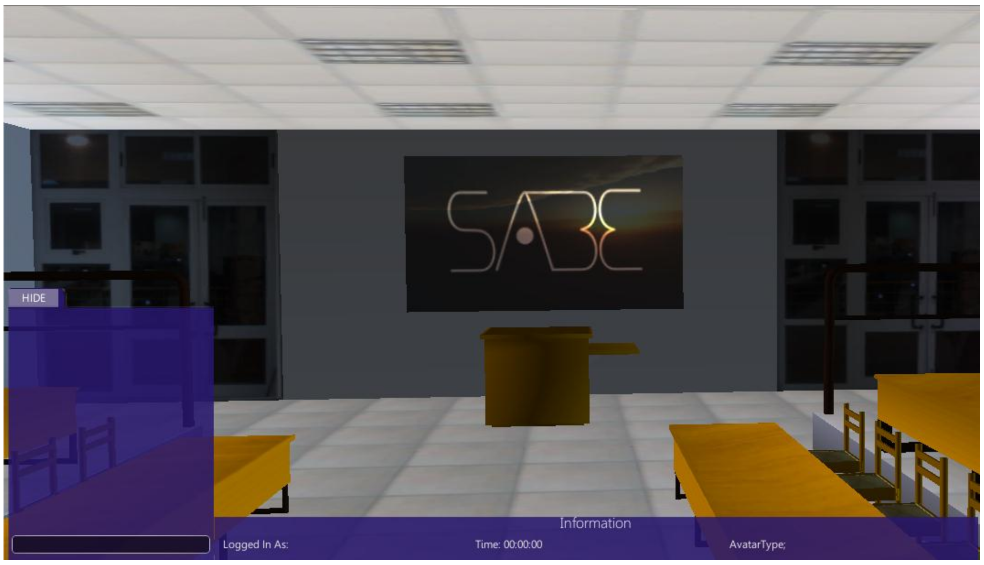


Figure 8(Second Meeting Venue)

2.3.2 - SABE Sign up

It is Adwoa's first time of trying out SABE. She saw met a friend on Facebook who encouraged her to try SABE. So

1.) Adwoa logs in with her Ashesi username and password.

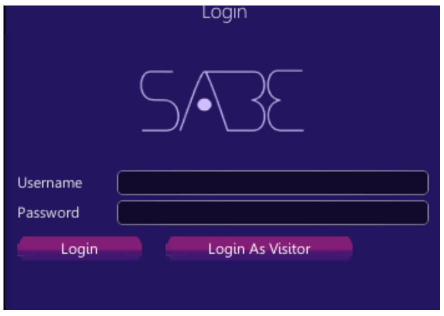


Figure 9(Login Form)

2.) Since it is her first time of login, she sees a screen that asks for Adwoa's information being her preferred screen name, her interests and something about her self

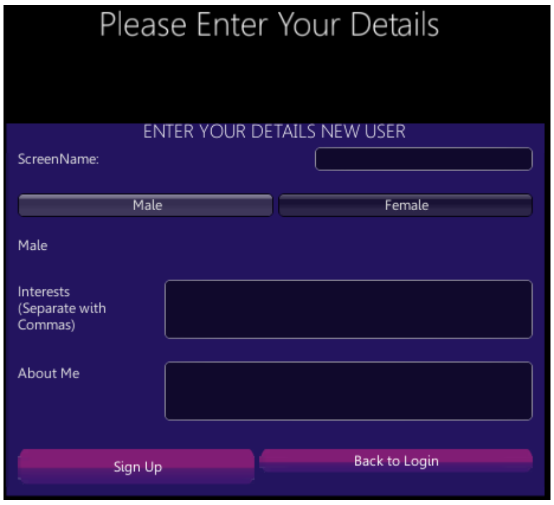

Please Enter Your Details
ENTER YOUR DETAILS NEW USER
ScreenName:
Male
Female
Male
Interests (Separate with Commas)
About Me
Sign Up
Back to Login

Figure 10(Entering user Details)

3.) After she is done, she is taken to a dressing room to modify her avatar.

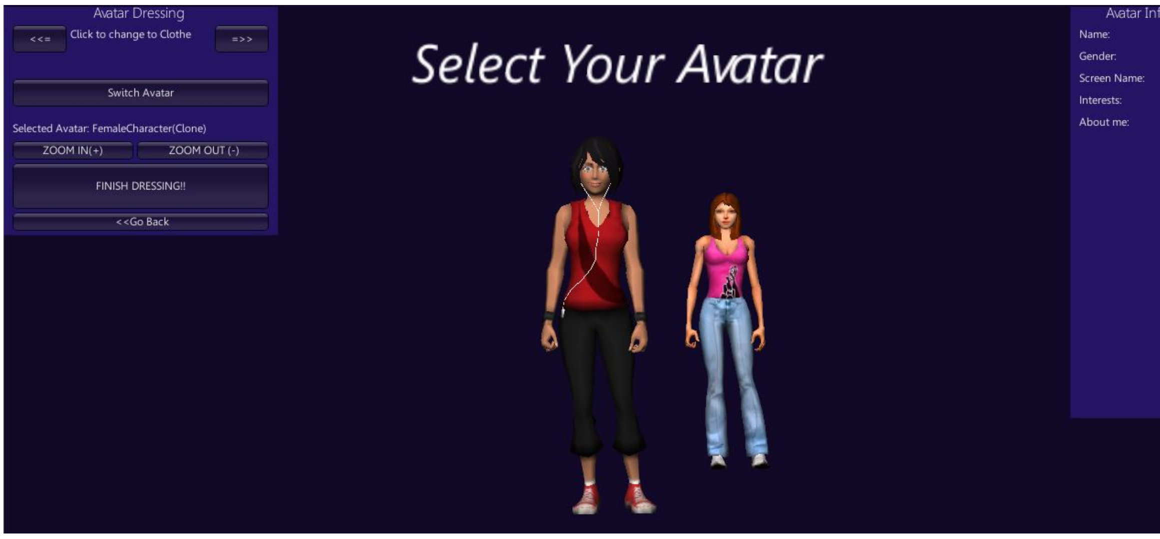
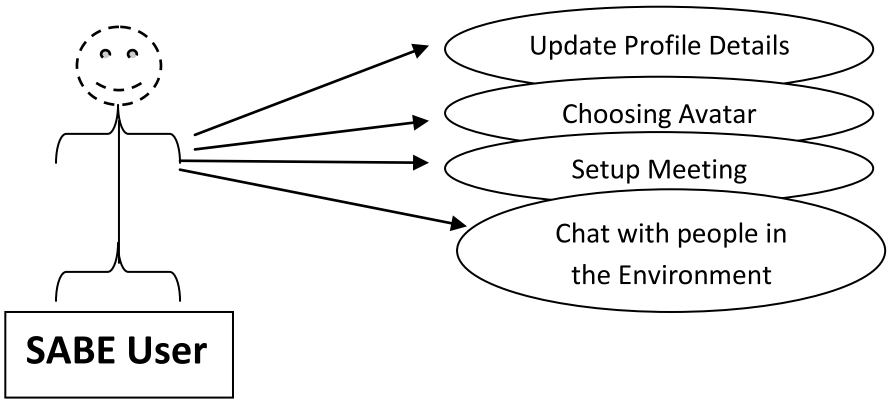


