

# Card-based Macropayment for Mobile Phones

Philip Garner, Reuben Edwards, Paul Coulton  
Lancaster University  
p.garner, r.edwards, p.coulton @lancaster.ac.uk

## Abstract

*Most new entrants into the mobile commerce marketplace are faced with a paradox; how can we attract users without merchants, and how can we attract merchants without customers? Without a huge investment in marketing to build consumer recognition, starting a new m-commerce service is risky. In this paper we analyse how successful companies on the web have managed to create successful payment services by utilising the brand recognition and ubiquity of the popular card networks. Attempts to marry credit cards and mobile phones are not new – encrypted phone wallet systems have been bundled with devices for a number of years. However, use of these wallet systems remains limited as the public failed to embrace shopping via WAP, frustrated by slow speeds and poor user interfaces. With mobile terminals more suited for browsing, and higher speeds from mobile data becoming the norm, we present a system to enable remote macropurchases using existing credit card technology.*

## 1. Introduction

Mobile commerce continues to attract academic study and interest from the business and banking communities but mobile terminals remain underused as a means of paying for goods and services.

As is so often the case with mobile technologies, the successful business models for transactions on mobile terminals have confounded analysts' predictions, with proximity payments stalling and the market for mobile content delivery booming. The UK market for content delivered over mobile networks, such as ringtones, games and wallpapers is estimated to be worth a huge £740m per year [1]. Contrast this with turn-of-the-21st Century predictions that over half of all mobile payments would be either in retail stores or via vending [2], and the need to re-evaluate our expectations of mobile commerce is clear. The space where mobile terminals were expected to dominate, proximity payments, is now being filled with RF-enabled smartcards, such as the Octopus card popular in Hong

Kong, and the Oyster card used in London for public transport. Both of these systems are known as stored value cards, and they can be topped up at compatible kiosks. These systems both require user to physically touch (or at least come to within a few centimetres of each other) a receiver, but with contactless cards, using Near Field Communications, (NFC) even this is not always necessary. Ideal for mass-transit applications, users can potentially pay for their travel without even removing the card from their wallet. Of course, with the modern push towards convergence devices, it seems likely that such technology will become embedded into a mobile terminal, as seen with the Mastercard PayPass project [3]. This is expected to become a pervasive technology, with 50% of mobile terminals expected to be RFID/NFC enabled by 2009 [4].

Whilst the future for proximity payments for mobile appears bright, the models originally touted for m-commerce flounder. With the popular introduction of the Wireless Application Protocol (WAP) in 2000, an implementation of SSL/TLS was provided to allow 'secure' remote purchases. One of a number of public relations blows to befall the WAP concept was the 'WAP Gap' – the necessary bridge between Wireless Transport Layer Security (WTLS) on the wireless internet and Secure Sockets Layer (SSL) that was present in the early implementation where an encrypted session from WTLS had to be decrypted to plaintext before being encrypted for an SSL session [5]. Of course, this meant that end-to-end security was compromised, although the actual risk of a successful attack remains small, as the plaintext would usually only be present in a mobile network operator's (MNO) network.

In contrast the to the experience with WAP, the proliferation of SMS driven services continues to invigorate mobile commerce throughout Europe, with premium SMS services expected to deliver 12% of non-voice operator revenue by 2007 [6]. Users have adopted the technology with a vigour that surprised most in the mobile community. Its success as a platform for m-commerce completely belies its fundamental lack of suitability for transactions. Messages do not have guaranteed delivery and can be

delayed for days, are repudiable and offer little assurance that the person who initiated the transaction is the one paying. Despite all this, not only are premium SMS services being used to pay for ringtones and games, but for other, more interesting models such as information service subscription and ticketing. While SMS is dominant for remote micropurchases (typically less than €10), it is unsuitable for higher priced goods and services, requiring a number of messages to complete a transaction and being prohibitively expensive to merchants, with SMS-operators retaining up to 50% of the transaction.

A number of factors can be identified when attempting to explain the success of SMS as a payment platform:

### **Installed user base**

SMS services are available to most GSM handsets from the mid 1990s onwards; with handsets from 2000-on supporting downloads via WAP, used for the delivery of ringtones, games and wallpapers. This means that virtually all of the handsets in use in the GSM ecosystem can be used make premium SMS transactions.

