iDOCENT

Indoor Directional Orientation Communication and Enabling Navigational Technology

Android Smartphone Application

April 26, 2011

ECE480
Design Team 2 Members:

Gordon Stein
Andrew Kling

Matthew Gottshall
Jacob D'Onofrio

Facilitator: Michael Shanblatt
Project Sponsor: Resource Center for Persons with Disabilities (RCPD)
Sponsor Representative: Stephen Blosser

Resource Center
for Persons with
Disabilities

Executive Summary

Docent is defined as an individual who conducts guided tours which provided the influence for the acronym iDOCENT. iDOCENT is an Android smartphone application designed to locate and route a user to a desired destination inside a building. Since obtaining a GPS signal indoors is unreliable due to building infrastructure materials, a new method for accurately locating a user indoors was needed. In an effort to provide assistance to a variety of users, including the blind, a universal design for iDOCENT was implemented considering these objectives.

The application operates by using Wi-Fi access points (APs) within a building and applying a triangulation algorithm. The signal strength of each AP is determined by iDOCENT and paired with its respective coordinates. This set of data is then averaged to provide the user's location which is displayed on the smartphone. Given the known location, a routing algorithm is applied to determine the shortest path between the user and a selected point of interest.

iDOCENT currently operates within the Engineering Building at Michigan State University. It can locate and route a user to within four feet of several different destinations such as classrooms, labs, and offices. The intuitive touchscreen interface displays simple menus and bright visuals. A map of the Engineering Building is shown onscreen upon application launch. Rooms are color-coded based on type to further increase readability. A black "X" designates the user's current location while a continuing red line highlights the suggested path to the destination. In addition, an audible feedback feature states directional instructions and user selected menu options to facilitate accessibility.

iDOCENT is a valuable application not only for the blind, but to new students and visitors on Michigan State's campus. Being a software package, it may be easily distributed and applied to a large, existing consumer base. iDOCENT is open-ended in design and its functionality will prove to be a solution for many different applications.

Acknowledgement

Michigan State University's ECE 480 Design Team 2 would like to give a special thanks to the following individuals who helped contribute to the success of the iDOCENT smartphone application for indoor navigation. We would like to specifically give recognition to Mr. Stephen Blosser of the Resource Center for Persons with Disabilities for the opportunity to design an innovative product to benefit Michigan State University. He has provided the objectives and guidance in order to create a successful project. In addition, we would like to thank Al Puzzuoli of the RCPD for his insight and support of the project. We would also like to thank Dr. Michael Shanblatt for his facilitation and support of our project. His interest and motivation helped provoke innovative ideas within our group and inspire key product features for our project. We would also like to thank Mr. Gregg Motter of Dow Chemical Company for visiting ECE480 to give influential lectures on important business practices and entrepreneurship. We would also like to thank Mr. Steven Noll of Schiff Hardin for providing firsthand knowledge and input on intellectual property for technical products and applications. We would also like to thank Dr. Richard Enbody for taking the time to share his knowledge on indoor navigation and provide contact information for obtaining layouts of the Engineering Building. We would also like to thank the Division of Engineering Computing Services (DECS) for providing the spreadsheet of access point MAC addresses and their location within the Engineering Building. Finally, we would like to thank Mrs. Roxanne Peacock for her assistance in ordering parts for testing in a timely fashion. Without the help and contribution from the mentioned individuals, the ideas and functionality of the iDOCENT smartphone application would have become a reality.

Table of Contents

List of Table and Figures

Chapter 1

Introduction

Currently available technology for indoor navigation is limited and underdeveloped. There is not a single best approach for achieving an accurate and appropriate solution to indoor navigation. Methods being explored include the use of signal strength triangulation, radio frequency, and ultrasonic noise applications. However, there are several problems encountered when incorporating the different technologies into a solution. The feasibility of many approaches declines for large scale implementations due to infrastructure challenges. In addition, many of the buildings where indoor navigation will be deployed are often constructed with steel or aluminum, where a true GPS signal is weak or unavailable. This is the very problem that the iDOCENT application addresses.

The objective of iDOCENT, or Indoor Digital Orientation Communication and Enabling Navigational Technology, includes prototyping a feasibly priced solution that will locate an individual inside of a building. The key functionality behind iDOCENT is its ability to operate without the use of a GPS signal. Instead, a modified version of signal strength triangulation will be implemented to determine a user's location on a smartphone within four feet accuracy. iDOCENT will address issues caused by weak phone service and provide navigation to different points of interest indoors.

iDOCENT operates as a smartphone application, wirelessly calculating a user's specific location within a building. Its operation is akin to a GPS system found in today's cars. iDOCENT will allow an individual to seamlessly locate himself by providing audible feedback and turn-by-turn instructions on how to reach a desired destination. Location results are given in real time and points of interests may be found on the move. The indoor guidance system is the collaborative effort between the Resource Center for Persons with Disabilities and Michigan State's Electrical and Computer Engineering department.

Background

Research was initially performed on a variety of different methods for indoor navigation. Several of the early attempts included utilizing one of the following wireless technologies: Wi-Fi, Bluetooth, and RFID. Additionally, an Italian company was discovered who determined location using a method called dead-reckoning, where no external wireless signals are used.[1] All of the approaches were considered in the development of the iDOCENT application.

[1] http://en.wikipedia.org/wiki/Dead_reckoning, retrieved January 3, 2011.

Bluetooth is an open standard wireless technology that is not defined by the Institute of Electrical and Electronics Engineers (IEEE), unlike other popular wireless standards such as Wi-Fi. Bluetooth is popular for the use of Personal Area Networks, or PANs where a master and slave relationship is created between two or more devices. The technology has been growing in popularity because of the power efficiency features and ease of use. Nearly all smartphones today contain Bluetooth functionality. Although the technology is widely available, manufacturers who use Bluetooth do not always follow the standard, because it is not regulated, and often modify functionality. Also, Bluetooth does not by default offer an easy method of signal strength calculation.[2] When comparing to other existing wireless technologies, Bluetooth does not fit the criterion for indoor navigation applications.

Wi-Fi, or Wireless Fidelity, is a standard used today for broadcasting network connectivity. It is defined by IEEE under 802.11x. Most commonly, Wi-Fi is used for IP based networking equipment such as personal computers. The advantage of choosing Wi-Fi for a location based service is its high compatibility and frequency of availability. The majority of today's smartphones also have Wi-Fi connectivity. Newer revisions of Wi-Fi broadcast at the 2.4Ghz frequency, allowing for signals to more easily travel through obstructions like doors and walls. Unlike other wireless technologies such as Bluetooth, Wi-Fi incorporates signal strength functions into all the firmware drivers and Application Programming Interfaces (APIs) which are defined by the manufacturers and backed by IEEE.[2] This feature will provide a large benefit when using Wi-Fi to determine a location based on signal strength triangulation.

Dead-reckoning is a method for determining location strictly based on three main factors: velocity, direction, and known initial location.[1] Many of today's smartphones contain an accelerometer and digital compass hardware. Combining the sensor readings from both pieces of hardware, software can be used to track a user's movement throughout a building, without the need for wireless signals. The accelerometer can act as a pedometer, tracking steps taken, and the compass will determine the direction the user is walking. As long as the initial location is known, each new step taken can be recorded in the application.[3] Error is evident in dead-reckoning due to compounding miscalculated steps and compass calibration issues.[4] Additional algorithms may be incorporated to compensate for error but may require an excessive amount of trial-and-error based debugging.

[2] Patil, Abhishek Pramod. "Performance of Bluetooth Technologies and Their Applications to Location Sensing." 2002. January 2011 <https://www.msu.edu/~patilabh/research/Thesis_Report.pdf>.

[3] CRS4. Indoor Navigation System Prototype in a modern smartphone (No radio beacon, no GPS, no WiFi) . 30 September 2010. January 2011 <http://www.youtube.com/watch?v=HTZFRDBdQck>.

[4] Carboni, Davide. "Indoor Mobile Pedestrian Navigation System." n.d. CRS4. January 2011 <http://geoweb.crs4.it/lib/exe/fetch.php?id=indoornavigation&cache=cache&media=paper_mobile_hci_2010.pdf >.

While considering each technology, the most practical method for indoor navigation for iDOCENT's requirements will be met through Wi-Fi signal strength triangulation. It offers easily accessible and reliable hardware and the ability to use a smartphone to perform all necessary calculations. In addition, a combination of dead-reckoning and Wi-Fi positioning will be designed and tested. Sufficient results will influence final incorporation into the iDOCENT prototype.

Objectives

While concluding that Wi-Fi positioning would provide the best functionality to meet the iDOCENT proposal, a list of objectives or features was analyzed in order to realize the project's complete potential. Every building on Michigan State's campus is fully Wi-Fi capable, so the need to purchase and install additional Wi-Fi access points (APs) is eliminated. An additional advantage is a unique identifier assigned to every access point called a MAC address. In this manner, individual information about each AP can be stored in a database for later use. The database will be set up and maintained in-house. In order to determine indoor positioning, an algorithm needs to be written and executed within the smartphone application. Two key factors in the calculation of positioning are the following: documented location of all available access points and real time signal strength measurement. A custom coordinate system will have to be generated within the building and each AP's location must be documented. After the iDOCENT application is launched, location is obtained by first scanning for available APs, assigning a signal strength factor to each AP, and uploading one entry per factor to the database. Each entry will contain the known building coordinates of that particular AP. The stronger the signal strength, the more entries will be uploaded, and thus increasing the weight of the particular AP closest to the user. By averaging every entry in the database, a known location can be determined. This method has been referred to as triangulation and is illustrated in Figure 1.[5]

[5] Work, Eric. "UW Campus Navigator: WiFi Navigation." July 2003. Electrical Engineering Department, University of Washington. January 2011 <http://uwcampusnav.sourceforge.net/WifiNav.pdf>.

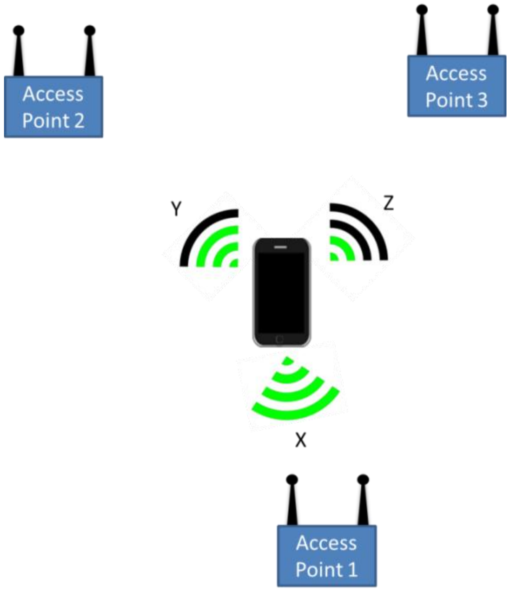


Figure 1: Illustration of Triangulation with Signal Strength

Figure 1 can interpreted as follows: The smartphone surveys the area and detects three access points. Each of the three APs location within the building has been previously documented and is stored in a database. The smartphone calculates and signal strength while retrieving the APs' locations. Following, one entry per signal strength factor will be submitted to a new database containing the location coordinates of the respective AP. Access Point 1 will have the most entries in the database because its signal strength is greatest, while Access Point 3 will have the fewest entries. Finally, all of the submitted coordinate entries are averaged to determine the coordinates of the user. After determining the user's location, a short-path algorithm may be applied to route the user to any particular point-of-interest.

