Secure Architecture for M-Learning Bluetooth Services

Cătălin BOJA, Lorena BĂTĂGAN, Alin ZAMFIROIU
Economic Informatics Department
Academy of Economic Studies, Bucharest, Romania
catalin.boja@ie.ase.ro; lorena.batagan@ie.ase.ro; alinz10@yahoo.com

The paper describes a secure architecture for an M-Learning system that uses Bluetooth wireless networks to offer services for mobile devices. The solution is implemented on the Java ME platform using SATSA-CRYPTO. The access to these services is made possible by using Bluetooth capabilities of a mobile device that is part of a distributed system. The paper analyzes the security aspect of accessing the Bluetooth service from the confidentiality, integrity and authenticity, point of view. There are described the security options implemented by the Bluetooth Protocol, as described in JSR 82, for the Java ME platform. The proposed secure architecture solution authenticates users and their devices, using cryptographic techniques, without using devices IMEI, International Mobile Equipment Identity.
Keywords: *Bluetooth services, M-learning, security, mobile devices, Java ME*

1 Introduction

In a world where distances are growing smaller every day, an important part of our lives, for most teachers and students, are mobile technologies. Now days mobiles offer a wide variety of services to talk and share with other people information, anytime and anywhere. So, we can access information, make photographs, record a course with our mobile and then we can share this to our friends, colleagues or put this to the web.

The efficiently mode to use the resource – mobiles device for deliver information in educational systems are very important. Today, the models for using and developing mobile applications for learning are becoming part of our life.

The mobile learning process is seen as an auxiliary instrument of the traditional education methods [2]. In our days, increasing the efficiency of existing learning methods is made by implementing the new technologies and the mobile devices area in educational environment. At present there is an estimated ratio of 90% between the number of worldwide mobile devices and global population and that the growth rate in EU mobile devices market for 2009 is of 119% [1]. We can say [2] that in theory each person in the world will have the physical resources to be reached by mobile learning process in some way. In developed countries, these rates are greater and studies revealed that young people integrated more rapidly mobile devices in their daily activities.

M-learning has become an attractive target application area for mobile devices because their capabilities to offer resources for a wide range of software solutions for M-Learning:

- Stand-alone applications distributed to students to be used offline as a support;
- Web solutions, [19], based on the mobile device capacity to connect and browse web sites;
- Mobile solutions that use the device connection capabilities based on Bluetooh, WiFi or IrDA to access remote m-learning services.

Bluetooth is a connectivity technology for creating Personal Area Networks (PANs). The Bluetooth specification uses a short range radio standard to implement its communications protocols. Bluetooth provides a simple way of connecting devices in close physical proximity. It can be found in devices such as mobile phones, PDAs, laptops, printers, games consoles and digital camera.

The Bluetooth technology is growing rapidly. The number of Bluetooth devices shipped per year has grown exponential. The majority of these Bluetooth chipsets are used in mobile phones. An interesting aspect is that users are dependent on having mobile phone, and the Bluetooth technology is included in

all new phones. As an increasing number of useful Bluetooth applications become available, many users have Bluetooth devices and are ready to start using Bluetooth where all their Bluetooth devices communicate with one another.
Sun introduced the Java ME (Java Micro Edition) development environment in 1999 to provide a platform programming standard geared towards mobile and embedded devices with limited storage and processing power.
The features of mobile devices are represented by the need of users to have high quality audio, nearly constant network connectivity and to be open to many opportunities, these being achieved by using Java ME and developing creative applications. Java ME can be used for extending the facilities of mobile devices.
Java ME platform runs on many mobile devices, which have installed a Java Virtual Machine. [4] The biggest benefit of using the Java platform for mobile device development is that is possible to produce portable code that can run on multiple platforms. Not always we can implement the complete functionalities of an application to all mobile devices because of wireless device drawbacks in terms of memory, processing power, battery life, display size, and network bandwidth.
Mobile devices come with different form, features and functionality, but often use similar processors and have similar amounts of memory. Therefore configurations were created, defining groups of products based on the available processor power and memory of each device. A configuration outlines the following:

- the Java programming language features supported;
- the JVM features supported;
- the basic Java libraries and Application programming Interfaces (APIs) supported.

In [3] the limited program space requires very compact code for each application. Most handsets already enforce a maximum MIDlet (a Java application framework for the Mobile Information Device Profile that is typically implemented on a Java-enabled cell phone or other embedded device or emulator) size to ensure proper installation and execution of applications on their respective devices. MIDlets are applications, such as games.
The limited amount of heap space may cause problems during runtime.
However, when the hardware problems will be solved complete the mobile devices will come with innovative and greatest challenge solutions as Harold S. Osborne predicted regarding the impact of mobile technologies on our lives [8].

