

# Speech-Enabled Web Services for Mobile Devices

M. Hu, Z. Davis, S. Prasad, M. Schuricht,  
P. M. Melliar-Smith and L. E. Moser

Department of Electrical and Computer Engineering  
University of California, Santa Barbara, CA 93106

mhu@umail.ucsb.edu, zachdavis@umail.ucsb.edu, shreyasp@umail.ucsb.edu,  
mschuri@umail.ucsb.edu, pmms@ece.ucsb.edu, moser@ece.ucsb.edu

**Abstract.** *Mobile devices offer small keyboards and displays, which are poor interfaces for the users of those devices. We have developed prototype software for a mobile device that provides a better form of interaction between the user and the mobile device. To enhance the user experience, we exploit the convenience of speech input and output, using speech recognition and synthesis technology. In addition, we incorporate context-aware capabilities, using Global Positioning System technology to provide geographical coordinates. For the applications, we exploit the powers of the Web to provide Web Services, such as stocks, maps, weather forecasts, and other services. The synergy between speech input and output, context-aware capabilities, and Web Services truly empowers a mobile device, and creates a more friendly experience for the user.*

**Keywords.** Web Services, speech recognition and synthesis, context-aware capabilities, mobile devices, user interfaces.

## 1. Introduction

The increasing popularity of, and technological advancements in, mobile phones and PDAs, primarily mobile phones, is leading to the development of applications to fulfill growing consumer needs. The Simple Message Service (SMS) [7, 10] is available on most mobile phones today, and some mobile phones even provide support for the Multimedia Messaging Service (MMS) [10] to exchange photos and videos. The mobile phone manufacturers are no longer focused on making a “mobile phone” but, rather, on producing a mobile device that combines phone capabilities with the power of handheld PCs.

Many mobile phone manufacturers realize that the numeric keypad, not to mention the small screen size, common to phones of the past, does not carry over well to a handheld PC [8]. In the current state-of-the-art, mobile phone hybrids are more of an inconvenience than a benefit to the user.

We aim to bridge this gap by providing a better form of interaction between the user and the mobile device. In particular, we aim to enhance the user experience by:

- Exploiting the convenience of speech input and output, using speech recognition and synthesis technology
- Incorporating context-aware capabilities, using Global Positioning System (GPS) technology
- Exploiting the powers of the Web to provide Web Services for the user, such as stocks, maps, weather forecasts, and other services.

The combination of these three technologies truly empowers a mobile device, and creates a more friendly experience for the user.

Combining speech recognition and synthesis with the Web is not a new idea. In fact, much progress has been made in creating the multi-modal Web [5], which allows not only keyboard and mouse navigation but also speech input and output. To show the potential of multi-modal user interaction, IBM has produced a software demonstration, which can be found at <http://www-306.ibm.com/software/pervasive/multimodal/>. To run the demonstration and see what it can do, you need a multimodal browser such as Opera [14]. As [4] points out, the current use of speech recognition and synthesis technology reflects only the tip of the iceberg of what that technology has to offer.

With the advent of Web Services [18], the Web can now provide services, whereas in the past it provided only data. Web Services are at the heart of the applications that we have developed. Those applications provide stock quotes, location information for restaurants, movies, etc, and weather forecasts. As more Web Services are developed, many of them can be utilized in much the same way on a mobile device.

The speech-enabled and context-aware capabilities that we have bestowed on these Web Services are what makes them unique and appropriate for mobile devices. Although context-aware applications have existed for some time, the first serious commercial applications

have been released only recently [16]. We intend to follow this direction and provide speech-enabled context-aware Web Services bundled together for mobile devices.

## 2. Background

Being able to respond to, and recognize, casual speech is our ultimate goal. A user need not form his/her request in a particular rigid format in order for the applications to understand what the user means.

For speech recognition, we chose to use a small-footprint, high-accuracy, speaker-independent speech recognition engine, called DynaSpeak, supplied by SRI, Inc [17]. DynaSpeak is based on a statistical language model, which is suitable for natural language dialog applications. It includes speaker adaptation to increase recognition accuracy in order that individuals with different accents or tone pitches do not suffer from loss of performance. The DynaSpeak engine can also be configured so that it performs speech recognition specific to a particular individual.

Speech recognition must be complemented with speech synthesis or text-to-speech capabilities so that voice input is complemented with voice output. The applications, that we have developed, have the ability to respond in voice when it is appropriate to do so.