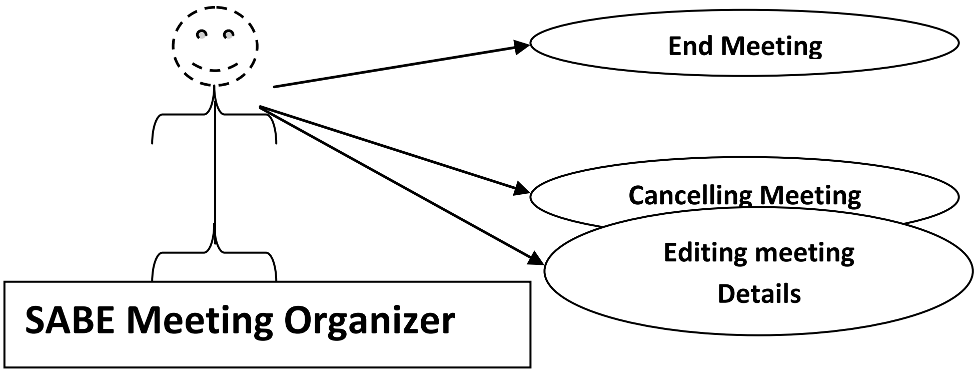
Figure 11(Choosing Your Avatar)

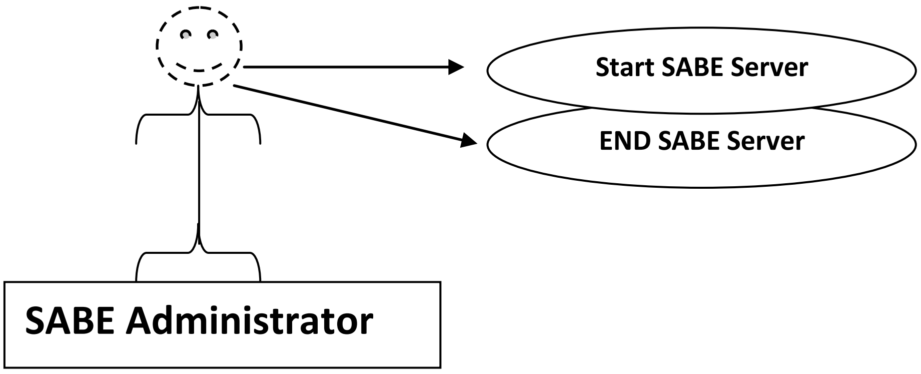
4.) From there, she sees the SABE default world and that marks a successful sign up.

2.4 - Use Cases

Shown below are the use cases of the user of SABE. It shows the actions each of SABE users can perform.







CHAPTER 3: IMPLEMENTATION

3.1 – System Design Overview

The design of the SABE system comprises of the usage of the MySQL Server database that serves as a host for the registered members with their details, and the details of the meetings that are held. It also houses an activity log that keeps track of the times and dates at which meetings are created. On top of that, there is an interface in PHP that interacts with the database and retrieves information from the server to be used by the application and made available to the members connected as and when it is needed. Some of the information made available to the application and clients connected are available meetings, avatar information and many others.

The layer in *figure 2* named ***APP Resources*** is house for the textures and GUI skins used to display all the elements on screen like the windows, 3D models of the avatars and the world props like the lecture hall etc. It is also the main engine that keeps SABE running having the code for creating meetings, establishing connections, signing up, updating user information, and the main SABE logic. On top of the SABE APP is the SABE WORLD server which everyone connects to when there is a successful login. After logging in, a client can convert itself to a server by creating an activity like a meeting and would be hosted on the Master Server as an available event one can connect to. The Master Server acts as a mediator between the clients connecting and the other running servers or activities. Explaining further, it helps clients find running meetings or other events hosted on other servers and when a client finally chooses a server to connect to, the rest of the communication goes on between the user who started/organized the meeting (Meeting server) and the clients connected to it (Meeting attendants). Also to speed things up, I made the client talk to the APP resources directly without going through the SABE server because this would greatly increase the performance of the system. So this means that the instance of SABE running in the client web browser already has access to the animations, textures and other resources by default because they were built into it.

