Developing an Interactive Kiosk for Mobile Devices: A Practical Experience

Stan Kurkovsky, Bhagyavati, Manish Shah
Department of Computer Science
Columbus State University
{kurkovsky_stan, bhagyavati, shah_manish}@colstate.edu

Keywords: Multimedia, kiosk, mobile computing, XML, Java.

Abstract

This paper discusses the need for extending the functionality of interactive multimedia kiosks to wireless mobile devices. We present Kiosk Engine, our implementation of a framework for creating and presenting interactive multimedia kiosks. Kiosk Engine is implemented in Java and it can be used to present the kiosk content on both a stationary kiosk and a mobile wireless device. We conclude the paper by demonstrating Kiosk Engine's capabilities using a Nokia simulator.

1. Introduction

Interactive kiosks have become an every day reality. They are most commonly used to present up-to-date and/or dynamically changing information in a visually appealing format [7]. Historically, kiosks have been created as stationary computers placed in locations convenient to the customers of many organizations [8], such as banks, libraries, museums and reception areas of larger companies. However, with the rapid growth of the numbers and popularity of mobile devices, their users also demand access to the most current information. Businesses that currently use kiosks can cater to the mobile users by providing real-time information that changes dynamically and is displayed on the mobile user's handheld wireless device.

Because information presented in a kiosk should be updated in real-time, it is pulled by the user's mobile device and not pushed by the server. In today's world of busy travelers and privacy-minded individuals, pull technologies make more business sense than push technologies. Moreover, kiosks extended to wireless devices that can provide secure access to information have the potential to increase the visibility and market share of a business by providing new avenues for the clients to access business information. Due to the pull model of transmission, the customer feels in control of when and how information is shown on the wireless mobile devices.

Kiosk customizability ensures its appeal to businesses that are unable to spend resources such as time and labor on adapting a generic kiosk to their needs. Since kiosks display multimedia data, kiosk developers can develop various modes of delivering information depending on the target device. For example, a mobile user who downloads a kiosk application onto a handheld device may not appreciate the increased capacity required for viewing video streams. On the other hand, it should not pose any problems for viewing such multimedia content on a standalone kiosk. If the proposed model is followed, businesses need to develop the content of their applications just once; they can then present it in various ways, depending on where the application will be executed.

In this paper, we present our solution for developing and presenting kiosks that are rich in multimedia data. Our solution called Kiosk Engine (KE) allows kiosk data to be viewed on a standalone computer or a mobile device, such as any Internet-enabled PDA or smart phone. The rest of the paper is organized as follows. Section 2 makes the case for building or porting a kiosk application for mobile wireless devices and presents a range of possible kiosk-enabled applications specific to the needs of mobile users. Section 3 discusses the implementation of the core of the Kiosk Engine. Section 4 follows by describing the

architecture and implementation of the KE extension for mobile devices. Section 5 concludes the paper with a summary.

2. Mobile kiosks and their applications

Traditionally, kiosks were created as interactive applications and this made them very attractive to both businesses and end users. Businesses could expect reaction and/or feedback from users in real time and could modify the content of the kiosk accordingly. Users had the satisfaction of getting interactive information that updates dynamically. For example, a static map in a museum provides the users with an idea of where they are, and presents information about the exhibits for possible visitation [4]. However, if an exhibit is temporarily unavailable, the static map is outdated and not of much use. The same museum map can be incorporated into a kiosk application serving as a museum tour guide, which can be used for both a stationary kiosk and also be downloaded by the users into their handheld devices. In this scenario, real-time updates ensure that the information about the temporary unavailability of an exhibit is presented to the user in a timely fashion [12].

Kiosk applications can be especially appealing if applied to mobile devices. Features of such applications include user interactivity, ease of use and access, and high quality of communication [11]. A kiosk implemented as a lightweight client application should be easily adaptable to run on a device with limited processing power to present information in a visually appealing manner to mobile users. The interactive capabilities mean that people on the go do not merely access the content pushed to them, but can also interact and respond with real-time feedback to the provider of the content.