For speech synthesis, we chose to use the Natural Voices software from AT&T [2]. Natural Voices generates speech that sounds natural, rather than electronic, so the user does not feel like he/she is talking to a computer. It supports many languages, male and female voices, and several difference interface standards, including VoiceXML, which we used in our prototype. We created text-to-speech software that runs in the background and accepts messages in that format. Each message contains the name of the voice engine ('Mike' for a male voice and 'Crystal' for a female voice) and the corresponding text to speak.

For our prototype, we needed a platform on which we could not only develop the applications but also illustrate the concept that the applications were to be used on a mobile device. Thus, we chose the OQO personal computer [15], which is shown in Figure 1. The OQO is a full-fledged computer that resides in a 5" x 3" compact handheld unit. It is based on the 1 GHz Transmeta Crusoe processor, which emulates the Intel instruction sets, and it runs the Windows XP Professional operating system. It has an 800 x 480 touch screen LCD display, with a full-on QWERTY thumb keyboard and mouse, as well as WiFi and Bluetooth wireless interfaces, and a built-in microphone.

With Windows XP, it is possible to develop many kinds of applications in various programming languages, the choice of which depends on the specific application.



Figure 1. The OQO handheld computer.

The built-in microphone is essential for our voice-enabled applications. The widescreen display, which is much larger than conventional handheld PCs, allows us to provide a multi-modal experience for the user with nice graphics, as well as speech input and output.

## 3. Managing Stocks

With the advent of stock trading on the Web, anyone can trade stocks with the click of a button. It is important for an individual to know the behavior of a stock at any given time to make the correct decision to either buy or sell. Thus, many investor Web sites now provide stock quotes for their visitors to view. Although many of these quotes are delayed, they still provide a fairly good indication of the behavior of the stock. However, not everyone has access to a desktop computer and the Internet at all times. Our Stock Portfolio Service fills this need by providing an individual with the ability to gather quotes that are already available on the Web and bring them to the individual's mobile device.

The Stock Portfolio Service is built with the intent that an individual should be able to manage his/her investment portfolio, anywhere and anytime. Mobile devices, in particular cell phones, are an established technology that can enable an individual to do this. The Stock Portfolio Service could even be used as a personal investment assistant, allowing actual exchange of stock to take place via Web Services. Of course, such usage requires resolution of certain security risks.

There are many stock quote services available on the Web, and the choice of one of them is a matter of personal preference. With Web Services, little or no effort is required to adapt the code from one to another. However, our Stock Portfolio Service requires a more detailed stock quote than what most current services offer. Thus, we have developed a custom solution.

Yahoo! Finance [20] provides a Comma Separated Values (CSV) file that contains extended quote information via a simple HTTP GET query. It then requires only simple parsing of that file to obtain quote information. A simple parser can itself be provided as a Web Service, and the Stock Portfolio Service can then communicate with it. Many stock quote services are already implemented in this way. An example of a query that looks up the Microsoft (MSFT) quote is:

```
http://finance.yahoo.com/d/quotes.csv?s=MSFT&f=s11d1t1c1ohgv&e
```

The Stock Portfolio Service that we developed provides an overview of the performance of all of the stocks in the portfolio that are currently monitored, as well as the current performance of the stock exchanges. A more detailed quote can be obtained via voice request or by navigating the graphical user interface. Stocks can readily be added or removed, and can be supported by voice. If stocks are ‘purchased’, the amount of gain or loss from them is also displayed. Here, we use the term ‘purchased’ to mean that a record is created and stored in a local database with a certain buy price. The user can use this database to track the stocks later.

The Stock Portfolio Service is best illustrated with an example. Consider a businessman who wishes to monitor some stocks. He starts the program and is confronted with the indicators of the various stock exchanges. He has an empty portfolio, so he proceeds to add a few stocks to his portfolio. He can either state “Add a stock to my portfolio” or use the graphical user interface to do so. A screen appears to allow him to either search for a particular index or input the stock symbol directly. He types in “MSFT” to add the Microsoft stock to his portfolio, and verifies that this addition is correct. The stock is added to the portfolio, and the businessman then makes a request for a quote, and the device sends back information, via voice, to the businessman, as shown in Figure 2.