2 Bluetooth and Java ME

The Java APIs for Bluetooth is a Java ME specification for APIs that allow Java MIDlets to use Bluetooth on supporting devices. The specification was developed under the Java Community Process as JSR 82.
Java APIs described in the JSR 82 interface for following Bluetooth Protocols/Profiles:

- SDAP - Service Discovery Application Profile: Provides the ability to find available services to access remote functionalities.
- RFCOMM - Serial Cable Emulation Protocol: Provides a stream-based connectivity between Bluetooth applications.
- HID - Human Interface Device Profile: Provides support for devices such as mouse, joysticks, keyboards, as well as sometimes providing support for simple buttons and indicators on other types of devices.
- *GAP - Generic Access Profile:* Provides the basic building blocks of a Bluetooth application, such as local device, remote device, Bluetooth address, and device discovery.
- GOEP - Generic Object Exchange (OBEX) Profile: Provides support for OBEX protocol, which allows applications to exchange simple objects such as business card data.

The JSR 82 Bluetooth API allows application developers the use of the Bluetooth interface of the device. With this API it is possible to search devices and services, as well as the exchange of information with others devices.

The Java APIs for Bluetooth wireless technology (JABWT) standard, defined by the JSR 82 specification, supports rapid development of Bluetooth applications that are portable, secure, and highly usable. Developers are now able to write Bluetooth applications independent of hardware vendors. Most important, JABWT-compatible applications are portable across various JABWT-equipped devices. Wireless device manufacturers have responded to the JABWT specification by announcing that mobile phones and other products will run JABWT applications.

We can put in evident some aspect about Bluetooth package:

- It is simple, and easy to understand the capability of the Bluetooth protocol.
- Simple text-strings can be easily encoded.
- Easily exchange messages.

Java ME provides several classes to aid developers in adding Bluetooth functionality to their applications. These classes are contained within the javax.bluetooth package. The classes are used to gain information about the Bluetooth capabilities of the device on which the software is running, seek other devices and services, then make and receive connections [6]. The Java APIs for Bluetooth define package javax.obex for the Object Exchange (OBEX) protocol. According to the JSR 82 specification, the underlying Bluetooth system must support a Bluetooth Control Center (BCC), which is a control panel much like the application that allows a user or OEM to define specific values for certain configuration parameters in a stack. Any Bluetooth application has these components: stack initialization, device management, device discovery, service discovery, and communication.

The research tray to introduce the reader to Bluetooth package and to put in evidence a way in which this technology can easily integrated into Java ME.

3. Distributed architecture for Bluetooth services

A Bluetooth network is a wireless network type PAN - Personal Area Network.

When two mobile devices set a Bluetooth connection, one is working in the role of master and the other as slave, with the possibility that any device to function as a link master and the slave in another connection. In a master can connect up to 7 active slave sites, the network is called *piconet*. The role of master of a device does not give any privilege or authority, the master is responsible only for synchronizing devices connected to it. To build such an ad hoc network of PAN type up to eight mobile devices, nodes.

More *piconets* networks communicating with each other can form a *scatternet* network, Figure 1.

Each piconet contains a master node and up to seven slave nodes. A node can be part of one or more piconets [7].

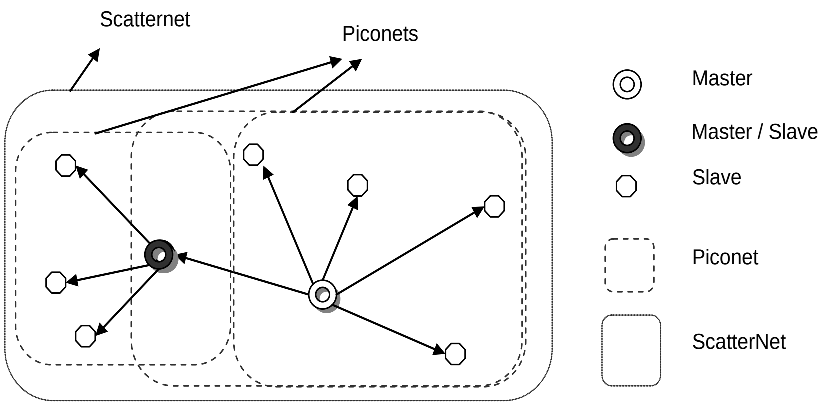

Fig. 1. Scartternet wireless network

The architecture presented by this paper is based on building a connection between two devices. One will function as master and one as slave.

This architecture provides to users of mobile devices a service through which students can report if they are present at classes or workshops. This is a simple scenario that can be extended to more complex services.

This implies the existence of two components client Component and server Component.

The client component is the slave device of the connection and contains a Java application that will run on "Bluetooth API for Java ME” platform.

It will be installed on students' mobile phones, the students having to connect to the existing server in the classroom and to report their presence that day.

In order for mobile devices to run the client application, they must provide support for Java ME and must have implemented Bluetooth technology. At present, all new mobile phones are implemented Bluetooth and Java API for Bluetooth.

