SOFTWARE ARCHITECTURES FOR MOBILE LEARNING

D. PARSONS & H. RYU

Institute of Information & Mathematical Sciences
Massey University at Albany, Auckland, New Zealand

Mobile learning (M-learning) systems may be built using any of a number of generic software architectures, each of which has its own benefits and limitations. This paper describes four generic software architectures that can be applied to M-learning systems and provides an overview of their comparative strengths and weaknesses in meeting the various requirements of the mobile learner.

1 Introduction

As mobile technologies have increasingly found their way into all aspects of our lives, researchers have explored how these technologies might contribute to learning. Many mobile learning (M-learning) applications have been developed that extend the electronic learning (E-learning) experience into the mobile context (e.g., Chang and Sheu 2002; Chen, Kao et al. 2002; Liu, Wang et al. 2002). M-learning is an approach to E-learning that utilises mobile devices, but should not be seen as just another channel for delivering the same content. In fact quality M-learning can only be delivered with an awareness of the special limitations and benefits of mobile devices. This is not to say that awareness of known quality issues in E-learning is not relevant to the mobile learning context, but it is also important to identify features unique to M-learning when considering how to deliver a quality education experience.

This paper examines what kinds of software architecture might be used to build M-learning systems and outlines what factors and issues should be considered in terms of the benefits and drawbacks of each generic architecture. In the following section we review some relevant literature on key aspects of M-learning systems. We then outline a number of software architectures that may be used to build M-learning applications, looking at how each approach may contribute to the requirements of M-learning while considering the practical challenges and limitations. We provide a summary of the key issues associated with each architecture and suggest some recommendations for software architectures appropriate to different types of mobile learning system.

2. Mobile devices and mobile learning

Mobile technologies, and particularly mobile telephones, are as much social objects as technical ones. They impact how we organise our lives, how we work, and, perhaps, how we learn. Designing M-learning systems therefore is not simply a technical challenge but one that has to take into account the social context of using a mobile device as a learning tool.

The primary success factor of the mobile phone is its ability to support a range of individual requirements, e.g., sending and receiving telephone calls, text messages,

multimedia messages and so on. Indeed, each person uses their mobile phone in different ways. For instance, teenagers frequently use Short Message service (SMS) for their communication medium (e.g., Schiano, Chen et al. 2002), while business professionals are more likely to use their mobile device as a personal communication centre. This aspect – different user profiles and their roles in use of mobile technologies – becomes even more important in an M-learning environment. For example, the use of SMS (or picture messages) in M-learning (Seppälä, Sariola et al. 2002; Stone, Briggs et al. 2002) has been identified as an effective tool to enhance both students' learning experience and teachers instructing experience, yet the requirements from both parties were quite different.

The numerous constraints on the technology and user interface of mobile devices are an important consideration in M-learning environments. At the moment mobile devices suffer from small screens, poor input methods, and limited battery life. Therefore, M-learning services must be carefully designed to fulfill users' needs, without overloading them with unnecessary complexity and slow operation. The software architecture chosen for a given M-learning system will therefore have a major impact on the utility of the system from the user's perspective. There is, however, some positive evidence that M-learning is able to succeed even in limited technical environments. For instance, a study by Ericsson in 2002 showed that even with a simple Wireless Access Protocol (WAP) browser interface, 77% of the participants felt that M-learning actually increased their learning performance (Ericsson 2002). Stone et al. (2002) developed a simple M-learning application using SMS as an interactivity mechanism. They concluded that the lack of sophistication of the platform was not a major barrier to the quality of the learning experience, and M-learning applications can encourage active and wide participation of both the learners and the tutors, even where it might be expected that technical limitations would discourage them. It seems therefore that technological sophistication is not necessarily a measure of usefulness, so in designing the software architecture for an M-learning system we should consider not only the technical sophistication of the platform but how well that platform meets the needs of the learner. Mobile applications should not distract with unnecessarily rich media objects (e.g., Uther 2002), so the format of content has to be chosen with care. The learning context, in particular interactivity, is more significant than the media types used. Luchini et al. (2004) stress the importance of the user participating in and learning about underlying concepts and processes rather than learning by rote, and content should be up to date and highly interactive, enabling mutual feedback between education providers and learners and assisting in the identification of knowledge gaps (Lehner and Nösekabel 2002; Massy 2002). Therefore in designing a software architecture for M-learning we should consider support for collaborative learning as a priority.