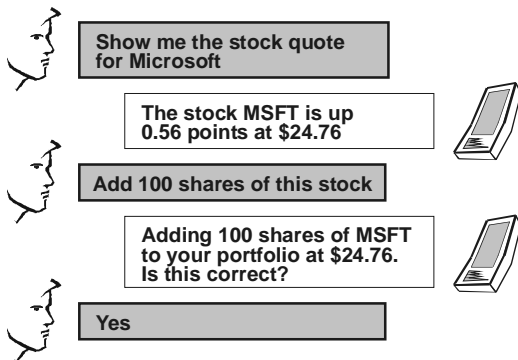


Figure 2. Dialogue with the Stock Portfolio Service.

When the businessman requests that the stock be added, a local database entry is made for that stock. The database allows previous quote information to be cached, so that a personal quote history can be maintained. This caching also solves a performance issue. Because quotes from the Web are not updated every second, there is no need to grab quotes repeatedly when they have not been updated and, thus, waste network bandwidth. The timestamp of the previous quote is checked against the time of the request. If the difference between the times is greater than a certain value, a new quote is fetched; otherwise, the quote information is loaded from the cache.

Next, as shown in Figure 2, the businessman requests that 100 shares of Microsoft stock be added to his portfolio. The system responds that it has done so and asks for confirmation. The database entry for MSFT is updated to reflect the changes made. When he has finished updating his portfolio, the businessman can check on the performance of his stocks wherever he goes. He needs only to say to his handheld device something like, “How is the stock MSFT doing?” To see how much he made on his investment, he notes that the profit displayed for MSFT is \$284.00, calculated from the current market price and the purchase price.

A flowchart for the Stock Portfolio Service, which is implemented in C#, is shown in Figure 3.

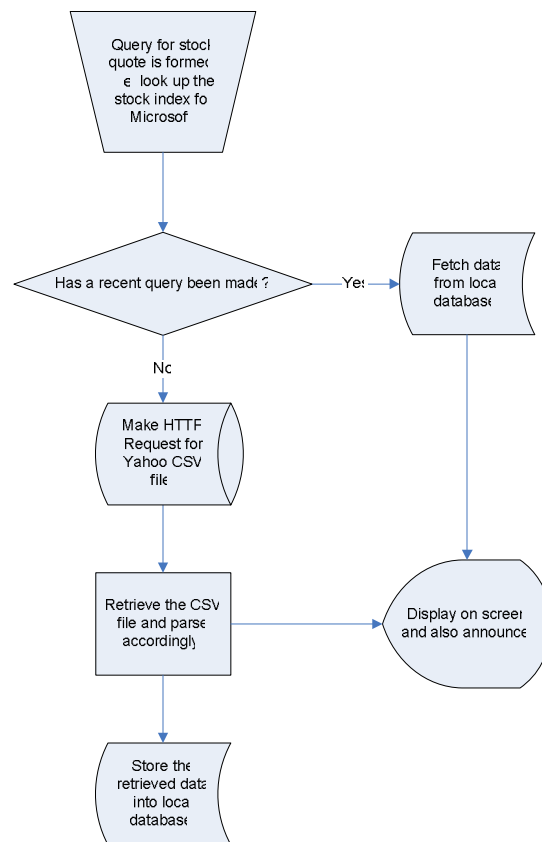


Figure 3. Flow Chart for the Stock Portfolio Service.

## 4. Better than GPS

Within recent years, GPS has become a widely used technology. Many cars, and also mobile handheld devices, are now equipped with GPS, and there is a plethora of standalone GPS units available. Many GPS units come with software that allows a user to search for places of interest, based on his/her current location or intended destination.

Our objective is to exploit the GPS unit, already present in many mobile handheld devices, to offer a Location Service that provides maps and directions for restaurants, movies, etc, so that someone not in a car can also reap the benefits. Of course, we intend to exploit speech recognition for voice input and also speech synthesis for voice feedback. Such capabilities allow users to request information in a more user friendly manner than by pushing buttons, especially on mobile phones where inputting letters is a particular challenge.

In this application, we also use Web Services, in particular, the Yahoo! Local API [17]. The Web Services provided by Yahoo! are Representational State Transfer (REST) services for distributed network systems [4]. REST allows request URLs to be composed either in a browser or in code to achieve a certain kind of response. In the Location Service, a request query for restaurants near a certain area returns a list of results in XML, corresponding to the restaurants that were found. An example of a query for pizza places around zip code 93106 looks like:

```
http://api.local.yahoo.com/LocalSearch
Service/V1/localSearch?appid=Demo&query
=pizza&location=93106&results=1
```

The result then yields XML similar to that shown in Figure 4.

```
<ResultSet>
...
<Result>
  <Title>Woodstocks Pizza Parlor
  </Title>
  <Address>928 Embarcadero Del Norte
  </Address>
  <City>Santa Barbara</City>
  <State>CA</State>
  <Phone>(805) 968-6969</Phone>
  <Rating>5</Rating>
  <Distance>0.56</Distance>
</Result>
...
</ResultSet>
```

Figure 4. Result of a Location Service query in XML.

