

Nabu: A Mobile Location Based Question and Answer System

Mohammad Asgharneya
masgarnia3@gatech.edu

Lance Dudkowski
ldudkowski3@gatech.edu

Supraja Narasimhan
supraja.n@gmail.com

Vedrana Novosel
vnovos@gmail.com

INTRODUCTION

Nabu is a location based question and answer system for the iPhone. The goal is to let users find and share real-time useful information related to their surroundings. An example application of Nabu is if someone notices fire trucks in front of a building. He or she could use the program to ask "What is going on here?" Someone else in the area could answer "There has been a chemical spill." The next few people to come in range of the same building might have the same question and they would find the answer quickly and easily. Questions and their answers are archived with their locations, and users are able to access all the questions in a certain range (currently set at 3 miles) around them. Some questions might be timeless (e.g., "What year was this building built?") and others only applicable temporarily (e.g., the chemical spill example). Multiple answers are allowed for each question. Using a rating system, posts are quality controlled by users for accuracy and temporal relevancy.

To implement this idea we developed an iPhone application, focusing both on effective back end software and a satisfying user experience. Location is obtained through GPS. The iPhone program uses wireless or 3G internet to communicate with an online database which stores the questions, answers, locations, ratings, and more. Through this project we gained experience with mobile phone software development, databases, user interface design, and location-based applications.

Nabu is not a social networking tool, distinguishing it from several similar projects. Rather it is meant to be a community information sharing system that is useful on the spot. Nabu uses location awareness in a compelling and helpful way, allowing users to get answers to all kinds of questions easily. It has uses in way finding (e.g., "How do I get to the nearest ATM?"), tourism (e.g., "What is the Eiffel Tower made of?"), advice (e.g., "Is this restaurant any good?"), information aggregation (e.g., "How often does this bus come?"), emergency notification (e.g., "Is there a real fire or is this just a drill?"), entertainment, and more.

PREVIOUS/RELATED WORK

While location-aware applications are increasingly prevalent in today's market, few relate to our concept for a location-based Q&A forum. We briefly review this closely related subset here.

GeoGraffiti is a location-based voice message board, in which voice messages are paired with geographic

coordinates. Intended as a platform for public voice mail, the application promotes sharing of location-based intelligence and leverages the "wisdom of crowds." While available as an iPhone app, GeoGraffiti does not require any specialized phone or interface. In fact, it can be accessed by simply dialing a phone number [1].

Graffiti, exclusively hosted on the iPhone, encourages users to post information on location-specific "Walls" initiated by people in the community. Users have the option to search for existing, nearby Walls, or to create a new Wall to post their thoughts about the place they are in, be it a restaurant, park or a time-sensitive event [2].

Loopt, which supports iPhone, Blackberry and Android, as well as other platforms, displays an interactive map to help users quickly locate nearby friends in proximity to restaurants, bars, events, etc. [3].

Besides offering social networking features, Brightkite, available on Android, Blackberry and iPhone, allows users to retrieve posts based on ranges of distance (e.g., Close - 20 m, Block - 200m, Neighborhood - 2 km, Area - 4 km, City - 10 km, etc.). Users can rate posts by clicking on thumbs up or thumbs down icons [4]. Brightkite entrepreneurs are collaborating with Layar to provide a new augmented reality feature where icons for posts are superimposed on an image of the world around the user, as it appears on his/her phone screen. The user can choose posts to browse based on a combination of factors. The criteria include whether the queried post satisfies the user's keyword-based filter, how close it is to the user's current location, and how recently it was published [5].

The Twibble package for enhancing Twitter-based micro-blogging is built for smart phones including Blackberry, Nokia and Sony Ericsson platforms. Available as either a desktop or a mobile client, the mobile version allows the user to attach a GPS position to each tweet communication [6]. Taking a cue from applications such as Twibble, Twitter announced a location-aware API in September 2009. Users who opt in to the location-aware service can have geographic coordinates attached to their tweets [7].