According to the specifications in [5], [11], the application will search for Bluetooth-enabled mobile devices in its vicinity, or proximity and display to user the devices list. The user will choose the device that is the server (server component) and then the application will search the services offered by the selected device user via UUID value (Universally Unique Identifiers). Universally Unique Identifiers are 128-bit values that uniquely identify a Bluetooth service. Every service has a UUID [6]. If a service communication protocol RFCOMM (UUID = 0x0003) is found, a service provided by the server component, the application will connect to the service and through it will send a verification code and name of student who make presence. The verification code is used for a better security service. After sending the verification code and name, the application expects a response from the server through which it is confirmed that this was done. After this, the application closes the connection for leaving the server free for other devices.

The component "server" is the master device and contains an application to be installed on a single machine. On this machine client applications will connect.

According to Bluetooth specifications presented in [5] the application creates an RFCOMM protocol communication service, UUID = 0x0003, registers it and then waits to connect any device to the service. After connecting a device, the application will receive the verification code and name of student, seeks the student in the database, and send back the confirmation. After this, it will be waiting for another mobile device to connect.

This typical example of how a wireless connection can be used has the following characteristics and needs that must be met in order to work properly:

- scalability – given by the permanent fluctuations in the number of students present in a class or workshop; it must preserve its functionality either if is only one student present or maybe hundreds;
- reliability – the power to treat each and every single individual as a unique entity, more, as a unique process inside the application, in this way assuring the CIA of security which this topology must comply to: Confidentiality, Integrity and Authenticity;
- availability – is defined outside of the three security characteristics because of its imperative need to be present, without which the Scatternet can’t be formed; if the server application is not working then the slave users can’t connect;

Nonetheless, other kind of hybrid wireless networks can be used for building such collaborative e-learning systems. In [9] is conducted a study regarding the performances of the following interconnection methods:

- overlaid Bluetooth Piconets known as OBP;
- temporary Scaternetts known as TS.

The most important aspects of these methods of interconnection are:

- OBP permits that every Piconet could change its stages and connect information about the other piconets in its proximity; a slave could be a master nod only if it’s disconnected from the previous master;
- TS is actually more entitled for the example provided above because it creates a temporarily scaternett to interconnect

nodes when needed.

A conclusive measurement process is conducted in [9] with regards to the following indicators that can reveal the performance of such a wireless network communication in an e-learning environment:

- throughput vs. speed, rate and time;
- efficiency vs. speed, rate and time;
- probe rate vs. speed;

4 Java ME and Bluetooth security

Bluetooth technology allows different wireless devices to communicate with each other over the air in a scatternet topology. The technology is an industry standard that has continuously evolved allowing applications to transfer data at a throughput up to 24 Mbps with the release of the Bluetooth high speed technology 3.0 [18].

Being an open source standard, Bluetooth is struggling against the underworld community formed by mall intended users which are starting to turn their attention on mobile devices. Until two or three years ago, there was no big interest for mobile devices regarding the attackers because mobile users hadn't anything of value which could be used with mobile devices. Now, when mobile payments and bank transactions catch up the time and sensitive information can be stored on these devices, hackers and virus writers seem to catch the bait.

Applications can use the device Bluetooth connection by implementing JSR 82 API, which defines two packages, [11]:

- *javax.obex* defines the independent OBEX (Object Exchange Protocol) protocol that may be used independently of the Bluetooth API;
- *javax.bluetooth* provides support for secure connections, for registering and discovering services and for establishing connections.

The *javax.bluetooth* package includes classes and methods that secure the Bluetooth connection. Setting parameters for the *Connector.open()* function, [17-18], an application can implement data security from three points of view:

- authentication in order to verify the identity of the remote device; authentication is made by checking a 128-bit shared link key; the key is generated by a PIN code that is shared by both devices; the authentication fails and the connection is not accepted if the PIN code is not the same on both devices;
- encryption for sending encrypted data between the devices; data is protected by symmetric cryptographic algorithms, AES starting with Bluetooth 3.0, that use a 128 bit shared linked key; because the key is generated in the authentication phase, the two stages are dependent;
- authorization in order for a user of the server application to grant access to a specific service by a specific client device; users of server devices have the power to decide if a trusted or untrusted device is allowed or not to access the service.

These methods for controlling security layers on Bluetooth devices are working on the basic Bluetooth protocol stack, some of them being specific to the Bluetooth technology other adopted.

In Serial Port Profile – SPP, the security level of the connection is defined by a single parameter in the *getConnectionURL()* method, named *required Security*, integer value, which can have three possible values:

- NOAUTHENTICATE_NOENCRYPT
- AUTHENTICATE_NOENCRYPT
- AUTHENTICATE_ENCRYPT.

After getting the URL of the remote service the *Connector.open(URL)* will be called. This connection process is managed via RFCOMM layer using SDP for discovering remote devices.

Note, that are other possibilities of Bluetooth connection such as L2CAP – Logical Link Control and Adaptation Protocol which is a Bluetooth specific protocol but the connection process is very similar to RFCOMM.

