Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 2 / ISSUE 3 / DECEMBER 2007 / 94 - 109
© 2007 Universidad de Talca - Chile

This paper is Available online at
www.jtaer.com

A Survey of Electronic Signature Solutions in Mobile Devices

Antonio Ruiz-Martínez[1], Daniel Sánchez-Martínez[2], María Martínez-Montesinos[3] and Antonio F. Gómez-Skarmeta[4]

University of Murcia, Department of Information and Communications Engineering, [1]arm@dif.um.es, [2]dsm@dif.um.es, [3]maria@dif.um.es, [4]skarmeta@dif.um.es

Received 23 January 2007; received in revised form 3 October 2007; accepted 15 October 2007

Abstract

The development of electronic signature in mobile devices is an essential issue for the advance and expansion of the mobile electronic commerce since it provides security and trust in the system. E-signatures provide security for the transactions with authenticity and integrity characteristics that make non-repudiation of the transactions possible.

In recent years, different technologies and infrastructures have been developed with the aim of implementing mobile signature processes. Some are based on the SIM card. Others work over the middleware of the mobile device and cryptographic providers. Finally, there are already some frameworks which are independent of specific mobile device technologies and make mobile signatures available to application providers. Therefore, there is a great range of possibilities. In this paper we review the different solutions to date to provide electronic signature in mobile devices (SMS signature, SATK, WIM, USAT-i, SATSA, Mobile signature service, etc). We will comment on the most important goals of each solution and analyse the advantages and disadvantages. From this analysis we will obtain a global view of the current and future tendencies of mobile signature and thus help to provide mobile signature solutions.

Key words: electronic signature, non repudiation, qualified signature, mobile signature, SIM card, Java ME, signature services, mobile devices, mobile commerce

1 Introduction

Nowadays, the use of mobile handsets is widespread. In fact, according to some information provided by operators or organizations such as the ITU [20], mobile handsets have reached a significant penetration rate in many countries such as Luxemburg (164%), Italy (128%), Hong Kong (117%), Spain (109%), Chile (74%), Argentina (64%), and so on. That is, in many countries, almost everybody has a mobile handset (mobile phones, Personal Digital Assistants (PDA)…). This is due to the fact that these devices growth continually incorporate new advanced features related to communication and entertainment, for example, downloading of information, e-mail, instant messaging, video telephony, etc. Furthermore, during these last years, the computation and storage capabilities offered by these handsets have been improved considerably in order to provide these advanced features. Thus, as a consequence of its growth and to these advanced features, today the development of new services for these mobile devices constitutes one of the most important business markets because any service developed could, potentially, be offered to any person in the world.

At the same time, the mobile network infrastructure has been improved and, along with GSM (first generation in mobile technology -1G-), other technologies such as CDMA and W-CDMA (UTMS) have appeared. They constitute what is known as the third generation technology or 3G. These new technologies allow the simultaneous transmission of both voice data and non-voice data up to 2 Mbps. Currently, according to the ITU report [20], there are already 61.5 million subscribers worldwide with at least 256 kbps.

The improvement in both mobile devices and network capabilities leads us to think about the development of the advanced services that we have mentioned above: e-mail, instant messaging, video telephony, and so on. Thus, our mobile devices mean that we are able to be “always on” everywhere. Needless to say, one of the fields that can take advantage of these features is electronic commerce, which in this mobile scenario, has been named mobile commerce. The aim of mobile commerce is to allow users to do business and commerce from their mobile phones, for example, payments (named m-payments [17], [24]), location-based services [15], banking transactions, and so on. Basically, m-commerce consists of developing new applications and many of the services already offered in the electronic commerce, but now, taking advantage of the mobility features that the user has with their mobile handsets. Therefore, m-commerce is a promising field with many potential users willing to consume these services.

Like in the electronic commerce, in m-commerce, non-repudiation services are the basis to develop the secure mobile commerce. Thus, non-repudiation services provide the evidence to prove that some party participated in a transaction [16], [22]. In this kind of services, electronic signature is the foundation that allows its development making the creation of evidences associated to a transaction and a user possible.

In the digital world, electronic signature aims to become equivalent to handwritten signatures. An **electronic signature (e-signature)** is obtained by applying a series of cryptographic operations on the document to sign. Usually, these operations are based on the use of asymmetric cryptography and hash functions. The purpose of an electronic signature is to guarantee authentication and integrity in the information signed. For this reason an electronic signature satisfies three properties. Firstly, it is bound to a document or message and it is not valid for any other message or document. Secondly, it is associated to the signer’s identity and only that signer can generate it. Thirdly, it is publicly verifiable, and therefore it is possible to detect any later change in the signed data. In electronic signature, the element that is used to identify a signer is the digital certificate. In the digital world, this certificate is equivalent to a personal identification document (identity card, passport, etc) and it associates a key to an identity. Thus, the certificate can be used to verify that the key used to sign a document or message belongs to an identity.

This e-signature in some circumstances may be even legally equivalent to handwritten signatures and therefore, the information signed could be used in legal proceedings. In Europe, the conditions which an e-signature should fulfil to be legally equivalent to a handwritten signature are established in the in the directive 1999/93/EC of the European Parliament and the Council [12]. These conditions establish some minimum security levels in the signature processes and could be also extrapolated to other countries or regions. This directive makes a distinction between e-signatures, advanced signatures and qualified signatures. Thus, according to this directive [29], an e-signature is considered advanced if it complies with four requirements. First, “it is uniquely linked to the signature”. Second, “it is capable of identifying the signature”. Third, “it is created using means that the signatory can maintain under his sole control”. Finally, “it is linked to the data to which it relates in such manner that any subsequent change of the data is detectable”. With these conditions in mind an electronic signature is, basically, considered *advanced*, if it was generated using a *Secure Signature Creation Device* (SSCD). More discussion about this topic can be found in [29], [30]. The requirements that these devices should comply with are defined in [11]. Currently, some of the devices that satisfy these requirements are some cryptographic smart cards [10] that meet to the EAL 4+ specification [11]. Furthermore, in this directive it is also established that advanced electronic signatures which are based on a qualified certificate (named *qualified signatures*) and which are created by a secure-signature-creation device can be regarded as legally equivalent to handwritten signatures. A certificate is considered qualified if a qualified certificate service provider issues it. The provider is named qualified if it satisfies some requirements for the security of the certificate’s lifecycle as well as if it is liable for damage caused to any entity whom relies on that certificate.

In the first generation of mobile devices it was impossible to think of generating these advanced signatures due to the limited cryptographic and computational capabilities of these devices. However, currently the mobile devices are able to generate electronic signatures based on asymmetric cryptography or even based on elliptic curve cryptography. As a consequence of these improvements, several solutions have appeared to provide e-signature in these devices. The present paper presents a survey of the different technologies that have appeared over the years to provide these signatures. The survey describes them and analyzes them from the security point of view as well as examining whether the solution evaluated can be considered equivalent to a handwritten signature, that is, if it is a qualified signature as was previously defined. Thus, this survey will allow us to know what the most mature technologies are and develop non-repudiation-based applications or services to the mobile commerce. Furthermore, the survey allows the developer of mobile commerce solutions to decide, based on the advantages and disadvantages of each technology, which is best suited to develop his/her mobile commerce service. This analysis is summed up at the end of the paper where we compare the different technologies.

The rest of this paper is organized as follows. Section 2 establishes the comparison criteria that are used to analyse the electronic signature solutions for mobile devices. Section 3 classifies the different solutions taking into account different points of view. Then, in section 4 we analyze the technologies based on SIM card. In section 5, we provide an analysis of the technologies based on handset. In section 6, we evaluate some technologies that are a hybrid of the two previous ones. Section 7 describes some solutions based on services that are independent of the mobile handset technology. Thus, in section 8 we present a comparison of the different solutions based on the criteria and classifications established in sections 2 and 3. Finally, we conclude the paper and we introduce some open issues.

