Pervasive Computing: The Cause for Including Mobile Application Development in Undergraduate IS Curriculum

Kathryn A. Marold, Metropolitan State College of Denver, Campus Box 45, P.O. Box 173362, Denver, CO 80212, USA; E-mail: maroldk@mscd.edu

INTRODUCTION

Computing for the end user has changed dramatically with the shrinkage of the personal computer and the availability of wireless connections. Not only is the Web ubiquitous and the Internet universally available, but the saturation and the size of the hardware used for personal computing has changed as well. What is commonly referred to as mobile computing has changed application development. Users have moved from cabinet-sized desktop computers to laptops and notebook computers, to personal digital devices, Palm Pilots, Blackberries, Pocket PCs, and finally to smart phones. Cell phones with Internet access and various software applications in addition to established telephony software are now common. The situation of ever-present computing power in all aspects of our lives has become known as *pervasive computing* (Bergeron, 2001.)

The importance of teaching our IS majors with modern GUI software programming tools that use new Java and .NET programming languages is well-recognized in most American computer information systems departments granting undergraduate degrees. However, teaching IS majors programming that can be ported to smart devices and interfaces with Web services that are accessed wirelessly is relatively new. The importance of new IS graduates knowing how to develop appropriate mobile computing applications should not be ignored. IS curriculum for the undergraduate should include programming for mobile devices. This gives our new graduates more competitive employment opportunities: entry level IT positions remain competitive, despite slight improvements in the job market (Bureau of Labor Statistics, 2006.)

The required programming course(s) in most undergraduate IS programs has always concentrated on the basic logic and programming techniques needed for entry level positions in industry. And that is still true: there is no substitute for basic problem solving methodology, design tools, logic of basic constructs, and rudimentary coding skills. Yet when employers advertise available positions, they almost universally request specific skills. Academics know that keeping current in IT curricula is a constant struggle, but one with definite paybacks. Keeping programming language curricula current has never been easier, or more rewarding. It *is* possible to teach mobile computing applications in our programming courses, and thus give our students experience in programming for a pervasive computing environment.

When GUI tools and IDE's (integrated development environments) became common, the old command-line instruction and heavy emphasis on syntax gave way to teaching modern fourth and fifth generation programming languages that were object oriented. *Cobol* was replaced by *C*, then *JAVA* and/or *Visual Basic*, and *C++* . Building console applications with a *C* compiler within *UNIX* often was replaced with building *Windows* applications within microcomputer operating platforms. Attention to user interface and connecting to data sources for database file processing and more robust transaction processing systems became a reality within undergraduate programming courses. The RAD (rapid application development) tools available provided an environment where students could do more in-depth programs within a semester time-frame. Next, development of Web-based transaction processing systems as well as *Windows* applications was possible when programming suites were introduced. The same opportunity now exists for students to develop Web services and applications for Smart Mobile Devices such as Pocket PCs and cell phones, using the emulators built into programming suites. Students in an existing programming class at Metropolitan State College of Denver used Pocket PCs with wireless access to develop smart device mobile applications while learning to program in *C#.NET*. The response was positive, the programs were relevant and immediately useful, and the experience obtained with WI-FI (wireless fidelity) and mobile devices were true resume builders.

DISTINCTIONS WITH MOBILE COMPUTING APPLICATIONS

The interface of modern programs is vastly different from the command-line interface of the mainframe and early microcomputer days. Text-based systems are scarce today, and are foreign to most computer users –even those in the IT industry. The early GUI appearance of application programs has evolved into a Web-like interface. Ample RAM, high resolution devices, and faster processors have given us the luxury of highly sophisticated multimedia interfaces. The smart device programs that students learn to program must take into consideration the audience using them; they are culturally, occupationally, and geographically dispersed. Blom, Chipchase and Lehikoinen conducted extensive research on contextual and cultural challenges for Nokia, Inc. (2005.) They studied the length of typical smart device use, the cultural idiosyncrasies, the personal nature of the use, the privacy and security concerns of users, the use of smart devices while commuting, walking, sitting in meetings, on public transportation—even use while driving or cycling. In 1997 Page developed a model of requirements for *Windows* mobile-based products (2005). There is a profile for the typical smart device user.

