Hermes — A Lean M-Commerce Software Platform Utilizing Electronic Signatures

Sebastian Fischmeister, Günther Hagleitner, Wolfgang Pree
Software Research Lab, University of Constance
Campus PO 188, 78457 Constance, Germany
{*firstname.lastname*}*@uni-konstanz.de*

Abstract

The world-wide growth of the mobile-telephony market opens the door for mobile commerce (m-commerce). For the restricted target platforms used for m-commerce such as mobile phones or personal digital assistants, domain solutions need to be lean but still secure. Especially mobile contracting and mobile payment as parts of m-commerce require solutions resting upon a sound legal basis. Electronic contracts and slips require solutions that are admitted as evidence in court. This is considered a key aspect for m-commerce applications to be widely accepted.

This paper presents the Hermes platform developed at the University of Constance. It is a lean and secure trading platform that utilizes digital signatures for authentication conforming with the European electronic-signature laws.

1. Introduction

Today a number of gadgets exist that help us to make phone calls, to remember meetings, to surf the Internet, or to read electronic books. Mobile phones, electronic books, web pads, personal digital assistants, and hybrid devices are spreading fast additionally new devices will emerge on the market. Compared to workstations, such hand-held devices suffer from hardware restrictions. This gap is intrinsic, because workstations will always use the latest hardware, which is not optimized for mobility [13]. In future, software systems will need to support multiple devices as front ends and servers as back ends. This paper introduces a representative of these systems—a mobile-commerce (m-commerce) payment system.

Mobile telephony is a growing market in Japan, Europe, and the USA. Still, m-commerce is delaying its take off. Among others, security considerations are an obstacle to overcome in order to create a satisfied user group. The European Community introduced the electronic-signature guideline [7] in 1999 and each member state enacted a corresponding law. Austria has an electronic-signature law since 1st of January 2000 [5]. This law provides the legal basis for authentication and contracting via electronic devices. It can be seen as the counterpart to the widespread personal identification number (PIN) and transaction authentication number (TAN) model. The PIN/TAN model does not reflect the process used by bank clerks. In the clerk model the TAN equals the PIN—you just sign the transaction and the bank verifies your signatures. Electronic signatures can also be applied to e-commerce and m-commerce [8]. Thus, it should be an intrinsic part of an m-commerce system.

Section 2 briefly describes three related m-commerce systems. Section 3 examines the functional and non-functional requirements of Hermes. Section 4 describes the resulting system in terms of overview and in-depth descriptions. This section also includes a description of the communication subsystem that is especially tailored to support mobile devices. Section 5 outlines simulation environment and describes specific technical details in a sample run. Section 6 presents an evaluation of the Hermes system and demonstrates that it is open for the integration of new service and emerging standards. Finally, the conclusion (Section 7) sums up the key points of the work.

2. Related work

Payment solutions for m-commerce already exist; Paybox [2], for instance, offers a lean system controlled via voice, whereas Brokat [1] provides a full-fledged system for m-commerce and e-commerce applications.

With the Paybox system, the service provider sends a transaction request to the customer's paybox. In turn, this paybox calls the customer, who has to authorize the transaction by entering her PIN. After that, the Paybox company

This work has been sponsored by mobile-network operator mobilkom Austria.

obtains the money via debit notes and transfers it to the service provider. This system has drawbacks: (1) the service provider must know the number of the client to call her, (2) the customer does not know what she is going to authorize (the paybox message that the customer receives merely states the amount of money to be transferred), and (3) courts do not admit this form of authorization as evidence.

In contrast to Paybox, the X-PAY suite of Brokat provides a complex environment that facilitates various configurations for m-commerce systems (e.g., they support the wireless-application protocol (WAP), short message service (SMS), interactive voice response). The system provides authentication and authorization of customers and merchants. However, this system also has drawbacks: (1) courts will not admit this form of authorization as evidence unless in incorporates electronic signatures, and (2) the m-commerce solution of Brokat involves seven different parties and the functionality is spread across three major products, therefore it is difficult to deploy this system.