The Local Service client application form queries URLs, fetches the resulting XML, and then parses it. The application also returns a hyperlink to the restaurant, if one is available, as well as a URL to Yahoo! that indicates the location of the restaurant. The application fetches the JPEG file for the map, and displays it along with the search results. In this way, all of the associated information is located on the map. The various results are linked together, so that the user can obtain additional details related to the restaurant, by clicking a button or using voice navigation.

Much of what we've accomplished in our prototype is achievable with standard GPS devices. However, as stated previously, those devices require additional software to provide information on restaurants, movies, and such. That software can become outdated, and requires frequent updates. By using Web Services, this problem on the client's mobile handheld device no longer exists, not to mention the tremendous amount of memory space that is saved on the device.

What makes the Yahoo! API the preferred choice is its ability to locate positions on the map on the basis of longitude and latitude, which makes it an ideal choice to use with GPS. Thus, the user is no longer limited to requests involving a city or zip code. The user now has the ability to create requests that are truly location-aware, such as the dialogue in Figure 5 shows.

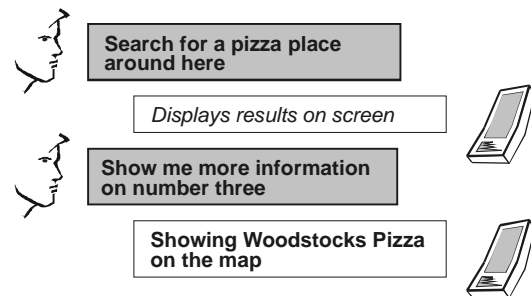


Figure 5. Dialogue with the Location Service.

The word 'here' in the above dialogue is recognized by the application and replaced with the GPS coordinates, and formed into a query to Yahoo!, which then returns a map of the user's current location, indicating where the user is, along with the 10 nearest pizza places.

Actually, we have taken this application a step further, making it not only location-aware but also context-aware. The application works in conjunction with another one of our speech-enabled applications, a Contact Service that supplies contact information. Thus, the user can make more complex requests, such as "Search for a pizza place around Susan's house." The application uses the Contact Service to replace 'Susan's house' with a specific address, so that the actual search query carried out looks something like this: "Search for

a pizza place around 232 Kings Way, Goleta, CA, 93117". The Location Service then searches for a pizza place in the vicinity of that address.

A flow chart for the Location Service is shown in Figure 6. Because the Yahoo! API is given in Java, we implemented the Location Service in Java.

