University of California, Berkeley
School of Information

Karen Hsu
Kesava Mallela
Alana Pechon

Nujj • Table of Contents

Abstract

Nujj is a location based service for mobile device users that enables users to tie electronic notes to physical locations.

It is intended as an initial exploration into some of the many scenarios made possible by the rapidly increasing ubiquity of location-aware mobile devices. It should be noted that this does not limit the user to a device with a GPS embedded; location data can now also be gleaned through methods such as cell tower triangulation and WiFi IP address lookup.

Within this report, both social and technical considerations associated with exposing a user's location are discussed. The system envisioned addresses privacy concerns, as well as attempts to overcome the poor rate of adoption of current location based services already competing in the marketplace.

Although several possible use cases are offered, it is the authors' firm belief that a well-designed location-based service should not attempt to anticipate all, or even most, of the potential ways that users will find to take advantage of it. Rather, the service should be focused on building a system that is sufficiently robust yet flexible to allow users to pursue their own ideas.

Processes and lessons learned during the course of building a functional prototype are examined, as well as possible ways to expand on this work.

Introduction

Problem Statement

Few would deny that the internet provides a wealth of information about almost any topic imaginable. In fact, an entire generation of tools and services have grown up to help users manage online information. There are tools that will extract and store citation data from library listings, services that will compare prices from dozens of online shopping sites, and tools to aggregate the top news stories from multiple news outlets. While there is unquestionably a great deal of utility to be gained by managing, storing and offering information back to the user in the context of the desktop or laptop computer, what has been conspicuously absent is an easy way to perform those same basic tasks in other environments. Specifically, there is an entire category of information that is much more interesting when taken in the context of the real, live, sun-is-shining world. It is only relatively recent, however, that we have begun to commonly see devices that are capable of taking information on the road.

Mobile devices have reached a level of saturation in most modern cultures that could scarcely have been imagined just 20 years ago. Propelled partially by legal mandates and partially by market demand, cell phone manufacturers have begun producing more and more sophisticated devices. Some of today's smartphones qualify as fully functional computers, complete with capacious onboard memory and advanced applications that not only offer standard phone and computer capabilities, but also take advantage of integrated hardware features like cameras and GPS systems.

Finally, there are platforms smart enough and powerful enough to give users access to the information they want, where and when they want it. Somehow, despite the steady and predictable progression to this point, the information technology community has been caught flat-footed. Services are still centered around a decades-old model of forcing the user to actively seek information, instead of leveraging knowledge about where they are and plausibly even what they are doing, to offer just-in-time information that is contextually appropriate.

Objective

The central goal of Nujj is to offer a method to deliver information to users while they are at the physical location to which it is related. The fundamental pieces of technology necessary in order to solve this problem are now available. It is up to Nujj to draw the disparate pieces together into a coherent service that will offer users a truly valuable and fun tool.

Hong et.al.[1] studied user preferences when giving their location information. They gave their test group devices with an installed location program. Instead of directly asking them to reveal their location information, they were instead asked if they are likely to reveal their location information given the person requesting the information is say, a spouse, a boss or a friend or a parent. The key insight from the project is that user preferences for revealing information highly depended on who is requesting it.

Iachello et.al[2] in a similar study proposed a list of guidelines that designers of location based systems should consider. Their design guidelines start with a no automation rule, in which they advise against starting with automation. Rule #2 asks designers to support flexible replies where the user decides to send what she wants instead of what the system decides to send out. Rule #3 asks to provide the ability to ignore requests. Rule #4 lets the users have the ability of deception. In other rules, they ask the designers not to handle a lot of unwanted user data, support person-to-person conversation before expanding to a group conversation, etc. They support their study with discussion on how they implemented their own guidelines.

Kaasinen[3] in a study about user needs for location aware mobile services draws several interesting insights. First, the author found that users were looking for topical information that is not static, mainly because they could get static information from other sources. The author also found that users' needs may be dependent on past, present and future locations. The author also found that the willingness of users to be active should not be overestimated.