The mSign approach documented in [6, 11] (recently joined forces with the Radicchio approach) is similar to the Hermes approach, however, it relies on a different technology. mSign bases on the WAP and WMLScript. The mSign protocol consists of three levels. Level one relies on the GSM cryptographic algorithms to provide security between the signature service provide and the mobile phone. Level two introduces additional cryptographic algorithms to secure the communication between the mobile phone and the signature service provider. Finally, level three uses end-to-end security by using asymmetric algorithms between the mobile phone and the recipient of the signature recipient. Level one and two are not compliant with the electronic-signature guideline of the European community. Furthermore level one has severe security drawbacks as the GSM cryptographic algorithms have been successfully attacked as reported by Golic in [9]. The network operator also can tamper the data as it can access the data after it has been decrypted by the GSM network.

The drawbacks of the Paybox and Brokat's approach were the driving idea to build Hermes. This new system proves, that it is possible to build a lean, easy-to-use, open, and especially secure (in terms of enforcing payments) m-commerce platform (at least level three of the mSign approach).

3. Requirements

The following paragraphs introduce the usage model and non-functional requirements of the Hermes system. They represent agreed-upon goals which stem from discussions with the stakeholders.

3.1. Use case

The use cases are described as step-by-step lists. For larger systems numerous use cases of different abstraction layers exist, however, we concentrate on an overview of the most important one.

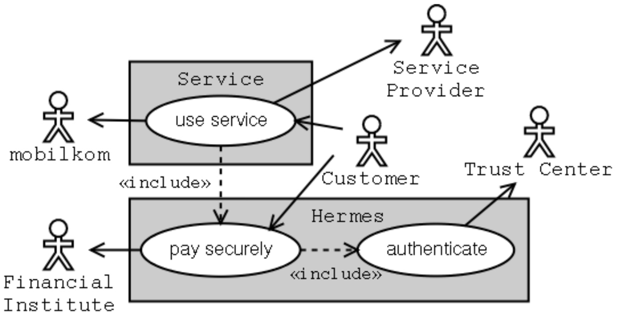


Figure 1. The main use case diagram. The use case use service **is generic—actual services will specialize it and eventually trigger the use case** pay securely **that describes how the system reacts to payment requests by the service provider.**

Figure 1 depicts the most important system function, the parties involved and their interactions. The main use case is called pay securely. The goal in context of this use case is to transfer the desired amount of money from the customer's account to the account of the service provider. The precondition is that the customer has to be registered with the system. The success end condition is reached when the financial institute has transferred the desired amount of money and the service provider has obtained the receipt. The primary actor is the service provider, secondary actors are the customer, the mediator, the financial institute, and the trust center. The pay securely use case requires participation of the secondary actors and the use case authenticate to accomplish its task. It is triggered by the generic use-service use case that stands for arbitrary m-commerce applications. The flow of events in the best case scenario is as follows:

1. **Ask the customer for acknowledgment.** The system sends information about the fund transfer in terms of a payment request to the customer's mobile equipment. The customer is prompted to electronically sign the payment request. Hermes receives the signed request.

2. **Send the signed request to the financial institute.** Hermes forwards the signed request to the financial institute and gets a signed receipt.

3. **Validate the receipt of the financial institute.** The systems checks the signature of the receipt with the aid of the trust center.

4. **Send response to the service provider.** Hermes notifies the service provider of the successful transfer of funds by forwarding a copy of the receipt.

Figure 1 does not include extension points such as: If the customer does not receive the payment request, then use alternative payment-request mechanisms (a different use case); if the authorization for the bank transaction is invalid, then renegotiate authorization (a different use case).

3.2. Design goals

The discussion of the Hermes system concentrates on three attributes that distinguish it from other m-commerce systems: ease of use, leanness, and open standards/modifiability.

Ease of use. Ease of use concerns the user-group stakeholder and service providers who want to integrate Hermes in their service. The user group wants the system to be fast and simple, so they do not have to stay connected for too long and, in the best case, do not require instructions to use the service. The service providers want Hermes to have simple and straightforward external interfaces, so that they can integrate it easily.

Leanness. Leanness concerns the corporation partner. They want the system to be lean, so (1) the hardware requirements are low, (2) its deployment is simple, and (3) the maintenance of the system is straightforward.