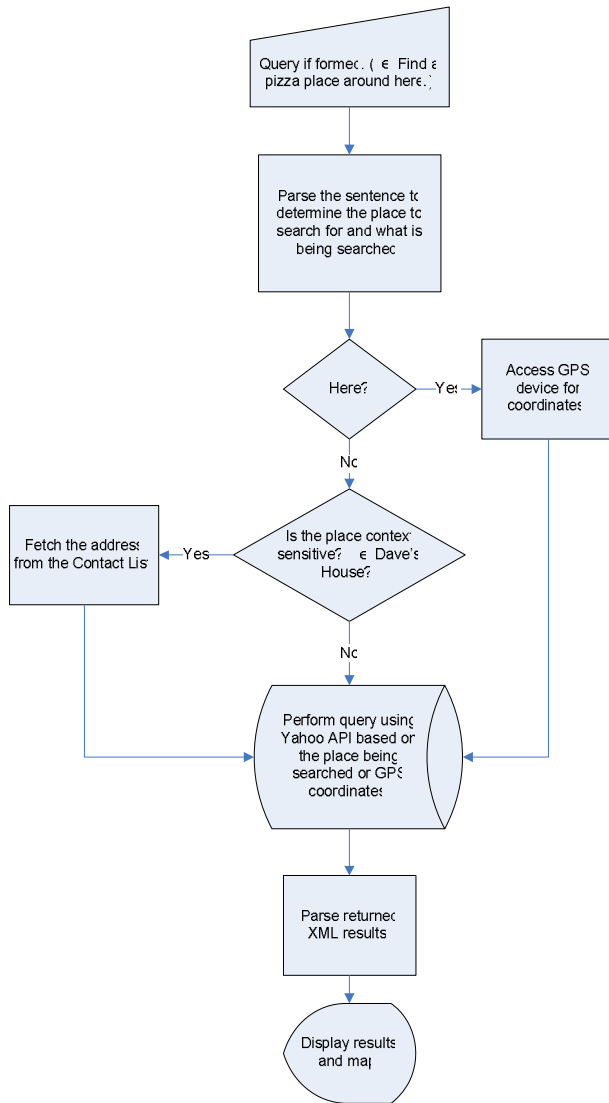


Figure 6. Flow Chart for the Location Service.

As mentioned previously, the Location Service can locate not only restaurants, but also movie theatres, banks, and other points of interest. Moreover, in addition to speech output, it can display maps and lists, which are more appropriate for some kinds of information. For example, in response to the request, "Search for a movie theatre around here," the Location Service displays the map shown in Figure 7, when the user is in downtown Santa Barbara.

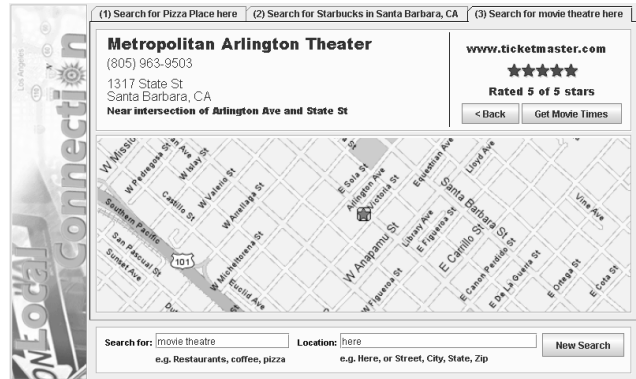


Figure 7. Map displayed by the Location Service.

After the user has obtained the map for a specific movie theatre, the user can issue the command "Get movie times" to obtain a list of movies and their times at the particular theatre. If the movie times are available for the selected theatre, they are also displayed as shown in Figure 8.

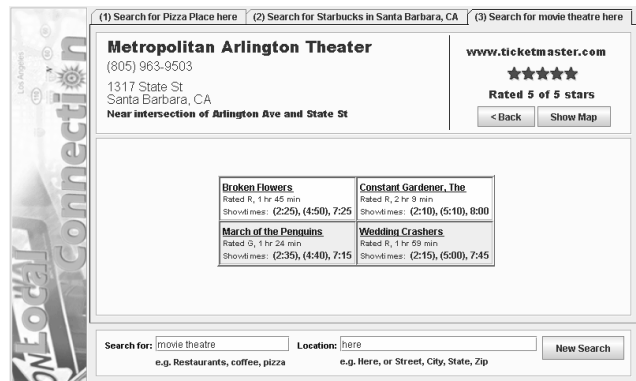


Figure 8. List displayed by the Location Service.

From this screen, the user can switch back to the map showing the location of the theatre, return to the theatre results, or perform a new search (which can be done at any point).

For information like maps and lists, it is desirable to use a graphical or textual display, as well as speech output, in a multi-modal user interface. Thus, the most appropriate kind of output can be chosen, depending on the kind of information, the capabilities of the mobile device, and the context in which the user finds himself or herself.

## 5. Rain or Shine

Yet another popular Web Service that we exploited is the National Weather Service offered by the National Oceanographic and Atmospheric Administration [13]. The National Weather Service provides not only the

current weather and weekly forecasts, but also a map showing cloud formations and warm and cold fronts. This service is very useful for planning and scheduling events. Having this application on your mobile device is like having your own weatherman at your side, and adds a level of convenience to everyday life.

Our goal, of course, is not simply to provide a speech-enabled weather forecast service but to incorporate location-sensitive capabilities similar to those provided by the Location Service. Thus, for example, using the Contact Service, a user can look up the weather around his/her house or work. The Weather Forecast Service is not just location-aware but context-aware because it depends on both location and time.

