Mobile Companion to the Glass Infrastructure

by

Brett S. Lazarus

S.B., CS. M.I.T., 2012

Submitted to the Department of Electrical Engineering and Computer Science in Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Electrical Engineering and Computer Science at the Massachusetts Institute of Technology

May 2013

Copyright 2013 Brett S. Lazarus. All rights reserved.
The author hereby grants to M.I.T. permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole and in part in any medium now known or hereafter created.

Author:

/ Department of Electrical Engineering and Computer Science
May 24, 2013

Certified by:

Dr. Catherine Havasi, Digital Intuition, MIT Media Lab, Thesis Supervisor
May 24, 2013

Accepted by:

Prof. Dennis M. Freeman, Chairman, Masters of Engineering Thesis Committee
Mobile Companion to the Glass Infrastructure

by

Brett Lazarus

Submitted to the Department of Electrical Engineering and Computer Science

May 24, 2013

In Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Electrical Engineering and Computer Science

Abstract

MessageMe is a real-time, location-based messaging system for the MIT Media Lab. Users compose messages that are delivered as recipients are detected at specified locations in the Lab. MessageMe builds on the Media Lab Glass Infrastructure, utilizing the RFID readers at each kiosk to determine users’ locations.

GI Mobile is a mobile application that acts as a companion to the Glass Infrastructure. It incorporates the MessageMe messaging system to deliver a suite of location-aware features that complement the Glass Infrastructure. These include locating others in the Lab, browsing projects by location, and sending location-triggered messages. In addition, GI Mobile generates project recommendations based on other projects a user has “liked.” It will alert users as they pass by recommended projects, helping visitors explore the plethora of projects at the Media Lab.
Acknowledgements

Thanks to Shell and the MIT Energy Initiative for partnering with me in this project. Additionally, Jon Ferguson (Senior programmer, MIT Media Lab staff) provided invaluable guidance in working with the Glass Infrastructure and in brainstorming ideas for this project. Also, thanks to Stacie Slotnick (Assistant Director for Digital and Internal Communications, MIT Media Lab) for her help in coming up with a solid use case for GI Mobile, and for her help in distributing installation instructions during the 2013 Spring Member Week. Finally, thanks to Dr. Catherine Havasi (Digital Intuition, MIT Media Lab) for her help and guidance throughout the year as my thesis advisor, without which this project would not have been possible.
Table of Contents

Acknowledgements .............................................................................................................. 3
Introduction ............................................................................................................................ 5
Background Information ......................................................................................................... 8
  Digital Intuition ......................................................................................................................... 8
  ConceptNet ............................................................................................................................... 8
  AnalogySpace .......................................................................................................................... 9
  Luminoso ................................................................................................................................ 9
Glass Infrastructure ................................................................................................................ 10
  Overview .................................................................................................................................. 10
  Charms .................................................................................................................................... 13
CharmMe .................................................................................................................................. 13
  ConnectMe .................................................................................................................................. 13
  CharmMe .................................................................................................................................. 14
Indoor Positioning Systems ..................................................................................................... 18
  Overview .................................................................................................................................. 18
  Positioning algorithms ............................................................................................................... 18
  RFID .......................................................................................................................................... 19
  Other technologies .................................................................................................................... 19
Web technologies .................................................................................................................... 20
  JQuery ....................................................................................................................................... 21
  Bootstrap ..................................................................................................................................... 21
  Backbone.js ............................................................................................................................... 21
  Node.js ....................................................................................................................................... 21
  MongoDB ..................................................................................................................................... 22
  Socket.IO ..................................................................................................................................... 22
  PhoneGap .................................................................................................................................. 22
Related Work .......................................................................................................................... 24
  Google Field Trip ...................................................................................................................... 24
  Find My Friends ....................................................................................................................... 24
  Intellitix ...................................................................................................................................... 25
Previous Work ......................................................................................................................... 26
MessageMe and GI Mobile ....................................................................................................... 32
  Overview .................................................................................................................................... 32
  MessageMe .................................................................................................................................. 33
  GI Mobile ................................................................................................................................... 36
Results .................................................................................................................................... 54
Future Work .............................................................................................................................. 56
Conclusion ................................................................................................................................ 58
References ................................................................................................................................. 59
Introduction

Messaging on smartphones is generally not thought of in the same context as location-based applications. The most common forms of messaging such as email, text messaging, and instant messaging, have no notion of location. Yet, location aware apps have become widespread in their own right, ranging from navigation with Google maps to more socially focused apps like foursquare. Most smartphones use GPS as their primary means of location tracking. However, these systems have their shortcomings when greater accuracy is required in an application. This becomes an even greater problem indoors, where poor satellite coverage decreases accuracy even further. As a result, Indoor Positioning Systems (IPS) have become an important area of research in the past decade. A variety of wireless technologies offer potential solutions to either replace or enhance GPS, including Wi-Fi, Bluetooth, RFID, infrared systems, and more [1, 2].

With the ubiquity of GPS and the rise of IPS, it is natural that applications will surface that combine messaging and the ability for devices to locate and configure themselves to their surroundings. Indeed, some of these applications are already cropping up, including an application from Google called Field Trip that sends the user notifications and recommendations about their surroundings. This class of application has relevance in many contexts. For instance, in a retail setting a company might send coupons to users for items they are physically close to. Or, by incorporating purchase history and preferences, the company could direct the user to specific areas of interest within the store. The company might also message the user when they have walked near an area containing a product they might be interested in purchasing. This streamlines the in-store customer experience and could drive sales when the customer might otherwise have overlooked their vicinity to an interesting product. In fact, a company called
Placecast is currently working on a product called ShopAlerts, which is an opt-in service for location-based text message advertising that some retailers have already begun adopting [3].

Besides commercial applications, similar technology could be useful in large events such as conferences or music festivals. For example, a demonstration booth at a conference could send a message to anyone that passes near the booth to alert people of the demo. Or, perhaps two attendees that are separated could message each other to set up a meeting spot. The application would automatically send a message to one attendee when the other has arrived at the designated meeting location. Alternatively, in a museum exhibit a location-based application could replace current expensive audio guide systems and allow users to leave comments at specific locations in the exhibit [4].

