Multimeetmobile project

TIETOTEKNIIKAN
TUTKIMUSINSTITUUTTI

Mobile e-commerce transaction model

18.12.2000

Vagan Terziyan,
Jari Veijalainen
Henry Tirri

Abstract

The document is a deliverable of the Multimeetmobile project performed by the University of Jyväskylä and financed by the Finnish National Technology Agency (TEKES), Hewlett-Packard Finland, Nokia, and Yomi Solutions.

The main goals are to present the requirements for the mobile e-commerce transactions, some business cases, and a tentative implementation plan for the model.

The document is composed as follows. We first discuss mobile electronic commerce features in general and location based information services in particular. Then in third chapter we consider the basic structure of wireless payment service. In chapter 4 we present basic security requirements for mobile e-commerce transactions. Some business cases in m-commerce and the implied requirements are considered in chapter 5. In 6th chapter we introduce the abstract transaction models and their properties for e-commerce. The 7th chapter is devoted to a system's architecture supporting the model(s) and contains also an implementation sketch for a pilot system in a concrete terminal and server environment.

1 Introduction

The recent cellular phones have a lot of extra capabilities compared to the older ones. Consequently the telephone companies can offer a lot of so called value-added services. Percentage of users for such services is expected to increase due to the fact that with new technologies new (and better) services can be offered. In the future several new services are expected. The ability to pay by phone is one of such services, which is interesting to various companies are interested. For example in Finland one is already able to pay for certain products by using their cellular phone. Sending and receiving images and video are also gaining interest. Since the wireless network will be more data oriented, Internet will become a serious possibility as well. Mobile devices, which provide service not only for ordinary users but also for mobile applications software developers, are of great interest for the telephone companies. Nokia has recently announced the first Symbian-based Communicator - the Nokia 9210 [1]. The Nokia 9210 is the world's first next generation mobile phone to be based on the Symbian open software platform that has been designed to enable third party developers to write powerful additional applications, services and content for users. Users of the Nokia 9210 will be able to benefit from a wide range of additional applications and services currently under development by third-party software developers, which can register free to download the Java and C++ software development kits. The platform closely integrates mobile phone technology with mobile computing capabilities.

Because it is small, secure, personal, familiar and carried at all times, the mobile phone is rapidly evolving into much more than a wireless telephone. It is transforming into Personal Trusted Device with the ability to handle a wide variety of new services and applications, such as banking, payments, ticketing, and secure access operations [2].

Advances in wireless network technology and the continuously increasing number of users of hand held terminals make the latter a possible channel for offering personalised services to mobile users and give space to the rapid development of Mobile Electronic Commerce (m-commerce). M-commerce operates partially in a different environment than Internet E-Commerce due to the special characteristics and constraints of mobile terminals, wireless networks and the context, situations and circumstances that people use their hand-held terminals.

Ericsson, Motorola and Nokia, as the key facilitators of the mobile Internet and the mobile information society, have together established the MeT initiative [2], the first major co-operative effort to provide common open-core technology for the m-commerce market. MeT is targeted to define a common framework for m-commerce applications by extending existing industry standards and technologies and to ensure a consistent user experience across multiple phones, access technologies and usage scenarios. The MeT initiative was founded to establish a framework for secure mobile transactions, ensuring a consistent user experience independent of device, service and network. Creating a standard for secure mobile transactions also opens up the possibility for small transactions to be handled via mobile devices, for example ticketing applications. Mobile devices will also be used to handle short distance (local) payments and transactions enabled by the Bluetooth technology, for example to vending machines and parking meters. The initiative includes methods for service providers to expose their brand in mobile devices.

IBM, Hewlett-Packard, Bowstreet, Oracle, and Sun Microsystems recently unveiled plans to develop a new standard to manage Internet-based transactions [3]. The initiative, called Transaction Authority Markup Language (TAML) will ensure that if XML-based data is not received by an application, the transaction will be abandoned without any changes to the participating systems. TAML resembles online transaction processing (OLTP), a specification that guarantees that if a business-to-business transaction fails, it is reversed so that corporate databases are not altered. The five companies are working together to develop the specification, but will implement the technology separately. The group plans to have XAML available to the public in six months, at which time the specification would be given to an independent standards body.

2 Location based services

The evaluation of mobility lies in satisfying the needs of immediacy and location [9]. The ability to make an immediate decision at the user's location sets high requirements for connectivity. There were three questions to answer before entering the mobile data world:

- Where shall I be active (my own competence and potential, regulatory environment, social environment, competition)?
- Which services are attractive to my customers (market segments, horizontal vs. vertical services)?
- Which technology and products do I need (network infrastructure, platforms, applications, terminals)?

Users of mobile services already accept charging per usage, and they would accept advertising if that would make the service cheaper. However, there are not yet many such users who approve paying for products and services via their phones. There has to be a great effort to convince the public of the security of m-commerce via wireless channels. Integrated service bundles create more value for the user. If a user wants to go to the theatre, he should be able to use a location-dependent service to find a nearby show, book a reservation, get navigational directions to an event, receive relevant traffic information and use micro-payments to pay for all the transactions related to the night out at the theatre.

Location is an element that can be used to filter and differentiate services. Location services are well suited to the mobile environment and can be used when on the move or to track things. They also are useful for locating specific places, such as a hospital or tourist attraction. Conversely, location services help other people locate a user in the event of an emergency, such as when the user has a flat tire. Location capability also allows a user to personalise information based on vicinity.

Motivation for location-based services [12]:

- Convenience - Navigation, concierge services, real-time traffic info, tourist info.
- Safety and security - emergency, roadside assistance, asset tracking.
- Mobile information access – Data filtering, location-based promotions.

Combining positional mechanisms with information about location of various objects one can develop very powerful and flexible personal information services [10]. Suppose there is some geographical area that contains a certain number of objects (points of interests [11]). Each point of interest is assumed to have its virtual representation or, rather, a source of relevant to it information. A user of this information is expected to be mobile. The aim of the location-aware service is to provide a user with information about the objects taking into account spatial relationships between him and the objects. One of the main input parameters is user's location. It is obvious that system should have information about all objects with their spatial location and links to their information sources. If system has this information, it is able to find the close-by objects. Note that for mobile objects system has to periodically update location information via location service, or request it directly from the objects. The actions in the first alternative could be done automatically if we have an access to a location service. In latter case, the user can input his location by himself as a street address or the name of a region. After that, the service is able to provide a geographical description of user's surroundings. These data act an auxiliary role of a navigator or a guide in order to connect real objects with their virtual representations.

3 Wireless payment services

Banks start with huge advantages in m-commerce services such as ticketing, auctions, and value-added information: they control the existing payment infrastructure and understand the security issues [4]. Merita Nordbanken, having established Internet payment system, Solo, for WAP, is a pioneer in this area. Deutsche Bank is working with Nokia and Visa to develop “dual-slot” devices for m-commerce payments. One slot accepts the usual SIM (subscriber identity module) card, the other a “smart card” (such as Visa Cash). Customers of France Telecom can now pay for their purchases by inserting charge cards into the second slot of their handsets. In effect the mobile phone is becoming also a payment terminal.

The Pepsi vending machine piloted by Finnish mobile-security specialist Sonera SmartTrust is a striking example of another use for mobile phones. The company’s service allows the user to buy a drink from a vending machine by dialling its number on a mobile phone. Bluetooth, the emerging standard for short-range radio transmissions, will make it even easier for consumers to make small payments through mobile devices.

A user could, for example, pay for groceries by debiting the sum owed from a card or a bank account or by adding the sum to the phone bill. Support for such services could not only provide a new revenue stream for banks but also reduce the amount of cash they handle and the accompanying costs.

The WAP Payment Service (for example Bipit [5]) enables merchants to accept payments through WAP. By using such service merchants avoid problems in taking care of their own WAP payment servers. Customers of that service are actually merchants, which sell products or services online using WAP, and therefore need a flexible and extensive payment solution. The different WAP payment solutions, international and local payment methods, multiple credit cards can be supported by such service. When new payment method appears it can be integrated into the payment service. Service takes care also of security issues by using use state of the art firewalls, encrypted databases and all the logical protection to ensure a high safety level. Service takes care also of administration and reconciliation of all transactions. Immediately after each payment, whether it has been successful or not, the merchant is notified automatically by the service. This notification can be sent using various methods and platforms. At any given moment, merchant has an access to his shop's payment history through a secure web interface. The payment process works as follows (see Figure 1): When a consumer has selected the desired products in the merchant's WAP store and is ready to pay, merchant redirects him to the WAP payment page of the service. Here, the consumer can choose a suitable payment method and make the payment. The payment is processed directly by service.

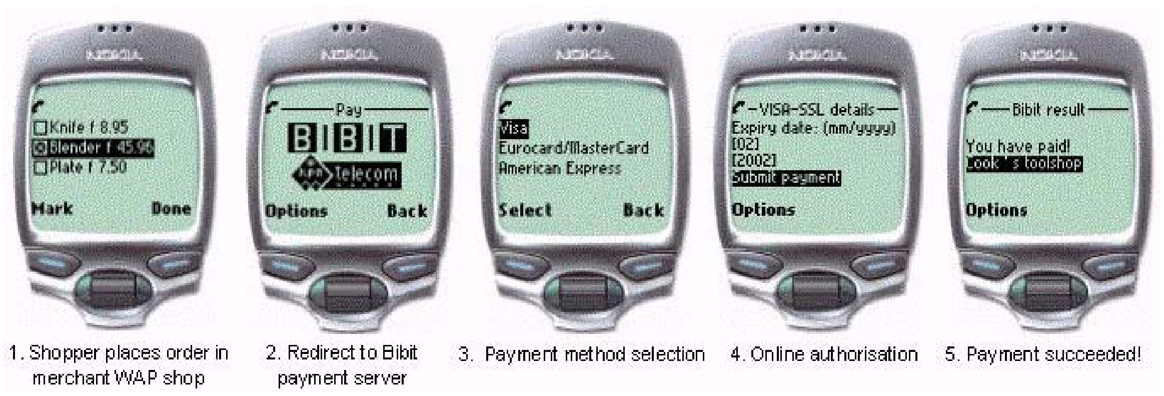


Figure 1: The payment process provided by Bipit [5].

Also UC.COM, the business-focused Internet company, announced in June 2000 [14] that it is launching a newly developed WAP payment service through one of its companies, Securetrading Ltd, the UK's leading Payment Service Provider. This means that e-merchants can now accept credit/debit card payments with real-time authorisation from WAP phones or other WAP devices, on their WAP-enabled Internet sites using payment service. Securetrading will be launching also its new Universal XML Payment Client, which provides XML access to its full range of e-payment services. One of the reasons for it is to allow seamless information exchange between different and potentially incompatible applications or operating systems. This means that the new XML API Payment Client will replace the various existing scripts that were needed to support the wide range of platforms already in use. It can be configured to work with different shopping cart system and the single protocol used in the integration can be used for all payment types supported (debit card, credit card, cash, smartcard, invoice, virtual cash, bank transfer, etc.), and major currencies and acquiring banks across the world [15].

Atomic Software Inc. announces its ecomPort Wireless Payment Gateway. This secure web service allows transaction data to be reviewed from anywhere the merchant has web access. The Merchant Console uses a simple pull down menu to provide real-time reporting for current transactions as well as a full year of transaction history. Transaction information may also be downloaded in a comma-delimited file for integration into a merchant's business management accounting/spreadsheet application. It also features a web terminal feature that allows the merchant to enter credit card transactions in a secure manner through any web browser. These features add up to provide the merchant with exceptional control of his transaction processing in a simple, easy to use package [16].

4 Transaction Security

In the wireless world, security concerns are even greater than on the wired Internet. Wireless Internet applications can succeed only if all the players are confident that transactions cannot be fraudulently generated or altered that transactions are legally binding, and that the confidentiality of private information is adequately protected. Such confidence depends upon the deployment of public-key infrastructure (PKI) technology [6].

"M-commerce clearly requires - as a minimum - encrypted data links and authentication (i.e. the client can be sure that they are communicating with the trusted business that they intended, rather than a charlatan). Secure m-commerce is viable now. However, mobile web merchants must be ready to address consumers concerns on security. It is possible today, that a proportion of your potential consumers will be unable to make secure WAP connections to secure servers, either because they have an older generation phone that does not support it, or because they are currently configured to connect to a WAP gateway that does not support encryption" [13].

