BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **6**, No 3

Sofia • 2006

Mobile Learning Applications – Ubiquitous Characteristics and Technological Solutions[1]

Danail Dochev, Ivo Hristov

Institute of Information Technologies, 1113 Sofia
E-mails: dochev@iinf.bas.bg collin@iinf.bas.bg

Abstract: *The paper describes ubiquitous characteristics and technological aspects of m-Learning. The most common abstract architecture is presented. Today's mobile OS and mobile browsers are described in the context of available tools and frameworks for design and development of mobile e-Learning applications.*

Keywords: *e-Learning, ubiquitous learning, m-Learning*

1. Introduction

The development of ubiquitous learning is tightly connected with the general e-Learning progress. According to the short- and middle-term prognoses of the European technology platform "Networked and Electronic Media" (NEM) [1] the e-Learning solutions will be driven by the following scenarios:

- Personal environments will be populated by personal communication and computing devices, accessories, wearables, implants. e-Learning services will be adapted to the user's individual situation, location and preferences.
- Business environments will benefit from e-Learning solutions creating a competitive advantage for European business and will facilitate especially SME's exploring new markets.
- Mobility and ubiquitous access will be a key challenge for in-field job training needs.

Nowadays e-Learning, in order to obtain real iniquitousness, demands high bandwidth broadband, new high quality graphical environments, the introduction of new and innovative services in handling digital content. e-Learning services also require interoperable networks, such that the content could be accessed through different channels in a seamless fashion by the end users. The widespread broadband

[1] The research has been supported by LdV project RF-8103 CHIRON "Referring Innovative Technologies and Solutions for Ubiquitous Learning".

access implies for the user an overflow of information. In this way, intelligent agents and other smart service discovery mechanisms are needed to allow users to have a personalized media experience. A user should be able to search for content he/she likes without dealing with different access networks (like UMTS, DVB-H etc). Further research on semantic preferences at the various levels is needed.

2. Specific characteristics of mobile learning

2.1. Learning characteristics

Mobile learning is regarded as an emergent paradigm in a state of intense development fuelled by the confluence of three technological streams – ambient computing power, ambient communication and development of intelligent user interfaces [2]. Mobile learning /m-Learning/ has been defined as learning that takes place via wireless devices as mobile phones, Personal Digital Assistants (PDAs), or laptop computers. According to many definitions in the literature, it is only the employment of specific types of technology that seem to differentiate mobile learning from other forms of e-Learning.

However, when considering mobility from the learner's point of view rather than the technology's, it is much more important that mobile learning goes on everywhere – e.g. pupils revising for exams on the bus to school, doctors updating their medical knowledge while on hospital rounds, language students improving their language skills while travelling abroad. All these instances of formal or informal learning take place while people are on the move. That's why a definition of mobile learning should be widened to include:

Any sort of learning that happens when the learner is not at a fixed, predetermined location, or learning that happens when the learner takes advantage of the learning opportunities offered by mobile technologies [3].

2.2. Mobile Devices

Capabilities of mobile devices include hardware and software features. The focus will be kept on base OS software equipment of devices, because it encapsulates specific hardware implementation for the needs of development of application that supports ubiquitous learning. The mobile devices may be classified as four major types: Personal Device Assistants (PDA), Smart Phones, Cell Phones, Tablet PC and Notebooks. Bellow some basic characteristics of devices are summarized:

- Personal Digital Assistant (PDA) – has small sizes and significant processor power. New models support more than 65000 colours. Recognize handwritten text and can play different types of multimedia files. Screen size is larger then the size of Smart Phones.
- Smart Phones – those kinds of devices are hybrid devices which combine the abilities of cellular phones and PDA. They have smaller sizes than PDAs and bigger than cellular phones. Typically there is no full sized keyboard. Windows Mobile OS [4], Symbian OS [5], Palm OS or Linux Mobile [6] OS is used.
- Cell Phones – low class devices mainly can be used for voice communication and sending and receiving of text messages (SMS). Some of their disadvantages are low memory capacity and low data transfer rate. The cellular phones from the higher class can be used to Internet access via WAP or GPRS technologies.