In the MIT Media Lab, RFID readers around the building form an indoor location tracking system for the people in the Lab. The readers are a part of a larger system called the Glass Infrastructure, which deploys large touch screens around the Lab that act as virtual kiosks. Each screen includes an RFID scanner that can detect users wearing RFID tags. The Glass Infrastructure, while providing a wealth of useful features, is limited by the fact that its features are only available on the public kiosks in the Lab, prompting frequent requests from visitors for something more mobile. The MessageMe and GI Mobile projects seek to fill this niche by extending the system onto mobile devices. Moreover, they explore the possible uses of adding a location-aware messaging system on top of the existing infrastructure. MessageMe is a platform targeted for mobile devices that implements a real-time messaging system that allows arbitrary messages to be delivered to recipients as they pass by (detected by RFID) a specific location. GI
Mobile is a mobile companion to the Glass Infrastructure that incorporates MessageMe along with a suite of other features. These include a mini project browser, project recommendations, managing charms, and locating other people in the Lab. Implemented both as a mobile ready web application and a native Android application, and deployed for the Spring 2013 Member Event, GI Mobile provides a useful addition to the Media Lab’s Glass Infrastructure.
Background Information

Digital Intuition

*ConceptNet*

Ongoing work by Dr. Havasi and the Digital Intuition group is an integral part of the Glass Infrastructure. The base technology for the group is a project called ConceptNet. ConceptNet is a semantic network containing information computers should know about the world. Its goal is to, “provide intuition to AI systems and applications by giving them access to a broad collection of basic knowledge, along with the computational tools to work with it.” [5]. These are typically phrases that may seem simple or obvious to humans, but which a computer would not intrinsically know. Much of this data was captured through the Open Mind Common Sense project where contributors entered phrases they thought to be common sense information about the world. This might be a phrase such as, “an apple is a type of fruit.” ConceptNet would then be tasked with taking this phrase and analyzing the text using natural language processing techniques to uncover that “apple” and “fruit” are concepts, and that there is an “IsA” relationship between them. Over 17 million pieces of knowledge of this kind have been captured just in the English language (other languages are now supported as well). This knowledge is stored in a structured form consisting of concepts and relationships between those concepts, forming a directed graph. In more recent versions of ConceptNet, other sources of information besides the Open Mind Common Sense initiative have been added. One such external source is WordNet, a popular natural-language processing tool that contains a large lexical database of the English language [6]. By adding in these external sources, the amount of data in ConceptNet was greatly increased.
AnalogySpace

However, no source is going to be able to create every relationship between concepts. Therefore, a process called AnalogySpace was developed by the group to allow new information to be inferred from the knowledge already in ConceptNet. Specifically, it is a “matrix-based representation that uses dimensionality reduction to infer new knowledge (which can then be added to ConceptNet),” [7]. Representing ConceptNet as a sparse matrix and using singular value decomposition reveals the most important components of the matrix. The top $k$ components are kept and the rest discarded in order to form a lower dimensionality approximation to the original matrix, making matrix operations less computationally intensive. From here, it is simpler to find concepts that are similar to each other by comparing their vector representations. Finally, by analyzing similar concepts, new relationships can be inferred and added back into ConceptNet, resulting in a more complete data set.

Luminoso

The next important aspect of this work is how ConceptNet is used effectively to enhance other data mining and machine learning tasks. Luminoso is the project that formalized a method for using ConceptNet to enhance understanding in text analysis. Starting with relatively simple natural language processing techniques, Luminoso then uses a blending technique to add common sense knowledge from ConceptNet. In this way, Luminoso is able to extract a much richer understanding from a small amount of text [7]. In fact, the Glass Infrastructure relies heavily on this technique to understand relationships between projects in the Media Lab.
Glass Infrastructure

Overview
The Glass Infrastructure is a project developed jointly by multiple groups in the Media Lab including Information Ecology, Digital Intuition, Viral Interfaces, and NecSys. It consists of more than thirty large touch screens placed around the Media Lab that act as visitor information kiosks. These kiosks are context dependent, so that the content shown depends on their physical location in the Lab. They allow visitors to explore the research projects going on in the various sections of the Lab in both physically and thematically organized ways. In addition, visitors can catalogue, share, and organize the projects they are interested in.

To be able to organize projects by theme, a specialized version of the Luminoso software is used to take advantage of ConceptNet’s semantic understanding of common sense knowledge. The input to this process is information in the Media Lab’s Project List Database (PLDB). This can be seen in the diagram of the GI’s components shown in Figure 1. Because this database contains only short descriptions of the projects, Luminoso and ConceptNet are key to adding meaning to this text so that the descriptions can be used to effectively find relationships between the research projects. The main user interface, know as the Project Browser, can be seen in Figure 2. The sets of projects arranged in circles demonstrate the ability of Luminoso to group projects into ad-hoc categories on the fly. While most screens show the Project Browser by default, others display maps of the Lab for the benefit of visitors. In addition, the screens are equipped with an application-switching interface to allow for arbitrary applications to be run on the screens. Over the years these have included Twitter streams, Media Lab event calendars, and more.
Figure 1: A diagram showing the technical layout of the Glass infrastructure. There are three main components: the kiosks, the server, and legacy databases. Tagnet is an important component of the server, acting as a layer of abstraction that many other components communicate through.
Figure 2: An example of the Project Browser, displaying information on the Viral Communications research group [8].
**Charms**

In addition, the Glass Infrastructure provides a mechanism for bookmarking favorite projects, called ‘charms’. This begins with the ability to log in to a particular screen, which in turn is made possible by the combination of RFID readers attached to each screen along with the RFID tags included in the badges issued to all Media Lab members and sponsors. A central server called Tagnet, seen in Figure 1, is responsible for repeatedly polling each RFID reader to see which tags are currently being detected. By matching RFID tag unique identifiers with Media Lab account usernames, Tagnet is able to provide each screen with a list of everyone currently standing at that location. The screen modifies its interface to allow the users detected in front of it to log in. From there, as a user browses projects, they can add, remove, and view their charmed projects. Besides being a useful bookmarking feature, charms also provide valuable feedback to the GI system about a user’s likes and interests.

**CharmMe**

**ConnectMe**

A set of projects developed in the Digital Intuition group that take advantage of the GI charm system are ConnectMe (2009) and most recently CharmMe (2012). ConnectMe, a project of Dr. Havasi’s, is a social discovery application that helps people meet other people at the Media Lab. The application uses algorithms based on Luminoso and other machine learning techniques to suggest connecting with other like-minded individuals in attendance using information from the user’s activity around the Lab. Such activity includes checking in to conference talks or charming projects. In addition, possible opening topics of conversation are suggested based on expressed similar interests. This application helps connect people by encouraging purposeful
interactions, as well as decreasing the opportunity for a missed but potentially meaningful connection.

**CharmMe**

In the extension project CharmMe [9], developed by Victor Wang of the Digital Intuition group, the application makes available the location of all recommended people using the Glass Infrastructure’s RFID readers. CharmMe uses this location information to take into account the distance between people when recommending meetings. The application was also made available on mobile devices, and later on the Glass Infrastructure touch screens.

I worked on CharmMe in the spring of 2012, where I helped to revamp both the frontend and backend of the application in order to get it ready for sponsor week. My primary task was to improve the user interface. This involved some design work followed by modifications of the HTML and CSS to produce a more professional and aesthetically pleasing web page. Further, it was important to have the same web page work well on various screen sizes in order to support mobile devices as well as Glass Infrastructure kiosk screens (which run applications as web pages in a browser). I designed the CSS so that the size and layout of elements on the page adapt seamlessly to varying screen sizes. Screenshots of the web application at various screen widths are shown in Figure 3 and Figure 4.
Figure 3: This shows CharmMe when logged in as Dr. Havasi when viewing on a screen the size of a typical smartphone.
Figure 4: This shows CharmMe when logged in as Dr. Havasi when viewing on a screen the size of a typical laptop, or on a larger screen such as those used for the Glass Infrastructure.

