INFOmap

Indoor Navigation For the Optically-challenged

Project Part III: Design

March 8, 2010

University of Nevada, Reno
Department of Computer Science
Instructor & Advisor: Eelke Folmer

Team 01:

Kevin Grant
Angela Proffitt
Alex Tam

Table of Contents

1 Abstract

INFOmap is a handheld navigation software primarily focused on helping blind patrons navigate the interior of buildings. It utilizes voice command and crowdsourcing to build a database of objects within a building. A map of the building can be downloaded off the internet and the user can then enter key items such as vending machines, restrooms, and chairs through voice and photo. Since the information is collected from a variety of users, the data will be compared to other entries to confirm the object and its location to prevent false information. As more objects are entered, the more accurate and fully navigational the area will become. While INFOmap is focused on helping the blind, it can also be used for any user unfamiliar with a building.

2 Introduction

INFOmap is a handheld navigator. The project is an extension of a cell phone navigation application Kevin Grant and Alex Tam have been developing for Dr. Eelke Folmer and IBM. The application is designed to help blind patrons and users unfamiliar with specific areas navigate and find objects and destinations.

Users can map anything, such as a park or building, by running the application while walking. A map of the area is uploaded onto the phone via internet connection through the phone or by connecting to the computer. Once the patron has a map of the area, the user will then be able to enter items into the INFOmap database. The user will choose a room to enter items. If the user is in non-blind mode, the user can point at an area of the room and enter the object by voice command or by selecting the object using the 'selection option'. If the user is in blind mode, the user can simply add an object into the room by voice command. The specific location within the room will not be handled. The user can also delete objects in the room by voice command or by selecting the object by pointing at it and deleting the object. At a later stage of the project, the user can also take pictures of the objects to create a compiled image of the whole room.

The main characteristic of the project is that INFOmap uses "crowdsourcing" to collect data. Crowdsourcing is the practice of outsourcing work to a group of people, usually in the general public, that is interested in the work. Anybody running INFOmap can enter objects into the database. The entries will be compared to confirm location and name accuracy for each object. The idea is to get as many entries as possible, thus making the database more accurate and filled with objects. Once INFOmap is filled with objects such as restrooms, vending machines, and chairs, a blind user or someone unfamiliar with the area can ask the phone where a certain object is. INFOmap will then navigate the user to the desired objects by giving visual and audible directions. INFOmap will be using the Android platform, which is based on Java.

Blind patrons often have a hard time finding rooms in an unfamiliar building. Using INFOmap, he/she can ask the phone where room A is, and INFOmap will give audible directions to room A making traveling for the blind much easier and less stressful. INFOmap is useful for anybody; a user no longer has to roam around the building and ask other people where the nearest restroom is. Another useful application would be to navigate through a theme park. The user can ask where a certain ride is, and INFOmap will direct him/her to the ride.

One of the biggest challenges will be to control crowdsourcing. Having unreliable sources may mean unreliable data. The problem is prominent in crowdsourcing websites such as Wikipedia. Since anyone can contribute to data entry, there can be false or mistaken information entered into the database. We will have to develop a method to compare entered data to confirm accuracy. We will also have to figure out in which manner to hold data and map out the buildings. Will the coordinates be taken where the data is entered or will user enter coordinates? If the coordinates are taken where the user is standing, will the person have to be on the object? With multiple photo entries, which photo will be chosen to use for mapping? What if the photo is unrelated, how will we detect its accuracy? How can we motivate people to enter data into the database? With no entries, crowdsourcing will not work. If the user is in a loud room, how will voice detection work? Will we need to implement manual entry? With a project this big, we are bound to run into many problems.

Beyond CS 425, INFOmap can be developed to become the largest database of directions. All the buildings will be connected in a full scale world-wide navigation similar to Google Maps. However, INFOmap can also navigate inside buildings. It can also become a social network application to find friends, lovers, or reconnect with lost family members by adding even yourselves into the database, not just objects.

Professionally, this project will expose all the team members with programming on a mobile device. Many companies are interested in crowdsourcing and mobile programming, and thus will be excellent to have on a résumé. Since this project has many social uses and has the potential to grow into a full scale application, it has the potential to become a grad thesis or even start a small company.

3 High –level and Medium-Level Design

3.1 High-level Diagram

Figure # shows the layered architecture pattern for INFOmap. The interface is part of the presentation. The user interacts with the interface which will use services such as speech recognition and data I/O to access the database to send and receive data about a room.