Use of the Weather Forecast Service is easy and straightforward, especially because it is speech-enabled. Consider, for example, the dialogue shown in Figure 9 where the user asks:

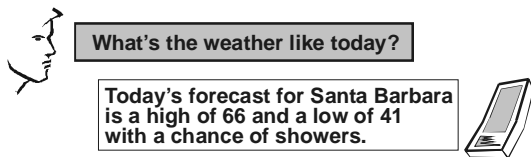


Figure 9. Dialogue with the Weather Forecast Service.

The program responds with the high and low predicted temperatures and an indication that there is a change of showers in Santa Barbara. A user can also ask, “What’s the weather like here two days from now?” The program then determines the corresponding date and states the forecasted temperature to the user. The Weather Forecast Service provides a quick and efficient way for the user to obtain weather information, without having to flip through channels to find The Weather Channel or to log onto the Internet and search [www.weather.com](http://www.weather.com) [13].

A flowchart for the Weather Forecast Service is shown in Figure 10. The Weather Forecast Service is implemented in C#. With the hooks provided by Microsoft’s .NET platform [12] and the use of C# in Visual Studio, the Web Service function calls for the National Weather Service are easy to set up.

## 6. Related Work

The application that is closest to the applications that we have implemented is Google’s Simple Message Service (SMS) [7]. That service allows users to send SMS queries to Google, such as “pizza 93106”, “weather Santa Barbara CA”, and “stock MSFT”. Those requests fetch the pizza place nearest 93106, check the weather in Santa Barbara CA, and look up the

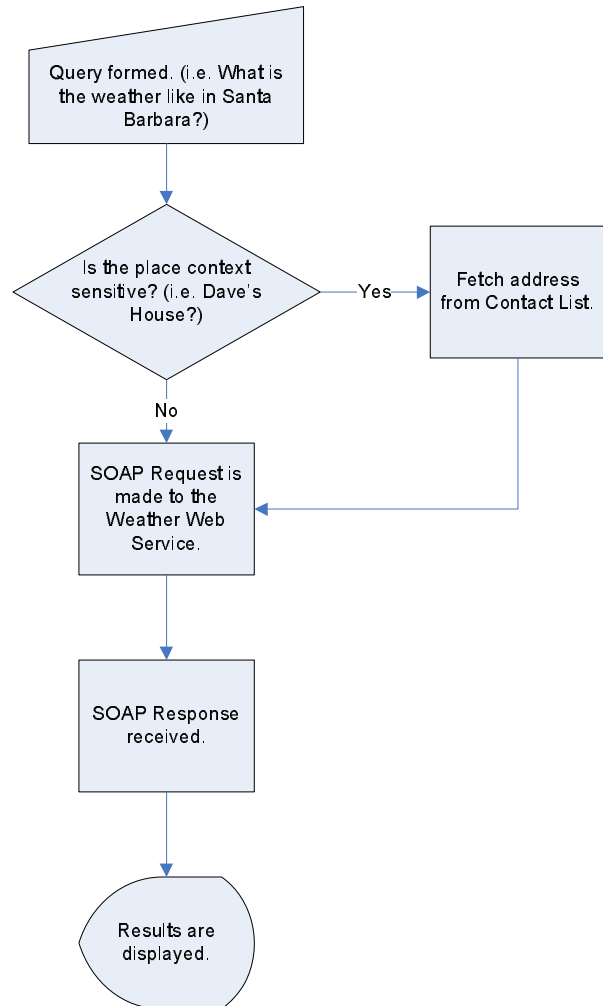


Figure 10. Flow Chart for the Weather Forecast Service.

stock quote for MSFT, respectively. However, they do not have voice input and output capabilities, based on speech recognition and synthesis technology like our applications do.

Of the various Web Services available today, the map applications seem to be the most popular, with online map services available from Yahoo!, Google and Microsoft. There also exist several other projects underway with specific thrusts and variations. The Cyberguide project [11] focuses on providing position-aware tour guides for tourists. The CoolTown project [9] allows a user to interact with Web resources corresponding to physical places using his/her mobile phone. Smart Sight [21] provides navigation assistance to tourists while also attempting to remove language barriers. The feature that sets our map-related services apart from the others is its ability to make queries for places using speech input and output.

As for the other applications, similar services exist on the Web. Free stock quotes are provided by Yahoo!

Finance [20], and real-time quotes are available from online stock trading companies such as Ameritrade [1]. To check the weather, one simply needs to click on [www.weather.com](http://www.weather.com) [13] to obtain detailed forecasts for the week. To obtain location-based information, one can go to Yahoo! Local [19] or to Google Maps [6]. We have made such services more accessible by using a mobile device and have added speech recognition and speech synthesis capabilities to them.