Open standards/modifiability. Since new devices and protocols are emerging rapidly, open standards and modifiability are also a matter of concern of the corporation partner.

Besides these three design goals the system must meet other criteria so the corporation partners can deploy it. As a representative for the other requirements that have been taken care of by the designers, the following paragraph describes the requirement security. It also highlights the dependencies between design goals and stakeholders.

Security. Security is important for all three stakeholders: the corporate partner, the service providers, and the user group. The corporate partner is liable for the system, so it must be secure. The service providers do not want anyone to misuse the system. And finally, the user group must consider the system to be safe, since otherwise they will not use it.

Electronic signatures solve this security problem. As a large amount of money is involved in services (e.g., bet-and-win games), electronic signatures are a perfect means for the authorities of the service to claim the fees. However, if the system uses electronic signatures the jurisprudence becomes a stakeholder. It requires the system to comply with the electronic signature law of the specific country.

4. System description

The following description of the Hermes system includes an overview of the components and connections and a detailed depiction of its communication component.

4.1. Overview

Figure 2 depicts Hermes consisting of the computational nodes and the components of the system. In the following paragraphs this diagram is discussed; separated in a description of the components and a description of the communication between the components.

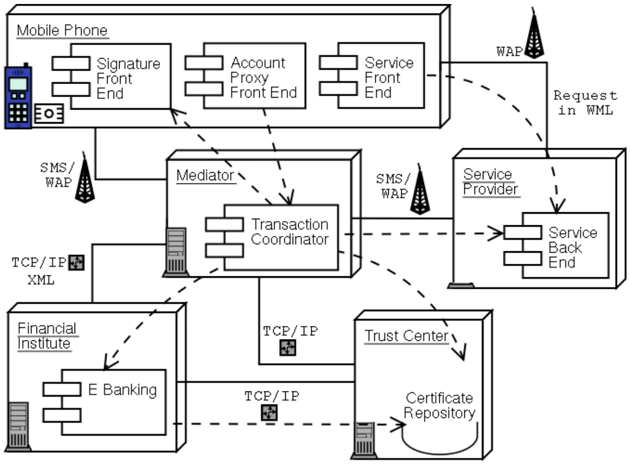


Figure 2. The Hermes system overview. The seven main components are spread across five nodes. The components located at the ME are front ends to services offered by service providers or the mediator. Service providers transmit requests to the transaction coordinator **that collaborates with the financial institute and the trust center to process these requests.**

Components. The service–front-end and the service–back-end components are developed and maintained by the service provider. These two components implement the business logic of the m-commerce service that the service provider offers to the customer. The service back end eventually sends transaction requests to the transaction coordinator. The mediator node provides the transaction coordinator component that is in charge of the transaction processing. It receives transaction requests from the service back end, forwards them to a signature front end, eventually routes them to a financial institute, and notifies the service provider and the customer of finished transactions. Every service provider uses this transaction coordinator component to issue transaction requests. Services communicate with the transaction coordinator component via specific extensible markup-language (XML) documents. The signature front end (located at the mobile equipment of the customer) receives transaction requests from the transaction coordinator. This component allows the customer to electronically sign the transaction request. The e-banking component is located at the financial-institution node. It receives signed transactions from the transaction coordinator and executes them (i.e., transfers money from one account to another). The communication protocol between the two is based on XML. Both, the e-banking and the transaction coordinator component require the trust center to validate signatures. The e-banking component needs to verify the customer's electronic signature; The transaction coordinator component needs to verify the request sent by the service provider, the signed request returned by the customer, and the response of the financial institute to the transfer-execution call.

Communication. The service-provider, the mediator, the financial-institute, and the trust-center node communicate via encrypted-TCP/IP connections. The communication between the service–front-end and the service–back-end component depends on which technology the service provider wants to use (e.g., wireless application protocol, wireless markup language, SIM application toolkit). The signature front end and the transaction coordinator communicate via the short message service (SMS). To communicate via SMS an SMS center (SMSC) is required by the transaction coordinator. The SMSC enables the transaction coordinator component to send SMS messages to a mobile equipment, which makes them accessible to the signature front end. The communication between the financial institute and the trust center depends on the preferences of the financial institute and the trust center.