• Tablet PC and Notebooks – Modern notebooks and Tablet PCs are equipped with wireless network card and Bluetooth or/and infrared ports. Processor power, screen resolution an operation system memory of today's machines is enough to allow usage of reach multimedia content. An obvious disadvantage of such as devices is that their mobility is less then mobility of PDAs and Mobile Phones. Different versions of Windows or Linux distributions are used as operating system.

3. Abstract architectures for mobile applications

Without support of concepts and approaches of "traditional" e-Learning, using mobile devices and technologies cannot serve efficiently for educational needs. On the other hand, mobile technologies give to the e-Learning the power of ubiquitous access independent of to time and place. Ubiquitous learning can be often being presented as binding e-Learning with diverse communication technologies, e.g. mobile technologies [7]. By technical point of view an appropriate architecture have to be able to support solutions that overcome limitation of mobile devices such as screen size, screen resolution, ability to input data and etc., and to be able to take the advantage of PDAs that exceed in mobility traditional PC used in e-Learning.

A generic architecture that utilize usage of existing Learning Management System (LMS) is presented on Fig. 1 [8]. The architecture provides all possible e-Learning services and additional services to the mobile users by enhancing LMS functionality in three aspects: "Context Discovery"; "Mobile Content Management and Presentation Adaptation" and "Packaging and Synchronization". The following services of an existing LMS have to be modified to support mobile learning:

• Resources

– Support of learning objects (LO) – any written digital material, applet, link to other sources, simulations, etc. Breaking the educational content into small pieces allows modularity and reusability of the content.

– Tests and quizzes – the lecturers can a priori define questions and the corresponding answers for automatic formal examination or self-assessment of the students.

– Learning Metadata repositories – specific data can be kept to additionally describe the learning content elements, which can be used to catalogue learning objects, to facilitate searching and reusing.

• E-Learning specific services

– Content management services – for grown people studying by default is arranged in courses, lectures, classes, etc. thus an e-learning system must have the notion of Course and Lecture. The course can be composed by series of resources: syllabus, one/many lectures, structure for describing lecture sequence, forum, board, etc. A lecture is usually composed by many resources: presentation section, exercise section, additional material section. All these components should be organized and accessed through a proper engine. There could be searchable directories of courses, programs, etc.

– Self-assessment – one of the main advantages of computer-supported learning is the automation of some important processes. The a priori defined questions and corresponding answers allow automatic generation of different versions of tests and

quizzes but also automatic checking of the results, evaluation of performance and comparison with others' results.

– Tools to support learners and tutors in managing their learning resources – some systems can allow different users to have their own workspace and to upload personal resources (links, documents, notes, etc.). The access to resources must be controlled by permissions, checked against user authentication.

- Common services

– Support of different actors (User management) – students, teachers, tutors, administrator and sometimes guests. Different registered users can have different levels of permissions.

– Collaboration – synchronous (whiteboards, chat rooms, web-cast), asynchronous (discussions, forums, message/news boards, e-mail, mailing lists), cooperative work (shared electronic whiteboards, video/audio conferencing, multi-party game simulations). Usually few different services are offered for communication between users of the system (learners, lecturers, tutors, mentors). Some activities group the users for cooperative work, other are for posting important or topical information.

– Events management – usually calendars and schedulers are provided for all the users. They can take into account the single student/lecturer events and also group events.

- Presentation

– Presentation of content – The common requirement is that e-learning system must be accessible from a single point (e.g. a portal) by using a WEB browser, but also special applications can be used.

– User/activity tracking and monitoring – history of the interaction of the actors and the system and statistics on the performance are often important sources of information and basis for adaptation of the system.