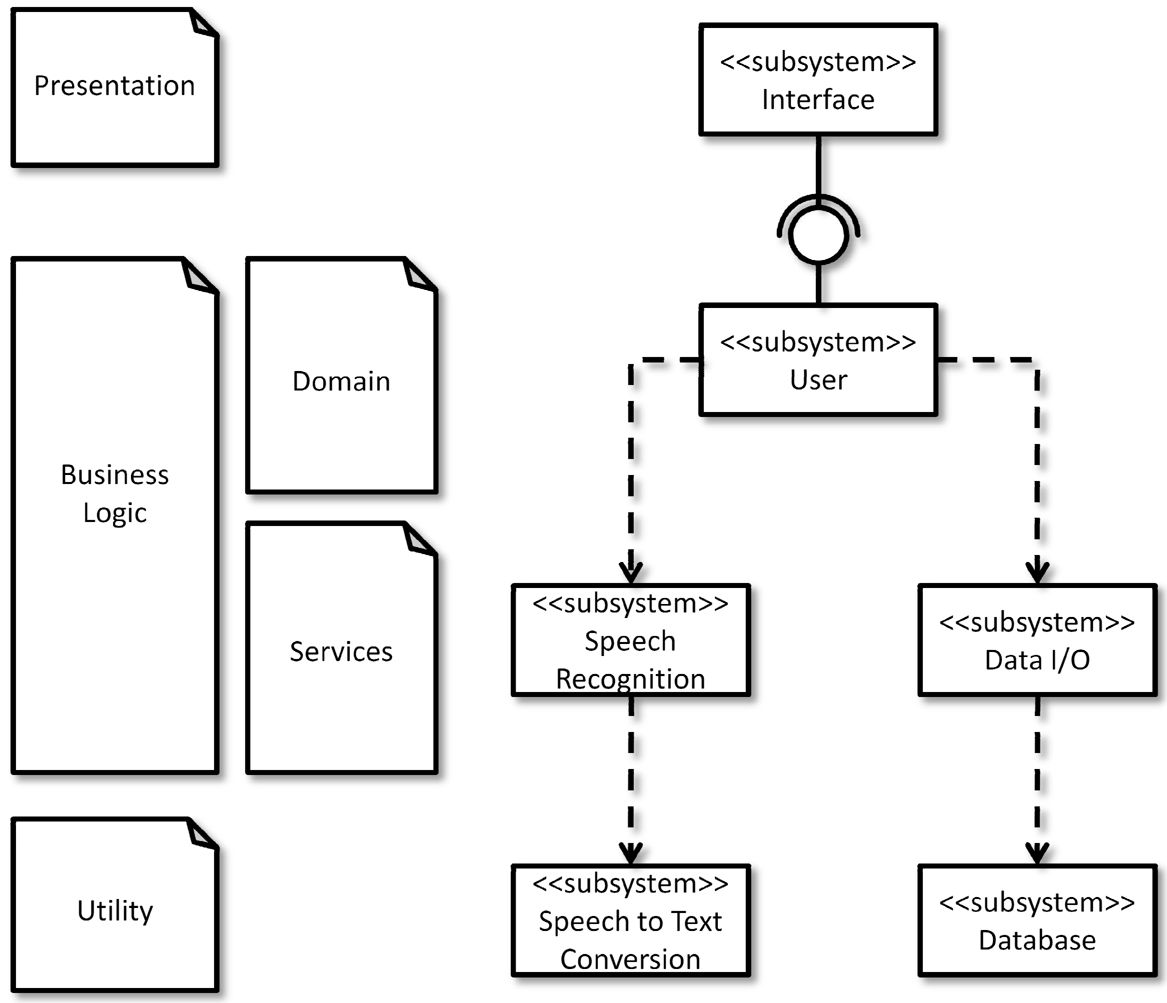


Figure #: Layered architecture pattern.

3.2 Class Diagram

The class diagram contains eight classes: UIHandler, Map, MapObjectList, MapObjectNode, MapObject, DataHandler, VoiceHandler, and DrawMap. Figure 2 shows the class diagram for INFOmap.