2 Comparison criteria

As commented in the introduction, in the digital world, e-signature aims to become equivalent to handwritten signature. E-signature is the basis to make the mobile commerce secure and guarantee the non repudiation of transactions. In mobile commerce and mobile business, in order to provide non repudiation, it is fundamental that this electronic signature can be considered legally equivalent to a handwritten signature. For this purpose, e-signature has to satisfy some requirements that were commented in the introduction. Therefore, the most important comparison criterion between different mobile signature solutions is whether e-signature generated is legally equivalent to handwritten signature. In order to satisfy this criterion, we have to accomplish with other smaller criteria.

These are:

- The solution should be based on asymmetric cryptography or in elliptic curve cryptography (ECC).
- The signature should be generated in a Secure Signature Creation Device (SSCD). As a consequence, the solution should support the generation of private keys and these keys cannot be exported.
- The solution should provide qualified signature. It should be possible to obtain certificates from a qualified certification authority.

Other interesting criteria are:

- The format signature should be according to a standard format like PKCS#1 or PKCS#7/CMS. Thus, the signature could be easily verified by using the different cryptographic suites already existing.
- With the purpose of having a solution that can be used in any mobile device, the e-signature solution should be independent of the operating system.
- A user can have more than one identity. Therefore, it should be interesting that the solution could manage the keys and the certificates of the different user's identities.

3 Classification of the electronic signature solutions

There exists different ways of classifying the electronic signature solutions for mobile devices. We can classify them according to several criteria such as signature platforms, technologies, standards and features supported. Thus, from the point of view of signature platforms used to perform electronic signature processes in mobile devices, we can classify them into four main groups. The first group involves electronic signature solutions based on SIM card (section 4). All these solutions have the common characteristic of achieving the signature process inside the SIM card of the mobile device, through its own cryptographic processor. The second group includes all the solutions based on handheld sets (section 5). The third group covers hybrid solutions (section 6) between the two previous groups. Therefore, this sort of solutions needs collaboration between the SIM card and the mobile devices for performing the different tasks of the e-signature process. The last group (section 7) is focused on some high level services that are independent of the mobile device's specific signature technology from the point of view of the application provider.

Another criterion to classify them is from the point of the technology used. In this case, we have solutions based on: asymmetric cryptography, solutions that are independent of the operating system, solutions based on smart cards, solutions developed in Java and, finally, solutions that make use of elliptic curve cryptography.

It is also interesting to classify the solutions according to the signature standards they support. From this point of view we can find solutions based on advanced signature, qualified signature, SSCD, PKCS#1 and PKCS#7/CMS.

Finally, we can classify them in function of the different features they support such as private key export, private key generation, whether the solution support the management of more than one private key and certificate, whether the manufacturer has the control and, finally, whether the application has the key control.

In the following sections we describe and analyze the different e-signature solutions as regards the first classification introduced in this section. We consider this classification is the most generic and interesting way (from the point of view of an application provider) to study them. These solutions are analyzed throughout this survey with the aim of exploring whether they satisfy the criteria established in the previous section. In the next sections, each solution is examined taking into account those criteria. After that, in spite of using the first classification to describe the solutions, we compare the different solutions taking into account all the classifications introduced in this section (see section 8).

4 Technologies based on SIM card

Nowadays SIM cards are multi-application cards, in which different services are working at the same time (figure 1), and the central point of the mobile communications. Each one of these applications has its own functionality and some of them offer electronic signature support. A SIM card usually has two different kinds of memory. In ROM memory you can find the physical layer that includes the SIM card operative system, the memory management and the input/output interface. Next, on top of it, you can see the JavaCard Virtual Machine that interprets the applications [28], the card manager that manages the life cycle of each application, the SIM Toolkit Security that adds security headers to the short messages and different APIs for developers. In this memory zone, you can also see the GSM application that is stamped by the manufacturer and cannot be removed from the SIM card.

In EEPROM memory you find the different applications. This memory zone can be modified in the life cycle of the SIM card by the card manager. The GSM application controls communication over GSM networks and stores the GSM files inside EEPROM memory. These files contain the GSM keys, the address book, the short messages, and so on. The USAT applets are applications developed with SIM Application Toolkit technology, like USAT interpreter. The WIM application enables the possibility of performing cryptographic operations inside the SIM card. In this section we are going to describe and analyze these different technologies that facilitate the performing of electronic signature processes on a SIM card application.

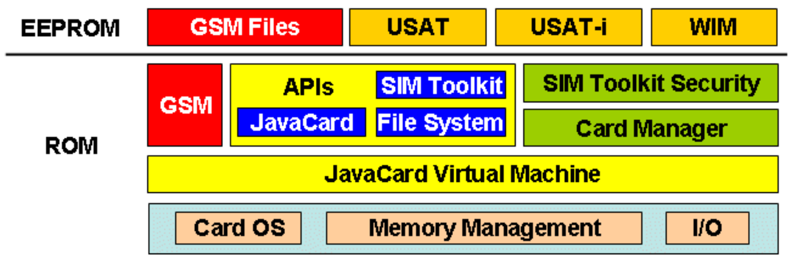

Figure 1: Multi-application SIM card architecture

4.1 SMS

The Short Message Service (SMS) [1] is a service available on most digital mobile devices that provides the sending of text messages (up to 140 bytes) between different applications and users. This service defines the possibility of authenticating and ciphering the communication through the use of security packets. These packets are special headers, called “security header”, which are prepended to the application or user data. The sending application prepares the data and forwards it to the sending entity of the SIM card, which adds the header to the data, packs it and sends it as an SMS to another mobile device. The receiving entity of the destination SIM card receives the SMS and unpacks it according to the security parameters indicated in the security header.

From the electronic signature point of view, the security header contains three important elements: the Security Parameter Indicator (SPI), the Key Identifier (KID) and the Digital Signature (DS). The SPI encodes the kind of cryptographic operation in two bytes, the KID signals the key and algorithm used in one byte, and DS contains the electronic signature of the data in a variable number of bytes. The first four bits of KID byte encode the digital signature algorithm between DES and triple-DES options. Thus, the electronic signature in mobile devices is based on the use of Message Authentication Codes (MACs). The last four bits of KID byte indicate the key used in the digital signature process. The number of keys depends on the manufacturer implementation.

This mechanism was the first approach to provide e-signature by means of the specifications for SIM cards, but it has several important disadvantages. First is it only supports symmetric cryptography for performing e-signature processes. Another important problem is the dependence of the card manufacturer on the creation and stamp of the symmetric keys. Moreover, this mechanism cannot perform an “advanced signature” or a “qualified signature” [12] from a legal point of view since we are not using cryptographic keys that are under the sole control of the end-user. Finally, SMS has a very small bandwidth that only allows small pieces of text to be sent in one message. This disadvantage is overcome by the Multimedia Message Service (MMS) [1], which allows the transport of electronic signatures as well as their certificates inside its bigger payload through a GPRS or UMTS [1] connection. Another important difference of MMS is that it is not supported from the SIM card perspective, but from the handheld.

4.2 SAT - USAT

The SIM Application Toolkit technology (SAT) [1] defines a complete set of commands and events between a GSM handset and a 2G SIM card. This communication interface facilitates the development of new services based on SIM cards [21], [25], which are independent of mobile devices or card manufacturers. These applications, which are allocated on SIM cards, are able to show different menu items in the display of the handheld for interacting with the user, and can begin communication processes through the device, as the establishment of a phone call or the sending of a short message. As you can see in the figure below, these services exchange data with the GSM network through the Short Message Service. This figure shows a common scenario of an incoming event towards the SIM card and a proactive card response towards the handheld. At present, the SAT technology is extended over a large quantity of mobile devices, and has evolved towards Universal SIM Application Toolkit (USAT) [1], in the third-generation technology of mobile phones, with similar principles and concepts to the previous version. The main improvement of USAT over SAT technology is the possibility of opening HTTP connections with IP devices from a proactive command.

