A Web Services Framework for Mobile Payment Services

David McKitterick

A dissertation submitted to the University of Dublin, in partial fulfilment of the requirements for the degree of Master of Science in Computer Science

September 2003

Declaration

I declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

____

David McKitterick

15th September 2003

Permission to lend and/or copy

I agree that Trinity College Library may lend or copy this dissertation upon request.

____

David McKitterick

15th September 2003

Acknowledgements

I would like to thank my supervisor, Jim Dowling, for his enthusiasm and guidance throughout the course of this project. I would also like to thank the members of the PerSL project, in the Distributed Systems Research Group, for their assistance with the development of the real mobile network payment system.

Finally, I would like to thank my parents for all their support throughout my university years and in particular during this project.

Abstract

Next generation mobile services are readily emerging into the mainstream services market and this growth is dependent on mobile technologies and their support infrastructure. 2.5G and 3G mobile technologies presently are beginning to be adopted as a platform for the deployment of communication, business and leisure mobile services. Mobile payment services are one of many necessary support services, which will enable improved development of next generation services. In addition to this necessity for support services, the growth of m-commerce relies vitally on effective payment solutions, provided by mobile payment services.

This dissertation describes a solution for a generic framework for mobile payment services. The issue of diverse, enclosed payment solutions is addressed by presenting an open, extensible and interoperable framework for deploying payment services to the mobile services domain. This framework is based on the distributed architecture offered by Web Services. A Web Services solution provides integration over existing Internet protocols to current mobile payment infrastructures. The framework uses a combination of object oriented design patterns and web based structuring and description styles to allow mobile payment services to be deployed to meet both content provider and mobile user needs.

The development of the Web Services framework was extended to support three independent payment methods, covering mobile network operator billing, credit card payment and reverse sms payment. To further evaluate the framework, two payment systems were implemented to establish the comprehensive relationship between the main actors in a mobile payment system. In addition to Web Services, this implementation integrated such technologies as J2ME, SMS gateway messaging and various networking protocols.

Table of Contents

Table of Figures

Chapter 1

Introduction

The demand for next generation mobile services is increasing as more mobile services are becoming available to the mainstream services market. Additional growth in this area is dependent on the technological and infrastructural support available. 2.5G and 3G mobile technologies presently are beginning to be adopted as a platform for the deployment of communication, business and leisure mobile services. With the technology gradually becoming available, the development and deployment of mobile services is increasingly an attractive market for Internet service providers, content providers and M-Commerce solution providers. For greater acceptance of these services, quality and performance must be ensured through the integration of mobile support services.

An example of necessary support services are mobile payment services, which provide common payment solutions to mobile services. In addition to this necessity for support services, the growth of m-commerce relies vitally on effective payment solutions, provided by mobile payment services. Mobile payment services are currently provided by mobile network operators, financial institutions and independent vendors. Many differences exist between these enclosed proprietary payment solutions. Although, there are a few organisations which were setup to develop a common mechanism for deploying mobile payment services, but as yet no common standard has been adopted for mobile payment services.

This dissertation outlines the background research into mobile technologies and the mobile payment domain, which was a necessary investigation for the compilation of the requirements for a generic, open solution for mobile payment services. A technical overview of existing mobile network technologies and mobile communication services is followed by a description of the generic features of a mobile payment system, including some mobile payment principles and various types of mobile payment.

Following the investigation into the mobile payment domain, a proposed solution for deploying mobile payment services was realised. This solution is based on an open, extensible, interoperable framework for the implementation of various categories of mobile payment services. The distributed architecture of Web Services was selected to provide the extensibility and interoperability requirements of this framework. This Web Services solution allows integration over existing Internet protocols to current mobile payment infrastructures.

This dissertation describes the implementation of the proposed Web Services framework for three different payment approaches, to represent the diverse payment techniques in the mobile payment domain. The realisation of the framework implementations is illustrated through the development of two payment systems. These systems attempt to establish the comprehensive relationship between the main actors in a mobile payment system. The first payment system demonstrates the integration of the Web Services framework into a real mobile network system, using a SMS gateway and Web servers. The second payment system demonstrates a simulated version of the first system, but with a more complex mobile client implementation, with the use of J2ME technology.

1.1 Dissertation Roadmap

Chapter two provides a technical outline of some of the existing mobile network technologies and mobile communication services. Also, various mobile platforms are described for operating systems and programming environments.

Chapter three introduces the mobile payment domain, discussing the various principles within a mobile payment system, such as the actors involved, characterisation of mobile payment, the generic operations for mobile payment and a selection of possible payment scenarios. Mobile payment is then discussed under three areas, providing examples of current systems and infrastructures association with these areas.

The framework design methodology is presented in chapter four. This chapter outlines the extensibility and flexibility aspects of designing a framework with WSDL interfaces.

Chapter five describes the implementation of the Web Services framework for three different payment methods, and the integration of these implementations in the development of two mobile payment systems.

Chapter six discusses various evaluation criteria for this dissertation and how the implemented systems address real mobile networking issues. Following this, chapter seven concludes the dissertation.

Chapter 2

Mobile Technologies

This chapter describes various mobile network technologies, where some are currently in existence on global mobile networks, while the other technologies are gradually becoming adopted by mobile operators. A technical description is outlined for some of the communication services for these network technologies. The chapter concludes with a description of current mobile platforms available for mobile operating systems and application environments.

2.1 Mobile Network Technologies

Mobile network [1] technologies have evolved from analog based systems to digital based systems and from circuit switching to packet switching technologies. This evolution can be described by different generations of mobile technologies, i.e. first-generation (1G), second-generation (2G), 2.5G and third-generation (3G) technologies. Only 1G is based on analog technology. Some of the main standards for each generation technology are:

- 1G: Advance Mobile Phone System (AMPS) in North America, Total Access Communication System (TACS) in UK, Nippon Telegraph & Telephone (NTT) in Japan, Code Division Multiple Access One (CDMAONE).
- 2G: Global System for Mobile Communication (GSM), Code Division Multiple Access 2000 (CDMA2000), High Speed Circuit Switched Data Technology (HSCSD).
- 2.5G: General Packet Radio System (GPRS), Enhanced Data Rate for GSM Evolution (EDGE).
- 3G: Universal Mobile Telephone Standard (UMTS)

2.1.1 GSM

Global System for Mobile Communication [1] is a second generation standard for mobile communication, developed by the European Telecommunications Standards Institute (ETSI) and now currently owned by the Third Generation Partnership Project (3GPP). Operating in the 900 MHz and the 1800 MHz frequency band [2], GSM is the most widespread mobile standard currently in use across Europe and the Asia-Pacific region. Recently it is believed that GSM technology is in use by more than one in ten of the world's population [18] and it is estimated that at the end of 2002 there were 787 million GSM subscribers across the 190 countries of the world. By late 2003 or early 2004, it is forecast that the global GSM market will reach the one billion subscriber mark.

GSM was designed using digital techniques, unlike with previous analog cellular systems like AMPS in the US and TACS in the United Kingdom. The techniques [1] used are a combination of Time Division Multiple Access (TDMA) and Frequency Division Multiple Access (FDMA), which are primarily for voice transmission and control. Since all users must share a limited radio spectrum, these techniques are used to divide the bandwidth among as many users as possible. Also, Space Division Multiple Access is used to provide a system based on a series of base stations each covering a limited area.

FDMA [19] divides the radio frequency into several frequency carriers of 200 Hz, while TDMA enables 8 voice channels in each 200 Hz carrier by dividing each one in time.

GSM Services [19]:

- *Teleservices:* telecommunication services can be divided into bearer services, teleservices, and supplementary services. The most basic teleservice supported by GSM is telephony.
- Data Services:
    - Internet Services: GSM users can send and receive data, at rates up to 9.6K bps, to users on POTS (Plain Old Telephone Service), ISDN, Packet Switched Public Data Networks, and Circuit Switched Public Data Networks.
    - SMS (Short Messaging Service): unique to GSM technology, SMS is a bidirectional service for short alphanumeric (up to 160 bytes) messages. (will be discussed later in more detail)
    - Facsimile: Sending and receiving of fax messages, using a GSM phone and a laptop computer.
    - Secure Corporate LAN Access: secure access to e-mails, faxes, and file transfer via an encrypted link to a corporate LAN.

- *Supplementary Services*: [1] Such services include call forwarding, call barring, caller identification, call waiting and multiparty conversation. These services can be controlled via service applications using a GSM network API, such as those specified by the Parlay Group, allowing application developers to access GSM network capabilities.

GSM technologies are limited due to its low data transmission speed, therefore with the growth in data services the long term future of GSM is uncertain, unless it is changed to offer high bandwidth data services. Also, internet browsing using GSM phones is subject to charging of on-line duration and reconnection is necessary for each browsing session, as opposed to with GPRS (General Packet Radio Service), in which charging is based on the data received or viewed and all time connectivity is available.

2.1.2 HSCSD

High Speed Circuit Switched Data [1, 19] is a circuit switched protocol based on GSM, providing an enhancement of data services. HSCSD enables higher rates by using multiple channels as apposed to single voice channel with GSM. Transmissions rates can be up 57.6 Kbps by using 4 radio channels simultaneously. Typically, HSCSD [1] was directed at mobile PCs rather than smart phones, where a PCMCIA card is used with transmission speeds of 42.3 Kbps downstream and 28.8 Kbps upstream. HSCSD was intended as a temporary substitute for GPRS, to improve the transmission rates of existing mobile data applications.

2.1.3 GPRS

General Packet Radio Service [2] is packet switched wireless protocol providing non-voice value added services that allows information to be sent and received across a mobile telephone network. It is described as a 2.5G technology which supplements Circuit Switched technology such as GSM. Data transmissions speeds of 9.6 kbps to a theoretical maximum speed of up to 171.2 kbps are achievable with GPRS using all eight timeslots at the same time. In addition to higher data rates, GPRS provides users with all time connectivity while only charged for the data viewed or received with a minimal on-line charge. GPRS is an evolutionary step towards 3G technologies, such as EDGE (Enhanced Data GSM Environment) and UMTS (Universal Mobile Telephone Service).

GPRS [1] may be considered as an overlay network on the GSM networks, using the GSM resources to the fullest potential. To enable this, extra network elements are required for this packet based mobile network. Certain hardware elements are added to provide the IP infrastructure needed for packet based services. The SGSN (Serving GPRS Support Node) and GGSN (Gateway GPRS Support node) are the mobile network equivalents of routers and gateways. Other main additions are the upgrading with new software to existing cellular infrastructure.

GPRS [19] only uses its radio resources when users are actually sending or receiving data, therefore the available radio resource can be concurrently shared between several mobile data users, rather than dedicating a radio channel to a single user for a fixed period of time. This efficient use of scarce radio resources means that large numbers of GPRS users can potentially share the same bandwidth and be served from a single cell. GPRS [1] uses the same radio channel as voice calls, a channel that is 200 kHz wide and which carries a raw digital radio stream of 271 kbps. For voice calls this channel is divided into 8 separate data streams, each carrying about 34 kbps. After protocol and error correction overhead, 13 kbps is left for each voice connection or about 14 kbps for data. Packet-switched data can use several channels where as circuit-switched data uses one voice channel. GPRS can combine up to 8 of these channels, and with 14 kbps of data throughput each, the delivered bandwidth can be up to 100 Kbps. Most economical phones will be ones that are limited to 56 kbps, as not all eight voice channels have to be used. A mobile station can request the amount of bandwidth it desires at the time it establishes a data session.

GPRS applications includes Intranet and Internet access, E-Mail, Fax, and Unified messaging, using a single mailbox for all messages, including voice mail, faxes, e-mail, short message service (SMS), and pager messages.

Limitations of GPRS [1, 19]:

- The limited cell capacity during voice and GPRS transmission calls. The use of a bearer for a different type of radio resource, such as SMS, would better utilize the cell capacity.

- Achieving the theoretical maximum GPRS data transmission speed of 172.2 kbps would require a single user taking over all eight timeslots which is unlikely that a network operator will allow all timeslots to be used by a single GPRS user. The bandwidth available to a GPRS user will therefore be severely limited.
- Suboptimal Modulation - GPRS employs a modulation technique called Gaussian minimum-shift keying (GMSK) while the EGDE uses a new modulation technique to allow a much higher bit rate across an air interface, called eight-phase-shift keying (8PSK) modulation. This type of modulation is used for 3G systems, so upgrading to 3G technology seems inevitable for a network operator.
- Transit Delays - GPRS sends data packets through different channels to reach a destination, therefore data corruption or data loss may occur. Data integrity and retransmission capabilities are used to avoid this, but the result is that potential delays can occur.
- No Store and Forward - Unlike SMS technology, GPRS doesn't provide a store and forward mechanism for data transmission, therefore SMS may be need to enable sending and receiving of short messages.

2.1.4 EDGE

Enhanced Data for Global Evolution [2] is a higher bandwidth version of GPRS permitting transmission speeds of up to 384 Kbps. It is compatible with the GSM protocol, but it requires higher quality radio signals to reach the increased speed. Deploying EDGE will allow mobile network operators to offer high-speed, mobile multimedia applications. It allows a migration path from GPRS to UMTS, because the modulation changes that will be necessary for UMTS at a later stage will already be implemented. The opportunity window for EDGE may be very short, unless major delays occur during UMTS deployment.

2.1.4 3G

3rd Generation [2] is the generic term for the next big step in mobile technology development. The formal standard for 3G is the IMT-2000 (International Mobile Telecommunications 2000). There are three optional modes as part of the 3G standard. W-CDMA (Wireless Code Division Multiple Access) is for Europe and for the Asian GSM countries, CDMA (Code Division Multiple Access) is for North America, and then TDD/CDMA (Time Division Duplex/CDMA) for China.

2.1.5 UMTS

Universal Mobile Telephone System [1] is designed to provide for 3G mobile data services. Realistic expectations suggest a maximum capacity in metropolitan areas of 384 Kbps, at least in the early years of its deployment. The same transmission rate can be achieved much earlier with EDGE. This third generation mobile phone system is already available in Japan [2]. The system enables the transmission of video, data and voice communication at a high speed and low cost.

2.1.6 CDMA

Code Division Multiple Access [1] is a proprietary standard for mobile communication, where GSM is an open standard. CDMA was pioneered by Qualcomm and enhanced by Ericsson. Both standards are in competition for dominance in the cellular world. CDMA is adopted mostly in US where it has a large subscription base. CDMA is a spread spectrum technology, which means that it spreads the information contained in a particular signal of interest over a much greater bandwidth than the original signal. A CDMA call starts with a standard rate of 9.6 kbps, which is then spread to a transmitted rate of about 1.23 Mbps.

2.2 Mobile Communication Services

2.2.1 SMS

Short Messaging Service [1] was created as a part of the GSM Phase 1 standard to send and receive short text messages, of 70-160 alphanumeric characters in length, to and from mobile phones. The number of characters which can be sent is dependent on the language in use, with language support limited to the European Languages, Chinese and Arabic. This service is widely popular in Europe and Asia while in the US it is practically non-existent. SMS requires digital wireless interface standard (GSM) which is slowly being adopted in the US. In the US the ‘mobile-party-pays’ pricing model is commonly used, so mobile users pay for incoming as well as outgoing calls. Similarly this is the case with text messaging, so paying for messages received will not help the adoption of SMS in the US.

SMS is a smart service, as it can store messages when to the target mobile device is switched off and forwards the messages when the unit is again in use. SMS applications are voicemail/fax notifications, delivery of replacement ring-tones, operator logos and group graphics, unified messaging, personal communication (text messaging), and information services. Basically, any information that fits into a short text message can be delivered by SMS.