Wireless Transport Layer Security (WTLS) is intended for use with the WAP transport protocols and has been optimised for use over narrow-band communication channels. WTLS may be used for secure communication between terminals, e.g., for authentication of electronic business card exchange. Applications are able to selectively enable or disable WTLS features depending on their security requirements and the characteristics of the underlying network. WTLS provides the following features [7]:

- Data integrity – WTLS contains facilities to ensure that data sent between the terminal and an application server is unchanged and uncorrupted.
- Privacy – WTLS contains facilities to ensures that data transmitted between the terminal and an application server is private and cannot be understood by any intermediate parties that may have intercepted the data stream.
- Authentication – WTLS contains facilities to establish the authenticity of the terminal and application server.
- Denial-of-service protection – WTLS contains facilities for detecting and rejecting data that is replayed or not successfully verified. WTLS makes many typical denial-of-service attacks harder to accomplish and protects the upper protocol layers.

WTLS was created in order to facilitate secure wireless transactions without the need for extensive processing power and large memories. WTLS promotes the fast processing of security algorithms. This is accomplished by reducing protocol overhead and enabling increased data compression in comparison to that of traditional SSL solutions. The resulting effect is that WTLS can conduct appropriate security within a wireless network. These advances allow portable, wireless devices to communicate securely over the Internet.

The following problems may be encountered [13]:

- *Not all WAP gateways currently deployed support WTLS security.* In the race to be first to market, the earliest WAP gateways put into service did not support encryption.
- *The mobile phone must be able to support encryption.* The majority does, although a few, notably Ericsson and Mitsubishi, have released phones that do not support WTLS encryption. However, again, this situation will likely be resolved as newer generation phones are released to the market. For example Nokia 7110 supports WAP gateway authentication i.e. the ability to request a WAP gateways certificate, authenticate it and query the consumer as to whether it should be accepted. All earlier phones that support WTLS use it only for encryption, not for the additional security safeguard of authentication.
- *The WAP Gateway must be operated by a trusted body in a highly secured operating environment.* Unfortunately, the current WAP security set-up does not guarantee end-to-end security. This is because of the need for the WAP gateway to perform protocol conversions between the WTP/WTLS used on the wireless side, and the http/https used on the wired side. When encrypted data is received by the gateway from either side, it must be briefly decrypted to clear text, and then immediately re-encrypted in the other side's protocol for onward transmission.

One of the major discrepancies between wired and wireless PKI is the certificate used to authenticate both parties in a transaction. The Internet standard is the X.509 certificate. It can be difficult to determine where to store the certificate, because it must be stored in protected memory, such as on a SIM card. Given the fact that a certificate can be as large as 10 Kbytes, securely storing certificates can be a real engineering challenge. WAP addresses this problem by supporting both X.509 and WTLS certificates. WTLS certificates reduce the number of bits, and the WAP gateway reconstructs a X.509 certificate from the WTLS certificate. WTLS certificates can be as secure as X.509 certificates. The question is whether or not you trust the WAP gateway [8].

Industry is progressing in the area of wireless security, infrastructures, and applications simply by the onslaught of new standards, products, and concepts that are being introduced. One of these new innovations is the Wireless Public Key Infrastructure (WPKI), which will continue to aid in the progression of wireless efforts. WPKI is expected to become an m-commerce security solution that encrypt and decrypt the wireless-transmitted data with the means of public key and private key. It supports functions of electronic signature and certification as well as information security.

Authentication is very important feature of security, which provides a confidence that both parties in a conversation are who they say they are. There are several types of authentication: anonymous authentication, server authentication, and server/client authentication. In some cases the value of the information in the data transferred is so insignificant that one may safely assume that no hacker has any interest in compromising it, and the data can run naked. The device receiving the data assumes that because the data is available to anyone, there is little chance that a hacker will try to corrupt it. This type of data currently represents most data passed during general Internet surfing. Anonymous authentication provides a minimum level of data protection for the lowest system cost. The importance or value of data depends upon the cost of having the wrong data. Server authentication fills the role of verifying that the source of data is reliable. Before you send your credit card number over the Internet, you want to make sure that the server you are sending it to is the one you think it is. The receiving server, however, does not authenticate you and instead trusts that you are who you say you are because you possess "secret" data (that is, the credit card number). This class of authentication services is typical to current online transactions. The highest class of security, and the one that requires the most processing resources, is server/client authentication. When you buy $10,000 worth of stock, it is important for the brokerage to be able to confirm that it was indeed you who made the transaction in case you later try to deny having done so when the stock value unexpectedly drops. This concept is called nonrepudiation. If the brokerage has not authenticated you, the company has no way to prove that it was you who made the purchase and not a hacker pretending to be you [8].

In the above we have concentrated in the security issues related to authentication assuming that if proper authentication can be ensured by the transaction architecture no frauds can occur. This, however, is not the case even in the traditional or Internet-based transaction systems. There is always a possibility that the proper authentication code (password, key, SIM-card etc.) will be illegally acquired by an outside party, and then used in a fraudulent manner. Thus a trusted m-commerce transaction management will by necessity be accompanied by fraud detection systems similar to those currently used for example by credit card and telephone companies. Although this typically is an application specific issue, it should be observed that in m-commerce environment also the location information could be used more effectively for fraud-detection. For example the purchase location patterns become more important features in the detection, and due to more detailed information also provide a faster detection cycle. On the other hand the detection becomes more complex as one cannot anymore assume that the user is accessing the purchase process typically from any particular (fixed) location.

5 Some business cases

Mobile electronic commerce (m-commerce) is e-commerce where transactions are conducted with portable physically small terminals. Currently, the most typical such terminals are WAP-terminals and i-mode terminals, as well as PDAs with wireless communication capabilities, like Nokia Communicator 9110 and 9210, Ericsson MC218 as well as Palm, to mention a few. With the rapid development the set of terminals will become larger.

M-commerce can be divided into two broad categories, mobile Internet e-commerce and location-based services (LBS). The former consists of e-commerce with sites in Internet, which could be performed by stationary terminals like PCs or set top boxes, but where mobile terminal is used to conduct the transactions instead. LBS consists of transactions that are launched from a mobile terminal and which rely on the location of the terminal. Here we present some examples on the business cases from both groups.

5.1 Mobile Internet e-commerce with pre-existing customer relationship

We first assume that the customer (C) has accessed the e-commerce site using PC or other similar means and that he/she has let record the customer profile at the site. Thus he/she has address, credit card number, name, user id, password and so on installed already. We further assume for simplicity (this is the current practical situation, but will most probably change over time) that the customer cannot negotiate with the merchant, but rather, the merchant offers a catalogue with fixed prices and the customer can choose among them. The customer only interacts with the merchant server (M).

The interactions in this case are:

- C: opens a connection to the merchant's site
- M: requests customer's id and password
- C: identifies himself using user's id and password
- C: chooses a product from the catalogue
- C: orders the product using the product number
- M: pre-charges the customer's credit card
- M: confirms the order to the customer's device
- C: stores the order at the device
- M: delivers notification to the customer's e-mail address when the goods leave
- M: delivers the goods to the customer's address

Possibly:
- C: return the goods
- M: receive the returned goods and return the money paid

5.2 Mobile Internet E-commerce without pre-existing customer relationship

As above, but when it comes to identification, the user must ask for an id and a password and provide the information for the user profile.

- C: opens a connection to the merchant's site
- M: requests customer's id and password (fail)
- M: requests customer to register
- C: chooses a "register" button
- C: submits profile (id, e-mail, credit card information)
- M: sends the customer's password for the merchant's site by e-mail
- C: opens a connection to the merchant's site
- M: requests customer's id and password
- C: identifies himself using user's id and password
- C: chooses a product from the catalogue
- C: orders the product using the product number
- M: pre-charges the customer's credit card
- M: confirms the order to the customer's device
- C: stores the order at the device
- M: delivers notification to the customer's e-mail address when the goods leave
- M: delivers the goods to the customer's address

Possibly:
- C: return the goods
- M: receive the returned goods and return the money paid

5.3 Mobile Internet E-commerce with pre-existing customer relationship (secure)

As the above insecure case, but while placing the order it is encrypted with the private key of the customer and the confirmation is encrypted with the private key of the merchant. Also the mobile device (D) of the user in the critical points of the transactional protocol asks a user to enter Personal Identification Number (PIN) to be protected from cases when the mobile device is being stolen during the transactional process.

- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- C: opens a connection to the merchant’s site
- M: requests customer’s id and password
- C: enters id and password
- D: encrypts id and password with private key and submits to merchant’s site
- C: chooses a product from the catalogue
- C: makes an order for the product using the product number
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- D: encrypts an order with private key and submits to merchant’s site
- M: pre-charges the customer’s credit card
- M: encrypts the confirmation of the order with private key
- M: sends the confirmation of the order to the customer’s device
- D: encrypts and stores the order at the device
- M: encrypts and delivers notification to the customer’s e-mail address when the goods leave
- M: delivers the goods to the customer’s address

Possibly:
- C: return the goods
- M: receive the returned goods and return the money paid

5.4 Mobile Internet E-commerce without pre-existing customer relationship (secure)

As the above insecure case, but when the user profile is asked, it will be provided using the private key of the customer and extra PIN checkpoints.

- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- C: opens a connection to the merchant’s site
- M: requests customer’s id and password (fail)
- M: requests customer to register
- C: chooses a "register" button
- C: enters profile (id, e-mail, credit card information)
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- D: encrypts profile with private key and submits it to merchant site
- M: encrypts and sends the customer’s password for the merchant’s site by e-mail
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- C: opens a connection to the merchant’s site
- M: requests customer’s id and password
- C: enters id and password
- D: encrypts id and password with private key and submits to merchant’s site
- C: chooses a product from the catalogue
- C: makes an order for the product using the product number
- D: asks user to enter PIN

- C: identifies himself for the mobile device using PIN
- D: encrypts an order with private key and submits to merchant’s site
- M: pre-charges the customer’s credit card
- M: encrypts the confirmation of the order with private key
- M: sends the confirmation of the order to the customer’s device
- D: encrypts and stores the order at the device
- M: encrypts and delivers notification to the customer’s e-mail address when the goods leave
- M: delivers the goods to the customer’s address

Possibly:
- C: return the goods
- M: receive the returned goods and return the money paid

5.5 Explicit payment with pre-existing customer relationship

Similar to 5.1, but here a customer himself pays for the goods via payment service (P).

- C: opens a connection to the merchant’s site
- M: requests customer’s id and password
- C: identifies himself using user’s id and password
- C: chooses a product from the catalogue
- C: orders the product using the product number
- M: sends paying request to the customer and redirect customer to payment service
- C: opens a connection to the payment service P
- P: requests customer’s id and password
- C: identifies himself using user’s id and password
- C: chooses and submits a payment method from the catalogue
- P: requests customer’s data according to payment method
- C: enters requested data and pays
- P: authorises payment
- P: confirms payment to the merchant
- M: confirms the order to the customer’s device and reports "success" to payment service
- C: stores the order at the device
- M: delivers notification to the customer’s e-mail address when the goods leave
- M: delivers the goods to the customer’s address

Possibly:
- C: return the goods
- M: receive the returned goods and return the money paid

The protocol below is for the Solo payment, i.e. after placing the order the customer goes to the bank (B) and pays there by money transfer, then returns to the merchant with an electronic slip. The interactions look like this:

- C: opens a connection to the merchant’s site
- C: identifies him/herself using user id and password
- C: chooses a product or service from the catalogue
- C: orders the product using the product number
- M: asks the customer to choose method of payment
- C: chooses direct funds transfer through the bank

- M: delivers the electronic bill with the order details and bank account detail for the customer
- C: takes contact with the bank using the URL in the electronic bill above and a (fresh) PIN
- B: presents the transfer details and asks the customer to confirm the transfer with the PIN or TAN
- C: accepts the transfer by putting in a PIN or TAN and sending the page to
- B: returns the electronic receipt to the customer showing which bill was paid
- C: returns electronic receipt to the merchant by using the URL in the receipt
- M: confirms/denies the order to the customer device
- C: stores the order safely at the device
- M: delivers notification to the customer's e-mail address when the goods leave
- M: delivers the goods to the customer's address

Possibly:
- C: returns the goods
- M: receives the returned goods and returns the money paid

5.6 Explicit payment without pre-existing customer relationship

Similar to 5.2, but here a customer himself pays for the goods via payment service.