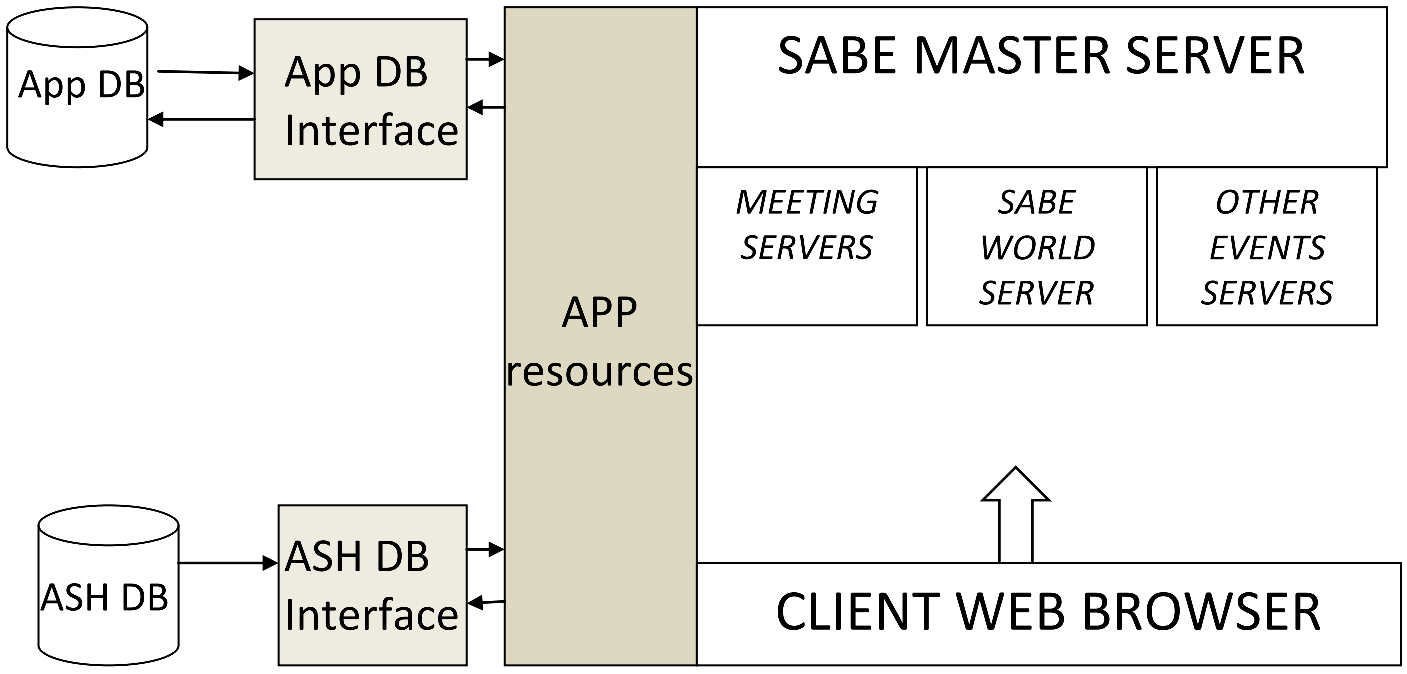


Figure 2 (The system architecture for Simulated Avatar Based Environment)

3.2 – Tools Needed

The main tool that would be used to build this system is the Unity3D Game Engine.[9] Unity3D is a 3D game authoring application. As a game engine, it is optimized to process the transformations of 3D elements at an appreciable rate. It is also compatible with 3rd party 3D modeling software like Blender3D or Maya making it possible for developers to import various 3D file types into Unity3D. With Unity3D, there is a medium through which it can communicate with a MySQL database using PHP. In essence, it can make calls to server side scripts like PHP and Perl. Apart from that, it has the ability to port the finished product to a Unity3D Webplayer compatible file which in essence makes it possible for the system to run in a web browser that has the **Unity Webplayer**[10] installed. Moreover, Unity3D is built on top of the Mono Framework making some .NET functionality available to the developer. Also, there is an active forum of Unity developers that provide help with all kinds of questions developers might have with regards to the game engine. Included in the Unity3D package is a networking capability built on top of ***RakNet*** which is a networking middleware for use in the computer game industry. This takes care of the networking capabilities of the project.

One powerful feature that led me to choose Unity3D as the authoring application is a new implemented system called the **Mecanim**

[9] http://unity3d.com
[10] http://unity3d.com/webplayer/

Animation System[11] in the newest version of Unity3D. This animation system takes the job of developers having to create new animations for every new character by making it possible for developers to reuse animations on different set of characters speeding up development. Again this tool was chosen because of the language options it provides developers. Developers have a choice of using C#, Unity Script which to some extent is a strongly typed version of Javascript syntactically and Boo which also has the characteristics of Python from syntax point of view.

The second major tool used in this project is Blender3D[12] modeling and animation software. This tool was chosen because it is free and open source. Usually 3D visualization software that has the amount of power Blender3D aims for are very expensive ranging from $800 to $1500. For example, 3D Studio Max, a 3D modeling and animation tool, cost about $1500. With what I want to achieve, Blender works just fine. The software also has a large community support meaning that your troubles with the software are likely to be already solved by someone in the Blender Users community. The output files it generates are also compatible with Unity3D, the main tool I am using. With regards to the project, Blender is the software I used to model the 3D props, that is the school, the lecture all with furniture and seminar room. Apart from this, Blender has the ability to animate humanoid characters using motion capture files making the character animations more human.

[11] http://docs.unity3d.com/Documentation/Manual/MecanimAnimationSystem.html
[12] http://www.blender.org

Tools like Corel Draw X3[13] and Adobe Fireworks[14] also aided in the development of SABE. I mainly used Corel Draw X3 to design the mockup UI and all the SABE UI elements, like the buttons, windows etc. Adobe Fireworks was also used to fine tune the images before sending them to Unity3D to be used by the game engine. Usually, in developing games, there is a necessity to keep the resources to be used in the game as small as possible to make the game run efficiently and Fireworks helped with that. Notepad++[15] and Monodevelop[16] were the basic text editors I was using. I would use Notepad++ for the server side scripts and Monodevelop for the scripts meant for Unity3D. Fortunately, the version of Monodevelop I am using was customized for Unity3D so the intellisense that come it with is actually in the right context. This version comes with the Unity3D version 4 installation. (see Appendix iv)

During the development of SABE, the development build tests were running in the Windows Operating System. The final build is actually meant to run in a web browser though.

When it comes to the hardware used to accomplish this project, I used a D-Link Gigabit Switch which was used to connect all the computers together. The connection media used were CAT 6 patched cables. The default SABE server is a HP G60 Dual Core AMD Notebook.

[13] http://www.corel.com/corel/category.jsp?cat=cat3430091&rootCat=cat20146&storeKey=us&languageCode=en
[14] http://www.adobe.com/products/fireworks.html
[15] http://notepad-plus-plus.org
[16] http://monodevelop.com

3.3 – Software Process Model

The software process model used to design the system is the Agile model. Agile was used because as the development process continued, new ideas were coming up and since they would make the system better and more usable and useful, they were incorporated in it. There were times where some features got modified for example, initially, all the members connected could see every available server and if you are really supposed to be connected to that particular server, you insert the password and you are allowed. However I changed this by making the availability of server dependent on the member connected. If the member has been invited, then, he sees the available server, if not, he/she does not see it.

3.4 - Object Oriented Structure

Below is a logical structure of the Object Oriented Organization of project SABE. The class architecture that a developer might end up with using Unity is likely to be a flat structure. This is because, every script that you create automatically inherits from a parent class called Monobehaviour. It is from this class that a lot of things are achieved like draw GUI elements on screen, making calls to Unity based functions etc. Now, what happens is, another script or class cannot inherit from it because it does not support multiple inheritances.

In SABE, the class structure is implemented almost laterally in that individual scripts have the ability to talk themselves to get information from each other for instance, the meeting class has to retrieve the world time from the World class in order to validate the date the user sets as a date for a meeting. This is to make sure the user does not enter a past date. The only inheritance that is happening here are the meeting and public event class inheriting from the Activity class.