Nokia was a pioneer of smart mobile device applications. The designers of their third party application software deemed the most important aspect of the design was to provide the user with feedback. The use of cell phones and smart mobile devices by a younger population drove the user requirement for a very visual application. The proliferation of ring tones, IM (instant messaging), text messaging, and the exchange of digital photos and video by cell phone users confirm Kangas and Kinnumen's multimedia recommendations (2005.)

Sprint, Inc. holds that previously there were two basic consumer screens: the TV and the PC. In 1999 they set out to develop a third—the mobile phone screen (Balaji, 2005.) Done correctly, it would evolve into the most used personal screen, since users would use the same device—a variety of the smart phone—to communicate, to access the Web, to view TV and movies, and to compute. One device would be used individually for all computing and communicating uses; it would be small and mobile. The PCN (personal communications network) that was advocated in the late 1980s would be common by 2010 (Marold, 1996.)

THEORETICAL UNDERPINNINGS OF TEACHING MOBILE APPLICATIONS

As Roger Schank points out, scholars from Aristotle, to Galileo, to Dewey, to Einstein have concurred that real learning is by doing (2001a). We learn when we need to use the concept, and we understand when we physically perform the steps toward mastery (Schank, 2001b). Learning by doing with mobile devices is nearly impossible to implement in a traditional college computer lab setting.

However, when a disconnected practice environment is provided, when each student is responsible for his/her own server and Pocket PC, when the student has enough exposure for a comfort level, application development for mobile devices can be successful. Mobile devices and wireless connectivity are integral to the learning process. Bergeron notes that the majority of mobile technologies simply fit into the process of everyday business and personal life. If they do not fail unexpectedly, they fade out of conscious existence (2001).

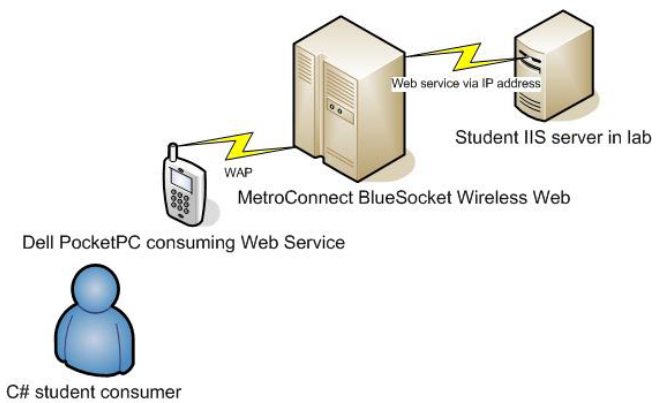
Adapting for programming instruction in mobile smart device applications requires some adjustments and some capital outlay for the institution. A special needs lab for information systems majors at Metropolitan State College of Denver was completed five years ago (Fustos, 2004). Once the lab was in operation, the potential for teaching mobile applications and Web services was evident. The programming software, the IIS (Internet Information Services) server software, and development stations were already there. All types of hand-held devices, from Palms and PDAs, to Blackberries, to cell phones have become less expensive (Tam, 2005). The prototype applications students build can give IS majors a "feel" for what modern programs they will be designing for the industry.

DETAILS OF PROGRAMMING FOR MOBILE DEVICES

Using computer labs to teach programming classes is recognized as an optimal setting—provided that the environment can be controlled. Metro's special needs lab has twenty stations in the lab, giving students plenty of room to work. There is an instructor station with display equipment and "smart classroom" touch pads that are easy to use. The instructor station controls all student client stations with *SynchronEyes* software. This allows the instructor to lock stations, broadcast the instructor monitor to all stations, and display any student client machine at any time—either to the whole class, or individually and privately. Students can work along with the instructor in interactive sessions, or easily follow an instructor demonstration or *PowerPoint* presentation on their monitors.