In 2002, there were about 24 billion SMS messages sent per month within the Global GSM world, according to a European SMS Guide, by Netsize [20]. The majority of these were peer-to-peer (mobile-to-mobile) text messages at around 90% of SMS traffic, and the remaining 10%, were mobile transaction services such as news, stock prices, weather, horoscope, etc. SMS continues to grow more as a payment medium, e.g. reverse SMS billing, premium SMS numbering, and as a combination with advanced messaging solutions built around instant messaging via GPRS or e-mail.

2.2.2 WAP

Wireless Application Protocol [1] is a technology which provides a mechanism for displaying internet information on a mobile phone or any wireless device. This is done by translating internet information in to a format which can be displayed within the constraints of a mobile device. WAP is an open standard, developed by the WAP Forum, which has over 500 members. Its founder members include the major wireless vendors of Nokia, Ericsson and Motorola, plus the US software company, Phone.com (formerly Unwired Planet).

To obtain Internet access on a mobile device, the device should be WAP-enabled and the web site information should be described in WML (Wireless Markup Language) format. WML is the mobile equivalent to HTML for web pages. A WAP gateway is also necessary between the client mobile device and the WML host server, to translate the WAP request. The response from the host server is translated into a WAP response by the WAP gateway, which can be displayed on the mobile device. An application environment, called WAE (WAP Application Environment), is defined by the WAP standard to enabling the development of advanced services and applications. These include micro-browsers, scripting facilities, e-mail, www-to-mobile messaging, and mobile to telefax access.

There has being difficulties with the launch of WAP, especially in Europe, due to the slow speed and high charges when using WAP on GSM technology. The increase use of GPRS will see an increase popularity of WAP usage. WAP has been very popular in Asia, except in Japan where I-mode is dominate in this market. WAP is an open standard in contrast to I-mode, which is a proprietary standard. Also, there are difficulties with the configuration of a WAP phone for new WAP services. 20 or so different parameters are needed to be entered to gain access to the WAP service, which may discourage users.

2.2.3 I-Mode

I-mode (I standing for information) [1] is a wireless technology developed by a Japanese company called NTT DoCoMo, which enables uses to access Internet services via their cellular phones. I-Mode can be used to exchange e-mail with computers, personal digital assistants (PDAs) and other I-Mode cellular phones. I-Mode has already dominated the Japanese market and is being considered a success story in the world of M-Commerce.

I-Mode's underlying technology is uncomplicated, which makes it easy for content providers to create new I-Mode services and easy for customers to find and use them. The service is based on the Asian cellular standard PDC and uses Compact HTML (cHTML) markup language [3]. cHTML [2] is basically a scaled down version of HTML. It is relatively easy and it takes little time to rewrite HTML into cHTML. I-Mode's transmission speed is just 9.6kbps, but fast enough for its services. DoCoMo operates a packet-switched network, which means that customers pay not for time elapsed but for the packets of data they download. Packet switching also means that I-Mode is always on, so customers don't have to log into the service or wait for a connection, but have immediate access to services, similarly with GPRS.

2.2.4 USSD

Unstructured Supplementary Services Data [1] is a mechanism of transmitting information via a GSM network. Similar to SMS, but it is only basically a store and forward service. USSD offers a real-time connection during a session. It is said that USSD will grow with the further market penetration of WAP. Its main uses will be for mobile financial services, shopping and payment.

2.2.5 Cell Broadcast

Cell broadcast [1] is a technology that is designed for simultaneous delivery of short messages to multiple mobile users within a specified region or nation-wide. Cell broadcast is similar to SMS, but it is a one-to-many service rather than a one-to-one or one-to-few. It is a mass distribution media mainly for news and generic information. Usually, cell broadcast services are distributed to the consumer on a no cost basis. The network operator charges the content provider for sending the messages and the content provider will try to make money on follow-up services, such as advertising.

2.2.6 SIM Toolkit

SIM (Subscriber Identity Module) Toolkit [4] is an ETSI/SMG standard for value added services and e-commerce using GSM phones to perform the transactions. SIM Toolkit programmed into the special GSM SIM card enables the SIM card, using the GSM handset, to build up an interactive exchange between a network application and the end user and access or control access to the network. Therefore, it provides the SIM card with a proactive role in the handset. This means that the SIM initiates commands independently of the handset and the network. SIM Toolkit [1] is targeted at phones that do not yet fall into the smart phone category. Although SIM Toolkit was being heavily pushed by the smartcard industry, it will be an interim technology and will not be able to survive once GPRS terminals take over the market, since WAP is be the GPRS-supported protocol.

2.2.7 Web Clipping

The Web Clipping [1] service for 3Com's Palm handheld device has been very successful, utilizing Palm's 75% market share of PDA market in the US. Web clipping is a Palm proprietary format for delivery of web-based information to Palm devices via synchronization or wireless communication to the Palm platform. Web clipping may co-exist with WAP in the fragmented US market. However, in Europe it is likely to be superseded, even on the Palm platform, by WAP based services.

2.2.8 MExE

The Mobile Station Application Execution Environment [1] is the incorporation of a Java virtual machine into the mobile phone, allowing full application programming. The protocol is integrating location services, sophisticated intelligent customer menus and a variety of interfaces, such as voice recognition. MExE will incorporate WAP, but also provides additional services exceeding the WAP functionality.

2.2.9 Network Protocols

Infrared data association (IrDA) [5] is a protocol stack which represents the physical characteristics of infrared communication. This wireless communication mechanism enables establishment of connections between devices, which must be in line of sight of each other.

Bluetooth [5] has become the predominant standard for lower power and short-range radio link to exchange information, enabling wireless connectivity between devices and peripherals. It had been adopted by many mobile phone manufacturers, and introduced as an addition communication feature on most new phones.

Hypertext transfer protocol (HTTP) is a text-based protocol for content transfer over the internet. This protocol is used to access web content via a web browser on a mobile device.

Transmission control protocol/ Internet protocol (TCP/IP) is a protocol suite consisting of several protocols at the transport and network layers. At the transport layer, there is the TCP and UDP (User Datagram Protocol) protocols, which are considered reliable and unreliable protocols, respectively. TCP is a stream-oriented and UDP is a packet based protocol. Both can be used to establish socket connections between networked devices.

2.3 Mobile Platforms

2.3.1 Mobile Operating Systems

Symbian [5] was formed from Psion Software, by Nokia, Motorola, Psion (UK PDA manufacturer) and Ericsson in June 1998. In 1999 Matsushita (Panasonic) and in April 2002 Siemens joined the Symbian group. It was based on Psion's earlier software, EPOC operating system. It was a modular 32-bit multitasking operating system especially designed for two types of mobile devices: smart phones and communicators. After EPOC release 5, the operating system was renamed 'Symbian OS'. Symbian is a joint venture between leading phone manufacturers formed to develop a common operating system suitable for mobile communication devices. The operating system is quite simple: Symbian develops and licenses Symbian OS containing the base (microkernel and device drivers), middleware (system servers, such as the window server), a large set of communication protocols, and a test user interface for the application engines of the operating system.

The **Series 60** Platform [5] (Smartphone Platform), designed for Symbian OS, supports mobile browsing, multimedia messaging service (MMS) and content downloading, as well many personal information management and telephony applications. The Series 60 Platform 1.0 provides communication technologies needed in smartphones such as e-mail, WAP 1.2.1 stack, SyncML, MMS, Bluetooth and GPRS.

Microsoft has developed a lighter version of its Windows operating system, called **Windows CE** that has been created especially for small palm-size, hand-held PCs and other consumer electronics devices. A large number of handheld computer/PDA manufacturers mostly coming from the PC industry, such as HP, Casio, Philips and Compaq, have developed their devices around CE. However, CE has faced problems surrounding ease of use, robustness, synchronization and memory requirements. Windows CE is now upgraded to **Windows Pocket PC**.

3COM is the smallest player for mobile terminal operating systems, but it is the global market leader in the PDA market (72% according to IDC in 1998) with the Palm Pilot product and its proprietary **Palm OS** [1]. The operating system is regarded to be inferior to its competitors', but the Palm is much simpler to use in both software and hardware terms. 3COM has spun-off its Palm division (Palm Inc.) in 2000. The Palm OS has a particular wide acceptance in the US.

Wysdom has recently developed a mobile network operating system called **Wysdom MAP-OS** [10]. It is the mobile industry's first complete service delivery platform designed exclusively to meet the challenges mobile operators face in delivering feature and margin-rich mobile data services to their consumer and enterprise customers, according to Wysdom.

2.3.2 Mobile Programming Platforms

Java 2 Micro Edition (**J2ME**) [5] provides an environment to develop applications for mobile devices. Due to the limited resources for memory and execution on a mobile device, J2ME was designed to limit the usage of these resources, more than other Java 2 editions.

J2ME defines two device configurations, the Connected Device Configuration (CDC) and the Connected Limited Device Configuration (CLDC). CDC is designed for high-end devices, such as set-top boxes and communicators, with numerous UI capabilities, high bandwidth and available memory of about 2-4 Mbytes. Alternatively, CLDC is designed for low-end devices, such as mobile phones and personal organisers, with simple UI capabilities, low bandwidth and limited available memory about 128Kbytes.

The CLDC is the base runtime environment defining the common characteristics of a constraint device and specifying the VM (KVM for resource limited devices) and a set of

APIs for this category of device. The CLDC is extended by the Mobile Information Device Profile (MIDP).

MIDP defines the characteristics for a mobile device on top of the CLDC, providing support for user interface, networking and persistent storage. The latest version, MIDP 2.0, provides support for TCP/IP socket connections between different mobile devices and also server applications. There is also support for SSL to provide network level security for the communication of sensitive information, which is especially important in payment systems.

2.4 Summary

This chapter outlined a selection of network technologies and communication services available in the mobile domain. Some of these technologies are currently in existence on mobile networks and the other technologies have still to be fully deployed. The amount of next generation technologies available emphasises the potential of the mobile domain as a platform for mainstream business, commerce and leisure services, leading to the need for support of mobile payment services.

Chapter 3

Mobile Payment

This chapter outlines the main concepts of the mobile payment domain relating to potential mobile payment systems. The principle attributes of mobile payment describe actors, characterisation of payment, possible scenarios and operations involved. Dividing mobile payment into three areas, operator payment, out-of-band payment and proximity payment, further describes the diverse areas in mobile payment, as explained in following sections with current examples of existing payment systems.

3.1 Mobile Payment Principles

The increase in mobile commerce services [6] and the demand for these services is related to the quality of service provided by existing mobile networks (e.g. GSM). These networks are not designed to support data services beyond SMS, therefore providing slow connection speeds and limited choice of applications. With the increase in deployment of packet-switched networks (e.g. GPRS) and the imminent deployment of 3G networks, users will have access to bandwidth levels as good as fixed network access with the added benefit of mobility. This provides an ideal environment for improved payment services for the use of content (digital and physical goods) and services.

3.1.1 Actors within a Mobile Payment System

The mobile payment value chain has various roles which need to be managed [18]. Such roles may be the provision of a service or content, consumer authentication, payment authorization and payment settlement. In a general sense these roles can be assigned to four actors in the payment system; the consumer, the content provider/merchant, the payment service provider (PSP) and the trust third party (TTP), as shown in figure 1.

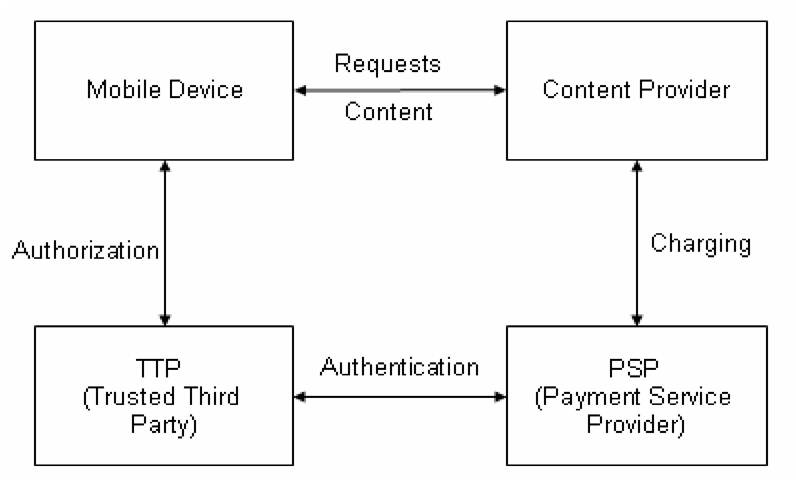


Figure 1: Relationships between all parties in a Mobile Payment System with participation from a Trusted Third Party

The consumer is the person owning the mobile device and is willing to use it to pay for a service or the supply of content. In this report the consumer is referred to as the mobile user. The content may be either physical goods or downloadable digital content and the service may be either a physical or digital service. The role of the mobile user may involve registering with the PSP, initializing the mobile purchase, authorizing the payment and accessing/acquiring the purchased service/content.

The content provider or merchant, depending on whether digital content or physical goods and services are being purchased, is someone or some organization that sells products to the consumer. Their roles may involve forwarding purchase requests to the PSP, relaying authorization requests back to the consumer and delivery of the content. In this report this actor is always referred to as the content provider, as the service is usually the provision of digital content.

The payment service provider is the party responsible for the payment process. They control the payment between the mobile user and the content provider. A consumer may register with the PSP to avoid repetition of keying payment details into the mobile device, such as credit card details, every time a purchase is initiated. A PSP could be a network operator, a financial institution, a credit card company or an independent payment vendor.

The trusted third party is a company used to perform the authentication of transaction parties and the authorization of the payment settlement. This actor could be a network operator, a financial institution or a credit card company. Therefore, their main role is authentication and authorization of payment requests. A network operator or bank could merge roles and act at the same time as the PSP, the TTP, and the content provider. In this dissertation there is no further reference to a TTP, and it is to be assumed that the PSP is responsible for all its roles, as it may be in many cases. Therefore only three actors exist in the mobile payment system as shown in figure 2.

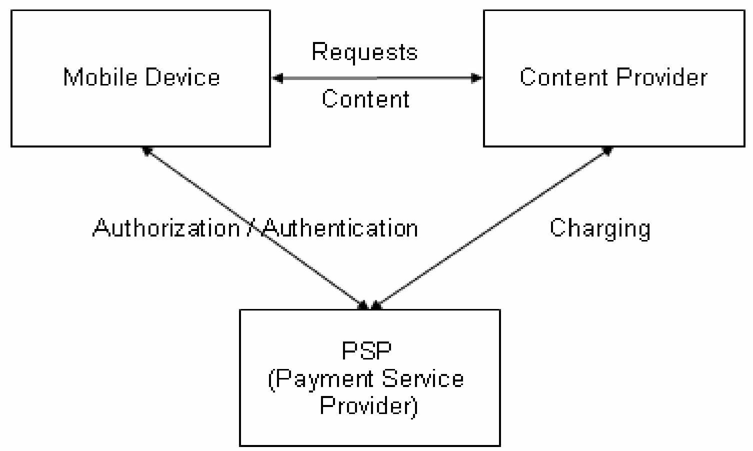


Figure 2: Relationships between all parties in a Mobile Payment System excluding participation from a Trusted Third Party

3.1.2 Characterising Mobile Payment

Mobile payment may be characterized into various categories, such as transaction type, settlement type, content type, and content value. An outline of some of the possible payment characteristics regarding these types is presented below.