Our forum concept is distinct from existing applications because it enforces a structured Q&A format on the person who is posting. In addition, while some related services may focus on geographically-localized data, our Q&A model will help resolve time-sensitive problems

and questions as well. Our aim is to provide community-generated answers in real-time via mobile phones. With less emphasis on random, unsolicited comments, we hope a question-driven format can provide additional value to community members.

OUR WORK

Development and User Experience

We chose to develop for the iPhone because it is used ubiquitously and its programming language is related to C. Before creating the interface, we sketched out interaction and layout possibilities on paper. After deciding upon our desired functions and flow, we learned the basics of iPhone programming and started our application using a hybrid of the visual Interface Builder and coding. We created a list file to store questions, answers, ratings, dates, and more. The program reads and displays these values in lists of questions and answers (Figures 1 and 2). Integrating custom list displays and search capabilities was the next step. We had to develop separate views (pages) for the main questions list, the list of answers, the asking questions function (Figure 3), the answering questions function, and the rating function (Figure 4). These all had to communicate with each other, and toolbars and navigation had to load properly throughout. We considered several types of rating systems such as numerical scales and stars but settled on using thumbs up and thumbs down because of its simplicity.

Feedback for user actions was incorporated into the program. For example when a user presses the button to submit a question, a message is displayed indicating a successful post, and the view returns to the previous page. We also satisfied other user experience principles such as providing contextual clues and clear labels.

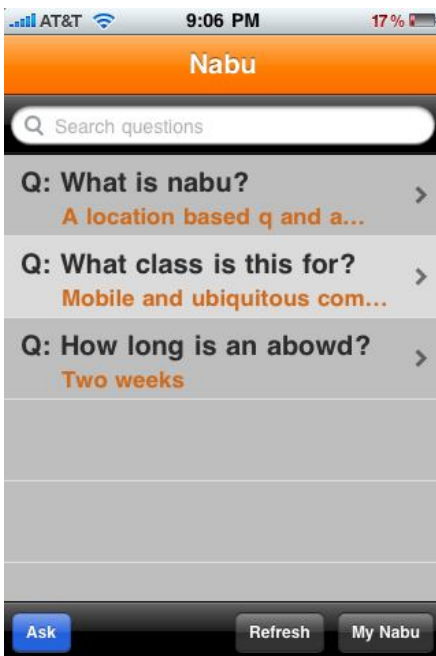


Figure 1: Questions page screenshot.

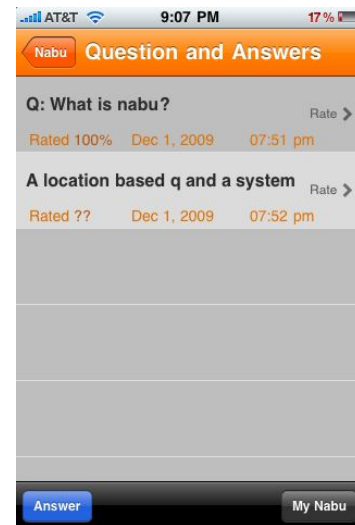


Figure 2: A view showing one question and its answers.

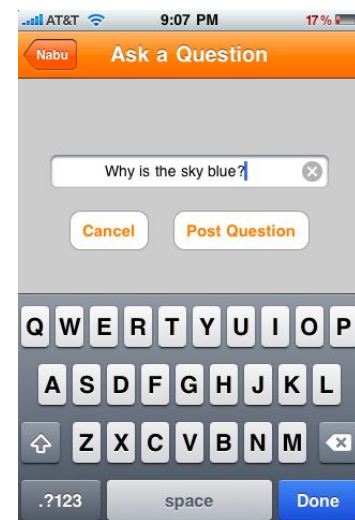


Figure 3: Asking a question.