- C: opens a connection to the merchant's site
- M: requests customer's id and password (fail)
- M: requests customer to register
- C: chooses a "register" button
- C: submits profile (id, e-mail)
- M: sends the customer's password for the merchant's site by e-mail
- C: opens a connection to the merchant's site
- M: requests customer's id and password
- C: identifies himself using user's id and password
- C: chooses a product from the catalogue
- C: orders the product using the product number
- M: sends paying request to the customer and redirect customer to payment service
- C: opens a connection to the payment service
- P: requests customer's id and password (fail)
- P: requests customer to register
- C: chooses a "register" button
- C: submits profile (id, e-mail, credit card(s) information)
- P: sends the customer's password for the payment service site by e-mail
- C: opens a connection to the payment service
- P: requests customer's id and password
- C: identifies himself using user's id and password
- C: chooses and submits a payment method from the catalogue
- P: requests customer's data according to payment method
- C: enters requested data and pays
- P: authorises payment
- P: confirms payment to the merchant
- M: confirms the order to the customer's device and reports "success" to payment service
- C: stores the order at the device
- M: delivers notification to the customer's e-mail address when the goods leave

- M: delivers the goods to the customer’s address

Possibly:

- C: return the goods
- M: receive the returned goods and return the money paid

5.7 Explicit payment with pre-existing customer relationship (secure)

Similar to 5.3, but here a customer himself pays for the goods via payment service.

- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- C: opens a connection to the merchant’s site
- M: requests customer’s id and password
- C: enters id and password
- D: encrypts id and password with private key and submits to merchant’s site
- C: identifies himself using user’s id and password
- C: chooses a product from the catalogue
- C: makes an order for the product using the product number
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- D: encrypts an order with private key and submits to merchant’s site
- M: sends paying request to the customer and redirect customer to payment service
- C: opens a connection to the payment service
- P: requests customer’s id and password
- C: enters id and password
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- D: encrypts id and password with private key and submits to payment service site
- C: chooses and submits a payment method from the catalogue
- P: requests customer’s data according to payment method
- C: enters requested data and payment confirmation
- D: encrypts data and payment confirmation and submits to paying service site
- P: authorises payment
- P: encrypts payment confirmation with private key and submit to the merchant
- M: encrypts and submit confirmation of the order to the customer’s device and encrypts and reports "success" to payment service
- D: encrypts and stores the order at the device
- M: encrypts and delivers notification to the customer’s e-mail address when the goods leave
- M: delivers the goods to the customer’s address

Possibly:

- C: return the goods
- M: receive the returned goods and return the money paid

5.8 Explicit payment without pre-existing customer relationship (secure)

Similar to 5.4, but here a customer himself pays for the goods via payment service.

- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- C: opens a connection to the merchant’s site

- M: requests customer's id and password (fail)
- M: requests customer to register
- C: chooses a "register" button
- C: enters profile (id, e-mail)
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- D: encrypts profile with private key and submits it to merchant site
- M: encrypts and sends the customer's password for the merchant's site by e-mail
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- C: opens a connection to the merchant's site
- M: requests customer's id and password
- C: enters id and password
- D: encrypts id and password with private key and submits to merchant's site
- C: identifies himself using user's id and password
- C: chooses a product from the catalogue
- C: makes an order for the product using the product number
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- D: encrypts an order with private key and submits to merchant's site
- M: sends paying request to the customer and redirect customer to payment service
- C: opens a connection to the payment service
- P: requests customer's id and password (fail)
- P: requests customer to register
- C: chooses a "register" button
- C: enters profile (id, e-mail, credit card(s) information)
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- D: encrypts profile with private key and submits it to payment service site
- P: sends the customer's password for the payment service site by e-mail
- C: opens a connection to the payment service
- P: requests customer's id and password
- C: enters id and password
- D: asks user to enter PIN
- C: identifies himself for the mobile device using PIN
- D: encrypts id and password with private key and submits to payment service site
- C: chooses and submits a payment method from the catalogue
- P: requests customer's data according to payment method
- C: enters requested data and payment confirmation
- D: encrypts data and payment confirmation and submits to paying service site
- P: authorises payment
- P: encrypts payment confirmation with private key and submit to the merchant
- M: encrypts and submit confirmation of the order to the customer's device and encrypts and reports "success" to payment service
- D: encrypts and stores the order at the device
- M: encrypts and delivers notification to the customer's e-mail address when the goods leave
- M: delivers the goods to the customer's address

Possibly:

- C: return the goods
- M: receive the returned goods and return the money paid

5.9 Information Society Services

These services are either digital goods delivered to the customer terminals or digital services obtainable through the network. The interactions are similar as above from the beginning but the end is different. If it is the question of a service (access to a site) that is prepaid, then the interactions end at the confirmation of the order. If, however, the goods are digital and they are delivered to the mobile terminal - for instance a piece of music - then it makes sense to continue the protocol by confirming the delivery. We also assume that digital goods are not returned physically to the merchant. Assuming this, the interactions become as follows:

<The beginning as in some of the above scheme; depends on the payment methods and Customer-merchant relationship>

- M: confirms/denies the order to the customer device
- C: stores the order safely at the device
- M: delivers the goods to the customers device
- C: issues a confirmation to the merchant that the goods have arrived

Basically, the digital goods can be delivered to another device, too. A typical case could be video-on-demand service where the order is placed using the mobile terminal but the delivery address is the stationary video device accessible through the set top box at home. The difference to the above case is that the user must manually (or communicating with the set top box) check that the goods have arrived and then send the confirmation either from the mobile terminal or using the set top box interface.

The above cases are all such where the mobile terminal is used in e-commerce transactions instead of a stationary terminal (PC). The business models are essentially the same as in case of stationary terminals, but the access network is different and the terminals have rather different capabilities as compared to stationary terminals. The problems are primarily two-fold ones. First, all the problems that are encountered in "normal" Internet e-commerce, and additionally those coming from the wireless access networks and terminals with small resources. The questions raising in the latter context are at least:

- Is it really possible to use the same material (e.g. HTML pages) for stationary PC's with big memory, fast processor and big colour terminals? From display point of view no, but maybe there is a clever piece of software translating dynamically the complicated pages into simpler ones (cf. WAP gateway) and vice versa?
- Is the metaphor of taking contact with the e-commerce server eligible if we take into account the properties of the wireless access network (WAP, i-Mode, UMTS, etc)?

5.10 Location-based mobile e-commerce

The mobility of the terminals opens new possibilities in e-commerce, because it is possible to use the location of the terminal as information used by the services. From business model point of view there is also a difference in the sense that the customer often already has a *billing relationship* with the merchant. Thus, the transactions do not necessarily involve any immediate payment mediator, but the service is delivered and the merchant just writes down a billing entry. Later, e.g. once a month, the bill is sent to the customer who then pays it using e.g. banking infrastructure. We assume for simplicity that the services are provided by a Mobile Network Operator (MNO) who bills the customer in the framework of a roaming contract. Thus, the device identity is used in the transaction. We assume that the MNO has fetched already the user profile from the home location register when the transaction starts.

The interactions look like following in an insecure case.

C: sends a location request "X"+ the device identity DI to the merchant (MNO)
M: checks the device identity DI; if X is not allowed for DI in this network or for some other reason (e.g. stolen device, service not available), sends denial of service; otherwise performs the operations needed to service "X"
M: sends the response to the customer device (e.g. map or the bare co-ordinates)
C: stores the response and issues confirmation for the service
M: stores the confirmation and add the price of the service to the billing database

In a secure case the customer encrypts the request "X" with the private key. The merchant opens the request by the public key. This is obtained from the HLR by using the device identity parameter that is not concealed and asking the keys. The response is encrypted with the public key of the customer. Thus, only the customer can decrypt the response and thus find out the information about his or somebody else's location. The confirmation of the service is also encrypted by the private key of the customer. Thus, it functions as a non-repudiation of the used service.

6 Transaction modelling for mobile e-commerce

6.1 Some history

Transaction management was first developed in the database management context. In the modern sense the requirements were set up in the context of relational database management systems. The idea was that the interface of the database management system only offers a relational language to manipulate the data. The main targets of transaction management, concurrency control of the simultaneously running applications and recovery in the case of crashes, are taken care of automatically by the database management system.

In this context a modelling incentive was needed. The idea was to model program executions within the DBMS from the concurrency and recovery perspective. The executions should exhibit "ACID" properties. Atomicity means that either all the retrievals, updates and deletions incorporated into a transaction program (i.e. transactional application) are performed against the database state or all of its effects are nullified (aborted) in a way or the other. Consistency means that the transaction program is semantically correct, i.e. it keeps the database in a consistent state if run alone. Notice that this property cannot be primarily guaranteed by the DBMS but it is assumed to hold by the DBMS designer. Isolation means that the concurrent executions of different transaction programs should behave towards the user and database as if there were no concurrency in running them. Durability means that the results of successfully terminated i.e. "committed" transaction program executions must persist in the state of database irrespective of the other concurrently running transactions or system crashes that might destroy the state of the system produced by the transaction. In other words, only another successful run and committed transaction can change the state of database, i.e. nothing else can do it.

ACID properties are rather abstract and one can imagine many ways of implementing them. It is also easy to see that they are not orthogonal. Too much concurrency in executions makes it impossible to guarantee any other property. This necessitates a more elaborate modelling where the properties can be presented more formally. This boiled down to what could be called "RW model", where a transaction execution is modelled as a sequence of reading and writing pages on the disk. The concurrency control problems are modelled forming an interleaving of the R and W operations, called history (or schedule). Among those histories some are correct, some are clearly incorrect resulting in reading inconsistent database states, lost updates or producing incorrect database states due to the above reasons. The basic correctness criterion (equivalence of an arbitrary history with a serial history) is introduced in 1975 by the group at IBM developing one of the first relational systems. This property is commonly called serializability of the history. From the practical point of view conflict serializable histories is the most important class of histories, because they can be recognised by time-efficient (i.e., polynomial) algorithms. Using locking mechanism and e.g., two-phase locking policy (2PL) or rigorous locking policy (RL) one can guarantee serializability.

Later, several equivalence relations are investigated, among them state and view and also conflict serializability and their relationships with each other shown. Towards the end of the seventies, the model is enhanced to a distributed database case. It turns out that the most practical method is to combine the RL at each site and a global atomic commitment. This is especially powerful in that respect that the global level software does not need to be aware at all of the local implementation of the concurrency control. Thus, there is no need for special interfaces, the SQL interface is enough. The local systems can have different locking granularities and they do not even need to use locking. It is enough that the local RW histories are guaranteed to be rigorous. To guarantee the atomicity of commits, one needs a new distributed protocol called two-phase commit protocol (2PC). The component systems need to support this, i.e. prepare command in addition to begin-transaction Commit/Abort.

The results of the earlier studies based on the RW model are exhaustively represented in [23] and [24].

During the eighties the transaction modelling ideas were put into new contexts. One context was CAD/CAM [25] the other was international banking [17, 18]. Further, also hierarchical modelling incentives for execution modelling were suggested [26, 21]. New serializability classes for RW model, like quasi-serializability [22], and two-level serializability [25] were presented. A rather comprehensive source for the developments during the eighties is [22]. The heterogeneous transaction management developments can be found in [20]. A more extensive history of transaction modelling can be found in [19].

From e-commerce point of view the modelling incentive of [27, 28] is the most important. From the application perspective he introduces the concepts money atomicity, goods atomicity and certified delivery. The concepts are however, not formalised in a rigorous way. What m-commerce transactions must provide are basically the same properties.

While looking at the atomicities above, one should keep in mind that in order to achieve them, the customer needs to pay the credit card bill to the credit card company or phone bill to an MNO. Further, the credit card company must transfer appropriate funds to the merchant. These steps are abstracted from below, because they belong to the common steps provided by the current infrastructure and the problems attached to them are not m-commerce specific.

6.2 Transaction modelling in a more general setting

The original database-driven transaction modelling incentive assumed that there is one application program run and a transaction management system guaranteeing the ACID properties (both in a centralised and a distributed case). This view allows the model to abstract from all properties of the program, except consistency (i.e., correct semantics) and the behavioural requirement that the program must issue Commit or Abort at the end towards the system interface. This abstraction is not anymore relevant for cases where the application consists of a bunch of application programs communicating over a computer network and some accessing database systems through a database interface.

The consistency above becomes now a much more complicated issue.
1) It is question of an interdependent set of application programs and their overall effect on the databases involved
2) It is question of the protocol(s) they obey (or do not obey) while communicating with each other

Durability is not anymore and issue of only the databases but also program executions that must be able to continue from the point on they crashed (forward recovery for programs).

Atomicity becomes a novel kind of problem because for guaranteeing as much autonomy as possible for the sites, local subtransactions are committed prior to the global commit or abort. Thus, in case of a global abort, something called compensating transactions must be run at those sites where local subtransactions are run and committed. If they were not yet committed they can be aborted.