An important aspect which shouldn't be forgotten in implementing Bluetooth security concerns the consumption level of the battery especially when mobile processors are 100% used for different cryptographic calculations.

A measurement procedure could be used for determining the amount of energy spent on cryptographic processes:

1. charging the device 100%;
2. transmitting one unit of raw data through a Bluetooth connection;
3. measuring the level of battery after this operation, this being BLS – battery level with simple data;
4. charging again the device 100%;
5. transmitting one unit of encrypted data through a Bluetooth connection;
6. measuring the level of battery after this operation, being BLE – battery level with encrypted data;
7. determining the amount of power consumption, APC, used in a encrypted transfer compared to a normal one, APC = BLS – BLE, meaning a ratio of (100-BLE)/(100-BLS)*100% from the normal process.

One of the good parts of using Bluetooth encryption communications for sensitive data is that is very hard to beat even for the most expensive Bluetooth traffic analyzers on the market. If one of the following information [10] is missing, which were used for creating the symmetric key for the encryption process, then the on the fly decryption process will result in a failure:

- slave or client BD address involved;
- the PIN for creating the binding session;
- when the pairing or bonding session is carried out;
- confirmation of a good capture of the pairing or binding session.

In order to support cryptographic services, Java ME platform includes a package, the Security and Trust Services API (SATSA), [1], that is flexible enough to run with many types of cryptographic algorithms and protocols. The SATSA framework has been designed to run on any Java ME-based virtual machine, including the CDC and CLDC virtual machines. This Java standard specification has been defined by the Java Community Process (JCP) in JSR 177, [14].

The API provides interfaces that allow developers to implement secure solutions based on a smart card, the mobile device or a combination of the two. This survey concentrates only on the second solution, using only the mobile device processing unit, because there are other restrictions, legal and technical, that will not allow a smart card solution intended for a wide range of devices.

From all the SATSA packages, the one that does not require a smart card is the SATSA-CRYPTO package. It provides classes for implementing data security architectures based on message digests, digital signatures and symmetric and asymmetric encryption / decryption algorithms. A short description of the package is provided in Table 1, [14-16].

Table 1. SATSA-CRYPTO package.

Package	Classes and Interfaces	Description
java.security	Key PublicKey KeyFactory MessageDigest Signature	Public key encryption using RSA, [11], asymmetric algorithm; SHA-1, [11] message digest computation; SHA1withRSA digital signature;
java.security.spec	EncodedKeySpec X509EncodedKeySpec AlgorithmParameterSpec	Key specifications and algorithm parameter specifications;
javax.crypto	Cipher	Data symmetric encryption / decryption using DES, [11], DES-EDE and AES algorithms in ECB or CBC modes;
javax.crypto.spec	IvParameterSpec SecretKeySpec	Initialization vectors for DES in CBC mode and RSA ciphers Key specifications for DES or Triple DES

In [17] there are described all the symmetric and asymmetric, pubic key, encryption algorithms implemented in the SATSTA-CRYPTO package.

5 General architecture for accessing resources

The mobile device capability to use a Bluetooth connection to access remote resources is an important factor that defines what the system can do:

- give access to resources for hundreds of mobile devices in a client-server architecture; the student devices request services or resources from another device or node;
- allows student devices to make peer-to-peer connections for collaboration or for an efficient distribution of resources;
- eliminates physical barriers as devices connect without a data cable;
- defines fixed access points, using Bluetooth wireless nodes, that promote m-learning services

However, this technology has some draw backs when the m-learning solutions is developed and implemented as an enterprise Bluetooth application:

- the connection is possible only in a 10 meters radius around the Bluetooth node;
- the Java ME application from the master device can handle a maximum of seven active connections;
- if the architecture has more than a master device and users can move between them, there is no session management; if the user is connected to a node then he loses its connection and working sessions when he gets outside the coverage of the master device, despite the fact he gains a connection to the second one;
- the master device becomes a dedicated Bluetooth node with particular technical specifications like battery lifetime, processor or memory, because other capabilities, like phone, display, aren't used.

So, from a technical viewpoint, an m-learning enterprise solution, which is based on Bluetooth connections, requires dedicated nodes that overcome the limitations of a usual Bluetooth device. Figure 2 describes such a solution based on the Ericsson BlipNet, [21], characteristics.