3 Mobile versus mobile and wireless

One of the more obvious, but nonetheless important distinctions to be made between different implementations is whether the application is mobile, wireless or both. As Mallick (2003) indicates, wireless is a usually a subset of mobile, but of course there are many applications that are mobile without being wireless, and also those that are wireless but not mobile (Figure 1)

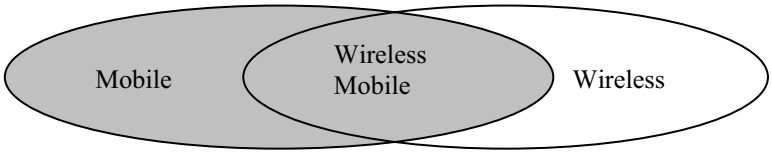

Figure 1: The relationship between wireless and mobile, with the area of interest for mobile learning highlighted (adapted from Mallick).

For the purposes of mobile learning, applications that are wireless but not mobile are not relevant to our discussion, however the distinction between mobile on the one hand, and wireless mobile on the other, is significant. Within this distinction there is another dimension, which is the level of interactivity of a solution that requires wireless connectivity. For example, mobile learning content that may be downloaded wirelessly from the Internet, whether that content is in the form of a podcast, MPEG file, Java MIDlet or some other downloadable and/or installable component, may not require further wireless connectivity after download. In contrast, an application may be downloaded that will require further contact with a wireless server while it is being used. This requirement will have a potentially negative effect on the cost and availability (i.e. being dependent on network availability) of the mobile learning content, while having the positive effect of enabling greater interactivity, breadth and currency. These are the kinds of trade-offs we examine in this paper.

4 Non adaptive mark-up

A number of mobile learning applications have been developed that use some specific form of browser mark-up for their client side presentation. This mark-up may be Wireless Markup Language (WML) or variations on the HyperText Markup Language (HTML) such as cHTML (compact HTML), XHTML (eXtensible HTML) Basic or XHTML Mobile Profile, depending on which types of mobile device are being supported. Regardless of the mark-up, the content may be served as static pages or generated dynamically on the server, using technologies such as server pages and/or eXtensible Stylesheet Language Transformations (XSLT) (Figure 2). Fowler classifies these to approaches to dynamic mark-up as the template (server page) and the transform (XSLT) approach (Fowler 2003), though in fact it is possible to combine the two, for example by using Tag libraries (such as the JSP standard tag library, the JSTL) in server pages that support transformations.

The advantage of this approach is that it is lightweight from the client device perspective requiring only the device's normal browser. The problem with non-adaptive mark-up is that using a particular mark-up language, for example WML, means that the content can only be rendered by browsers that understand that particular type of mark-up. Even though the content may be dynamically generated, it is only being generated for a specific type of client.