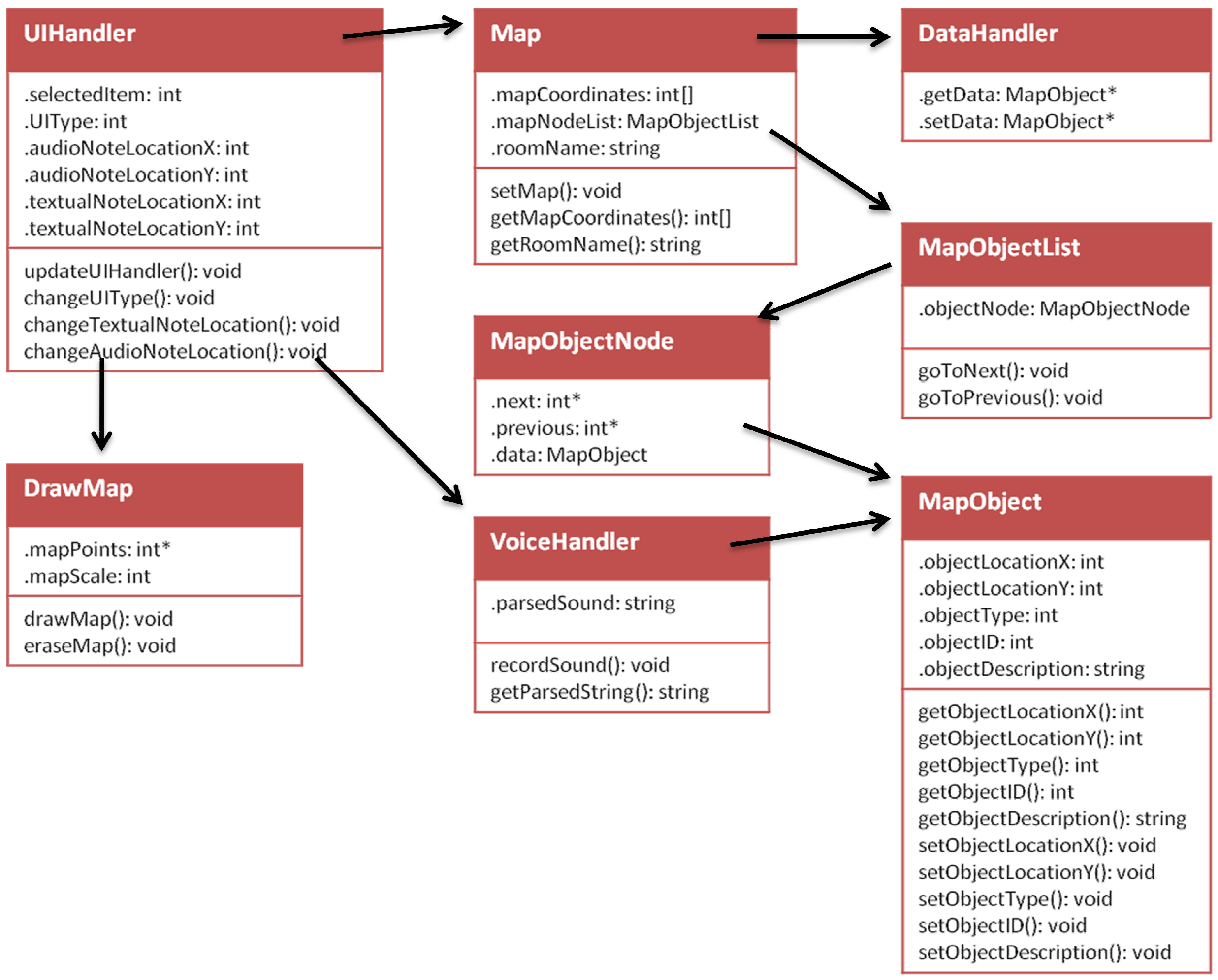


Figure 2: Class diagram for INFOmap.

3.3 Class Method Descriptions

UIHandler

Private Variables

.selectedItem: int
.UIType: int
.audioNoteLocationX: int
.audioNoteLocationY: int
.textualNoteLocationX: int
.textualNoteLocationY: int

Public Functions

updateUIHandler(): void

This function uses the classes UIType to determine which portion of the UI to show and display. This can vary drastically depending on which part of the UI the user is in.

changeUIType(): void

This function takes an integer from the main class and changes the UIs variable type. The UI will then be updated on the next call of UpdateUIHandler

changeTextualNoteLocation(): void

Dynamically depending on where the map is filling up the screen, some of the locations of the UI buttons will need to be changed as not to obscure the map elements. This function handles the positioning of the Textual note location.

changeAudioNoteLocation(): void

Dynamically depending on where the map is filling up the screen, some of the locations of the UI buttons will need to be changed as not to obscure the map elements. This function handles the positioning of the Audio note location.

DrawMap

Private Variables

.mapPoints: int*
.mapScale: int

Public Functions

drawMap(): void

This function is called continuously as the program is running, it is a java draw function call that will handle all of the map parameters that are passed to it.

eraseMap(): void

Erase map will blank out the map from the screen so that no map will be visible. In doing so it will also delete the currently saved map from memory to free up RAM on the device.

VoiceHandler

Private Variables

.parsedSound: string

Public Functions

recordSound(): void

This function will be a system call to the devices built in voice processing software. Using the functions available to us we will record the sound, process it and depending on the command we received, perform various actions to control the system.

getParsedString(): string

getParsedString will return the actual string that was parsed. once we have this parsed string, we can make decisions for the program based off of the string that was parsed. For instance, if the word "menu" is parsed, then we know to bring up the menu, if the word "back" is parsed, then we know to return to the previous screen. If an unknown word is parsed, we will know to ask the user to repeat their request.

Map

Private Variables

.mapCoordinates: int[]
.mapNodeList: MapObjectList
.roomName: string

Public Functions

setMap(): void

SetMap will take in an array of coordinates that map out all of the corners of the room. This array will be declared dynamically as to save on memory space. It is important for it to be dynamic sited of static because various room have varying numbers of corners and points

getMapCoordinates(): int[]

This function will return what the actual coordinates of the map are so that they can be used for determining location

getRoomName(): string

Return the physical name of the room so that a person will know exactly what the name of the room they are in. It is returned as a string instead of a number in case different buildings have unique naming schemes for their rooms.

MapObject

Private Variables

.objectLocationX: int
.objectLocationY: int
.objectType: int
.objectID: int
.objectDescription: string

Public Functions

getObjectLocationX(): int

Returns the X location of this specific object

getObjectLocationY(): int

Returns the Y location of this specific object

getObjectType(): int

Returns the type of this object (Chair, door, window, misc, etc)

getObjectID(): int

The objects specific and unique ID. This will be generated through a timing algorithm so that no 2 IDs will ever be the same.

getObjectDescription(): string

This is the note that a user has left for this specific object. This is left as a string and will also be tied to an audio note that is saved in an off-location database.

setObjectLocationX(): void

Used in setting the objects X location, when creating a new object

setObjectLocationY(): void

Used in setting the objects Y location, when creating a new object

setObjectType(): void

Used in setting the objects Type (Chair, door, window, misc, etc), used, when creating a new object.

setObjectID(): void

Used in setting an objects ID. This ID will be based off of a timing algorithm so that no 2 objects have the exact same ID.

setObjectDescription(): void

This function sets a description for this object. The description can be of any length

3.4 Database Table

The record for the database is very straightforward. The primary key for the record is an identification number consisting of a concatenation of the room number and the item number. The item number is an arbitrary sequential number as items are added to the room. The record also contains the item's name and coordinates. Table 1 shows an example of a record.

Room Item ID	Item Name	Item Location X	Item Location Y

Table 1: Online database table.

4 Detailed Design

Figure # shows the activity diagram for the DrawRoom function which gets the room item's coordinates and draws the room on the screen (describes it to a blind person). Figure # shows the activity diagram for the AddRoomItem function which gets the room item's coordinates and the audio or text note about the item and then sends the data to the database. Figure # shows the state diagram for the entire program.