Mobile devices capable of running kiosk applications must be able to present information graphically and in an easy-to-use format. In particular, because the proposed Kiosk Engine occupies a very small amount of memory and does not require a lot of resources, it can run on most modern PDAs and smart phones that have Java support. Memory requirements for large data objects can be mitigated by converting them into other data formats that are supported by the target PDA. We discuss our approach to this issue is Section 4. Also, businesses can minimize manual intervention and kiosk maintenance efforts by providing updated content to users on their wireless devices. For example, in a museum, the number of human tour guides can be minimized by providing the information as a kiosk application on the users' mobile devices.

Kiosks used to be limited to retail check-out counters, but now they are diversifying into mainstream areas such as banking and tourism [3]. Applications for kiosks abound in the areas of locator services, banking, gaming, Internet access, self-checkout terminals, information access and tourism. As many research reports indicate, there is a growing need for multimodal access to the information presented by kiosks [1,5,6,9]. Reverting to our museum example earlier, this is an illustration of how a kiosk application is used in the tourism industry. More examples follow.

On a golf course, route information can be provided to users via a kiosk client running on handheld devices. Since route information can change dynamically, a kiosk presents the perfect platform to provide up-to-the-minute information for users. Such information can be useful in preventing users from going to temporarily inaccessible areas. It can also prevent new visitors from getting lost on larger golf courses.

In national parks and animal sanctuaries, visitors can survey route information via mobile devices using a light-weight kiosk application client. Not only will this prevent visitors from losing their way, but it will also help steer more interest into little-known areas of the park. The benefits of providing interactive, real-time content to mobile users can translate to more visitors and higher revenues.

At sport events such as the Super Bowl, fans can easily and conveniently find the most up-to-date information about the current sport scores and player statistics. If such a kiosk application were executed on mobile devices, then fans and visitors could keep in touch with each other via their handheld devices, while obtaining the most current information about the sport in real-time. For example, hardcore fans would not want to miss an important event in the game just because they are at the concession counter at that particular time.

In a ticketing application for a kiosk, people can buy tickets for events via handheld devices. Since clients pull updated, real-time information about events of interest, there is the potential for higher sales volume. In addition, the interactive capabilities of the kiosk imply that busy mobile users can buy tickets after checking the most current information about it on their handhelds.

Kiosks are already being used in the retail industry, such as self-checkout counters in supermarkets. While these applications bring value to the users, businesses and their customers, they can also be used on handheld and wireless devices to provide additional value for both businesses and end users. Such wireless extensions of existing kiosk applications provide a way for businesses to increase profits while incurring incremental costs for small changes and extensions [2]. Furthermore, a kiosk in a supermarket can allow users to directly go to the aisle with the products of their interest or browse the supermarket catalog pulled by a handheld device in search for a specific product. It saves shoppers time and provides ease of use and convenient access to products.

3. Kiosk Engine implementation

Kiosk Engine (KE) is implemented as an application that can interpret the description of and present interactive multimedia kiosks differently based on the different inputs provided. Our design and implementation guidelines were as follows:

- It should be easy to create, access (from either web or stand alone), modify and store the kiosk description. Consequently, upgrades and changes to a kiosk should be easy.
- KE should be lightweight and be able to run on the majority of the existing software and hardware platforms.
- KE-driven kiosks should support five basic types of multimedia data: text, images, documents (containing plain/rich text or HTML), audio and video.
- The data (i.e. text, images, audio, video, etc.) of a kiosk should be isolated from the processing program so that data independence is maintained, which allows for portability.
- The same version of KE should be able to act as a driver for multiple kiosks.

To achieve platform independence, KE is programmed in the latest version of Java 2 programming language with Java Media Framework that provides support for a wide range of audio/video formats. The resulting application is truly lightweight – compiled code occupies less than 26 Kbytes. A UML diagram showing the class structure of KE implementation in Java is presented in Figure 1.