---

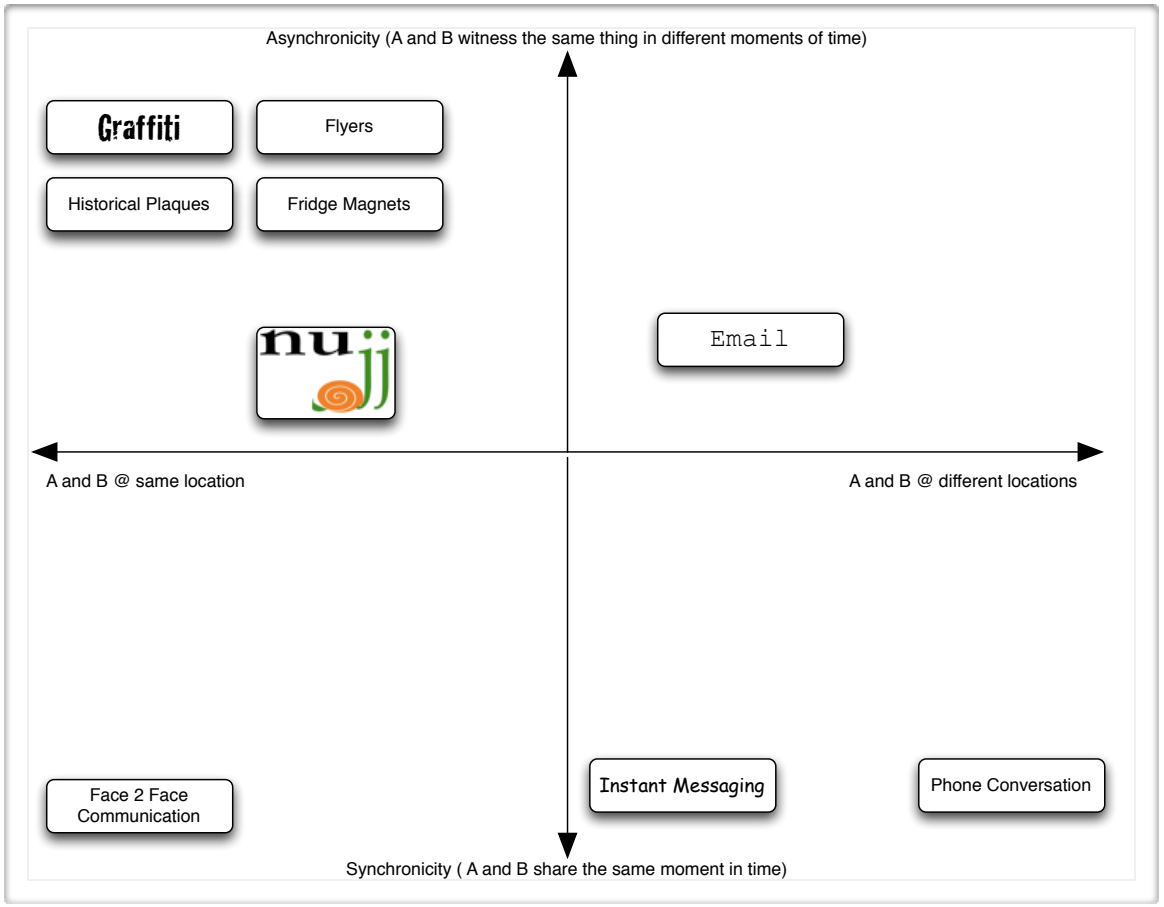
[1] Cornwell, J.; Fette, I.; Hsieh, G.; Prabaker, M.; Rao, J.; Tang, K.; Vaniea, K.; Bauer, L.; Cranor, L.; Hong, J.; McLaren, B.; Reiter, M.; Sadeh, N., "User-Controllable Security and Privacy for Pervasive Computing," Mobile Computing Systems and Applications, 2007. HotMobile 2007. Eighth IEEE Workshop on , vol., no., pp.14-19, 8-9 March 2007

[2] Iachello, G., Smith, I., Consolvo, S., Chen, M., and Abowd, G. D. 2005. Developing privacy guidelines for social location disclosure applications and services. In Proceedings of the 2005 Symposium on Usable Privacy and Security (Pittsburgh, Pennsylvania, July 06 - 08, 2005). SOUPS '05, vol. 93. ACM, New York, NY, 65-76. DOI= http://doi.acm.org/10.1145/1073001.1073008

[3] Kaasinen, E. 2003. User needs for location-aware mobile services. Personal Ubiquitous Comput. 7, 1 (May. 2003), 70-79. DOI= http://dx.doi.org/10.1007/s00779-002-0214-7

Competitive Analysis

Comparison matrix



Competition

	Socialight	FireEagle	Loopt	Whrrl
URL	http://socialight.com/	http://fireeagle.yahoo.net/	https://loopt.com/	http://www.whrrl.com/
According to the website:	lets people and editorial publishers connect rich media content to real-world physical locations.	way to share your location with sites and services online while giving you unprecedented control over your data and privacy.	cellphone-based GPS sharing system with the goal of providing an innovative social mapping tool that allows friends to visualize one another using their cell phones and share information about interesting places	mobile and web service that allows users to find, explore, share, meet up at, and rate points-of-interest in their cities.
Download to phones?	Yes	Yes	Yes	Yes
Mobile Alerts	Yes	No; based on declaration	Yes	Yes
Cell tower / GPS ?	GPS	Cell tower	Cell tower	Cell tower
auto location sensing	Yes	No; based on declaration	Yes	Map based desktop client
publicly available?	Yes	Alpha	Yes	Yes
compatible hardware	Motorola GPS enabled phone	Wide range of devices and channels	Boost Mobile & Sprint Nextel	Any cellphone
Social Network Support?	Invite your friends	Share on existing networks	Invite your friends	Invite your friends
Location granularity while disclosure	No	Yes	Yes	Not about personal location disclosure
Share with all/select group of friends	Only 'all friends'	Both	Both	

Hypothesis

It is the Nujj team's hypothesis that in order for a location-based service to be successful, it must overcome the following obstacles:

- Redundancy: users are not inclined to join another social network in which they would have to, yet again, define the same relationships that they have already defined many times over.
- Sparsity: all services begin with a low number of notes, which renders the service un-compelling to users if they find notes only very rarely.
- Incentive: users have very low incentive to leave a note if they have little reason to believe the note will actually be discovered.
- Mobility: The user must be able to access all or almost all features and functions from the mobile interface.
- Learning Curve: The new knowledge and behaviors that a user must master in order to take advantage of the service's functionality should be limited as much as possible.