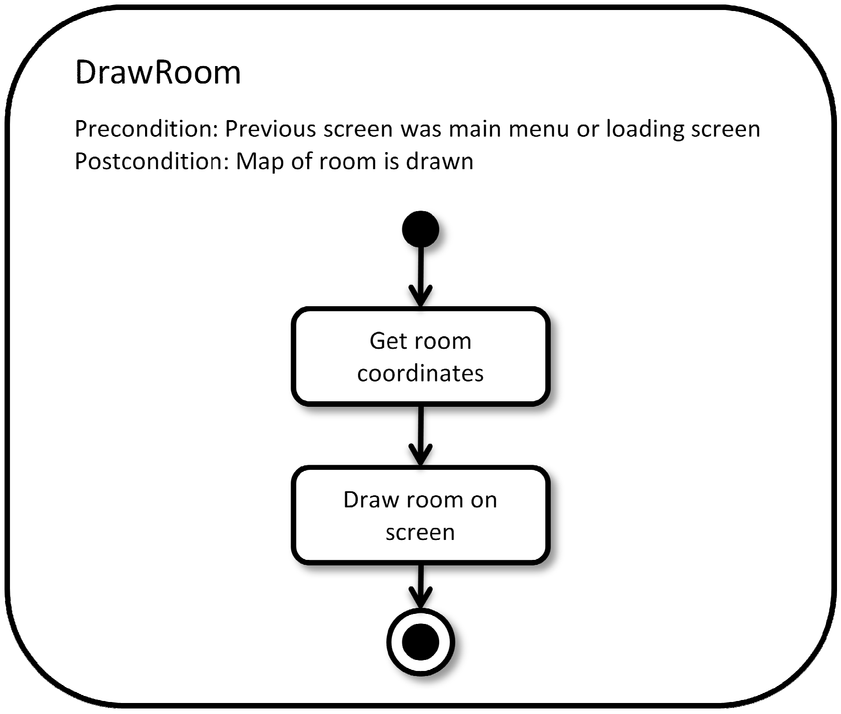

DrawRoom
Precondition: Previous screen was main menu or loading screen
Postcondition: Map of room is drawn
Get room coordinates
Draw room on screen

Figure #: Activity diagram for DrawRoom function.

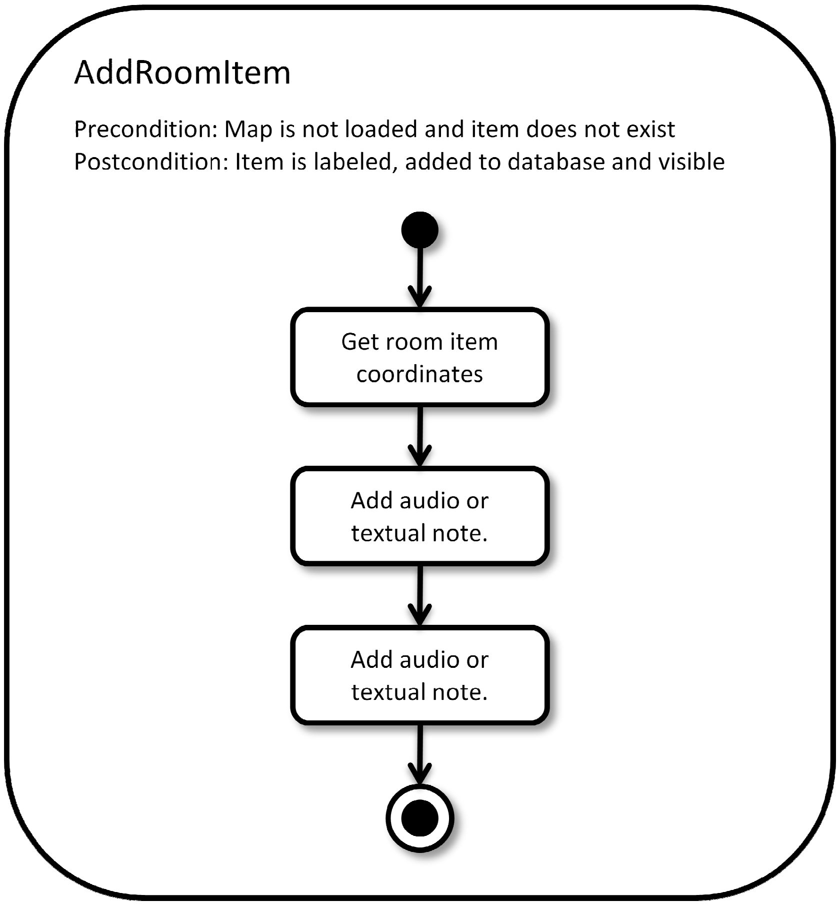

AddRoomItem
Precondition: Map is not loaded and item does not exist
Postcondition: Item is labeled, added to database and visible
Get room item coordinates
Add audio or textual note.
Add audio or textual note.

Figure #: Activity diagram for AddRoomItem function.