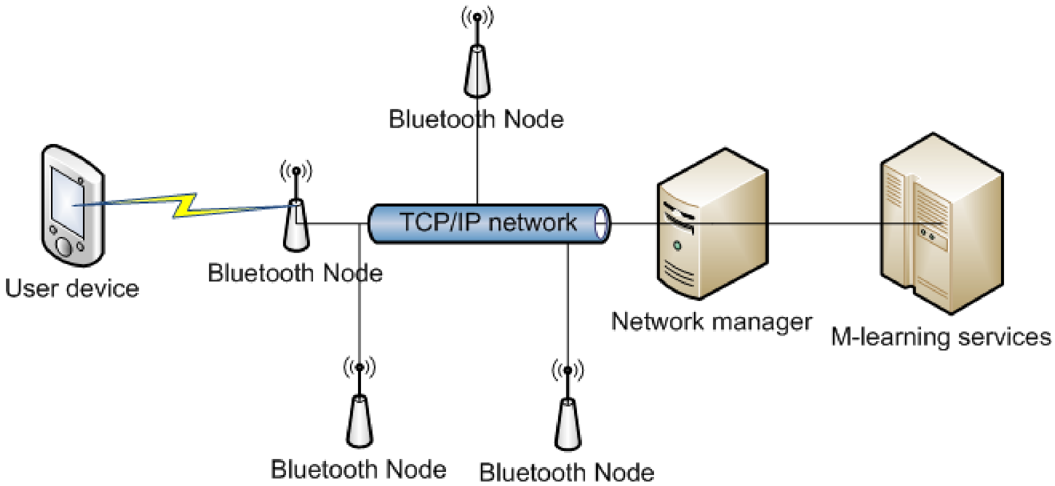

Fig. 2. Physical architecture of Bluetooth nodes for an m-learning technical solution, based on Ericsson BlipNet [10].

Using dedicated Bluetooth nodes as described in [10], the M-learning solution is developed on an infrastructure that will allow the system to be scalable and robust:

- connections greater than 10 meters;
- multiple user connections per node;
- session management for clients that move between nodes.

From the functionality viewpoint, the platform is based on the Service Oriented Architecture. In [20] and [21] it is presented a general three-tier m-learning architecture, described in Figure 3 that has been constructed on an e-learning architecture.

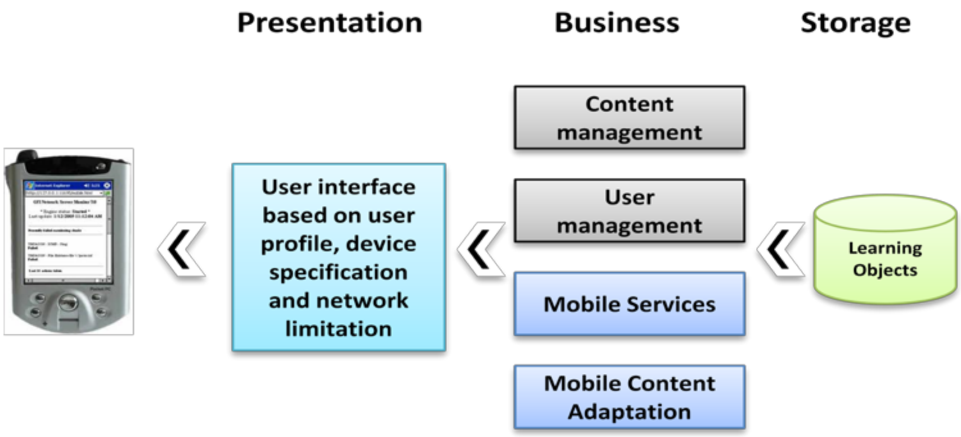

Fig. 3. M-Learning Bluetooth Services Architecture

The m-learning architecture is developed on top of a classic e-learning system. Its particularities are derived from the special characteristics of the mobile device and of the Bluetooth connection.

The architecture is designed to support mobile users using Bluetooth connections, therefore the services interface is specially defined to fit the client devices on low and medium sized devices screen with optimized navigation structure. The architecture must provide user oriented interfaces. The interface is generated based on the user' device specification and also its profile.

Because of the limits imposed by the connection, the system manipulates data, using content resize techniques, in order to achieve minimum data transfer and high interaction experience over available Bluetooth connections.

5 Secure architecture for accessing resources

The proposed secure architecture, described in Figure 4, offers a solution for accessing Bluetooth services in a Piconet wireless network that provides a high level of security. This solution has been implemented using JSR 177 [14].

Each mobile device has a unique identification number, IMEI (International Mobile Equipment Identity), but from a software development point of view, it is difficult to access it on every type of device, because the solution depends on what API is made available by the producer.

The solution takes into consideration:

- a central point for the M-learning system that manages user credentials, like user name, password and Users Unique IDs, UUIDs, used to access Bluetooth services;
- a master device that offers Bluetooth services; to access these services, mobile devices, seen as slaves in Figure 3, use their Bluetooth capabilities;
- there are multiple users with mobile devices; access to the device is secured and each one stores sensitive data in a configuration file or in a data management system; to prevent unwanted users to access the device, a user account is required and secret data is stored encrypted on the device; because its maximum connections limitations, the master device must respond to clients as the requests are made and as fast as possible; in a real scenario, that has multiple clients it required an enterprise solution like the Ericsson BlipNet
- each user has its own authentication credentials and this can be managed from the central point; problems generated by forgotten passwords are managed only by the central point because it is the only one that can generate new encrypted RMS IDs and User Unique IDs;
- MIDlet applications can o store persistent data, with the RMS (Record Management System) system, [12], within a controlled environment, while maintaining a basic system security; RMS offers security by

the way it is implemented; each application defines its RMS space by using a unique ID; because the data location within the device memory is dependent on the RMS ID and is not exposed to an inquiring MIDlet, data is secured, if the ID value is keep secret and is known only by the application;