Nujj attempts to address these problems in the following ways:

- Redundancy: Nujj piggybacks off of Twitter, reusing the existing social network they have already defined.
- Sparsity: Nujj offers users the option of subscribing to feeds from content sources of interest. For instance, users can subscribe to a yelp.com feed, containing Italian restaurants with ratings of 4 stars or above.
- Incentive: Every note left with Nujj is also simultaneously sent to Twitter. In this way, the user is assured that their note will be received by their peer group.
- Mobility: Every function so far built in to Nujj can be accomplished from the cell phone interface. Although users can log on to a web portal to view larger maps and access their account from their desktop, there is nothing a user can do online that they cannot also do from the phone application.
- Learning Curve: Nujj intentionally mimics the interaction of sending a text message, maintaining a minimal interaction for sending notes. Receiving notes would ideally be as simple as receiving a phone call or text message.

Recommendations

One Thursday night, Joanne and her friend Ross decide to try the sushi place down the block from where they live. They request the chef to select a menu for them, and order a small bottle of sake to share. As they sit talking about their plans for the weekend, they each finish their cups of sake, and pour more from the carafe. No sooner have they done so, than the server comes by with a large bottle and tops up the carafe. Surprised, they shrug and smile at each other. 15 minutes into the meal, the server returns and tops up the carafe again. This time, the smiles turn to grins. When the bill arrives, it is obvious they were charged only for the one small bottle of sake. Joanne is thrilled with the deal. She pulls out her cell phone, opens up the Nujj application, and types "Wow, the fish at Coach Sushi is super fresh, and you get free refills on sake when you order a carafe! Try the hamachi, it's fantastic." She then presses the "Send to Nujj" button at the bottom of the screen, and puts her phone away.

All of Joanne's followers on Twitter receive her message as a regular tweet according to their own personal notification preferences. Some friends may receive Joanne's message as an SMS on their phone, while others will receive an instant message through their preferred IM client. Her friend Gary notices the message, and files it away in that "things and places recommended by people I know" section of his memory, which essentially means that he forgets it. However, the following Monday, he happens to be walking down the block where Coach Sushi is located, and gets a notification on his phone that one of his friends has left a Nujj there. He checks his phone, and sees Joanne's note. Ready for a light lunch, he decides to stop in and try the hamachi, since he respects Joanne's opinion on restaurants in general, and sushi in particular. Pleased with the meal, he decides to come back again with friends. Gary pulls out his phone, opens up the Nujj application, and types, "Hey, anybody want to get sushi with me this weekend at Coach? Good food, great prices, bottomless sake! Email me." He then presses the "Send to Twitter" button at the bottom of the screen, and puts his phone away.

Recommendations using Nujj present a new way for people to share information with their peers. There is a strong human tendency to value the opinions of friends above the opinions of strangers, but it is impractical to carry long lists of places that have been mentioned in passing, or to try to remember them all. Nujj shoulders some of this cognitive load by delivering the recommendations and reminders in the place where they are most useful.

Lost & Found

It's a sunny Saturday morning, and Vlad is waiting at the corner of 16th and Mission in San Francisco for his sister Jill and her daughter to come up from the BART train below. He sits on a bench reading Andrew Keen's "The Cult of the Amateur". He had to wait for ages to get it from the library, and finally picked it up just this morning. He's only a few chapters in, but already has decided to return it to the library. Suddenly, he feels a tiny pair of hands cover his eyes, and a shrill voice call out "Guess who-oo?". Turning, Vlad swoops up his delighted niece Hazel, and then he and Jill start discussing where to start the day's hunt for their mom's birthday gift. It's not until hours later, as he sits down on the bus home, that he realizes he must have left his book on the bench that morning.

He sighs, gets off at the next stop, and walks the few blocks back to 16th and Mission. As he walks, he gets out his cell phone, opens the Nujj application, and types a message: "#subscribe lostfound". He presses the "Send to Nujj" button, and the application, recognizing the command and directing his message exclusively to Nujj's server, sends back a confirmation message by the time he reaches his destination: "You are subscribed to the lost and found service on Nujj!". Arriving at the corner where he waited that morning, he is delighted to see that he has a notification on his phone. He reads: "Found here: library book. Reply to this note with the title to claim it." He presses reply, and types in, "Cult of the Amateur? If so, please call 415-555-1234." It's only a moment or two before he gets a call from one of the local shopkeepers, and within 15 minutes, he's headed home again, dropping the book off back at the library on the way.