The final approach is new and practical in that is uses existing infrastructure and works on a popular, existing platform. Maintenance will be self-sustaining due to the nature of the algorithm. If an access point is removed from a building, iDOCENT will simply not detect it, thus causing no issue in location detection. If a new access point is installed, code may be written to ping the developers that new hardware has been detected in the building and its location coordinates must be mapped out and uploaded to the database. The functionality of iDOCENT will become useful to a variety of users. Audible feedback features will provide assistance to blind users while an intuitive graphical interface will be useful to others. iDOCENT offers a complete navigational package with an open-ended design that may prove to be a solution for many different applications.

Chapter 2

Design for Six Sigma Tools

A key industry tool used in the design process to identify the ultimate goals and streamline overall development is the FAST Diagram. The diagram is an exercise to identify critical customer requirements and their respective solutions. As the diagram progresses rightward, the question, "How?" must be asked for each step. Alternatively, as the diagram is evaluated leftward, the question, "Why?" must be asked as a check for each step. The diagram starts with the main problem or focus of the project, which for iDOCENT is to direct a user to their preferred destination. Following the task of directing the user consists of three main ideas: relaying information, locating the user, and giving directions. In order to relay information between the user's smartphone and a documented database requires a supported server to hold the data. The server will store and maintain the database. Locating the user is dependent on the value of signal strength, utilized within the Wi-Fi algorithm. This is accomplished by surveying the site for available access points. Finally, directions will be provided to the user through an on-screen map and audible directions. These steps are illustrated in Figure 2.

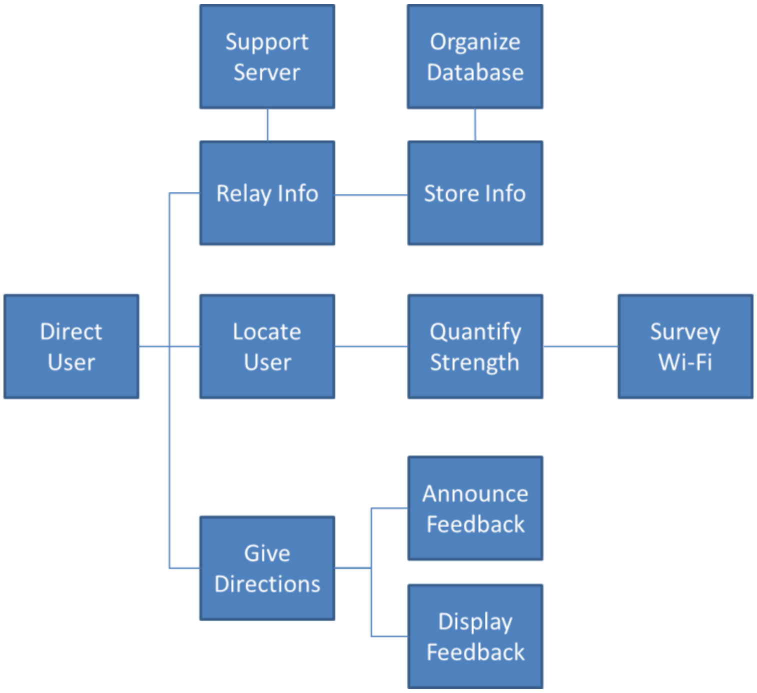


Figure 2: FAST Diagram

After research was gathered on a variety of different approaches for indoor navigation, a decision matrix was constructed on the set of conceptual designs. Each approach was ranked between one and five, one being the best, on qualities that would produce a feasible design. The qualities that were evaluated include: the availability of current tools, the ease of use and deployment, overall cost, precision, and previous familiarity with components required to fully realize the given design. The lowest score will determine the most feasible design.

While taking careful consideration into each technology, the most practical method for indoor navigation for iDOCENT's requirements will be met through Wi-Fi signal strength triangulation. This approach is seen with the lowest or best score, followed by the dead-reckoning design. As seen by the matrix in Table 1, Wi-Fi signal strength triangulation offers easily accessible and reliable hardware and the ability to use an existing smartphone to perform all necessary calculations. A combination of dead-reckoning and Wi-Fi positioning may be ultimately implemented based on the lack of variance in ranking.

Table 1: Decision Matrix

Row Labels	Bluetooth Design	Wi-Fi Triangulation	Dead-Reckoning Design	Ultrasonic Receivers
Available Tools	3	2	2	4
Deployment	4	2	1	5
Ease of Use	2	1	3	3
Overall Cost	4	2	1	5
Precision	4	3	4	1
Previous Knowledge	3	1	2	4
Grand Total	**20**	**11**	**13**	**22**

	Criteria Rank Legend
1	Best
2	Good
3	OK
4	Not Good
5	Worst

The conceptual design in Figure 3 illustrates a smartphone surveying an example building for Wi-Fi access points and detecting their signal strength. The smartphone will access a database containing the known coordinates (5, 80), (90, 120), and (100, 10) and make matches to the surveyed APs. One entry, containing AP location, per signal strength factor will be uploaded to a new database. For example, the AP located at (100, 10) will upload four entries, while the AP at (90, 120) will only upload two entries. The averaging function will be applied across all entries and the smartphone's location in Office A will be determined. The smartphone will generate a route to the destination and provide instructions to do so. In this case, the destination will route the user in Office A to the restroom.

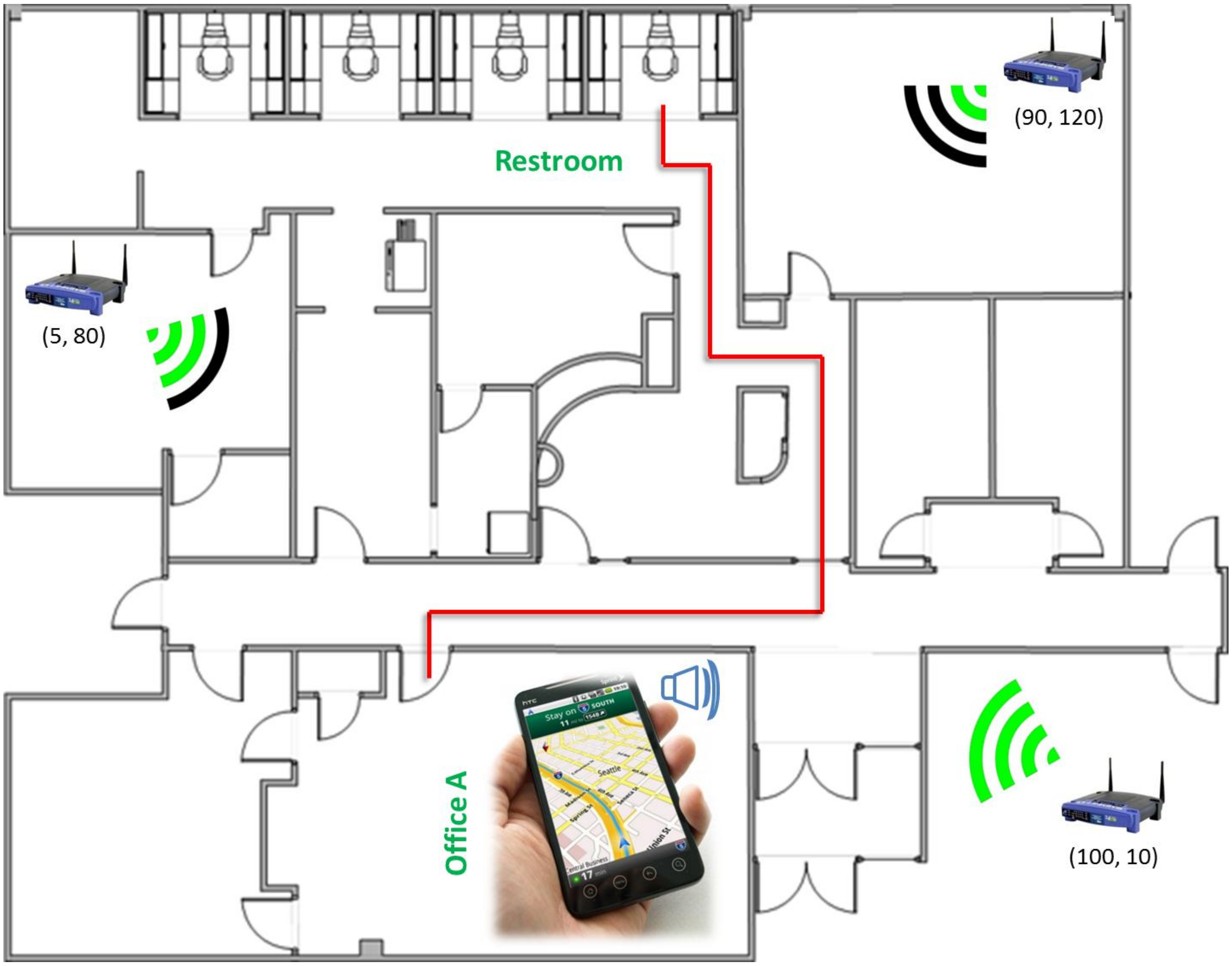


Figure 3: Conceptual Design

The risks assessed are based on the critical path shown in the following project management plan in Table 2. The plan highlights the fact that the tasks involving Wi-Fi software development, navigation mapping and algorithms, and overall testing all have substantial impacts upon schedule and completion. The analysis of risk was assessed to three key events that pose future challenges and concerns.

Table 2: Risk Assessment

Potential Pitfalls			
Task at Risk	Description	Number	Color
Cannot obtain the building coordinates of all access points in the designated building (e.g. Engineering Building))	Serious, Likely	12	Yellow
Wi-Fi algorithm cannot be refined to specified tolerance (±4 feet)	Major, Low Likelihood	5	Green
Navigation algorithm and specified paths cannot be determined	Serious, Low Likelihood	8	Yellow

		Risk Analysis				
Likelihood	Near Certainty	5	10	15	20	25
	Highly Likely	4	8	12	16	20
	Likely	3	6	9	12	15
	Low Likelihood	2	4	5	8	10
	Extremely Improbable	1	2	3	4	6
		Minimal	Minor	Major	Serious	Catastrophic
		Severity / Impact				

Risk Value Legend		
Low	< 5	Green
Medium	> 5 & < 12	Yellow
High	> 12	Red

Budget

The costs accrued in the design and testing of the iDOCENT application are itemized below. As seen in the financial analysis, costs are kept to a minimal. The backbone of iDOCENT is the smartphone software that was developed in-house at no cost. The main expense of iDOCENT is the hardware component, where four wireless routers were purchased for initial testing purposes. This financial analysis shows that $400.04 is still available from the initial department funding.