- **Transaction Type**:
  - *Pay Per View* – the mobile user pays for each view, or increment, of the desired content. For example downloadable MP3 files or video clips.
  - *Pay Per Event* – the mobile user pays for an event. This event may be the use of a service for a particular time interval or value.
  - *Pay Per Unit* – the mobile user pays for each unit of content provided by the content provider. Units can be based on volume or duration of content, such as per byte or per minute. The amount of units used for each session will be billed to the consumer. Such examples of this type could be used in downloadable games or streaming video content.
  - *Flat Rate* – the mobile user pays a recurring periodic amount to access the content on an unlimited basis during the period. For example unlimited access to online newspaper articles.

- **Settlement Type**:
  - *Pre-paid* – mobile users pay in advance of obtaining the content with pre-paid accounts which are deducted after each payment session.
  - *Post-paid* – mobile users receive and use the content before they paid for it. The consumer is billed after the access to the content is obtained, for example on a phone bill.

- **Content Type**:
  - *Digital goods* – e.g. downloadable music or video content, value-added information

- *Digital services* – e.g. video streaming services
- *Physical goods and services* - e.g. pay-parking
- *Voting* - e.g. TV voting polls
- *Ticketing* – e.g. booking plane tickets

- **Content Value**:
  - *Micropayments* – describes same purchases usually less the 10 Euro, for example pay parking and ring tones.
  - *Macropayments* – usually large purchases over 10 Euro, for example purchasing plane tickets.

3.1.3 Payment Scenarios

Content Download

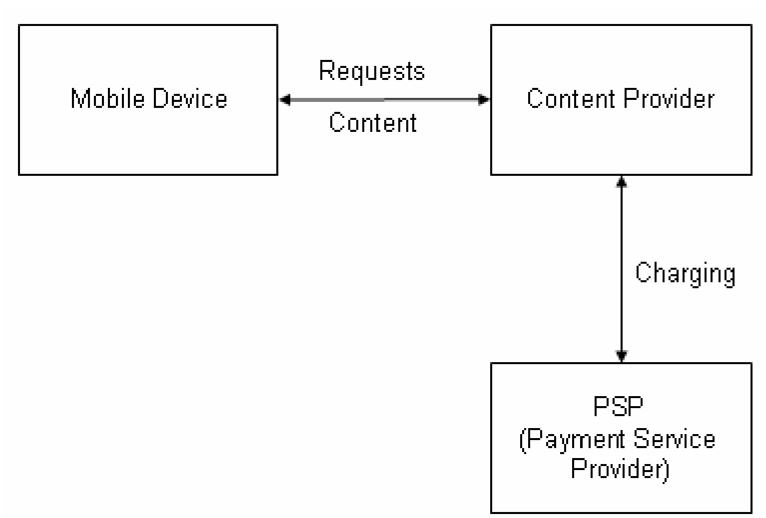


Figure 3: Payment System for Content Download

In this payment scenario, as shown in figure 3, the content provider offers digital content to mobile users. The content can be purchased by either a metered or event pricing model.

Metered content may be a streaming service, such as a video, a radio channel or an on-demand game service. The payment of the transaction is dependent on a metered quantity of the provided service, such as the duration of the service, the data volume delivered, or type of gaming sessions (e.g. different levels).

Event content may involve the full download of digital content, in which the consumer pays a predefined price per item downloaded. The transaction is dependent on a successful download, as the content is worth the purchase price only when it is complete downloaded. This pricing mode may also cover recurring charge agreements or subscriptions, e.g. to a monthly online magazine subscription.

The content may also be purchased via a PC internet connection, where the mobile device will be used to authorize the payment transaction and authenticate the content recipient as the mobile user.

Once a service request is made by the mobile user to the content provider, then the content provider will initiate a charging session with the PSP. The PSP will seek authorization from and authentication of the mobile user to complete the payment transaction, using either a post-paid or prepaid method.

Point of Sale

In this payment scenario, as shown in figure 4, the content provider or merchant will offer services or the sale of goods to the mobile user, at a point of sale location, e.g. paying for a taxi service or purchasing a physical good in a shop. The payment will be initiated at the point of sale by the content provider. The PSP may request authorization from the mobile user either directly, such as a sms pin request, or indirectly via the content provider, such as using a wireless Bluetooth link.

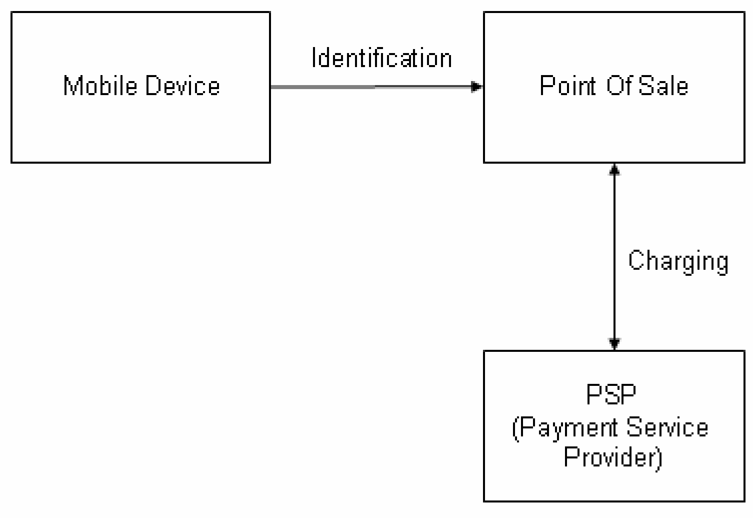


Figure 4: Payment System for Point of Sale

Also vending machine scenarios apply here. A mobile user can pay for goods and services at a machine, such as buying public transport tickets or paying for parking. Identification of the mobile user may also involve using a wireless link such as Bluetooth or Infra Red, and may be in addition with the use of a sms pin request for authentication. Point of sale applications could be developed using the MIDP 2.0 profile for J2ME, as support is provided for Bluetooth and SSL (Secure Socket Layer).

Content on Device

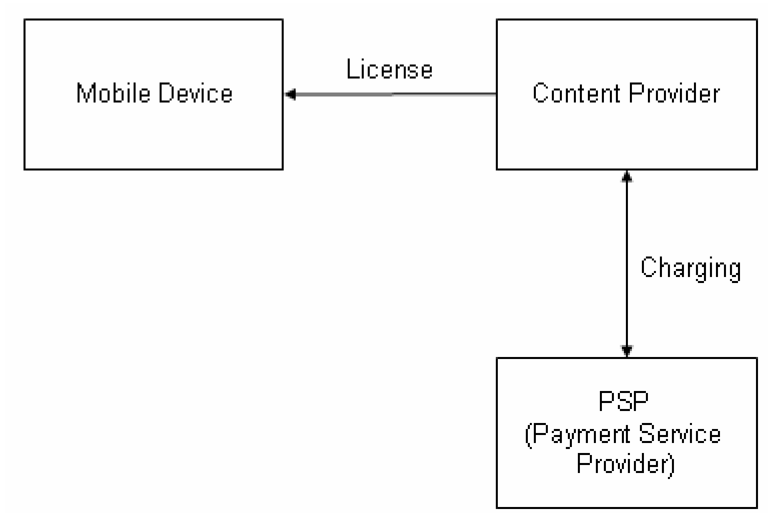


Figure 5: Payment System for Content on Device

In this payment scenario, as shown in figure 5, the content may already exist on the mobile device, in which the use of this content may involve obtaining a digit license. The license may be based on usage, duration or number of users. Such content using this scenario will be an on-demand gaming service. In this scenario, the license is a form of content which payment is required to obtain it.

3.1.4 Generic Operations in Mobile Payment

Within a mobile payment system, there exist certain interactions between the parties involved, which are necessary during a payment session. The sequence diagram in figure 6 outlines some generic operations between each party. These operations may vary depending on the payment scenario.

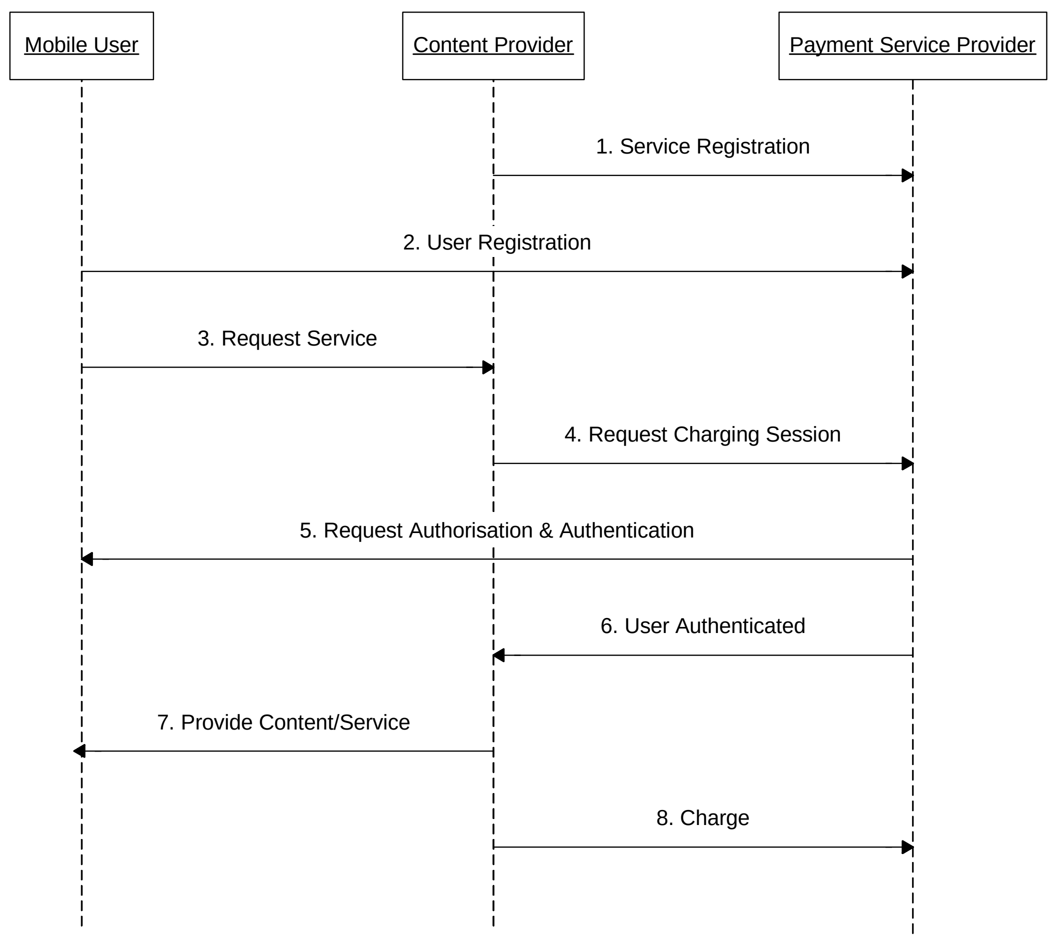


Figure 6: Generic Operations in a Mobile Payment System

- **Service Registration** - The content provider needs to register with the payment service provider, to make their service available to mobile users via the payment system. Service details, such as payment characterisation for the service, and also the financial information for receipt of payment after use of the service. Following registration, the content provider will receive a service identification number, for identification in further operations with the payment service provider.

- **User Registration** - The mobile user needs to register with payment service provider before they can avail of services offered by a registered content provider. The user will submit details of how they wish to pay for the services and they may be able to personalise the payment to suit their needs. Also the user may be requested to submit a personal identification number (PIN), which will be used during the user authentication process. A user identification number will be returned to the user following the completion of registration. This id will uniquely identify the user during each payment transaction.

- **Request Service** - Once the mobile user has registered with the payment service provider, they are able to request the use a services provider by a registered content provider.

- **Request Charging Session** - On receiving the service request, the content provider will initiate a charging session request with the payment service provider. Both the user's id and the service's id will be sent to the payment service provider to uniquely create a charging session, which will be represented by a unique session identification number.

- **Request Authorisation and Authentication** - Before a charging session can be started, the mobile user must authorise that they are willing to pay for the service. An authorisation request will be sent from the payment service provider to the user, usually in the form of a payment contract. This contract will state the terms and conditions of the payment agreement between the mobile user and the content provider. The mobile must reply to the payment service provider with confirmation of acceptance of the terms in the contract, before the charging session process can continue. The mobile user will also be requested to authenticate with the payment service provider. This will usually be done by submitting their PIN from the mobile device being used. Authorisation and authentication may be performed within the one request and the authorisation confirmation may include the authentication PIN.

- **User Authenticated** - The payment service provider will notify the content provider whether the mobile user has been authenticated successfully. If authorisation and authentication requests return a positive result, then the charging session will be initiated. The payment service provider will issue a unique session id to the content provider, signalling the start of a charging session.

- **Provide Content or Service** - The content provider will now provide the requested content or service to the user.

- **Charge** - Depending on the payment scheme related to the requested service, the content provider will request a charge operation to the payment service provider at the end of the service or at different intervals during the service. On receipt of the charge request, the payment service provider will settle the payment transaction between the two parties and notify each party of the result of the transaction. This will usually be presented to the mobile user as a payment receipt.

3.2 Mobile Payment: Mobile Operator Payment

Network Operators are well suited to deliver payment services for mobile content due to their expertise in the area of billing. This type of payment is sometimes referred to as "in-band" [6], where the content and the payment channel are the same, e.g. a chargeable WAP service over GPRS. Mobile users will either be offered subscription or per usage payment models, with the amount of payment usually being small, i.e. micropayments. Applications that could be covered by in-band transactions included video streaming of sports highlights or video messaging.

An example of In-Band payment: Bob is on holiday, and uses his smartphone to take a photo, adds audio comment, and sends it via MMS (Multimedia Messaging Service) to Tom. He is charged 1 Euro to his prepaid account. This would use MMS and 2.5G technologies, involving a mediation system integrated with real-time stored value micropayment system.

3.2.1 Network Operator Payment Systems

Mediation systems provide the systems to manage the charging models and integrate with various payment methods, such as billing systems and prepay systems [6]. Operators are generally not interested in providing a standalone payment application because charging and payment are at the centre of their wireless data systems and form part of the network operator's infrastructure.

Operators deploying these systems will take on the role of the payment service provider. They want to control the relationship between the user and the content providers, but also they are faced with investing heavily in payment systems that can support these complex models. They can choose to outsource transaction processing to an ASP (Application Service Provider) that is an expert in aggregating micropayment, or they could upgrade their systems to support data services billing. The second option is usually the better option to maintain trust and security confidence with the mobile users.

3.2.2 Mobile Payment Standards

In mid 2003, Orange, Telefonica Moviles, Deutsche Telekom, unit T-Mobile International and Vodafone formed the **Mobile Payment Services Association** to foster development of an open, commonly branded system designed to work across all mobile networks [8]. A second group, PayCircle, recently released its reference implementation for the first version of its payment service specification.

PayCircle[1] [9, 11] was formed early 2002 by Internet and telecom network technology vendors CSG Systems, Hewlett Packard, LogicaCMG, Lucent Technologies, Oracle, Siemens and Sun Microsystems to develop uniform application programming interfaces (APIs) for payment systems based on Internet languages. The companies realize they have more to gain by contributing to a standard than by competing with proprietary offerings. With a proprietary-segmented market, mobile commerce activity would be a fraction of what it can be with standardised APIs.

The **OSA/Parlay** standard [12] is a merged standard derived from OSA and Parlay. OSA (Open Service Access) is developed by the 3GPP organization and Parlay is developed by the Parlay Industry Consortium. The objective of OSA and Parlay was to simplify application development for fixed and mobile networks and open up to a larger development community than what traditionally existed for telecom networks. Via standardized OSA/Parlay interfaces, the Application Server interconnects with the mobile operator's network to use functionality in the network. For charging the application server interfaces the billing system via a charging gateway.