In order to be able to address the above concerns, the scope of the transaction modelling framework must be enhanced for the traditional one. What we need is
- formal model able to express transaction specifications consisting of a set of application programs (e.g. definition graph)
- correctness properties for individual program specifications
- a formal model for the executions of the above set of programs
- correctness properties for the executions
- an abstract algorithm that transforms a formal specification to a formal execution preserving the properties
- real representation of the application programs
- real distributed implementation for the protocols and system software needed to implement the above abstract transformation algorithm and that guarantees the formal properties for the executions

For further discussion see [19, Ch. 3].

6.3 Properties of the mobile e-commerce transaction model(s)

6.3.1 Relationship with environments analysed earlier

The mobile e-commerce environment is rather similar to the international banking environment, for which the S-transaction model was developed (See [17]). In fact, the banking environment is a (hidden) part of the mobile e-commerce environment in some cases.

Shortly, the common features are:

- both are distributed, global environments with autonomous players;
- in both cases economic values are transferred;
- in both cases the main goal of the transactions is to guarantee atomicity of the executions; in banking it is the money atomicity, in mobile e-commerce the certified delivery. This leads to compensation problematic in both cases and similar considerations concerning the programs;
- in both cases the durability and serializability are of similar importance and are solved at the same level in the system architecture (at the database level syntactically);
- the same modelling ideas can be used in both environments.

There are some differences:

- in the mobile e-commerce environment also complicated real actions such as physical goods delivery are included; these are missing from the international banking environment;
- the mobile e-commerce environment is hostile in the sense that the customers or merchants might be traitorous which is not the case within the banking environment;
- the terminals of the international banking environment are very well secured against unauthorised use whereas in the mobile e-commerce environment a terminal can easily be stolen and taken in an unauthorised use;
- the terminals in the mobile e-commerce have much less processor, memory etc. resources than the terminals in international banking;
- the security mechanisms are different ;
- in international banking there is one big trusted "middle-player" (SWIFT; www.swift.com) offering messaging services and other services whereas in the e-commerce this is not (yet) the case;
- the transaction scopes are different (reflected in the TIDs; there is not a global TID in the mobile e-commerce case, whereas in international banking transaction there is a global TID);
- in S-transaction model it is reasonable to assume that all the global programs belonging to the S-transaction environment run the same request-response-ack/nack protocol, in the mobile e-commerce environment this is not clear. The reason is that there is an existing infrastructure and this is not homogeneous, but rather different from country to country. Neither can it be made homogeneous by adding the mobile e-commerce channel;
- "location invalidation", i.e., transaction becoming invalid due to out-dated location information is specific to wireless, nomadic environment and thus has not been considered in banking contexts.

6.3.2 The scope of the mobile e-commerce transactions

The main purpose of the transactional mechanisms is to guarantee the money atomicity, goods atomicity and at the end the certified delivery. When looking into the business cases, one sees immediately that there is not one a single point of control that could be allocated the responsibility to guarantee this in the same manner as in a conventional environment the 2PC client runs the 2PC protocol and guarantees that all subtransactions started at the diverse sites either commit or abort. Rather, each autonomous player has a partial responsibility here. So they start different transactions that are, however, interdependent, and moreover, partially subordinate. For instance, the contact with the credit card Company in business case 1 only follows after the customer has placed an order. Second, the delivery is only started if the merchant has become assured that he/she will get the funds for the goods ordered. Thus, the delivery is conditioned on the outcome of the payment.

What is common to all these cases is the fact that the customer is the ultimate origin of the interactions. Thus, we can in a natural way assume that the customer is the entity being the root in an execution hierarchy. The cases then differ in number of other players the customer must take direct contact with - and thus run a concrete protocol with the player. Case by case it seem as follows

Cases 5.1-5.4: Customer only interacts with the merchant. There are, however, two types of protocols run at the customer terminal:
- the actual ordering of the goods (embedded in request-response type of protocol, place order - confirm/deny order)
- identification of the customer to the merchant (embedded into several request-response pairs); optional

Cases 5.5.-5.8 Customer interacts both with the merchant and with a bank/payment service. There are four different kinds of protocols run at the customer terminal:
- the actual ordering of the goods (imbedded in request-response type of protocol, place order - confirm/deny order)
- identification of the customer to the merchant (embedded into several request-response pairs); optional
- payment protocol against the payment service or against the bank (embedded into several request-response pairs)
- identification of the customer to the payment service (embedded into request-response pair)

Case 5.9. Customer interacts only with merchant (MNO) requesting a service/contents (embedded into request-response pair)

In Figure 2 a simplified view on the overall transaction structure in the cases where payment is implicit is given. The circle within which the components are described models the atomicity sphere of the transaction. The payment and delivery are subtransactions of controlled by the Merchant. Customer is at the root of the transaction tree. The picture abstracts from the identification step, because it can be excluded from the scope of the mobile e-commerce transaction (i.e., it is step similar to scanning of the catalogues that do not need to be included within the transactional boundaries). The figure also abstracts from the possible cancellation of the order. It could be modelled as a subtransaction at the merchant that waits for the possible return of the goods and starts the reverse payment.

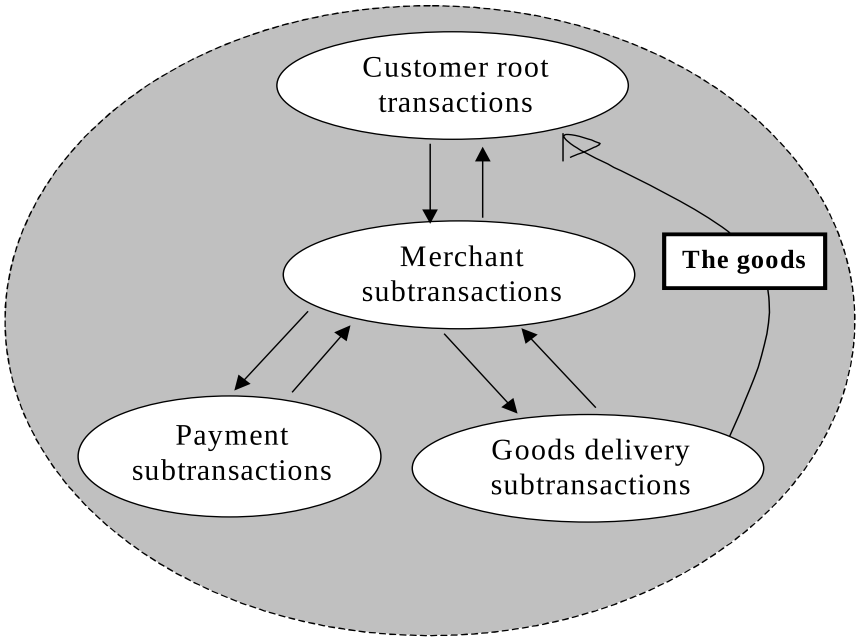


Figure 2. Hierarchical view on the m-commerce transactions in cases 5.1-5.4

Figure 3 shows the overall abstract structure of the cases is shown where the customer is directly responsible for the payments. Again, the cancellation is not shown.

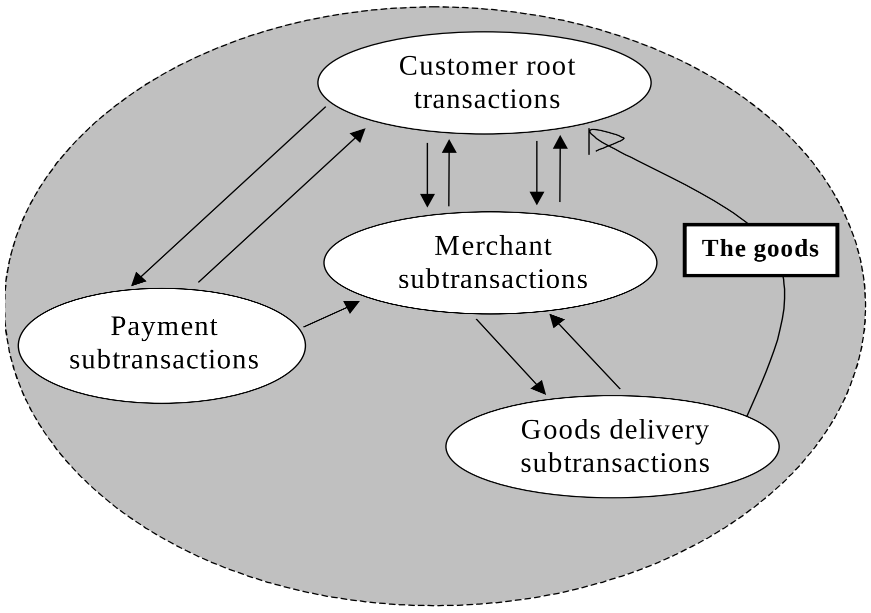


Figure 3. Hierarchical view on the m-commerce transactions in cases 5.5-5.8

What is noteworthy here is that from structural point the cases are not different, whether or not the PKI is used. Using the keys only changes the processing of the messages and the interaction between the customer and the device. This is not of interest from protocol point of view.

How many protocols a customer terminal should be able to run? The answer is difficult to give. It depends on the definition of the protocol. Basically, if the terminal is able to run something like HTTP or the corresponding WAP protocol, then it is able to run any protocol embeddable into HTTP request-response interaction pattern. Thus, the concrete form of requests and differences in how many requests and responses there are do not matter. From atomicity point of view there are still differences, because the state of protocol at the terminal must perhaps be stored between each request and response.

What is the scope of the mobile e-commerce transactions? **Our answer is that part of the global transaction tree originating at the customer terminal**. Thus, those parts of the overall tree that do not directly involve the terminal are outside the scope of mobile e-commerce transactions. Second, only **atomicity-sensitive** actions are within the scope of the mobile e-commerce transaction. For instance, scanning of catalogues is not within the scope. Neither are identification steps (unless the acceptance of the order at the merchant depends on the identification). From this it actually follows that the transaction borders should be explicitly marked at the terminal. Only so the terminal knows when to start to log the transactional operations.

6.3.3 Properties of the mobile e-commerce transactions

As it was discussed earlier, the main property of the transactional mechanisms is to guarantee certified delivery. As we saw above this can only be guaranteed by the transactions in the overall sphere. The mobile e-commerce transactions are the roots only. So what they should guarantee is atomicity in one sense: That each initiated operation (place order/request payment) is run into an either accepted or denied end state.

Second, they should provide the end-to-end security that guarantees non-repudiation, authentication and authorisation, as well as the principle of one user/transaction. Third, they must be implemented in such a way that C-autonomy of the terminals can be maintained. In practice this means that the transaction management software at the terminal must store the state of the transaction in a non-ambiguous way, so that after a break in communication with the other players or after a crash, the transaction can be continued into a proper end.

7 Implementation issues

7.1 Problems with heterogeneity

Looking into the cases in Fig. 2 and 3 above, one sees several problems for the detailed design of the transaction model. There are at least two structurally different protocols visible at the customer terminal (the implicit and explicit payment). This means that if one wants to support these scenarios as such, one needs two different methods, or at least versions of the transaction manager functionality. In addition, the cancellation of the order that is possible when ordering physical goods, is not paid attention to. This adds another, conditional (sub)transaction step, into the picture. Furthermore a practical complication is that in the scenarios some of the protocol exchanges happen through email, some through more formalised protocol exchanges that can be tracked through the same transactional software component.

The structural problem is, seen from a different angle, that of including certain actions dynamically into the transactional scope or excluding some of them dynamically and possibly conditionally. This is solved at the database clients offering a dynamic GUI for users by letting the user to set the transaction borders. If we accept the same paradigm in this environment, then the mobile user **must** indeed set the transactional borders. This approach has far-reaching ramifications, as the entity responsible for the transactions would be the customer. Can we really put the burden on the customer?

Another way of doing this is an implicit transaction border setting by the software at the terminal. The latter requires that the software can recognise when the user has initiated such an action that should start a transaction, and correspondingly, which action or incoming PDU should close the transaction. This might be doable, but there are subtle problems. First, although all the merchants might follow the request-response protocol while letting the customer to place the order, the forms for doing it would most probably differ in details. The same would hold for the payments. Thus, without further unification of the forms themselves or additional unified meta-data (saying "I am an order form, start transaction, if the user really wants to place the order") in them, it is hard to know when to start the transaction. For the ending action we have the same problem.

Incorporating email messages into automatic protocol exchanges is extremely difficult, even if they were received at the mobile terminal with e-mail software. This is because the latter is not at all integrated with the transactional software and because the current email confirmations are meant for humans not for the machines.