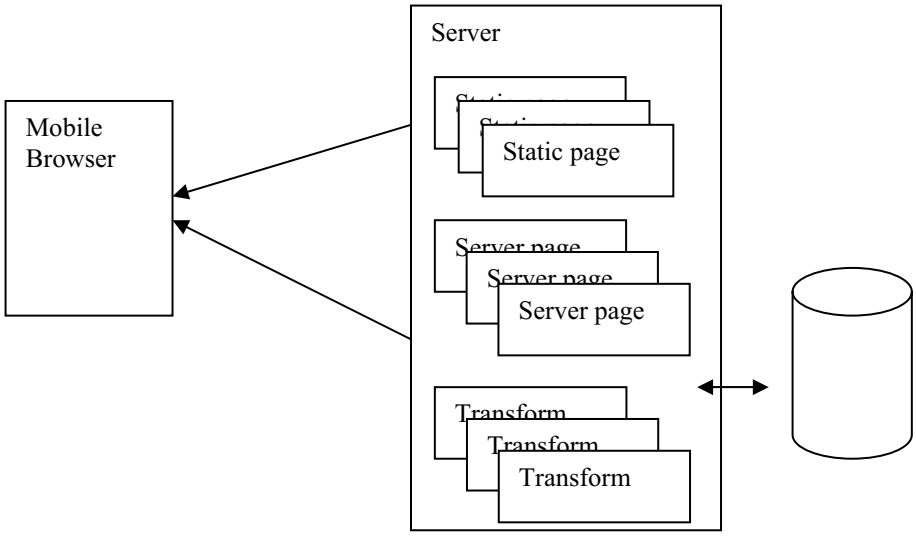

Figure 2: Non-adaptive mark-up architecture

5 Adaptive mark-up

Adaptive mark-up requires a server side process that is able to generate mark-up appropriate to the mobile device from a common set of contents. There are a number of approaches to this, for example an application can interrogate the HTTP (HyperText Transfer Protocol) header of the request and identify the client browser type from the 'user-agent' field, then generate client specific mark-up using various XSL transformations. An alternative approach is to use a tag library such as Wireless Abstraction Library (WALL), a JSP tag library that builds on the Wireless Universal Resource File (WURFL) (Passani and Trasatti 2002-6). WURFL is able to recognise user agent information and identify different devices, while WALL generates device specific markup. Either way, the advantage of this approach is that it enables an M-learning application to support multiple types of mobile browser (Figure 3).

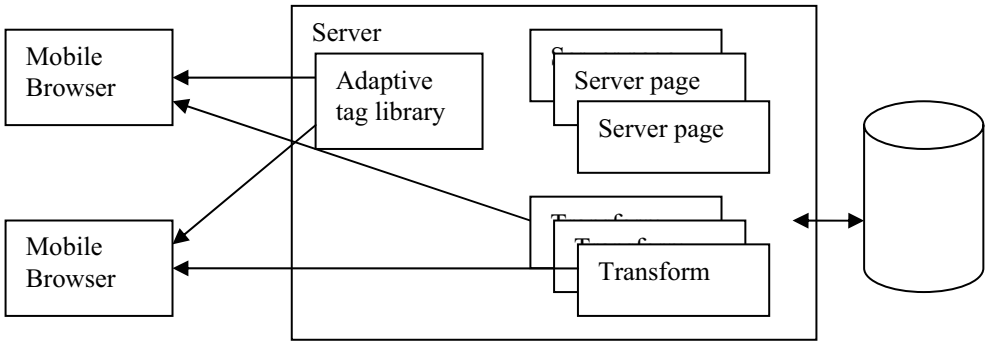

Figure 3: Adaptive mark-up architecture

6 Mobile client side application

While the mark-up approach to M-learning architectures can provide a good range of content across a wide range of devices, confining the learner's activity to what can be supported by a mobile browser can limit the range of learning activities that are possible on the device. For example, interactive learning games may be difficult or impossible to support. An alternative approach to using mark-up to provide content

via the mobile Internet is to provide applications that can be downloaded to the mobile device. There are three general categories of application that may be developed for mobile devices. The first approach is an application written for a specific mobile device platform, for example targeting a particular model or make of mobile phone. These applications can take advantage of the special characteristics of that particular device, which can make them, for example, highly performant, but of course these applications cannot be used on other devices. A second approach is to use Microsoft Windows based applications, for example building an application using Windows Mobile components. This approach is more generic than writing for a particular device, since there are a number of devices that support Widows Mobile. However such applications cannot run on the majority of mobile devices, since there are many other operating systems being used including Palm OS, Symbian and Linux. The third, most generic approach, is to use Java Micro Edition (Java ME)*. Using this software platform enables us to deliver a relatively rich client experience to a wide range of devices. Although Java ME has its limitations compared to some other application platforms such as Symbian and BREW, it works across a large proportion of mobile devices, including many Windows phones (Coulton, Rashid et al. 2005). In the context of mobile phones, the specific configuration and profile typically installed is the Connected Limited Device Configuration (CLDC) supporting the Mobile Information Device Profile (MIDP). Java ME applications that run using this profile are known as MIDlets. Of course coding at the higher level of abstraction that enables interoperability has its costs in performance terms. For example Java ME applications have been shown to execute at about half the speed of equivalent C programs (Domer, Nanja et al. 2004). There are also issues regarding the version of both the CLDC and MIDlet specifications that a given phone may support. Important differences between versions include floating point number support and security management, among many others.