The interface is arranged vertically so that it will only require vertical scrolling, which is especially important on mobile devices. It displays the logged in user’s information and last seen location. Below that it shows the top three recommended sponsors and the top three Media Lab
members to connect with, along with their information and last seen location. Also present is the option to sign in with LinkedIn, which is another feature I helped to implement. After signing in with LinkedIn, the user’s LinkedIn profile appears on the page. The user can click the “We’ve Met” button to alert the system that they have successfully met the recommended person. Once a person has been met, they will no longer show up as a recommendation. In addition, a new page will open up with the person’s LinkedIn profile so that the logged in user can connect with this person on LinkedIn to form a more lasting bond after sponsor week.

On the backend, the lead student for the project, Victor Wang, and I made substantial modifications. The major task was updating CharmMe to work with a new backend model for the Glass Infrastructure. One issue that arose from this was the long wait time for Luminoso to generate recommendations, which could take upwards of thirty seconds and cause the page to feel unresponsive. While we couldn’t avoid the problem entirely, I was tasked with building a database to serve as a cache for the recommendation results. The cache in the previous version of CharmMe was done in memory so it was lost any time the server crashed or restarted. In addition, there was no expire time on this cache, so there was frequently stale data showing on the application. Therefore, I implemented a SQLite database to act as the cache, allowing the cache to be persistent over crashes and restarts. A limit of one hour was set for data in the cache so that users logging into the system would occasionally get updated information. This experience working with CharmMe ultimately carried over into the implementation of GI Mobile.
Indoor Positioning Systems

Overview
With its RFID polling system, the Glass Infrastructure can be thought of as a kind of Indoor Positioning System (IPS). IPS aims to overcome the shortcomings of the Global Positioning System (GPS) while indoors [4]. While GPS is a global standard, IPS refers to a class of technologies where active research is still being done. An IPS system may return a number of different types of location information depending on the needs of the application. These include physical location (latitude and longitude coordinates), symbolic location (a natural-language description of a location), absolute location (coordinates in a reference grid), and relative location (proximity to known reference points) [2]. The Glass Infrastructure IPS can be thought of as a course-grained version of symbolic and relative positioning, as users are detected at kiosks, which are known reference points. Each kiosk has an identifier that is generally related to its symbolic location in the building. For instance, the screen identifier “e14-274-1” indicates that the screen resides in room e14-274 of the Media Lab (the Swatch Lab), which also implies that it is on the second floor of building e14 (as is customary of MIT’s room numbering scheme). Finally, it is course-grained because rather than giving actual distances, the system can only tell whether the user is in front of the screen or not.

Positioning algorithms
In general, most of these systems use some combination of triangulation and scene analysis to generate the location of the target device. Triangulation uses the geometry of the target device relative to other devices that have known positions, which are satellites in the case of GPS, in order to pinpoint the location of the target. The target device communicates with the satellites via wireless transmissions and can then estimate the distance to each satellite using various
properties of the transmission. Scene analysis can be used in addition to triangulation. It attempts to collect features of a "scene," such as the received signal strength at various locations, and then match online measurements against this *a priori* knowledge. [2]

**RFID**

One system that uses RFID for indoor location tracking is described in [1]. Unlike the Glass Infrastructure, which tracks people, this system is primarily meant for tracking robots, resulting in different designs. This system works by embedding an array of close-range RFID tags in a surface such as the floor of a factory. Each tag has a unique identifier that can be mapped to its physical coordinates on a central server. The robots being must have RFID readers that communicate with the embedded tags. As a robot moves over an embedded tag, its reader detects the tag's id and can then map it to a physical location in the grid. This is an interesting proof of concept showing that RFID can be used in a scalable way to track fine-grained, absolute location. However, because the device being tracked must keep an RFID reader within a few inches of the ground, it does not work well for tracking humans.

*Other technologies*

Besides RFID, various other wireless technologies have been employed in different indoor location systems, including Wi-Fi, Bluetooth, ultrasound, pseudo GPS satellites, cellular-based, and more. The cellular-based systems bootstrap the existing mobile network infrastructure, using the wireless communications between the mobile device and a cellular base station to pinpoint the device. Scene analysis is used as the location estimation algorithm rather than triangulation so as to avoid having to be in range and communicate with multiple base stations. Research using this method has managed accuracy as low as 2.5 meters. [2]
IPS using Wi-Fi again takes advantage of existing infrastructure, making its deployment less costly. Again scene analysis is used on signals from Wi-Fi base stations to determine location. Existing projects in this area each use different machine learning algorithms to estimate location using offline measurements, and employ varying signal processing tricks to improve measurements accuracy. While accuracy results differ slightly, they are in general comparable to those achieved using cellular networks. [2]

Bluetooth devices, like RFID, each contain a unique identifier. Solutions using Bluetooth are generally comprised of access points, tags, and positioning servers. This architecture is similar to the RFID system described above, but differs in the communication standard used and the methods for estimating location. Moreover, because Bluetooth is already embedded in most mobile phones, these systems can be suitable for tracking humans. To determine location, some combination of triangulation and scene analysis is used from communication with one or more Bluetooth “base stations.” The accuracy of Bluetooth systems is comparable to those using Wi-Fi and cellular networks. These technologies comprise just some of the methods being used for IPS. Active research is also being done to combine various sensing methods and algorithms for even greater accuracy. [2]

**Web technologies**
This section presents a brief overview of the web technologies used in building MessageMe and GI Mobile.
**JQuery**

JQuery is a JavaScript library that is commonly used in web applications to interact with the Document Object Model (DOM) tree of HTML elements in the browser. It aids in tasks like HTML element traversal and manipulation, event handling, animation, and asynchronous HTTP requests (AJAX). Further, it works across many browsers.

**Bootstrap**

Bootstrap is a frontend JavaScript library developed by Twitter. It includes various tools to help develop a better user interface including a responsive grid for layout and base CSS for better looking typography, form controls, and many other web interface elements. It also includes many UI components and JavaScript plugins such as navigation, alerts, or transitions that make it easy to develop interfaces that look good on multiple devices.

**Backbone.js**

The Backbone.js JavaScript library helps organize frontend web application code into a Model and View pattern. Models and collections of models represent the data in the application, while views are responsible for monitoring models for changes, responding to user events, and rendering the user interface. In addition, Backbone.js provides a Router object that simplifies the development of single page applications (no page refreshing when navigating within the app) by providing methods for routing client-side pages and updating the browser’s URL accordingly.

**Node.js**

Node.js is a server-side platform built on a JavaScript runtime that allows web application servers to be written in JavaScript. According to the Node.js website at http://nodejs.org, it “uses
an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.” While many other web server technologies spin up threads for database queries or network I/O, Node.js is single threaded. It instead uses an event loop and callbacks to deal with these asynchronous I/O events.