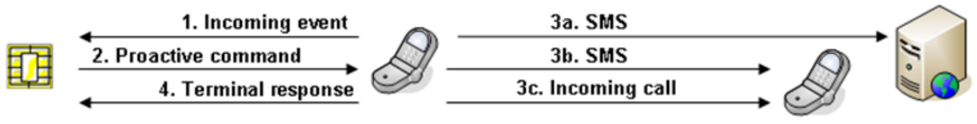

Figure 2: SAT communication model

A SIM Toolkit application is usually a JavaCard applet. JavaCard technology has meant an important advance in the field of the development of applications based on SIM cards. Among its most important advantages are: the secure execution environment for applications, the possibility of deploying different applets on the same card, and the use of the Java language programming as high level language to develop smart card applications. This language has two specific APIs, JavaCard API and SIM Toolkit API. The JavaCard API has a set of different classes and methods that perform both key pair generation (asymmetric and symmetric) and e-signature processes, through the cryptographic capabilities of the SIM card. These applets can be stamped on the card in the manufacture phase, or dynamically downloaded and installed through an “Over The Air” (OTA) server in the life cycle of the card. There are different initiatives about the standardisation of OTA processes like Visa Open Platform and OpenCard Platform [27].

This solution faces some disadvantages. The first involves the downloading of applets through an OTA server, which is a heavy process and implies the fragmentation of the applet into several linked short messages. The installation process of applets is also a sensitive task which involves security risks and performance problems. Another important problem is the slowness of the key generation and the signature process that a SIM Toolkit applet performs, causing a bad experience for the end user. Finally, the signatures generated cannot be “qualified signatures” [12], because the asymmetric keys have no special hardware protection in the SIM card and could be tampered without any visible trace. The only available mechanisms for protecting the keys in this kind of applets are based on software techniques. Therefore, a mobile device that uses the SAT technology to perform an electronic signature process cannot be considered a Secure Signature Creation Device (SSCD) [11].

4.3 WIM

Wireless Identity Module (WIM) [27] is a security specification that defines how to store and manage cryptographic credentials (symmetric and asymmetric keys, user and trusted third parties certificates, and authentication objects like Personal Identification Numbers). It also defines how to perform e-signature processes in a tamper-resistant SIM card. A tamper-resistant module is a device that has certain physical hardware protection to make the extraction of the information contained in the module unfeasible. The WIM standard is based on PKCS#15 which enables a flexible information format on a cryptographic token. The WIM is defined as an independent smart card application, like GSM or SAT applets. Therefore, it can be used to perform cryptographic operations from different protocols from the handheld, like TLS or S/MIME, and different applications of the SIM card, like proprietary SAT applets or USAT-i.

A SIM card with a WIM application inside can be considered a SSCD because the key pairs are generated inside the card, the signature processes are performed by the WIM application, and the private keys never go outside the card. For all these reasons, an electronic signature obtained through a WIM could be accredited as a "qualified signature". Therefore, this technology should be considered as one of the basis in the development of electronic signature applications as regards the generation and management of keys. At present the manufacturers are including this application in SIM cards as a standard of electronic signature.

4.4 USAT-i

The Universal SIM Application Toolkit – Interpreter (USAT-i) [1] is a new SAT application of the 3G SIM cards. This technology is alongside the other standard 3G SIM card applications, such as GSM, USAT or WIM applets. The USAT-i application is a byte-codes interpreter which works as a small browser inside the SIM card. The USAT-i browser receives these byte-codes inside linked short messages, processes them and finally deletes them. This technology is supported over a network infrastructure (figure 3) with two main elements, an application server and a gateway. The application server hosts pages written in a subset of Wireless Mark-up Language (WML) [27] which can contain tags to define a user interface or a call to SAT commands. The gateway packs and transforms these pages until getting byte-codes, which are usually sent to the USAT-i application of the handheld SIM card through linked short messages. These byte-codes are optimized for small bandwidth and are ready to be interpreted inside the USAT-i application. The short messages are not the only way to communicate the gateway and the SIM cards. The 3G networks offer other protocols like HTTP or TCP/IP, but their availability depends of the network operator infrastructure. At present SMS is the most extended transport protocol.

The 3GPP [1] defines that the USAT-i browser of a SIM card can be extended by each manufacturer through plug-ins. A USAT-i plug-in is a JavaCard applet which implements a shareable interface and is installed on the SIM card in the manufacture phase, allowing USAT-i browser to invoke the plug-in commands. When a manufacturer supports plug-ins for the USAT-i browser of a SIM/WIM card usually offers PKI plug-ins. These special plug-ins are able to work with the WIM application of the card for performing cryptographic processes. The main PKI plug-ins are the Asymmetric Decryption plug-in (AD) for asymmetric decryption, the PKCS#7 plug-in (P7) for asymmetric signature of plain text with What-You-See-Is-What-You-Sign (WYSIWYS) interface, and the Fingerprint plug-in (FP) for asymmetric signature of byte array in PKCS#1 format. These last two plug-ins provide a feasible mechanism for performing electronic signature processes from the USAT-i browser inside a SIM/WIM card.

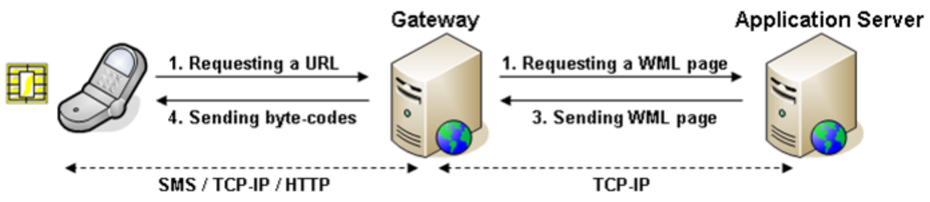

Figure 3: USAT-i infrastructure

At present there are two main implementations of USAT-i technology, the S@T Browser of SIMAlliance and the Wireless Internet Browser (WIB) of SmartTrust. Both of them are based on the 3GPP specifications and add proprietary extensions to the WML subset for application developers and to the gateways. The messages sent between the different parties are shown in figure 3.

The main benefit of USAT-i browser on USAT applications is that the installation of heavy applets "over the air" is not necessary. The applications are packed into byte-codes that are sent to the SIM card, interpreted by it and easily removed. The second advantage is the integration with the WIM application through the PKI plug-ins. Therefore, the private keys are never compromised, and the electronic signatures could be considered "qualified signatures". Finally, like USAT technology, USAT-i browser does not need any software in the device side and is able to communicate with the GSM network directly via short messages. On the other hand, this technology is supported over an additional network infrastructure which must be provided by the network operator. Therefore, this solution requires some investments by the network operator. Additionally, in order to offer electronic solutions based on this infrastructure, it would be required that all network operators offered this infrastructure to their users.

5 Technologies based on handheld

The cryptographic capability of mobile devices is ever greater. There are different mobile technologies like Symbian OS, Windows Mobile OS and Java ME that let us perform electronic signature processes on a handheld device. We analyze these technologies below.

5.1 Windows Mobile OS

Windows Mobile Operating System (Windows Mobile OS) is the operating system developed by Microsoft for handheld devices. This OS constitutes the base for the development of two main kinds of platforms: Pocket PC and Smartphone. A Pocket PC, also called PDA (Personal Digital Assistant), is a handheld-sized computer and a Smartphone is more oriented towards mobile phone capability, although it has advanced data functionality too.

Microsoft's cryptographic system basically consists of several components; applications, operating system, and some Cryptographic Service Provides (CSP). Applications communicate with the OS through the cryptographic API, called CryptoAPI (CAPI), and the OS communicates with CSPs through the Cryptographic Service Provider Interface (CryptoSPI) as shown in the figure below:

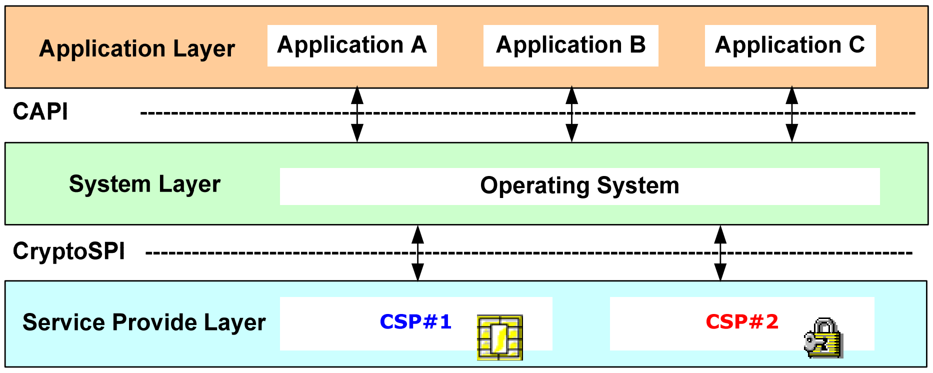

Figure 4: Windows Mobile Security Architecture

As we can see in figure 4, the cryptographic API works with a number of Cryptographic Service Providers (CSPs). A CSP (Cryptographic Service Provider) is an independent module that contains cryptographic implementations of diverse algorithms and standards for authentication, encoding, encryption, key storage, and digital signature. Within the Crypto API points out the following functions: the generation and exchange keys, data encryption and decryption, encoding and decoding certificates, management and enhancement of the security of certificates, creation and verification of digital signatures and computing hash.