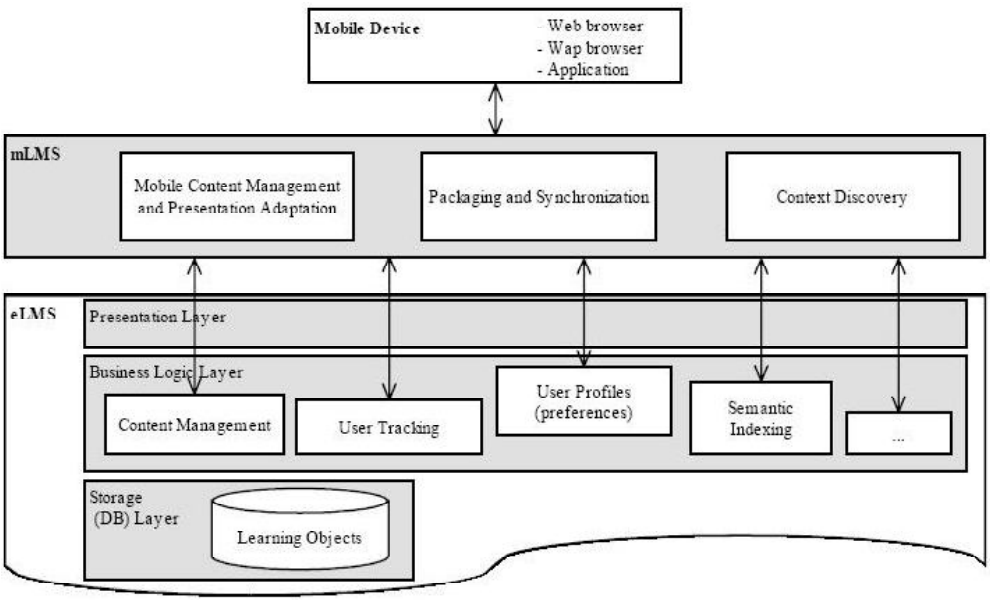

Fig. 1

On Fig. 1 eLMS [8] denotes traditional LMS and mLMS is an extension that utilize usage by of learning content by the mobile users. The functionality of eLMS functional blocks is widely know and won't be discussed. The mLMS layer is composed by the "Context Discovery", "Packaging and Synchronization" and "Mobile Content Management and Presentation Adaptation" [8].

The "Context Discovery" automatically detects the devices' capabilities and limitations (software and hardware) and check what services can be provided. "Context Detection" adds additional abstraction that can hide the details about the different physical methods of context discovery. For example for finding location different positioning systems can be used – in one case the user will be outside and can use a GPS system and in other will be inside the building and will use the local network.

The "Mobile Content Management and presentation adaptation" is responsible for selecting the services proper for the device capability and adapt them. The presentation adaptation can include adaptation of the structure, adaptation of the media format, quality or even type, etc. This subsystem is also used to adapt the presentation for auxiliary services, not only presentation of content.

The "Packaging and Synchronization" subsystem is responsible for supporting disconnected scenario. During offline operations the subsystem continues tracking of the user activities and feeds back the statistics to the LMS.

More details for "Mobile Content Management and presentation adaptation" block are used by the project Walkabout U-Learning [7, 9]. On Fig. 2 basic elements of the project architecture are shown. Common store is a place where learning objects, learning tasks, learning expositions, learning communication and administrative functions are stored for access of both mobile devices and desktop computers. The filters all objects from the common store according to a set of filtering criteria. Such criteria might include: whether the object can be implemented technically on a particular mobile device, or whether the object is useful for the learner in a mobile context. An application of a technical criterion might determine that, for example, downloading large files to a handheld is not feasible, while streaming audio lectures is quite feasible. An application of an educational criterion might further determine that the streaming of audio lectures to a mobile device is also quite useful, and so this

