Pushing Web Pages into Personal Digital Assistants: Need, Tools and Solutions

E. Costa-Montenegro, F. J. González-Castaño, J. García-Reinoso, J. Vales-Alonso, L. Anido-Rifón

Departamento de Ingeniería Telemática, Universidad de Vigo, ETSI Telecomunicación, Campus, 36200 Vigo, Spain
{javier, kike, reinoso, jvales, lanido}@det.uvigo.es

Abstract. In this paper, we analyze the problem of pushing Web contents into Personal Digital Assistants (PDAs), with minimum client participation. Typical HTTP or Javascript implementations require periodic client reloads, which impose an inefficient network utilization. We justify the existence of this problem, review existing alternatives, and propose a market-oriented implementation for wireless iPAQ Pocket PCs.

1 Introduction

Cell phone or PDA-based m-commerce [1, 2] has a promising future. Wireless clients will allow customers to purchase items and services, and to receive (possibly profile-driven) store coupons, advertisements and guidance [3]. A m-commerce *service server* determines user position with the help of an *external location system*: 3G location [4], GPS or short-range beacons [3, 5].

We can identify a number of situations in which a service server needs to push Web information into PDA clients in real-time. For instance, pushing location/profile-driven advertisements into user Web pages (when a customer arrives to a new mall store), sending alarms related to mall events (movie schedules, restaurant reservations), updating user location on a map or updating context information in a museum.

In the typical HTTP or Javascript approaches, the clients reload page contents periodically. In a mall with hundreds of users with wireless PDAs, this means a large network overload. To avoid it, reloads could be less frequent. However, the resulting elapsed time between service server events and customer notifications could not satisfy real-time constraints. Note that *a single push* at the right time would be enough. Also, note that, since users cannot be expected to be static, location information must be necessarily reloaded, and Web caching is useless.

For example, in the scenario in [5], if there is a new mall store each 10 m, there must be periodic updates each 12.5 s or less for a walking speed of 0.8 m/s. If a location update requires 10 KB, network load is 2,880 KB per user and hour. However, typical users are relatively static when shopping (say 50% of the time). Therefore, if the service server only updates information when user position changes, only 1,440 KB per user and hour would be necessary.

We are interested in developing a *connection server* for PDAs, to support server-side notifications. The target scenario is a m-commerce *intranet*, i.e. either external or internal malicious individuals cannot saturate the system, and all information is handheld-oriented (page transcoding [6] is not necessary).

This paper is organized as follows: In section 2 we describe two platforms for context-oriented mobile services and the technologies that are relevant to the purposes of the paper. Subsection *base software technologies* discusses PDA network programming capabilities. Section 3 presents our connection server architecture for Windows CE and its implementation. Finally, section 4 concludes.

2 Background

CoolTown [3] is a Hewlett-Packard distributed architecture for mobile PDAs with a WLAN interface and an infrared port. The user must aim the infrared port to location beacons, which push URLs into a PDA daemon that displays the corresponding Web pages.

In [5], the authors proposed an alternative system, m-Mall. M-Mall users carry a mobile PDA and a Bluetooth location *badge*. The badge interacts with an external Bluetooth location network, which provides the service server with real-time user position. Then, the service server may push context-oriented URLs into user PDAs via TCP/IP sockets. Thus, the main advantage over CoolTown is that no client action is required to generate user position (in CoolTown, the user must locate a location beacon and point his PDA). CoolTown authors argue that automatic detection of location information (without user participation) may have severe consecuencies in terms of nuisance (for example, if a Web page is suddenly supplanted by another one advertising frozen peas from a grocery store nearby). Obviously, we must expect the service server to be reasonable. Consider, for example, a museum, where the only updates are associated to new halls, and suppose that the current page is reloaded with a tiny icon at its bottom meaning "do you want to update context information"? We believe that asking the user to locate a beacon each time he enters a room full of visual distractions may be tiring, and signaling beacons with large red arrows unsightly. In any case, CoolTown may be more advantageous than m-Mall or viceversa depending on the specific application.

The philosophy in this paper is compatible with any of those two systems. In case of CoolTown beacons, GPS or 3G, location is obtained by some user interaction at the client side. The user would upload his position to the service server, which would push URLs back via TCP/IP sockets. In case of an user-independent location network, like [5], the client would simply receive URLs via TCP/IP sockets, when pushed by the service server.

Next, we will describe the technologies that are relevant to the purposes of this paper.

Hardware: Palm and Pocket PCs dominate the PDA market. Both systems support wireless TCP/IP socket programming: Palm Mobile Internet Kit [7]

connects Palm handhelds with cell phones via cable or infrared ports, and there exist GPRS and WLAN Pocket PCs (like iPAQ) [8].

Base software technologies: We have identified some Java Virtual Machines (VMs) for PDAs, which would allow portable socket programming [9, 10]. Java is relatively new in the PDA world. Pocket Internet Explorer does not include a VM, and the examples in [9, 10] do not include all typical libraries. For instance, when this paper was written, Kada did not support swing libraries, which would be useful to let programs open Web pages themselves. Another reason to discard those VMs is the high PDA CPU load demanded, when compared to small binaries generated from high level languages.