The operating system offers mechanisms for the keys generation/storage, certificate management and cryptographic operations through CryptoAPI and three predefined CSPs: RSA Base Provider, RSA Enhanced Provider, and DSS and Diffie-Hellman Cryptographic Provider. In spite of this feature, the use of an external cryptographic library is required in some cases for CSPs development and for improving CryptoAPI functions.

Therefore, the CSPs development allows the easy incorporation of signature creation and key storage safe devices like cryptographic smart cards. There are smart card readers that allow use with Windows Mobile device, such as SDIO or Bluetooth smart card reader. Hence, it is possible to develop a Smart Card CSP that uses an external cryptographic smart card to store private keys and compute digital signatures. This is an alternative method for the secure handling of the private keys. The signature obtained can be in PKCS#1 or CMS/PKCS#7 [18] standard formats. This is, therefore, a good alternative for digital signature solutions in mobile devices.

After mentioning the main features of this solution, we can conclude that Windows Mobile OS provides a robust security infrastructure that allows us to support the whole lifecycle of an e-signature. That is, it provides through a well-defined API all the functionality needed to generate keys and obtain a certificate associated to this key. With this key and its certificate, we are able to make e-signatures using different standard formats. The CryptoAPI also offers functionality to verify e-signatures and certificates. Moreover, it provides an extensible API that allows us to include different secure signature creation devices by means of CSPs. Thus, the solutions provided by this environment can guarantee that they are advanced signatures and therefore, legally equivalent to the handwritten signatures. In spite of the fact that is a good solution, it could be improved by means of some components, like those commented in [2]. Basically, these components are: an advanced certificate validation service that supports both building and validation of a certificate chain, form the certificate to validate as far as a trusted point of trust, the development of some components to support PKCS#12 structures in order to include keys and certificates in the Windows system and applications to create and validate complex PKCS#7 structures, for example, co-signing and timestamping.

5.2 Symbian OS

Symbian Operating System (Symbian OS) is an operating system which has been especially designed for mobile devices. The source code is provided to Symbian OS phone manufacturers (Nokia, Sony Ericsson and Motorola amongst others) and many other firms. This operating system has been developed over these last years and its latest version is SYMBIAN OS v9.3, which was released in July 2006.

As far as security is concerned, among its major valuable characteristics, we can point out that it provides important functionality in aspects related to authentication, data confidentiality and integrity. It is also worth mentioning the secure mechanisms for the installation of applications that allow the authorization and authentication of the installation process of software through the verification of digital signatures. Furthermore, it provides a cryptography module for certificate management, standard cryptography algorithms (such as symmetric/asymmetric ciphers), hash functions, key generation and random number generation. These functions cannot be used directly, but are used by other security modules like the certificate management module.

In the figure 5, we can see that the security architecture of Symbian OS mainly consists of two high level components. One is the certificate management, whose main purpose is to provide storage and retrieval of certificates, assignment of trust status to a certificate on an application by application basis, certificate chain construction and validation, and verification of trust of a certificate. The other component is a cryptography element, whose purpose has been mentioned previously.

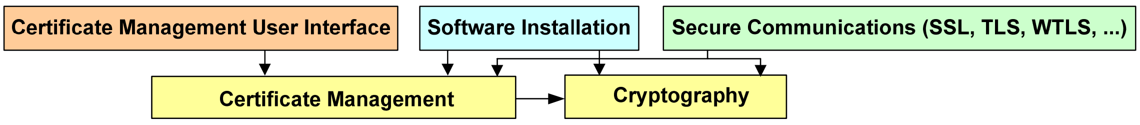
Certificate Management User Interface
Software Installation
Secure Communications (SSL, TLS, WTLS, ...)
Certificate Management
Cryptography

Figure 5: Symbian OS Security Architecture

These modules are the basis of a number of higher level components, which include a User Interface that provides certificate management, Software installation (authentication/digital signatures) and Secure communications (SSL/TLS, WTLS, IPSec, etc.). In a similar way to Windows Mobile OS, the Symbian OS defines a cryptographic token framework that enables licensees to integrate cryptographic devices, such as WIMs.

In spite of the fact that Symbian OS has mechanisms to perform electronic signature, it is difficult to use this option because this functionality is provided only to Symbian OS phone manufacturers and partners. Thus, the access to the elements needed to develop applications that use cryptographic capabilities is restricted. Therefore, the number of applications providers willing to adopt this system is more reduced.

5.3 Java ME

Java Platform, Mobile Edition (Java ME), previously known as Java 2 Platform, Micro Edition or J2ME, is a technology for developing Java applications for mobile devices such as cellular phones and personal digital assistants (PDAs). Java ME consists of programming specifications and a special virtual machine that allows a Java ME-encoded program to run in the mobile device. The Java ME services are based on local programs (MIDlets) that the user automatically downloads into the device. These MIDlets can be executed locally in the device or can maintain a client-server session. There are two configurations for the Java ME: Connected Limited Device Configuration (CLDC) and Connected Device Configuration (CDC). CLDC defines a set of programming interfaces and a Java Virtual Machine (JVM), the K Virtual Machine (KVM), for small devices (for example, mobile phones, PDAs, etc). CDC extends CLDC functionality and defines a new JVM, the C Virtual Machine (CVM), to work on more capable devices (for example, high-end PDAs, TV set-top boxes, embedded devices, etc). While a configuration offers a common functionality for a set of devices, a profile adds an additional layer on a configuration, offering APIs for each specific type of device. There are four standard profiles: Mobile Information Device Profile (MIDP) on CLDC and Personal Basis Profile, Personal Profile, and Foundation Profile on CDC. Next figure depicts the Java ME architecture:

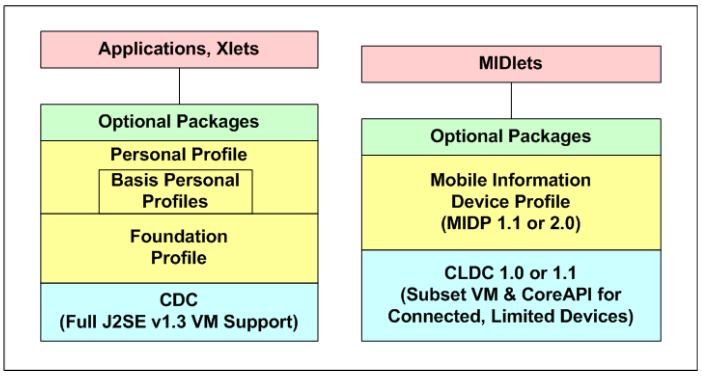
Applications, Xlets
Optional Packages
Personal Profile
Basis Personal Profiles
Foundation Profile
CDC (Full J2SE v1.3 VM Support)
MIDlets
Optional Packages
Mobile Information Device Profile (MIDP 1.1 or 2.0)
CLDC 1.0 or 1.1 (Subset VM & CoreAPI for Connected, Limited Devices)

Figure 6: Java 2 Platform, Micro Edition

Next, we are going to analyze the different mechanisms used to provide the generation the keys, the request of certificates and finally, the storage of the certificates requested. We are also going to mention how the different operations required to these processes are provided as well as how we make e-signatures in mobile devices using this technology. The cryptographic operations in Java ME are provided by means of the incorporation of cryptographic libraries according to a set of standardized APIs defined in Java in the Java Cryptographic Architecture (JCA). Then, we analyze the available Java libraries to provide cryptographic functions to the Java ME devices:

- **Bouncy Castle**. This is a free cryptographic API that provides methods for the most important cryptographic operations: importing both keys and certificates (PKCS#12, PKCS#8 and PKCS#7 standards compliant), generating both keys and certificate signing request (PKCS#10), making digital signature and encryption. Particularly, there is a light version for JAVA ME that can be used with an MIDP 1.0 configuration or later.