The added expense for this class was the purchase of twenty Dell Pocket PCs with WI-FI capability, so that the Pocket PC could serve as a client and wirelessly access the student desktop computer station as its server. Development for all applications, whether Web applications or not, was on the desktop station using the *Visual Studio.NET 2003 Professional* software. The installation of *ActiveSync* software allowed students to move their applications to the Pocket PCs from their server computers using a USB (universal serial bus) connection—a simple drag and drop operation. The special needs lab computers have an additional removable hard drive where students have all privileges and may develop Web applications using IIS (Internet Information Services). The students programmed their Web applications to use the IP (Internet Protocol) address of their own servers for the Web reference, allowing them to wirelessly consume a Web service. (Although there is a Pocket PC Emulator built into the *Visual Studio.NET Professional*, the value of being able to port programs to individual mobile devices and use WI-FI 802.11.B technology to access servers cannot be underestimated.) The class could experiment with wireless technology—WI-FI, infrared beaming, and Bluetooth short distance FTP, all of which were included on the Dell Pocket PCs. The exposure to pervasive computing included projects that were developed within the semester-long class: there were independent smart applications, mobile Web pages, and creating and consuming Web services. Figure 1 shows how this was done.

Figure 1. Using WI-FI to consume Web services with pocket PCs



DISCUSSION

The existence of the campus wireless BlueSocket Web site simplified the process of using the Pocket PCs to connect to the student lab servers. In 2004 Metro installed Wireless Access Points(WAPs) on campus. BlueSocket is heavily used by students with laptops. Therefore, students in the C# class could simply login to the BlueSocket site with their Pocket PCs, and then connect to their individual lab servers via the IP address. The simplicity of the hardware connections allowed them to concentrate on the software—the programming content of the course. Aside from the security of their assigned lab servers and their vulnerability to modifications or removal of their programs (backups were constantly stressed), student implementation was optimal. Students were responsible for programming their own applications and maintaining their own Pocket PCs. They were kept in a locked cabinet in the lab, and checked out as needed. After the instructor corrected student programs and assigned grades for them, the Pocket PCs were returned and reset for reuse. The servers are on removable drives in the lab. They are re-imaged every semester, so there is a fresh start with the desktop computers as well. The use of WI-FI and Pocket PCs for the course allowed students to learn programming using the C# programming language in a three-tier architecture. It also exposed them to three popular wireless methodologies.

CONCLUSION

Incorporating mobile computing into the information systems curriculum at Metropolitan State College of Denver has received accolades from instructors, students, administration, and potential employers. Pervasive computing that has become the *modus operandi* of our society dictates an adjustment in how we educate our future information technology workers. Experience with wireless application development with smart mobile devices is another step toward keeping information systems curriculum current.

REFERENCES

Balaji, T.S, B. Landers, J. Kates, and B. Moritz. (2005). A Carrier's Perspective on Creating a Mobile Multimedia Service. Communications of the ACM. 48:7. 49-53.

Bergeron, B. (2001). The Wireless Web. McGraw-Hill. New York.

Blom, J, J. Chipchase, and J. Lehikoinen. (2005). Contextual and Cultural Challenges for User Mobility Research. Communications of the ACM. 48:7. 37-41.

Bureau of Labor Statistics. (2006) http://data.bls.gov/oes/sector.do. Accessed 1/2/07.

Fustos, J. and K. Marold. (2004). Using a Unique Campus Instructional Computer Lab for Information Systems Curriculum Delivery. Proceedings of the 46th Annual Mountain Plains Management Association Conference. Document 67.

Kangas, E. and T. Kinnunen. (2005). Applying User-Centered Design to Mobile Application Development. Communications of the ACM. 48:7. 55-59.

Marold, K. and G. Larsen.(1996). Beyond the Internet: Using Personal Computers to Communicate. CTI of Thomson Publishing. New York. Chapter 10.

Page, C. (2005). Mobile Research Strategies for a Global Market. Communications of the ACM. 48:7, 42-48.

Schank, R. (2001a). The Computer Isn't the Medium, It's the Message. Communications of the ACM. 44:3, 142-144.

Schank, R. (2001b). Revolutionizing the Traditional Classroom Course. Communications of the ACM. 44:12, 21-24.

Tam, Pui-Wang. (2005). The Hand-Helds Strike Back. Wall Street Journal. May 18, 2005. D1-6.