If a mobile client application is chosen as the software architecture, then the overall system design is simple. The application runs standalone on the client, so the only role of the server is to enable download and installation of the application (Figure 4). This may be done either wirelessly or using a cable. Wireless download of Java applications can be provided using OTA (over the air) provisioning, a standardised approach that provides a consistent client-server interaction and enables version control for application downloads. The mobile application may use the data store on the mobile device but will not require access to any server side resources.

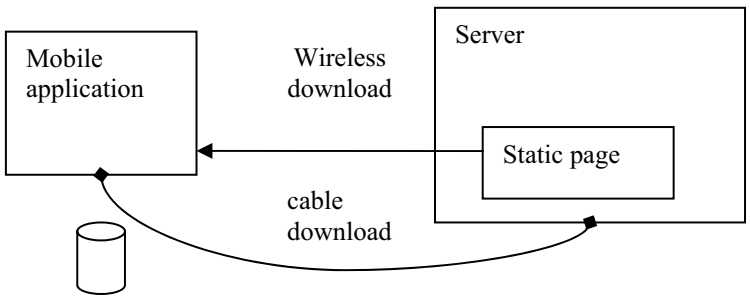

Figure 4: Client side application architecture

* Java ME was formerly known as Java 2 Micro Edition (J2ME)

7 Smart client with server connectivity

So far we have considered two very different types of software architectures for mobile learning systems, one based on providing a page based mobile Internet system, the other using a downloadable application client. However there is another approach that combines features from both of these architectures, the smart client that connects to the server. In this approach, there is a client side application, but that application does not run standalone. Rather, it communicates with the server to send and receive information while the application is running (Figure 5). There are a number of advantages to this approach. First, it is possible for the mobile learning application to provide a much wider set of content than is possible with a single downloaded application, since the storage size of the mobile device limits the amount of learning content that can be downloaded. A smart client that connects to the server can access learning content on demand without having to keep it all stored in the mobile device. Similarly, we can utilise the server to store information about the clients so we can, for example, maintain a sophisticated user profile on the server. In addition, the ability of the device to communicate data with the server in both directions (upload and download) means that it is possible to build a collaborative learning system where multiple users can send and receive data to and from each other.

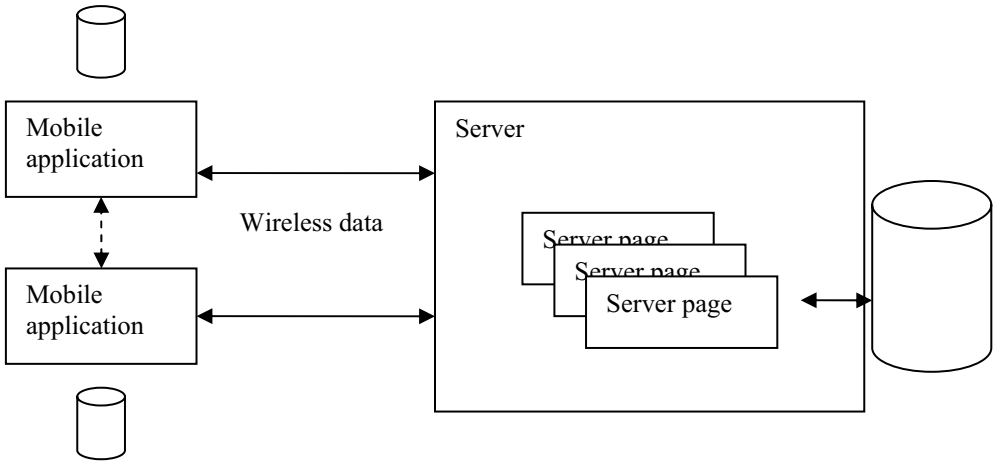

Figure 5: Smart client with server connectivity architecture

The price we must pay for this power and flexibility is complexity. For example, data management becomes a more complex issue because there may be data stored in a local data store on the client device as well as on the server. Mobile databases will need to be synchronized with central databases, and cache management has to be sophisticated to cope with the small memory size in most mobile devices. Distributed smart client applications using this architecture also need an appropriate communication protocol, such as XML over HTTP, to enable the client to access server side resources at run time.