### **Ease of use**

SMS use has accelerated strongly in the UK since true cross-operator messaging appeared in 1998. Nearly all users will be comfortable using SMS, so applying the same protocol to payments ensures a high degree of users willing to use premium SMS to purchase goods and services.

### **Merchant Opportunity**

Unlike other m-commerce solutions, integrating SMS payments into merchants' business models is relatively inexpensive. Also, premium SMS does not suffer from the crucial 'chicken-and-egg' problem of most new m-commerce systems, where the success of the system is dependant on the client uptake, and of course, merchants are reluctant to commit to any new payment system without a critical mass of users. With premium SMS m-commerce, the users are already in place and in huge numbers.

How to beat the odds and create a successful payment system, attracting clients and users, has been the subject of considerable research. A number of high profile enterprises have invested heavily in personal payment systems only to later withdraw them citing low

user uptake. In this article we will consider the need for mobile payments to follow the model established on the web, where entrenched card payment schemes have fought off challenges from web-centric payment systems. The success of payments driven by premium SMS should illuminate the critical factors for remote mobile macropurchasing (typically non-impulse purchases greater than €10). Despite all the technical advantage of these ultimately unsuccessful payment solutions, they failed to provide an operating model most users felt comfortable with. With this in mind we propose a system facilitating card-based payments on today's mobile terminals, and discuss some of the most suitable applications for such a system.

## **2. Card Payments on the Web**

Despite numerous attempts to create a successful micropayment solution on the web, card payments are so entrenched in the offline space that online payment systems that did not successfully integrate cards into their business model have failed. The two most often discussed are Flooz and Beenz, both of which collapsed in August 2001 [7]. Bridging the gap between voucher payment and traditional payment systems, users were free to spend Beenz at participating web merchants, or transfer them to another Beenz account holder. However, users were not able to convert any unused Beenz into a traditional currency, and when the service was dramatically suspended many users lost a significant amount as retailers refused to accept Beenz as a means of payment. Four years on, no other payment mechanism has managed to challenge the dominance of card payments online, despite the clear need for a workable micropayment solution.

PayPal is the most successful of a number of online payment systems, but for many transactions simply acts as a way of enabling person-to-person transactions with their credit or debit card [8]. It also enables merchants to economically add web storefronts, as set-up costs are significantly lower than performing card-processing with a traditional business bank account. By leveraging entrenched card technology that has a high level of consumer trust into the non-traditional payments arena, PayPal has been a tremendous success with \$19Bn. total payment volume in 2004 and over 70 million user accounts [10].

Some banks have partnered with card networks to improve the security of card transactions by offering one-time use credit card numbers for use when buying online. In the US, Citibank, Discover and MBNA have implemented a system known as "controlled payment numbers" which allows cardholders to shop online

without using their real card number. Despite warnings from banks and card network operators, many merchants retain card numbers on their servers. This means that when a merchant database that holds live credit card numbers and is compromised by a hack attempt, losses due to fraud can be substantial. By using a one-time card number, any numbers gained from a compromised database will not be able to be used in any further transactions. This kind of implementation requires the end-user to download some software to the desktop in order to generate unique card numbers associated with their real card. In some ways this improves the way the user makes the transaction, in that they are never forced to leave their computer in order to find their wallet and retrieve their card, interrupting the transactions and reducing the likelihood it will complete successfully. Such schemes have not been widely adopted however, and are disliked by merchants as they prevent the questionable practice of storing credit cards in order to easily facilitate repeat transactions. American Express removed their implementation of a similar system called “Private Payments” in April 2004, citing a lack of sufficient adoption [9].

Card fraud on the web continues to be a problem for card networks, with £505m lost to fraud in the UK during 2004 [11]. Some technical features have been added to modern credit cards in order to further secure ‘Cardholder Not Present’ transactions, such as the addition of Card Verification Value (CVV) digits. Not stored on the magnetic strip, CVV is intended to ensure that whoever is initiating the transaction actually holds the card in their possession. Unfortunately as most modern card harvesting techniques involve tricking users into submitting their card details to what they believe to be a trustworthy source (known as phishing) and thus any attacker is likely to be able to attain the CVV digits, rendering the system ineffective. Card payments on the web are by far the most popular way of paying for goods and services online, with 22m consumers using cards to buy online in the UK in 2004 [12]. The heavy investment in smartcard technology rolled-out throughout Europe during 2005 has been relatively successful in reducing card fraud in the offline world, but the additional security gained from smartcard implementation does nothing to prevent phishing attacks or mitigate other card security threats online.