- **IAIK Micro Edition**. This is a light cryptographic API, whose main characteristics are symmetric/asymmetric encryption, keys and certificates management. It is compatible with all JAVA ME profiles (MIDP, Personal, etc) and supports PKCS#12, PKCS#8, PKCS#7, X.509, ASN.1 BER/DER encoding, and Base64 complaint standards and SHA-1, MD2, MD5, HMAC, RSA, DSA, DES, 3DES, AES, RC2, RC4 and IDEA algorithms. It is not free.

- **Phaos Micro Foundation**. Cryptographic API that can be used with the CLDC/CDC configurations and MIDP, Personal, Foundation, Insignia Jeode VMs, Compaq IPAQ, DSharp Zaurus and Nokia 9200 profiles. Its main cryptographic characteristics are: AES, 3DES, DES, RC4, RC2 and RSA encryption, DSA and RSA signature, MD5 and SHA1 fingerprints, PKCS#12, PKCS#8, PKCS#5, and certificates x509v3 standard compliant. However, this library is not free.

- **NTRU Neo for Java**. This library contains a minimum set of algorithms specifically designed for devices with limited resources and functionality. It has symmetric (AES) and asymmetric (NTRU, his own algorithm) encryption and fingerprint (SHA1). Furthermore, it could be setup in CLDC/CDC configurations. However, it is not free.

These libraries provide the different methods to perform the cryptographic operations that require the execution of the cryptographic algorithms. However, apart form this functionality we need to provide the cryptographic material or parameters to use in these cryptographic operations, that is, keys, certificates, credentials, etc. Therefore, we also need some mechanisms to provide storage and management of the cryptographic material in the device [8]. The following list shows a series of possible methods to supply this cryptographic information (keys and certificates) to the mobile device:

- Generation of private keys inside the mobile device. Private keys would be generated inside the terminal. Then, the certificate signing request (CSR or PKCS#10) is generated and sent to a registration authority (RA) via HTTP. Finally, the issued certificate is retrieved in the device. This functionality could be provided, for example, by the Bouncy Castle library.

- Through OS file system. Keys and certificates are inserted in the device through the available mechanisms to file management (for example, pc-device synchronization). In this way, the JSR-75 (File Connection) API is required to access the OS file system later.

- Using a communication port in the mobile device. In this case, MIDP 2.0 is required and the virtual machine must implement the different features related to port communication management (USB, IR and serial).
- Via Bluetooth. The JSR-80 (Bluetooth) API is required.

- Through HTTP. This method is one of the most compatible, because this sort of connection is supported by any MIDP device. Thus, a web server is required where keys and certificates are available.

- Through HTTPS, TLS or TCP. These are specific MIDP 2.0 mechanisms. They are similar to HTTP, but if we use the SSL/TLS protocol, then, a security layer is added that encrypts data communication.

In addition, we need a procedure to store keys and certificates in a Java ME environment.

- Storage in the device using Java ME Record Management System (RMS). MIDP profile provides functionality for persistent data storage, using one or more records. In a record-oriented approach, MIDlets can persistently store data and retrieve it later. The record storage capacity depends on the specific characteristics of each device. In MIDP 1.0 each store is particular to the MIDlet suite that creates it. The MIDP 2.0 specification adds a very useful new capability to the RMS package: it enables one MIDlet suite to share a record store with other MIDlet suites. This would allow the cryptographic information to be shared by several applications.

- Through the device file system. The storage of the cryptographic information in the OS file system depends on whether the virtual machine supports JSR-75 API (File Connection) or not. Hence, if supported, it can access cryptographic information stored in the file system, for example, access to a PKCS#12 file which contains a private key and a certificate.

- Key and Certificate storage using a cryptographic smart card. Another option would be to use a smart card for key and certificate storage. An external smart card can be used since smart card Bluetooth readers for Java ME devices are on the market. The JSR-80 (Bluetooth) API is required for communication management between the mobile device and the Bluetooth reader. Furthermore, we can use internal smart cards such as SIM cards. This alternative requires a Java ME library that allows access to the SIM card. For this purpose, we have the Security and Trust Services API for Java ME (SATSA), which we describe in the next section of this paper.

Now this technology has been analyzed we can conclude that all the necessary elements needed to perform e-signatures processed in Java ME environments are present. The main advantage is that Java ME devices are widely extended. On the one hand, there are cryptographic libraries with the capacity to add functionality to perform e-signature methods with mobile devices, such as encryption and decryption of data and signature signing and verification. On the other hand, these digital developments require a set of repository of keys and certificates to be installed in the device. As mentioned previously, this storage can be done by means of several alternatives: using features of Java ME platforms, for example, RMS or file system, or using cryptographic smart cards. The latest solution is the most appropriate because it allows secure handling of digital credentials - the private key will never leave the smart card. The use of an internal smart card is explained, in section 5.2.

6 Hybrid Technologies

There are a large number of services that are not possible to implement completely in the SIM card side or in the device side. These services usually need to take advantage of the device characteristics (rich user interface and high processing capabilities) and the security of the SIM card. In this section we analyze the main technologies in order to integrate applications in both sides (device and SIM card).

6.1 WMLScript / XHTMLScript

This section describes how the Web applications developed for mobile devices can make use of the cryptographic features of the WIM application in the SIM card. At present, the most extended languages for mobile Web applications are Wireless Markup Language (WML) and eXtensible HiperText Markup Language (XHTML) [27]. Both languages have script libraries for developing asymmetric processes on the client side. One of the libraries of these languages is the Crypto library, which is included in the browsers that support WMLScript (for WML language) or XHTMLScript (for XHTML) and it is equivalent to the use of Javascript in HTML. The Crypto library [27] has only one function called *SignText* which carries out e-signature processes from plain text. When this function is invoked, the WIM application contained in the SIM card is responsible for performing the signature operation and returning that signature according to the PKCS#7/CMS format. In this process, the WIM application uses one of the user's private keys stored inside the SIM card. More details about the use of the WIM application can be found in section 2.

This kind of application is executed in the browser of the mobile device, so the manufacturer has to implement this functionality. The most important benefits derived from this solution are the use of the standard PKCS#7/CMS, the development of signature processes in a Web application and the integration of the WIM application of the SIM card side. This last advantage implies that the private key never leaves the card in the signature process. Therefore, we could consider that this technology provides qualified signatures.

6.2 SATSA

The Security And Trust Services API (SATSA) is a specification for Java ME which provides the possibility of opening a communication channel between a Java ME MIDlet and a "security element" (for example, a SIM card). Through the use of SATSA, also known as JSR-177, a SIM card can perform security processes (like electronic signatures or user's authentication) in Java ME applications (figure 7).

SATSA defines three different ways of communication with the SIM card. The communication mode depends on the type of SIM application. These alternatives are based on three packages; the APDU package for sending APDU commands to USAT applets through ISO 7816 [19] protocol; the JCRMI package for requesting JavaCard RMI objects, and the PKI package for using the WIM application of the SIM card in cryptographic processes like requesting a user's certificate or performing an electronic signature. Thus, the electronic signature processes are carried out through the class *CMSMessageSignatureService* (contained in the *javax.microedition.securityservice* package*)* that uses the CMS [18] standard. When a Java ME MIDlet invokes the methods of this class, a channel is opened with the WIM application of the SIM card in order to perform the cryptographic operation. As we commented before, as a result of the use of the WIM, the user's private key never leaves the security element.