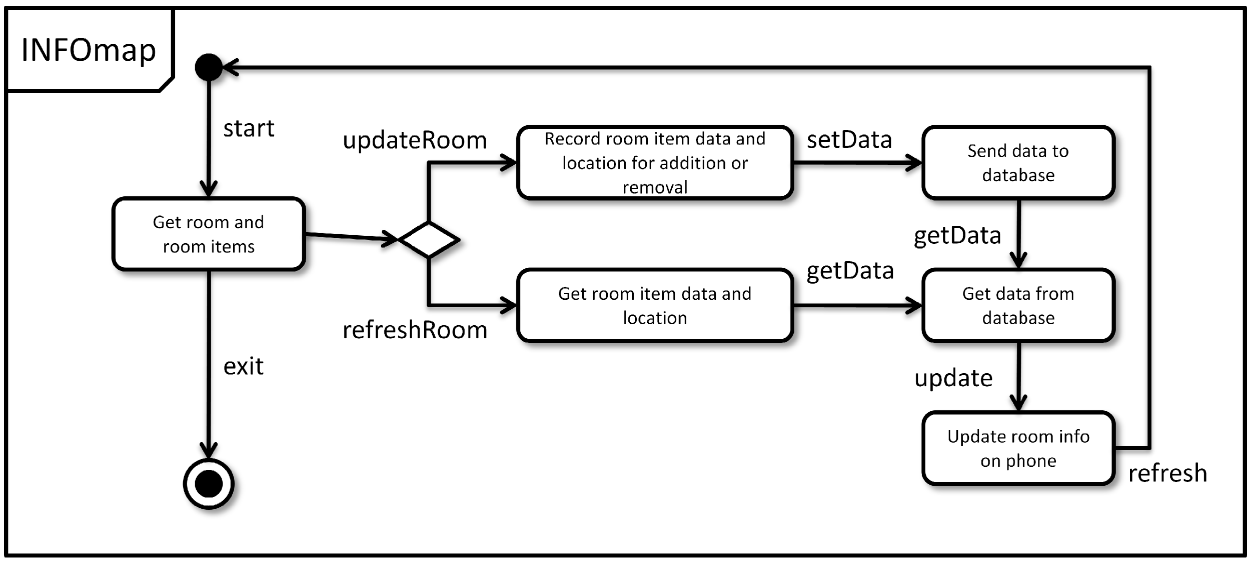


Figure #: State diagram for INFOmap.

5 User Interface Design

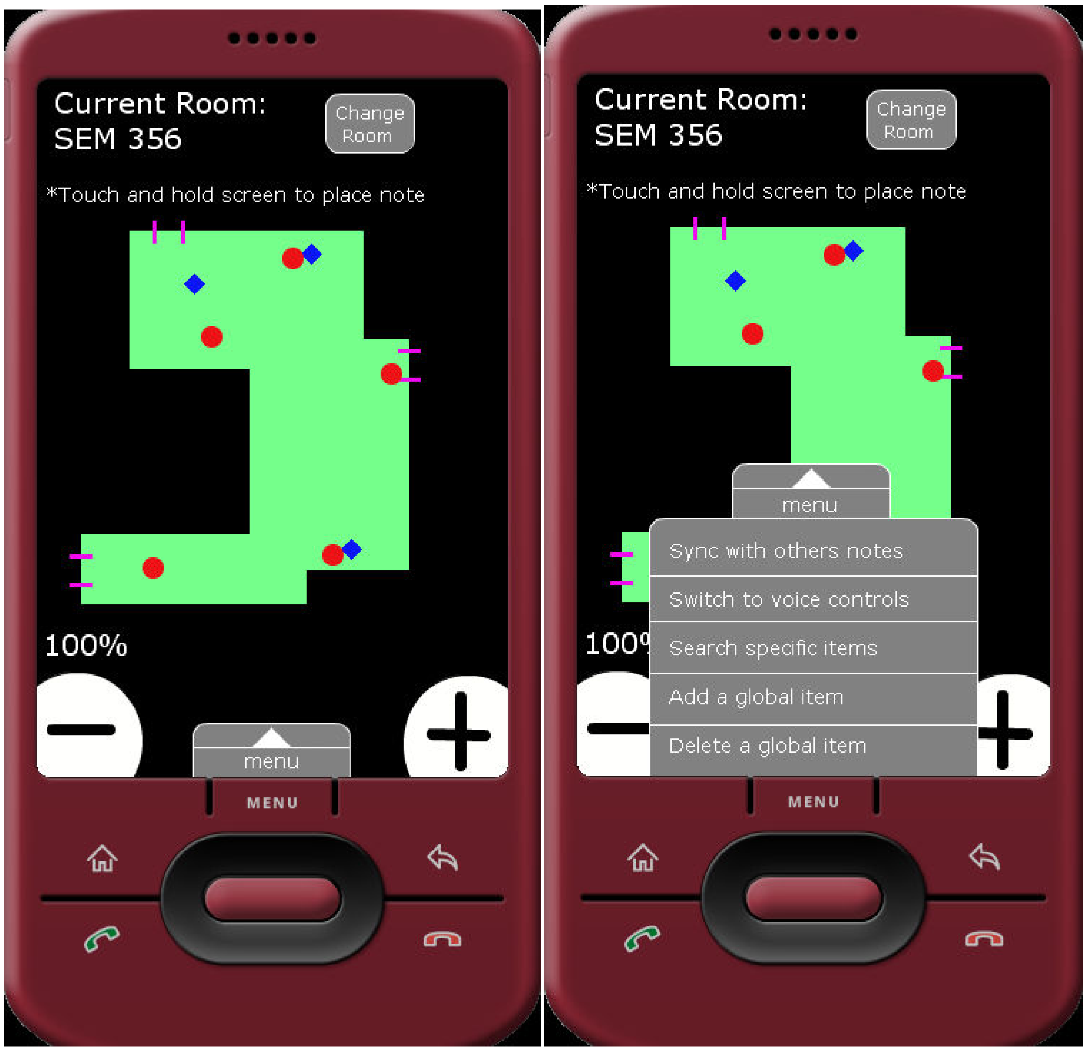


Figure 4: Screenshot of the main map screen after a room has been selected (Left). Screenshot of the menu after the user presses the menu button (Right).