### 3. Mobile Wallets

Business models mooted with the launch of WAP services suggested users may use a mobile wallet

solution which would allow WAP sites access to card details stored on the terminal authenticated with a user PIN. A good example is the Nokia Wallet, first implemented in the Nokia 6310 phone from 2002, and present in most models since.

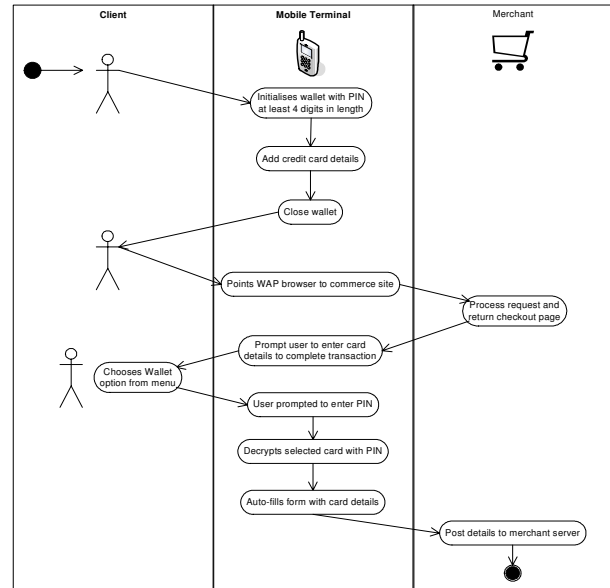


Fig. 1. UML diagram showing usual mobile wallet functionality.

#### 3.1 Client-side Wallets

Client wallets store payment information on a user’s mobile terminal. They may integrate with the user’s SIM in order to authenticate the user. Client side wallets are conceptually appealing to users, as they are analogous to credit cards in the responsibility for keeping them safe relies on the user. Wallets operating in hardware have a number of disadvantages, namely that such systems are very hard to update once deployed, and are usually associated with a single bank or network operator.

#### 3.2 MNO Server-side wallets

Hosted wallets mean that users’ payment details are not held on their device, but rather on a server operated by the Mobile Network Operator. This server may store card details or other methods of payment for use with mobile commerce transactions. For the user, such a system has a number of benefits. If their device is lost or stolen, they can still easily use their account on another device, unlike with a SIM-based system. Also, MNOs are able to easily update the services offered by the wallet solution, perhaps adding new payment

methods. MNOs are in a strong position to provide such services, with great experience in pre-pay and post-pay consumer billing. Also the consumer already has a trust relationship with the MNO, a factor many new m-commerce startups have found insurmountable.

### 3.3 Third-party server side wallets

Although most wallet-based solutions developed to date have been provided by MNOs, third party wallets add a critical feature; interoperability. The Vodafone m-pay system is limited to customers of Vodafone and works only on pre-approved m-commerce sites. This kind of restriction clearly limits the potential of an MNO-run server, whereas a third-party run system is free to recruit as many users and partners as possible.

## 4. Macro-Purchasing for Mobile

Consumers are now buying goods and services online more than ever, and growth in e-commerce transaction value continues to accelerate. In the second quarter of 2005, US retail e-commerce spending grew a staggering 26% over the same period in 2004, with online sales accounting for 2.1% of total sales [13]. In the UK, debit cards dominate offline purchases, with two thirds of card sales made with debit cards. This trend is reversed online, with 72% of all online sales made with a credit card [12]. These figures underline the dominance of card payments for online transactions. Consumer protection schemes with credit cards are significantly more robust than with debit cards; if card users dispute a card transaction, card companies usually refund the customer quickly and leave retailers to pursue the lost income. An interesting trend is some high-volume e-tailers, such as low-cost airlines, have introduced a surcharge on users paying with a credit card rather than a debit card. This reflects the charging scheme imposed on e-tailers, where often credit card processing fees are between 1% and 5% and debit cards are charged at a fixed fee.