When it comes to the server side scripts, there is a logical three layered functionality. It comprises of a series of functions that directly interact with the database like queries, connecting etc. On top of that is a class that utilizes the database interface. Finally, every action that has to be performed with respect to the database creates an instance of the functionality class and then through that deals with the database. For example, when a user wants to login, an http request is created with a URL accessing login.php on the server side. This login.php then creates an instance of the functions class and calls a function to authenticate when the details that were passed as arguments to them.

See Appendix (i) for class UML – Diagram

3.5 - Database Design

The Database Design for SABE consists of four table namely, the \`sabemembers\`, \`meetingtable\`, \`ashTable\` and \`activityinfo\` table. The \`sabemembers\` table keeps the profile of all SABE members would have signed up with their details. As part of the information that is kept for a SABE user, the meetings that it organizes too are saved. On the meeting table, there is a foreign key field named organizer that references the SABE user ID. Also included in the table for the SABE user is the avatar's state in the world. In this version of SABE the avatar has two states, the idle and busy state. The idle state is when it is in the avatar is free in the world not engaged in any activity and the busy state is when the avatar is in a meeting or another event.

In the meetingtable, there are fields that are associated with everything concerned with a SABE meeting.

Please refer to **Appendix (ii)** for the database structure of SABE.

3.6 - Network Architectural Design

With the networking design, by default, all the computers that login to SABE automatically become clients of the SABE default server. Now, when a client decides to have a meeting, that computer ceases to be client. This computer initializes the Unity server but before it does that, it stores the details that are needed to connect to it in the Master server which is actually a MySQL database. The other clients that are connected to the SABE default server get notified if they were invited to join the newly created server. If they connect to the newly created server, they cease being clients to the SABE default server and become clients of the newly created server.

3.7 - How it works

SABE is built with people in the Ashesi Community being (that is people with Ashesi username and password) the target users. At start up, a user is required to log in with his/her Ashesi username and password. After login, the server tries to check the authenticity of these credentials using the SABE server. If this check fails, then it falls back on authenticating with the Ashesi's Database server. After successfully authenticating with the Ashesi server, the user's login information is copied to the SABE DB server so that next time the user logs in, there is no need to authenticate with the Ashesi DB server. The idea is that when the authentication process has to fall back on using the Ashesi DB server, it means then that user is a first time user. However, if the user if authenticated with the SABE server, then the user is already logged into SABE at least once and hence his/her details have been copied to the SABE server. After checking whether or not the user is a first time, the program then decides whether not to show the sign up window and avatar creation scene or not. If it is determined that the user is a first time user, the next stage after login, is a window asking for more details about the user to construct a profile for the avatar. The details include your preferred screen name, interests and an about me section. These details are used to update the member's user profile in the database. From here, the user is asked to choose his/her avatar. If a user is satisfied with the choice of avatar, a button is clicked to move unto the SABE world. The details of the avatar are then saved to the SABE DB so that the avatar information of the user would persist. The user is then taken to the SABE world. The SABE world user interface has four main components; the chat window, the world information window, the activity window and the notification window.

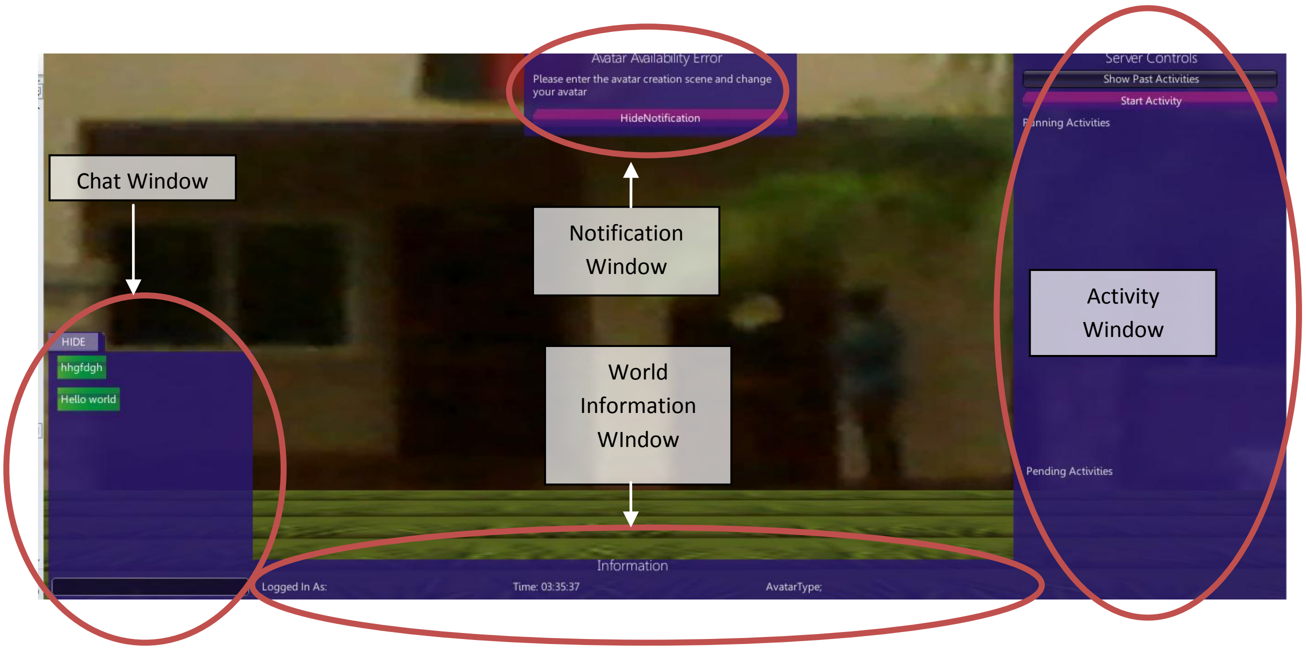


Figure 12 (User Interface of SABE World)

The chat window is the window the user interacts with he/she starts chatting. It can be activated by just typing or by you receiving a message from someone and hidden by pressing the ESC key or clicking on the Hide button. When a user presses enter to send a message, a RPC called is made to other clients connected to the SABE world server. In Unity, it is possible to make this RPC buffered. This means that future clients or clients that were not connected as at the time this called was made would be updated with every buffered RPC call once they connect. In essence, it is possible for someone to get all the chat messages that were sent to all clients before he/she connected at the time of connection. However, I do not buffer the chat RPC calls and this is because, apart from memory consumption and bandwidth usage, I think it is more natural for already connected clients to summarize events to the newly connected client rather than creating an instance of the chat history for the newly connected client. Otherwise, whilst the chat continues, the newly connected client would be reading past conversations instead of contribution to the conversation ongoing. During meetings, the chat history for a meeting is saved to the meetings table at every fifty chat entries.