Table 3: Budget Analysis

	iDOCENT Costs		
Row Labels	**Sum of Quantity**	**Sum of Cost**	**Sum of Total Cost**
iDOCENT Hardware	**6**	**$24.99**	**$99.96**
Smartphone for testing	2	0	0
Wireless Routers for testing	4	$24.99	$99.96
iDOCENT Software	**1**	**0**	**0**
Smartphone app	1	0	0
Grand Total	**7**	**$24.99**	**$99.96**

Financial Analysis	
Description	Amount
Available Funds	$500.00
Enterprise Costs	-$99.96
Extra for Error	$400.04

Schedule

The project management plan and Gantt chart describe the personnel and their tasks, the resources to be used, and the project schedule. The schedule contains a breakdown of related sub-tasks with their time requirement and the respective time frame for completion. Technical tasks, along with the non-technical deliverables, are all clear and well-defined within the plan. The critical path includes Wi-Fi software development, navigation mapping and algorithms, and overall testing. This project has a starting date of February 14th, 2011 and will complete April 29th, 2011 on Design Day.

Table 4: Gantt Chart Tasks

	Task Name	Duration (days)	Start	Finish	Predecessors	Resource Names
1	**App Programming**	76	2/14/2011	5/1/2011		
2	Refine Wi-Fi Algorithm	4	2/14/2011	2/18/2011		Matt, Jake
3	Dead Reckoning Consideration	11	2/14/2011	2/25/2011		Matt, Jake
4	**Navigation**	60	3/1/2011	5/1/2011	3,2	
5	Algorithm	28	3/1/2011	3/29/2011		Matt
6	Mapping	18	3/29/2011	4/16/2011	5	Team
7	Testing	14	4/16/2011	5/1/2011	5,6	Team
8	**Audible Feedback**	4	3/28/2011	4/1/2011		
9	Programming	4	3/28/2011	4/1/2011		Matt, Jake
10	Meet with Al	1	3/28/2011	3/29/2011		Team
11	**Interfacing with Database**	53	2/25/2011	4/20/2011	3,2	
12	Setup Database	4	2/28/2011	3/4/2011		Jake
13	Contact DECS	4	2/28/2011	3/4/2011		Andrew, Gordie
14	Overall Testing	53	2/25/2011	4/20/2011		Team
15	**Deliverables**	82	2/6/2011	4/29/2011		
16	Pre-proposal	1	2/6/2011	2/9/2011		Team
17	Final Proposal	5	2/10/2011	2/25/2011	16	Team
18	Technology Showcase Poster	21	2/20/2011	4/25/2011		Andrew
19	**In-class Presentation**	7	2/21/2011	2/28/2011		
20	Prep	1	2/27/2011	2/28/2011		Team
21	PowerPoint Slides	7	2/21/2011	2/28/2011		Andrew
22	Website	27	2/7/2011	4/28/2011		Jake
23	Group Photo	1	2/20/2011	2/22/2011		Team
24	FAST Diagram	7	2/14/2011	3/8/2011		Team
25	Progress Report	14	2/28/2011	4/12/2011		Team
26	Notebook Photocopy	1	3/11/2011	3/14/2011		Team
27	Demo for Facilitator	1	3/20/2011	3/22/2011		Team
28	App Notes	7	3/21/2011	4/12/2011		Team
29	**Technical Lecture**	3	4/3/2011	4/6/2011		
30	Prep	3	4/3/2011	4/6/2011		Team
31	PowerPoint Slides	3	4/3/2011	4/6/2011		Andrew
32	Design Issues Paper	8	4/3/2011	4/11/2011		Team
33	Demo 2 for Facilitator	1	4/10/2011	4/11/2011		Team
34	Progress Report 2	7	4/4/2011	4/11/2011		Team

35	Professional Self-Assessment Paper	7	4/11/2011	4/18/2011		Team
36	Final Report	25	4/1/2011	4/26/2011		Team
37	**Design Day**	21	4/8/2011	4/29/2011		
38	Demo Prep	2	4/8/2011	4/10/2011		Team
39	Presentation Prep	21	4/8/2011	4/29/2011		Team
40	Faculty Invites	1	4/18/2011	4/19/2011		Team
41	Scope the Union	2	4/8/2011	4/10/2011		Team
42	**Ordering**	4	2/5/2011	2/9/2011		
43	Router Order from Newegg	4	2/5/2011	2/9/2011		Team
44	**Meetings**	63	2/11/2011	4/15/2011		
45	Sponsor Update 1	0	2/11/2011	2/11/2011		Team
46	Sponsor Update 2	0	2/25/2011	2/25/2011		Team
47	Sponsor Update 3	0	3/25/2011	3/25/2011		Team
48	Sponsor Update 4	0	4/15/2011	4/15/2011		Team

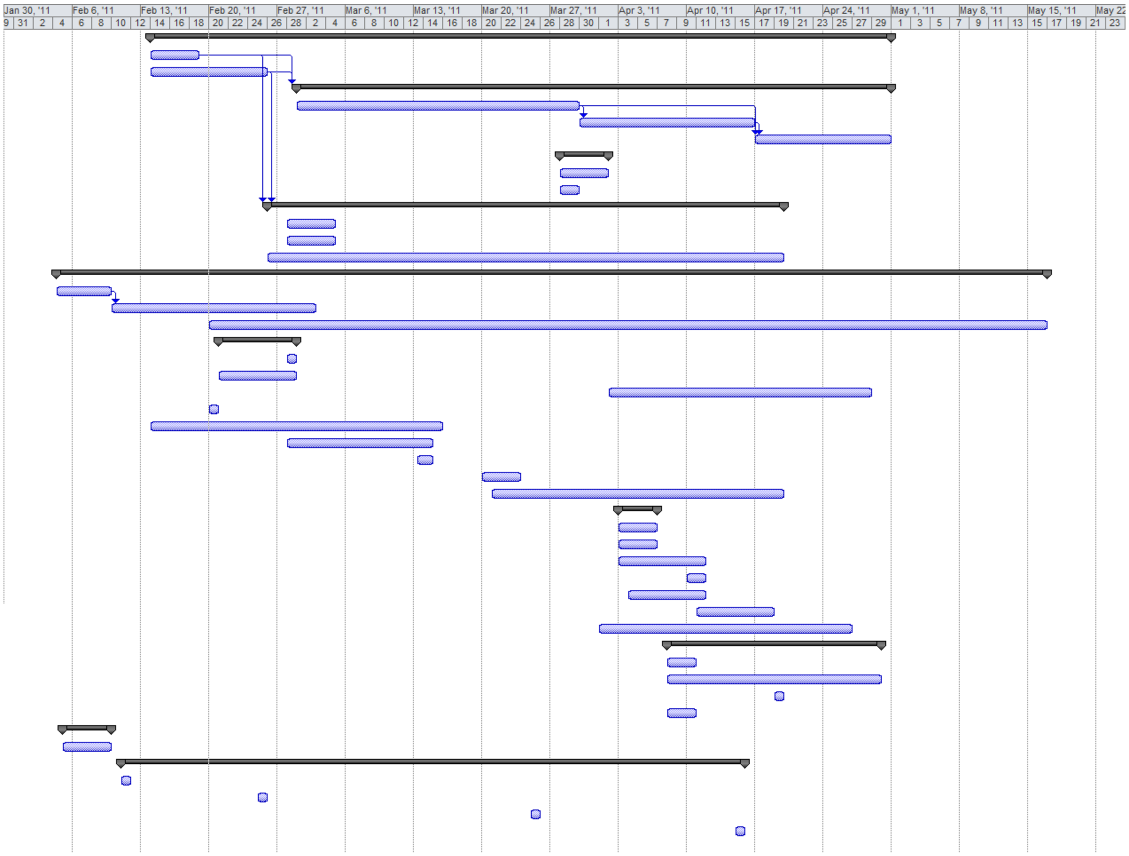


Figure 4: Microsoft Project Gantt Chart[6]

[6] http://www.egr.msu.edu/classes/ece480/capstone/spring11/group02/documents/gantt.mpp

Chapter 3

Key Terms

The following is a list of key terms commonly used in the technical writing within Chapter 3.

- API – Application Programming Interface – An interface between one program to another where the first program generally provides tools to assist in the programming of the second.
- (W)AP – (Wireless) Access Point – A device that allows the user to connect wirelessly to a network.
- App – Short for application.
- BSSID – Basic Service Set IDenitifier – This is another name for a MAC address.
- dBm or dBmW – deciBel milli-Watts – A signal strength rating in decibels as a power rating compared to 1 milli-Watt.
- IDE – Integrated Development Environment – A program designed to assist a programmer to make programming and executing applications easier.
- MAC address – Media Access Control address – The physical address of a network device.
- PNG – Portable Network Graphic – A bitmapped image format that utilizes lossless data compression
- SDK – Software Development Kit – A set of tools that assist in the development of an application. Note: This application is based on the Android ADK revision 10.
- SSID – Service Set Identifier – A name given to a network to describe it.
- Wi-Fi – Wireless Fidelity – The protocol most frequently used for wide local area networks.
- Thread – The smallest unit of processing that can be executed by the operating system.
- TCP – Transfer Control Protocol – A standard protocol in the transport layer of the OSI network model. TCP is connection oriented and provides flow control, congestion control, reliability, and error detection.
- DBMS – Database Management System – A set of computer programs that manages the use of a database.
- SSL – Secure Sockets Layer – A cryptographic protocol that provides communication security over the internet. TLS is the successor of SSL, but sometimes the names are used interchangeably.
- Socket Programming – Programming that deals with network communication.
- Schema – The databases structure described in a formal language by the DBMS.

Design Solution

It quickly became apparent that using Wi-Fi access point signal strength would be the simplest and most cost effective method for locating a user with a smartphone. It was also easily decided that the Android platform would be the best choice for developing this application since it is free to develop and test on Android and two members of the team already had access to Android based smartphones.

In order to accomplish all of the objectives, the problem was split into two interacting pieces: the server backend and the application frontend. The server backend provides information and support for the application frontend by using a database. There are three main elements to the application portion of the solution to this problem: location calculation, graphical drawing of the map of the current building, and the user interface. Both of these major pieces of the problem are described in more detail below.

Server Backend

The server backend for the iDOCENT application was created with the possibility of expansion. Having hard coded dynamic data is a bad idea to deploy across mobile phones. In this manner, a server was created to address this problem. The server performs the following functions: it holds location data and feeds it over the internet to mobile phone, and it calculates shortest path directions for the building maps.