Presently very few e-commerce websites are designed to allow mobile users to make purchases via a mobile browser. This is mainly due to the unfavourable trading conditions on consumer platforms, with difficulty ensuring a consistent user experience across mobile browsers, coupled with the high cost of mobile data, a particular problem in Europe. However, both of these problems may soon be resolved. QVGA (320x240) resolution is emerging as the de-facto standard screen size for mobile, catalysed by the adoption of the standard for the video-enabled iPod, and the associated video delivery platform. Unlimited

mobile data has been available in the US for some time, and have started to appear in Europe at a consumer-oriented price point. Also the success of the Opera mini, the mobile browser from Norwegian developer Opera Software, has gone some way providing a consistent platform for mobile browsing. As of April 2006, the Java-based browser had 2m downloads and 4m page impressions per day [16]. We can see from the browser wars of the mid-1990s on the desktop how important having a standardised platform is for growth; with Internet Explorer 4 gaining a majority market share, enterprises were able to invest in websites reasonably ensured of a consistent user experience. We expect the same to be true for the mobile internet when screen sizes, browsing technology and mobile data costs converge.

In anticipation of this convergence we have developed a solution that allows developers to easily integrate an m-commerce component into their mobile applications.

## 5. XEPS: SSL-Enabled Payments for Mobile

A number of banks use payment processing software from ClearCommerce Corp. to allow their customers who want to take credit card payments online to integrate payments into their e-commerce solution. ClearCommerce provides an XML payment API for e-commerce website builders to send complete information about a transaction to a capturing server at the associated financial institution. Typically an e-commerce webserver will generate an XML document associated with a transaction, including details about the products in the order, the delivery address, total price etc. This document is then sent to the bank's processing servers, often via a HTTPS (Secure Sockets Layer-enabled) POST form. Merchants are able to define specific security rules that control a transaction, specifically using the Address Verification Service (AVS) to determine whether the delivery address matches the address registered with the card account. Further, advanced fraud detection rules can prevent transactions continuing given suspicious buying patterns such as a card being used to repeatedly buy high-value items. After processing, the status of a transaction is then returned to the merchant via an XML document, which must then be parsed.

The XEPS (XML-Enabled Payment System) project leverages this system onto a client-side mobile device, providing a framework for developers to integrate credit card payments into their mobile applications and complete transaction entirely from the device.

This project uses Python for Series 60 [14] Nokia mobile phones in order to rapidly develop the concept on a mobile device. Python allows for much quicker development for applications that need greater interactivity with the device hardware than with either Java (J2ME) or Symbian environments. With the 1.2 release, finalised in October 2005, a number of new APIs were available to developers, including: GPRS networking with HTTPS support, graphics and sound recording. However, Python for Series 60 (PyS60) does not offer the complete XML package associated with standard Python releases. Fortunately, researchers at the Helsinki Institute for Information Technology have ported the pyexpat Python XML module from the full Python release for use with Python Series 60 [15]. Using this module, well formed XML documents can be created on the device, and using the appropriate function calls elements and attributes can be placed as required.

The system provides real-time card processing, with transactions being compared to pre-configured fraud rules with a user being informed of the transaction status within seconds, whether it succeeds or fails.

The ClearCommerce XML specification used in this project specifies as huge array of inputs that can be used to describe a transaction, such as a number of elements for describing repeat transactions that will rarely be used. Conceivably a complex transaction could be described with hundreds of lines of XML, but most m-commerce applications will only require basic information such as card details, totals, and a product description. However, the application was tested using the most verbose output XML that may be used to describe a transaction. This is quite a sizeable document for a device with only limited power to generate, approximately 200 lines of XML. From the results shown in table 3, the current prototype implementation takes a rather slow 65 seconds to simulate a complete transaction on a modern Nokia smartphone. However, this figure includes a number of complex steps:

1) Generating a typical XML file

As demonstrated in table 1, this achieved relatively quickly and only contributes 5 seconds to the total transaction time.

2) Initiating GRPS session

This establishing a connection on the 2.5G packet networks used in GSM is often time consuming. On some devices the connection remains permanently active. This is expected to drastically reduce the transaction time.