MongoDB
MongoDB is a popular open-source noSQL database. Instead of having schemas like a SQL database, MongoDB stores documents with flexible schemas. Each document is simply a set of key-value pairs in the style of JavaScript Object Notation (JSON). Yet, it still offers many of the features expected in a SQL database such as indexing, sharding, and powerful querying.

Socket.IO
Socket.IO is a technology that simplifies the development of real-time web applications. It consists of both client and server side components and allows data to be sent in both directions between the client and server in real-time. Normal HTTP requests are always initiated by the client and are short-lived. However, Socket.IO is built on top of websockets, which is a protocol where the connection between client and server stays alive until it is killed by one side or by a loss of network access. Because websockets are a newer protocol, Socket.IO can also fall back to other methods of real-time data transfer when used in older browsers.

PhoneGap
PhoneGap is an open source project that aims to allow developers to create native mobile applications without having to learn the intricacies of each mobile operating system. Developers can write applications in the familiar web languages of HTML, JavaScript, and CSS, and use
PhoneGap's JavaScript API to interact with common device features such as vibration, notifications, the accelerometer, and more. PhoneGap wraps this web content in an application container that allows it to run natively on a number of devices including Android and iPhone. The PhoneGap project is useful for developers using a single codebase to deploy both web and native mobile applications.
Related Work

Google Field Trip
MessageMe and GI Mobile bring location-based messaging features to the MIT Media Lab, but outside of this context there are numerous applications working to solve similar problems. First is the Google Field Trip mobile application. This travel-oriented tour guide app presents the user with location-based, virtual information cards to help the user get to know their surroundings. In this sense, it has a similar high-level goal to GI Mobile. While GI Mobile helps members and visitors explore the Media Lab, Google Field Trip helps users explore a new area anywhere in the world. However, because GI Mobile is specifically targeted for use in the Media Lab, the features of the two apps differ. In Field Trip, the user inputs the type of alerts they are interested in, including architecture, historic places and events, lifestyle, offers and deals, food and drinks, and more. Unlike this explicit input of user interests, GI Mobile relies on the implicit information gleaned from a user’s charms. In addition, in Field Trip the user can open an alert on a map, which is not currently possible in GI Mobile.

Find My Friends
Another location-based mobile application is Find My Friends, a popular app available for the iPhone. This app allows the user to track their friends’ locations (who also have the app and have accepted a request to share their location). This is comparable to GI Mobile’s ‘Locate a Friend’ feature. Find My Friends is most useful in crowded outdoor events such as music festivals or parades, or when on vacation. But because it relies on GPS, it would not work as well in a setting like the Media Lab, which is where GI Mobile excels using its RFID system and associated knowledge about each area of the Lab. Find My Friends further adds features such as location-based alerts, for instance when a family member arrives home safely. [10]
Intellitix
Finally, the company Intellitix develops RFID systems for large events and has some similarities to the Media Lab's Glass Infrastructure. Its primary use is as a ticket replacement solution using wristbands with embedded RFID tags. As added benefits, they also offer cashless payment systems, as well as checking in at kiosks called Live Click Stations to let users interact with social media at various locations in the event. Intellitix has installed systems at various music festivals including Lollapalooza in Chicago and Coachella in California [11]. Because their system uses RFID to tag users at specific locations, it might be interesting to implement a location-based messaging system similar to MessageMe for use in these music festivals. However, at this point it does not appear that Intellitix has built any such system.
Previous Work

The MessageMe and GI Mobile projects are the culmination of previous work and brainstorming done throughout the year. These projects are in collaboration with the Shell Corporation through the MIT Energy Initiative, and began by exploring how technologies like CharmMe, the Glass Infrastructure, and Luminoso could be used to improve organizational understanding. While CharmMe was tailored to the Media Lab, the goal was to build a tool tailored to a large organization spread out geographically over many office locations.

There were multiple possible use cases for this kind of technology, including team building for research projects, internal recruiting and succession planning, and domain expertise. For instance, when a new research team is being formed, can we predict a group of employees that will work well together to produce the best results? Using Luminoso and ConceptNet can help glean understanding from a limited set of data about each employee in the company, which can then be fed into an algorithm to find the best matches. For companies with so many employees, it is often impossible for staffing managers or project leads to know all of the employees that might be a good fit for the team. Likewise, internal recruiting and succession planning present the problem of filling positions in a company where it is impossible to know every good candidate for the job. This would require a similar type of algorithm to generate matches from employee data. Finally, for domain expertise, when an employee has a question, can we build a tool that will match them with others in the organization that have experience or expertise in that field?

Each use case required some kind of employee data from Shell. Obtaining this data presented legal problems for the team responsible at Shell. In the mean time, the research strategy was to
work in the context of the Media Lab to develop the necessary algorithms. As preliminary work on the problem of building teams, I built an algorithm to group sponsors into tables at a breakfast during the Media Lab Fall Liaison week. The goal was to group people with similar interests to spur better conversation. This project used charm data from the sponsors as well as a survey where sponsors wrote a short line of free text about their primary interests during the visit. A python package called simplenlp, developed by the Digital Intuition group, was used to extract concepts from the free text. The extracted concepts were then combined with charm data to represent each sponsor as a vector of charms and concepts.

To form groups, a modified version of the k-means clustering algorithm was used. The k-means algorithm starts with k random centroids in the vector space representing the k groups being formed. Data points are first assigned to the closest centroid, and each centroid is then adjusted to the average of the points assigned to it. This process is repeated until convergence. However, this algorithm has no control over the size of the groups created, which was necessary to generate equally sized sponsor breakfast tables. Therefore, I tweaked the assignment phase of the algorithm so that instead of each data point being assigned to the closest centroid, the centroids pick data points in a round robin fashion, each time picking the closest data point that has not yet been claimed to a group. This allowed the algorithm to generate roughly equally sized groups of sponsors with similar interests.

The participating sponsors generally reported that they had engaging and interesting conversation at their breakfast table. Yet, despite this positive feedback, because the project was not set up as a well-formed scientific experiment with a control group, it was difficult to
determine how much of the success to attribute to the grouping algorithm, or whether any number of different combinations of groups could have produced similarly interesting conversations.

Midway through the spring, however, Shell was still unable to secure data for this project. As a result the decision was made to proceed without Shell data and instead adapt the team building application to the context of the Media Lab, possibly as part of a cross-group initiative within the Lab. In addition, work began to improve the grouping algorithm. First, instead of an input matrix composed simply of the feature vectors for each sponsor, I applied a dimensionality reduction technique using singular value decomposition (SVD). The SVD takes the original matrix and decomposes it into a product of three matrices, one of which is a diagonal matrix of ordered singular values. By cutting out the lower singular values, it is left with only the most important components of the original matrix. This in theory should not only make the matrix operations in the grouping algorithm less computationally intensive, but also help to produce better grouping results. Another improvement considered was the ability to adjust the tradeoff between category consistency and divisiveness, exploring whether grouping the most similar people, the most diverse people, or something in between results in the best teams.