## 7. Conclusions and Future Work

We have developed a variety of speech-enabled, context-aware applications for mobile handheld devices, and have exploited Web Services in those applications. The applications include a Stock Portfolio Service, a Location Service, and a Weather Forecast Service. Our work shows that a user no longer has to put up with the annoyances of a 3-inch keyboard, nested menus, or handwriting recognition, nor does the user need to have a powerful desktop or mainframe computer in order to benefit from such services. By providing speech-enabled context-aware applications on mobile devices, we have attempted to bring a new level of convenience to the users of mobile devices, while exploiting the power of Web Services.

Use of speech input and output for applications that involve multiple different Web Services remains a challenging problem. Also, dynamically switching between alternative Web Services, for availability or performance reasons, is a challenge because the interfaces for the alternative services usually differ. Moreover, currently, relatively few Web Services are available on the Web. We hope that our work will encourage additional Web Services to be made available and to be geared towards mobile devices.

## 8. Acknowledgments

This research was supported in part by AFOSR/MURI grant number F49620-00-1-0330 with support from SRI, Inc., and T-Mobile, Inc.

## 9. References

- [1] Ameritrade, 2006, <http://www.ameritrade.com>.
- [2] AT&T Labs, "Natural Voices," 2006, [http://www.naturalvoices.att.com/products/tts\\_data.htm](http://www.naturalvoices.att.com/products/tts_data.htm).
- [3] Datamonitor, "Voice automation—Past, present, and future," White Paper, July 2003.
- [4] R. T. Fielding and R. N. Taylor, "Principled design of the modern Web architecture," *ACM Transactions on Internet Technology*, vol. 2, no. 2, May 2002, pp. 115-150.
- [5] A. R. Frost, "Call for a public-domain speechweb," *Communications of the ACM*, vol. 48, no. 11, November 2005, pp. 45-49.
- [6] Google, "Goggle Maps API," 2006, <http://www.google.com/apis/maps>.
- [7] Google, "SMS," 2006, <http://www.google.com/sms/>.
- [8] K. Holtzblatt, "Designing for the mobile device: Experiences, challenges, and methods," *Communications of the ACM*, vol. 48, no. 7, July 2005, pp. 33-35.
- [9] T. Kindberg and J. Barton, "A Web-based nomadic computing system," *Computer Networks*, Amsterdam, The Netherlands, 1999.
- [10] G. Le Bodic, "Mobile Messaging, SMS, EMS and MMS," November 2002, John Wiley & Sons.
- [11] S. Long, D. Aust, D., Abowd, and C. Atkinson, Cyberguide: "Rapid prototyping of mobile context-aware applications: The Cyberguide case study," *Proceedings of the International Conference on Mobile Computing and Networking*, Rye, NY, 1996, pp. 97-107.
- [12] Microsoft, ".NET," 2006, <http://www.microsoft.com/net/default.aspx>.
- [13] National Oceanic and Atmospheric Administration, "National Weather Service," 2006, <http://www.nws.noaa.gov>.
- [14] Opera, "Opera software," 2006, <http://www.opera.com>.
- [15] OQO, "OQO personal computer," 2006, <http://www.oqo.com>.
- [16] S. Pokraev, J. Koolwaaij, M. van Setten, T. Broens, P. Dockhorn Costa, M. Wibbels, P. Ebben, and P. Strating, "Service platform for rapid development and deployment of context-aware mobile applications," *Proceedings of the IEEE International Conference on Web Services*, Orlando, FL, July 2005, pp. 639-646.
- [17] SRI, Inc., "DynaSpeak," 2006, <http://www.speechsri.com/products/dynaspeak.shtml>.
- [18] N. Wickramage and S. Weerawarana, "A benchmark for Web Service frameworks," *Proceedings of the IEEE International Conference on Services Computing*, Orlando, FL, July 2005, pp. 233-240.
- [19] Yahoo! Developer Network, "LocalSearch Documentation," 2006, <http://developer.yahoo.net/search/local/V1/local-Search.html>.
- [20] Yahoo!, "Finance," 2006, <http://finance.yahoo.com>.
- [21] J. Yang, W. Yang, M. Denecke, and A. Waibel, "Smart Sight: A tourist assistant system," *Proceedings of the IEEE International Symposium on Wearable Computers*, Victory, Canada, 1999, pp. 73-78.