The notification window is responsible for displaying notifications and tooltips. Some of the notifications include informing the user of an impending meeting, cancelled meetings etc. It was created as a form of a component so whenever a notification has to be displayed, it is added as a component to the script that wants to display a notification and a call is made to display the notification. You can think of a component here as an instance of a class.

The next window is the Activities window. This window displays the active and pending meetings/events the user has to attend to. It is important to note that the meetings or activities that show up in this window actually the depend on the user. This means that the user is not bothered with activities that he/she has no business with. Initially the design decision was to let the user see every available event and if a user wants to enter meeting, he/she types in a password which accomplishes the rational that a meeting is only attended by a subset of users. However, after analyzing this decision, I realized it would be efficient if I let users see what is just meant for them. In that case, one enters an event he/she has been really invited without having to bother about passwords. If a user has a pending meeting/event, a countdown to that event is displayed. If the user is the organizer of a meeting, he/she has the ability to modify or even delete the meeting. Since the information displayed on this window is fetched from the database, the items are updated every **two** minutes and so this means if an organizer cancels a meeting, it would take at most **two** minutes for other invited clients to realize that the meeting has been cancelled. The decision for the time was primarily based on reducing the amount of server side calls that has to be made.

When it comes to the world information window, this window displays the information that a user needs to know concerning the status of the system like the world time, the username, the communication mode (shout, converse or whisper). When the system starts, the script behind the world information window queries the server about the current time and updates the world time and then keeps track of the time on the client from then on.

Once a user is in the SABE world, the user has the ability to communicate with other people in the environment. In SABE there are three communication modes. We have the shout, converse and whisper. A shout is where a user's message is broadcasted to anyone who has logged into the environment. This can be achieved by the user just typing on the keyboard. With converse mode, when a user's avatar comes close to another user's avatar, the converse mode is established. This is where a user or a group of users can chat with each other once they in a location where they all see each other. With whisper, if a user's avatar really close to another user's avatar, the whisper mode is activated. With this, even if there is a group conversation established, a subset of users can enter the whisper mode and converse without the others in the conversation knowing.

When it comes to setting up a meeting, a user clicks on a ***Create Activity*** button to show a window with text fields that ask the user for the details of the Activity he/she wants to set. After supply in the information, the activity is saved into the Database and the clients that have been invited or are supposed to be in that activity gets notified in their activity window. If the time for the activity was set to *Now,* then the organizer of the meeting would disconnect from the SABE world server as a client and makes itself a server. As part of the activity details that are stored in the database, the IP address and Port number of the newly created activity server are gotten and stored in the SABE DB, so other clients can connect to it. After the activity is over, the state of the activity is changed from Active to Pending so that clients would not see it in their notification window again.   (Figure 4).

When meeting is in progress, the user cannot see other meetings available to him/her so the window that displays activities now displays buttons and GUI controls that fit the context of the activity. For example in a meeting, a user would see the leave meeting button and the organizer would see End meeting button. After a meeting, the history of the chat is saved to the database.

3.8 - Future work

This project has many prospective uses in the future. In the future, there is going to be an inclusion of voice chat. Also, there is going to be support for a wide array of file formats apart of Powerpoint files like Microsoft word etc. Also, I plan to make it possible for users to advertise in the virtual world. All they have to do is to upload the image depicting the advertisement into the world. Another interesting feature I am trying to integrate is the connection with the user's Facebook account to increase the immersion of the system. Facebook has a PHP-SDK[17] that can be downloaded for free and then a developer would have the ability of communication with the Facebook API. Lastly, I want to find a way of introducing the concept of money in the virtual world so avatars in the world would be able purchase things with something like SABE coins or something of that sort. Imagine you want to top up your mobile phone airtime. I think it should be possible to enter the virtual shop inside of SABE and purchase airtime with your virtual coins and you actually get an account top up in the real world.

3.9- Challenges

During this, project, there were a lot of challenges there were faced. First, network programming, is not as that easy. I faced a variety of problems during this project which in effect made me make certain design decisions about project. Firstly, I was supposed to a Unity Master Server. This is actually a C++ code that I had to recompile and run. Initially it could work when I run both the client and server instances on the computer running the Master Server. However, when I make client run on a different computer, it would not communicate with Master Server for information about other hosted servers. Communication with the Master Server was socket based. After reading some forums online, I got learn that the

[17] https://github.com/facebook/php-sdk

Master Server had some bugs in it and so it made resort to making the communication with Master Server by http.

Another problem I faced was with the characters and their animations. The version of unity I am using comes with a new animation system called the Mecanim animation system that is supposed to ease character animation. However, this also meant that there are only certain kind of animations a developer can use, that is the Mecanim compatible animations. For example, I tried importing animations that I got from a Motion Capture site, however, when I did so, the left and right legs were inverted when brought to the Mecanim Animation System. So because of this I had a very limited number of animations to choose from.

Other problems I ran into were related to networking like IP addressing issues etc. When I connected the computers together with a switch, my first question was, which computer would be a DHCP server to be distributing IP address since there was not an explicitly set IP address. I realized Windows gave auto configuration address to itself anytime it connected to a network. So I connected three computers with two running windows 7 and one Windows Vista. I realized that anytime I connected the Vista to the network, the IP addresses that the two Windows7 PCs changed and there making communication impossible.

Another problem I faced also was the fact that technically, Unity3D did not support the use of an open File Dialog to select files and so the work around for this issue was that, since I was porting to the Web browser, I used Unity3D’s ability to run browser based Javascript to modify the

Webpage the world is running in to create an <input type=file> in a <DIV> tag below the virtual world so users can click on to upload files.

CHAPTER 4: TESTING AND RESULTS

Outlined below are the various tests that were performed and the results associated with it.