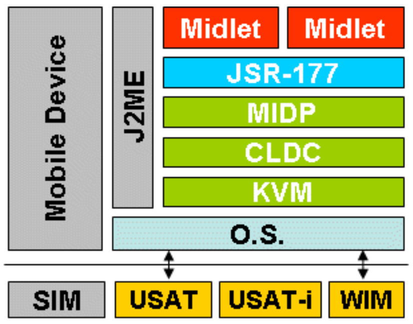

Figure 7: SATSA architecture

Although this proposal provides a high-level of abstraction to the Java ME applications, there are not many mobile devices today that support the SATSA specification. However, the manufacturers are gradually including this capability in the new handhelds. As a conclusion, we can say that SATSA has some important advantages, like the portability of the MIDlets between different devices and the use of a WIM application to performing signature processes based on standards. Therefore, these electronic signatures could be considered "qualified signatures". On the other hand, its main disadvantage is the lack of manufacturer implementations.

7 Independent-handheld solutions

As stated previously electronic signature is an important issue in mobile commerce to provide non-repudiation services. The application providers on the Internet need to provide applications that are based on electronic signature to cope with non-repudiation. This means that an application provider has to develop software that can make use of electronic signature in each different mobile device. However, due to the fact there are a lot of mobile devices, this approach involves a lot of investment, probably, more than the possible return. In order to solve this problem there are two approaches that are commented below. In both approaches the application provider requests a server that the client signs some information.

7.1 Server-based Signatures

Initially the mobile devices had very limited cryptographic capabilities. This fact meant that the mobile devices were not able to sign information using asymmetric cryptography. In order to overcome this situation, the solution proposed was to introduce a server with the responsibility of possessing the information needed to create electronic signatures on behalf of the user. This server was placed in the mobile network infrastructure. Thus, the application provider requests the signatures from the server instead of the user. Next, the server creates the electronic signature form the keys and the certificates stored by the client. In this kind of solutions the mobile phone is used to validate the creation of signature in the server. Thus, the server, before creating a signature, requested the user's authentication from the mobile device. This authentication is based on either Personal Identifier Numbers (PIN) codes or Message Authentication Codes (MAC) using symmetric keys, or a combination of a PIN code and MAC codes, or One-Time Signatures (OTS) which are also based on the use of hash functions. Usually, the symmetric keys used are 3DES keys. A deeper discussion on MAC and OTS can be found in [26]. Some solutions based on server signatures can be found in [4], [5], [6], [9]. In general, they follow the process shown in figure 8.

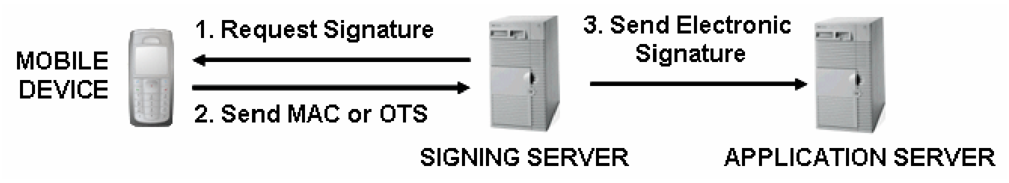

Figure 8: Process of e-signature generation in server-based solution

This is a kind of solution that can be used with devices with very few computation resources [7], [23]. Therefore, its principal advantage is its simplicity as well as an easy development of the solution. This is because any SIM module contained in a mobile device is able to generate and use symmetric keys since these capabilities are required by the GSM standard. It is important to point out that this kind of solutions is not standardised and it depends on the implementation of the signing server. However, the main drawback is that the electronic signatures generated according to this proposal can not be considered advanced electronic signatures according to the European Directive. The reason is that the electronic signature is not in the sole control of the user. Therefore, this kind of signatures is not valid from a legal point of view and it can only be used in a restricted set of services or applications. Another problem is due to the fact it is required that a signing server stores the user's private keys and certificates. This server should be provided by the mobile network operator or by a trusted third party because it contains very important information. Additionally, this signing server should provide several interfaces in order to set user's keys and certificates, generating new keys and certificates, etc. Another important problem is that there is no standardised solution that describes the interface between the signing server and the application provider. Therefore, the application provider should adapt the application to each different signing server. In the next section, we present a similar idea to this concept but providing a solution to the problems commented here.

7.2 Mobile Signature Service

The mobile signature service (MSS) was born to facilitate the development of solutions based on mobile signature for the application providers [32]. The definition of this service comprises several technical reports and specifications published by the ETSI in [13]. Next, we are going to describe how this service works.

In the Mobile Signature Service we can differentiate several roles: end-user, smart card issuer, registration authority (RA), certification authority (CA), Mobile Signature Service Provider (MSSP), application provider (AP), roaming MSSP and contractual management co-ordinator. These roles are depicted in figure 9.

In this proposal the end-user has a mobile device or handset containing a smart card provided by the smart card issuer (usually the mobile operator). This smart card contains a signing application that can provide electronic signature in order to validate the identity of the user. This signing application is protected with a Signing-PIN that the smart card issuer provides to the user. At any moment, the user is able to change the Signing-PIN; he could even be forced to change it after a time or a specific number of signatures with the same PIN.

The signing application is able to generate new keys as well as store the certificates associated to the generated keys. The certificates are obtained from a CA by means of a RA that is responsible for verifying the end-user identity. This application could be invoked by the MSSP (the entity that provides the mobile signature service) in order to obtain an electronic signature from some data. Before the signature is generated, the user has to introduce the Signing-PIN in his/her mobile device in order to validate that it is the user who is validating the electronic signature. Therefore, the electronic signature process is always under the control of the end-user.

Each end-user has to be registered with an MSPP in order to use the MSS. By means of this registration process, the MSSP activates the signing functionality in the user's mobile device. Additionally, the MSSP could provide some value-added services such as timestamping, user's web page to query the transactions, etc. Usually, this MSPP will be the end-user's mobile operator.

In this approach, the application providers do not have to provide any software to the end-user. Now, when application providers require an electronic signature in a transaction, they request it from the MSSP, and the MSSP is responsible for contacting the end-user's signing application to obtain the signature of some information that is provided by the AP when he invokes the MSSP. Thus, the process, in detailed, would be the following:

1. The end-user accesses the services provided by an application provider. Then, in order to access to a specific service or to make a transaction the end-user confirms it.
2. The AP informs the end-user that his mobile signature application is going to be invoked in order to confirm the transaction.
3. The AP sends a signature request to the MSSP in order to indicate that he wants a particular user to sign some data related with the transaction.
4. The MSPP processes the request and sends it to the end-user.
5. The data of the transaction is shown in the mobile device. Next, if the end-user agrees to the transaction, he/she is asked to introduce his/her signing-PIN in order to compute the electronic signature.
6. The electronic signature is returned to the MSSP.
7. The MSSP processes the response, makes some added-value services (if required) and returns it to the AP.
8. The AP processes the response and sends a transaction confirmation to the end-user.
9. The transaction or the service is provided by the AP to the end-user.

These are the steps followed when the end-user and the AP work with the same MSSP. However, in many cases, the AP and the end-user do not have the same provider or the user is not in his/her "home" network. For these cases, there is the role of roaming MSSP. This entity is responsible for forwarding the AP's signature requests to the end-user's home MSSP to communicate with the end-user's signing application. This process is similar to that followed in the GSM voice roaming.