[1] PayCircle is a vendor-independent non-profit organization. Its main focus is to accelerate the use of payment technology and develop or adopt open payment APIs based on XML, SOAP, Java and other Internet languages.

The alternatives to OSA/Parlay are solutions based on vendor proprietary interfaces between the application server and the billing systems. The disadvantage with proprietary solutions is the scalability, i.e. proprietary solutions are more expensive to maintain since each new application server needs to implement the proprietary interface.

3.2.3 Vendor Billing Solutions

The current vendors for providing mobile billing solutions for GPRS and 3G services are Amdocs, Cerillion Technologies, Convergys, EHPT, Geneva Technology, Kenan Systems, Portal, Sema, and TelesensKSCL [12]. Some of the services provided are content billing management and CRM (Customer Relationship Management) services.

3.3 Mobile Payment: Out-of-Band Payment

Out of band payment [6] refers to the fact that the content and payment channels are separate, e.g. a credit card holder may use their mobile device to authenticate and pay for a service they consume on the fixed line Internet or interactive TV. This type of payment usually involves a system controlled by a financial institution, maybe in partnership with a mobile operator. In order to make the wireless device suitable for authenticating payments, financial institutions are especially interested in wireless PKI, shared secret (or symmetrical key) schemes, or merging with their chip card programs via dual slot or dual chip devices.

An example of Out-of-Band payment: A SMS notifies Anna that music concert tickets have just gone on sale. From an Internet Café she browses to the ticket vendor site, books her tickets and pays with her Visa card. The payment authentication request appears on her mobile phone via SMS, and she authenticates using a personal PIN, digitally signing the order. A receipt is sent to her phone. Here wallet server technology with SMS and PKI support and an acquiring gateway is needed.

3.3.1 Financial Institutions

Banks are already seeing the opportunity for using mobile phones as a personal secure payment terminal [6]. Different payment schemes exist where a bank will deduct payment from a mobile user's account to pay for a service or virtual product. The payments involved here are usually of higher value than micropayments. Various methods are used to authenticate the payment transaction, such as using a dual slot phone for credit card payments, PIN authentication via a SIM toolkit application and also with the use of digital signature based on a public key infrastructure (PKI) mechanism. The adoption of a PKI system requires at least 2.5G technology, so therefore this type of system has been slow to reach the markets.

At the moment, there are schemes where the security is based on the mobile user being in possession of a registered mobile device and authentication is obtained via a PIN. The mobile user is required to register their mobile phone with the payment service provider, allowing the payment transaction to be authenticated using a variety of technologies. Such examples of these systems are offered by Paybox and MobiPay (will be described later on).

3.3.2 Reverse-Charge/Billed SMS

Reverse-billed premium rate SMS services deliver content to mobile phones for a charge [14]. Customers typically subscribe to a service and are then charged a premium for the messages they receive. The payment model enables consumers to use SMS text messaging to anonymously pay for access to digital entertainment and content. Reverse SMS billing means that the owner of the recipient phone rather than the message sender is charged for the cost of the SMS message received. There are various vendors offering reverse-charge SMS services to content providers, providing an alternative payment option not connected to mobile network operators' infrastructure.

3.3.3 Vodafone m-pay

Vodafone m-pay allows you to bill users directly on their mobile phone bill. There is no need to send them an SMS each time you wish to bill a user. Instead, users are billed when they enter their username and password details on the web or WAP site where they are buying content from.

3.3.4 MobiPay

MobiPay (formerly Movilpago) is a Spanish company, which launched a pilot scheme for mobile payments [8]. Based on a cooperative model between mobile telephone operators and financial institutions, MobiPay is owned by Banco Bilbao Vizcaya Argentaria and Santander Central Hispano, as well as all Spanish mobile network operators (which include Vodafone but not O2). MobiPay's payment system [7] can work in several ways. In a traditional merchant environment, the customer either tells the sales assistant their mobile phone number or (in larger retailers) allows the sales assistant to scan their phone using a special barcode reader. The POS (Point of Sale) terminal sends the phone number, a description of the goods and the payment amount to MobiPay.

MobiPay makes an Unstructured Supplementary Services Data (USSD) call [8] to the customer's mobile phone and sends the "invoice" and amount. The customer authorizes the transaction by punching in their PIN code. All of this takes a few seconds.

The service costs the customers nothing and the charge to merchants (apart from the special POS interface if they choose to have one) is "comparable" to credit cards. The system currently offers two payment options to customers:

- A pre–paid network wallet (separate from the operators' pre–paid wallet) that can be loaded manually or automatically.
- A post–pay (against a bank account) option.

There is an alternative for handsets incapable of placing USSD calls, where the customer calls MobiPay on receipt of a payment instruction and confirms the payment by a voice call.

3.3.5 Iti Achat

France Télécom launched a service whereby consumers can pay for goods (which they have ordered using a voice service) by inserting their bank card into the external slot in their mobile handset [8]. The pilot was called "Iti Achat", a name that may still be in use, although the operational service is now called CB Payments on Mobile.

3.3.6 PayBox

PayBox [2] was established EKS, Oracle, Compaq, Lufthansa Service and Deutsche Bank, where Deutsche Bank has the largest share in the company and deals with the client databases, clearing and settlement. This system allows for the debiting of bank accounts to pay for services via a PayBox voice message requesting payment confirmation and the customer using their mobile phone to authorize the payment with a PIN code.

3.4 Mobile Payment: Proximity

A payment system with good potential for mobile commerce is proximity transactions, such as using a mobile device to pay at a point of sale, vending machine, ticket machine, tolls, parking, etc. By using wireless technologies, such as Bluetooth and 802.11, mobile devices can be transformed into sophisticated payment devices that can process both micro and macro payments [6].

A proximity payment example: Bob is the photo and imaging shop. He transfers his holiday photos from his digital camera to the store computer over a Bluetooth link. The payment request is sent to his mobile phone, also over a Bluetooth link, where he accepts it, and his credit card information is returned to the store point of sale device. The technologies that could be used here are Bluetooth, or some other wireless technology, and a payment J2ME application, based on MIDP 2.0, running on the mobile phone and the point of sale device.

3.4.1 Smartcards

Smartcards, i.e. chip cards with a small microprocessor such as GeldKarte, Proton or Mondex, can have credit/debit functionality as well as digital signature or electronic wallet functionality [1]. The SIM cards used within the GSM phone are smartcards as well. Their size and compatibility with the magnetic stripe card theoretically makes the smartcard an ideal carrier for personal information, such as secret keys, passwords, customization profiles and medical emergency information. Although many smartcards have been delivered to customers for other reasons, such as ATM cards, and not as a debit card for direct payments, there is ongoing speculation about the success of smartcards as a mobile wallet.

A common standard for smartcards is still absent. The 20 member strong OpenCard organization grouped around IBM, Sun, Visa, Gemplus and Schlumberger have tried to push for interoperable smartcard solutions based on Java across many hardware and software platforms, but they do not seem to be overly committed to make it work. Visa, for example, has also developed a proprietary solution, called Open-Platform that it is pushing independently into the market.

3.4.2 EMV

In a relatively short time almost all European consumers, and many others around the world, may have a bank–issued smart electronic payment card [8]. These cards will be based on EMV: the Europay-MasterCard–Visa standard. Most schemes for moving existing ‘dumb’ credit, debit and charge cards over to smartcards have declared EMV compliance to be one of their goals. Even France, where smart payment cards have been in use for many years, has decided to switch from proprietary standards to EMV.

The current plan is for all European payments cards to be replaced by smart cards by 2005. This mass issuing of EMV cards is relevant to mobile commerce. A smart credit card can be used with a two–slot mobile phone to pay for a theatre ticket, for example, quickly and conveniently. Since the banks are issuing the cards anyway, many of them will choose to issue cards that have both EMV and mobile wallet applications on them.

3.4.3 Mobile Wallet - Micropayments

The use of the mobile phone as a payment device for impulse purchases at unattended POS may become very significant [8]. The Sonera Coke machine demonstrates how such a system might work and there are already other suppliers working to develop infrastructure. It isn’t only operators, but third–party service providers who are pushing forward in this area. Coca–Cola and its local bottling partners are investing in bringing vending machines online. The technology will allow customers to make cashless purchases and give bottlers greater flexibility in managing inventory. About 60,000 machines in the US, Australia and New Zealand are already on line.

The use of the handset as a payment device is spreading. Scandinavians pay for their car parking using their phones in several cities. In Ireland in 2003, O2 and Vodafone are involved with a public parking scheme run by Dublin City Council where drivers can pay for parking using their mobile phones [7]. Hundreds of thousands of French pre-paid users top up their accounts by slotting their bank card into their phones and Japanese consumers buy from vending machines by dialling the machines.

3.4.4 Mobile Wallet - Wireless Technologies

Due to the increasing emergence of m-commerce, mobile applications could become an important and widely adopted tool for use in financial transactions. However, at present, one of the outstanding problems is that certain resources limit mobile devices, most notably memory and communication facilities, battery power and security.

Improvements in wireless networks in terms of protocols, standards, infrastructure and user acceptance, have been significant in the last few years. Two of the widely adopted wireless technologies, Bluetooth and IEEE 802.11, are seen as the future communication solution for mobile payment mediums. Both standards have their inherent benefits and drawbacks, but neither has proven more suitable than the other for an application in the mobile payment domain.

Although Bluetooth can co-exist with IEEE 802.11 [16], they interfere with one another in close proximity. In the context of a mobile payment, trade offs exist with the two standards providing different properties that suit this payment environment in contrasting ways.

One of the main issues with a mobile wallet system using these wireless technologies is the lack of security for payment transactions. It is believed that additional application level security is needed, such as cryptographic mechanisms provided by DES and PGP algorithms or a Public Key Infrastructure (PKI) mechanism. These may be embedded on to a SIM card, possibly using the Dual-SIM technique, where two SIM cards exist in a mobile phone. Also PKI certificates can be pre-installed to memory on to smart phones for creating secure connections with an authenticated server application.

3.5 Summary

This chapter outlined the generic characteristics of the mobile payment domain. As explained under the three different areas, mobile payment is growing, but great diversity exists among mobile payment services.

The next chapter will outline the design for a generic mobile payment framework, based on the Web Services architecture.

Chapter 4

Framework Design

This chapter describes the distributed software design techniques employed in this dissertation project. Web Services standards are the main architectural components used in the design of this framework for mobile payment services. These standards provide a façade interface to any framework implementation of a payment system. This integration of Web Services technologies and framework design patterns explores the co-existence of object oriented programming techniques with web based structuring and communication.

4.1 Web Services

Web Services [21] are self-contained, self-describing XML based software components. They provide architecture for distributed loosely coupled services that can be published, located and invoked remotely over Internet protocols, by service clients written in a different language.

Web Services release distributed systems from the constraints of single network existence and opens the integration of heterogeneous systems over IP. This is accomplished without the difficulties associated with firewall re-configuration as endured by other distributed architectures such as CORBA and Java RMI. This interoperable architecture uses existing internet protocols, such as HTTP and IP.

The main components of the Web Services architecture are the open standards; XML, SOAP, WSDL and UDDI.

4.1.1 XML

The Extensible Markup Language is the foundation on which Web Services are built by describing all aspects of Web Services. XML defines a standard way to structure information for describing, storing and exchanging data via Web Services. This standard was developed for applications that require functionality beyond the current Hypertext Markup Language (HTML).

XML enables a structured communication of data between Web Services components. There are no predefined semantics, so the definition of data must be agreed in advance between communicating parties. One of the main advantages of XML is its feature of independently extensible documents, which may be extended without negatively affecting applications dependent on the XML document.

4.1.2 SOAP

Simple Object Access Protocol specifies a simple format for communicating XML encoded messages in the Web Services architecture. SOAP messages are carried over standard Internet protocols such as HTTP, SMTP and MIME. All SOAP messages are encoded in XML and each message is a XML document.

The message structure defined by SOAP consists of three major parts, the envelope, the header and the body. All parts are mandatory in a message, except the header element, which is optional. The envelope is the top-level XML element in a SOAP message. The envelope contains the header and body, and is the unit of communication. The header is used to extend the SOAP message with additional features and functionality, such as security, transactions, and other quality of service attributes associated with the message. The body contains the payload of the message, i.e. the application-defined XML data being exchanged in the SOAP message.

4.1.3 WSDL

Web Services Description Language defines a standard way to describe and publish the formats and protocols of a Web Service. A WSDL file describes how a service is located and bound to by clients. WSDL is written in XML and each WSDL file is a XML document.

In a Web Service interaction the WSDL file is produced and published by the service side and the WSDL file is used to obtain the necessary information about a service by the client side. Both parties need copies of the same WSDL file for the interaction to work.

The main components defined by WSDL Schema are as follows:

- Types: a container for data type definitions.
- Message: an abstract definition of the data being communicated.
- Operation: an abstract definition of the operation for a message supported by the service.
- Port Type: an abstract set of operations mapped to one or more endpoints.
- Binding: a concrete protocol and data format for a particular port type.
- Port: an endpoint definition defined as a communication of a binding and a network address.
- Service: a collection of related endpoints.

4.1.4 UDDI

Universal Description, Discovery and Integration specification defines the implementation of a registry for finding web services. It stores WSDL files which define web service interfaces. This Web Service registry is communicated to by SOAP and is intended to act as a search engine for services. The Web Service party will use the UDDI registry to publish the existence of their service and the Web Service client will use the UDDI registry to obtain location, description and binding information from the WSDL file stored in the registry.

4.2 A Web Services Framework

A Web Services framework introduces extensibility and interoperability to the generic mobile payment system. As shown in figure 7, the framework is situated in the position of the payment service provider, which is integrated with various framework implementations for different payment methods. This section describes the design patterns and WSDL interfaces implemented in the Web Services framework.

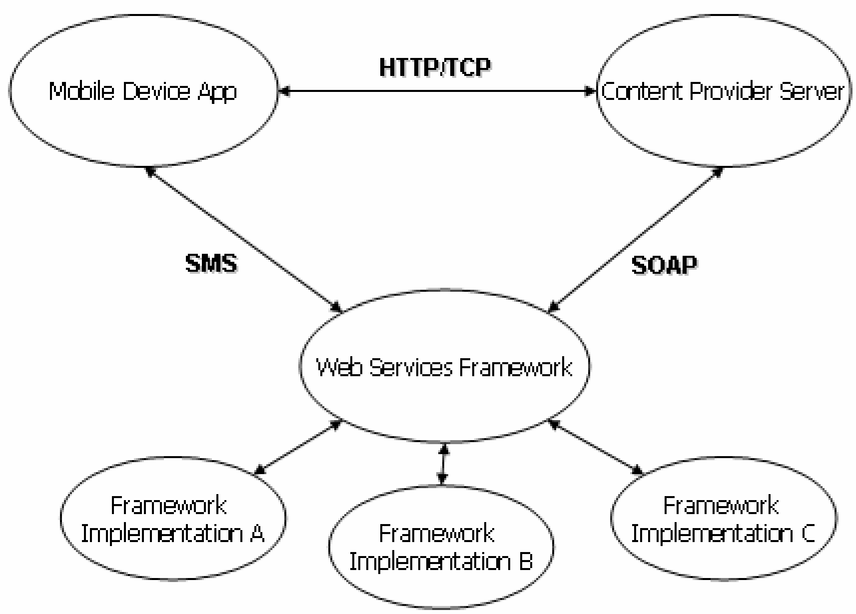


Figure 7: Mobile Payment System with a Web Services Framework

4.2.1 Abstract Factory Design Pattern

Abstract factory patterns provide an interface for creating families of related or dependent objects without specifying their concrete classes [22]. This is of benefit to the software designer when an abstract design is required for a system, which will be implemented in many various ways, possibly in different programming languages to suit different application environments.