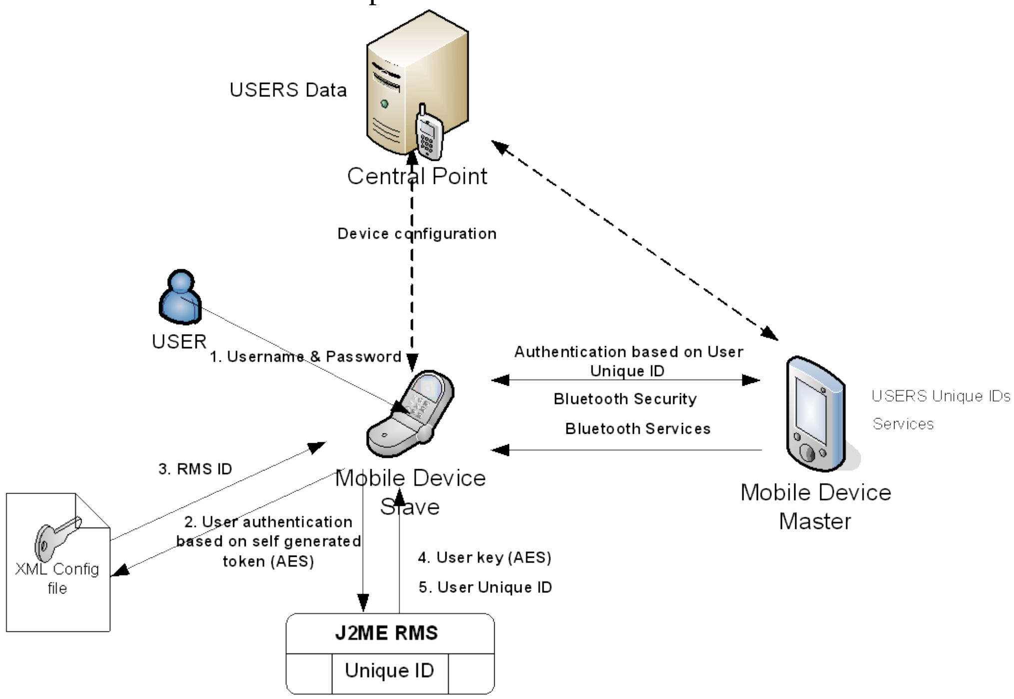

Fig. 4. Secure architecture for accessing Bluetooth services in a mobile environment.

The secure components of the architecture are completing the security objective as they hides clear data from anyone who tries to get it without proper rights:

- the user key (UK) is generated from the username and password; these are unique to each one of the users; this approach does not allow a user to change its account data because it must be done only by the central point; in the case of recreating or changing the password, the central point rewrites the encrypted configuration file and sends it to the device;
- the configuration file is a clear XML or ASCII file containing sensitive data as RMS record store ID (RSID), and a hash value obtained from the user name and password; the file is encrypted using a Base64 encryption key and a symmetric algorithm; this key, called user key (UK), is self generated from the user name and password;
- the hash value from the configuration file is used to verify user credentials; this value is a 160 bit SHA-1 digest generated with the SATSTA-CRYPTO package as in the source code:

```
public class SHA {

    MessageDigest md;

    public SHA()
    {
    try {

        md = MessageDigest.getInstance("SHA-1");
        }
        catch (java.security.NoSuchAlgorithmException e) {
            System.out.println("NoSuchAlgorithmException: "+e.getMessage()); }
    }
```

```
    public byte[] GenerateHash(byte[] input) throws DigestException
    {
            // Read in the cleartext bytes and generate hash value
            md.update(input, 0, input.length);
            byte[] hash = new byte[20];
            int dlength = md.digest(hash, 0, hash.length);
            return hash;
    }
```