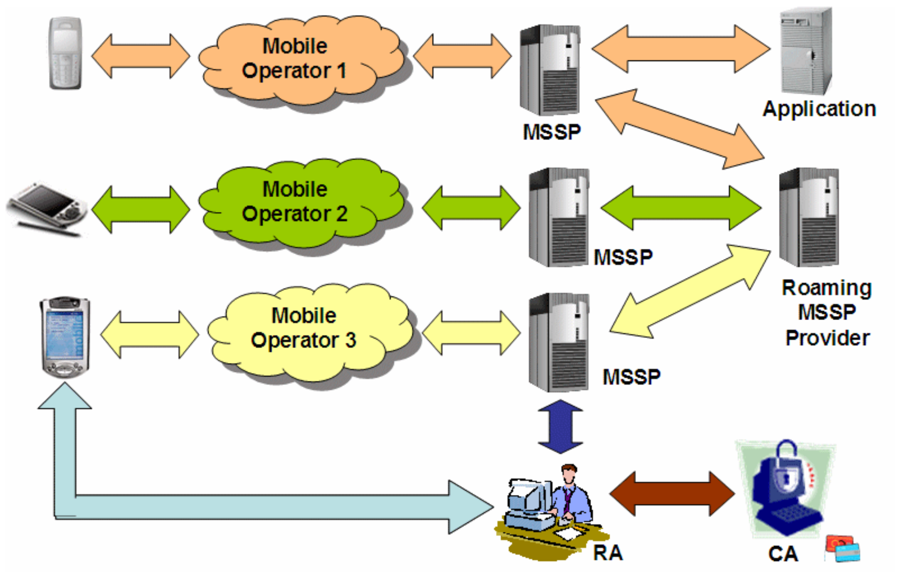

Figure 9: Mobile Signature Service Framework

Nowadays, the M-COMM research group is closed and the results are being moved to different researches groups. At this moment, there are some enterprises, such as Valimo Ltd, that are offering services based on this proposal. The main advantage of this proposal is that the APs do not have to know anything at all about the development of signing-application for specific mobile devices or handsets. This task is a responsibility of the MSSP. Therefore, with this solution no third party will be allowed by the operator to access the SIM card. Another interesting feature is that, although the application provider does not request the electronic signature from the end user, he can specify an advanced aspect related with the electronic signature such as signature format, data to be displayed, key to used, etc. Thus, the service provides flexibility so that the application provider can configure the signature process according to his needs. The main disadvantage of this proposal is the requirement that the AP has to establish an agreement with a MSSP that provides these services. Another problem is that not all the mobile operators offer this solution at the moment.

8 Comparisons

Table 1: Electronic signature platform-based comparison

Signature platforms → / Solutions ↓	SIM card - based	Handheld - based	Hybrids	Services - based
SMS	x			
SATK	x			
WIM	x			
USAT-i	x			
Windows Mobile		x		
Symbian		x		
Java ME		x		
WMLScript			x	
SATSA			x	
Server-based				x
MSS				x

Table 2: Technologies-based comparison

Technologies → / Solutions ↓	Asymmetric Cryptography	OS-Independent device	Smart card	Java	Elliptic Curve Cryptography
SMS		x	x		
SATK	x	x	x	x	
WIM	x	x	x		x
USAT-i	x	x	x		x
Windows Mobile	x				
Symbian	x				
Java ME	x	x_1		x	
WMLScript	x		x		
SATSA	x	x_1	x	x	x
Server-based					
MSS	x	x	x_2	x_2	x_2

Notes:
x_1 – Only if the device SO has Java ME support.
x_2 – It depends on the device side technology.

In this section we are going to compare the abovementioned mobile electronic signature solutions from several points of view: based on signature platforms, technologies implied, standards and features supported. In table 1 we can see a small classification of the solutions in relation to the signature platforms used in each one (SIM card, handheld, hybrids and services), as explained throughout the paper. In table 2 we can notice an interesting summary based on the different technologies implied in each solution. In the one hand, there are only two solutions that do not provide asymmetric capabilities, that is, SMS and Server-based signatures. In the other hand, there are only three solutions that support Java language, and there are only three that are capable of performing ECC cryptography. Another important conclusion is that only SATSA can offer an independent-device operating system solution as well as smart card possibilities, asymmetric cryptography and Java language.

Next, in table 3, the solutions are classified according to the standards supported. First, neither of the solutions is able to produce an advanced signature (and therefore, qualified) and cannot be a SSCD. The second consideration is that only WIM-related solutions are considered SSCD directly (USAT-i, WMLScript and SATSA). All of them need a WIM card inside the device to perform the cryptographic operations. On the other hand, for handheld-based solutions (Windows Mobile, Symbian and Java ME), an external cryptographic smart card for obtaining a SSCD accreditation is necessary. Finally, there are different possibilities to support the different signature standards, such as PKCS#1 or PKCS#7/CMS [18], with the handheld solutions being the most flexible option.

Finally, in table 4, we compare the solutions in function of their features. The first important idea is that all the alternatives support the management of several private keys, usually in function of the smart card size. It is also very important to point out that WIM card and related solutions assure the impossibility of exporting private keys. From a technical point of view, only the SMS solution has a manufacturer key control. In spite of this, the other technologies could be also limited by the manufacturing process.

Table 3: Standards-based comparison

Standards → Solutions ↓	**Advanced signature**	**Qualified signature**	**SSCD**	**PKCS#1**	**PKCS#7/CMS**
SMS					
SATK	x			x	
WIM	x	x	x	x	
USAT-i	x	x	x	x	x
Windows Mobile	x	x_1	x_1	x	x
Symbian	x	x_1	x_1	x	x
Java ME	x	x_1	x_1	x	x
WMLScript	x	x	x		x
SATSA	x	x	x		x
Server-based					
MSS	x	x_2	x_2	x_2	x_2

Notes:
x_1 – Only if the electronic signature is performed by a cryptographic external smart card.
x_2 – It depends on the device side technology.

Table 4: Features-based comparison

Features → Solutions ↓	**Private key export**	**Private key generation**	**More than one private key and certificate**	**Manufacturer key control**	**Application key control**
SMS			x	x	
SATK	x	x	x		x
WIM		x	x		x
USAT-i			x		x
Windows Mobile	x	x	x		x
Symbian	x	x	x		x
Java ME	x	x	x		x
WMLScript			x		x
SATSA		x	x		x
Server-based	x		x		x
MSS	x_1	x_1	x		x

Notes:
x_1 – It depends on the device side technology.

In recent years SATK and handheld (Windows Mobile, Symbian and Java ME) solutions have been the most extended. The main reason for this scenario is the large amount of devices that support them. Furthermore, there are a lot of complete sets of development tools on the market. In spite of this, the current applications have the important disadvantage of not using a SSCD for performing the electronic signature process. At present, there is an increasing number of handhelds with WIM cards inside; therefore this problem is beginning to be solved.

From our point of view, SATSA seems to be the best technology approach for the future. In fact, most devices support Java ME over its native operative system, and SATSA API is being added to new ones. Thus, SATSA shows all the desirable characteristics for performing electronic signature processes like being a SSCD (working over WIM card), making the use of different user credentials possible, supporting of signature standards and the impulse of network operators and partners. Therefore, this solution is ideal in the development of mobile commerce solutions that require the use of electronic signature as equivalent to a handwritten signature. Some of these scenarios of application are: mobile payments as in [17], [24], mobile banking, contract signing in mobile environments [16], etc.

9 Conclusions

Electronic signature is essential to provide non-repudiation services which make secure e-commerce possible. At the moment, the use of e-signature in e-commerce solutions is a mature and broadly extended technology. However, when we move this e-commerce to the mobile world, the m-commerce, we discover that the provision of this technology is not mature enough, despite the fact that this technology does not only suppose important benefits for the users and application providers, but also, as it is analyzed in [14], [30], it could represent important revenues to Mobile Operators and Certification Service Providers. In this paper we have presented the state-of-art of the most important solutions related to the provision of e-signature in mobile devices. For each solution we have commented on the main features as well as advantages and disadvantages. We also made a comparison between the different solutions so as to a developer or a researcher can decide which solutions suit best his purpose. As conclusion, we can say that we have identified which solutions provide a qualified signature, that is, WIM, USAT-i, Windows Mobile, Symbian, JavaME, WMLScript, SATSA and MSS; being SATSA the most interesting technology for this purpose.

As a result of this study, we can also conclude that there are still some open issues related to e-signature in mobile devices. At the moment, all the efforts of the solutions introduced are centred on providing the necessary technology to make the processes of creating and verifying e-signatures. However, up to now, the verification process in the different solutions is only based on checking that the e-signature is valid and matches the sender's certificate. However, in this process of verification, the validity of the certificate must also be checked in order to verify the certificate was not revoked. For this process, we need to build a certificate chain to a trusted point and, next to validate all the certificates in that chain. More details can be found in [3], [31]. Thus, this process must be covered with similar solutions to the mobile service signature framework, but now, applied to verify the status of certificate involved in a transaction. In fact, it could be provided as an extension of that framework. Therefore, it is necessary to improve the different solutions by defining and including a certificate validation service. This service, apart from building the certificate chain and validating it, could store the evidences collected in this process. Thus, these evidences can be used in processes where an electronic signature has to be validated for long periods of time.