Lost and found is an especially compelling scenario because it takes advantage of behavior that users already exhibit, and provides an entirely new way to manifest that behavior. Every day, we see paper notes on street corners and at bus stops about items that have been lost or found. These physical notes are subject to removal, weather, or simply being overlooked. With Nujj, the note is digitally attached to the location.

Q&A (Time- and location-sensitive)

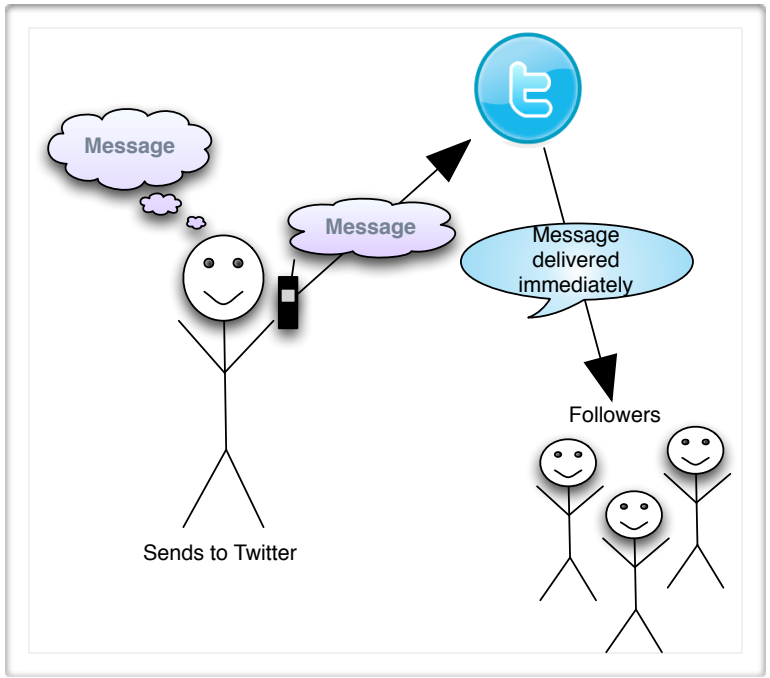
Ivan is walking to class in Soda Hall on the University of California campus when the campanile starts chiming the noon hour. The song played by the carillonist is familiar, but he can't quite place it. The Beatles, maybe? Dylan? A few minutes early for class, he pulls out his phone, opens the Nujj application, and types: "#question What song is playing from the campanile right now?" The Nujj application recognizes the command, and sends the message straight to Nujj's server. Nujj, in turn, makes a note available to all users who have subscribed to the Q&A service, and who happen to be within that area anytime in the next 15 minutes.

About 5 minutes later, Ivan receives a response to his query: Here Comes the Sun, by The Beatles.

A question answering service that is both time- and location-sensitive would be a unique offering in the marketplace. One can also imagine multiple permutations that take the form of a game, with users competing, for instance, to have the highest ratio of questions answered to questions viewed.

Overall System Design

The Nujj experience begins with a user deciding to attach a location-specific electronic message to a physical spot. The primary Nujj interface is intended to be a mobile phone client. The application would be installed onto the device, and a simple configuration screen allows the user to enter their Twitter username and password. This allows Nujj to retrieve the user's Twitter relationships, and also allows Nujj to authenticate to Twitter on the user's behalf.



The main screen of the mobile client offers a tabbed view, allowing the user to choose between leaving a note and viewing nearby notes that have already been left. On the note creation screen, a text box deliberately mimics SMS entry, with a choice to then either "Send to Nujj" or "Send to Twitter".