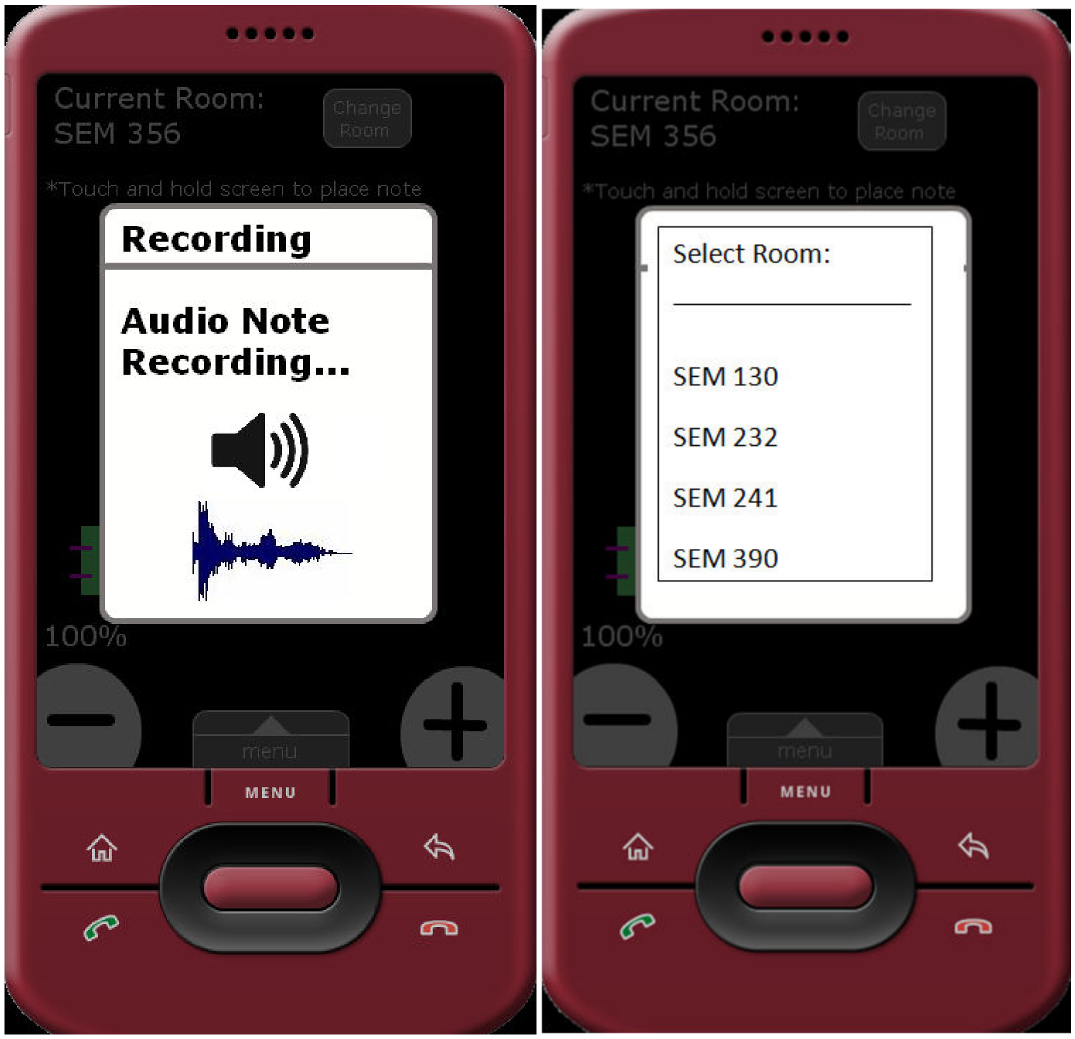
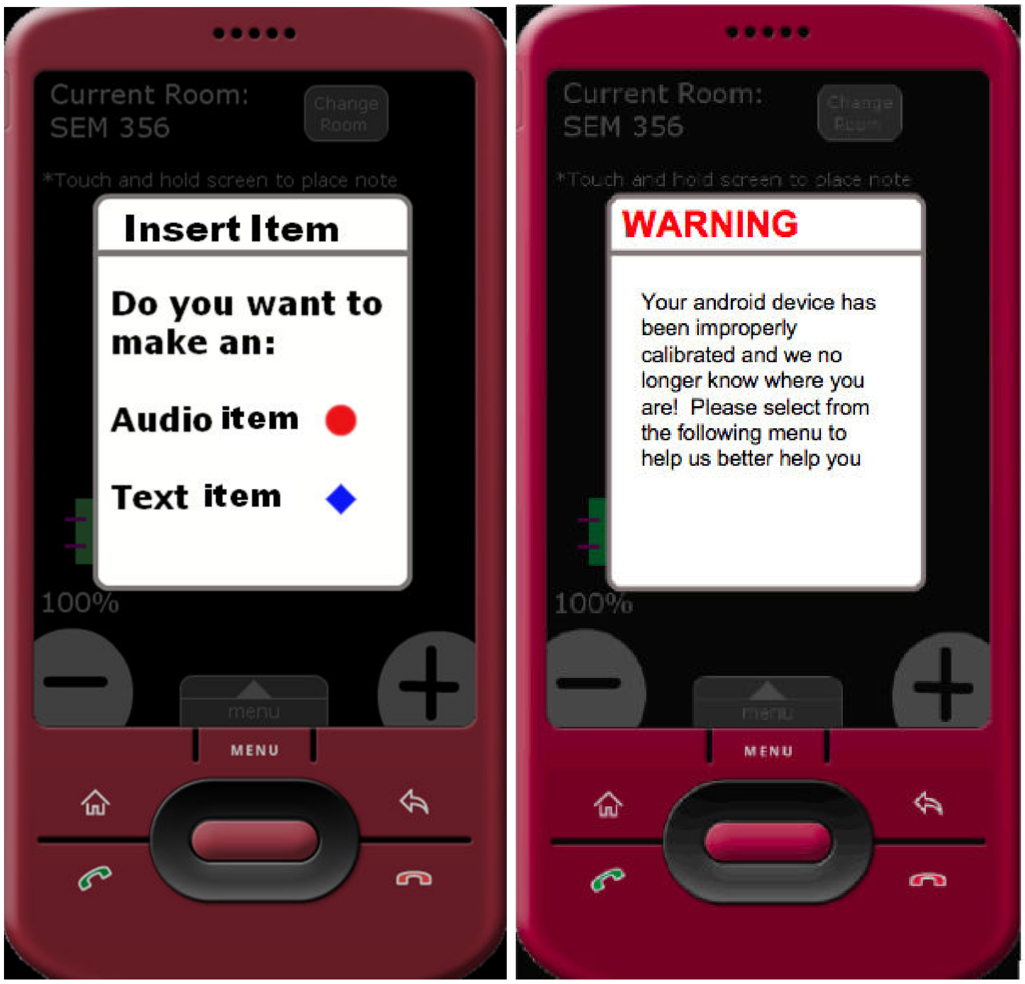