Another open issue is the definition of a simplified S/MIME format in order to be included in MMS, because this sort of message is long enough to send data and include e-signatures and certificates. The development of this new format would allow these special MMS to be processed properly by an e-signature application that automatically verifies the e-signatures contained in that message. Additionally, several interfaces should be defined so that the mobile handset responsible for managing the MMS could to communicate with the WIM to sign the MMS. Currently, the MMS are only processed by the mobile handset, unlike SMS that are processed by the SIM application. But, instead of sending the whole message to the WIM, we propose to send the hash instead of the whole MMS because, on the contrary, the processing of the MSS would be slow since the WIM would have to process an important amount of bytes.

Finally, taking into account that in a WIM card we could have several m-commerce applications (Javacard applets) installed, for example, banking applications, e-wallet applications, etc, a standardized interface or API should be defined so that these applications are able to sign information based on keys and certificates stored in WIM. Thus, we would avoid m-commerce application providers having to develop components in order to manage and use keys and certificates, as occurs now. This API could be provided by a new applet that would allow us to list the certificates stored and to choose which one to use to make a cryptographic operation. The access to this applet would be restricted to those applets which the user trusts. At the same time, it is an interesting feature for the end-user because he/she could use the same keys without having to generate new ones for each different application. At this moment applets are, in general, generating and using keys and certificates that are not stored in the WIM, but in the applet itself. Therefore, both the key and the certificate are only for one application. For the use of keys and certificates stored in the WIM, it is necessary that in this process it is clearly defined how these applications access to the WIM information in a secure way and protecting user's interests.

References

[1] 3GPP, The 3rd Generation Partnership Project. [Online]. Available: http://www.3gpp.org/
[2] Mª D. Barnés, D. S. Gómez, A. F. Gómez-Skarmeta, M. Martínez, A. Ruiz, D. Sánchez, An Electronic Signature Infrastructure For Mobile Devices, Procs. Securing Electronic Business Processes, V. Verlag. 2005.
[3] D. Berbecaru and A. Lioy, Towards Simplifying PKI Implementation: Client-Server based Validation of Public Key Certificates, in Proceedings of International Symposium on Signal Processing and Information Technology (ISSPIT). Marrakesh, pp. 277-282, 2002.
[4] K. Bicakci and N. Baykal, A new efficient server assisted signature scheme for pervasive computing, Proceedings of the 1st International Conference on Security in Pervasive Computing (SPC 2003), March 2003.
[5] K. Bicakci and N. Baykal, Design and Performance Evaluation of a Flexible and Efficient Server Assisted Signature Protocol, Procs. of the 8th IEEE International Symposium on Computers and Communication, 2003.
[6] K. Bicakcia and N. Baykalb, Improved server assisted signatures, Journal of Computer Networks, vol. 47, no. 3, pp. 351-366. Elsevier. February 2005.

[7] S. Campbell, Supporting Digital Signatures in Mobile Environments, International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises. June 2003.
[8] J. Dankers, T. Garefalakis, R. Schaffelhofer and T. Wright, Public Key Infrastructure in mobile systems, in IEEE Electronics and Communications Engineering Journal, vol. 14, no. 5, pp. 191-204, 2002.
[9] X. Ding, D. Mazzocchi, and G. Tsudik, Experimenting with server-aided signatures, Proceedings 2002 Network and Distributed Systems Security Symposium (NDSS'02), February 2002.
[10] J. Domingo-Ferrer, J. Posegga, F. Sebe, and V. Torra (Eds.). Journal of Computer Networks, Special Issue on Advances in Smart Cards. Elsevier, vol. 51, June 2007
[11] European Committee For Standardization (CEN), Secure signature-creation devices EAL 4+, in CEN Workshop Agreement (CWA) 14169. March 2004.
[12] Directive 1999/93/EC of the European Parliament and the council of December 1999 on a Community framework for electronic signatures. 1999.
[13] European Telecommunications Standards Institute (ETSI) Specifications, [Online]. Available: http://www.etsi.org/
[14] L. Fritsch, J. Ranke, and H. Rossnagel, Qualified Mobile Electronic Signatures: Possible, but worth a try?, in Proceedings of Information Security Solutions Europe (ISSE) 2003 Conference. Vienna, Austria. Vieweg Verlag.
[15] J. Gao, A. Küpper, Emerging Technologies for Mobile Commerce in Journal of Theoretical and Applied Electronic Commerce Research, vol. 1, no. 2, August 2006.
[16] S. Gürgens, C. Rudolph, and H. Vogt, On the security of fair non-repudiation protocols, in International Journal of Information Security, vol. 4, no. 4, 253-262. Springer-Verlag. 2005.
[17] M. Hassinen, K. Hyppönen, and K. Haataja, An open, PKI-based mobile payment system, in Proceedings Lecture Notes Computer Science, vol. 3995. pp. 86-100, 2006.
[18] R. Housley. Cryptographic Message Syntax (CMS), Request for Comments 3852. July 2004.
[19] ISO/IEC 7816-4:2005: Identification cards -- Integrated circuit cards -- Part 4: Organization, security and commands for interchange. [Online]. Available: http://www.iso.org/
[20] ITU Internet Report 2006: digital.life. [Online]. Available: http://www.itu.int/osg/spu/publications/digitalife/
[21] S. Jain and A. Jain, SIM Application Toolkit – Protocol Conformance and Implementation Challenges, in Proceedings of Second International Conference on Wireless and Mobile Communications. IEEE, July 2006.
[22] S. Kremer, O. Markowitch, and J. Zhou, An Intensive Survey of Non-Repudiation Protocols, in Computer Communication Journal, vol. 25, no. 17, pp. 1606-1621. 2002.
[23] Y. Lei, D. Chen, and Z. Jiang, Generating Digital Signatures on Mobile Devices, in Proceedings 18th International Conference on Advanced Information Networking and Applications (AINA'04). March 2004.
[24] J. Liu, J. Liao, and J. Liu, J. A System Model and Protocol for Mobile Payment, Proceedings of the 2005 IEEE International Conference on e-Business Engineering (ICEBE'05), 2005.
[25] G. Madlmayr, O. Dillinger, J. Langer and C. Schaffer, The benefit of using SIM application toolkit in the context of near field communication applications, in Proceedings International Conference on the Management of Mobile Business (ICMB 2007). IEEE, July 2007.
[26] D. Naor, A. Shenhav, A. Wool, One-Time Signatures Revisited: Practical Fast Signatures Using Fractal Merkle Tree Traversal, in Proccedings IEEE 24th Convention of Electrical and Electronics Engineers in Israel, 2006.
[27] Open Mobile Alliance Specifications. [Online]. Available: http://www.openmobilealliance.org
[28] J. Posegga, Java Smart Cards as a Platform for Electronic Commerce. In International IFIP/GI Working Conference on Trends in Distributed Systems and Electronic Commerce, Hamburg, Verlag, pp. 175-182, 1998.
[29] H. Rossnagel. Mobile Qualified Electronic Signatures and Certification on Demand, in Proceedings of First European PKI Workshop: Research and Applications, EuroPKI 2004, Samos Island, Greece, 2004.
[30] H. Rossnagel and D.Royer, Making Money with Mobile Qualified Electronic Signatures, in Proceedings of Trust, Privacy and Security in Digital Business. Lecture Notes in Computer Science, vol. 3592. August 2005.
[31] A. Ruiz, D. Sánchez, C. I. Marín, and A. F. Gómez. Advanced Certificate Validation Service For Secure Service-Oriented Architectures, in Proceedings Securing Electronic Business Processes. Vieweg Verlag. 2006.
[32] E. Saharanta. Outlining the Key Drivers for Mobile Signature Services and Short Overview of the ETSI — MSS Concept and How the SIM Card and Mobile Network Operator's Infrastructure Plays a Key Role, in Proceedings OASIS Adoption Forum. London, UK, November 2006.
[33] D. Scheuermann, The smart card as a mobile security device, in Electronics & Communication Engineering Journal, vol. 14, no. 5, pp. 205-210, October 2002.