Looking at this problem, the necessary condition is first that if the emails are sent as part of the protocol exchanges, they must conform to strict protocol formats understandable by the transaction management software running at the terminal. Further, an E-mail client should be able to sort out such email messages that are conceptually protocol messages of the mobile e-commerce transactions but come through a different channel. After sorting them out, the software should give the messages to the transaction manager. If e-mail messages are used, then they should have a strict format (conforming in fact, to a PDU) that makes recognition of them automatically possible. And of course, such email messages should be addressable directly to the mobile terminal so that no email client e.g. at the PC would be able to receive them.

7.2 Monolithic vs. embedded solutions

All in all, we have basically two broad possibilities: either the user controls the transaction borders or the transaction manager software does it. We can also combine the solutions into one; if the transaction manager does not know where the borders are, it asks for users assistance. This issue is closely related with whether we have only one (or only few) standardised mobile e-commerce protocol implementations or whether there are embedded or add-on solutions to the current situation. The most straightforward solution to overcome heterogeneity above, and solve also the transaction border problem, would be to define a complete mobile e-commerce protocol that should be then obeyed by all parties. This would make it easy for the terminal to recognise the transaction borders, because one could assume that upon starting any e-commerce transaction this protocol would be always be used. The terminal would then run up the software responsible for the protocol and this would do the logging and also automatic recovery if needed (see below). Thus the terminal would know what to do at any point of time.

Another implementation solution would be to enrich the currently used e-commerce protocols with a suitable "preamble" that actually contains the same PDU information as above. The preamble "start transaction here" could be either in XML, WML or HTML (or whatever is used) format included into the forms used by the merchants or banks. Upon receiving a form with such a preamble, the (micro)browser would start the transaction manager functionality. It would stop the latter upon receiving the "end transaction" preamble in the response from the merchant or from another participant server - or when the user would like to end the transaction. One maybe appealing idea in this solution would be to generate the TIDs at the server, not at the client. This would make recovery somewhat easier in the case where the TID would not be known yet at the server during the crash.

7.3 Logging and distributed recovery

This is a necessary component to keep the state persistent. When the transaction manager gets the begin transaction (either from the user or deduces it) it generates a new TID and a time stamp (TS) and writes the Begin-tr (TID, TS) record into the log. When new action is started within the scope of the transaction, the transaction manager writes Action (Params, TID, TS) i.e., the action, TID and timestamp TS into the log. Upon termination, the transaction manager writes the END-TR (TID; TS) into the log. If the encryption is used (secure cases) then the logged entries are also in the encrypted form. For efficiency reasons, TID and TS are kept unencrypted.

In a traditional transaction management one of the main problems is to guarantee that the log records and the actions are in synchrony. That is, if an action occurs in the log then it is clear whether it was executed or not. In this environment, where the C-autonomous behaviour of the terminals must be preserved, it makes sense to log both at the beginning as well as at the end of an action. Only so one would know, whether an action is in progress. Typically, if for example an order is being placed and the connection brakes before the merchant is able to confirm/deny the order, the terminal would not know whether the order was accepted or not. Placing the order again as part of the recovery might cause it to be placed twice at the merchant. On the other hand, if it is not placed in the first place and neither replaced, it will not placed at all.

Thus, we see that there is a need for a *mobile transaction recovery protocol* that is used upon recovery between the terminal and the merchant or between the terminal and the bank/payment service. This protocol is nothing new as such, the 2PC protocol also needs such an additional recovery protocol. We cannot, however, use the 2PC-recovery protocol here, because the problem is not only to run the 2PC either to committed or aborted end, but also to guarantee atomicity of placing an order or that of payment actions.

The log must be kept as persistently as possible in the terminal. The current PDAs already have begun to offer possibility to dump the memory contents from time to time onto either a special memory chip or onto the disk of a PC. This raises the degree of the persistency, but in the current form is quite tedious. In this respect hopefully the new generations of the terminals would offer an automatic way of making the memory contents more persistent.

In a distributed communication world there is always also the possibility to store the log onto a special server in the network. The log can be mirrored or stored solely at the server. In the former approach the time for mirroring the log can be optimised e.g. based on the time of the day and tariffs incurred (e.g., do not mirror while roaming, do it in the night). The latter possibility is attractive, if the terminal has a very tiny memory and the communication with the server and its actions are fast. The direct on-line storage of the log records causes naturally additional delay in the transaction processing and additional problems in guaranteeing the atomicity of log writing. These issues are for further study.

7.4 Transaction monitor scheme

Above, we have assumed that the customer acts directly with the merchant and payment systems and we have also seen that the recovery must be taken care of by the terminal in those cases. This functionality can be partially moved to the network e.g. by assuming that there is a transaction monitor at the MNO that is contracted to run the mobile e-commerce transaction. Thus, when the transaction starts, the terminal sends the request to start the transaction to the transaction monitor. The request contains the order information, possibly encrypted with the private key of the customer. The transaction manager then runs the transaction on behalf of the customer.

If the transaction manager needs some information known only by the customer privately (private key or PIN) then there are two possibilities to get it. Either it asks it from customer terminal explicitly when needed - or the customer provides the information already upon request. The latter way is of course risky, if the customer would externalise the keys or PINs to the transaction monitor and this is not reliable.

The benefit of the scheme is that the terminal does not need to log the transaction beyond the Begin-tr, end-tr and it does not need to keep up connection all the time. The asking information scheme requires, however, that the transaction monitor can set up a connection or otherwise "push" information to the terminal.

Another benefit is that the heterogeneity problems discussed above due to the different merchants etc. is moved to the sphere of transaction monitor. It can then run different protocols with different topologies between itself and the other players. The protocol between the terminal and transaction monitor can be more easily standardised.

This idea needs further studying.

7.5 Vulnerability analysis

Let us make a vulnerability analysis of this business scheme. Let us consider the following possibilities, which can happen at any step of the m-commerce transactional process:

- mobile device of a customer is stolen;
- connection fails;

- customer or merchant is assumed to be traitorous.

- **(1) C: opens a connection to the merchant's site**

if fail try (1) again or later;

- **(2) M: requests customer's id and password**

if connection fail start (1) again or later;

- **(3) C: identifies himself using user's id and password**

if id and password incorrect during 3 attempts then M asks C to register to his site;
if connection fail start (1) again or later;
device of C is stolen by C1 just after C identifies himself - very small probability because C is online waiting for response from M (not consider this case);

- **(4) C: chooses a product from the catalogue**

if connection fail start (1) again or later;
device of C is stolen by C1 just after C chooses product from the catalogue - very small probability because C is whether online going to order product he has chosen or close connection going to make order later (will be asked id and password again when login to make order and will fail) - not consider this case;

- **(5) C: orders the product using the product number**

if connection fail and ordering is not done start (1) again or later;
if connection fail just after ordering has been made go to (6);
if device of C is stolen by C1 just after C has made an order not a big problem, because C1 cannot control the process any more;
*if device of C is stolen by C1 just after C has made an order and forget logout from the M's site (**week point!**) C = C1 - wrong customer and can make new wrong order on behalf of C (case can be protected if device asks C to enter PIN before sending every new order);*

- **(6) M: pre-charges the customer's credit card**

- **(7) M: confirms the order to the customer's device**

if connection fail before (7) M will try (7) again or later;
if connection fail just after (7) M is not guaranteed that confirmation is delivered; M should try (7) again or later;
*if M refuses to do (7) - (**week point!**), C cannot prove that he did not receive confirmation;*
if device of C is stolen by C1 between (6) and (8) - not problem because C1 cannot control the process any more;

- **(8) C: stores the order at the device**

- **(9) M: delivers notification to the customer's e-mail address when the goods leave**

e-mail should be sent with automatic delivery confirmation, so if delivery fail M can try (9) again or later;
*if M refuses to do (9) or C refuses to recognise it - (**week point!**), C cannot prove that he did not receive notification or M cannot prove that he sent confirmation;*

- **(10) M: delivers the goods to the customer's address**

7.6 An implementation sketch for the LBS pilot

7.6.1 General business model for Information Services

The e-commerce trends are such that the basic triangle business model: Customer-Merchant-Bank is being replaced by the model: MultipleCustomer-MultipleMerchant-MultipleBank. Taking into account availability (or intentions to be available) of appropriate services for customers, merchants and banks the model actually at its higher level again can be treated as triangle: CustomerServices-MerchantServices-PaymentServices as it shown in Figure 4.

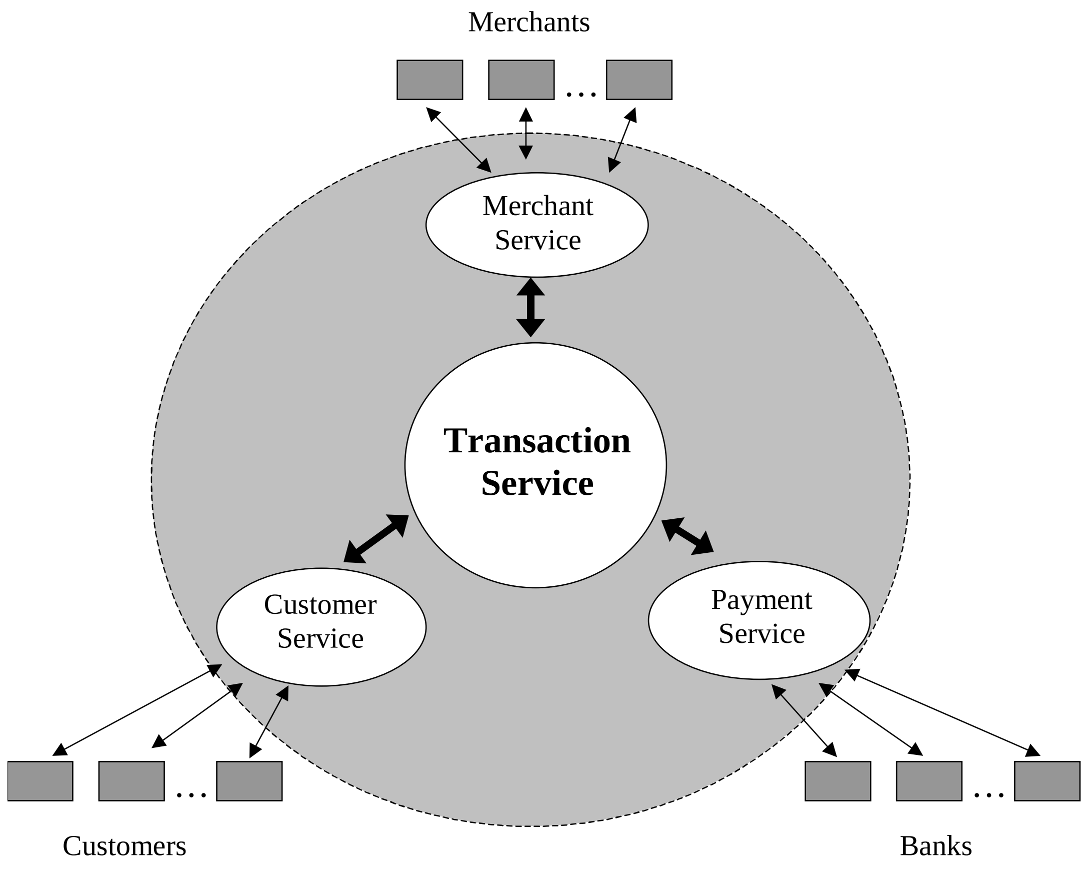


Figure 4: General business model for Information Services.

All three services in Figure 4 have generally very similar infrastructure.

Customer service (CS). Accesses merchant service to provide a customer with catalogues of information products and services based on requests and profiles, stores and uses customers profiles for better service, collects customers requests which are not available in catalogues and satisfies them offline, negotiates on behalf of a customer with merchant service, accesses payment service and provide requested billing information from customers profile, confirms delivery of information (service) to a merchant.

Merchant service. Accesses customer service to provide a merchant with catalogues of unsatisfied information requests, stores and uses merchants profiles for better service, collects merchants offerings in catalogues and introduces them to customer service, negotiates on behalf of a merchant with customer service, accesses payment service and provide requested account information from merchants profile.

Payment service. Stores and uses profiles of banks and different payment methods for better service, collects banks offerings in providing payment service, makes payment based on customers requests and informs merchants about this, collects bills for offline customers and payment requests for offline banks.

Transaction service. Provides management business transactions within the triangle: CustomerService - MerchantService - PaymentService.