Server Base

Portability is the main objective when programming for mobile devices. With the possibility of another design team continuing the development of iDOCENT, considerations were made to avoid providing code that was restricted to a certain platform. Java became an obvious choice to solve this problem. It also made coding for both Android and the server easier since they are both compatible. Java also enabled the reuse of code in a few instances such as socket programming for internet communication. The setup that will be used is a 64-bit Linux server running Java version 1.6_24.[7]

Network Connectivity

The server's first job is to listen for an incoming connection, in this case, from iDOCENT. The server listens for a TCP connection on port 1024. Once a connection is established, the following thread is created to continue servicing the client:

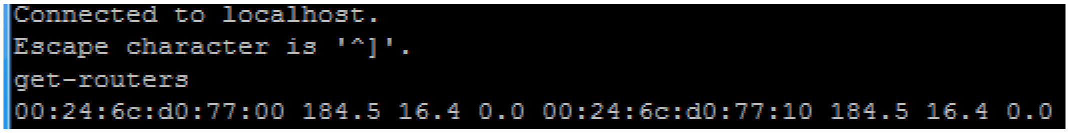
```
Thread t = new Thread(connection);
t.start();
```

The main thread then continues to listen for more incoming connections while the child thread branches off and services the requests of the client. This multithreaded design allows the server to efficiently multi-task in order to service multiple clients at the same time.

The child thread contains all the main logic for communication. When it reads in a command that it recognizes from the client, it will perform a certain task and feedback any information that was asked of it.

[7] http://www.java.com/

The communication protocol the client and server use contains eight commands each serving a unique purpose. When iDOCENT is initially powered on it initializes its Wi-Fi connections. A connection is established with the server and it sends “get-routers”. The server will immediately query the database and return a list of every known AP.



```
Connected to localhost.
Escape character is '^]'.
get-routers
00:24:6c:d0:77:00 184.5 16.4 0.0 00:24:6c:d0:77:10 184.5 16.4 0.0
```

Figure 5: “get-routers” Command

Figure 5 demonstrates the output of a Telnet session running this command and how the server responds. On the last line of Figure 5, the first entry set of numbers is the MAC address of the AP (00:24:6c:d0:77:00), followed by the X, Y, and Z coordinates (184.5, 16.4, 0.0) that are associated with it. The client also needs to build its list of rooms and thus runs a command similar to the first entitled “get-room-list.” This uses a similar formatting but instead, returns all of the rooms in the engineering building along with their associated type and coordinates. When the client is done querying the server, it will send a “quit” command which tells the server to close the connection. Also, included are commands that work with directions and differ slightly from each other which will be demonstrated later.

Database Management System & DBManager Class

The server stores all of its data in a SQL database called MySQL.[8] Like Java, MySQL can be run on many different platforms so the database can be moved from host to host just by running a few simple commands. MySQL was also chosen as the backend because it provides Java JDBC[9] libraries. In this manner, a search for finding ways to connect to the server would be unnecessary.

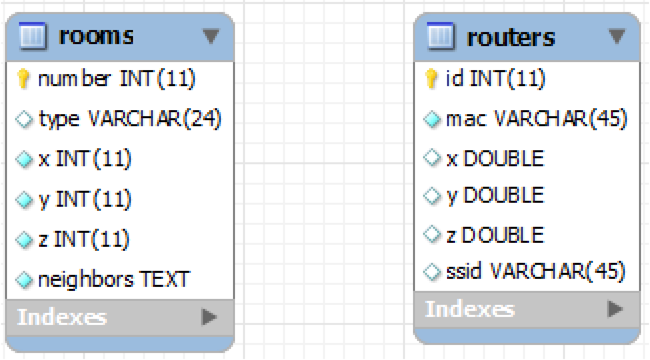
JDBC is a standard the Java uses that stands for Java DataBase Connectivity. It works in a similar manner as ODBC which stands for Open Database Connectivity, which is used directly with operating systems. JDBC is a wrapper library around the standard connection drivers which makes using different DBMSs easier. As an example, if someone wanted to program for more than just one type of SQL server, they don’t need to make two different db_connect() functions. Instead, they would just tell their JDBC connection to use a different driver and use the same db_connect() function. This will also make porting the data from one DBMS to another easier for a team that wishes to do so.

[8] MySQL. (2011). Download Connector/J. Retrieved 2011, from MySQL: http://dev.mysql.com/downloads/connector/j/

[9] Oracle. (2011). Java SE Technologies - Database. Retrieved 2011, from Oracle: http://www.oracle.com/technetwork/java/javase/jdbc/index.html

To provide more simplicity, another class was created for the server to use which wraps around the JDBC connection. The purpose for doing this was to make setting up connections easier and facilitate an easier build of SQL statements to be run against MySQL. The wrapper takes all of the settings that are required to setup the driver and directs them to the database. This makes managing database connections as easy as running two statements: db.Connect(), db.Disconnect().

With the type of data that the application will be working with, the schema setup was basic. Figure 6 shows the layout of the two tables used for the Engineering Building.



rooms
- number INT(11)
- type VARCHAR(24)
- x INT(11)
- y INT(11)
- z INT(11)
- neighbors TEXT

Indexes

routers
- id INT(11)
- mac VARCHAR(45)
- x DOUBLE
- y DOUBLE
- z DOUBLE
- ssid VARCHAR(45)

Indexes

Figure 6: Engineering Building Schema

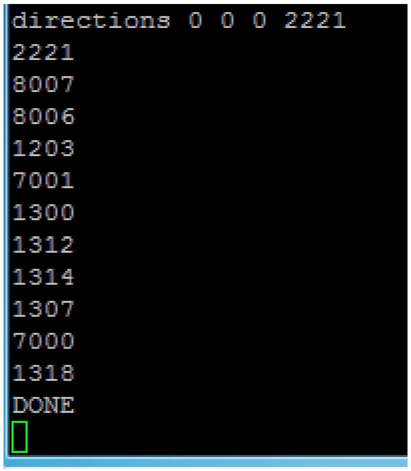
Database tables each have to have one row that will have a unique value. This is called a primary key and allows this row to be automatically indexed for fast lookups. In the room table, the primary key is set to be the room number, since each is unique, while the routers also have auto-incrementing integers.

Shortest Path Directions Calculation

One of the main functionalities of the server is its ability to calculate shortest path directions and give the client application point-by-point directions to a desired destination. The server runs an algorithm called A*,[10] which is a derivative of Dijkstra mapping algorithm. A* uses a "Best First Search" approach to find the least-cost path from a given initial node to the requested destination. What makes A* unique is its ability to use a heuristic estimate as a factor to add weight to the nodes. This can be changed depending on the application in which the algorithm is being used. For iDOCENT, this value was used to check total distance to the point-of-interest to discourage using high-cost paths which lead to the end. Another example would be when mapping from one city to another, a path that leads down a highway will be considered less costly than a path taking normal roads.

[10] A* Search Algorithm. (2011, April 20). Retrieved 2011, from Wikipedia: http://en.wikipedia.org/wiki/A*

There are quite a few defined commands that were created for the application to call. The first command is a simple directions query called “directions.” This function requires four arguments: a current location coordinate and a destination coordinate, each containing X and Y. Next, the server will build a map object from the database and create a node for each room, calculate the closest node from the location uploaded, and run the algorithm. Figure 7 shows the output of this command run from a Telnet session with the server.



```
directions 0 0 0 2221
2221
8007
8006
1203
7001
1300
1312
1314
1307
7000
1318
DONE
```

Figure 7: Example Directions Command

There are a few features implemented in the application which allow one to easily search for a route to the nearest restroom or exit. Extra commands have been implemented to easily make this functionality work. The respective commands are: “nearest-mens-restroom,” “nearest-womens-restroom,” and “nearest-exit.” These commands all take in three arguments which are the client’s current X, Y, Z coordinates. The feedback is the same as the direction coordinates shown in Figure 7.

When calculating directions it is important to note that dummy nodes were measured and added into the database to represent what are called intersections or breadcrumbs. These intersections make up a point which will be selected to comprise part of a path, ensuring that the user is not asked to walk through a wall. This also makes directions simpler to draw because, instead of connecting neighboring rooms to each other with little lines, one line is drawn from intersection to intersection instead. In turn, this long, clean line is then directly connected to a room. This also aids in calculating distances on the application portion as well as text to speech. These intersection nodes or breadcrumbs are stored in the rooms table where the type is labeled as “INTERSECTION.”

The rooms table contains a few different kinds of nodes, each with a specific purpose. To navigate from the first floor to the second floor, imaginary nodes were created and measured out for the stairs. They are created in pairs, consisting of one that rests on the first floor and another on the second. Each created node has the same X and Y coordinates with different Z coordinates. These nodes are the only nodes which have a neighbor that leads to the other floor, therefore guaranteeing that when a request is made to go to a location on a different floor, the user is directed to the closest set of stairs. The other types of nodes include classrooms, offices, and special rooms such as DECS or Sparty's Cafe. These are all treated as regular nodes that one can navigate to. The last type of node is an exit. Its primary role is only a way for the user to leave the building.

Application Frontend

Location Calculation and Navigation

The first requirement in order to use the phone's Wi-Fi capabilities for user location calculation is to activate the phone's Wi-Fi hardware. All Wi-Fi management within an Android application is done with the WifiManager class. This class provides the capability to turn on or off all Wi-Fi hardware and handle current and stored network connections on the phone. The first thing the iDOCENT application does when it starts is to activate the phone's Wi-Fi if it was not already active. The current version of iDOCENT requires that the user have a preconfigured network in the building where the application is being used. That way the phone will automatically connect to the network as soon as Wi-Fi is active. Once the phone is connected to a network it can then contact the server to download the stored locations of all the access points in the building and store that location information with the AP's physical MAC address. The MAC address is an address assigned to a device at its creation and is a static identifier making it the best way to identify individual APs.

With the locations stored in the application, iDOCENT starts a scan of all APs within range of the phone using the startScan() method in WifiManager. The Android system will perform this scan in a separate thread and then post an event once the results of the scan are available. iDOCENT has a class called ScanResultReceiver that is set to then listen for and capture that event so that it can retrieve the list of all of the APs in range. When the ScanResultReceiver receives the system's event it calls the getScanResults() method on the application's WifiManager class that returns a list of ScanResult objects that each contain all relevant information about the corresponding AP. This information includes the AP's MAC address, the name or SSID of the network that the AP is a part of, and the strength of the signal received from the AP id dBm, along with other information. The MAC addresses are then used to access the stored location of the corresponding APs. These locations are all weighted based on the signal strength of that AP and then averaged. This average is then used as the estimated location of the user. Since all navigation is only necessary within hallways, and all APs in the engineering building are only located in the hallways, iDOCENT then normalizes the calculated location to be placed in the center of the hallway where the algorithm calculates the location to be. This also simplifies the problem by changing it from a two-dimensional problem to a single-dimensional one for each floor.

Detecting which floor the user is currently on is done in the same way as determining the X and Y coordinates. Each access point is also assigned a Z value: 0 for the first floor, 20 for the second, and 40 for the third. On the first floor the calculated average z value will be between 0 and 10, the second floor will be between 10 and 30, and the third floor will be over 30. This works so well because the distance between the first and third floors is enough so that third floor APs are not within range of a phone on the first floor and vice versa.

To limit a decrease in performance while these calculations are begin performed, the AP location lookup and averaging is done in a separate thread from the main app. This means that these calculations will not affect the map drawing or any user interface performance.