8 Summary

In this paper we have provided a brief overview of four software architectures for mobile learning; non adaptive mark-up, adaptive mark-up, mobile client side application and smart client with server connectivity. All of these architectures have their own strengths and weaknesses, and in most cases we are trading flexibility against complexity. In addition there are different levels of server connectivity required for these different architectures, and successful applications depend not only

on the technical infrastructure but also the social context within which issues such as pricing come into play. Therefore deciding on a suitable software architecture for a specific M-learning system depends not only on technical factors but also an analysis of the user context. While we may see that a smart client architecture with server connectivity can provide us with the richest mobile learning environment, alternative architectures may prove easier to install, more robust in use, more easily deployed to a larger range of devices and cheaper for the learner to maintain. Therefore the most important aspect of designing an M-learning system architecture is to consider all aspects of the user context rather than just focus on the technical platform.

References

Chang, C. and J. Sheu (2002). Design and implementation of ad hoc classroom and eschoolbag systems for ubiquitous learning. IEEE Int. Workshop Wireless and Mobile Technologies in Education.

Chen, Y., T. Kao, et al. (2002). A mobile scaffolding-aid-based bird-watching learning systems. IEEE Int. Workshop Wireless and Mobile Technologies in Education.

Coulton, P., O. Rashid, et al. (2005). "Creating Entertainment Applications for Cellular Phones." ACM Computers in Entertainment **3**(3).

Domer, J., M. Nanja, et al. (2004). Comparative Performance Analysis of Mobile Runtimes on Intel XScale® Technology. 2004 workshop on Interpreters, Virtual Machines and Emulators (IVME'04), Washington, D.C., USA, ACM Press.

Ericsson. (2002). "Mobile learning in action: Report on the use of mobile telephones for training." Retrieved January, 2006, from http://learning.ericsson.net/mlearning2/project_one/mobile_learning.html.

Fowler, M. (2003). Patterns of Enterprise Application Architecture. Boston, Addison-Wesley.

Lehner, F. and H. Nösekabel (2002). The Role Of Mobile Devices In E-Learning - First Experiences With A Wireless E-Learning Environment. IEEE International Workshop on Wireless and Mobile Technologies in Education (WMTE'02), Växjö, Sweden, IEEE.

Liu, T., H. Wang, et al. (2002). Applying wireless technologies to build a highly interactive learning environment. IEEE Int. Workshop Wireless and Mobile Technologies in Education.

Luchini, K., C. Quintana, et al. (2004). Design Guidelines for Learner-Centered Handheld Tools. SIGCHI Conference on Human Factors in Computing Systems, Vienna, Austria, ACM.

Mallick, M. (2003). Mobile and Wireless Design Essentials. Indianapolis, John Wiley.

Massy, J. (2002). "Quality of eLearning Must Improve." Retrieved January, 2006, from http://www.learningcitizen.net/articles/QualityofeLearningmu.shtml.

Passani, L. and A. Trasatti. (2002-6). "WURFL." Retrieved February 9th, 2006, from http://wurfl.sourceforge.net/.

Schiano, D. J., C. P. Chen, et al. (2002). Teen use of messaging media. CHI, Minneapolis, MN, ACM Press.

Seppälä, P., J. Sariola, et al. (2002). Mobile learning in personnel tranining of university teachers. IEEE Int. Workshop on Wireless and Mobile Technologies in Education, Växjo, Sweden.

Stone, A., J. Briggs, et al. (2002). SMS and Interactivity - Some Results from the Field, and its Implications on Effective Uses of Mobile Technologies in Education. IEEE International Workshop on Wireless and Mobile Technologies in Education (WMTE'02), Växjö, Sweden, IEEE.

Uther, M. (2002). Mobile Internet Usability: What Can 'Mobile Learning' Learn From the Past? IEEE International Workshop on Wireless and Mobile Technologies in Education, Växjö, Sweden, IEEE.