4.2. Transaction coordinator

The central part of the system is the transaction-coordinator component (see Figure 3). It controls the workflow of the transaction starting with receiving a transaction request from the service provider up to the point where the provider gets the receipt from the financial institute. The system is modeled as a set of active entities that concurrently process a certain aspect of the payment request. Communication takes place via asynchronous message passing. The request receiver listens for new payment requests, parses the corresponding XML document, checks some basic criteria, eventually stores the requests and notifies the sender of successful or failed receipt of the request. Finally, the request receiver queues the payment request at the front-end communicator. The front-end communicator's responsibility is to generate messages suitable for displaying on the front end in response to events it obtains from the system. It makes use of the communication subsystem which provides means request/reply-message transactions and unreliable data pushes. Upon receipt of a payment request the front-end communicator constructs the message to be singed by the customer and sends it. When receiving the signed message from the customer, it is passed on to the verifier. If the communication fails it will queue the message at the acknowledger which informs the service provider of the failed transaction. The verifier is responsible for validating the signature of a given message. It checks the signature of the customer and eventually passes the data on to the financial-institute communicator. If the verification fails the verifier sends events to the front-end communicator and the acknowledger to inform both the service provider and the customer about the failure. The financial-institute communicator generates the XML document that requests the financial institute to execute the transaction and sends it. It waits for the signed receipt which it then transmits to the verifier. The verifier checks the signature of the financial institute, and passes data to the acknowledger and front-end communicator to inform the customer and the service provider of either success or failure of the transaction.

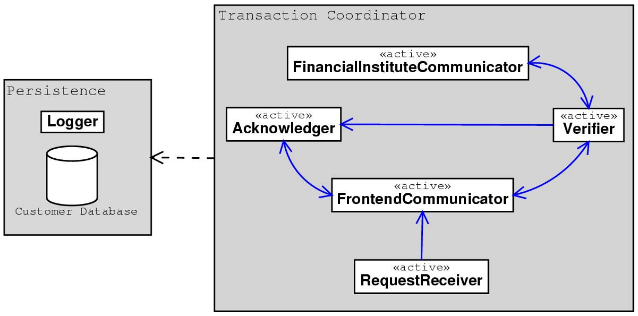


Figure 3. The Transaction Coordinator. It consists of five subcomponents that communicate with each other via asynchronous events.

4.3. Communication with the front end

The design of mobile payment systems requires taking special care for the subsystem that handles the communication with the front end. The ability to convey messages with many different devices is one of the key attributes of such a system. Hermes can communicate with different mobile equipment, especially with mobile phones and PDAs. Mixed channel communication is also possible. A signature request could be sent to a customer via TCP/IP to her PC, then downloaded to the mobile phone over a serial link and finally returned to Hermes with an SMS. For flexibility reasons the communication part in the Hermes system is abstractly coupled with the remaining components in the system.

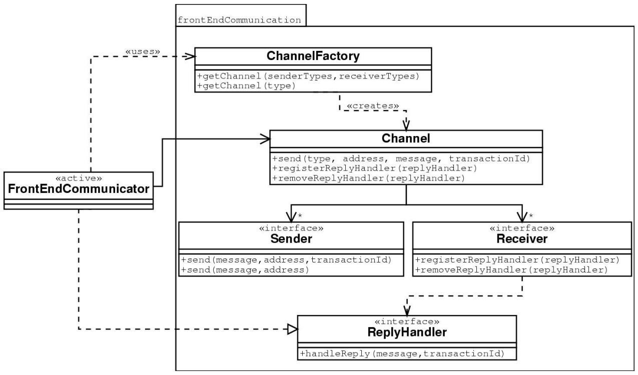

frontEndCommunication
«uses»
ChannelFactory
+getChannel(senderTypes,receiverTypes)
+getChannel(type)
«creates»
Channel
+send(type, address, message, transactionId)
+registerReplyHandler(replyHandler)
+removeReplyHandler(replyHandler)
«active»
FrontEndCommunicator
*
*
«interface»
Sender
+send(message,address,transactionId)
+send(message,address)
«interface»
Receiver
+registerReplyHandler(replyHandler)
+removeReplyHandler(replyHandler)
«interface»
ReplyHandler
+handleReply(message,transactionId)

Figure 4. Communication subsystem. The class diagram shows the encapsulation of the communication. The receiver and the sender are independent to realize mixed channel communication.