KE can work as a driver of a kiosk when provided with input consisting of multimedia data, their description and the overall organization on the kiosk. KE accepts the input written in XMLKiosk, a specially developed XML-based language,, which describes the sources, organization, presentation order, size and style of multimedia elements. We chose an XML-based solution for two favorable reasons. First,, we are able to create new tags for each basic data type and specify custom attributes for each tag. Second,, XML can be easily parsed in Java and thus we can avoid creating a custom parser, which would add more classes to the code and processing time to the end product.

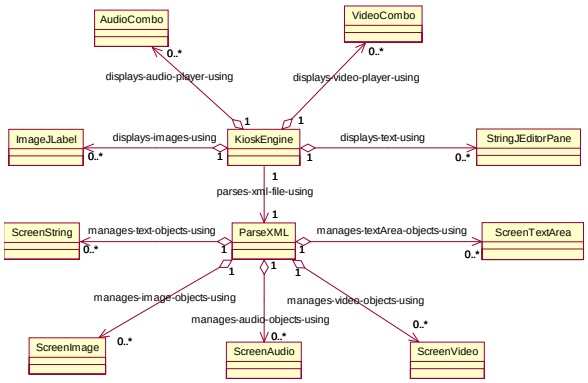

Figure 1. Class diagram of Kiosk Engine: Java implementation

Each Kiosk application must be described in XMLKiosk with a root XML node <Kiosk>, which contains all the different tags to describe the entire kiosk. Figure 2 presents a sample description of a fragment of a multimedia kiosk, as well as the resulting interactive campus map of Columbus State University (CSU). At the next depth level there are two elements called <header> and <screen>, which are the containers of different types of multimedia objects. Both <header> and <screen> elements can have all or any of the tags for the multimedia objects. To describe the five types of multimedia data, namely text, images, documents, audio and video, there are tags called <text>, <img>, <textArea>, <aud> and <vdo> respectively. For each instance of any of the data types on the kiosk screen, there is a corresponding tag describing it in the input XMLKiosk file. The <text> and <img> tags occasionally have <action> tags, which specify what tasks need to be performed in order to facilitate browsing the kiosk from one screen to another. The <action> tags are usually coded when some action occurs on the screen due to user interaction. Also, each <aud> or <vdo> tag can have multiple <file> tags, which allow a single audio/video instance to play multiple files alternatively.

```
<?xml version = "1.0" ?>
<kiosk red="0" green="102" blue="204" start="csumain">

    <header id="csuheader">
    <img src="images/csu/csu_logo.gif" alt="CSU MAP" xpos="0" ypos="0" width="72" height="68">
        <action type="onclick" alt="CSU MAP" dxpos="0" dypos="0" dwidth="72" dheight="68">
            <target>csumain</target>
            </action>
        </img>
    </header>

    <screen id="csumain" header="">
        <img src="images/csu/kioskNameplate.jpg" alt="Columbus State University Kiosk"
            xpos="0" ypos="531" width="800" height="69">
        </img>
        <img src="images/csu/map80.jpg" alt="CSU MAP" xpos="0" ypos="0" width="800" height="531">
            <action type="onclick" alt="About Columbus State University" dxpos="60" dypos="0" dwidth="181" dheight="45">
                <target>aboutcsu</target>
            </action>
            <action type="onclick" alt="Clock Tower" dxpos="344" dypos="201" dwidth="13" dheight="37">
                <target>clocktower1</target>
            </action>
            <action type="onclick" alt="Simon Schwob Library"
                dxpos="232" dypos="215" dwidth="51" dheight="25">
                <target>library1</target>
            </action>
            <action type="onclick" alt="Lenoir Hall" dxpos="107" dypos="243" dwidth="79" dheight="18">
                <target>lenoir1</target>
            </action>
            <action type="onclick" alt="Richards Hall" dxpos="356" dypos="195" dwidth="55" dheight="15">
                <target>richards1</target>
            </action>
            <action type="onclick" alt="Howard Hall" dxpos="256" dypos="202" dwidth="70" dheight="12">
                <target>howard1</target>
            </action>
            <action type="onclick" alt="Woodall Hall" dxpos="293" dypos="229" dwidth="33" dheight="18">
                <target>woodall1</target>
            </action>
            <action type="onclick" alt="Davidson Student Center" dxpos="343" dypos="249" dwidth="80" dheight="11">
                <target>davidson1</target>
            </action>
        </img>
    </screen>
    …
</kiosk>
```