This design pattern consists of a parent component, the abstract factory, which declares an interface for operations that create abstract product objects. These abstract products declare an interface for a type of product object. The implementation of this design releases a concrete factory which implements the operations of the abstract factory to create the concrete product objects. The concrete products define a product object to be created by the corresponding concrete factory. This product object implements the abstract product interface. Client objects will only use interfaces declared by abstract factory and abstract product classes.

4.2.2 WSDL Interface as a Stateless Façade

The characteristics of a web service are specified by the WSDL file which is published for the service; therefore the WSDL interface defines the web service. A best practice for designing Web Services is to use a stateless session façade pattern. The reason for a session façade pattern is to hide a complex subsystem by creating a coarse-grained façade to provide client access to the subsystem. Therefore the façade does no new work it just is a point for access to existing subsystem functionality.

All Web Services are generally stateless, due to no definition for a stateful connection in WSDL and the transport protocol HTTP does not retain state. It is possible to force state into a Web Service using information held in a HTTP 1.0 session or in a singleton, but it is best that services remain stateless for better scalability performance.

4.2.3 Registration Web Service

The design of the Web Service framework consists mainly of two WSDL interfaces, the registration and charging interface. These interfaces specify all the operations performed by the framework, the data type information to be transacted, and the location and binding information necessary to find and use the service.

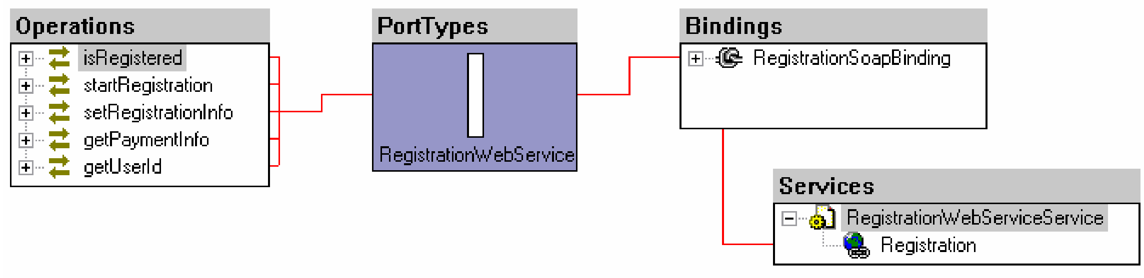


Figure 8: WSDL Design View of the Registration Web Service

The first WSDL interface defines the characteristics for a Registration Web Service (see figure 8 for design view). This service is designed for client registration with the payment service provider, which covers mobile user and content provider registration. The registration process involves sending client information as parameters in predefined data type objects to the Web Service. These parameters will be extracted and stored in a database for further reference. On receipt of this data the service will return a unique, client identification, data type object. This client id will be used by the client in future correspondence with the payment service provider.

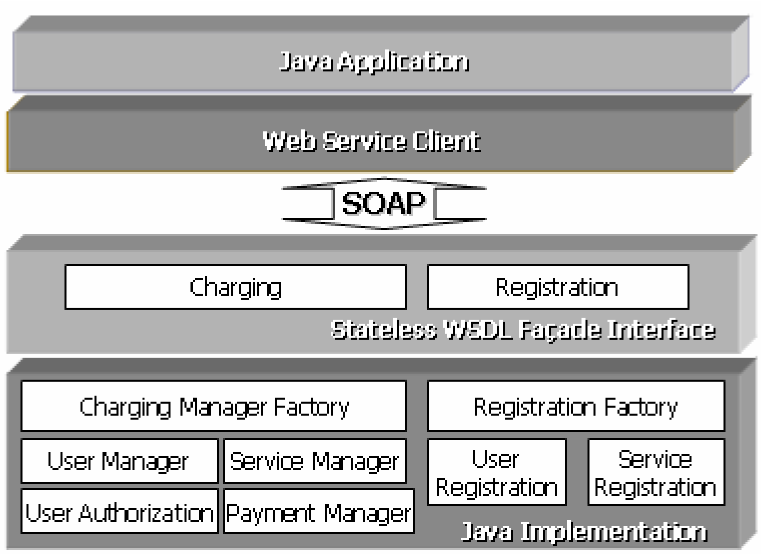


Figure 9: Component Architecture for the Web Services Framework

Using the abstract factory design pattern, outlined previously, the registration part of the framework is comprised of three interfaces. The registration interface is a façade abstract factory interface for the framework implementation, which creates two concrete products, user registration and service registration. They respectively process the registration information for the mobile user and the content provider. Examples of different concrete user objects the abstract factory can create include users that register for operator billing payment, credit card payment and reverse SMS payment.

4.2.4 Charging Web Service

The second WSDL interface defines the characteristics for a Charging Web Service (see figure 10 for design view). This service is designed for creating a payment transaction period, called a charging session, for the use of a content service by the mobile user. The main function of this Web Service is to transact an agreed amount from an authenticated mobile user to the content provider supplying the service used. The processes involved should be secure, atomic and transparent to the payment clients, i.e. the mobile user and the content provider.

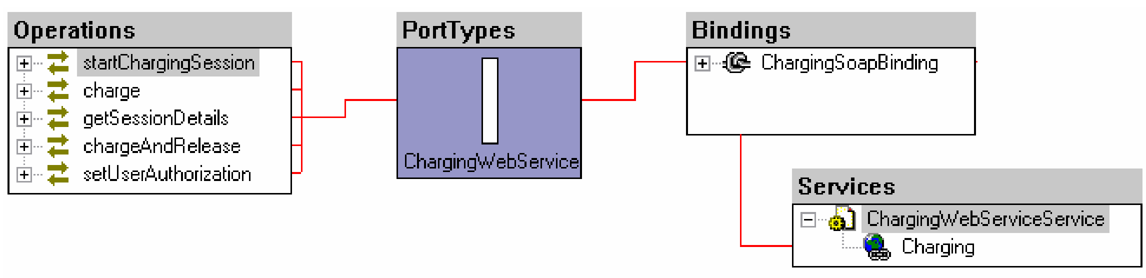


Figure 10: WSDL Design View of the Charging Web Service

A charging session is defined by the time interval starting from when a payment contract is agreed by all parties until the payment agreement is no longer valid. Each charging session can have only one mobile user, one content provider and one payment service provider. There may be numerous payment transactions in a charging session, depending on the transaction type used. A charging session is represented by a unique, persistent session identification data type object, which is related to every operation performed during the life time of a charging session.

Before a charging session can be started, various operations must be done to ensure that all parties agreed to the terms of the session and are aware of the consequences of the session transactions. A charging session is requested by a registered content provider after a registered mobile user requests to use a content service. At the onset of this session request a contract is produced using the service and user details, taking from the registration information. The contract outlines the payment agreement between all parties, specifying the amount to be paid for a unit value of a service during a certain interval. The contract is presented to each party, which must digitally authorise it to make the charging session valid.

To ensure that the mobile user is the same person from whom the payment will be deducted from, the user must authenticate themselves with the framework. To limit the number of messages to the user and the tasks which the user must perform, the framework will send the contract to the user for authorisation and then the user will be requested to authenticate themselves by replying with a PIN to the framework. For authentication to be complete the PIN should match the PIN entered by the user during registration. Once authorisation and authentication are complete then the charging session may become active.

Similarly to registration, the charging part of the framework is designed using the abstract factory design pattern methodology. The charging manager interface is a façade abstract factory interface for the framework implementation, which creates three abstract products called managers. Each of these manager interfaces effectively manages a proportion of the Charging Web Service from a perspective of one of the parties involved.

The user manager interface is responsible delivering the relationship between the mobile user and the payment service provider. The main functions involved are sending the charging session contract, requesting authorising of the contract and authenticating the user via a comparison of PIN numbers. The implementation of user manager interface will have an association with an instance of the user authorisation interface. This is an interface to the communication medium for transmission of the session contract and the receiving of the authorisation confirmation and authentication attempt. As discussed later in the next chapter, SMS technology is used to communicate the requests and responses.

The main function of the service manager interface is the creation of a charging session contract. The manager extracts relevant information from the user and service registration stored data to construct the payment agreement between the parties involved. All major aspects of the contract are specified by the content provider during the service registration process and some addition options may be chosen by the mobile user, e.g. the settlement type for the payment transaction. The contract is defined by the charging WSDL interface as a complex data type, which may be extended to fit any kind of service and payment scenario. This methodology will be explained further in the next section.

Finally, the payment manager interface controls the process of settling payment transactions. This would involve integration with existing mobile payment infrastructures. This interface uses the session contract details to determine the attributes of a payment transaction and following this, the transaction is completed and a payment receipt data type is returned to the clients. Once a charging session is no longer required, a charge and release operation is performed to charge any remaining amounts and followed by releasing the charging session. All session details are now obsolete and the session id will be invalid, therefore this information will be removed from the payment service persistent storage.

4.2.5 Client and Session Identification

To uniquely identify a client of the mobile payment web service, a XML type is designed with extensible elements to detail the unique characteristics of a client. Such characteristics may be the name of the client, the type of client i.e. mobile user or content provider and the payment method being used. Also an element is defined for a unique client id number. This client id type will be used to identify the client for all web service transactions and is obtain by the client upon completion of registration with the payment service provider.

```
<complexType name="SessionId">
    <sequence>
        <element name="id" type="xsd:string"/>
        <element name="user" type="com:UserId"/>
        <element name="service" type="com:ServiceId"/>
        <element name="payment_type" type="com:PayType"/>
        <element name="session_authorized" type="xsd:boolean"/>
    </sequence>
</complexType>
```

Figure 11: WSDL Complex Type for the Session Id

Similarly, to uniquely identify a charging session a XML type, as shown in figure 11, is designed with extensible elements to detail such information which will uniquely distinguish different sessions. The information covers the previously mentioned client types for the mobile user and content provider, which is accompanied by a unique session id number. Also elements are defined for the type of payment method and for session authorisation. The session authorisation element is set to true when the session has been successful authorised by the mobile user. The methodology behind the design of these types is explained further in the next section.

4.3 Designing with WSDL

WSDL provides a declarative way to design framework interfaces by utilising the extensibility features of XML. XML extensibility allows the developer to extend an existing type, simple or complex, by appending elements. These extensions do not break any existing systems and by extending additional functionality in the WSDL types, the framework will have various implementations as a result.

In addition to extensibility, WSDL frameworks may be designed using techniques similar to object oriented design, such as polymorphism and contravariance[2].

4.3.1 Abstract Complex Types

XML complex types have elements which contain other elements and/or attributes. These are user defined types which can be extended to any number of elements. The use of abstract complex types allow the creation of base types which can extended by other types with more additional specific elements. This concept imitates sub-typing from the object oriented paradigm. The abstract base type is equivalent to an abstract parent class in Java which is extended by a complex type, the implemented concrete class in Java. As described in the next chapter, the Web Services framework converts the defined WSDL base complex types and extended complex types to abstract Java classes and serializable extended Java classes.

The sample of WSDL code, as shown in figure 12, from the Charging Web Service demonstrates the use of abstract complex types. The base abstract complex type, ClientId, defines four elements, which two of the elements are of a defined simple type. The complex types, UserId and ServiceId, both extend the base type ClientId; therefore they inherit the elements defined by the abstract type.

---

[2] Contravariance [23]: a form of overriding in which an argument associated with a method in the child class is restricted to a less general category than the corresponding argument in the parent class.

```
<complexType name="ClientId" abstract="true">
    <sequence>
        <element name="name" type="xsd:string"/>
        <element name="id" type="xsd:string"/>
        <element name="client_type" type="com:ClientType"/>
        <element name="payment_type" type="com:PayType"/>
    </sequence>
</complexType>
<complexType name="UserId">
    <complexContent>
        <extension base="com:ClientId">
            <sequence>
                <element name="mobile_number" type="xsd:string"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="ServiceId">
    <complexContent>
        <extension base="com:ClientId">
            <sequence>
                <element name="service_URI" type="xsd:string"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

Figure 12 : WSDL Complex Type Definitions

Restrictions are used to control acceptable values for XML elements or attributes. An example of the use of restrictions is shown in figure 13. The simple type definitions are declared with restricted values for type of element. As shown in figure 12, these simple types are defined as elements in the ClientId abstract complex type. The restrictions ensure that only certain values are allowed, as in the case of the simple type, PayType. Here the restricted values define the payment methods of the framework implementations, which can be extended to include the implementation of another payment method.

```
<xsd:simpleType name="PayType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="Reverse SMS"/>
        <xsd:enumeration value="Operator Billing"/>
        <xsd:enumeration value="Credit Card"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="ClientType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="user"/>
        <xsd:enumeration value="service"/>
    </xsd:restriction>
</xsd:simpleType>
```

Figure 13: WSDL Simple Type Definitions

4.3.2 Polymorphism and Extensibility

The WSDL interfaces expand the use of abstract complex types to allow different implementations of complex types for heterogeneous payment systems by exploiting the powerful object oriented mechanism of polymorphism. In the WSDL code sample, figure 14, the abstract complex type, UserPaymentOptions, is extended by the concrete complex type for the operator billing payment system, OperatorBillingUserOptions. In any operation defined by this WSDL interface which might expect the complex type of UserPaymentOptions, we may use the type OperatorBillingOptions. Employing this mechanism allows the framework to be implemented to fit any payment system without changing the operations of the framework.

```xml
<complexType name="UserPaymentOptions" abstract="true">
    <complexContent>
        <extension base="reg:ClientPaymentOptions">
            <sequence>
                <element name="user_name" type="xsd:string"/>
                <element name="mobile_number" type="xsd:string"/>
                <element name="mobile_operator" type="xsd:string"/>
                <element name="authorization_PIN" type="com:PIN"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="OperatorBillingUserOptions">
    <complexContent>
        <extension base="reg:UserPaymentOptions">
            <sequence>
                <element name="IMEI_number" type="xsd:string"/>
                <element name="MSISDN_number" type="xsd:string"/>
                <element name="user_settlement_type" type="com:SettlementType"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

Figure 14: WSDL Complex Type Definitions using Polymorphism

Each complex type may be extended by additional elements to add functional if required for an implemented payment system. These extensions will not break any existing implementation in the original system due to the extensibility mechanism of XML. This will enable the payment system developer to easily integrate system changes by simply adjusting the Web Service WSDL interface. These changes will have no effect on the existing framework implementation and each party using the Web Service will just integrate the updated version of the WSDL file into their system.

4.4 Summary

This chapter outlines the design of the Web Services framework. Using the distributed architecture of Web Services enables integration of the framework with existing mobile payment systems through language independence and use of existing Internet protocols.

Object oriented design techniques have been combined with the extensibility of XML, to improve component reusability and interoperability within the framework. This design solution enables the creation of various concrete payment systems from the basic framework specification.

The next chapter will describe the implementation and development of a few payment systems using the designed Web Services framework for mobile payment systems.

Chapter 5

Implementation of Mobile Payment Systems

The WSDL interfaces for the charging and registration Web Services set out a flexible template for the development of different payment systems. Using the design methodology described in the previous chapter, a generic framework was developed using Java as the implementation language. This framework was extended to provide the implementation of three back-end payment methods. These included the network operator billing payment, credit card payment and reverse sms payment methods.

This chapter will describe the development of the Web Services framework, in particular to the three heterogeneous payment methods. Following this, the implementation of two operational payment systems will be outlined including the development of mobile user and content provider applications.

5.1 Implementation Environment and Tools

This section outlines the software environments and tools used in the development of the Web Services framework and the two mobile payment systems. Given that the implemented solution is designed to improve interoperability and reusability in the mobile payment services domain, no proprietary technologies were employed in the development process.