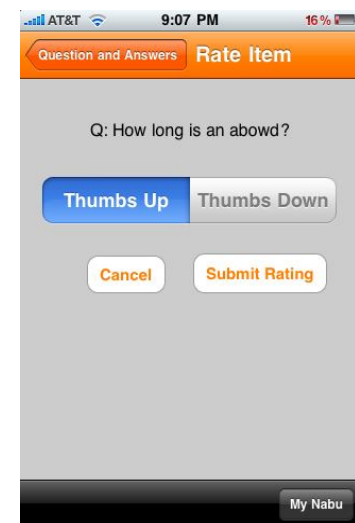


Figure 4: Rating a question.

Database

Another part of the project involved inserting and retrieving data from a MySQL database using PHP scripts. We created a schema for a MySQL database with the following tables: locations, questions, q_ratings, answers and a_ratings. We wrote PHP scripts to handle HTTP GET requests to ask a question, answer a question, rate a question and rate an answer. We wrote a separate PHP script to generate a view for the iPhone, rendered as a property list (.plist) on an XML page. This view script calculates the distance between the user's current location and the locations of previously submitted questions. It then orders questions by ascending distance, so questions closest to the user's current location are shown first. In addition, the view script computes ratings for both questions and answers. It also parses question and answer timestamps as readable dates and times.

Integration

We integrated the iPhone development with the MySQL database and related PHP scripts. We achieved this by coding the back-end for the iPhone application, so that the data obtained and displayed was accurate.

Design

For the design of the overall theme of the application, we looked into ancient symbols of knowledge and wisdom. We found Nabu, the Babylonian god of wisdom, to fit our application perfectly. We wanted the theme to be modern, yet have a rustic feel. However, we had to design within the limits of time and what the code and programs would allow. We chose to use orange and gray because orange is lively and unique, and gray balances this out with sleekness. We also thought that these colors fit the feel of our program well. We created an icon to represent our application on the iPhone menu screen (Figure 5).



Figure 5: iPhone menu icon showing Nabu in question

DISCUSSION

Location

We initially started to study WhereAmI, a location service at Georgia Tech, so that our application could be used indoors and with great accuracy on campus. While WhereAmI can determine coordinates outside of campus, the problem is that it will approximate these using the user's IP address. GPS is more accurate overall during general use, while WhereAmI is better for on campus users. We started with GPS and later incorporated WhereAmI functionality, though currently only one technology can work at a time. In a future version, the program would be able to decide which provided better accuracy at a given location.

Development and User Experience

Since we had no prior iPhone development experience, we learned how to use Xcode and Interface Builder (IB) from scratch. Though the IB is useful because it is graphical, coding is more powerful and often necessary to achieve the desired functionality. We learned how to use Objective-C, the programming language for the iPhone that is related to C. We also learned some key points of user experience design for mobile applications. The interface must be intuitive and easy to use. There is much less screen space than there is for other common devices such as personal computers. One also must account for the fact that fingers, instead of mice and cursors, are used as pointing devices. Buttons must be large enough so that users can accurately and efficiently press them.

Architecture

We learned how critical it is to understand architecture requirements and flow of information at the outset of a new project. It may have been useful to ask for pointers from experienced iPhone developers on these topics, rather than trying to navigate them ourselves. For instance, we initially thought we were restricted to using SQLite, since this database is included in the iPhone. However, we are hoping to use MySQL because of previous exposure and because it offers extensive functionality that a lightweight database cannot provide. However while wrestling with SQLite's limited features and syntactical quirks, we ran across a tutorial that used a remote MySQL database to communicate with an iPhone. Once we grasped the basics of the tutorial, development became much smoother since we could use MySQL after all.

Coding

We also learned about which functionality could be implemented in MySQL, versus which required a higher level language, in this case, PHP. After experimenting with both SQL and PHP, it was easier to determine which purposes each language served. For example, we first intended to use SQL itself to compute distance. We later realized PHP was better suited to the task.

As for the development process, we found it is much faster to write and debug code on a local environment than a commercial webhost's environment. The chosen webhost (000webhost.com) had outfitted their site with ways to directly modify the PHP code or the database. We did frequently access the database using the PHPMyAdmin instance the webhost offered. However it was more efficient to modify PHP scripts on a local machine and upload them to the webhost through FTP afterward.