Figure 2. Sample description of a multimedia kiosk written in XMLKiosk

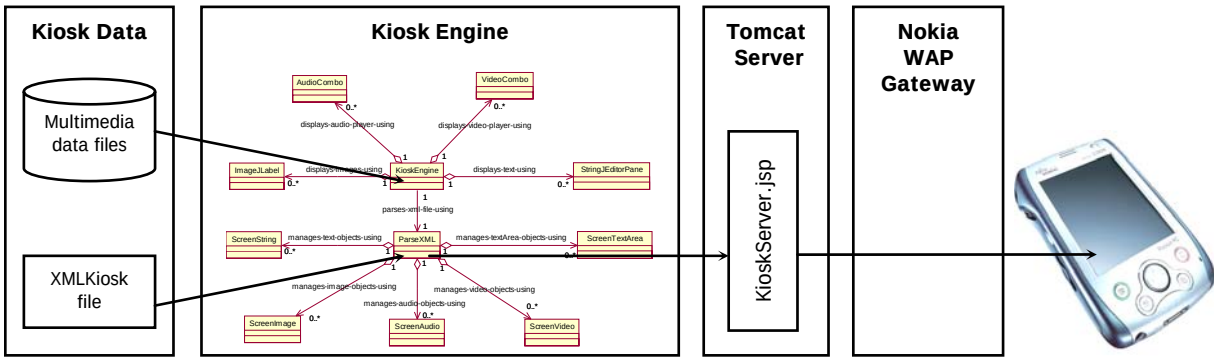

Figure 3. Architecture of the Kiosk Engine extension for wireless mobile devices

4. Extension of Kiosk Engine to mobile devices

The core implementation of Kiosk Engine has been extended to support mobile devices. We used a Wireless Application Protocol (WAP) gateway to convert XML tags to Wireless Markup Language (WML) tags (Figure 3). This gateway or proxy server also converted all image files into WBMP format and performed other conversions in order to make them suitable for a smaller screen size and constrained processing power and storage available on mobile devices.

The front end and user interface for the kiosk application is provided by WML generated by Java Server Pages (JSP) as the underlying enabling technology. WML tags are dynamically generated based on user input. Since the kiosk is an interactive application, user interface is simple and easy to use. The interface is also visually appealing and invites feedback.