Test	**Result**-	**Observation**
Login	Login worked as expected	It was observed that when the client trying to log is not physically connected to the server, it took a long time to determine that logging would not be possible.
Update user information	Updating users' inform-ation to the member table was successful	It was observed that once the user click on the "Go Back" button login again either with same credentials or with different credentials, the user will miss choosing an avatar and hence if this same user tries to login again, he/she will not get an avatar. Correction of this error is currently in progress.
Choosing an avatar	Choosing an avatar works correctly	No irregularities were detected during this session
Movement of Avatar in	Moving the avatar by	When I try to increase

the world	the user works as expected.	the speed at which the avatar moves, it seems to doing jumping from one location to another instead of walking
Retrieve Activity list	Retrieving the list of activities with respect to a user works as expected	No errors detected during this test.
Chatting	Chatting with other people in the world works	When connection is lost, and established again, the chat can continue just that the messages transmitted during the disconnected period is not sent later.
Setting up a meeting	Setting up a meeting both pending and immediate meeting works however, the countdown for the meeting has not been implemented yet.	For some reason, other clients are sometimes not able join in a meeting is currently under investi-gation.
Viewing Previous Meetings and history	Showing the history of meeting attended by a user works and showing of the meeting details (meeting name, agenda and minutes works)	There was no observable error with this test.
Switching or changing	This particular feature did not work as	The communication mode was supposed to

the communication according to proximity.	expected.	change when it comes in contact with another avatar how-ever nothing happens when that condition is established.
Giving notifications	The class responsible for showing notifications works as expected	No observable was detected.
Porting to Web browser	Porting the project to run the web browser worked to some extent, however when you try to run the project, it still wants to re download and reinstall the unity3D webplayer	The browser fails to recognize that the unity Webplayer has already been installed.

CHAPTER 5 – CONCLUSION

This nature of this communication project makes it usefulness applicable to other institutions like hotels, tourist sites, offices etc. When it comes to hotels, SABE can let people take virtual walkthrough of the hotel before they actually decide to lodge in that hotel. With SABE, they also get to communicate with some of the hotel attendants and the like. When it comes to tourism, people desiring to visit some tourist site would have the opportunity to do so virtually before actually the travelling to the tourist site physical. This will in turn serve as a form of advertisement for the tourist site operators.

This version of SABE developed is very basic version and as it is, there is a need for more work to improve the usability and stability in this basic state. Also it has to be modified in such a way people who want to build upon would find it easy to do so.

References

Fernando, T., Marcelino, L., & Wimalaratne, P. (n.d.). Constraint-based Immersive Virtual Environment. University of Salford, Salford, UK.

Fernando, T., Murray, N., Tan, K., & Wimalaratne, P. (n.d.). Software Architecture for a Constraint-based Virtual Environment. University of Salford, Salford, U.K.

Leahy. (2001, April 17). Scalable virtual world chat client-server system. United States.

Paiva, V. L. (2003, December 6). Feedback In The Virtual Environment.

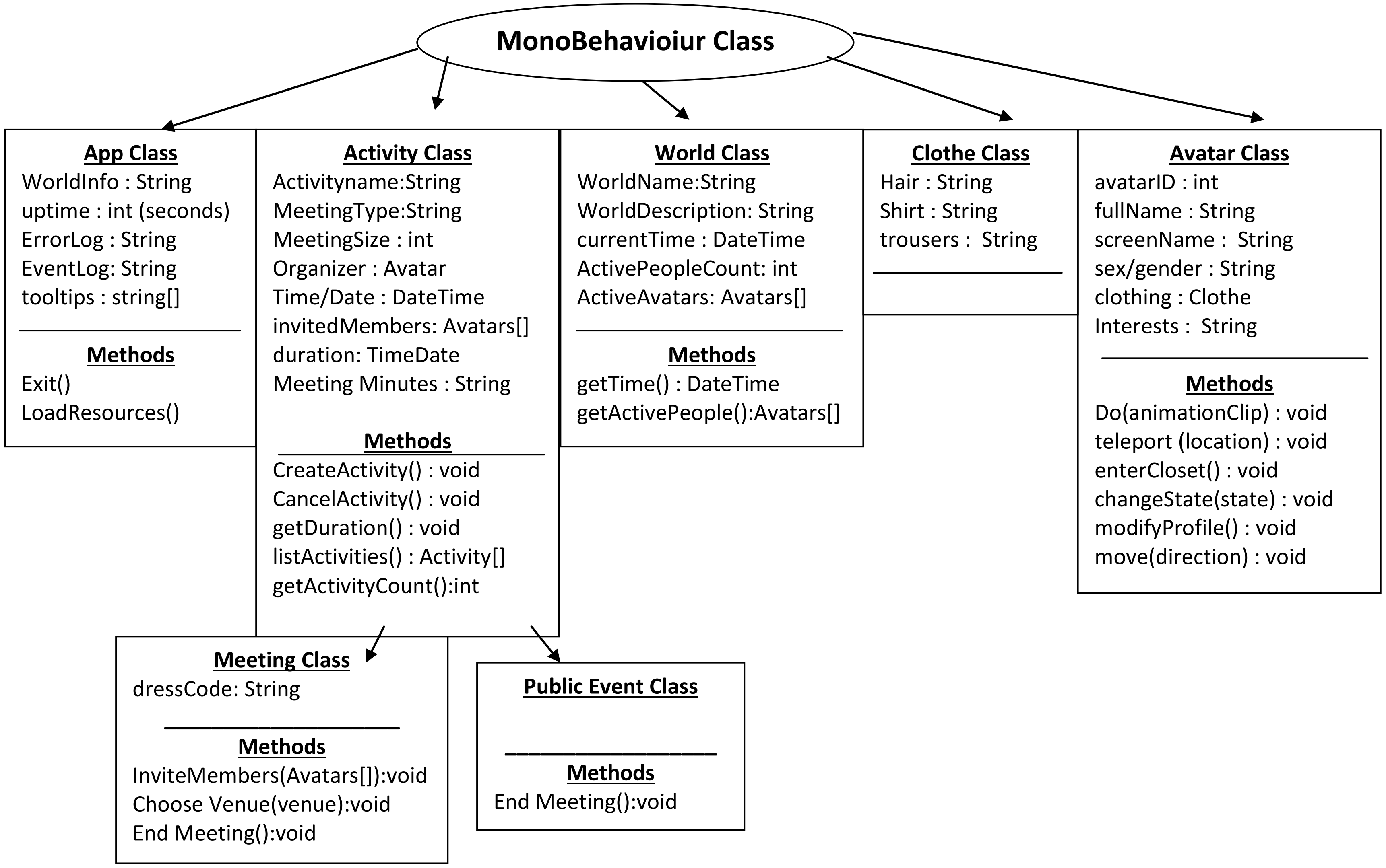
Universidade Federal de Minas Gerais, Brazil.

Peterson, M. (2005, July). LEARNING INTERACTION IN AN AVATAR-BASED VIRTUAL.