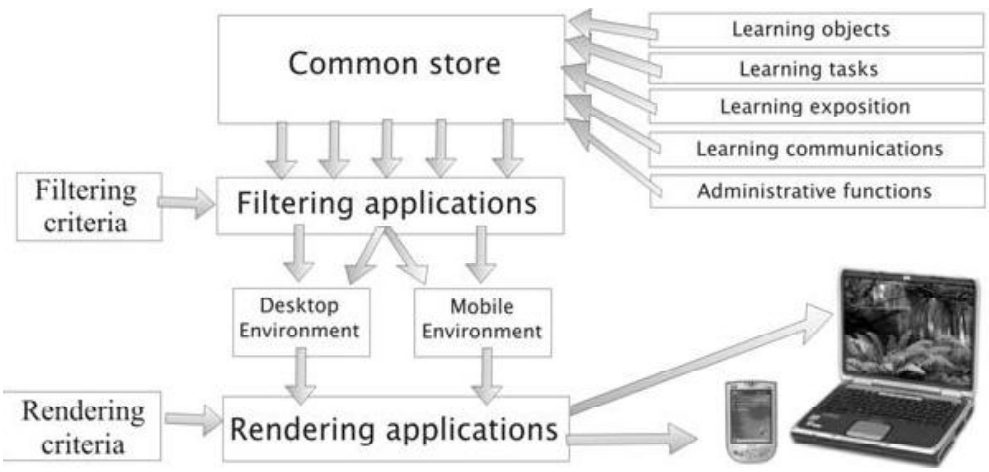

Fig. 2

feature would be implemented in the m-learning environment. As indicated by the arrows in Fig. 2, objects in the common store could be deemed technically feasible and educationally suitable in either the e-Learning or the m-Learning environments, or both. Having filtered all objects for their environments, the architecture needs to render each object according to its e-Learning, m-Learning or u-Learning requirements, taking into account the environments' browser capabilities and device technical limitations.

The architecture on Fig. 2 serves as base line for development of Ubiquitous learning application using different development tools and development environment. It can support fully connected scenario and disconnected scenario. In connected scenario the learner is permanently connected to Internet and he/she can request any learning object, if learning design allows it. This scenario may not be cost efficient in many situations. Disconnected scenario on the other hand suggests that learner connects to the Internet; take a part of the learning content locally in memory of his/her mobile device and disconnects from Internet. After getting a part of learning content, he/she connects to Internet and synchronizes local data with these stored on the central server. Sometimes this way of interacting is called "download on demand" and the control of when to reconnect/synchronize is responsibility of the learner. Support of the disconnected scenario is very important, because the Wi-Fi "hot spots" even becoming more and more popular, probably will not cover all places where learners may learn. The disadvantage is that learner cannot use all learning material while disconnected or the price for data transmission is relatively high.

The layered architecture described above is very general and deals with wide range of functionality the LMS has to support. The layered approach implemented allows extensibility and flexibility in supporting wide variety of learning activities. Implementation of such as architecture may not be cost efficient for very specific or small learning activities. In fact there are a lot of stand alone educational applications that are based on monolithic architecture where user interface, business logic and data are not separated. This kind of applications usually covers a very specific aspect of learning activities, represents a stand alone unit of learning (usually fixed) or support trainers and tutors in some specific task. Those kinds of applications are not intended to be integrated at any curriculum.

4. Development principles targeting mobile devices

Development of ubiquitous learning application is dependent of four key factors, affecting effectiveness of usage of application: capabilities of communication protocol being used, architecture implementation, learner device capabilities, software tools and technologies used for development.

The following characteristics are typical for development of mobile applications, which supports ubiquitous learning [10]:

- Narrow Bandwidth

The first big difference in contrast networking programming is the narrower transmission bandwidth available in the wireless medium today. In addition, developers need to think in terms of whether their applications can operate with different feature sets and image quality, depending on the bandwidth available.

- System Resources used – where the application is processed

Wireless bandwidths must be considered together with system resources in making decisions about client-server applications. With all networks, developers must determine where the processing will take place – in the local system or in the network server. Factors that enter into this determination not only include the transfer rate, but also the amount of performance available in the local system. In the case of mobile, battery-operated systems, the power consumption required by a program is an issue, and so is the available memory.

All of these factors make a difference in determining how much of the processing should be performed in the handset and how much can be offloaded to the network. Generally speaking, if the application produces a lot of data, especially in real time, the goal will be to reduce the need for data transfer through compression and decompression. On the other hand, if the application is computation-intensive but involves relatively little data, the goal will be to offload as much processing to the network as possible.

Different types of operations lend themselves to each approach. For instance, searching a large data base for information would produce relatively little data for the amount of processing involved. This application, which is limited by processing rather than by bandwidth, is clearly better performed on a server. On the other hand, decoding an MP3 file is more appropriate for handset processing. Bandwidth is the constraint in this case, and the goal is to minimize the amount of data that has to be transmitted.

- Memory Consumption

Within the system itself, an important constraint for programmers is the amount of available memory [10]. Whereas PCs today have gigabytes of program storage and virtual swapped memory, wireless handsets typically have up to 32 MB that is shared between stored and active program memory. This memory is not readily expanded by upgrading (in most cases), and it cannot be virtually enlarged, since hard disk drives are not a component of most hand-held wireless systems.