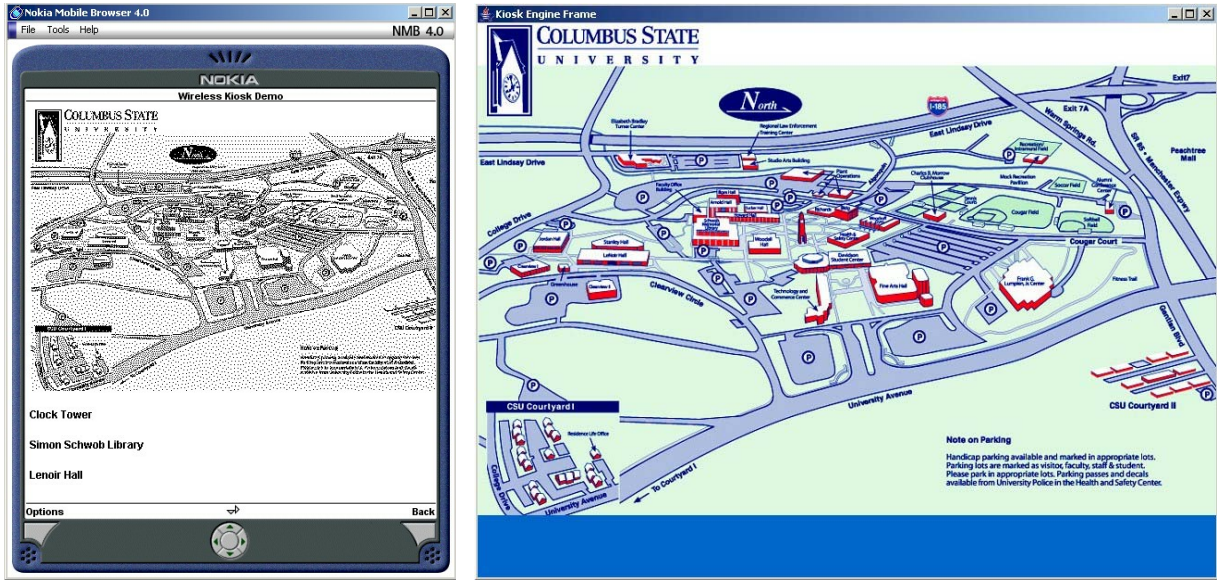

Figure 4. The same kiosk generated by Kiosk Engine as displayed on a mobile device and on a stationary computer

There are some definite tradeoffs in extending kiosk application to mobile devices. These tradeoffs include limited multimedia content and low resolution of images and graphics, as demonstrated in Figure 4. These limitations are due to the inherent drawbacks of the wireless medium and the intrinsic constraints that mobility imposes on handheld devices. For example, we used no audio or video in our pilot test due to limited processing power and restricted display size on the mobile devices. The images and maps on a handheld kiosk were grainy when compared to a desktop kiosk. Color images on the wired components were rendered as black and white images on the handheld by the Nokia proxy server, which handled conversion from XML to WML.

In the course of developing of Kiosk Engine and its extension for mobile devices, we used the following multifunctional array of tools and technologies:

- Java – the primary programming language. Borland JBuilder Personal Edition was used as the development environment.
- XML – the primary data description language. The structure and layout of the kiosk is encoded using XML.
- UML – used throughout the entire project lifecycle to design the Kiosk Engine. We used Rational Rose software obtained from Rational Corp. (now acquired by IBM) as a part of a $500,000 grant.
- WML – the kiosk data format for mobile telephones. We used the free Nokia Mobile Browser Simulator to simulate the Web browser within the mobile phone. We also used Apache Tomcat server to support the JavaServer Pages serving the data to Nokia WAP Gateway.

5. Summary

The task of creating a multimedia interactive kiosk, although transparent to the end user, is quite time-consuming. Because of this and other factors, only a handful of organizations are involved in developing kiosks. Among these organizations, a subset is involved in creating some kind of generalized software so that any type of kiosk can be generated according to real-time needs of users [10]. Like other software, kiosks consist of two necessary components: data and processing power. Isolating the two components at the design stage enables the focus to be on one or the other, thus relieving the designers of the burden of concern with both components simultaneously.

In our work, we focused on the data component of the kiosk. Images, audio and video data, the needs of the end-user, and how the data should be organized to make the user experience a successful one, were our primary focus areas in the presented work. We developed a generic program that states what data types are supported in a particular kiosk and how they are organized; the Kiosk Engine presented here can generate, in a fast and efficient manner, kiosks that support a wide variety of multimedia data types. KE is a lightweight application that can be used to create and implement multimedia interactive kiosks supporting multiple types of data, which depend on customer and end-user needs. Because the KE is generic for any type of kiosk data, organizations developing kiosk applications can focus on the organization of the data and not on the types of data itself.