To support a greater variety of communication channels and modifiability, the communication channel between the transaction coordinator and the signature front end is separated in a set of senders and a set of receivers. Figure 4 shows the class diagram for the communication subsystem. The front-end communicator uses the channel factory to obtain a communication channel suitable for the customers. This channel contains the senders and the receivers of the types requested by the front-end communicator (e.g., SMS, WAP, TCP/IP). Based on the preferences of the customer the front-end communicator can choose one of the different means of communication. The front-end communicator processes replies from any of the receivers in the same way the customer is free to choose the communication media for the response. The message exchange with the customer is asynchronous. Message sending does not block and the front-end communicator registers a reply handler with the channel that is called back upon receipt of a reply message.

5. Simulation environment and sample run

The following sections describe the simulation environment that has been used for the system prototype. Furthermore it includes a sample run that provides further information of the Hermes system.

5.1. Simulation environment

The simulation environment is shown in Figure 5. A workstation serves as simulation server. It hosts all the service providers: a virtual trust center, a virtual bank, and a virtual shop. Additionally it hosts the transaction coordinator. The SMSC is simulated by a mobile phone that is connected via the serial-line interface of the workstation. The transaction coordinator uses the extended advanced telephony (AT) commands (defined in the GSM standard 07.07 [4] and 07.05 [3]) to communicate with the SMSC. Finally, the client is simulated by another mobile phone that hosts the front ends and a library supporting cryptographic functions. The SMSC and the client use the GSM network as transmission medium. The short message service is the protocol used by both entities to converse.

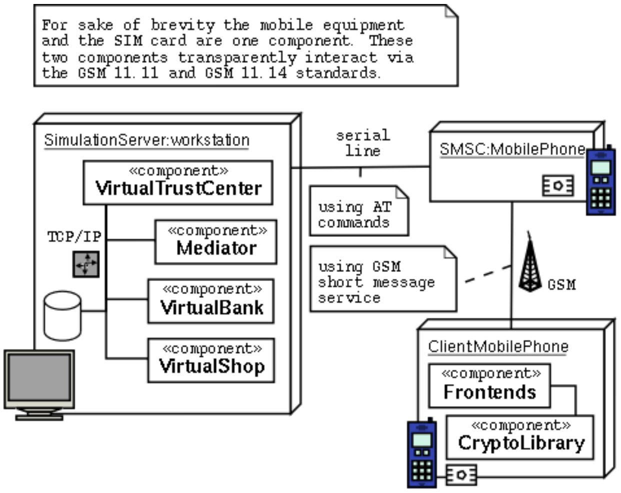

For sake of brevity the mobile equipment and the SIM card are one component. These two components transparently interact via the GSM 11.11 and GSM 11.14 standards.
SimulationServer:workstation
«component»
VirtualTrustCenter
TCP/IP
«component»
Mediator
«component»
VirtualBank
«component»
VirtualShop
serial line
SMSC:MobilePhone
using AT commands
using GSM short message service
GSM
ClientMobilePhone
«component»
Frontends
«component»
CryptoLibrary

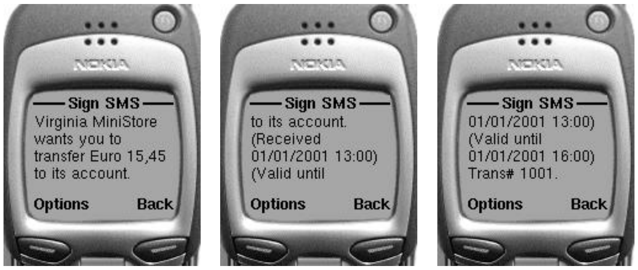
Figure 5. The simulation environment.