The list of rooms to which the user can navigate is dynamically filled at the beginning of the app by downloading the list of rooms from the server at the same time that the AP locations are downloaded. When the user chooses a room to which the user wishes to navigate, iDOCENT sends a request to the server to give a list of nodes from the user's current location to the destination room. This list is then passed along to the drawing system to draw the path.

Drawing

iDOCENT uses the OpenGL ES[11] (OpenGL for Embedded Systems) APIs to do all of the drawing. This was decided to be the best option because it is specifically designed for fully customized drawing. The default Android APIs only allow basic image displaying and text output, which is too limited for drawing a map of a building. The map is statically oriented with north pointing up and is drawn with mainly two GraphicsObjects: lines and squares. The hallways are simply black lines and the rooms are color coordinated squares drawn on top of black and white textures of the room numbers. Blue rooms are classrooms, green is for offices, and yellow is for labs. The current version also has large blocks of grey to indicate an area that has not yet been accurately measured out to be displayed with confidence.

There is a separate Vector of GraphicsObjects for each floor to make it easy to switch maps appropriately. When the map is drawn it checks on which floor the user is currently and then sets the drawn Vector accordingly so the correct floor is drawn. This is particularly important when the user navigates from floor to floor.

[11] Bergman, Per-Erik. OpenGL ES Tutorial for Android. 3 December 2009. 2011 <http://blog.jayway.com/2009/12/03/opengl-es-tutorial-for-android-part-i/>.

Adding a RoomSquare object to a map requires several pieces of information, but is ultimately easier to describe then manually setting the dimensions of the rectangle with just OpenGL. The constructor for a RoomSquare requires nine parameters: the X and Y locations of the door, a String describing on which side of the hall the room is ("left", "right", "top", or "bottom"), a String describing on which side of the room the door is (all of the same as the previous parameter plus "middle"), the width of the room (size in the X direction), the height of the room (size in the Y direction), a String describing the color of the room, a reference to the main iDOCENT object, and the room number. The first six parameters describe where the rectangle should be drawn and how large it is. The reference to the iDOCENT object is required for texture mapping of the room number which is also passed in to tell which specific texture should be used.

This seems like a lot of information, but it is all fairly simple to obtain and, with this information, the RoomSquare can do all of the vertex calculations for the programmer. OpenGL draws a rectangle as two triangles. To draw this way requires two arrays: an array of indices and an array of vertices. The indices array describes which vertices should be used to draw the triangles that make up the rectangle. The indices must be in counter-clockwise order. The vertices array describes the actual X, Y location of each corner of the rectangle. For the rooms in iDOCENT, the indices array is set to {0, 1, 3, 0, 3, 2}, so the triangles are described as shown in Figure 8.

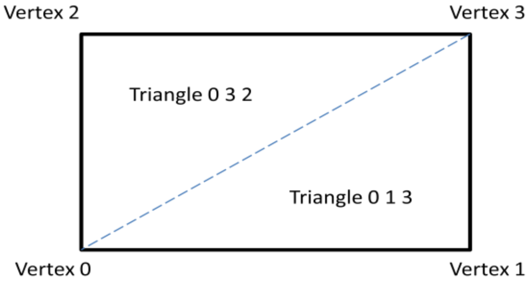


Figure 8: Triangle Drawing

Given that north is drawn as up, this means that the vertices need to be set such that the first vertex in the array is the south-west corner of the room, the second vertex is at the south-east corner, the third is at the north-west corner, and the last is at the north-east corner to maintain a counter-clockwise structure. The nine parameters passed into the constructor of a RoomSquare are used to automatically calculate the appropriate vertices for the desired room.

A RoomSquare is more than just one rectangle. It is actually three squares whose locations all need to be matched up, so the more automated the location placing is, the better.

The first rectangle is a solid white rectangle with a black and white room number texture mapped onto it. It is either drawn portrait or landscape based on the dimension ratio of the room so that the texture best fits the room. The second rectangle is a mostly translucent colored rectangle that is drawn on top of the first. An object's color is described with four parameters: red, green, blue, and alpha. The alpha parameter sets the opacity of the object. iDOCENT assigns the RoomSquare's an alpha value of 0.25 to make them translucent. When an object's color is set, it is set for the entire object. This means both the rectangle's border and fill will be that one color. This requires that a third rectangle be drawn on top of the second as just a black border so that the boundaries of individual rooms can be seen.

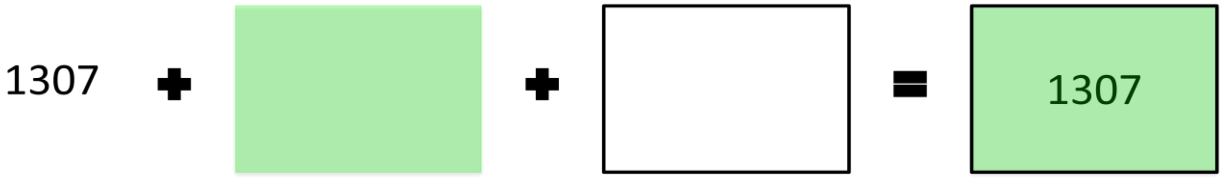


Figure 9: RoomSquare Drawing

The navigation path is drawn with a separate map object from the main map. This DirectionsMap takes in the list of nodes to the destination and creates red lines connecting adjacent nodes and stores those lines in lists. There is a list of lines for each floor so that when the user is navigating to a different floor only path lines for the current floor are shown. These lines are then drawn over the map of the hallways and rooms to display the shortest path to the user's destination. Before each line is drawn, the app determines if the user has already progressed past that line in the path. If the user has, then the line should not be drawn. This feature is used to only show the path that still needs to be traversed. The destination room's color is also changed to red so that the user can quickly see the location of that room.

User Interface

Since one of the objectives in the app is to assist the blind, it is important to note the text-to-speech, or audible feedback, features of the app. Quite a few things are automatically read aloud when the Talk Back feature in the accessibility options of the phone is active. These things include test in dialog boxes, the name of the app on the main screen, and reading the general main menu options. With these features noted, the rest of this report will only note additional text-to-speech features specifically carried out by the iDOCENT application.

When the app starts it immediately starts downloading AP and room locations from the server. So that the user is aware that this is occurring, a dialog box will come up during the duration of the download noting that information is being downloaded. This dialog box cannot be canceled by pressing the back button so that the user knows exactly when the information is available, and so that he or she cannot interrupt the download before it has finished.

The current user interface consists of several main menu options as shown in Figure 10. The Zoom Out and Zoom In buttons will zoom the map in or out so that the user can see less or more of the map respectively. The options button brings up the dialog box shown in Figure 11 which allows the user to set the volume of the text-to-speech voice and to turn off/on the display of room numbers. This second option is available because some phones cannot render the room number textures as fast as others and this can make the map look poor and distracting.

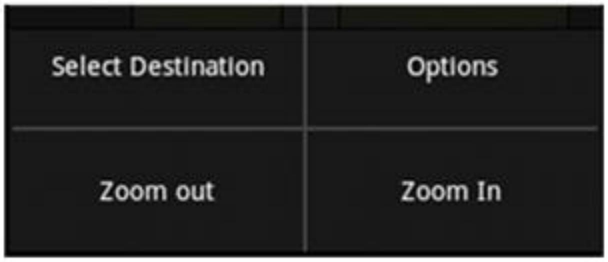
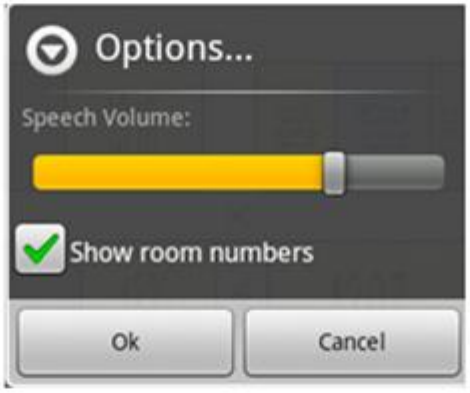


Figure 10: Main Menu



Speech Volume:

☑ Show room numbers

Figure 11: Options

The select destination menu option brings up a dialog box for selecting the room to which the user wants to navigate. Since there are a large number of rooms in a building, room selection is broken up into sections. A user operation series is shown in Figure 12 as an example of this process. First the user selects the floor of the desired room. This will bring up a new list to select within which range of numbers the room falls. A final list will then come up with all the rooms within that range. At each new list the app will read the entire list to the user if Talk Back is active when the top most option is selected (the label for that list). When a new room is selected, the app will say "Navigating to…“ and a certain room number and type for that room (e.g. classroom, lab, office). If the cancel button is pressed the app will say, "Canceling new room selection."

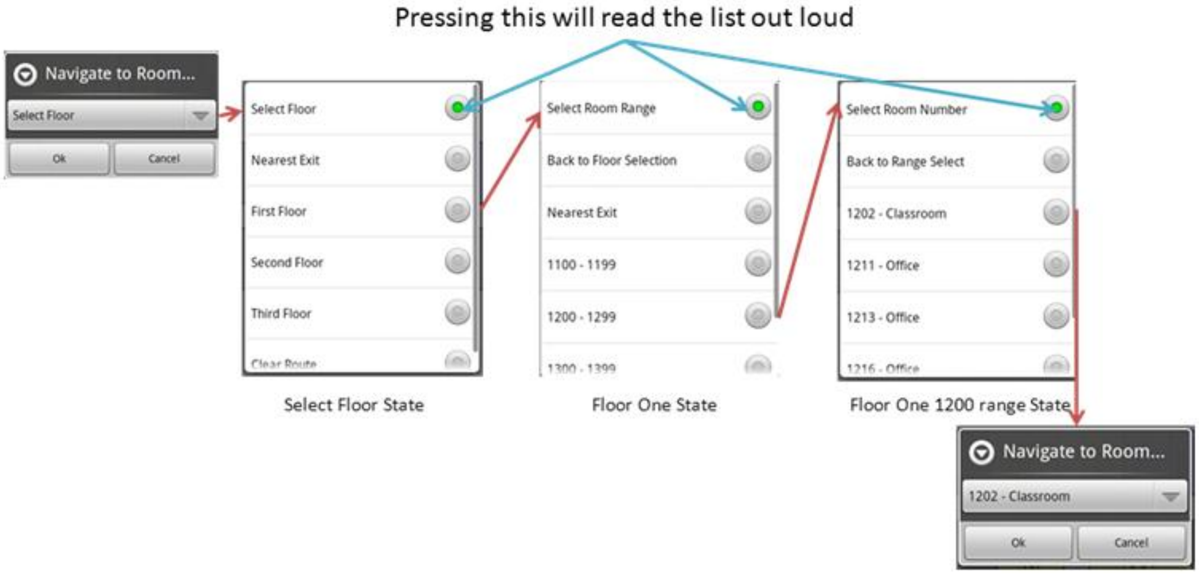


Figure 12: Navigation User Interface

The drop down element of the room selector is done with a RoomSelectSpinner object created specifically for the iDOCENT app. This class extends the basic Spinner class that acts as a drop down menu. The RoomSelectSpinner is unique because it changes which list is displayed based on which options the user selects. It quickly became apparent that iDOCENT would need a custom spinner object because the default Spinner would just close as soon as an option is selected. Which list is displayed, is handled by simple state machine where each list is a specific state and each selection changes the state appropriately. When the state is changed the window is closed and then reopened with:

```
super.onClick(dialog, 0);
this.performClick();
```

The spinner states and state transitions can be clearly seen in Figure 12 with each list in a different state and the red arrows originating at the selection that caused the state transition. When an end point is selected the list collapses back into the dialog box. An end point is any room choice or any nearest exit selection.