In this paper we presented Kiosk Engine, a lightweight application for creating and presenting multimedia kiosks. KE is a powerful, lightweight and multifaceted application that can be used for generically developing a variety of multimedia kiosks, either standalone or web-enabled, on a multitude of hardware and software platforms. The engine is built such that the kiosk developed through its use does not need to be modified based on platform. KE acts as a driver program running transparently behind the actual kiosk, whose structure is provided in an XML file fed as an input to KE. Since the data for the kiosk has been successfully isolated from the program, we could execute the kiosk on a variety of platforms to provide proof of the concept of separation between kiosk data and how it is organized, i.e. its structure.

Because of the data-organization separation, the structure of the kiosk and its data can be modified at any time, independently of one another [10]. KE also enables export of kiosk data and its structure in an independent manner. Kiosk Engine can work on stationary computers, but it was extended to work on mobile devices with limited memory, restricted display size and constrained processor power. The design and implementation of Kiosk Engine discussed in this paper started as a project at the Department of Computer Science of Columbus State University. A preliminary implementation [10] won the 2nd place in the student project competition at the 2003 ACM Mid Southeast Conference.

6. References

[1] A. Christian, B. Avery. "Speak Out and Annoy Someone: Experience with Intelligent Kiosks", In *Proceedings of the SIGCHI Conference on Human Factors and Computing Systems*, The Hague, The Netherlands, April 2000.

[2] R. Denton, P. Jensen. "From VTs to iMacs: Moving Public Computing Access into the 21st Century", In *Proceedings of the 28th annual ACM SIGUCCS conference on User services: Building the future*, Richmond, VA, October 2000.

[3] M. Douglas. "Retail Kiosks: Breaking New Ground", 12 December 2000. Available at http://www.kioskmarketplace.com/research.htm; last accessed on 23 April 2004.

[4] M. Fleck, M. Frid, T. Kindberg, E. O'Brien-Strain, R. Rajani, M. Spasojevic "From Informing to Remembering: Ubiquitous Systems in Interactive Museums," in *IEEE Pervasive Computing*, pp. 13-21, April 2002.

[5] M. Gandy, D. Ross, T. Starner. "Universal Design: Lessons for Wearable Computing," in *IEEE Pervasive Computing*, pp. 19-23, July 2003.

[6] E. Mäkinen, S. Patomäki, R. Raisamo. "Experiences on a Multimodal Information Kiosk with an Interactive Agent", In *Proceedings of the second Nordic conference on Human-computer interaction*, Aarhus, Denmark, October 2002.

[7] P. Prasarnphanich and M. Gillenson. "Virtual Extension: The Hybrid Clicks and Bricks Business Model". *Communications of the ACM*, Volume 46, Issue 12, December 2003.

[8] J. Raymond. "Electronic Kiosk Project: Distributed Access to e-mail and Web Browsing", In *Proceedings of the 28th annual ACM SIGUCCS conference on User services: Building the future*, Richmond, VA, October 2000.

[9] S. Ross, J. Hill, M. Chen, A. Joseph, D. Culler, E. Brewer. "A Composable Framework for Secure Multi-modal Access to Internet Services from Post-PC Devices". *Mobile Networks and Applications*, Volume 7 Issue 5, October 2002.

[10] M. Shah. "Kiosk Engine: A Platform Independent Solution for Multimedia Kiosks," in *Proceedings of The*

2003 ACM Mid Southeast Chapter Fall Conference, Gatlinburg, TN, November 2003.

[11] B. Yoshimi, N. Sukaviriya, H. Derby, B. Carmeli, B. Bolam, J. Elliott, J. Morgan. "Lessons Learned in Deploying a Wireless, Intranet Application on Mobile Devices," in *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*, pp. 31-39, June 2002, Callicoon, New York.

[12] S. Yura, T. Usaka, K. Fujimori, K. Sakamura. "Design and Implementation of the Browser for the Multimedia Multi-User Dungeon of the Digital Museum," In *Proceedings of the Third Asian Pacific Computer and Human Interaction*, pp. 44-49, July 15 - 17, 1998, Kangawa, Japan.