- to increase the system security, the password can be padded with a known string before generating the hash; this will reduce the possibility of a brute force attack on the password;
- sensitive data as connection strings, user credentials and User Unique ID (UUID) are stored in RMS using the unique ID for the record store, RSID; local data is encrypted with the UK key and the AES symmetric algorithm; the next code sequence describes the class that implements the encryption and decryption;

```
public class AES {

    Cipher ecipher;
    Cipher dcipher;

    byte[] buf = new byte[8];
    byte[] obuf = new byte[1024];

    public AES(byte[] SecretKey)
    {
    try {
        ecipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
        dcipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
        // CBC requires an initialization vector
        ecipher.init(Cipher.ENCRYPT_MODE,
                new SecretKeySpec(SecretKey, 0, SecretKey.length, "AES"));
        dcipher.init(Cipher.DECRYPT_MODE,
                new SecretKeySpec(SecretKey, 0, SecretKey.length, "AES"));
        }
        catch (java.security.InvalidKeyException e) {
            System.out.println("InvalidKeyException: "+e.getMessage()); }
        catch (javax.crypto.NoSuchPaddingException e) {
            System.out.println("NoSuchPaddingException: "+e.getMessage()); }
        catch (java.security.NoSuchAlgorithmException e) {
            System.out.println("NoSuchAlgorithmException: "+e.getMessage()); }
    }


    public void encrypt(InputStream in, long length, OutputStream out)
    throws ShortBufferException,IllegalBlockSizeException,BadPaddingException{
        try {
            int numRead = 0;
            int numProc = 0;
            long bytesToRead = length;

            while ((numRead = in.read(buf)) >= 0) {
                if (bytesToRead > buf.length)
                    numProc = ecipher.update(buf, 0, numRead, obuf, 0);
                else
                    numProc = ecipher.doFinal(buf, 0, numRead, obuf, 0);
                out.write(obuf, 0, numProc);
                bytesToRead-=numRead;
            }

        }
        catch (java.io.IOException e) { }
    }

   public long decrypt(InputStream in,long length, OutputStream out)
```

```
      throws ShortBufferException, IllegalBlockSizeException,BadPaddingException{
         try {
             int numRead = 0;
             int numProc = 0;
             long bytesToRead = length;

             while ((numRead = in.read(buf)) >= 0) {
                 if (bytesToRead > buf.length)
                     numProc = dcipher.update(buf, 0, numRead, obuf, 0);
                 else
                     numProc = dcipher.doFinal(buf, 0, numRead, obuf, 0);
                 out.write(obuf, 0, numProc);
                 bytesToRead-=numRead;
             }

         } catch (java.io.IOException e) {}
     }
}
```

- to authenticate users, is implemented another protection layer by generating a hash token from the user name and password; this information is also included in the configuration file; at login, the application first authenticates the user by generating its token and comparing it with the one in the configuration file; after this point it will generate the UK and it will decrypt the RSID value;
- knowing the RSID value, the application can access the RMS and get the User Key (UK) and the UUID;
- to communicate with the master device, data transmitted over the Bluetooth network could be encrypted with AES symmetric algorithm using the User Key (UK); this could generate a high processing volume with negative impact on the application performance and the battery lifetime; as seen in previous chapters, the Bluetooth protocol can provide a high level of security with its own encryption phases;
- the UUID is needed to authenticate users devices on the master device; for this scope there are not used users credentials because the system would not allow different users to use same device; in this way, each application provides uniqueness for a device;

The configuration file is managed using open source packages that provide classes and methods used to read XML based configuration files.

This solution has a high degree of security because in the end, only the user knows the key to the system. He starts the process by proving its own password. This information is not in the device. Also, the user does not know any other keys that provide access to the encrypted data. The information is self generated in a cascade type process. If the process is interrupted at some point, the access to clear data is denied.

This secure architecture must also take into consideration attacks that are intended to disrupt the service or to use it to mass distribute messages. Authenticated clients can use the Bluetooth services of the m-learning system to send multicast messages as spam.

6 Conclusions

Software security will become a more important concern in M-Learning systems because more and more data will be stored or can be accessed with a mobile device. Also, more services will be provided by the system through different data connections, like Bluetooth. This requires Confidentiality, Integrity and Authenticity. The proposed solution implements these concepts using the Bluetooth protocol and the cryptographic API of the Java ME platform.

As the security level is increased, so the growing processing volume of the application becomes an overhead for the mobile device processor and battery lifetime. Because performance and quality are important software characteristics mainly for users and the security has same important for system owners, developers must find a balance between them.

Acknowledgements
This work was supported by CNCSIS – UEFISCSU, project number PNII – IDEI 2637/2008, project title: *Project management methodologies for the development of mobile applications in the educational system.* Part of this solution has been presented in [22].