Apart from the algorithm itself, an important consideration was the experimental setup for the project. For instance, how does one measure the success of a team, and how does one determine if this was a result of the grouping algorithm used? In this project it would be necessary to show that my algorithm produces better results than other methods such as other grouping algorithms, random assignment, or self-selection. Ultimately, without a solid use case in the Media Lab and
with uncertainty surrounding the experimental setup, the decision was made not to move forward with this particular line of research.

As an alternative, I considered projects related to the Glass Infrastructure. Numerous discussions with Dr. Havasi and Jon Ferguson, a senior programmer in the NecSys computing department of the Media Lab and current administrator of the Glass Infrastructure, were integral in this brainstorming process. One possibility discussed was having the screens display relevant information as people walk by, such as upcoming talks or other events. This could even use charms to display only events that match a user's interests. Another option was looking into other ways for the screens to identify people besides the RFID readers. The readers work well during sponsor weeks and other large events when everyone wears their badges with the embedded RFID tags, but on a day-to-day basis, most Media Lab members do not wear their badges. Thus, for much of the year, a large part of the functionality of the screens goes unused. The Glass Infrastructure has the potential to benefit everyone in the Lab on a daily basis, but would need a solution besides the RFID badges for tracking people.

Smartphones might enable an elegant solution, as people generally already carry their smartphones with them wherever they go, and the devices already expose API's for pinpointing their location using GPS. However, to be able to reliably determine that a user is in front of a screen, the location system would need to be accurate within a few feet. Currently GPS has not reached this point, especially indoors. As a result, I began looking into other ways of locating the user in front of a screen, including using Bluetooth and even leveraging the cell signal. However, this is essentially equivalent to building an Indoor Positioning System (IPS). With the
infrastructure, signal processing, and location estimation algorithms needed, building an IPS was outside the scope of this work.

However, the idea of a mobile application working as a companion to the Glass Infrastructure did present some interesting possibilities, with numerous potential use cases. One conceivable use was wayfinding in the Lab using the Glass Infrastructure. A user would input a person or destination, and the Glass Infrastructure would help to navigate them there. This might be done with arrows on the touch screen kiosks, or a more personal navigator application on a smartphone. However, this would require location coordinates for each screen as well as elevators and stairs in the Lab. This task, essentially mapping the Media Lab, was outside the scope of this work.

In finding a use case for GI Mobile, an important consideration was that the usage scenario be personal enough that it belong on a smartphone rather than just on the GI kiosks. One possibility was being able to alert sponsors about project demonstrations as they walk by. This could be extended to deliver other kinds of advertisements for events around the lab. In order to implement this type of application, a system would be needed for pushing messages to users phones as they come in range of a particular Glass Infrastructure screen. This would ultimately result in the development of the MessageMe platform. Another possible feature for this use case was being able to share files to those in the audience of a demonstration. The audience, in this case, could be approximately determined by those recently detected by the nearest kiosk’s RFID reader. Additionally, a potential use for this project was an app to tell people near each other what to talk about and provide topics that lead to interesting conversation. This again is primarily
relevant in the context of Sponsor Week, where sponsors in the same physical space at a given time may not know each other, but probably share common interests. However, this is very similar to the problem addressed by CharmMe, and as a result was not a good fit for this work.

Finally, a solid direction for this project was set after a meeting with Stacie Slotnick, the Assistant Director for Digital and Internal Communications at the Media Lab. She was responsible for organizing a special visit to the Media Lab for the Director's Fellows, a new program that brings accomplished individuals from around the world into the Media Lab community. This visit, taking place before the actual start of Sponsor Week, included a block of time for Lab exploration. Stacie was interested in finding ways to use the Glass Infrastructure to focus Fellows' relationships with the Lab during the exploration time. One early idea discussed was using the MessageMe platform to offer a scavenger hunt to each person tailored to his or her interests. However, this idea shifted to having project recommendations tailored to the individual's interests rather than a formal scavenger hunt. Some recommendations would appear immediately upon logging into the application in order to direct their Lab exploration, while others would be more spontaneous, alerting the user as they walk by the location of a project they might be interested in. This would hopefully allow for a good mix of providing direction and allowing for self-discovery of projects in the Media Lab. Although originally intended only for the Directors Fellows, this idea was easily extended to all sponsors during Sponsor Week. Moreover, this was an application that fit well on a smartphone rather than a Glass Infrastructure kiosk. As a result, this idea became the primary use case for MessageMe and GI Mobile.
MessageMe and GI Mobile

Overview

GI Mobile is a mobile application meant to act as a companion to the Media Lab’s Glass Infrastructure. It incorporates the MessageMe location-based messaging platform along with a slew of other features that are meant to enhance Lab exploration and discovery for members and visitors alike. The messaging platform itself has many possible usage scenarios. For instance, one student could leave a message for a labmate that is delivered when they arrive in their lab area. The student might be letting their labmate know they had to move some materials to a different location. In a similar scenario, the student could leave a message that is delivered when their friend arrives in the café area, asking the second student to pick up the coffee mug that the first student had left by the sink. This kind of scenario happens on a daily basis and is a great use for the MessageMe platform. In addition, during sponsor week, there are so many project demonstrations that it is easy for any single project to get overlooked. With MessageMe, a student could compose a message that advertises their project demonstration and have it delivered to sponsors detected near the project’s location.

In GI Mobile, the project recommendations are particularly useful for discovering new, interesting projects in the Lab. During sponsor week, there is a limited amount of time for sponsors to explore the Lab, and the plethora of research going on in the Lab can be overwhelming. GI Mobile remedies this problem by delivering some project recommendations immediately to help focus Lab exploration. But, it also delivers further project recommendations as the user walks by the project, adding spontaneity to the process. Plus, if the user finds the project interesting, they can charm the project right from the app. Finally, GI Mobile allows the user to track the location of other people in the Media Lab. This is also useful in a large event.
such as sponsor week when there are a lot of people in the Lab and it is easy to get split up from colleagues.

**MessageMe**

At the core of GI Mobile is MessageMe. MessageMe is a location-based, real-time messaging platform for use in the MIT Media Lab. It takes advantage of the Glass Infrastructure’s system of RFID scanners to determine a user’s location in the Lab. The technology stack includes a backend web server that consists of node.js, MongoDB, and socket.io to deliver persistent storage of messages and real-time delivery to any web enabled device. The server exposes a set of URL endpoints that allow the front end to retrieve new messages for a user, retrieve read messages, and mark messages as read. In addition, by utilizing socket.io, the client establishes a long-lived, two-way connection with the server that allows the server to push messages to the client in real time. Although the frontend was built as a part of the GI Mobile application, the server architecture keeps open the possibility for other clients to interact with the MessageMe platform in the future.