Most PDA applications run on PalmOS or Windows CE. Since we are interested in mass-market m-commerce applications, we focused on those OS. Of course, there exist alternatives. For example, QNX is a real-time OS. Version 6.1 includes the IBM j9 Java VM, which is part of Visual Age Micro Edition 1.3 [9]. Pocket Linux is thoroughly described in [11]. Nowadays, despite of its potential, Pocket Linux is not an alternative for mass market applications. Finally, GENA is only a proposal for a Web notification system [12].

We finally selected Pocket PC + Windows CE for our prototype, because the programming tools are quite similar to Windows 9x ones, with a broad support [13]. Also, Windows CE includes Pocket Internet Explorer, and Palm OS has not included an Internet browser for a long time. Finally, for TCP/IP prototyping, there exist Pocket PC PCMCIA jackets and Ethernet cards.

3 Connection server implementation

Figure 1 shows our architecture. A PDA *connection server* process listens for socket connection requests at a given port. Once it accepts a request, it receives an URL (or any data) via a socket (A). Then, it submits the URL to the local browser (B), which requests and downloads the corresponding Web page (C, D).

We used a Compaq iPAQ H3630 PDA running Windows CE 3.0, with a PCMCIA Ethernet card. We wrote the connection server in C++ on a host PC running Microsoft Embedded Visual Tools 3.0. The host computer also acted as service server and, for that purpose, it ran a Java connection client. The host computer and the PDA had IP addresses in the same intranet.

The connection server code for Windows CE is freely available for research purposes (by sending e-mail to the authors). *The resulting binary only occupies 7 KB.*

4 Conclusions

We have adressed the need of a system to push URLs into PDAs via sockets, for state-of-the-art m-commerce platforms [3, 5]. A prototype (figure 1) has been developed. Future work is oriented towards developing PDA clients for meta-computers [14]. Also, we will follow the evolution of Java PDA VMs, which will allow individual frame upload, improving network efficiency.

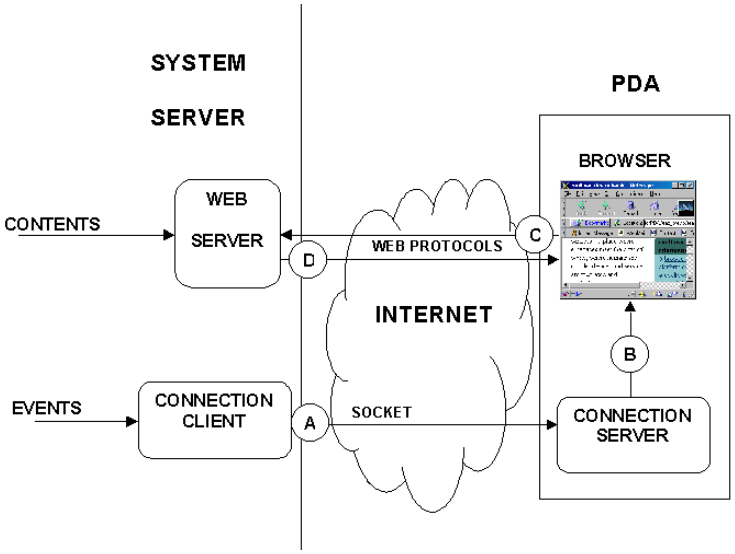

Fig. 1. Architecture prototype

References

1. Varshney, U., Vetter, R. J., Kalakota, R.: Mobile Commerce: A New Frontier. Computer. **10** (2000) 32–38
2. Darling, A.: Waiting for the m-commerce explosion. Telecommunications International. **3** (2001) 34–39
3. Kindberg, T., Barton, J.: A Web-based nomadic computing system. Computer Networks. **35(4)** (2001) 443–456
4. ETSI: ETSI document TS 122 071 V3.2.0
5. García-Reinoso, J., Vales-Alonso, J., González-Castaño, F. J., Anido-Rifón, L., Rodríguez-Hernández, P. S.: A New m-Commerce Concept: m-Mall. Lecture Notes in Computer Science. **2232** (2001) 14–25
6. Buyukkokten, O., García-Molina, H., Paepcke, A.: Accordion Summarization for End-Game Browsing on PDAs and Cellular Phones on the Road. Proc. of ACM CHI 2001 Conf. on Human Factors in Comp. Sys. 213–220
7. Palm Mobile Internet Kit. http://www.palm.com/software/mik/
8. A-Brand-New-World 4G jacket. http://www.abrandnewworld.se/
9. IBM Visual Age Micro Edition 1.3. http://www.embedded.oti.com/
10. Kada Systems. http://www.kadasystems.com
11. PocketLinux. http://www.pocketlinux.com
12. Cohen, J., Aggarwal, S.: General Event Notification Architecture Base. http://www.alternic.org/drafts/drafts-c-d/draft-cohen-gena-p-base-01.pdf.
13. Boling, D.: Programming Windows CE. Microsoft Press
14. González-Castaño, F. J., Anido-Rifón, L., Pousada-Carballo, J. M., Rodríguez-Hernández, P. S., López-Gómez, R.: A Java/CORBA Virtual Machine Architecture for Remote Execution of Optimization Solvers in Heterogeneous Networks. Software, Practice and Experience. **31** (2001) 1–16