3) HTTPS handshake

Only recently have smartphones (not more advanced PDA-style devices) shipped with certificate authority (CA) root certificates to facilitate an HTTP connection using SSL or TLS encryption. This is illustrated in Table 3, where the Nokia 3650, an early Symbian smartphone, is unable to complete the tests due to missing HTTPS support.

### 5.1 Program Flow

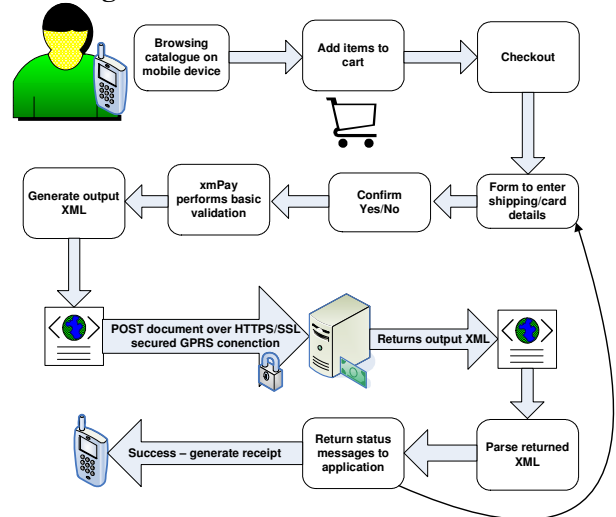


Fig 2. Typical XEPS use case scenario

Figure 2 describes the flow of information throughout a successful transaction. The actual implementation is dependant on usage, although it is envisaged that most applications will function in a similar way, with application developers providing a shopping cart and checkout front end, employing XEPS to power the payments back-end.

Upon completion of the successful transaction, the acquiring server returns a transaction ID to the device, which can be used by the user as a reference when querying the transaction. Although not yet implemented, using the returned information from the processing server to retain a receipt for users would be highly beneficial; users could easily see a log of their spending on the device, in contrast to awkward and easy-to-lose paper receipts.

### 5.2 Use Case Scenarios

The XEPS software is available as a Python module, developed as a proof of concept. It is envisaged that a release version would be suitable for inclusion by application developers in commercial software.

Platform	Time (seconds)
Nokia 3650, Python 2.2.2, Symbian 6, S60 v1.0	8
Nokia 6600 Python 2.2.2, Symbian 7, S60 v2.0	8
Nokia 6630, Python 2.2.2, Symbian 8, S60 v2.0	5

**Table 1: Time taken to generate the minimal XML required for a successful transaction.**

Platform	Time (seconds)
Nokia 3650, Python 2.2.2, Symbian 6, S60 v1.0	16
Nokia 6600 Python 2.2.2, Symbian 7, S60 v2.0	13
Nokia 6630, Python 2.2.2, Symbian 8, S60 v2.0	7

**Table 2: Time to generate complete XML document.**

Platform	Time (seconds)
Nokia 3650, Python 2.2.2, Symbian 6, S60 v1.0	No https support
Nokia 6600 Python 2.2.2, Symbian 7, S60 v2.0	75
Nokia 6630, Python 2.2.2, Symbian 8, S60 v2.0	65

**Table 3: Time to generate minimal XML file, initiate GPRS connection, establish HTTPS connection with remote webserver, POST generated document and write returned XML to local file.**

With more consumers using their mobile device to browse the web as conditions improve with better browsers, interfaces and connectivity, it is expected that more people will eventually use the mobile web to perform card-based transactions much as they do on the desktop, despite the drawbacks. In fact, reports suggest that as many as 28% of mobile phone owners will use their device to access the web, an increase of 24% over 2004 [17].

However, there are a number of scenarios where card payments through a browser would not be the ideal solution. Standalone applications, such as mobile games, would be able to integrate card payments to add value; a user could choose to unlock more levels on a game and pay for them directly from the application with a credit card. Integrating XEPS could provide such functionality, allowing developers to completely customise the interface and security levels to integrate with their application.

In another possible scenario, an enterprise could deploy a version of its catalogue to customers offline to their devices, and with XEPS, easily integrate online sales directly from the device, allowing users to browse a product catalogue without having the cost and inconvenience of using mobile data networks. Whilst the situation is improving, browsing a large product

catalogue using mobile data would be slow and expensive when compared to a catalogue stored on the device. Additionally, mobile browsers do not allow for sophisticated user interface controls that would make a product catalogue much easier to use on a constrained device, such as softkey-accessible popup menus. For a product catalogue that is regularly updated, when the device has access to a suitable broadband connection, updates could be downloaded and the product catalogue updated on the device.