MessageMe allows for sending arbitrary messages to a specific location or set of locations in the Media Lab. In order to do this, the server must keep track of a variety of data, which is persisted in a MongoDB database. The schema for this information can be seen in Figure 5. There are three top-level collections: User, Message, and Location. A User’s username corresponds to a valid Media Lab username, which is validated by checking with the Media Lab’s Project List Database (PLDB) via web call. Similarly, a Location’s screenid corresponds to an actual Glass Infrastructure kiosk in the Lab. In the current implementation, the client is responsible for telling the server when a user’s location has changed. This was done for the sake of simplicity; it was
easier for the client to constantly ping Tagnet (the central server in the Glass Infrastructure responsible for knowing the state of all of the RFID readers) for only the users it cares about rather than have the server be responsible for asking Tagnet about every user in the system, or alternatively attempt to keep track of the set of the users which some client cares about. In future iterations, this might be a worthwhile optimization to make, but was not necessary in the current implementation.

Figure 5: A rough diagram of the database schema for MessageMe. Names of the top-level document collections are in bold, and the properties of each collection are listed underneath. Properties in brackets indicate that they are arrays, and arrows indicate the property references the _id of another collection.

Messages can be delivered to a client in one of two ways, either a ‘push’ via websockets and socket.io, or a ‘pull’ using a simple HTTP GET request. Each client connected to socket.io tells the server the username of the logged in user, which is stored as metadata on the socket. The
HTTP request currently relies on sessions to get the logged in user. When a new message is submitted to the server via an HTTP POST method, this not only saves the new message in the database, but also triggers a global ‘new message’ event on the server. This in turn causes the server to loop through all connected websockets, query the database for any new messages, and push any resulting messages to relevant clients. The same process occurs when a client tells the server that a user’s location has been updated. So, normally messages are delivered via socket.io, but the server also supports normal HTTP requests. However, this is currently only used by the client to get any unread messages when the user first logs in, which would deliver any messages sent while the user was not logged into the application.

While a typical message in MessageMe will be directed to a specific set of recipients and will be triggered at specific locations, this is not always the case. MessageMe also supports the option of sending to all users, or being triggered immediately without waiting for the recipient to arrive at a certain location. To support this feature, the server recognizes a special “all” user as the recipient, and a “none” location as the message location. Thus, a query to find the unread messages for a user will first look for all messages where the user is a recipient or messages that have been sent to all users. It then discards any messages already read by the user by checking the readMessages list for that user. Finally, it will only deliver messages where the user’s last seen location matches one of the trigger locations for the message, unless the message’s location is “none.”

One nuance that was important to handle in both MessageMe and GI Mobile was dealing with stale location information. Since Tagnet returns the user’s last seen location and not whether they
are currently at that location, it is important to check the timestamp and make sure they were seen relatively recently. In my implementation, I chose three hours as the maximum time a user can go without being detected at a screen before marking that user’s location information as “stale.” In MessageMe and GI Mobile, having a stale location is equated to having no current location. Thus, in the query for new messages described above, even if a user’s last seen location matches the message’s trigger location but the user hasn’t been detected in over three hours, the message will not be delivered to that user. Since there are no RFID readers to determine that a user has left the Media Lab, MessageMe must rely on this “stale” number to determine that a user is no longer at their last seen location.

GI Mobile
The GI Mobile application is built on top of the MessageMe architecture. Specifically, it is a mobile companion to the Glass Infrastructure kiosks with the goal of further facilitating exploration of the Media Lab. GI Mobile was developed as a single page web application using Backbone.js, JQuery, and Twitter Bootstrap libraries on the front end. In addition to a mobile ready web application, I used PhoneGap to convert the web app into a native application for the Android operating system. Using PhoneGap’s vibration API, the native Android version is able to vibrate the user’s mobile device to alert them when a new message arrives. PhoneGap was very useful because it allowed me to develop both a web application running in the browser and a native mobile application using the same codebase.
Figure 6: The database schema for GI Mobile. Most arrows were omitted for simplicity. Instead, references are written explicitly in the properties.
On the server, GI Mobile uses the same technology stack as MessageMe. In fact, the server infrastructure for GI Mobile is actually integrated with that of MessageMe. In the future it would probably be best to fully decouple these applications. But, because MessageMe and GI Mobile share some necessary functionality and have overlapping data models, it made sense for this implementation to simply combine the two projects into a single codebase. Thus, the database schema builds on MessageMe’s schema and can be seen in Figure 6.

Besides the User, Location, and Message collections that were present in MessageMe’s schema, GI Mobile adds Recommendation, CharmActivity, Project, and Group collections. In addition, some properties have been added to the User and Location collections. These additional
collections and properties will be explained further as the corresponding parts of the user interface are discussed.

Upon opening GI Mobile in a web browser (or the Android application), the user is presented with a login screen, which can be seen in Figure 7. It allows the user to login either via Media Lab username or via webcode. The webcode is a four-character code printed on each RFID badge, which makes a convenient login option for many sponsors using the app. After a successful login (validating the username or webcode against the Media Lab database), the user is presented with the screen shown in Figure 8 and Figure 9. Figure 8 shows the view on a mobile device, while Figure 9 shows the same view from the larger screen of a laptop.

In this view there are a few things to note. First, we see that the navigation at the top is responsive to the screen size. On a desktop screen, the navigation tabs are laid out horizontally, while on a tablet or smartphone it takes the form of a collapsible vertical menu. Not visible in Figure 8 or Figure 9 is a dropdown menu under the Messages tab, which shows three more navigation options: “New Messages”, “Read Messages”, and “Post Messages.” The “Messages” tab also shows a count of the number of unread messages waiting to be viewed. The navigation is fixed to the top of the screen and does not change as the user navigates through the application.
Figure 8: Initial view after login as seen on a mobile device. The button at the top right toggles the navigation dropdown menu (hidden shown on left, expanded shown on right).
Logged in as blazarus | Your last seen location is stale... Walk in front of a screen with your badge to refresh

View messages

Project recommendation!
Based on your interests, you should check out BTNz! at BT Laboratory (e14-348-1).

Project recommendation!
Based on your interests, you should check out MessageMe at Teimex Laboratory (e14-474-1).

Project recommendation!
Based on your interests, you should check out Gi Mobile at Teimex Laboratory (e14-474-1).

Project recommendation!
Based on your interests, you should check out CharmMe at Teimex Laboratory (e14-474-1).

Please send feedback to blazarus@mit.edu
The same is true for the user information section, seen directly below the navigation panel. This section shows the logged in username, as well as the current location and timestamp of the user. This information is updated in real-time as the user moves around the Media Lab and is detected at different screens. The application keeps track of a pool of all of the users it cares about and pings Tagnet every half of a second to check for updates to those users’ locations. When a change in location occurs, the User model is updated, which in turn causes the user information view to be re-rendered with the new location and timestamp. Generally, this method of having portions of the page render themselves in response to changes in the underlying model is used throughout GI Mobile to achieve real-time updates on the user interface. As discussed above, if the user has not been detected at a screen in over three hours, their location is considered stale.