Figure 5: Screenshot of an audio note being recorded (Left).
Screenshot of room selection (Right).



Your android device has been improperly calibrated and we no longer know where you are! Please select from the following menu to help us better help you

Figure 6: Screenshot of "Insert Item" screen (Left).
Screenshot of error screen (Right).

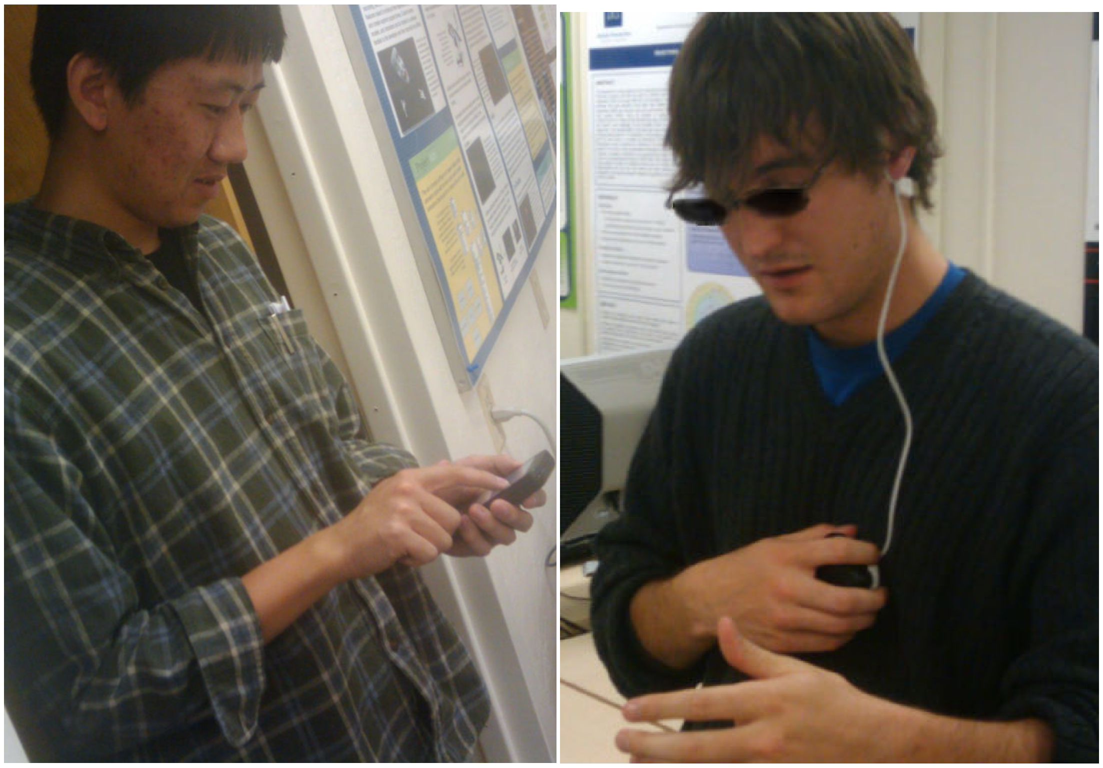


Figure 7: Sighted user (Left).

Blind user (right).

7 Annotated References

1. This is an article on Google utilizing crowdsourcing to collect data for the street view of Google Maps. It talks about the issue of correlating data to arrive at the best information. This is one of our big concerns and this article will help us get an idea of what we have to do to keep the data clean and without opposing information. It also gives reference to India's project of crowdsourcing. They have already set up a successful crowdsourcing application for navigating the streets of India which is similar to what we are trying to accomplish. Though the Google application is still being "piloted", it indicates that many map applications in the future, around the world, will be utilizing the crowdsourcing technique to acquire information.

   Caverly, Doug. "Google Maps Makes Use Of Crowdsourcing." 2007. WebProNews. 2009 <http://www.webpronews.com/topnews/2007/08/03/google-maps-makes-use-of-crowdsourcing>.

2. This is a very detailed article of IBM's software which navigates inside a building; workspaces specifically. It can search, track, and plot individual cubicles, rooms, employees, or assets. This is extremely similar to what we are trying to do which is great since IBM is funding our project. It can also graph the location of individuals or groups of employees based on job function or track unused office space visually. This application uses a 2-D image with Gimp's ImageMap plug-in to display the map of the building. It also gives sample code for extracting coordinates and identifiers. This will give us a good idea of an example of how they have organized their data entry information. However, this is not mobile and does not utilize crowdsourcing.