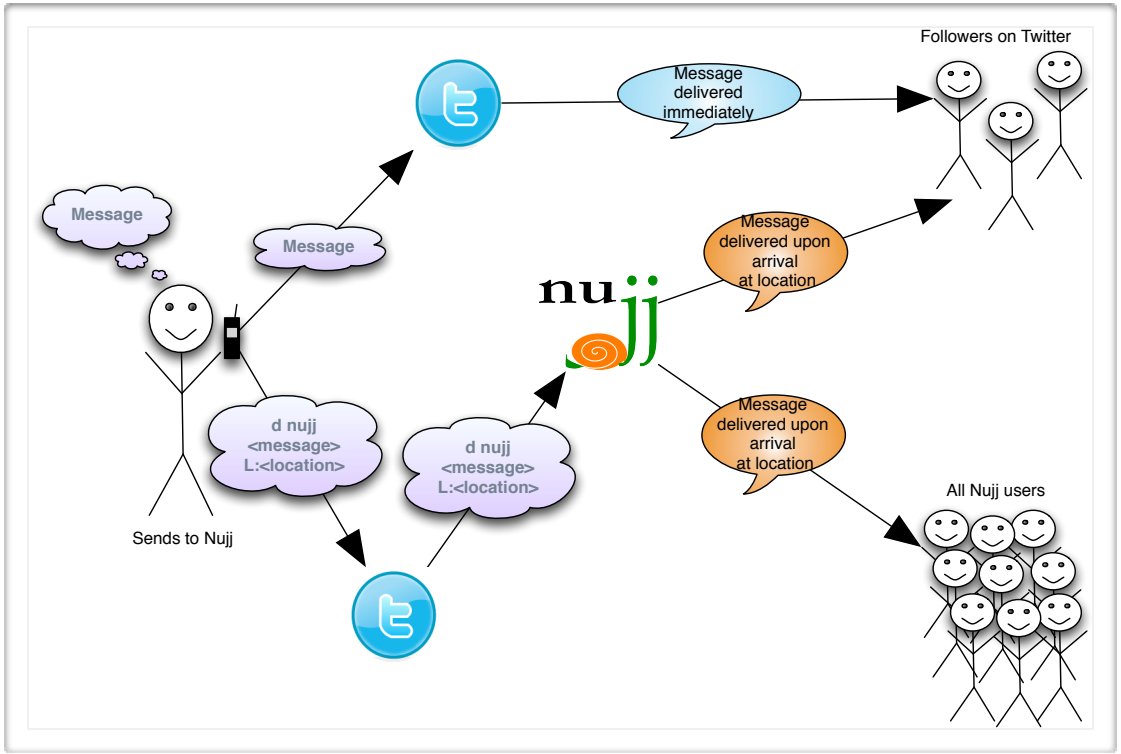
If the user chooses "Send to Twitter", the message is passed (using the Nujj phone application) through the web to Twitter as a normal update, and then delivered to all of that user's followers via the preferred methods they have set up previously. When a user sends an update to Twitter using the Nujj phone application, it is handled as data, rather than a text message. This is an advantage to most of Nujj's target users, who likely have an unlimited amount of data in their monthly mobile phone plan, but who generally must pay for text messaging.

If the user chooses "Send to Nujj", two messages are simultaneously and seamlessly sent from the Nujj phone application. The first message duplicates the "Send to Twitter" function described above, while the second sends a direct message to the Twitter user "Nujj". Encoded within this message are both the message entered by the user and the phone's current geographical coordinates. Direct messages are invisible to all other users, thus protecting the user from inadvertently revealing their precise location. Nujj

reveals no more to the web at large than the users themselves type.

The "Nujj" Twitter user receives and stores the location-encoded message until such time as a crawler operated by the Nujj server retrieves it. The crawler then stores the information in Nujj's database.

In the current implementation, users retrieve notes through the Nujj phone application, which displays all nearby notes. Users can choose to filter visible notes based on any of several parameters. For instance, a user could choose to show only notes left by friends (as defined by the user's Twitter relationships) or only notes left in the last week. Ultimately, the Nujj application will be able to run as a daemon in the background, updating the server with the user's current location. This way, the Nujj server can notify them when they are near a note: again, a user may choose to receive all notes or just a subset of notes, based on their preferences.



The Nujj team has chosen to make the default privacy setting of notes "public". With this setting, friends are able to view notes on the password-protected map interface from the moment they are sent. Friends also receive notification from Twitter. Members of Nujj who are not a user's friends will be able to see the note only when they are at the location where it was left.

An important (as-yet-unimplemented) part of the Nujj philosophy is an open API, enabling other services to re-use data. This will be partially obscured to reduce "stalkability", as precise coordinates will not be offered.

Design

Nujj is envisioned as a RESTful service. since much of the power of today's web is that data from many services can be mashed-up and re-used. In the same way that Nujj re-uses data from Twitter, Nujj's data will be available to others.

Implementation

Nujj is implemented using Ruby on Rails. Rails generates a full set of CRUD (Create, Retrieve, Update and Delete) operations and views on any database table. These CRUD operations have strong parallels to the POST, GET, PUT, and DELETE methods of HTTP requests, which have been employed by Nujj.

Furthermore, Rails uses MVC architecture. Nujj has a model, a controller and list of views for the notes database table.

Clients can post new notes to Nujj by using a simple HTTP POST. When Nujj's web server receives an HTTP POST request, the notes controller asks for HTTP authentication. Once the request is authenticated, the controller reads the data and writes it to the notes table. Similarly, Nujj implements a search interface that devices and clients can use to get notes relevant to a geographical area.

Nujj implements a geo-extension of open search interface by publishing a search interface. The search interface takes several URL query parameters, such as latitude, longitude, radius from a point, data format, and username.