The left screenshot in Figure 8 shows that the logged in user has been spotted recently near the screen with the identifier ‘charm-6’, while the right screenshot shows the user’s location is stale.

In Figure 8 and Figure 9, below the user information view is the “New Messages” view. This is the user’s hub for all unread messages on the MessageMe platform. Messages are listed with the newest at the top. Each message has a button to mark the message as read, which will cause the message to immediately disappear from the view. It will also send an HTTP POST request to the server to mark this message as read in the database. The message, along with any other previously read messages, will then be available in the “Read Messages” tab. As for the messages themselves, there are currently two variations. First is any arbitrary message, of which we see an example in the left screenshot in Figure 8. It shows the sender, a list of recipients, and a list of trigger locations (if there are any). Each message also has a subject and a body. The other kind of message is a project recommendation, which looks very similar to a normal
message but does not show a sender or receiver. In general, the MessageMe system is very extensible to new types of messages. Currently it recognizes a special message type based on a distinct keyword in the sender field. Then the application will match the message model to the correct HTML template to render the message. For instance, it would be simple to add the ability to display people recommendations in the future by adding a new special “people-recommender” username and a corresponding template.

![New Message](image)

Figure 10: Popup notifies the user they have received a new message.

As discussed earlier, because the client and server are connected via websocket, the server has the ability to push new messages to the client. When this occurs, a popup will appear notifying the user of the new message, no matter where the user currently is in the application. A screenshot showing this popup can be seen in Figure 10. The user has the option to be taken to the “New Messages” tab immediately or to view the message at some other time. When running
the native Android version of the application, the phone will also vibrate to accompany the popup to help notify the user of the new message.

Figure 11: Viewing read messages in GI Mobile offers a paginated interface for managing a large history of messages.

Besides viewing new messages, GI Mobile also provides an interface for viewing message history. The interface looks similar, but because read messages accumulate over time, it was
necessary to offer a paginated view. A screenshot of this is shown in Figure 11. In the current implementation there are ten messages per page, which are displayed from newest to oldest.

The final component to the interface for MessageMe within GI Mobile is the “Post Messages” view. A series of screenshots are shown in Figure 12 and Figure 13 that demonstrate some of the user interface features of the form. In both the recipient and trigger location fields, the form will display a type-ahead dropdown list of possible options as the user types. For the recipients, the photos of the possible recipients are even shown in the list. The purpose is to provide a user interface that makes it easy for the user to create a new message and to minimize mistakes while adding recipients and locations. A simple user interface will encourage more messages to be sent, which in turn will lead to more use of the application. When a recipient or location is chosen from the dropdown list, it is added below the form field, as seen in Figure 12. This allows the user to easily remove an added recipient or location by clicking on the “x” next to the name. In addition, to allow the sender to include all users as recipients, or send a message without a specific location, check boxes are available to toggle these behaviors. Additionally, checking these options will disable the corresponding text inputs. With these interface options, the user has full control over all of the MessageMe features.
Figure 12: Form view for sending a new message. The left screenshot is the initial, clear state, while the right screenshot shows the form after two recipients and one trigger location have been added.
As mentioned previously, one type of built-in message is a project recommendation. This feature is very useful while exploring the Media Lab, helping sponsors and Media Lab members alike discover other projects within the Lab that might fit their interests. Also, by taking advantage of the fact that MessageMe is location aware, the project recommendations can add a fun and spontaneous aspect to Lab exploration. To generate the recommendations, GI Mobile takes advantage of the matching capabilities of Luminoso. This software builds a model of the Media Lab, which it enriches by blending in ConceptNet data. On top of this, it uses a recommendation algorithm to match Media Lab entities based on similarity. For instance, GI Mobile asks
Luminoso for projects that match a specific person. Luminoso uses charm information and project descriptions to generate intelligent recommendations. Each match for a user consists of the project identifier of the recommended project as well as a weight. The weight represents Luminoso's confidence in the match.

When a user first logs in to GI Mobile, the server makes a request to Luminoso to get project recommendations for that user. These recommendations are read in by the GI Mobile server, which then generates a MessageMe message for each recommendation. There are two types of recommendation messages. First are messages that have no trigger location and are thus delivered to the user immediately. These are intended to start the user in their exploration of the Lab, giving them projects and corresponding locations to go check out. The second type is recommendation messages that are delivered when the user is detected at the project's location. Behind the scenes, the GI Mobile server looks up the project's location in its database, which has been cached from Tagnet, and uses this location as the message's trigger location. This type of message adds the spontaneous aspect of the recommendations. To determine which kind of message a recommendation will become, the matches from Luminoso are ordered by their match weight. The best matches (highest weights) become messages that are delivered immediately while the rest become location-dependent. The rational for this is that because the user is not guaranteed to pass by any particular location, they might never see a location-dependent message. Thus, it makes sense to have the best recommendations be the ones the user is guaranteed to see. In the current implementation, the top five matches are delivered immediately and the rest are location-dependent.
Besides its messaging system, GI Mobile includes a host of other features to complement the Glass Infrastructure kiosks. This primarily includes a mobile version of the Project Browser, the main application on the GI kiosks. To access this application, the user must navigate to the “Project Browser” tab. The initial view shows the research groups at the user’s last seen location. However, if the user’s location is stale or otherwise unknown, this will default to showing all research groups in the Media Lab. The user is then able to navigate to a specific group, showing the projects in that group, and subsequently view details of any of those projects. Screenshots of a sample location view and project view are shown in Figure 14. As the user navigates, the URL bar in the browser is updated using Backbone.js’s Router. Likewise, the user can navigate to any location, group, or project directly via URL. For instance, navigating to the project shown on the right in Figure 14 will cause the browser’s URL bar to show http://gimobile.media.mit.edu/project-browser/project/3727. Opening this same URL in a new page will bring the user to the exact same view in the application. This kind of deep linking provides access to the different views as if it were a normal, multipage website, despite the fact that it is actually a single page application. As a result, this necessitates that the data for each page in the Project Browser be loaded asynchronously from the server (AJAX). When the client navigates to a new research group, for instance, the client makes an AJAX request to the GI Mobile server, which acts as a proxy to in turn make a request to Tagnet, where this data is stored. The GI Mobile server also caches the information (populating the Location, Group, and Project collections in the schema in Figure 6) so that the client can continue to function normally if Tagnet is experiencing issues. However, while GI Mobile has been running, using the cache has not been necessary.
This sort of browsing is essentially a simplified version of what a user can do on the Glass Infrastructure kiosks. Not only is it convenient to have this functionality in the private setting of a user’s smartphone, but also the inherent mobile nature of the application means that unlike the kiosks, this Project Browser is not stationary in the Lab. Because GI Mobile moves around the Lab with the user, the Project Browser view actually updates itself in real-time to reflect the user’s current location. This is done with a smooth fading transition to show the groups present in the user’s new location. In the event that the user does not want the screen to change its view, they can simply use the browser’s back button to navigate to what they had been previously viewing. This is again possible because of the use of URLs to link to different views in the application.