The important advantage of information services business models is that no transactions are taking place in physical world (no goods delivery). This means that all necessary confirmations for management transactions can be done in wired/wireless network. Availability of appropriate services for customers, merchants and banks on the one hand makes it easy for them to play their roles in e-commerce or m-commerce of the information market, and on the another hand makes it easy to perform secure transactions between appropriate services.

There are two categories of all "players" (customers, merchants and banks) in Figure 2: based on their access type (wired or wireless). However in the same time every player can participate in business process being online or offline one. The reason of being offline might be whether accidental (connection fails) or technological (based on the role of appropriate player in business process).

Online customer. Himself surfing the catalogues, selects information (service) he wants to buy, pays on-line for it and gets it.

Offline customer. Submits request for certain service and logs off. Customer service search in catalogues based on customer's request and profiles, and selects information (service). When a customer login again he will find selection result, confirms order, and sign payment request. Then a customer can log off again and take ordered information later from customer service.

Online merchant. Himself surfing the catalogues, selects customers requests, sends him appropriate offering, asks to confirm order, checks payment confirmation, delivers information (service).

Offline customer. Submits offerings to be published in catalogue and logs off. Merchant service search in catalogues of customer's requests taking into account merchants profile, and selects appropriate requests. When a merchant login again he will find selection result, asks confirmation of the order, and sends payment request. Then a merchant can log off again and latter provides requested information when gets confirmation from payment service.

Online bank. Himself surfing the catalogues of payment requests, selects one which fits to his profile, request authorisation and makes payment.

Offline bank. Submits profile information to payment service and logs off. Payment service searches in catalogues of payment requests the one, which fits to the bank profile, requests authorisation. When login again bank finds one or several signed payment requests and makes payment.

7.6.2 General business model for Location Based Information Services

LBS business model is one of concrete applications of the model presented in Figure 4. General business model for LBS is in Figure 5.

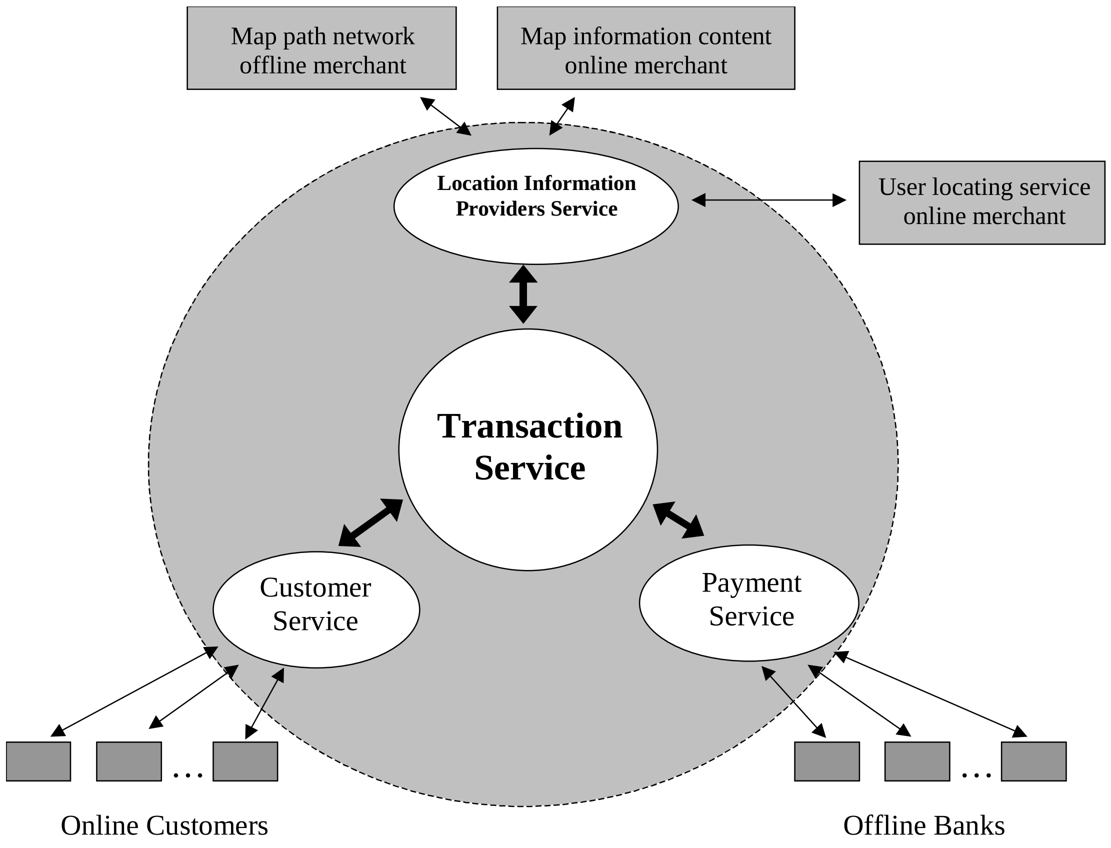

Map path network offline merchant
Map information content online merchant
Location Information Providers Service
User locating service online merchant
Transaction Service
Customer Service
Payment Service
…
…
Online Customers
Offline Banks

Figure 5: General architecture for LBS.

Figure 5 presents LBS case studied in Multimeetmobile Project [11]. Merchants here are:

Map path network offline merchant (MM). He assumed to be a landowner or another authority who has right to sales maps or provide geographical data. It is assumed that data should be prepared in XML form (i.e. GML - Geographic Markup Language) and include topological relationships between geographical objects (path network). His role in business model is offline because he assumed to sale his complete data once to location service provider and does not participate any more in on-line business process.

Map information content online merchant (IM). He sales information about geographical objects from the maps which can be possible points of interest to customers. This data also should be prepared in XML form and include descriptions of hotels, restaurants, museums, shops and other points of interest for customers. His role in business model is online because he should provide on-line and dynamic information about certain point (for example availability of free places in a hotel, or menu of a restaurant).

User locating service online merchant (LM). He tracks location of customer on demand and sales this information. His role in business model is online because he should provide on-line and dynamic information about certain customer.

Location Information Providers Service (LS) here is a service for MM, IM and LM, which can integrate their information and sail it to the customers.

7.6.3 Business submodels for Location Based Information Services

The transactional model for the general scheme from Figure 5 can be clustered to several submodels of a type Customer-Merchant-PaymentService by changing roles of main "players". In Figure 6 one can see the model of transactions between customer and customer service.

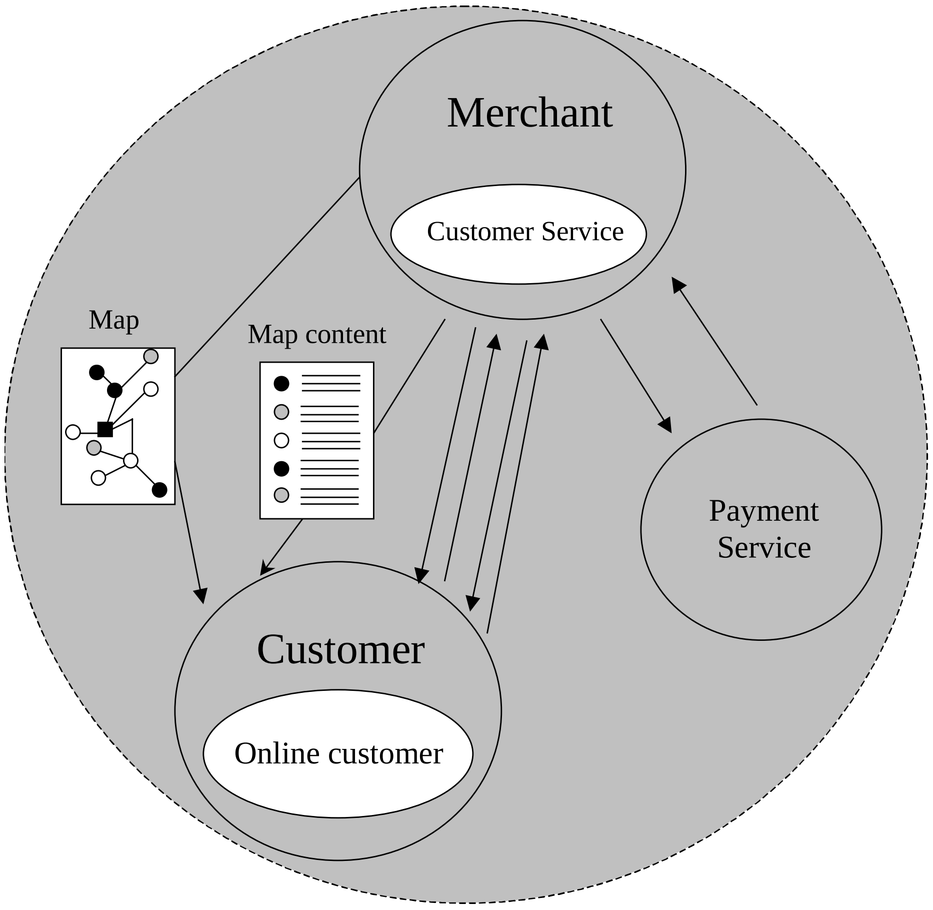


Figure 6. Transactional model Customer - Customer Service

In transactional model in Figure 7 Location Information Providers Service takes the role of a Customer and the role of Merchant is taken by Map Path Network Offline Merchant. In this scheme it is supposed that Location Information Providers Service buys once the Map Database from the Map Path Network Offline Merchant to use it further online itself.

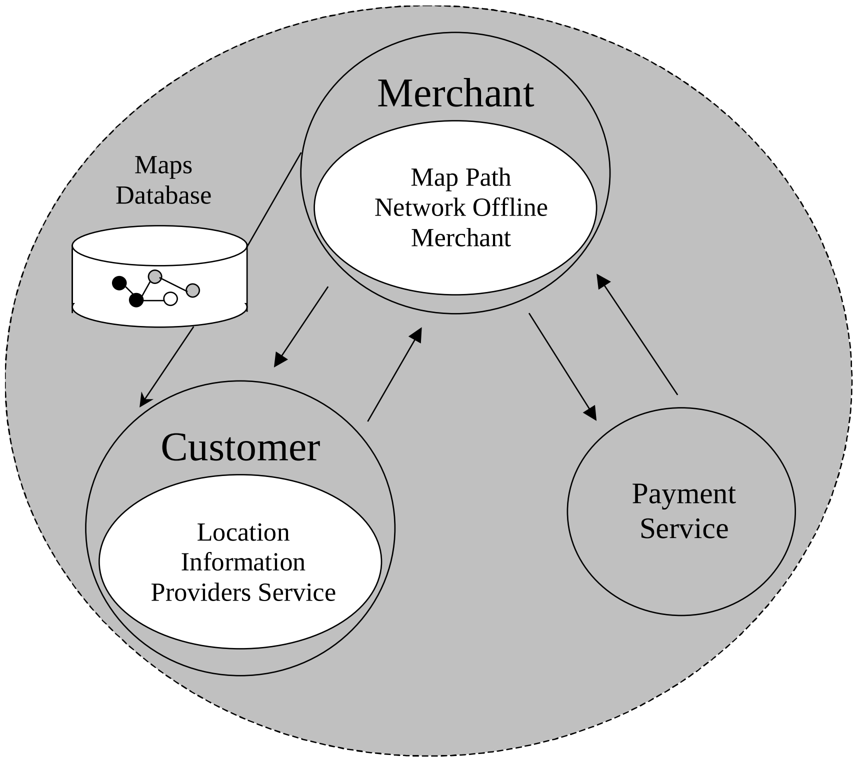

Merchant
Map Path Network Offline Merchant
Maps Database
Customer
Location Information Providers Service
Payment Service

Figure 7. Transactional model LS - MM

In transactional model in Figure 8 we have the same Customer as in previous one but the role of Merchant is taken by Map Information Content Online Merchant. In this scheme it is supposed that Location Information Providers Service buys "fresh" map information content online according to the requests of customers.