References

[1] P. Pocatilu, C. Boja, "Quality characteristics and metrics related to m-learning process," *Amfiteatru Economic*, pp. 346-354, Vol. 11, no. 26, ISSN 1582-9146
[2] European Commission (January 2010) Mobile use up, consumer prices down: Europe's telecoms sector weathering economic downturn, says Commission report. Available: http://europa.eu/rapid/pressReleasesAction.do?reference =IP/09/473;
[3] Shwetak N. PATEL, Gregory D. ABOWD (March 2010). Beyond Mobile Telephony: Exploring Opportunities for Applications on the Mobile Phone Handset, College of Computing & GVU Center, 801 Atlantic Drive, Atlanta, GA 30332-0280, USA, 2004. Available: http://abstract.cs.washington.edu/~shwetak/papers/Beyond_Mobile_Telephony.pdf
[4] P. Pocatilu, C. Boja, "Survey on Multimedia Technologies for Mobile Learning Applications," *Informatică Economică*, vol. 13, no. 3, 2009, pp. 75-87, ISSN 1453-1305
[5] *** (June 2010). Introduction to Bluetooth technology, Available: http://en.kioskea.net/contents/bluetooth/bluetooth-intro.php3
[6] J. Mchale (June 2010). Developing a Bluetooth-enabled J2ME Game, Information Technology at The University of Glasgow, September, 2007, Availabe: http://www.dcs.gla.ac.uk/~daw/masters-projects/dissertations/McHale.2007.pdf
[7] A. N. Klingsheim (March 2010). J2ME Bluetooth Programming, Department of Informatics University of Bergen, June 2004, Available: http://www.dcs.gla.ac.uk/~daw/masters-projects/dissertations/McHale.2007.pdf
[8] L. Richard, *The mobile connection: the cell phone's impact on society*, Elsevier Printing House, United States of America, 2004, ISBN 1-55860-936-9
[9] S. Jung, A. Chang, M. Gerla, "New bluetooth interconnection methods: Overlaid Bluetooth Piconets (OBP) and Temporary Scatternets (TS) ," *Computer Communications*, Volume 30, Issue 10, 2007, pg. 2258-2273, ISSN 0140-3664
[10] B. Hopkins, A. Ranjith, *Bluetooth for Java*, Apress Printing House, 2003, 352 pg., ISBN 1590590783
[11] Java Community Process (March 2010). Java APIs for Bluetooth Wireless Technology (JSR 82), Specification 1.1.1, 29 July 2008, Available: http://jcp.org/en/jsr/overview
[12] M. De Jode, J. Allin, D. Holland, A. Newman and C Turfus, *Programing Java 2 Micro Edition on Symbian OS*, Wiley, 2004, ISBN 0-470-09223-8
[13] Guérin, R., E. Kim, S. Sarkar – *Bluetooth Technology Key Challenges and Initial Research*, http://www.site.uottawa.ca/~ydaadaa/thesis/GKS.pdf
[14] Java Community Process – *JSR-177 - Security and Trust Services API for J2ME*, http://jcp.org/en/jsr/detail?id=177
[15] C. E. Ortiz (June 2010). The Security and Trust Services API for J2ME, Sun Developer Network (SDN), 2005, Available: developers.sun.com/mobility/apis/articles/satsa1/
[16] *** (July 2010) – SATSA Developer's Guide, SATSA Reference Implementation 1.0, December 2004, Available: http://java.sun.com/j2me/docs/satsa-dg/
[17] A. J. Menezes, P. C. Van Oorschot, S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997 (http://www.cacr.math.uwaterloo.ca/hac/)
[18] *** – *The official Bluetooth website*, http://www.bluetooth.com
[19] A. Muntean, N. Tomai – "A Simple Web Platform Solution for M-Learning", *Informatica Economica*, vol. 14, no 1,

2010, pp. 172 – 181, ISSN 1453-1305
[20] A. Trifonova, M. Ronchetti, “A general architecture for m-learning,” *Journal of Digital Contents*, 2003, paper also available at http://www.trifonova.net
[21] E Chryssafidou, S. Sotiriou, et. al “Developing Tools that Support Effective Mobile and Game Based Learning: The COLLAGE Platform,” *Architectures for Distributed and Complex M-Learning Systems: Applying Intelligent Technologies*, Information Science Reference (an imprint of IGI Global), ISBN 978-1-60566-883-3 (ebook)
[22] C. Boja, L. Batagan, M. Doinea, A. Zamfiroiu, “Secure Bluetooth Services in an M-Learning Environment,” *The Proceedings of the 9th WSEAS International Conference on SOFTWARE NGINEERING, PARALLEL and DISTRIBUTED SYSTEMS (SEPADS '10)*, University of Cambridge, Cambridge, UK, February 20-22, 2010

Catalin BOJA is assistant professor at the Economic Informatics Department at the Academy of Economic Studies in Bucharest, Romania. In June 2004 he has graduated the Faculty of Cybernetics, Statistics and Economic Informatics at the Academy of Economic Studies in Bucharest. In March 2006 he has graduated the Informatics Project Management Master program organized by the Academy of Economic Studies of Bucharest. He is a team member in various undergoing university research projects where he applied most of his project management knowledge. Also he has received a type D IPMA certification in project management from Romanian Project Management Association which is partner of the IPMA organization. He is the author of more than 40 journal articles and scientific presentations at conferences. His work focuses on the analysis of data structures, assembler and high level programming languages. He is currently holding a PhD degree on software optimization and on improvement of software applications performance.

Lorena BĂTĂGAN is PhD, Lecturer at The Department of Economic Informatics from Academy of Economic Studies. She has graduated the Faculty of Cybernetics, Statistics and Economic Informatics in 2002. She has become teaching assistant in 2002. She has been assistant professor since 2009. She is university lecturer at faculty of Cybernetics, Statistics and Economic Informatics from Academy of Economic Studies. She holds a PhD degree in Economic Cybernetics and Statistics in 2007. She is the author and co-author of 4 books and over 50 articles in journals and proceedings of national and international conferences, symposiums.