### 5.3 System Limitations

In order to operate XEPS, a merchant requires a card processing facility with a financial service provider that supports the ClearCommerce payment gateway. XEPS sends authentication information to the payment processing server when initiating a transaction, in order to identify the merchant account. This is comprised of a username, password and account number. In the current prototype software, there is no way of protecting this information, and can be read from the source Python files with any text-viewer. Clearly this represents an unacceptable security risk, as an attacker with these authentication details would be easily able to create fraudulent transactions. However, valid credit card details would still be required so the impact of the attack is diminished. A production version of the software would most likely be in a compiled language so the attack would not be as trivial. However, this still represents an unacceptable security risk, so some other measures have been considered to alleviate the problem.

A solution would be TrustZone technology championed by ARM, in order to provide a separate, secure domain at a chip level. TrustZone is implemented in within the processor core, with system security elements designed into core hardware. A trusted application would be able to secure and encrypt the necessary login details to prevent the attack described above.

## 6. Conclusions

It is clear from the sources presented herein that card payments are going to remain the de facto standard for personal electronic payments in some form or another for the near future. Many entrants more suited to use on either the web or mobile devices have tried and failed to gain enough market share to present a viable payment solution; the mobile industry must look to familiar payment methods to encourage user adoption of new payment schemes. Although only a proof of concept, the XEPS payment system described here

allows developers to utilise the consumer awareness enjoyed by card networks to easily deploy retail applications in a mobile environment, and provides a way of purchasing higher value goods and services in a way familiar to most end-users.

## 10. References

- [1] Mintel, *Mobile Downloads*, August 2005.
- [2] Electronic Payments International, "Mobile Payments: Is there anybody out there?", vol. 167, May 2001.
- [3] IT Week, "Motorola turns mobile phone into wallet", Oct. 2004, <http://www.itweek.co.uk/2126002>
- [4] C. Swedberg, "Developing RFID-Enabled Phones", *RFID Journal*, July 2004, <http://www.rfidjournal.com/article/articleview/1020/1/1/>
- [5] K. Zafar, "The WAP Gap: Wireless Security and the Wireless Application Protocol", viewed Oct. 2005, <http://www.barnesandnoble.com/offers/wapgap.asp>
- [6] The Shosteck Group, "Premium SMS becomes a significant contributor to AMPU", <http://www.shosteck.com/news/jul03.htm>, July 2003
- [7] H. Godschalk, "Failure of Beenz and Flooz Indicates the End of Digital Web-Currencies?", *Electronic Payment Systems Observatory Newsletter*, Nov. 2001.
- [8] P. Garner, R. Edwards, "Person to Person Mobile Commerce", presented at Mobile Multimedia Communications Conference, Munich, Germany, Oct. 6-8th, 2003.
- [9] J. Quittner, "A Battle's Breaking Out Over Phishing Prevention", *Bank Technology News*, July 2004, available <http://tinyurl.com/4zqwu>
- [10] BusinessWeekOnline "Paypal Spreads Its Wings", May 2005, available <http://tinyurl.com/bruk9>
- [11] CardWatch, "Card Fraud Overview", viewed Oct. 2005 available <http://tinyurl.com/bg8ka>
- [12] APACS, "The Way We Pay" Sept. 2005, [www.apacs.org.uk](http://www.apacs.org.uk)
- [13] US Census Bureau, "Quarterly Retail E-Commerce Sales 2nd Quarter 2005, available <http://www.census.gov/mrts/www/data/html/05Q2.html> viewed Nov. 2005.
- [14] Nokia, Python for Series 60, <http://www.forum.nokia.com/main/0,,034-821,00.html>
- [15] Pyexpat for Series 60, "Helsinki Institute for Information Research", <http://pdis.hiit.fi/pdis/download/pyexpat/>
- [16] Opera Software, "Operators making money on Opera Mini: Two million users surf 4 million Web pages every day" <http://www.opera.com/pressreleases/en/2006/04/06/>, April 2006.
- [17] IPSOS Insight, "The Face of the Web", April 2006.