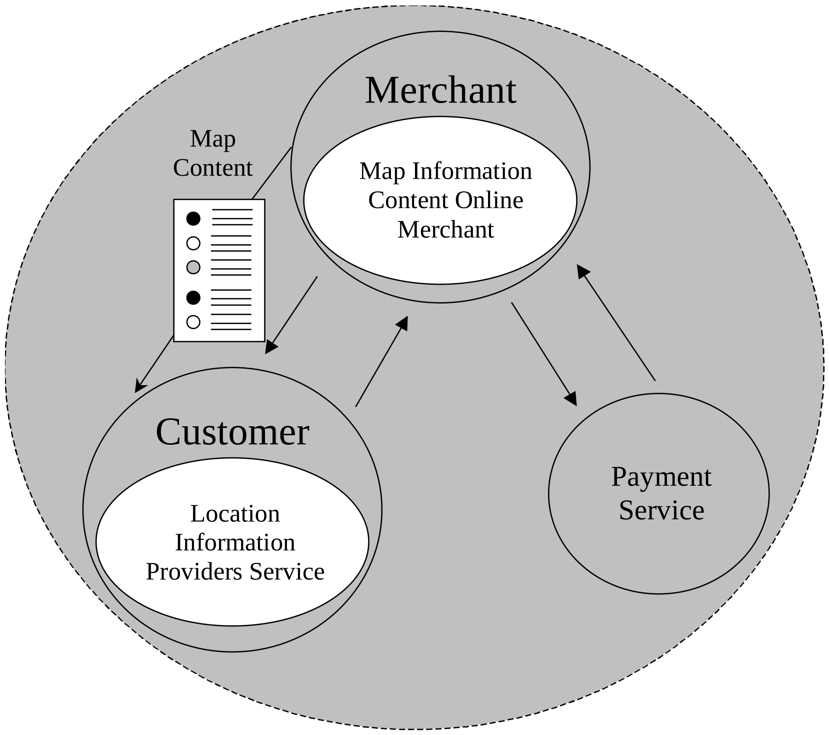

Merchant
Map Information Content Online Merchant
Map Content
Customer
Location Information Providers Service
Payment Service

Figure 8. Transactional model LS - IM

In transactional model in Figure 9 we again have the same Customer as in previous two ones but the role of Merchant is taken by User Locating Service Online Merchant. In this scheme it is supposed that Location Information Providers Service buys service to locate the customer for further use of customers co-ordinates.

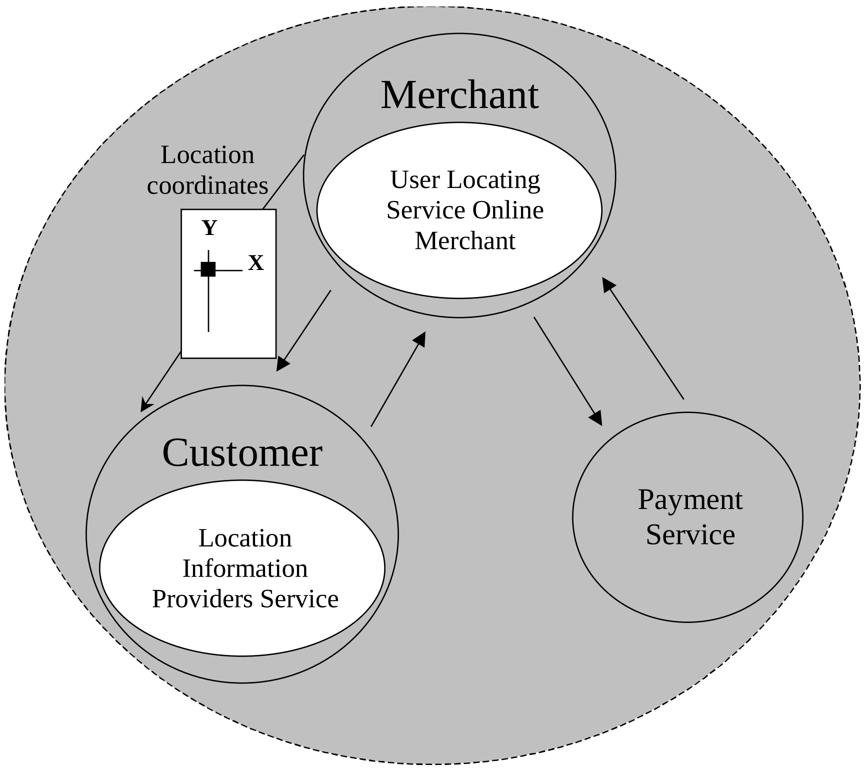

Merchant
User Locating Service Online Merchant
Location coordinates
Y
X
Customer
Location Information Providers Service
Payment Service

Figure 9. Transactional model LS - LM

7.6.4 Transactional protocol for Location Based Information Services

Below there are main parts of the transactional protocol for the LBS based on models from Figures 5-9:

```
{begin transaction
            {begin LS-MM subtransaction
-   LS:     opens a connection to the MM
-   MM:     requests LS's identification
-   LS:     identifies itself
-   LS:     delivers a request for a map database to MM
-   MM:     sends a bill to LS
                  {begin LS-P subtransaction
-   LS:           opens a connection to P
-   P:            requests LS's identification
-   LS:           identifies itself
-   LS:           delivers a payment request to P
-   P:            requests LS's authorisation
-   LS:           LS's authorises itself
-   LS:           LS's selects payment method
-   LS:           Pays the bill
```

- P: Reports "success" to LS

end LS-P subtransaction}

{begin P-MM subtransaction

- P: Delivers payment confirmation to MM
- MM: Confirms delivery of confirmation

end P-MM subtransaction}

- MM: delivers the requested database to LS
- LS: send confirmation of delivery to MM

end LS-MM subtransaction }

end transaction}

{begin transaction

{begin C-CS subtransaction

- **C: opens a connection to the Customer Service**
- **CS: requests customer’s identification**
- **C: identifies himself**
- **C: requests a map related to his location**
- **CS: makes a bill for the location-based map delivery single request plus CS’s single service**

{begin CS-LS subtransaction

- *CS: opens a connection to the LS*
- *LS: requests CS’s identification*
- *CS: authorises itself*
- *CS: delivers map request with customer’s phone number to LS*
- *LS: makes a bill for the LS’s map delivery single service*

{begin LS-LM subtransaction

- LS: opens a connection to the LM
- LM: requests LS’s identification
- LS: identifies itself
- LS: delivers locating request with customer’s phone number to LM
- LM: makes a bill for the LM’s customer locating single service
- LM: defines customer’s location
- LM: delivers customer’s location to LS
- LS: sends confirmation of delivery to LM
- LM: updates LS’s monthly bill by adding a bill for the LM’s single service

end LS-LM subtransaction}

- *LS: picks up necessary data from map database based on customer’s location*
- *LS: delivers map data to the CS*
- *CS: sends confirmation to LS on delivery of map data*
- *LS: updates CS’s monthly bill by adding a bill for the LS’s map delivery single service*

end CS-LS subtransaction}

- **CS: sends map path network data to the customer’s device**
- **C: confirms delivery to CS by pressing OK when client software reports on**

the screen "Map delivered, Press OK button"(otherwise client software will not display the Map on the screen)
- **CS:** **updates the monthly bill of the customer**
- **C:** **selects point of interest on the map**
- **C:** **sends information request to CS concerning point of interest**
- **CS:** **makes a bill for the information delivery single request plus CS's single service**

{begin CS-LS subtransaction
- *CS:* *opens a connection to the LS*
- *LS:* *requests CS's identification*
- *CS:* *authorises itself*
- *CS:* *delivers information request to LS*
- *LS:* *makes a bill for the LS's information delivery single service*

{begin LS-IM subtransaction
- LS: opens a connection to the IM
- IM: requests LS's identification
- LS: identifies itself
- LS: delivers information request to IM
- IM: makes a bill for the IM's information delivery single service
- IM: picks up requested information about point of interest
- IM: delivers information to LS
- LS: sends confirmation of delivery to IM
- IM: updates LS's monthly bill by adding a bill for the IM's information delivery single service

end LS-IM subtransaction}
- *LS:* *delivers information to the CS*
- *CS:* *sends confirmation to LS on delivery of information*
- *LS:* *updates CS's monthly bill by adding a bill for the LS's information delivery single service*

end CS-LS subtransaction}
- **CS:** **sends information about point of interest to the customer's device**
- **C:** **confirms delivery to CS by pressing OK when client software reports on the screen "Content delivered, Press OK button"(otherwise client software will not display information on the screen)**
- **CS:** **updates the monthly bill of the customer**

end C-CS subtransaction}

end transaction}

7.6.5 Screenshots for the main C-CS subtransaction protocol in Location Based Information Services

Below one can see the protocol for the main subtransaction C-CS and some main screenshots.

{begin C-CS subtransaction
- **C:** **opens a connection to the Customer Service** (see Figure 10)

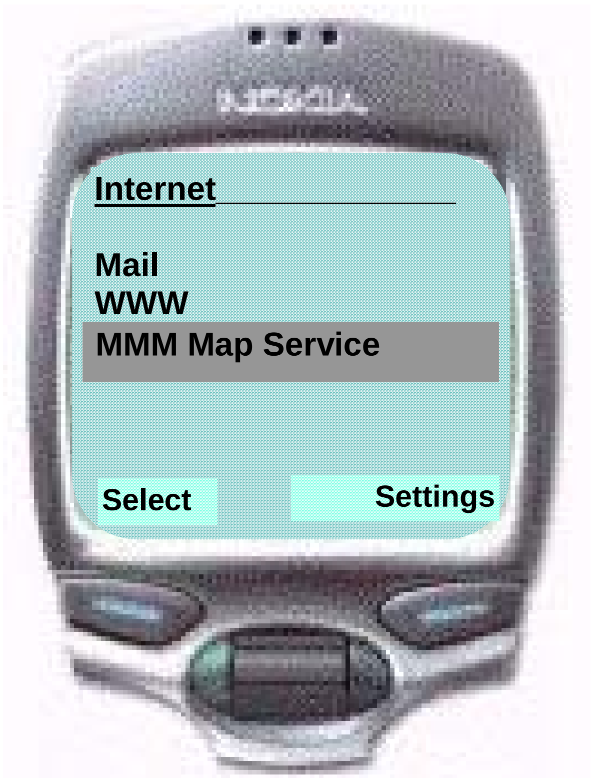


Figure 10. Opening a connection to Location Service

- **CS:** **requests customer's identification**
- **C:** **identifies himself** (see Figure 11 and Appendix 1)

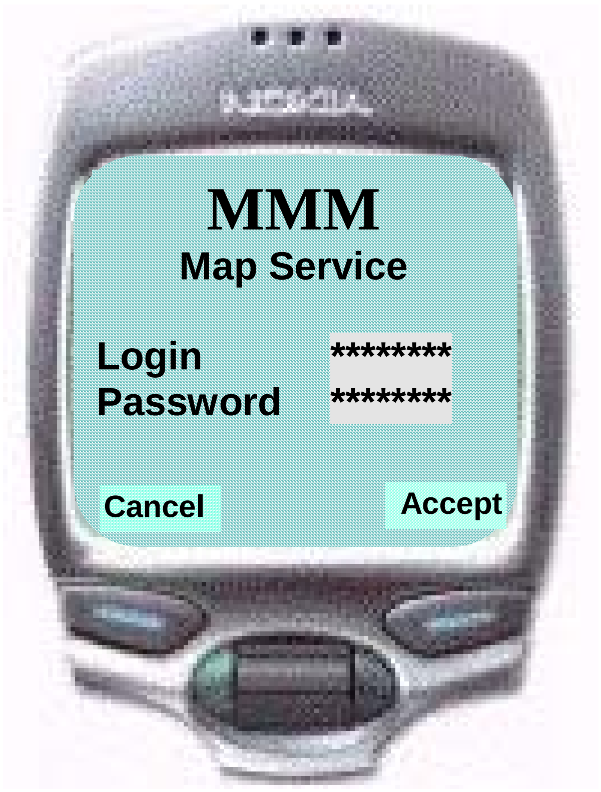


Figure 11. User identifies himself for Location Service

- **C:** **requests a map related to his location** (see Figure 12 and Appendix 2)

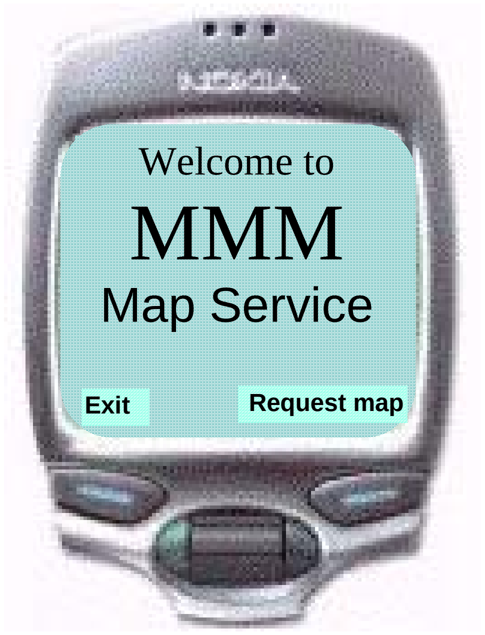


Figure 12. User requesting a map from the Location Service

- **CS:** **makes a bill for the location-based map delivery single request plus CS's single service**
- **CS:** **sends map path network data to the customer's device** (see Appendix 3)