   Harrington, Nathan. "Map places, people, and relationships inside a building with open source software." 2007. IBM. 2009 <http://www.ibm.com/developerworks/library/os-gimpmap/>.

3. This web discussion is centered on what will be a good application for the blind patron to have which can help him/her navigate a building they have never been in. There are responses from blind patrons asking for what they feel is needed and how much this application would help them if available. It helps us keep in mind what we need to include helping our target audience. Since our main focus is on blind patrons, we can use this site to better understand what is needed from the application to make the

user's life as easy as possible. After all, an application is useless if the user is unable to use it. This site will help us get actual feedback from blind patrons through the forums.

"Navigating a Building for the First Time?" 2008. American Foundation for the Blind. 2009 <http://www.afb.org/message_board_replies2.asp?TopicID=4064&FolderID=3>.

4. This website showcases a mobile application which allows people to navigate and explore places-- "Google Maps inside a building." It also shows relevant information depending on location such as sale prices and who is attending class. The application has a much higher interaction level than what we are going for which is good for future expansion for our application. It would be amazing if we can make a device which can help blind patrons navigate and interact with people all through one simple application. This sort of indoor navigation with detailed information is what we are trying to create and is a good reference for ideas and organization. The application is still in beta and runs on the iPhone.

"The Mind of a Building in the Palm of your Hand." 2009. Micello. 2009 <http://www.micello.com/>.

5. This website was introduced to us by Dr. Folmer as a reference for how well crowdsourcing works. Waze is a social mobile application which enables drivers to build and use real-time road intelligence. The information for the map is provided by crowdsourcing; constantly updated by people to provide the most accurate up-to-date info. For example, if there is an accident, the user can upload that information and thus share with the world not to take that road since there is an accident. "The more you drive, the better it gets." The idea of more users equal better data is exactly what we are trying to go for except we want this kind of information available for inside buildings, not just on street maps.

"Real-time maps and traffic information based on the wisdom of the crowd." 2009. Waze. 2009 <http://www.waze.com/>.

6. The book Semistructured Database Design discusses how to properly format and store data for web applications. The authors discuss the importance of normalization and efficiency using algorithms and XML. For our application, we will need to set up a database to record the objects in the room with the location and name of the room item. We will use XML tags to categorize the data and efficiently scan the data to find objects. We will be storing the data on a remote web server which will allow centralization to make sure the data is not fragmented among different users.

   Dobbie, Gillian, Mong Li Lee, and Tok Wang Ling. Semistructured Database Design. New York: Springer, 2005.

7. Surowiecki discusses the psychology of crowd mentality and how as a group they are smarter than the smartest single member of the group. He also discusses how as a group, they can also be smarter than proclaimed experts in a subject due to the diversity and opinions. The subject matter relates to how we should use crowdsourcing for the project because it is virtually impossible to have a team scour every room in every building to catalog what is in the rooms. By allowing the user to submit data, it saves on research cost and it will be more up to date because the people who use the room regularly will catalog room items quicker than sending a team every few months to catalog the room.

   Surowiecki, James. The Wisdom of Crowds. New York: Doubleday, 2004.

8 Glossary of Terms

Word	Definition
Accelerometer	A device that measures acceleration in a specific direction.
Android	A mobile operating system developed by Google based on Linux.
Audibly/Audible	The ability to make sound to be heard.
Birds Eye View	Literal top-down perspectives view as if the viewer was a bird looking down on the land.
Blind	A person who cannot see.
Crowdsourcing	The act of outsourcing work to a crowd with interest in contributing to the work.
Database	A centralized location of data accessibly through the Internet.
Demographic	Selection of people who match specific criteria.
Eclipse	The integrated development kit used to program and compile the source code.
Google	Famous search engine. Also the company who developed Android.
GPS	Global Positioning Satellite.
Gyrometer	A device used to measure the rotational speed of an object that is rotating.
Java	The programming language that is used for development on Android.
KML Model	The data format of the maps. It is based on XML.
Map	The drawing of the rooms and buildings.
Overlay	An image or part of an image composited over another image.
Room Item	Any object that is in the room such as furniture.
Sighted	A person that can see (not blind).
Speech Recognition	Function of the program that converts spoken text to a string for comparison to determine the next step.
Textual	Based on text (words).
Text-To-Speech	The conversion of text to a sound.
Visually-Impaired	Person who is blind or has a vision handicap.

9 List of Differences from DD 2009

The main difference with the project in its current form is its move from being a partially Navitar-based system to a system that is completely independent of Navitar. This decision was made because the two software systems are now being developed completely independently, and it doesn't make sense to try and integrate them along every step of the process.

In light of this different development path, we revaluated our priorities in the program and are now focusing more on a searching and finding algorithm for objects and their generalizability rather than navigating around them. What this means for our project is less emphasis on the actual interface for the user, and more on the core functionality of what really makes this a worthwhile project. Its value comes in being able to properly elicit and explain an accurate description of your surroundings and to respond to queries about the immediate surroundings.

Previously, we were also concerned with the data collection because we were worried about the filtering of the data that would need to come into the system. This is no longer the case as we are not concerning ourselves with the filtering at this time, since we are assuming that users will police themselves. Instead, the time we were going to spend developing this aspect is now being spent in making the user uploading process more streamlined and user friendly, for sighted and non-sighted users alike.

10 Contributions of Team Members

Kevin Grant:

Primary focus is project management and primary programmer for the program's main architecture.

Angela Proffitt:

Developing the voice command parser and working on poster.

Alex Tam:

Primary focus is creating the graphical user interface and room map parser. Updated website and worked on poster.

10.1 Hours Worked

Poster: 2 hours

Presentation: 6 hours

Programming: 42 hours