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
      <address>University Ave and Shattuck
Ave,          Berkeley, CA</address>
      <created type="timestamp">Wed May 07
            01:06:00 UTC 2008</created>
      <id type="integer">30034105</id>
      <lat type="float">37.87222</lat>
      <lng type="float">-122.2684</lng>
      <msg>This is the smallest Bear!!</msg>
      <noteid type="integer">30848518</noteid>
      <username>kesava</username>
</note>
```

The latitude and longitude refer to the coordinates of the current location of the device. Radius refers to the bounding circle within which the user thinks notes are

relevant. The default radius is set to .25 miles, approximately two city blocks.[4]

The other query parameters expected are format and username. In the current implementation, format is set to GeoRSS, which is native to Google Maps and other map service providers. Username is the screen name of the user whose notes will be displayed.

Nujj's iPhone application makes a query by sending out a well-formed URL with query parameters. The Nujj server then asks the application for HTTP authentication. The application then attempts to authenticate by sending the username and password from configuration settings.

An authenticated search request will be served with an aggregated set of notes, consisting of notes from the user, followed by notes from friends of the user and finally notes from everybody else that belong ONLY within the specified radius of the given coordinates.

The search controller uses the notes table to retrieve notes. It also uses the friendships table to get a list of all the friends of a given user, which it then uses to sort the list of the notes that is served back to a search request. The search controller also publishes the search interface at a publicly accessible folder on the web server, which allows new clients and devices to take advantage of the Nujj service.

The notes controller uses a Ruby on Rails extension called GeoRuby. GeoRuby implements the Haversine [5] formula if a database schema uses field names <lat> and <lng> to represent latitude and longitude. It also provides find methods that perform calculations like distance between any two points, all points within a given radius, etc.

Other modules of Nujj consist of regular user registration and configuration. Friends of a user are defined through a has_many relationship on Ruby on Rails. This relationship provides several ready to use methods like list of all friends of a given user. The relationship is defined by creating an additional table friendships that maps user ids of friends.

Nujj has a minimalist website that lets users delete notes they have created. The web interface uses some of the controllers described earlier, but generates HTML instead of GeoRSS and other syndicated formats.

---

[4] On an average, a US city block is 1/8 th of a mile. Source: Wiki Answers

[5] $$\operatorname{haversin}\left(\frac{d}{R}\right) = \operatorname{haversin}(\Delta\phi) + \cos(\phi_1)\cos(\phi_2)\operatorname{haversin}(\Delta\lambda)$$

Twitter and Nujj

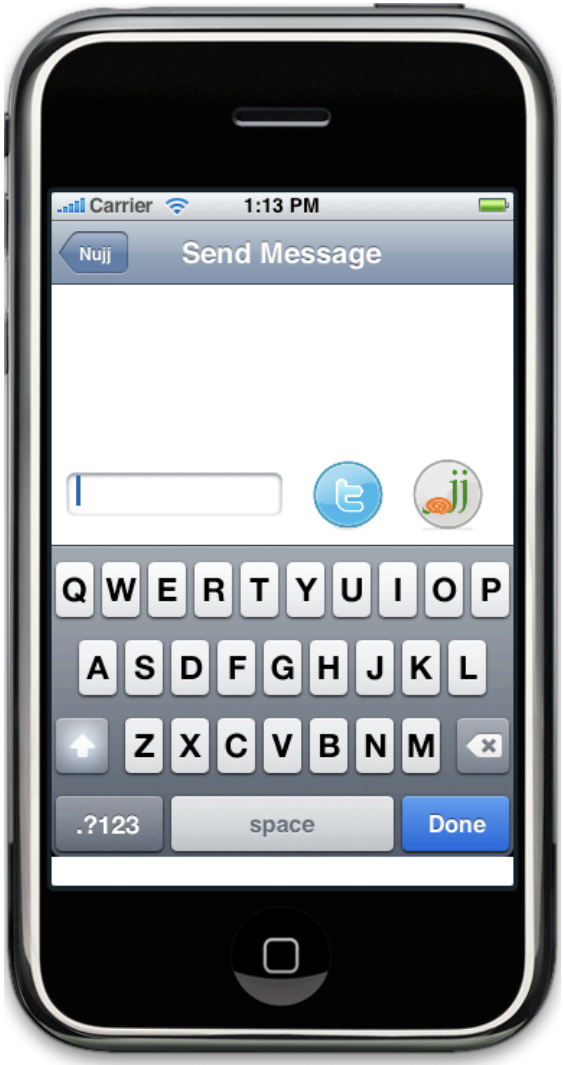
Since Nujj is RESTful in nature, we rely on cron jobs to transfer tweets from Twitter to Nujj. Python scripts crawl our twitter user twiiter.com/nujj for direct messages and HTTP POST them to Nujj after geo-coding the location included in the message. In other words, in addition to being able to use the phone application to create location notes, the user can also use regular SMS to send messages to Twitter with human-readable addresses that can be in any one of the following formats:

```
d nujj <msg> L:Berkeley, CA
d nujj <msg> L:94720
d nujj <msg> L:Shattuck and Bancroft
d nujj <msg> L:California or CA
d nujj <msg> L:37.223 -122.34
d nujj <msg> L:South Hall, Berkeley, CA
```

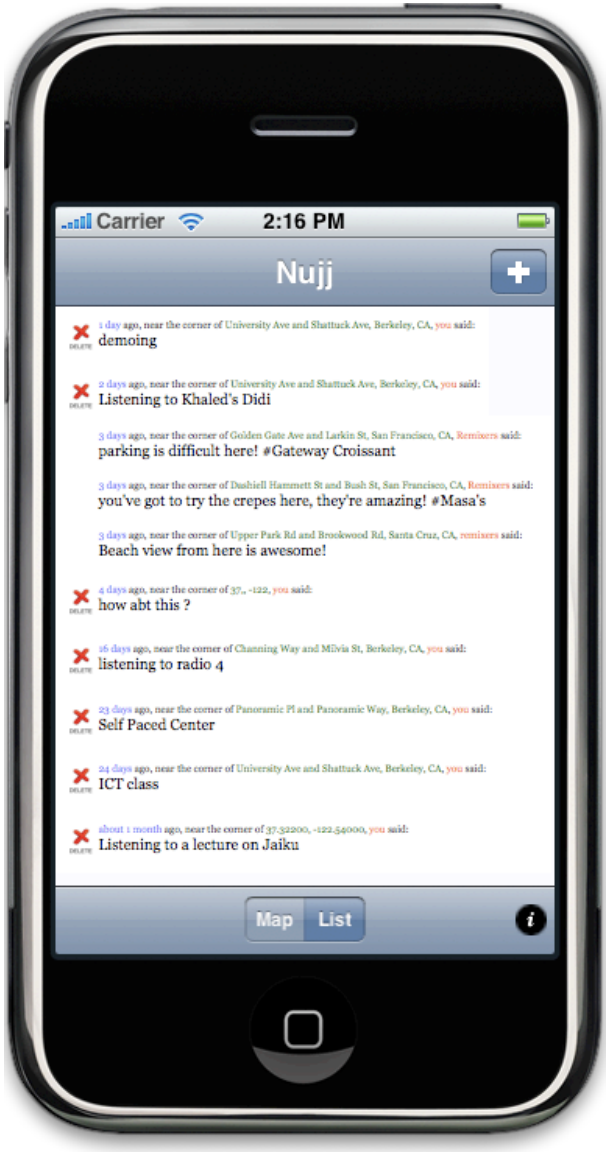
Our python scripts also query for the user's friends and re-creates those relationships in Nujj's friendships table.

Client-side


Carrier
11:54 AM
Photos
Settings
Simpl...Down
Nujj
Contacts
Safari


Carrier
1:13 PM
Nujj
Send Message
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
.?123
space
Done


Carrier
2:07 PM
Nujj
Sapporo
札幌
Morioka
盛岡
Niigata
新潟
Sendai
仙台
日本
Japan
Matsue
松江
Fukuoka
福岡
Osaka
大阪
Shizuoka
静岡
Oita
大分
Map
List


Carrier
2:16 PM
Nujj
1 day ago, near the corner of University Ave and Shattuck Ave, Berkeley, CA, you said:
demoing
2 days ago, near the corner of University Ave and Shattuck Ave, Berkeley, CA, you said:
Listening to Khaled's Didi
3 days ago, near the corner of Golden Gate Ave and Larkin St, San Francisco, CA, Remixers said:
parking is difficult here! #Gateway Croissant
3 days ago, near the corner of Dashiell Hammett St and Bush St, San Francisco, CA, Remixers said:
you've got to try the crepes here, they're amazing! #Masa's
3 days ago, near the corner of Upper Park Rd and Brookwood Rd, Santa Cruz, CA, remixers said:
Beach view from here is awesome!
4 days ago, near the corner of 37,, -122, you said:
how abt this ?
16 days ago, near the corner of Channing Way and Milvia St, Berkeley, CA, you said:
listening to radio 4
23 days ago, near the corner of Panoramic Pl and Panoramic Way, Berkeley, CA, you said:
Self Paced Center
24 days ago, near the corner of University Ave and Shattuck Ave, Berkeley, CA, you said:
ICT class
about 1 month ago, near the corner of 37.32200, -122.54000, you said:
Listening to a lecture on Jaiku
Map
List

Design

Although the team considered using the Android platform, there are, as of this writing in late spring of 2008, no phones with a solid release date on the horizon for this operating system. Other open source phone operating system projects , such as OpenMoko and Maemo, suffer from a similar lack of hardware available to consumers. Having decided to develop for a proprietary operating system, the major choices were Symbian, Window Mobile, or iPhone. Ultimately, it was decided to build a client for Apple's iPhone due in large part to the fact that there is a great deal of publicity, as well as a tremendous "fan base", surrounding Apple products.

iPhone is a relatively new product, yet users have very strong expectations about how applications on it will look and act. There is little doubt that with the forthcoming availability of independently designed applications, there will be a surge of interesting, creative interfaces. However, it is likely that the applications that will be most successful will feel almost as though they were native to iphone during original release, straight from Apple. Therefore, the Nujj team took great pains to follow iPhone design and behavior conventions.

Implementation

Although the famed Apple GUI builder was much hyped before release in the SDK, once explored, it turned out to be more of a hindrance than an aid, due to its immaturity and general instability. Also, there is a lack of documentation. In the end, it was simpler to develop code from scratch than to use the tool supplied by Apple.