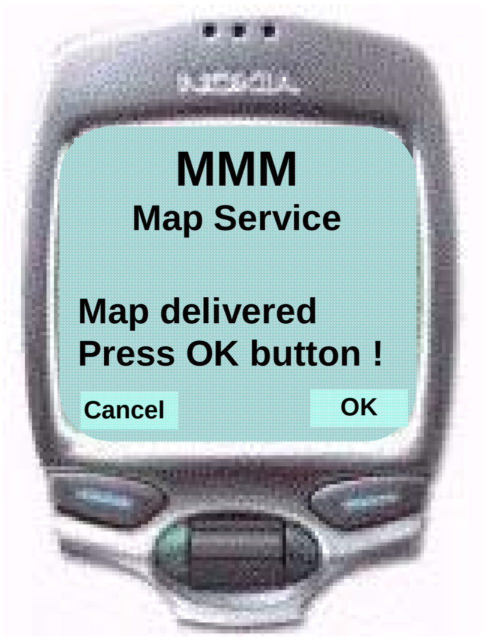


Figure 13. Location Service asks the user to confirm map delivery

- **C:** **confirms delivery to CS by pressing OK** (see Figure 13) **when client software reports on the screen "Map delivered, Press OK button" (otherwise client software will not display the Map on the screen), and gets the map on the screen** (see Figure 14)

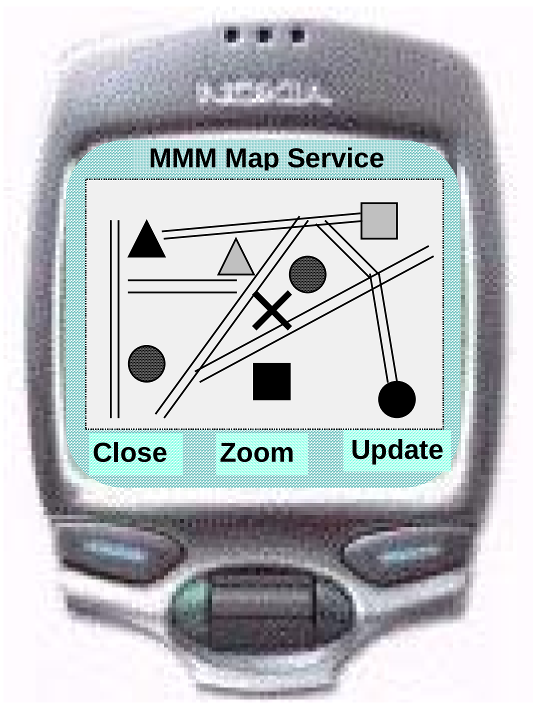


Figure 14. Requested map displayed on the screen

- CS: **updates the monthly bill of the customer** (see Appendix 4)
- C: **selects point of interest on the map** (Figure 15)

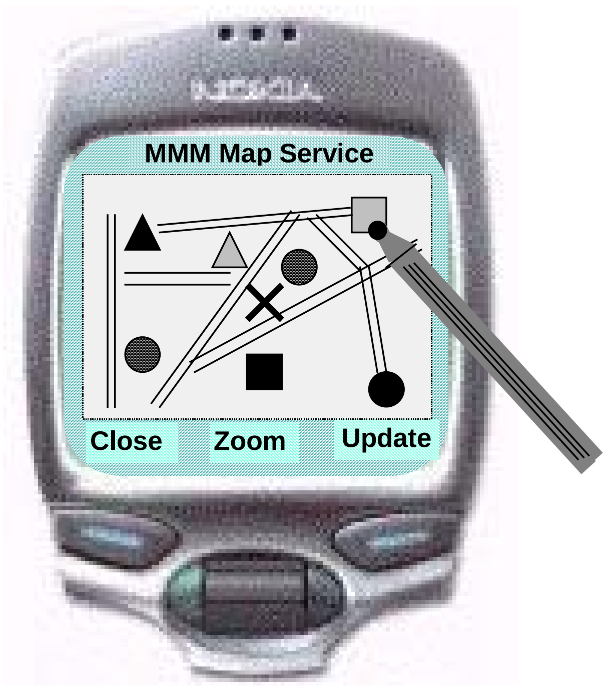


Figure 15.User selects point of interest on the map

- C: **sends information request to CS concerning point of interest**
- CS: **makes a bill for the information delivery single request plus CS's single service** (see Appendix 5)
- CS: **sends information about point of interest to the customer's device** (see Appendix 6)

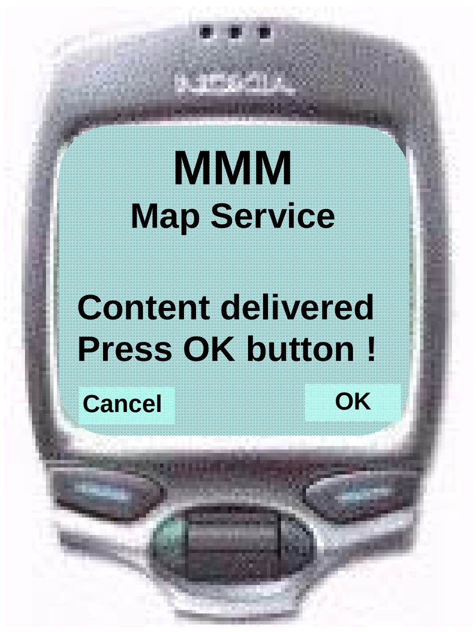
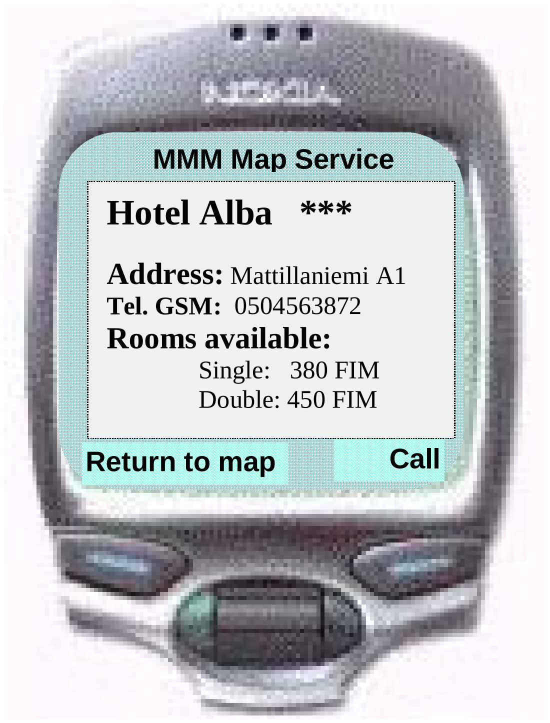


Figure 16. Location Service asks the user to confirm content delivery

- **C:** **confirms delivery to CS by pressing OK** (see Figure 16 and Appendix 7) **when client software reports on the screen "Content delivered, Press OK button" (otherwise client software will not display information on the screen) and gets requested content on the screen** (Figure 17)



Hotel Alba ***

Address: Mattillaniemi A1
Tel. GSM: 0504563872
Rooms available:
Single: 380 FIM
Double: 450 FIM

Figure 17. Requested content displayed on the screen

- **CS:** **updates the monthly bill of the customer**
  end C-CS subtransaction} (see Appendix 8).

References:

1. Nokia Announces the World's First Symbian-Based Communicator, *Mobic News*, 21 November 2000, available in: http://www.mobic.com/ .
2. The MeT Initiative - Enabling Mobile E-Commerce, *Met Overview White Paper*, 2 October, 2000, available in: http://www.mobiletransaction.org/pdf/MeT White Paper.pdf .
3. A. Gonsalves, Big Vendors Launch B-to-B Initiative, *TechWeb News*, 25 October 2000, available in: http://www.techweb.com/wire/story/TWB20001025S0012 .
4. D. Maude, R. Raghunath, A. Sahay, P. Sands, Banking on the device, *The McKinsey Quarterly*, No. 3, 2000, pp. 86—97, available in: http://www.mckinseyquarterly.com/.
5. The Bipit Wap Payment Service, available in: http://www.bibit.nl/payment/WAP/.
6. Secure Wireless E-Commerce with PKI from VeriSign, available in: https://www.verisign.com/server/rsc/wp/wap/index.html.
7. WAP Architecture Specification, The WAP Forum, 2000, available in: http://www1.wapforum.org/tech/terms.asp?doc=WAP-100-WAPArch-19980430-a.pdf.
8. N. Cravotta, Securing the Wireless World Wide Web, August 2000, available in: http://www.ednmag.com/ednmag/reg/2000/08172000/17tt.htm.
9. B. Wassum, Mobile Data Service Models for Mobile Network Operators, *Mobile Internet & Information Services 2000*, March 2000, San Diego, available in: http://www.the-arc-group.com/ebrief/2000/mobileinternetis/executive_summary.htm.
10. G. Swedberg, Ericsson's Mobile Location Solution, *Ericsson Review*, 1999.
11. A. Garmash, A. Katasonov, XML-Based Geographical Information Service for a mobile Environment, MMM Project Report, 16 November 2000, TITU, University of Jyvaskyla.
12. K. Chadha , Location-Based Services: The Next Differentiator, *Mobile Internet & Information Services 2000*, March 2000, San Diego, available in: http://www.the-arc-group.com/ebrief/2000/mobileinternetis/executive_summary.htm
13. A. Wigley, Implementing Secure WAP E-Commerce Applications, June 2000, available in: http://www.securetrading.com/wap/technicalbriefing.htm.
14. UC.COM Subsidiary Securetrading Launch First Fully Available UK Retail WAP Payment Gateway, Press Release, 26 June 2000, Available in: http://www.securetrading.com/pr200600.htm.
15. Securetrading Launch New Universal XML Payment Client at e-Business EXPO 2000, Press Release, 17 November, 2000, Available in: http://www.uc.com/index.html?/firstpage.htm
16. Wireless Payment Gateway Service from Atomic Software, 3 September 2000, available in: http://www.cellular.co.za/news_2000/news09032000_wireless_payment_gateway_service.htm.

17. Jari Veijalainen, Transaction Concepts in Autonomous Database Environments. (Ph.D. thesis). GMD-Bericht Nr. 183, ISBN 3-486-21596-5. R. Oldenbourg Verlag, Munich, Germany, April 1990.
18. Jari Veijalainen, Frank Eliassen, Bernhard Holtkamp: The S-transaction Model. Chapter 12 in: Ahmed Elmagarmid (ed.): Database Transaction Models for Advanced Database Applications. Morgan-Kaufmann Publishers, San Mateo, USA 1992.
19. Rolf de By, Wolfgang Klas, Jari Veijalainen (eds), Transaction Management Support for Cooperative Applications. Kluwer Academic Publishers,December 1997.
20. Jari Veijalainen, Antoni Wolski, Transaction-based Recovery. Chapter 11 in: A.Elmagarmid, M. Rusinkiewicz and A. Sheth (eds.), Management of Heterogeneous and Autonomous Database Systems, Morgan Kaufmann Publishers, San Francisco, CA, USA, October 1998, pp. 301-350.
21. Gerhard Weikum, Principles and Realization Strategies for Multilevel Transaction Management. ACM TODS 16(1), March 1991, pp.132-180.
22. Ahmed Elmagarmid (ed.) Database Transaction Models for Advanced Applications. Morgan Kaufmann Publishers, 1992.
23. Christos Papadimitriou, The Theory of Database Concurrency Control. Computer Science Press, 1986.
24. Phil Bernstein, Vassis Hadzilacos, Nathan Goodman, Concurrency Control and Recovery in Database Systems. Addision-Wesley, 1987.
25. Hank Korth, Won Kim, Francois Bancilhon, On Long-Duration CAD Transactions. Information Sciences 46 (1988), pp. 73-107.
26. J.Elliot.B. Moss, Nested Transactions, An Approach to Reliable Distributed Computing. MIT Press, 1985.
27. J. D. Tygar, “ Atomicity in Electronic Commerce”, PODC 96, pp. 8 – 26.
28. J. D. Tygar, “Atomicity versus Anonymity: distributed Transaction Electronic Commerce”, Proc. of the 24th VLDB Conference, 1998, 1 – 12.