Lastly, the map always centers on the user's location, but the view can be temporarily moved by using the touchscreen.

Design Complications and Procedures

During this design process there were many obstacles and complications that needed to be overcome. This section will discuss the complications that arose and how they were solved.

New Language and APIs Issues

Before starting this project, no one on the team had previously programmed for a smartphone, let alone in Java. Java is the programming language used for Android devices and was a completely unfamiliar language for the programmers. This required the programmers to learn a whole new set of APIs for programming an Android device as well as write an entire application with an unfamiliar language. Fortunately the Java programming language is fairly easy to learn if one is already familiar with another higher level language like C++. There is quite a bit of documentation online describing what all of the available classes do on the Android Development[12] website, but examples on how to use some of the classes is limited. However, the site is often adding new examples to remedy this. While some methods for coding with Android APIs were readily available, others were hard to come by and often needed to be developed through some trial and error.

Interface with the server Issues

The original idea for downloading access point locations was to download the locations only when a new AP was picked up. This means the stored AP list would start out empty and whenever a new AP was detected by the app, the app would then send a query to the server to obtain that specific AP's information. This, however, caused a major problem. As the user walks down a hall he or she will eventually walk out of range of the AP that the phone was originally connected to. At this point the phone needs to tear down the connection to the network through that AP and rebuild the connection using another AP. This takes time and means that when this transitioning is occurring no new queries to the server are possible. Therefore, new AP locations cannot be downloaded until the connection is restored and the user's location cannot be updated during this time as well. To solve this issue, all information downloading occurs as the app opens when the user is not moving. This way, all information needed for the entire life of the app is readily available whenever it is required.

Location and Navigation Issues

The first issue with using Wi-Fi signal strength and AP locations to locate is user is getting an accurate location for all of the APs in a building. This information is not readily available since nobody else needs exact locations of these APs. The team went around the engineering building with a 100' tape measure and measured the AP locations to the nearest foot.

Obtaining digital copies of floor plans within the engineering building has presented several issues. Initially, GIS was contacted to try to obtain the floor plans. After several follow up emails, GIS replied with an End-User License Agreement that each of Team 2's members was required to sign in order to access the data. The files were sent in an unfamiliar format and

[12] http://developer.android.com/index.html

required an additional compiler to be programmed in order to view the floor plans. Several hours were put into coding this compiler in JavaScript with mediocre success. Only a portion of the maps could be viewed. It was determined that it would require an extensive amount of time to re-create the layouts of the engineering building and other methods were sought out.

DECS was contacted in an effort to obtain the engineering building's Wi-Fi access point locations. After meeting the network administrator, Adam McDougall, Team 2 was provided with a list of every AP's MAC address, location, and broadcasting range. Additionally, digital copies of every floor were provided.

Another issue that arose during testing of the location algorithm was that when the phone is connected to a network and the phone leaves the range of the AP to which it was connected, it will stop scanning for new APs. This is a major issue because if this scanning ceases, then the app will not be able to locate the user at all. To remedy this, iDOCENT disables all network connections during the running of the app except for when it needs to make queries to the server. When the app starts, it makes sure that it is connected to a network and then closes that connection once the AP and room location information has been downloaded. In order to restore this connection later, the app must store the ID number assigned to that network by the phone. The connection will then stay disabled until the user starts choosing a navigation destination. At that point the app will activate the previous network using that saved ID and then disable it again once the navigation route has been downloaded. Making sure the phone is only connected to a network when needed solved the issue described above. In order to return the phone to the state in which it was before the app started, iDOCENT restores the connection it disabled during its running.

With constantly varying AP signal strengths, there can sometimes be accuracy issues with this method of location sensing. To help compensate for this, iDOCENT will average multiple scan results together to limit the effect of APs near the edge of the phone's range that might be picked up in one scan and not another. There is also a very weak low pass filter that performs a weighted average of the current sensed location with the last (heavily weighted on the new location) to limit large variations in calculated location.

The A* algorithm works by checking each node's neighbors and calculating a shortest path. With the initial setup, everything on a single hall way was connected to everything. This led to having directions that would give someone a path in short increments from room to room since they were all linked down the hallway. The best way to solve this would be to find a value to add weigh to traversing from room to room to encourage using the intersections and keeping the node traversal shorter. However, this approach proved to be difficult because what would work for one hallway would not necessarily work for another. Therefore, the final approach ended up having the application parse the list of nodes and only draw lines if the next node was an intersection or destination. Ideally, filtering the results in the application was undesired so dynamically computing a number that would encourage intersection use, only for directions to just non-neighboring rooms, was required. To also help in cleaning up the results, more intersection nodes in between large sections of rooms were added so the neighboring nodes path meshes were smaller.

Drawing Issues

Similar to measuring out the location of the APs, the team also had to measure out the lengths of the hallways in the engineering building and the locations of the rooms. This was done at the same time as the AP measuring with the same 100' tape measure. Again, there is no readily available accurate map of the engineering building. There are some resources available, but their use is restricted for security reasons.

OpenGL ES is a very powerful graphics API and provides a lot of options for drawing on a phone. This results, however, in some complex code to draw exactly what is desired. There are a lot of settings that need to be enabled and disabled each time something needs to be drawn. This can be quite complicated; therefore, one major problem arose from the complex nature of the API.

Since OpenGL has no way of drawing text directly, images in the PNG format of every room number were needed to be used as textures in order to be mapped onto rectangles to be displayed. Drawing the textures for the room numbers ended up being the most complicated part of drawing. It was difficult to find a good example on how to do this easily, but one was eventually found at the NeHe Productions website.[13] After reading this code it was incorporated into the iDOCENT app in the correct locations. The original images worked fine on an Android emulation device and one of the two phones the team had at their disposal, but they didn't show up at all on the other phone. At first it was thought that phones with a specific type of processor could not draw textures at all, but when the NeHe tutorial project worked correctly this theory was proved incorrect. More investigation uncovered that the only difference between the method used by iDOCENT and the NeHe app's method was that the images used in the NeHe app had dimensions of 256x256 pixels while the iDOCENT images were all 50x15 pixels. It was then concluded that for these processors, images to be used for texture mapping needed to have dimensions that were in powers of two. The team resized all of the iDOCENT images to be 64x16 pixels and then they were properly displayed on the map.

[13] NeHe Productions. Lesson 6. 2011. 2011 <http://nehe.gamedev.net/data/lessons/lesson.asp?lesson=06>.

It is important to note that images using alpha values to make parts of it transparent at textures can cause some unexpected drawing. If the textured object is drawn first it can sometimes cause a subsequent object to be drawn transparent even if the alpha value passes to the OpenGL drawer is reset appropriately. Drawing order and the order in which OpenGL options are enabled and disabled play a major role in how the drawing behaves.

User Interface Issues

There were also some basic complications when creating the user interface for iDOCENT. Since the Android APIs were so new and unfamiliar to the team, some processes had to be learned through trial and error.

One such process was making a dialog box. The original example found was missing a key element. A dialog box can be built up using an AlertDialog.Builder object that allows the programmer to "build up" a dialog box by adding desired elements like scroll bars and buttons and provide information on what actions should be performed by those objects. An AlertDialog.Builder object can itself be shown as a dialog box, but does not show programmer defined sizes for the objects. This issue is quickly resolved by calling the create() method on the builder object which returns a fully configured dialog box that can then be displayed with the correct dimensions and configurations with the show() method. This one line of code took some time to figure out when working on the UI, but made a large difference in the overall look of the UI for iDOCENT.

Chapter 4

Testing

The first test that was done was to determine the feasibility of the first design approach: Dead Reckoning. Initially, this concept appears to have many advantages stemming from the fact that it only relies upon the smartphone and the user. This advantage was quickly rejected. A basic smartphone application was built in order to utilize the compass and accelerometer sensors of the smartphone. This basic application tested the types and accuracy of data obtained from these sensors. The concept of Dead Reckoning rests upon readings from these hardware pieces and was simulated within this test. This initial test concluded that all of these sensors suffer from great imprecision which therefore ruled out Dead Reckoning as a legitimate approach.

Since the first design approach proved unsuccessful, the second leading design approach also needed to be tested for basic feasibility. This approach was Wi-Fi triangulation of router signal strength. The first step in testing this design was to program an Android application to scan for Wi-Fi signals and obtain their respective signal strength. The success of this basic test proved that scanning for and quantifying signal strength was possible and could be refined with calculations and algorithms. Location determination within a building, let alone a classroom, was the next test.

The following requirement was to be able to acquire a user's location based on the further developed signal strength averaging algorithm. This test was conducted with at least four Wi-Fi broadcasting devices, to successfully triangulate, and their placement at defined locations around an accurately measured room. These locations were added to the program to calculate the position of the phone. Once the hardware and software were in place, the calculated location was monitored to see the accuracy and overall practicality of this system. This specific test failed to show great results due to the close proximity of the test routers. Location determination was unsuccessful because the signal strength values averaged were too similar. These results immediately led to moving the test to the hallway outside the classroom, in an effort to spread out the router's physical distance. This test approach allowed the averaging algorithm to correctly function and locate the user. After successfully completing this control case, it was determined that this approach of Wi-Fi triangulation could be applied to a larger scale implementation, such as the Wi-Fi broadcasting devices already in place in the Engineering Building or MSU Union. This would require mapping the respective building and the wireless routers to an appropriate coordinate system. Ultimately this testing was refined to an acceptable tolerance of plus or minus four feet, allowing the navigation aspect of the application can begin.

Once the building and access point coordinates were tolerable, the navigation in and directions to and from a specific location were determined by making use of Dijkstra's Algorithm. This algorithm finds the shortest path to and from specific points of interest stored in the database. This was easily tested by comparing the given path to the path already known from previous experience.

Chapter 5

Project Findings

With the conclusion of the semester and the contribution to the iDOCENT project by Team 2, many technical and non-technical aspects regarding indoor navigation were found. The objective of iDOCENT was to prototype a feasibly priced solution that would locate an individual inside of a building. Research was initially performed on a variety of different methods currently being developed for indoor navigation. While taking careful consideration into each technology, the most practical method for indoor navigation for iDOCENT's requirements is developed through Wi-Fi signal strength triangulation. It offered easily accessible and reliable hardware and the ability to use an existing smartphone to perform all necessary calculations.