Limitations of the Apple iPhone SDK shaped the development of Nujj. First, the lack of support for the creation of daemons (which goes against iPhone convention) removed the possibility of a serendipitous use case. In other words, the Nujj phone application cannot push its coordinates to the Nujj server unless the application is currently running, and this eliminates the ability for the user to be serendipitously alerted when they are within a certain vicinity of a placed note. Moreover, the lack of a GPS limits the accuracy of each placed note.

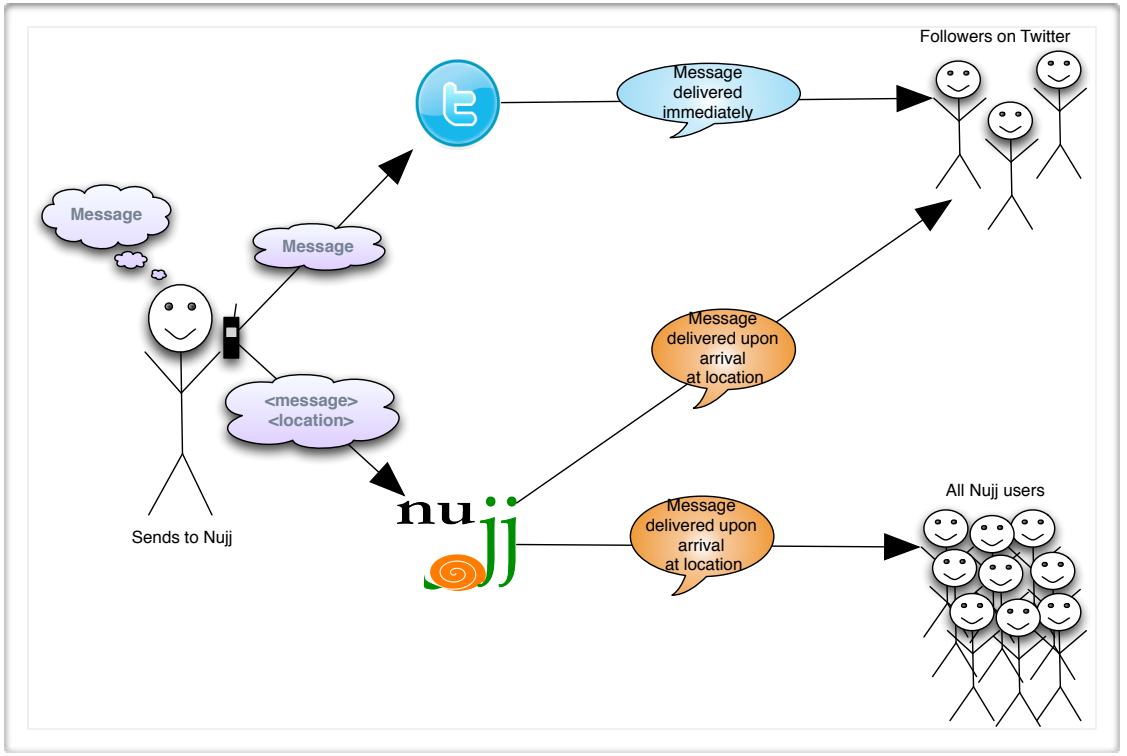
Another limitation is the unavailability of an iPhone maps application API. Without a clear way to create a map mashup on the phone, Nujj instead creates a webview of a Google maps mashup for note discovery. While this solution works, it does not make use of the elegant user interaction capabilities of the iPhone maps application, such as pinching for zooming.

Lastly, although the iPhone SDK is rich with advanced graphics and features, it is still in its early stages and lacks in documentation. In particular, the GUI builder that comes with the SDK is immature and for the most part undocumented, which rendered it more difficult to use than simply generating the application's user interface via code.

Extended Functionality in Future Implementations

The following are extended functionalities that could be included in future implementations:

- Enable location polling on the handset, in order to send that information back to the Nujj server. (Currently, this ability is disabled by the hardware manufacturer.)
- Use the native iPhone maps application for notes display.
- Remove Twitter from the transaction between the user and the Nujj server.
- Add the ability to designate specific recipients for notes.
- Add the ability to force expiration of notes.



- Add a "trusted superuser" status that allows specific entities (emergency services, transit authorities, etc.) to send messages to all users within a specific area.
- Add support for users to select subscription services from providers who interest them (specific businesses, groups, or individuals).

Acknowledgements

We are most grateful to our families for their long-standing unconditional support.

We thank our patient and gifted advisors, Kimiko Ryokai and Erik Wilde, for helping us frame the problem in a way that made it approachable.

We would like to thank everybody at the School of Information for making our Masters program such an exciting experience. Especially, we would like to thank Igor Pesenson, Shufei Lei and Chris Volz for sharing our research goal of finding a stable implementation platform for Nujj. We also thank Chris Volz for his assistance in crawling Twitter to recreate friend relationships within Nujj. We thank Kevin Heard for helping us set up the environment. We thank Kevin Mateo Lim and Andrew McDiarmid for brainstorming during the inception of the idea. We also thank Kevin Mateo Lim for using location coordinates in his tweets.

We thank the Rails and Twitter community for all the great things built around them. We thank Apple for designing such wonderful computers and devices. We thank Tetris for entertaining us when it became too stressful. We thank Cushman for giving us wonderful open roof rides late in the night.