The virtual services running at the workstation are implemented in Java version 1.3. To reduce the required hardware for the simulation the different processes were collocated on the same host and the different databases were managed by one database management system (mySQL version 3.23.38). To parse the XML message the prototype uses the JAXP 1.1 API and the reference implementation of Sun.

5.2. Sample run

A customer browses in the virtual shop using conventional e-business and decides to buy a sound clip. To bill the customer, the virtual shop messages the transaction coordinator that it wants the customer to sign a bank transaction. The transport mechanism for the message is TCP/IP. In the prototype this communication is not secured, however, in the deployed application the shop will sign the transaction using an asymmetric cryptographic algorithm and the transmission itself is encrypted using the secure socket layer. The transaction coordinator receives the message, parses the XML, verifies the signature, stores the request in the database, and returns an acknowledgment. The front-end communicator generates an SMS message and forwards it to the mobile phone of the client using the SMSC.

The front-end communicator uses the AT commands to communicate with the SMSC (i.e., AT+CMGS for sending and AT+CMGR for reading). Most mobile phones include an internal modem. The mobile phone in the simulation that acts as SMSC has been a Siemens S35i.



Sign SMS
Virginia MiniStore
wants you to
transfer Euro 15,45
to its account.

Sign SMS
to its account.
(Received
01/01/2001 13:00)
(Valid until

Sign SMS
01/01/2001 13:00)
(Valid until
01/01/2001 16:00)
Trans# 1001.

(a) Part I. (b) Part II. (c) Part III.

Figure 6. The SMS to be signed. The three figures show the complete SMS that Hermes asks the user to sign.

Figure 6 shows the SMS that the customer receives. It contains information about the person or organization that requests the money transfer, the creation date, the expiration date, and the transaction number. The customer can use the transaction number to lookup a more elaborate description of the transaction (e.g., who exactly the Virginia MiniStore is, a detailed description of the receipt). The lookup service is not part of the signature prototype. It can be easily implemented using a WAP service.

Now the customer signs the message and returns the message to the SMSC using commercial crypto-plugins. The SMSC notifies the transaction coordinator that a new message has arrived (AT+CNMI=1,X,X,X,X,X and +CMI:"1"). The Mediator parses the short message and generates an XML message that contains the original request from the virtual shop, the original answer of the client, the transaction (i.e., payer account information, payee account information, transfer amount information), and signs the message. The virtual bank receives the message and transfers the money from one account to the other. Then it returns a signed receipt to the transaction coordinator. The transaction coordinator receives this receipt and sends an acknowledgment short message to the client and one to the virtual shop.

6. Evaluation

The evaluation of the architecture of the presented system follows the software-architecture analysis method (SAAM) described in [10]. One of the main arguments of this method is that architectures are not intrinsically good or bad, but they have to be evaluated bearing in mind the stakeholders' needs and goals. The set of requirements presented before lead to meaningful benchmark tasks, against which the architecture is evaluated. The SAAM evaluation presented here concentrates on the third design goal of the architecture: open standards and modifiability. The following section shows that these design goals are met by evaluating the software quality attributes integrability and modifiability (especially adaption of new operating environments and extension of capabilities [12]).

6.1. Integrability

Additional service providers. We consider a mobile counterpart of a well understood e-commerce application as a representative benchmark task to test the integrability of the system into the service-provider's software. The scenario assumes that a service provider has built an electronic bookstore based on WAP. The customer can browse for books, put them in a shopping cart, and eventually order the contents of the shopping cart. Now she wants to make use of the payment service offered by Hermes.

The integration of the Hermes system into the electronic bookstore is straightforward. After the customer submits the order, the bookstore empties the customer's shopping cart and transmits a fund-transfer request to the Hermes system, by generating and sending a specific XML document. Now the customer can stay at the site of the bookstore or pay her open invoices. The bookstore will not process the customer's order until it gets the payment receipt in XML together with the signature of a financial institute from the Hermes system. After this has happened the service provider sends an SMS to the customer telling her that the order was successful.

Multiple GSM network operators. The next benchmark task deals with giving access to customers of other GSM network operators. The intent of this task is to evaluate the openness of the system.