The key functionality behind iDOCENT is its ability to operate without the use of a GPS signal. Instead, a modified version of signal strength triangulation was implemented to determine a user's location on a smartphone within four feet accuracy. iDOCENT addressed issues caused by weak phone service and provided navigation to different points of interest indoors. Therefore enabling location results to be given in real time and points of interests may be found on the move.

By completing all of the objectives, the iDOCENT application proved to provide the best functionality for indoor navigation.

Future Design Ideas and Suggestions

The iDOCENT app is clearly not complete. It currently only works in the engineering building, and does not draw all rooms correctly yet. These two major pieces aside, this section will discuss ideas and suggestions for the design elements of iDOCENT in the future that this team unfortunately did not have time to accomplish this semester.

Location and Navigation

The main issue with the location sensing is that it can be inaccurate at times, and is limited by the placement of the APs. There are several things that can be added to the app to hopefully help with these issues.

Dead reckoning uses the accelerometer and compass to determine the distance and direction travelled. It is possible to use this technology to determine relative location to a starting point. Dead reckoning would be a worthwhile addition to iDOCENT to handle location sensing in areas with limited AP coverage or intersections with limited Wi-Fi based location sensing accuracy. It would be difficult to use this technology alone since it only gives estimated location relative to a starting point and makes use of two sometimes inaccurate sensors to determine location. This results in compounding error as the app is used for extended periods of time. To avoid this, Wi-Fi and QR codes on rooms could be used to set a new starting point and recalibrate the dead reckoning.

The Wi-Fi algorithm's accuracy could be altered in a few ways as well. The rate at which the app scans for APs could be altered. The programmer must be careful in using this approach since there is no guarantee that scan results will be returned in the order in which they were submitted. This could bring old information into a newer calculation, thus causing more problems. Also, the number of scans averaged together could be varied to get a more solid averaging or a faster response. The perfect combination of these two alterations could result in a very accurate location sensing.

The current version of the app requires that a network in the building be already configured on the phone. It is possible to have the app actually connect to networks on its own. The information to do this is provided in the application note written by Matt Gottshall and is posted on Team 2's webpage.[14]

Drawing Future

One of the most time consuming parts of expanding this application will be in getting accurate maps and AP locations of every building. A future team will want to work with groups on campus that have extremely accurate maps of the buildings and work with the legal issues to find what can be used to expand this app. The maps can be created manually with a tape measure if needed, but this is a very inefficient way of doing it.

Currently all rooms for the map drawing are hard-coded into the app. This is not a very good method especially for expanding into more buildings. When using the app in the engineering building, the phone does not need to be cluttered with information on how to draw all of the other buildings on campus. To make this better the information needed to draw these maps can be stored on the server. It would be beneficial to develop a document in a format like XML that contains all of the necessary information and can be downloaded by the app and parsed to draw the map.

The X for displaying the location of the user is not very useful. It is not very interesting to look at and does not give any sense of the direction in which the user is facing. Accessing the phone's compass will enable the app to show which direction the user is facing and some sort of directed object could be used instead of an X.

User Interface Future

The most important changes that will need to be made to the UI involve making the app more easily accessible to people with disabilities.

Since this application is to be used by visually impaired users, more text-to-speech functionality will need to be added. Currently, the list of rooms is split up to make each list smaller and easier to remember when the list is read by the phone. However, some of the lists will still be unacceptably long for listing to, with text-to-speech. Some more splitting can be done by categorizing the choices by room type or even by splitting the room ranges into smaller chunks than 100. Some trial and error will be required to find the best UI, and there are a lot of people at RCPD that will be able to help find the best way to do this.

The current version of iDOCENT accepts all input the first time it is selected. This should be edited so that if a visually impaired user accidentally selects the wrong item he or she will not need to go back and find the correct selection. The app already has the capability to check if accessibility options are turned on the phone, so it will be relatively easy to add these

[14] http://www.egr.msu.edu/classes/ece480/capstone/spring11/group02/documents/default.html

functionalities only for users who want them. Also, the room selection lists currently require the user press the top option to have the app read the list aloud. It may be preferred that the list be read once the list opens. This is a question that should be asked to the RCPD.

Internet Security

Currently the application communicates with the server through the internet. For the most part, there is no potential harm caused by the information that is being sent because a completely unique, self-designed mapping system is being used. This causes no harm because it would not make sense to anyone who could get a hold of this information. If, however, everything was mapped in GPS coordinates, it would be possible to snoop on users current locations. To take care of this, it would be easy to add SSL protection to the connection between the application and the server. SSL[15] works by using a certificate that is signed and verified by a third party in order to encrypt the traffic flowing down public networks.

Current connections to the server do not require any kind of verification and does not check for suspicious activity. If we used a more standardized coordinate system, it would be easy to figure out how to create a connection to the server and try running commands to extract information from its database. The need to make the server smarter and able to suspect attack and close connections on anything that seems suspicious would be a must.

Database Dependence

Currently the database is very basic and only holds room and access point information for one building. The current map is hard coded into the application to be rendered by the OpenGL graphics system. This is not a very good idea if this application were to be expanded to support multiple buildings. The map is made up of a bunch of lines and it would be easy to store this kind of information off the application to only be fetched when needed. By storing this information, the application footprint would be physically smaller and updating maps would be made easier. This also goes along with the idea that the creation more detailed maps needs to be streamlined.

New Environments Future

While Android is quickly becoming more popular, the iPhone is still more widely used. This, plus the fact that Apple has more advanced accessibility options for people with disabilities means that it would be beneficial to have a version of iDOCENT that worked on an iPhone. Ultimately, there should be a version of iDOCENT for every available platform, so that anybody may use it.

[15] Transport Layer Security. (2011, April 12). Retrieved 2011, from Wikipedia: http://en.wikipedia.org/wiki/Secure_Sockets_Layer

QR Codes

QR codes, or Quick Response codes, are quickly becoming commonplace in everyday life. Products all over will have QR codes printed on them that can be scanned by a phone to give information about, or direct the user to a webpage about, the item being scanned. Since QR codes are so useful, there are plans to place them on the rooms of the engineering building with information about that room. It would be useful to add the ability for the iDOCENT to read QR codes so the user can scan these codes as they are navigating the halls to learn more about the building. These QR codes could also serve as a calibrator for dead reckoning. Additional information can be obtained in the Application Note[16] written by Team 2's Andrew Kling.

[16] http://www.egr.msu.edu/classes/ece480/capstone/spring11/group02/documents/default.html

Appendix 1

Gordon Stein - *Manager*

As previously stated, there are five areas in the design process which outline the critical path for completion of the iDOCENT smartphone application. In order to accomplish indoor navigation, location determination, mapping, dead-reckoning, navigation, and a database server must all be programmed and extensively tested. I was in charge of the obtaining and mapping the data and to design the overall coordinate system that facilitated location and navigation.

This started with contributing to the team brainstorming and researching basic ideas that would provide indoor navigation. After the research of various methods was conducted, a final design approach was decided and the applicable data needed to be obtained. Obtaining digital copies of floor plans within the Engineering Building for location determination and navigational routes presented several issues.

Initially, I contacted GIS to try to obtain the floor plans, but after several follow up emails and phone calls, the files sent would require an extensive amount of time to re-create the layouts of the Engineering Building and I determined that other methods were necessary. Instead, DECS was contacted in an effort to obtain the engineering building's Wi-Fi access point locations for a more manual mapping approach. After meeting the network administrator, Adam McDougall, I was provided with a list of every AP's MAC address, location, and broadcasting range. Additionally, digital copies of every floor were provided.

After finally gaining the data needed, I created the coordinate system that would be used by the application. This coordinate system needed to facilitate accurate measurements and scale, along with providing easy to store and query data for the database. Every access point in the Engineering Building was measured by hand, with a tape measure, in feet, and then converted into the appropriate coordinate system. The respective coordinates and type of each classroom, lab, and office was stored in the database for use by the application.

Matthew Gottshall - *Documentation*

I was in charge of programming the iDOCENT app and communication with the server from the application side.

This started with implementing the initial Wi-Fi access point signal strength based location finding that was discovered by the team. Once it was shown that this algorithm would work from initial testing, the team measured out the engineering building so that I could add the buildings access points into the app (which were eventually added to the server by Jake), and start drawing a map of the building within the app. Jake suggested using OpenGL to do the drawing and showed me how to use OpenGL APIs. I was then able to start drawing the map. I used lines for the hallways, and once the team got the room locations measured out I used three rectangles to draw rooms. Since OpenGL does not have text drawing available I had to research how to draw text for the room numbers in OpenGL. I discovered that the best way to accomplish this is to map images as textures onto regular objects that OpenGL can draw. I decided that to get the desired room drawings on the maps I would require three rectangle objects as described in the report.

Once Jake had the server running, I had to add the functionality to the app to connect to and download the locations of the rooms and access points from the server. I had to manage these connections and all of the location calculations without interfering with the map drawing and user input. This way the user can still see the map updating in real time and interact with the app to choose a new destination or change the current view of the map. This required that I use new threads to execute these downloads and calculations.

The last major element of the app was the user interface. I had to work with the app a lot myself to find the best way to handle the user interface to make it easy to use and look good. This took a lot of time, testing, and careful thought to come up with the current initial user interface.

Andrew Kling – *Presentation Preparation*

There are four particular areas in which I made significant technical impact on Team 2's project this semester: smartphone user location methods, QR code research, graphics design, and video capturing and editing. These key technical contributions provided a more streamlined process for the design team and were necessary in order to complete the project.

I discovered and performed the initial research for the ideas outlined in Dead Reckoning and Wi-Fi triangulation. Initially, I proposed to our team and facilitator that Dead Reckoning would be the most practical approach to locate a user on a smartphone because it does not rely on any external sensors or signals. It also appeared to be a method that would fit into the time constraints of the semester. After learning about the significant error compounding issues and additional correction algorithms, I discovered the ideas behind Wi-Fi triangulation and wrote a flowchart of how the algorithm operates. This was the chosen method in which Matt Gottshall programmed a majority of the source code.

I also performed research on how to implement QR codes into our overall design, as it became a design objective requested by our facilitator later into the semester. I found out the cost of deployment and researched open-source code in an effort to add it to iDOCENT.[17] Time had become a restraint towards the end of the semester and this technology was decided to be unfeasible for the semester.

Graphic design was necessary for developing a friendly, visual experience within iDOCENT. I created the early map design for the application before OpenGL was used. In addition, I created all of the room number graphics for the final design.

Finally, I was in charge of capturing and editing the videos taken to demonstrate the functionality of iDOCENT on Design Day. With the help of Gordie Stein, I designed rig to film stable video of the Android smartphone during real-time navigation. Also, I proposed an additional camera be used to film the hallway and room numbers that the user is passing. After syncing the two videos and displaying them in a split-screen fashion, the viewer will be able to understand the true operation of how iDOCENT works without being in the Engineering Building.