If starting a similar project from scratch, we would probably make all PHP and database changes on our local machine, and run our own PHPMyAdmin process. We would only post to the webhost after making substantial progress. Besides improving efficiency, this model of

organization would allow for clear separation between development and production environments.

Integration

In terms of linking the database with the iPhone, we learned that it was crucial for the PHP-generated XML .plist to be constructed precisely so the iPhone could interpret it. Even errors that may have appeared minor from the XML perspective prevented the iPhone application from loading all together.

Design

Creativity keeps design for applications relatively simple. We learned that colors and images can be inserted both with written code and through the Interface Builder, though sometimes not as simply as one would imagine. Anything outrageous in design may not accurately show up in the program due to the code having set limitations. The iPhone automatically adds some effects to certain elements to increase ease of use and to ensure that applications do not stray too far from the standard iPhone look and feel.

FUTURE WORK

All the basic necessary functionality is present in the program but there are several features we could add to enhance usability, robustness, and usefulness. We could give users the ability to view their own questions/answers and to set their current locations if their GPS connections were faulty. This could be done through a map interface coupled with city, street, and zip code entry boxes to narrow down the view.

We could also prevent them from rating their own posts or rating items multiple times. Keeping track of users would additionally enable Nabu to flag those whose answers were consistently rated low. Another useful addition would be a more advanced rating algorithm. Old ratings could be discarded to ensure temporal relevancy. Questions and answers could be sorted by an advanced method combining rating, time posted, and distance from user. We could also implement an option to allow users to be alerted of newly submitted questions or answers in their location.

The user interface could use some improvement as well. We would like to add automatic refresh to most pages, and provide more feedback for actions. Added customization and user settings are additional possibilities. We would also like to improve the aesthetic appearance of the program and stray more from iPhone defaults.

On the database end, we would like to change the architecture so that we can host the application on our own server and thus modify the timestamp. Currently timestamps are stored as Pacific Time, and we do not have database access privileges to change the time zone on the commercially hosted web host we are using. We

would also like to improve the PHP code to provide the ability to use special characters, such as apostrophes and ampersands, in URL GET requests. More efficient algorithms to order questions by distance or compute ratings would be useful since these tasks may slow down the application as the database size increases.

We want to keep working on the program and possibly deploy it for testing and evaluation eventually. Before doing that we would have to switch to a server that can handle high traffic. We would also have to ensure that our code is robust and try to predict any possible errors. It would be useful to implement Nabu on different mobile platforms, such as Android, as well.

A usability test of Nabu would have to involve a large number of users because of the community nature of the program. It could continue for a month, with testers providing feedback both throughout the process and at the end. Also, just keeping track of when, where, and what questions/answers were posted would provide an indication of the usefulness of the program. Cognitive walkthroughs would be useful as well to understand the flow of user interaction.

CONCLUSIONS

The concept of using a mobile device for a location-based question and answer system offers convenient access to multitudes of data about the surrounding area. Questions are able to be answered by multiple people who may have different knowledge of the area. Newly asked questions in the nearby area could be answered right away to aid the inquirer and keep the location continuously updated with helpful information. Nabu may be the easiest way to ask for help and help someone without actually verbally communicating!

REFERENCES

1. <http://brightkite.com/objects>. Accessed 23 October 2009.
2. Dybwad, Barb. "Twitter's Location Aware Platform Going Live "Any Day Now."" Mashable: The Social Media Guide. 22 Sep 2009. (<http://mashable.com/2009/09/22/twitter-local-api/>). 23 Oct 2009.
3. <http://www.geograffiti.com>. Accessed 23 October 2009.
4. <http://graffit.io>. Accessed 21 October 2009.
5. <http://www.loopt.com/>. Accessed 2 December 2009.
6. <http://www.twibble.de/twibble-mobile>. Accessed 22 October 2009.
7. <http://www.vimeo.com/6139889>. Accessed 22 October 2009