As seen in Figure 14, the project details view also allows the user to add or remove charms right from the GI Mobile application. This change is persisted in the Glass Infrastructure via a web call to Tagnet. In addition, for the purpose of logging application usage, this action is recorded in the GI Mobile database as a CharmActivity, as seen in the schema in Figure 6. Besides adding and removing charms, the user can also view all of their charmed projects in the “View Charms” tab, which can be seen in Figure 15. This information is pulled dynamically from Tagnet via AJAX, so the content will represent the most up to date information. This feature is also present on the Glass Infrastructure kiosks, but this limits its use to physically being in the Lab and interacting with the screens. By including it in GI Mobile the user is able to browse and manage their charms from anywhere.
GI Mobile

GI Mobile is a mobile companion to the Media Lab Glass Infrastructure system. It incorporates the MessageMe messaging system to deliver a suite of location-aware features that complement the Glass Infrastructure. These include locating others in the Lab, browsing projects physically near you, and sending location-based messages. In addition, GI Mobile will alert you when you pass by projects you may be interested in based on what projects you have "liked."

You've charmed this project!

Remove this charm

Figure 14: Project Browser view. The left screenshot shows the initial view for a user last seen at the screen "charm-6", which shows the research groups in that physical location of the Lab. The right screenshot shows details of a specific project, and the ability to add or remove the project as a charm.
The final tab in GI Mobile’s interface is the “Locate” tab, used for locating other users in the Media Lab. Screenshots of this view can be seen in Figure 16. As seen in the right screenshot, the form includes a similar type-ahead feature to that used in the “Post Message” view. In the screenshots in Figure 16, the user “blazarus” has already been added, and so his location information shows up in the list below the text input. Like the user information view at the top of the page that shows the logged in user, these user information views update in real time as the members or sponsors being “followed” move around the Lab. This is a great way for sponsors to
find colleagues they have been separated from, or for a student to track down a labmate. Like GI Mobile as a whole, this feature makes the tradeoff that while it is limited to locating people within the Media Lab, it can be much more accurate and specific in this environment.

Figure 16: Screenshots of the “Locate a user” view. Right screenshot shows the type-ahead feature while searching for a user to locate.
Results

GI Mobile was deployed for the Spring 2013 Member Event at the Media Lab. This event included three Lab exploration periods where Media Lab sponsors could browse the project demonstrations around the Lab. During this time, Google Analytics was used to collect usage statistics for GI Mobile. Starting on April 23 (demo day), there were 16 unique visits by nine unique users. On average, there were 3.75 pages viewed per visit with an average duration of 7:46 per visit. Sixty total page views were recorded. A graph of the visits during this time is shown in Figure 17. In addition to Google Analytics, the GI Mobile server recorded charms that were added or removed using the app. Over this time period, seven charms were added and three removed.

![Google Analytics graph showing the number of visits over a two-day period during Sponsor Week.](image)

Besides quantitative results, I was also able to obtain valuable feedback from the sponsors who stopped by the GI Mobile demonstration. In general, many sponsors seemed genuinely interested in the idea and impressed with the implementation. Some were able to envision how a system like GI Mobile could be used in a retail environment. For instance, in a department store a message could alert a user when they are physically near clothes that might fit their style based on past preferences. Or, a deal website such as Groupon could alert users when they walk by the location of a deal they might be interested in. In addition, the “Locate a user” feature in GI
Mobile proved useful when one sponsor lost one of her colleagues and couldn’t reach them by phone. With GI Mobile, we were able to see that this particular sponsor’s location was stale, meaning they hadn’t been seen in over three hours in the Lab. As a result we were able to determine with high probability that this sponsor was no longer in the Lab and had most likely gone back to their hotel.
Future Work

After the initial prototype of GI Mobile, there is a variety of different directions this project could take in future work. First, we consider various ways in which MessageMe can be expanded. For instance, adding a templating system for messages would allow anybody to create messages where the subject or body could look different depending on the receiver and on the location. This could be as simple as including the recipient’s name in the body of the message, which is not currently possible in MessageMe. Or, it might include more complex logic to show one version of the message if the recipient is a Media Lab member, and another if the recipient is a sponsor. Besides templating, another layer of sophistication that could be added to the "Post Message" view is the ability to easily send to groups of people or locations. For example, a user might want to send a message to all members of the Digital Intuition research group, or they might want to send a message that will be triggered anywhere on the fifth floor (which is comprised of multiple screens). Currently this is only possible by adding each recipient or location individually, which can become cumbersome very quickly. Finally, to add even greater control to message delivery, adding a time-to-live field would allow the sender to specify that after a certain period of time, a message will no longer be delivered to recipients that have not already read it. This is useful if the content of the message is time-dependent, and will not be relevant for a visitor that enters the Lab in the future. For instance, the sender might want to advertise a talk happening in an hour, which would not be relevant after the talk has already occurred. These enhancements to MessageMe would give the user even more power and ability to customize messages than is possible in the current implementation.
One sensible addition to GI Mobile would be to incorporate CharmMe features. This would take the form of people recommendations, similar to the project recommendations that are currently a part of GI Mobile. Most of the infrastructure for this change is already in place and so would not be a difficult addition to make. Beyond this, there are other ideas surrounding GI Mobile that are somewhat more involved. For instance, one area of exploration is to add actual interaction between the smartphone app and Glass Infrastructure kiosks, although it is unclear what specific use cases would be beneficial with this kind of platform. Another idea is to use GI Mobile for navigation around the Media Lab. This would require map-like coordinate data of the Lab and the locations of each kiosk, which is not currently available. One possibility to obtain this kind of data would be to use the accelerometers and compasses in smartphones along with the GI Mobile app to map out the distances between screens as users walk around the Lab. It is still unclear how feasible this would be, but is a direction for possible future research. In addition, with this kind of rich coordinate data of the Lab, an entertaining extension to the “Locate a user” feature of GI Mobile would be to show users’ locations on a live updating map of the Lab. The GI Mobile application is just the start of the many uses of this kind of mobile platform.
Conclusion

MessageMe and GI Mobile provide a suite of features that compliment that MIT Media Lab’s Glass Infrastructure. By utilizing the existing RFID system in the Lab, MessageMe allows the user to send arbitrary messages to other users that are triggered to appear at certain locations in real-time. GI Mobile extends this to include a mini Project Browser, project recommendations, an interface to view, add, and delete charms, and the ability to locate any other user in the Lab. Although this prototype system has been designed for use specifically in the Media Lab, the concepts are extensible to many other environments. Through research in related works and conversations with sponsors, it is clear that this sort of functionality has applications in large events such as music festivals, retail settings, travel situations, and more. Although location tracking is becoming much more ubiquitous, we are only just beginning to see the possible uses for location-aware applications. MessageMe and GI Mobile not only represent a successful proof of this concept, but also a meaningful addition to the Media Lab’s Glass Infrastructure.
References