[17] http://www.egr.msu.edu/classes/ece480/capstone/spring11/group02/documents/andrew_appnote.pdf

Jacob D'Onofrio - *Webmaster*

Over this past semester, I have helped not only research, but also program pieces of the iDOCENT navigation programs. My main focus was the iDOCENT Server backend. I used a personal server of mine to host the application during its development. The server is a 64 bit Linux machine that is also hosting the database. Java was a relatively new language to me so I had to research the proper libraries and the best practices of creating a basic stable server to relay data over the internet. The server stores all of its persistent data on a SQL database managed by MySQL. I designed the schemas that are currently in place for the database and have also populated the tables with the data that we collected pertaining to the engineering building.

To make it easier for the next team to pick up where we have left off, I have left plenty of comments and generated "javadoc" web pages that have been put up on the team web site. Although most of the application was designed solely by the other programmer, I helped in the initial design of the 2D map along with coding the original version of it in the application. I have also helped in setting up the android application's network communication.

Design Team 2

Appendix 2

References

A* Search Algorithm. (2011, April 20). Retrieved 2011, from Wikipedia: http://en.wikipedia.org/wiki/A*

Android Developers. 2011. 2011 <http://developer.android.com/index.html>.

Bergman, Per-Erik. OpenGL ES Tutorial for Android. 3 December 2009. 2011 <http://blog.jayway.com/2009/12/03/opengl-es-tutorial-for-android-part-i/>.

Carboni, Davide. "Indoor Mobile Pedestrian Navigation System." n.d. CRS4. January 2011 <http://geoweb.crs4.it/lib/exe/fetch.php?id=indoornavigation&cache=cache&media=paper_mobile_hci_2010.pdf>.

CRS4. Indoor Navigation System Prototype in a modern smartphone (No radio beacon, no GPS, no WiFi) . 30 September 2010. January 2011 <http://www.youtube.com/watch?v=HTZFRDBdQck>.

MySQL. (2011). Download Connector/J. Retrieved 2011, from MySQL: http://dev.mysql.com/downloads/connector/j/

NeHe Productions. Lesson 6. 2011. 2011 <http://nehe.gamedev.net/data/lessons/lesson.asp?lesson=06>.

Oracle. (2011). Java SE Technologies - Database. Retrieved 2011, from Oracle: http://www.oracle.com/technetwork/java/javase/jdbc/index.html

Patil, Abhishek Pramod. "Performance of Bluetooth Technologies and Their Applications to Location Sensing." 2002. January 2011 <https://www.msu.edu/~patilabh/research/Thesis_Report.pdf>.

Transport Layer Security. (2011, April 12). Retrieved 2011, from Wikipedia: http://en.wikipedia.org/wiki/Secure_Sockets_Layer

Wikipedia. Dead Reckoning. 3 January 2011. January 2011 <http://en.wikipedia.org/wiki/Dead_reckoning>.

Work, Eric. "UW Campus Navigator: WiFi Navigation." July 2003. Electrical Engineering Department, University of Washington. January 2011 <http://uwcampusnav.sourceforge.net/WifiNav.pdf>.

Appendix 3

Business Start-up Plan

Description of Industry

iDOCENT is an indoor navigation system looking to take the market by storm. This new navigational application is looking to emerge into a completely new industry that is expected to experience plenty of growth. iDOCENT can be considered to take stake into many different industries, yet the industry of GPS-like devices is where Team 2 considers its greatest competition.

One of the main components that need to be considered before entering into a new industry is the competition that already has a hold of the industry. Without any major established companies in this industry, Team 2 has the advantage.

Another factor for consideration before committing to an industry is the market forecast and future outlook or trends. Currently, the field of GPS devices is on the rise and looking to grow substantially within the next few years. Technology and Engineering jobs specifically are expected to grow in the next decade which shows great potential for new and emerging products. This industry will also continue to grow as the advancement of technology continues to show drastic improvements in becoming more rugged, providing faster processing, and improving wireless applications. The industry of indoor navigation and the iDOCENT application will definitely highlight these developments.

Financial Plan

One of the most important aspects to a business plan is the overall planning of the finances. The financial plan that follows shows that the iDOCENT application is not only a great idea, but an opportunity with real potential.

In order for the financial plans of Team 2 to be fully realized, only one assumption was made. Team 2 assumed that the funds needed for startup would be achieved through loans granted from banks totaling in an amount of $60,000. This assumption was made based on the idea that each initial member of Team 2: Andrew, Jacob, Matt, and Gordie could take a loan out for $15,000. This amount seemed reasonable considering this amount is close to the amount of tuition needed for the university and should be easily attainable.

The costs incurred by developing the software application will sum to next to nothing for Team 2 to produce. By making use of many open-source navigation processing materials and coding the smartphone application from scratch, Team 2 can produce the software element for no cost. Making modifications to the existing code to suit iDOCENT system updates will only consume time. The only cost incurred regarding software was the price of the website domain.

Other costs incurred include: the costs for the State of Michigan's basic business startup (Sources: www.michigan.gov/businessstartup), the basic costs for online advertising for 6 months (Sources: Advertisement), and the cost for Android Market application posting.

By selling the application, advertisement space within the application or by selling the patented technology behind the application, iDOCENT proves to be very profitable.

Table 5: Financial Analysis

Financial Analysis	
Description	**Amount**
Loans	$60,000.00
Patent Costs	-$15,000.00
Website Domain and Android Market	-$32.00
State Business Startup	-$65.00
Price per app download	$30.00
Potential revenue (assuming 10,000 sold)	$300,000.00
Break-even units (to cover all loans)	2000
Profit (after 10,000 units sold)	$240,000.00

Overall, this financial analysis not only states the true economic potential of iDOCENT, but also solidifies the fact that Team 2 and the iDOCENT application is an opportunity with real potential.

Organizational Plan

In order for a company to be successful, the company has the need to define an organizational plan in which the business structure and the role and responsibilities of the members are duly noted. Team 2 currently is a Limited Liability Company (LLC) where the company is represented as a number of members that have liability protection (Sources: Company). This type of business allows for more flexibility and the distancing of a company from the owners, yet creates complications as the business grows. The corporation business structure may suit Team 2 as it continues to expand.

Currently, the initial members of Team 2 are: Andrew Kling, Jacob D'Onofrio, Matthew Gottshall, and Gordie Stein. Team 2 holds to a matrix type structure where these members have equal stake in the company and share an equal part of the responsibilities. This matrix structure functions best in smaller business ventures in the beginning and eventually gives way to a more hierarchical structure as the company expands underneath each initial member. Each initial member of the management team has a background in Electrical Engineering or

Computer Engineering at the undergraduate level. All members have at least covered classes in basic economic principles in addition to the hardware/software engineering core. While each member has an equal part in the company, however each member does contribute their own expertise. Hardware and marketing will be the focus of Andrew Kling, software and advertisement will be the focus of Matthew Gottshall, database support and navigation will be the focus of Jacob D'Onofrio, and production and business solutions will be the focus of Gordie Stein.

Appendix 4

Reference

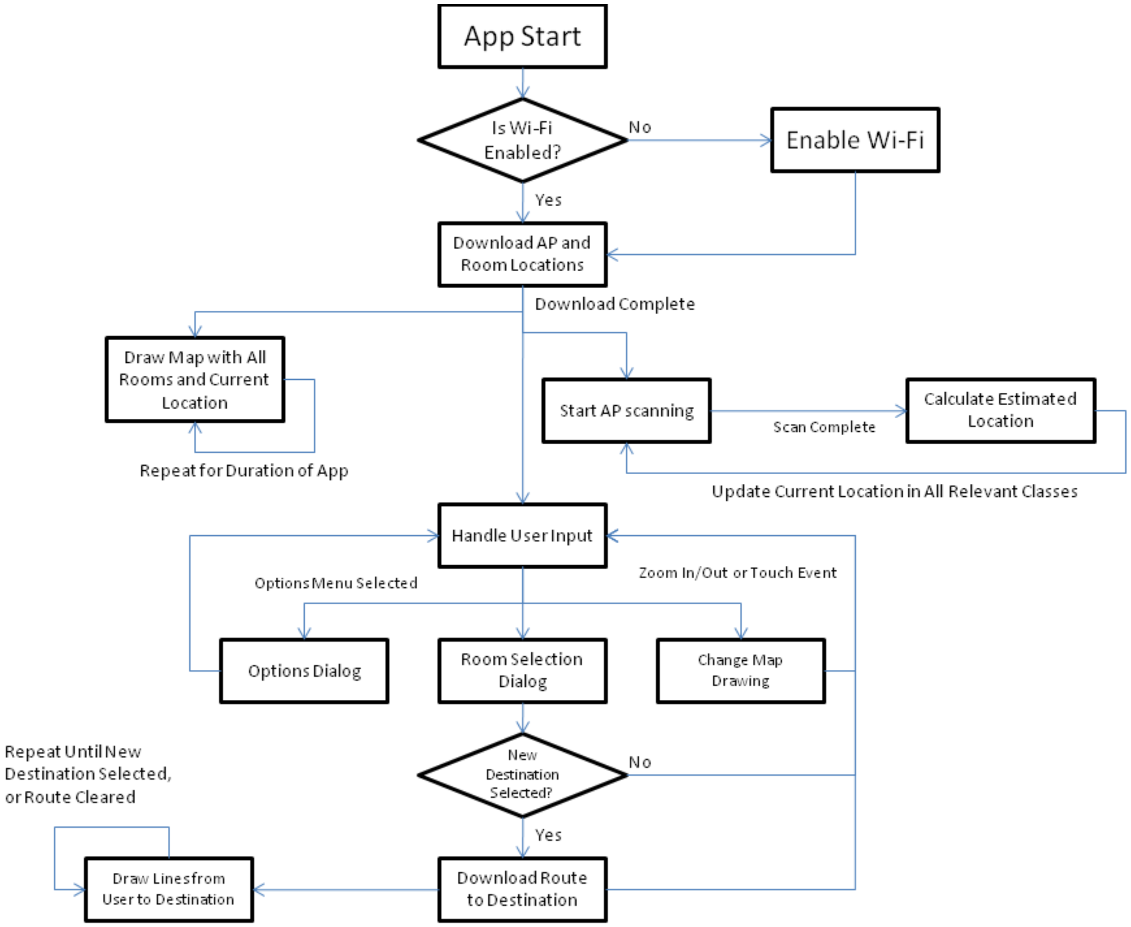

App Start
Is Wi-Fi Enabled?
No
Enable Wi-Fi
Yes
Download AP and Room Locations
Download Complete
Draw Map with All Rooms and Current Location
Repeat for Duration of App
Start AP scanning
Scan Complete
Calculate Estimated Location
Update Current Location in All Relevant Classes
Handle User Input
Options Menu Selected
Zoom In/Out or Touch Event
Options Dialog
Room Selection Dialog
Change Map Drawing
Repeat Until New Destination Selected, or Route Cleared
New Destination Selected?
No
Yes
Draw Lines from User to Destination
Download Route to Destination

Figure 12: iDOCENT Flowchart

Appendix 5

Source Code

To obtain the source code used in the iDOCENT Android smartphone application and iDOCENT server, please visit Team 2's website and access the Documents[18] page.

[18] http://www.egr.msu.edu/classes/ece480/capstone/spring11/group02/documents/