Memory constraints make it essential that applications programmers minimize program and data space requirements by optimizing software and removing any unnecessary features. Optimization may require a more granular approach than PC developers are accustomed to, with a line-by-line analysis of the source code to examine how compact it can be made, as well as how efficiently it executes. Programs may need to be designed modularly, so that more routines operate at the server end than with PCs, or so that individual program features can be downloaded through the network to the handset only when they are required during the course of a session.

Programmers who are accustomed to the transparency of memory utilization in PCs and workstations should realize that memory management in embedded OS is not as sophisticated. Since there is no virtual swap space on hard disk drives, dynamic memory allocations must be monitored carefully to avoid running out of memory. In addition, some embedded OSes will not clean up all allocated memory when exiting processes. Therefore, the application should not only eliminate unnecessary memory allocations, but it should also free all memory allocations on exiting processes to prevent memory leaks.

Among programming techniques, developers should beware of recursive functions and other procedures that push stacks to large sizes. When applications must use a function that calls itself, the function should not be deeply nested. Similarly, copying large objects or passing them by value should be avoided whenever possible in favour of using pointers or passing the objects by reference.

• Display Limitations

Anyone who has looked even briefly at a handheld system realizes that the display is smaller and has a lower resolution than that of a PC. While 1024×768 pixels is a common resolution on PCs, mobile devices typically have screens with resolutions of 240×320 pixels or less. Within this small space, wireless OSs do not normally support multiple windows, though dialog boxes for input, messages and so forth may be available.

The limitations of handheld displays seem obvious, yet they have profound implications when it comes to designing the "look and feel" of the application. Developers must take care to eliminate unnecessary data from the screen in order to present a simple, intuitive interface that best uses the available space. Often the appeal of an application in a larger system lies in taking advantage of the extensive capabilities of the display and system graphics. In a handheld system, with its small, low-resolution screen and simple graphics, the application will have to be more limited in its video output. Here the challenge for the software developer is to take less and make the most of it to create a satisfactory visual experience for the user.

• Power Consumption

In mobile systems power consumption is an overriding concern, so developers should be aware of and use low-power system features that are available to them. Wireless OSs typically provides power management features that allow for the partial shutdown of the system when there are idle cycles. Therefore, it is important for the application to return control to the OS when it is waiting for a system resource. For example, if the application needs input from a button on the keyboard, it should create an event, and then wait for the OS to inform it that the event has occurred. Doing so eliminates so-called "busy waiting", when the application does not return control to the OS while it is idling, thus saving power and enabling longer use of the system between battery charges.

• Development Environment Features

The wireless development environment is different from that of PCs. Embedded OSs offer fewer application programming interfaces (APIs) than PCs do, and the APIs in some OSs will feel unfamiliar to those who have programmed only PCs. OSs that use a subset of PC APIs can reduce the learning curve and make it easier to port software, but programmers need to be aware that not all PC functionality is supported in mobile systems. The various OSs available each have different advantages, but in all cases the functionality of mobile systems is more limited than that of PCs.

5. Development environment and tools

User interface of mobile application is HTML based or based on functionality of OS. Design of application that use Mobile WEB browser for presenting user interface has advantages like: central application management and versioning – on the web server; interoperability – developers do not target specific APIs that are different for different OS; usage of standardized widely accepted features – like XHTML, HTML, and XML. Some difficulties of development of mobile applications comes from the fact, that different browsers shows content in slightly different way, and optimization dependent of every request may be required targeting specific WEB browser used.

Popular Mobile WEB browsers used in mobile devices are Opera Mobile, Opera Mini [13], Pocket Internet Explorer [11] and Mozilla Firefox Minimo [12].

Mobile applications that are not using mobile WEB browsers for presenting user interface have to be developed and optimized for concrete OS and/or device being used. In that scenario application developers have access to the specific OS APIs and application can be optimized in more detailed aspects. Upgrading the applications cannot be processed on the central server. User interface can be reach and including some graphic elements, but this can lead to intensive power consumption and/or memory usage – resources that are limited in mobile devices.