5.1.1 Programming Language

The object oriented programming language, Java, was used for the majority of programming. The latest version of the Java development environment, Java 2 sdk 1.4.2, was used. The main advantage for developing with Java was its significant support for working with XML, and in particular WSDL. The Web Services SOAP environment, Apache Axis, is written in Java and has support for the deployment of a web service by exposing Java classes as the service. Apache Axis provides a tool for compiling a WSDL interface into Java classes. This tool is called 'WSDL2Java', which generates all the stubs, skeletons and data types necessary deploy and remotely locate and access the web service. Also Java delivers platform independence and, as will be discussed in a following section, has support for programming for mobile devices.

5.1.2 Apache Axis

The deployment of Web Services requires a framework for constructing SOAP processors such as clients and servers, which is called a SOAP engine. The SOAP engine used in framework is Apache Axis 1.1. The main advantage of using Axis, the Apache Extensible Interaction System, is that it is open source and designed to be very configurable, and also its implementation is written in Java.

Axis offers additional features of a simple stand-alone server, a server which plugs into servlet engines such as Tomcat, extensive support for WSDL, and a tool for generating Java Classes from WSDL files.

5.1.3 Apache Tomcat Web Server

The Tomcat 4.1 servlet container was used to install Axis and deploy the Web Services on a network. This web server was also used to deploy the content provider servlet applications in one of the developed mobile payment systems.

5.1.4 J2ME Wireless Toolkit

The Java 2 Platform, Micro Edition, Wireless Toolkit 2.0 [24] supports the development of Java applications that run on devices complaint with the MIDP 2.0. The Wireless Toolkit also supports the development of Java applications complaint with the Wireless Messaging API (WMA) and the Mobile Media API (MMAPI). The WMA specification supports SMS and CBS (cell broadcast messaging), which allows peer to peer messaging or client to server messaging.

Included in the Wireless Toolkit, is the KToolBar which is a simple development environment for compiling, packaging and executing MIDP applications. It is necessary to use an additional IDE for editing and debugging of Java source files. Also a selection of emulated mobile devices is provided for executing and testing of applications. The emulated devices are capable of emulating the features in the CLDC, MIDP, MMAPI, and WMA specifications. The reason for integrating the Wireless Toolkit into the implementation of the payment systems was primarily for the development of J2ME applications, and also version 2.0 of the MIDP specification is not yet supported on an available version of the Symbian platform.

5.1.5 Kannel SMS Gateway

To enable the deployment of one of the payment systems in the real mobile domain, the use of an SMS gateway was necessary for the framework implementation to send authorisation messages to a mobile device. An open source SMS and WAP Gateway was configured and integrated into the payment system. The gateway used was the Kannel gateway, version 1.2.1. Kannel is primarily an open source WAP gateway, but it also works as an SMS gateway for GSM networks. As explained later in this chapter, the Kannel gateway was integrated into the payment system to provide the communication link between the mobile user and the payment service provider.

5.2 Framework Implementation

The framework was implemented to support three different payment methods. The mobile user would have an option of choosing a payment method, if the service requested supports that payment method. Both the mobile user and content provider would have to be registered with the same payment method for the charging process to begin. A content provider would register with all the payment methods they wish to support for a particular service. The three payment methods implemented using the Web Services framework are operator billing, credit card and reverse SMS payment.

5.2.1 Operator Billing Payment Method

Operator billing represents the payment mechanism used by mobile network operators to acquire payment from their customers for services provided. A content provider using this payment method with the Web Services framework would be requesting that the network operator will bill the mobile user according when the service provided in being used. The network operator will then forward that payment on to the content provider after deducting a payment service charge. The amount to be billed may be deducted from a mobile user's pre-paid account or added to their next post-paid phone bill. The process of which way the cost of the service is transacted from the mobile user to the network operator should be transparent to the content provider. The payment service provider in this case may be the network operator or an independent organisation with access to the network operator's payment infrastructure. In the development of the framework the network operator infrastructure is simulated as it is unrealistic to be able to gain access to an enclosed network operator system for the purposes of a research project.

5.2.2 Credit Card Payment Method

Similarly to operator billing, credit card payment represents the payment mechanism used by credit card companies and financial institutions to acquire payment from their customers for services provided. Also in the development of the framework, the credit card payment infrastructure is simulated. Using this payment method, the mobile user will only enter their credit card details during the registration process. This process is assumed to be executed only once via a secure connection to the payment service provider and further communications will only involve the transmission of user and session ids, thus decreasing the security risk of exposing valuable information.

5.2.3 Reverse SMS Payment Method

Reverse SMS payment represents an alternative payment method not connected to network operator's infrastructure or financial institution payment systems. The mobile user wishing to use a service will pay for access to this service by being charged for receiving a SMS message, usually at a premium rate. This method of payment may be restricted to small payments for a single delivery or usage of a digital service, therefore certain next generation services may not support this payment option. Once the charging message is sent to the mobile user, then the amount will be either deducted from their network operator's pre-paid account or added to their post-paid bill. It may be in this case that the content provider may also be the payment service provider.

5.2.4 Development of the Web Services Framework

The first step in the development of the framework was to implement the web service defined by the WSDL interfaces. Apache Axis provides a tool for converting WSDL interfaces into Java service side skeletons, client side stubs and type objects. The client stubs were generated so that a web service client application can use instances of these objects to locate and bind to the web service and then remotely invoke the web service operations. Also, all the defined XML complex types are generated as serializable Java objects, usually beans, which can be used by both client and service when passing type information.

On the service side, the generated Java skeleton objects provided a blank template of the web service interface with all the defined operations represented by empty methods. The web service is now a façade interface to the functional components in the implemented framework. Services realising three different configurations of the framework were implemented three times to exhibit the functional of the previously described payment methods. The façade web service interface can interpret the destination of a client operation call by examining the value of payment type object, and therefore directing the information and request to the intended framework implementation for a particular payment service provider.

The implementation of the three payment methods uses the abstract factory pattern to create their specific concrete products. The implementations are similar in structure due to the generalisation in the framework design, but the passing and parsing of information was different due to the variant data required for each payment method. The back-end each payment implementation was comprised of separate database components, corresponding to the heterogeneous payment infrastructures.

5.3 *Real Mobile Network Payment System*

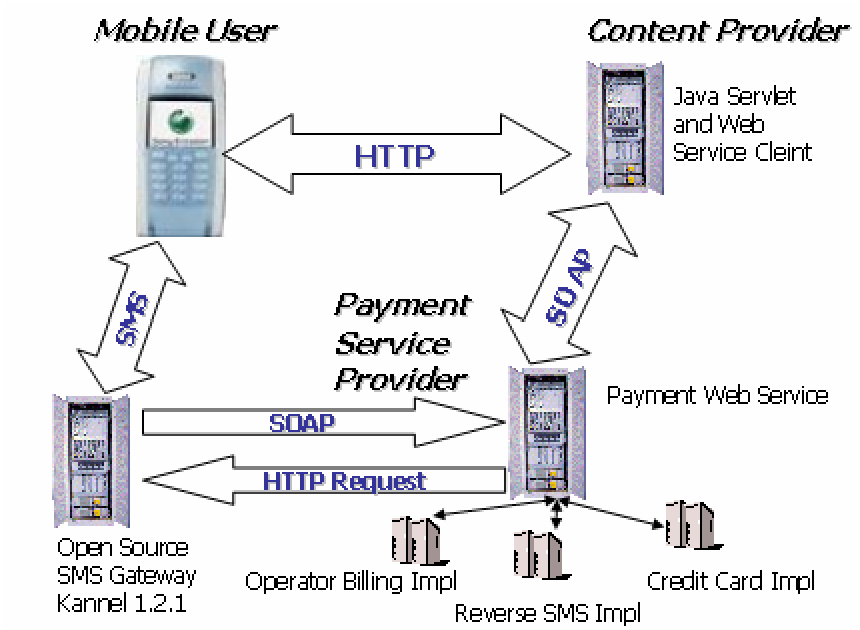


Figure 15: Topology of the Real Mobile Network Payment System

This mobile payment system, figure 15, was developed and deployed using a real GSM mobile network for communication. The overall system consists of three distributed systems, the mobile device operated by the mobile user, the content provider server and the payment service provider infrastructure.

5.3.1 Mobile Client – HTML and WML Browser

In this payment system the mobile user requests content via a HTML browser, with support for WML. Initially, before they begin to access the content, they will be asked whether they have registered with the payment service provider. If they have not, they will be redirected to start the registration process via the web browser on their mobile device. Communication to the content provider server is performed via HTTP requests and responses.

The registration process between the mobile user and the payment service provider in this payment system is transacted via the content provider, but in other systems this process may be performed directly between the two parties. The mobile user will be presented with the available payment methods and payment characteristics associated with each method. Once the user selects a payment method, they will be required to fill in the relevant details pertaining to that payment type. Following the completion of this registration process with one of the implementations of the payment service provider, the mobile device will receive a user id in the form of a cookie. This will be stored on the mobile device for future use in relation to the payment service provider. Now registration is complete and the mobile user can now request to access a service supported by the payment method which they have just previously registered with.

After the charging session is initiated, the user will receive a SMS message. This message will display the payment contract of the charging session to the user. They will be requested to then reply to this SMS with their four digit authorisation PIN. If this PIN is correct, then they should be able to return to the web browser where they can access the requested web content via a URL link.

5.3.2 Content Provider – Java Servlet Application

In this payment system, the content provider deploys the web content on a web server, which is access by sending HTTP requests to a Java servlet. All HTTP requests, which are payment related, are invoked as web service client operations using the generated stubs from the WSDL interface. This results in SOAP requests being sent to the payment web service. The content provider will have registered each service with each payment implementation of the framework for which the service has support. Any web service calls, resulting from a mobile user request, will propagate through the façade web service to the payment implementation of the mobile user's registered choice.

The web service client parses the information from the mobile user from HTTP request parameters to element values of the predefined XML complex types, which are generated as Java objects. Similarly, the return objects from the web service calls are parsed into html text for the mobile user's web browser. The content provider is some what of an intermediary for the mobile user and the payment service provider, until the charging session has been authorised and the service has begun. Depending on the terms of the payment contract, the content provider will request a charge operation to the payment web service following supply of the service to the mobile user.

5.3.3 Payment Service Provider – Asynchronous Authorisation

The payment web service was deployed on using the Apache Axis running on a Tomcat Web Server and it waits for SOAP requests from the content provider. The registration details of both the mobile user and the services of a content provider are parsed from SOAP requests to Java objects at the façade web service and propagated through the payment implementation being used. All the registration details are stored in the database for the relevant payment method.

At the onset of a charging session request from the content provider, a contract is produced by the framework from the registration details previously stored. This payment agreement must then be authorised by the mobile user before the charging session can proceed. In this payment system the framework implementation will call a SMS Gateway API which makes a HTTP request to the Kannel SMS Gateway server. The information sent to the gateway includes the mobile user's number, the session id number of that particular charging session and a text version of the contract object. At the SMS gateway the session id and the contract text are combined into the payload of a SMS text message and using the mobile number, the message is sent to the mobile user's device. On receipt of the SMS message, the mobile user will be requested to authorise the payment agreement. If the mobile user agrees with the terms of the contract, then they must reply to the SMS message with their four digit authorisation PIN, which was set at the registration stage. When the SMS gateway receives the returning message, containing the session id and the user's PIN, then these parameters along with the mobile number will be sent in a SOAP request back to the charging web service. The web service will process this information by comparing the user's entered PIN with the original registered PIN. If they are identical, then the user is authenticated. The charging session may now be started.

To account for network delays, especially during the SMS communication process, this payment system was developed to be asynchronous. On completion of the authorisation and authentication stages, the framework implementation will only update the database record of that charging session. The content provider will not be informed by the payment service provider when the charging session has been activated. An activation request will be initiated by the mobile user after they have authorised the payment agreement. This will involve the user accessing a URL link presented to them by the content provider prior to the authorisation stage. The URL will have the session id number appended as a parameter, which will be used by the content provider to reference the particular charging session when forwarding the charge request to the payment web service. If the charging session is active, then the charge request will be processed and a payment confirmation object will be returned. Otherwise, an empty payment confirmation will be returned and the user will be asked to complete the authorisation process or start again. When the charging session is active, the payment confirmation will contain information of how much the mobile user is being charged for the service. This will be displayed to the mobile user as a payment receipt.

Depending on the characteristics of the charging session, the final charge request, which may be the only charge request, will be accompanied by a release request from the content provider to the payment web service. Any remaining amounts to be settled will be processed and a final payment confirmation will be returned. The charging session will now be released and all the details will be removed for the active session in the database.

5.4 Simulated Mobile Payment System

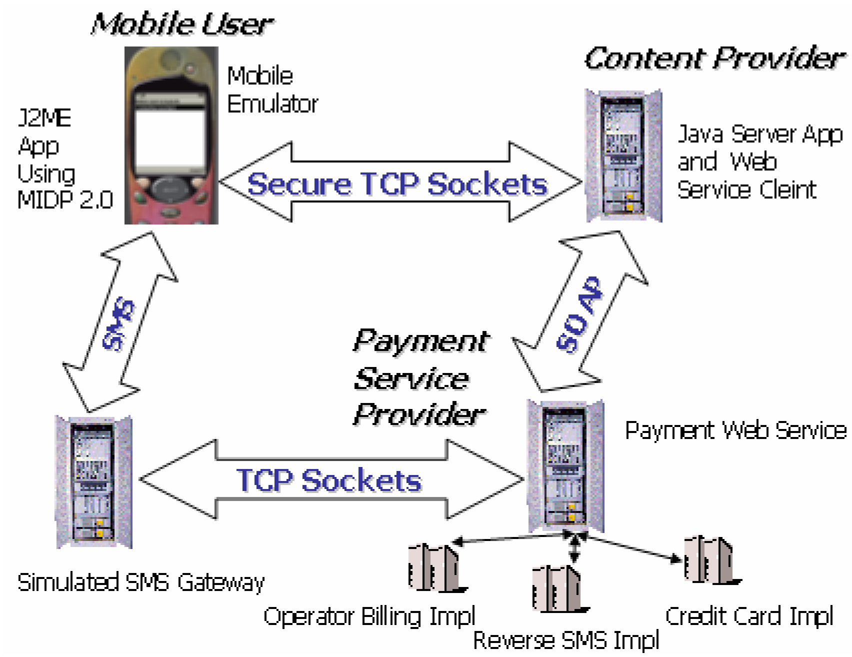


Figure 16: Topology of the Simulated Mobile Payment System

This is the second payment system, figure 16, developed using the implemented Web Services framework for mobile payment. The main difference with this payment system is additional functionality on the client mobile device due to the introduction of a J2ME application. This expands the communication capabilities, by using TCP sockets, between the mobile device and the content provider server and which also allows the integration of more complex services into this payment domain, such as video streaming services. Due to the current unavailability of device platform support for communication via TCP sockets with a J2ME application, the mobile client application was simulated on a J2ME emulator and the SMS Gateway was simulated as a simple server application.

5.4.1 Mobile Client - J2ME Application

In this payment system, the mobile can download two J2ME MIDlet applications, one for registration with a payment service provider and the other for access to a sample service offered by a content provider. Both applications use secure TCP socket connections, conforming to the Secure Sockets Layer (SSL) protocol, for communication to the content provider TCP socket server and via this server to the payment web service.

As described with the previous payment system, the registration MIDlet application communicates with the content provider server to obtain registration information and to return the mobile user's details. The MIDP application displays the payment methods available and once the user selects a method, they will be requested to enter their details in the fields provided. The parameters of the user id object are returned to the MIDP application in string format as TCP object streaming is not supported by MIDP 2.0. The user id parameters should be stored in persistent memory on the mobile device for further use in payment operations.