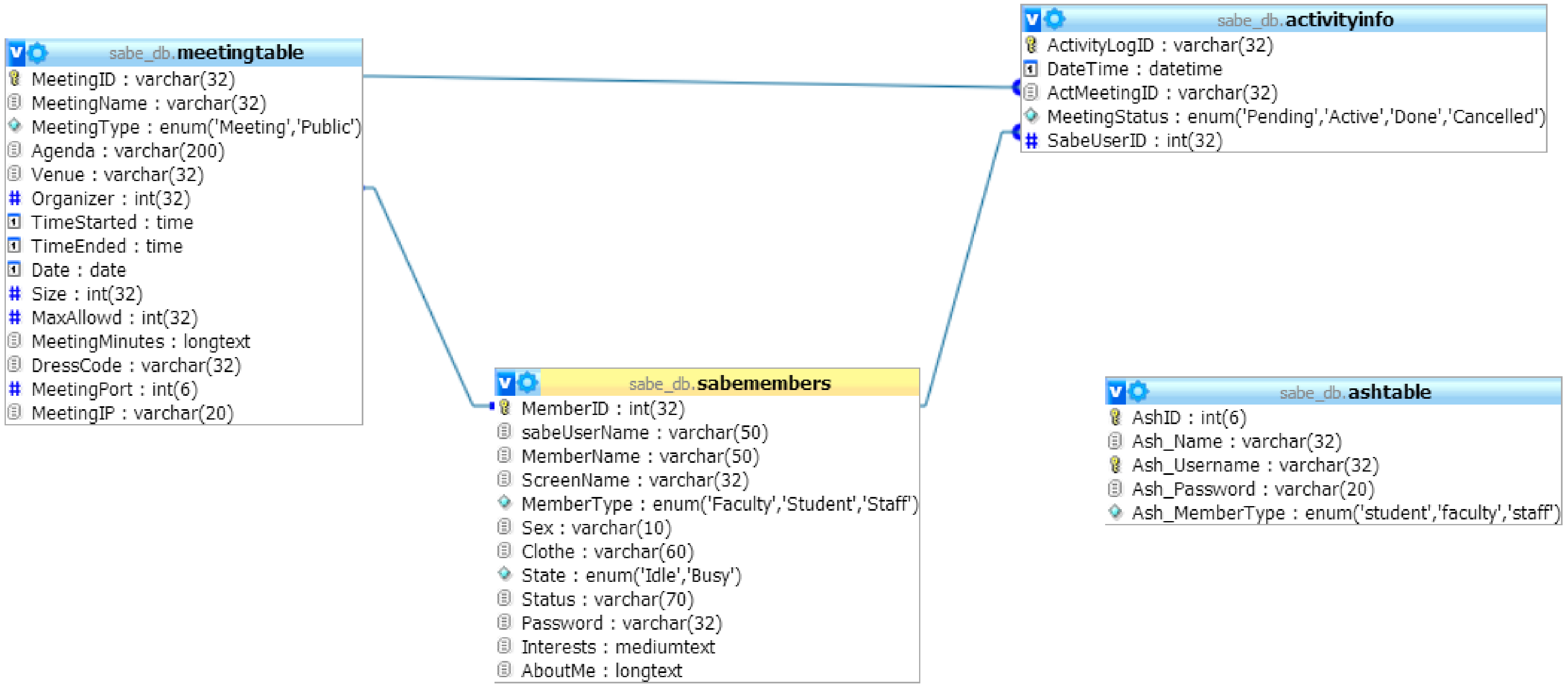
APPENDIX

Appendix (i) - UML of Class Diagram

UML of Class Diagram(Figure 3)



Appendix (ii) – Database Structure



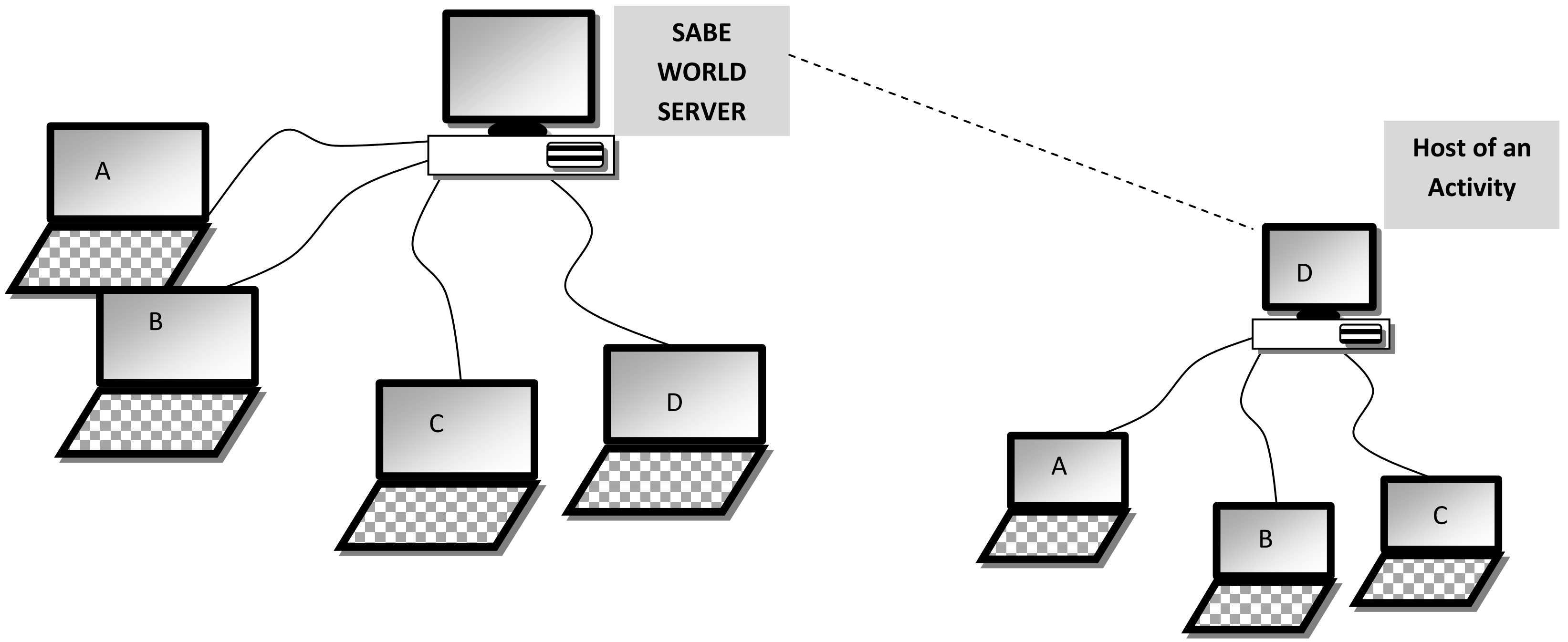
sabe_db.meetingtable
MeetingID : varchar(32)
MeetingName : varchar(32)
MeetingType : enum('Meeting','Public')
Agenda : varchar(200)
Venue : varchar(32)
Organizer : int(32)
TimeStarted : time
TimeEnded : time
Date : date
Size : int(32)
MaxAllowd : int(32)
MeetingMinutes : longtext
DressCode : varchar(32)
MeetingPort : int(6)
MeetingIP : varchar(20)

sabe_db.activityinfo
ActivityLogID : varchar(32)
DateTime : datetime
ActMeetingID : varchar(32)
MeetingStatus : enum('Pending','Active','Done','Cancelled')
SabeUserID : int(32)

sabe_db.sabemembers
MemberID : int(32)
sabeUserName : varchar(50)
MemberName : varchar(50)
ScreenName : varchar(32)
MemberType : enum('Faculty','Student','Staff')
Sex : varchar(10)
Clothe : varchar(60)
State : enum('Idle','Busy')
Status : varchar(70)
Password : varchar(32)
Interests : mediumtext
AboutMe : longtext