Today's most popular development environments for mobile devices are: Java 2 Platform Micro Edition (J2ME), Microsoft NET Compact Framework, Opera AJAX Platform and Palm OS Developer Suite. These development tools are targeting mobile OS and browsers. In fact many application environment features has common purpose, based on the some standards, but specific realizations are different. Development environment described below are primary used for development of mobile application.

- Java 2 Platform Micro Edition (J2ME) [13]

The Java 2 Platform, Micro Edition (J2ME) provides a robust, flexible environment for applications running on consumer devices, such as mobile phones, PDAs, and TV set-top boxes, as well as a broad range of embedded devices. Like its counterparts for the enterprise (J2EE), desktop (J2SE) and smart card (Java Card) environments, J2ME includes Java virtual machines and a set of standard Java APIs defined through the Java Community Process, by expert groups whose members include leading device manufacturers, software vendors, and service providers.

J2ME architecture comprises a variety of configurations, profiles, and optional packages that implementers and developers can choose from, and combine to construct a complete Java runtime environment that closely fits the requirements of a particular range of devices and a target market. Each combination is optimized for the memory, processing power, and I/O capabilities of a related category of devices. The result is a common Java platform that takes full advantage of each type of device to deliver a rich user experience.

The J2ME platform can be extended by adding various optional packages to a technology stack that includes either CLDC or CDC and an associated profile. Created to address very specific application requirements, optional packages offer standard APIs for using both existing and emerging technologies such as database connectivity, wireless messaging, multimedia, Bluetooth, and web services. Because optional packages are modular, developers can avoid carrying the overhead of unnecessary functionality by including only the packages and application actually needs.

- Microsoft NET Compact Framework [14]

The .NET Compact Framework is a hardware-independent environment for running programs on resource-constrained computing devices, encompassing personal data assistants (PDAs) such as the Pocket PC, mobile phones, set-top boxes, automotive computing devices, and custom-designed embedded devices built with the Windows CE .NET operating system.

The .NET Compact Framework is a subset of the .NET Framework class library and also contains classes exclusively designed for it. It inherits the full .NET Framework architecture of the common language runtime and managed code execution. Even the Integrated development environment includes a reach set of available predefined classes, efficient application have to follow principles mentioned.

The .NET Compact Framework provides the following key functionalities:

– Runs programs that are independent of hardware and operating systems.

– Supports common network protocols, and connects seamlessly with XML Web services.

– Provides developers with a model for targeting their applications and components to either a wide range or specific category of devices.

– Provides benefits of design and optimization of limited system resources.

– Obtains optimal performance in generating native code using just-in-time (JIT) compilation.

This documentation assumes a general understanding of the .NET Framework. It focuses on technologies and components of special importance or that are unique to the .NET Compact Framework. Accordingly, this documentation does not repeat topics it has in common with the full .NET Framework. The .NET Compact Framework uses the same class library documentation as the full .NET Framework.

The .NET Compact Framework supports Visual Basic and Visual C++ development. C++ development is not currently supported. Some applications developed before .NET framework is using eMbedded Visual C++ and/or eMbedded Visual Basic.

• Opera AJAX Platform [15]

The Opera Platform™ SDK is a framework for developing and running Web based applications on mobile phones. It consists of three parts: the Application Player, the Application Framework and Applications.

The Application Player is an extended version of the Opera browser, which is currently deployed on millions of mobile phones and personal computers. The Application Player enables Web applications to utilize native functionality in the phone such as messaging, calendar, battery and signal status.

The Application Framework runs on top of the Application Player and provides interaction between installed Web applications. The Application Framework also offers developers predefined UI elements, such as menu systems and dialog boxes, to ease enable fast and easy application development.

Opera Platform applications are Web applications created in established, open standard technologies such as HTML, CSS and JavaScript. They have access to the phone's functionality through the Opera Platform DOM interface (provided by the Application Player) and can communicate with servers using XMLHttpRequest.