The service MIDlet application when launched starts a service request with the content provider. The sample service developed was a video streaming service. Video streaming over HTTP is supported by the Multimedia API provided by MIDP 2.0. The process to start a charging session is comparable to the previously discussed payment system. The only difference is that the client mobile application waits for confirmation that the charging session has been initiated after the SMS authorised process is complete. The application will automatically launch the video streaming service to the user when session confirmation is received. Another difference is in the charging model of the video streaming service, in which charge requests are sent at regular intervals during the usage of the service. The length of a charging interval and the amount of charging units in the interval are specified by the payment agreement.

5.4.2 Content Provider – TCP Socket Server

Only one outstanding contrast to the server application in this payment system compared to the previous system is that this application is a TCP socket server waiting to accept sockets connections from client application, while the other was a Java servlet running on a Tomcat web server waiting for HTTP requests. Also the communication between the mobile user and the content provider is on a secure connection, while with the previous system the connection is not secure.

5.4.3 Payment Service Provider – Synchronous Authorisation

In this payment system the same framework implementations are used with a change to the authorisation process. This process was the original design, but was found incompatible with the real network payment system due to latency issues of SMS messages. The synchronous authorisation process continuously waits for a change in a database value which depends on the authorisation reply from the user. The continuous waiting will timeout if the authorisation was not confirmed by a certain time. This process did not address any network latency and scalability issues, which then lead to the adoption of the asynchronous model, as described in the previous payment system.

5.5 Summary

This chapter outlined the implementation environments and tools integrated into the development of the Web Services framework and the two mobile payment systems. Three payment methods with implemented in the development of the framework using the abstract factory design patterns outlined in the previous chapter. Various technologies and programming techniques were used to develop the two working mobile payment systems.

The next chapter will outline and discuss the evaluation criteria resulting from the issues found in this dissertation.

Chapter 6

Evaluation

This chapter presents some of the criteria in which the implementation in this dissertation should be evaluated. Each section will discuss various aspects of the payment systems, with a focus on the integration of the Web Services framework into operational mobile payment systems. Also, this chapter outlines some further work which could be added to the results of this dissertation.

6.1 Design and Implementation

6.1.1 Network Failure and Latency

As with any wireless network, a mobile phone network is affected by transmission failures and high latency, which have to be accounted for in the design of systems which rely on these networks. Also system devices may suffer from communications or application failure, which will make vital services unavailable to network clients.

In the payment systems developed in this dissertation, the design addressed various failure issues which could affect a real life payment system. To avoid the effects of node failure, such as data loss or depreciated information, within the mobile payment system, certain techniques were employed to limit these problems. The design of the system on top of stateless web services means that state and session information must be persisted in the case of failure. All important data is stored in a database which is accessed by a client or session id. If a client mobile device was to fail or lose network coverage and messages were lost, then the persistently stored user id or session id can be used to re-transmit when the failure is rectified, thus re-establishing the failed session. If the content provider and the payment service provider servers were to suffer from failure, then replication of databases and redundancy of servers may address this problem.

Network latency precautions were not originally designed into the payment system, which is implemented as the simulated mobile payment system using the mobile emulator. In network simulation, latency is not an issue, but in the implementation of the real mobile network payment system it was found necessary to adapt some design features to support problems resulting due to late communication of data. The main area of change was in the process of user authorisation. Originally, the system was designed with a synchronous process for acquired user authorisation from the mobile user via SMS messaging. The payment framework implementation would wait for a reply from the user until the process could continue. Unpredictable latency of SMS transmission resulted in problems with the synchronous approach. An alternative asynchronous process was implemented into the system, which did not depend on the arrival of SMS messages. Once the original SMS message was sent from the payment service provider to the mobile user, the authorisation process would cease. Only after the user has replied to the SMS message and attempted to access the content service again, then the authorisation process would be re-launched and process would continue as before.

6.1.2 Security

Security is a major issue with mobile communications, especially concerning the transfer of payment related sensitive information via an exposed network. The growth of e-commerce depended heavily of effective secure communication; therefore the growth of m-commerce is also reliant secure payment transactions.

The registration process of the Web Services framework provides single transaction of sensitive financial information to the payment service provider in return for a unique user id, which can be used for numerous transactions with various registered content services. The two valuable sets of information which are necessary for a payment transaction are the user id, which contents various parameters including a unique number id, and the user's registered PIN. The registration process must be done using a secure connection, such as secure sockets or HTTPS. Further communication in a payment transaction will involve separate mediums for communication. The user id will be sent via a secure connection to the content provider and then forwarded to the payment service provider via SOAP, while the authorisation PIN will be sent via SMS communication. This separation of communication increases the protection against eavesdropping attacks. The implementation of the payment system supports secure TCP socket connections between the mobile user and the content provider, but no web service security was added. The lack of security in Web Services is still an issue, even with XML and SOAP encryptions as this does not cover end-to-end security. Web Services security is considered a large area of unresolved issues and was not intended to be within the focus of this dissertation.

6.1.3 Extensibility

Framework design with WSDL allows for declarative interfaces which utilise the extensibility features of XML. To maximise the general characteristics of a framework, it is essential that extensibility is at the core of the framework design. The WSDL interfaces designed for the Web Services framework use extensibility provided by abstract complex XML types to adapt the functionality of the three different framework implementations, thus it opens the framework design to any amount of implementations for diverse payment methods.

6.2 Future Work

Further work may involve the implementation of the framework with Enterprise Java Bean (EJB) technologies, the inclusion of Web Services security and the integration of more complex payment systems with the Web Services framework.

The frameworks implementations comprised of a basic Java object oriented environment of Java classes which are integrated with a simple SQL database. The implementation of an EJB framework will provide additional support for concurrency, scalability and data storage.

Web Services security is still inconclusive, especially concerning end-to-end system security. Although, there are some XML and SOAP encryption techniques available for application level security, there is still a need for further research to investigate the development of a complete security solution for Web Services.

6.3 Summary

This chapter outlined the evaluation criteria for the design and implementation of the Web Services framework for mobile payment services. Mobile networking creates unpredictable problems which must be addressed in the design stage of a payment system to avoid network issues upon deployment. Security is one of the major areas of importance when developing any payment system, especially an exposed mobile payment system. A combination of different communication techniques, secure connections and less frequent transfer of sensitive information are good improvements in the protection of a mobile payment system.

Chapter 7

Conclusion

The goal of this dissertation was to research the area of mobile payment and to design and build a generic framework for mobile payment services. At the time of publication of this dissertation there are various enclosed proprietary solutions for mobile payment, but currently there is no standardised approach to mobile payment services. Research of the domain revealed three main actors in a mobile payment system; the mobile user, the content service provider and the payment service provider. From examining existing mobile payment systems it was observed that there are common relationships between each actor, which can be expressed as a generic set of interactions between the actors. To apply these interactions to a generic solution, which would be suitable with various payment implementations, a framework design was developed as an extensible template for the solution. In addition to extensibility, there is a requirement for an open and interoperable solution, which the distributed architecture of Web Services provides. With the combination of web based technologies and well known design patterns, a generic Web Services framework was designed and implemented for three diverse mobile payment methods. To further evaluate the framework, two developed mobile payment systems are integrated with the diverse framework implementations, providing a generic solution for a mobile payment system.

During the initial stages of research for the dissertation, a lot of effort was focused at accumulating information from existing mobile payment systems, but due to the enclosed proprietary nature of the domain, this became difficult. The final design was in part realised from public information describing proposed common platforms for mobile payment. Other difficulties encountered included restrictions of current mobile client application capabilities and network latency issues, which were addressed through design adjustments.

Future development of this dissertation may entail a more complete solution with an improved commercially accepted implementation of the framework and integration to real mobile payment networks.

In conclusion, the proposed design and implementation of a Web Services framework for mobile payment services does not provide a complete solution for all mobile payment systems, but suggests an open, extensible and interoperable solution for the development of mobile payment services, thus driving the mobile domain in the direction of a common generic platform for payment services.

Bibliography

[1] Amit Vyas, Peter O'Grady, "A Review of Mobile Commerce Technologies", Department of Industrial Engineering, University of Iowa, May 2001

[2] Kathleen De Clercq, "Dividing the pie of mobile payment revenues: opportunity or threat to the traditional banking sector", Lessius Hogeschool, 2001-2002

[3] Digital 4Sight Corporation, "NTT DoCoMo's I-Mode", White Paper, 2001

[4] GemPlus, "SIM Toolkit", White Paper, version 3, May 2001

[5] DIGIA Inc., "Programming for the Series 60 Platform and Symbian OS", Helsinki, 2003

[6] Trintech, "Mobile Payments: Preparing for the mCommerce Revolution", White Paper, March 2002

[7] M-Commerce, The Sunday Business Post, "Paying by text, down to a tea", November, 2002

[8] Consult Hyperion, "Merging Models for Mobile Commerce", White Paper, May 2002

[9] TelecomWeb, "Mobile Gaming Market needs Winner in Payment Space", E-business, March 2003

[10] Wysdom, "www.wysdom.com", May 2003

[11] PayCircle, "www.paycircle.org", May 2003

[12] NorthStream, "Charging Mechanisms for Mobile Services", White Paper, 2002

[13] VendorZone, "www.mobilepaymentsworld.com", May 2003

[14] ICSTIS Guideline, "Reverse-Billed Premium Rate SMS", Technical Paper, May 2002

[15] Mobtastic.com, "Reverse Charge SMS", Technical review, May 2003

[16] Enda Cahill, Goran Matic, John Keane, Martin Commins, "Bluetooth and 802.11: co-existence or conflict?", Trinity College Dublin, April 2002

[17] David Buhan, Yu Chye Cheog, Cheng-Lin Tan, “Mobile Payments in M-Commerce”, Telecom Media Networks, September 2002

[18] GSM World Technology, “www.gsmworld.com”, June 2003.

[19] John Scourias, “Overview of the Global System for Mobile Communication”, Technical Paper, May 2003

[20] Netsize, “European SMS Guide”, Mobile Business and Entertainment, February 2003

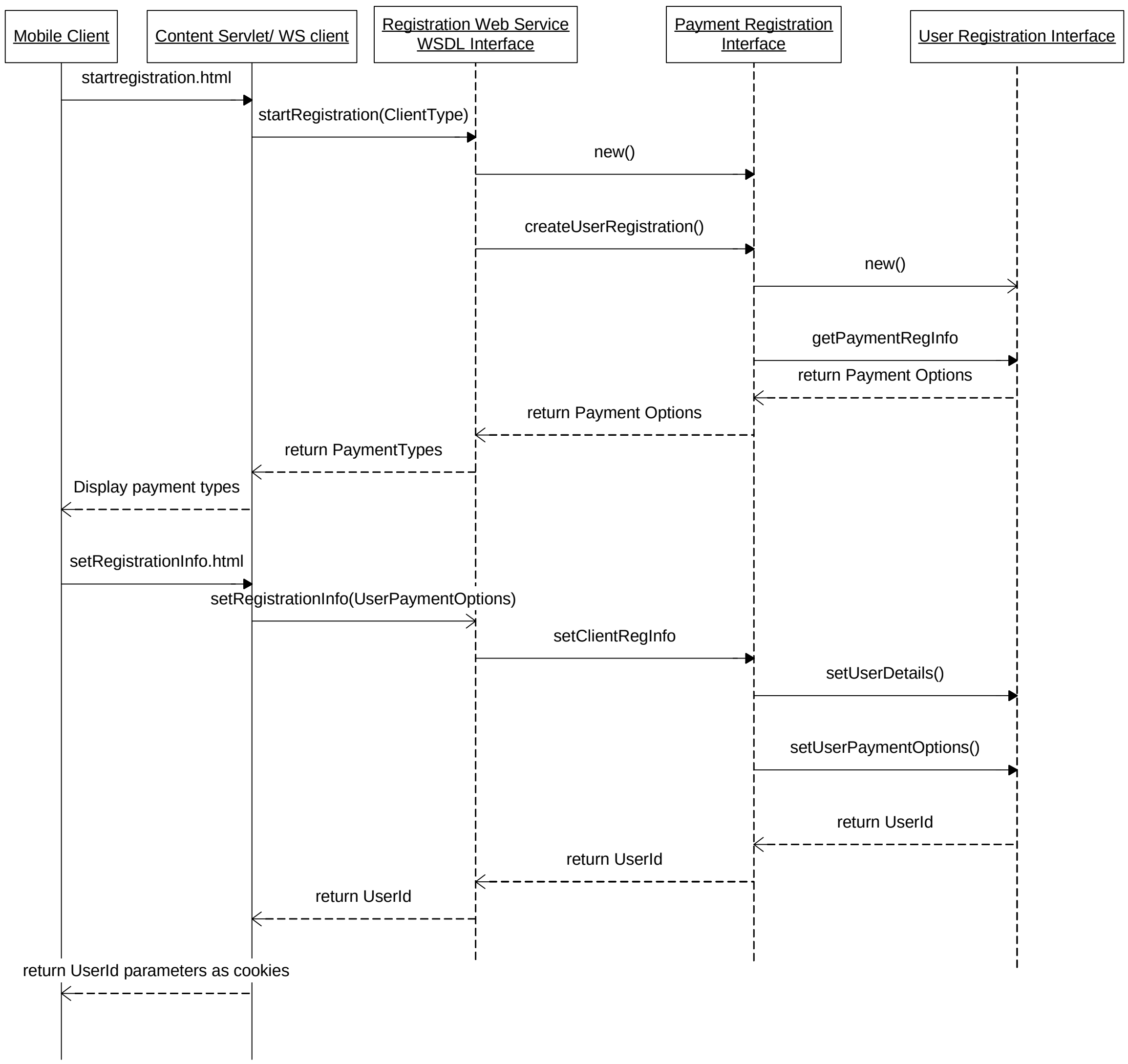
[21] Eric Newcomer, “Understanding Web Services”, Addison-Wesley, 2002