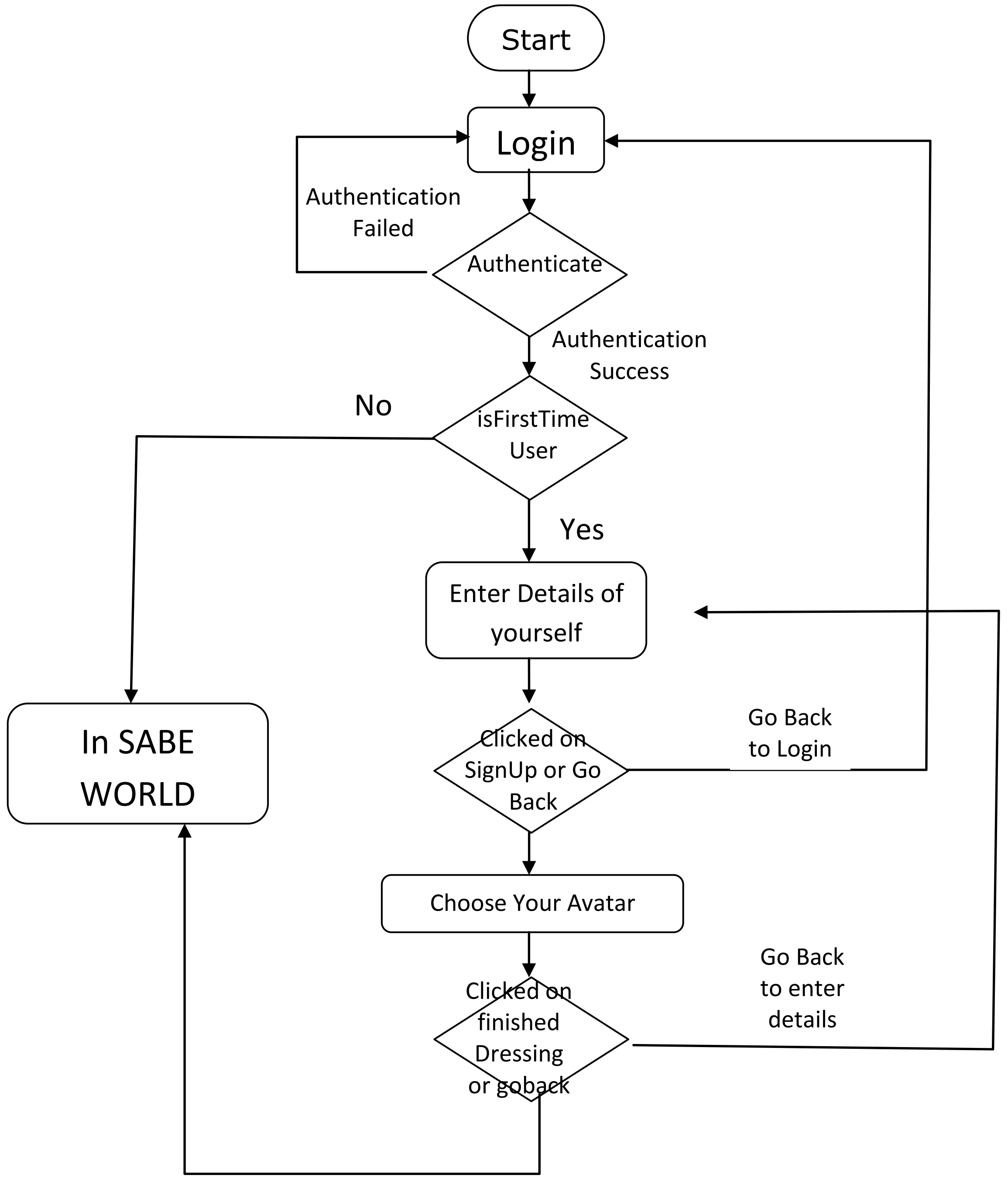
sabe_db.ashtable
AshID : int(6)
Ash_Name : varchar(32)
Ash_Username : varchar(32)
Ash_Password : varchar(20)
Ash_MemberType : enum('student','faculty','staff')

Appendix (iii) – Network Structure



By default, clients are connected to the SABE WORLD server when they are started. For a client to become a host for an activity, it has to become a server. In the diagram above, client D is initially a client, however, it started an activity and so became a server and had clients B, C, D join that activity. When this happens, the SABE world server keeps a record of the new server that has been started so that the clients connected to it will see the availability of other servers.

Appendix (iv)Flow chart to enter the SABE world



Appendix(v) - Installing Unity3D

1.) Navigate to http://www.Unity3D.com/download

2.) You can either download the pro version which goes $1500.00 or you can go for the free version which I used for the project. Be sure to install version 4.0 and above.

3.) After Download, click on the setup executable to install Unity3D unto your computer.

4.) After installation, you will be asked to register your copy of unity.

5.) There are two ways of accomplishing the registration. You can register manually or automatically.

6.) With the manual registration, you are required to click on save license which will save a certain file to the location you ask it to go. Then in the dialog, you will find a button that will take you to http://store.unity.com for you to get license. This is done by uploading the file you saved initially unto the site which will download the license automatically.

7.) From here, you click on the Read license button and open the file you just downloaded.

8.) If it is successful, it will proceed to open Unity3D game engine.

Appendix(vi) - Installing Blender3D

1.) Navigate to http://www.blender.org.

2.) Click on the download link on the right side of the page to go the download page of Blender

3.) Choose the Build that is suitable for your platform. Eg. Windows 32bit or 64bit, or MacOS etc.

4.) If you are using windows, you can either choose the .zip version (which is unzip and use without installation) or you can choose the Blender installer

5.) After this, run the unzipping command or the installer to install Blender.

6.) From here, blender is installed.