• Palm OS Developer Suite [16]

Palm OS Developer Suite combines compilers, debuggers, simulators, PalmSource SDKs (Software Development Kits), and related tools into a comprehensive, integrated development suite that addresses our developers' tool needs. Palm OS Developer Suite simplifies the development of wireless, multimedia, and enterprise applications for Palm OS by providing developers with a single, integrated too chain based on the highly successful, open source Eclipse IDE licensed from IBM. The Palm developer suite include following tools: Palm Simulators; Wizards; Palm OS Protein C/C++ Compiler and Integrated Debugger; Palm OS Resource Editor – visual resource editor that creates and edits XML resource descriptions (XRD) files for Palm OS applications; Update Manager – makes it easy to install new features into development environment without having to perform a complete download and install; only the new features you choose are downloaded and installed; PalmSource™ Installer – builds product installers quickly.

6. Conclusion

Although the functions which the mobile devices offer to eLearners are similar with those of a PC, the experience is different due to design and interface issues – small screen, lack of keyboard, slower Internet connection, and relatively expensive connection time. At the same time there are added functionalities like GPS navigation, ability to distribute the information between several devices designed for different purposes to aim the needs of the learner better, e.g. to target different learning styles [17].

With the specific m-Learning features – portability, location awareness, and ubiquitous access to information the researchers and practitioners have opportunities to get closer to the "anytime, anywhere, for everybody" aims of the ubiquitous e-Learning.

References:

1. Networked and Electronic Media – European Technology Platform. Strategic Research Agenda. Version 3.031. January 2006.
   www.nem-initiative.org
   (last accessed 28 May, 2006).
2. Sharples, M., N. P. Jeffery, et al. (2000). Structured Computer-based Training and Decision Support in the Interpretation of Neuroradiological Images. – International Journal of Medical Informatics, **60(30)**, 2000, 263-28
3. MOBIlearn project, Report of WP 4 "Guidelines for Learning/Teaching/Tutoring in a Mobile Environment", 2003.
   http://www.mobilearn.org
   (last accessed 29 May 2006)
4. Devices Running with Mobile Windows OS.
   http://www.windowsfordevices.com
   (last accessed 30 May, 2006)
5. Devices Running with Symbian O.
   http://www.symbian.com/phones/#A290
   (last accessed 29 February, 2006)
6. Devices Running with Linux OS
   http://linuxdevices.com/articles/AT8728350077.html
   (last accessed 26 May, 2006)
7. Casey, D. u-Learning = e-Learning + m-Learning. – In: Proceedings of e-Learn 2005 World Conference on e-Learning in Corporate, Government, Healthcare & Higher Education, Vancouver BC, Canada, October 24-28th 2005, p. 2864.
8. Trifonova, A., M. Ronchetti. A General Architecture to Support Mobility in Learning. The Fourth IEEE Conference on Advanced Learning Technologies, Joensuu, Finland, 2004, 26-30.
9. Walk about U-Learning project.
   http://walkabout.netcomp.monash.edu.au/walkabout/fit1011/index.html
   (last accessed 30 May, 2006)
10. Justin, Helmig. Texas Instruments, Programming Considerations for Developing Next-Generation Wireless Embedded Applications.
    http://focus.ti.com/pdfs/vf/wireless/designingforwirelessapplications.pdf
    (last accessed 25 May, 2006)
11. SDK Documentation for Windows Mobile-Based Smartphones. Pocket Internet Explorer Overview.
    http://msdn.microsoft.com/library/default.asp?url=/library/en-us/devguidesp/html/sp_designing_web_sites_for_internet_explorer_for_pocket_pc_lmlt.asp

(last accessed 30 May, 2006)
12. Mozilla Firefox Minimo.
http://www.mozilla.org/projects/minimo/
(last accessed 30 May, 2006)
13. Java 2 Platform, Micro Edition (J2ME).
http://java.sun.com/javame/index.jsp
(last accessed 30 May, 2006)
14. Microsoft NET Framework.
http://msdn.microsoft.com/netframework/programming/netcf/default.aspx
(last accessed 30 May, 2006)
15. Opera Platform™, Enabling AJAX applications on mobile phones.
http://www.opera.com/products/mobile/platform/
(last accessed 30 May, 2006)
16. Palm OS Developer Suite.
http://www.palmos.com/dev/tools/dev_suite.html
(last accessed 30 May, 2006)
17. S h k o d r o v a, R., D. D o c h e v. Distance Learning through Mobile Devices – Some Problems and Applications. Cybernetics and Information Technologies, Vol. **6**, 2006, No 2, 54-62.