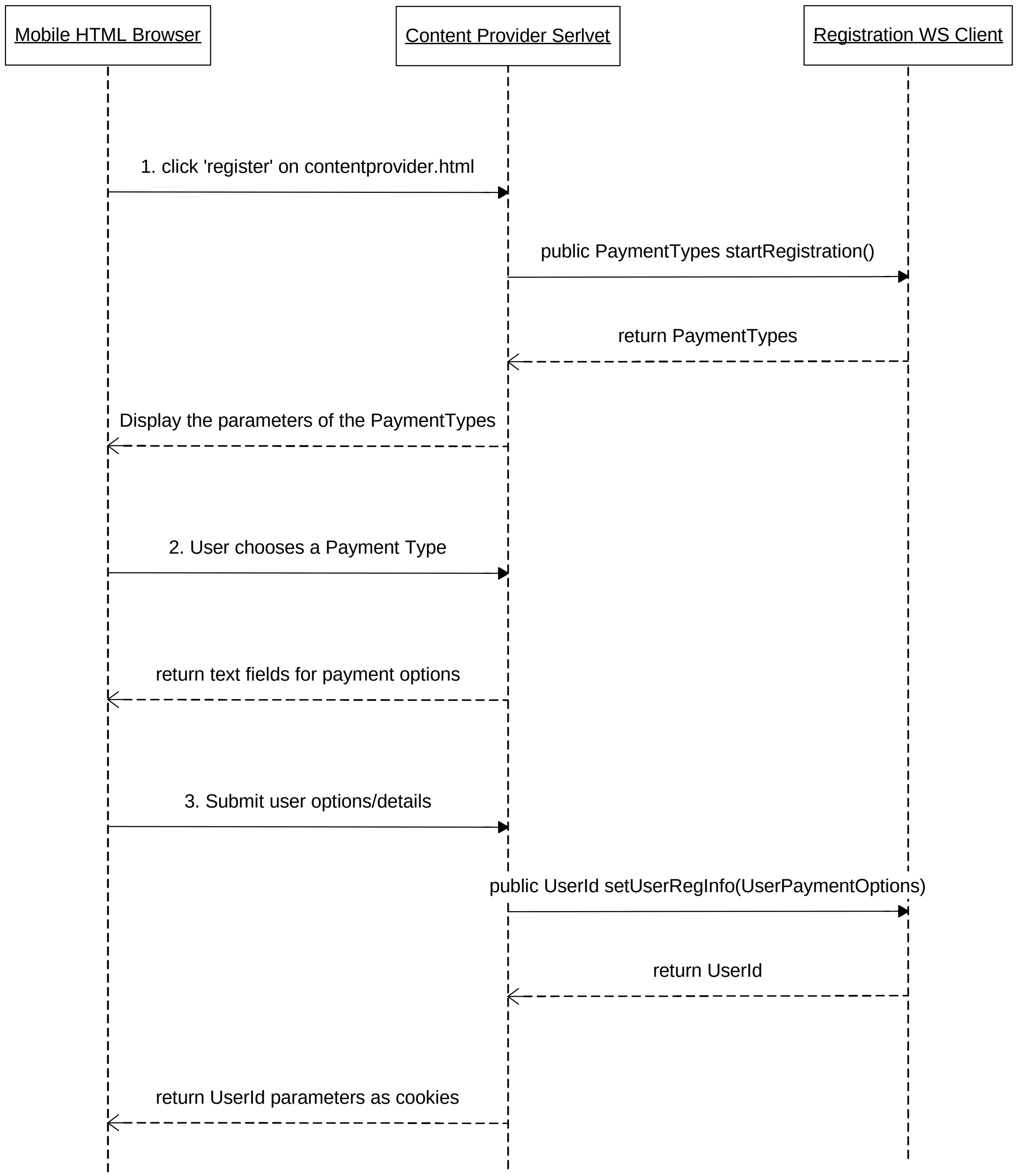
[22] Gamma, E., Helm, R., Johnson, R., Vlissides, J., “Design Patterns: Elements of Reusable Object-Oriented Software”, Addison-Wesley, 1995

[23] Timothy Budd, “An Introduction to Object-Oriented Programming”, second edition. Addison-Wesley, 1997

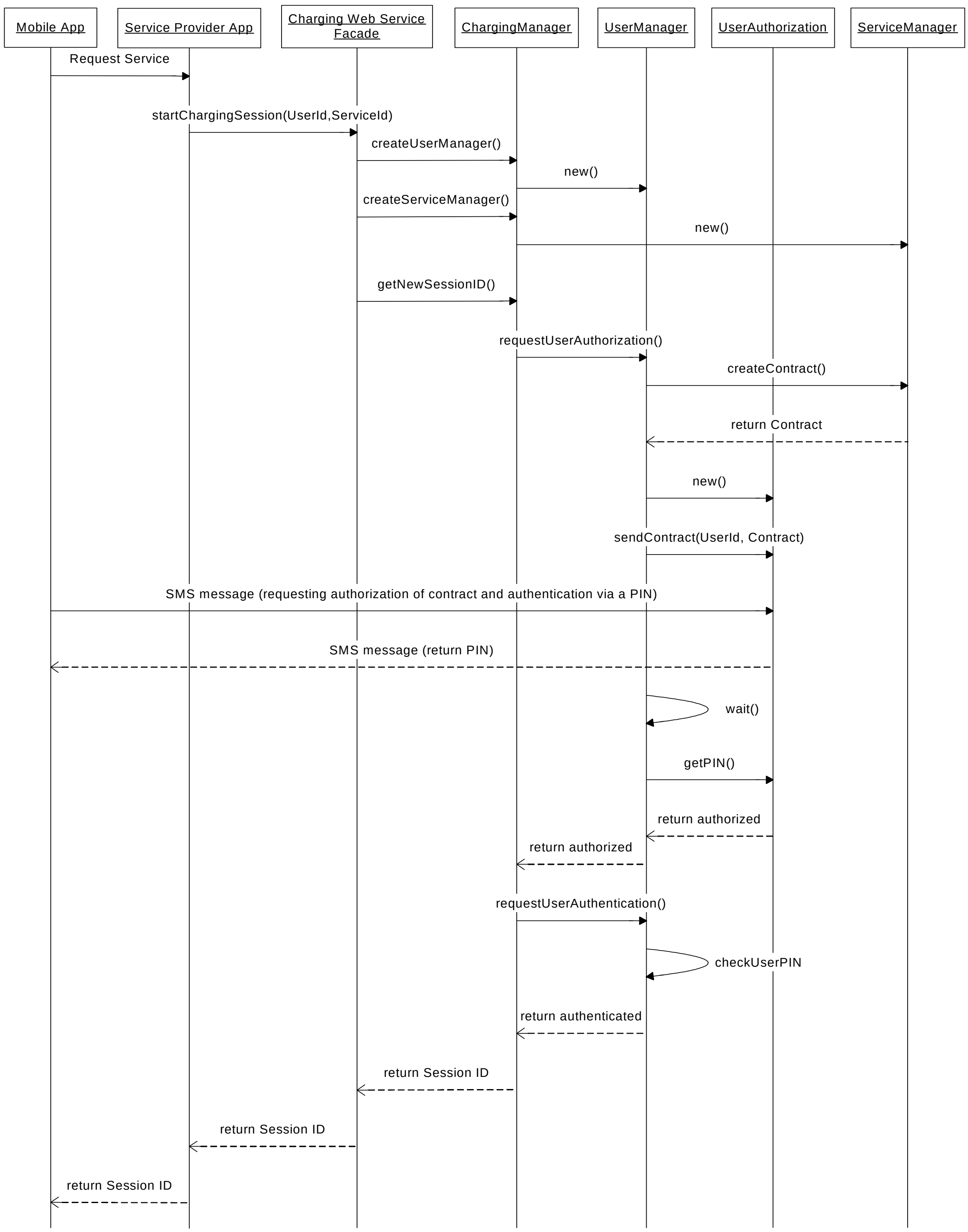
[24] Sun Microsystems, “J2ME Wireless Toolkit User’s Guide”, version 2.0. March 2002

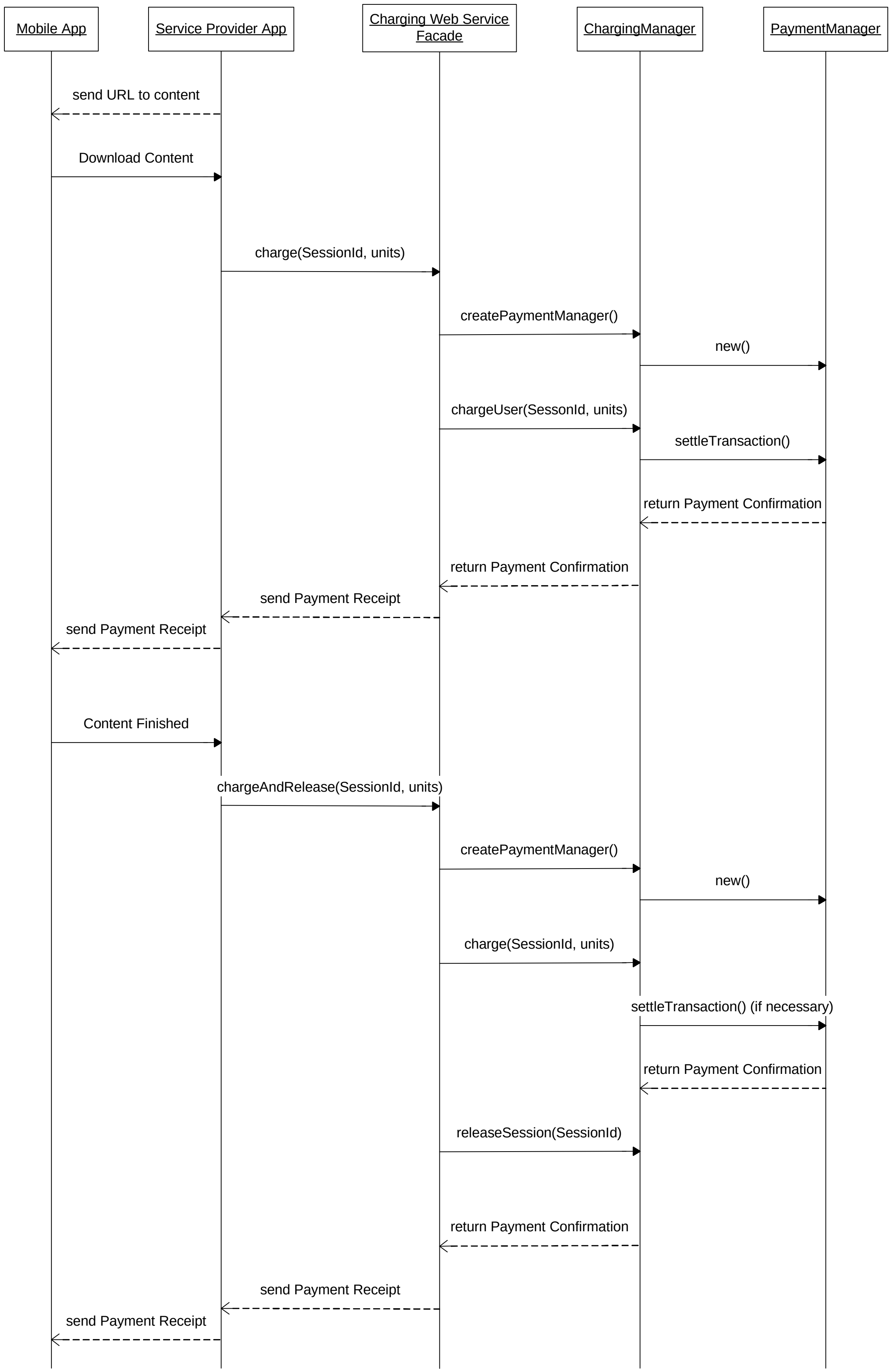
Appendix A – Registration Sequence Diagrams

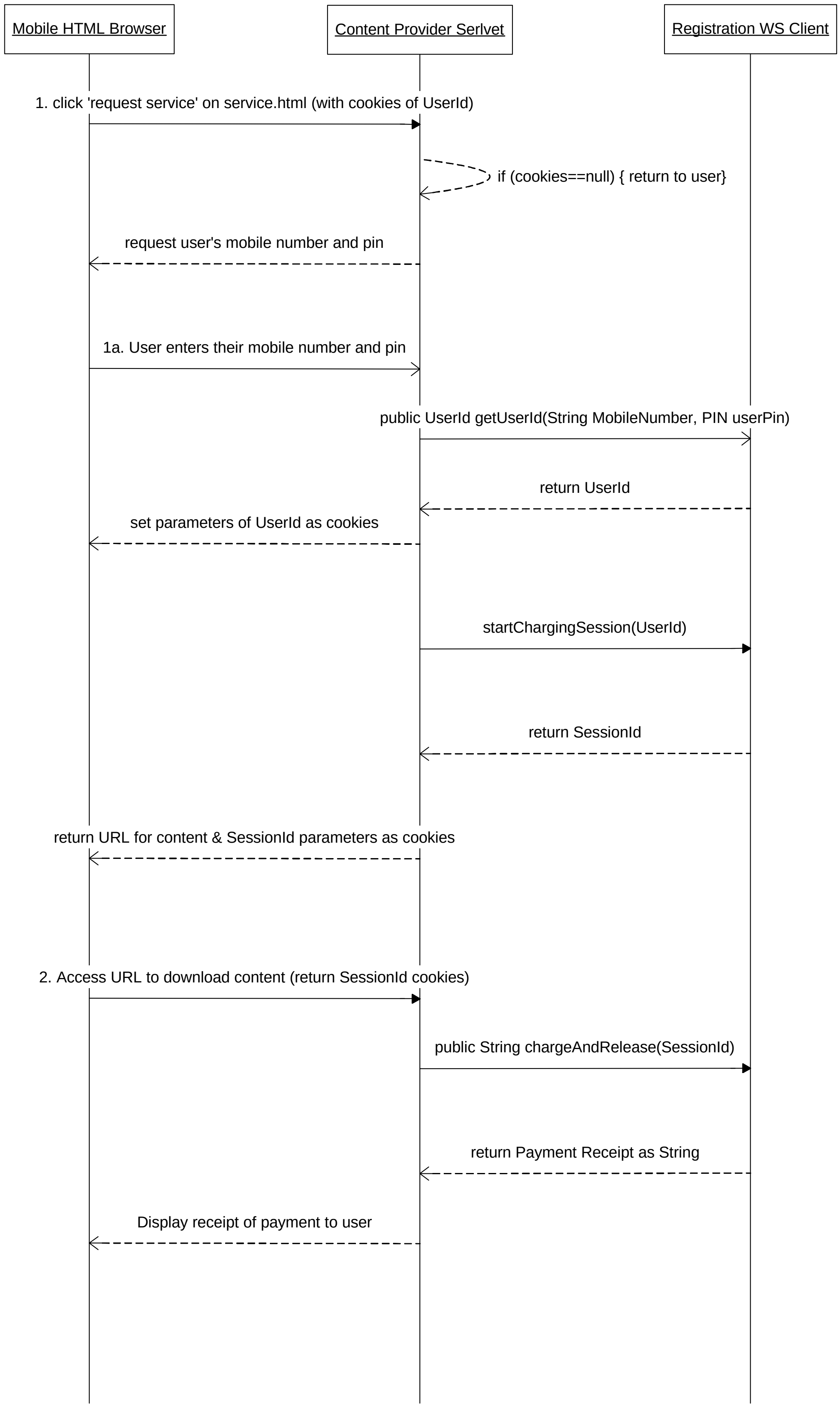



Mobile HTML Browser
Content Provider Serlvet
Registration WS Client
1. click 'register' on contentprovider.html
public PaymentTypes startRegistration()
return PaymentTypes
Display the parameters of the PaymentTypes
2. User chooses a Payment Type
return text fields for payment options
3. Submit user options/details
public UserId setUserRegInfo(UserPaymentOptions)
return UserId
return UserId parameters as cookies

Appendix B – Charging Sequence Diagrams




Mobile App
Service Provider App
Charging Web Service Facade
ChargingManager
PaymentManager
send URL to content
Download Content
charge(SessionId, units)
createPaymentManager()
new()
chargeUser(SessonId, units)
settleTransaction()
return Payment Confirmation
return Payment Confirmation
send Payment Receipt
send Payment Receipt
Content Finished
chargeAndRelease(SessionId, units)
createPaymentManager()
new()
charge(SessionId, units)
settleTransaction() (if necessary)
return Payment Confirmation
releaseSession(SessionId)
return Payment Confirmation
send Payment Receipt
send Payment Receipt


Mobile HTML Browser
Content Provider Serlvet
Registration WS Client
1. click 'request service' on service.html (with cookies of UserId)
if (cookies==null) { return to user}
request user's mobile number and pin
1a. User enters their mobile number and pin
public UserId getUserId(String MobileNumber, PIN userPin)
return UserId
set parameters of UserId as cookies
startChargingSession(UserId)
return SessionId
return URL for content & SessionId parameters as cookies
2. Access URL to download content (return SessionId cookies)
public String chargeAndRelease(SessionId)
return Payment Receipt as String
Display receipt of payment to user

Appendix C – Static Diagram of the Framework for Charging