A customer attempts to buy a desired book. The service provider sends a fund-transfer request to the Hermes system, however, the customer is not yet registered. So the service provider redirects the customer to the account-proxy component. There the customer signs on the system by filling in the information needed by the transaction coordinator component. Since this component uses telephone numbers as identification numbers and electronic signatures as means of authentication, it is able to deal with customers belonging to different GSM networks. However, there is an important restriction on the openness of the system; the mobile equipment of the customer needs an algorithm to create electronic signatures. Latest chip plugins include such algorithms.

Support of small and restricted devices. Another benchmark task is to evaluate the architecture with respect to how well it corresponds to the shortcomings of mobile devices. So it tests whether the system scales down to the limitations of mobile equipment.

The business logic dealing with the sign process is separated into the signature–front-end, account-proxy–front-end, and the account-proxy component. This separation allows to tailor each component to the capabilities of the mobile equipment, whereas, remaining business logic is located in the account-proxy component at the mediator node. For example, WAP allows for dynamic content and therefore it is used by the data-centric component (account-proxy–front-end). In contrast to WAP, SIM application toolkit programs (STK) provides better security and performance, therefore the CPU-bound component (signature–front-end) is an STK application.

6.2. Modifiability

Adaption of new operating environments. A number of powerful mobile devices will emerge on the market. Hermes has to support them as they become connected to the network and become more widely accepted. Integrating such new devices typically requires adapting the communication and transport protocols. Therefore the integration of new PDAs is a representative benchmark task to evaluate modifiability.

In the first place the signature–front-end component has to be ported to the new device. In the worst case the component has to be written from scratch, because modern PDAs allow more complex interfaces than older ones. The account-proxy–front-end and the signature–front-end component may even be integrated into one component. As the architecture fulfills the quality attribute “support of small devices”, the portion of the system that has to be recoded is little. On the other hand the transaction coordinator component has to be changed only in two points: First, the communication protocol has to be changed, which is completely encapsulated in the communication with the signature–front-end component. Second, it might be necessary to make changes in the account-proxy component. These changes affect only the representation of the content and not the way the content is handled. Therefore, these changes are minor ones. In addition it is likely that the PDA supports WAP, so only the front-end–communication and the signature–front-end components need to be changed, leaving the account-proxy component unaffected.

Therefore the modularity of the architecture makes it apt for the new operating environments (e.g., new PDAs, new mobile phones) that are likely to emerge.

Extension of capabilities. If the system becomes available on more powerful devices, its functionality will increase. Extending a shopping service, the mediator could decide to provide a general-document–signing service. This service could be used to electronically sign emails as well as contracts. Such a service would be of limited value as long as documents have to be transmitted via the SMS protocol and displayed on a mobile-phone screen. But with a PDA and an apt transport protocol with greater bandwidth this document-signing functionality might come in consideration. This extension would be a suitable benchmark task to evaluate the modifiability quality attribute.

Depending on the way the signature–front-end component has been implemented it might be necessary to modify it. It needs to distinguish between payment requests and a more general-document–signature requests. Special care when designing this component makes these changes unnecessary. The account-proxy component needs to be enhanced to support more flexibility with documents of different classes. The processing entities in the transaction coordinator can reused with little and localized changes. Since there is no need for a financial-institute communicator, the order of processing has to change. For the active entities communicate via event queues, there are no direct dependencies and the can be easily reordered.

7. Conclusion

In this paper we presented Hermes; it is an m-commerce platform that utilizes electronic signatures in a way which complies with the electronic-signature directive of the European Council. It enables service providers to offer services to mobile-equipment customers such as general-purpose shops, bet-and-win games, or commercial databases. As the system uses electronic signatures, it prevents fraud and misuse (e.g., a customer already used the service but does not want to pay his bill—he claims that the payment system has no legal basis and does not pay his bill).

The results of the SAAM evaluation presented before demonstrate that at the architectural level the system meets the intended design goals integrability and modifiability: (1) the special communication structure offered by Hermes eases integration of new services and new devices, and

(2) its encapsulated structure supports modifiability, so extending the functionality or changing system components is done on a local scale.

The future work of this project is twofold. First, such systems cause much administrative and legislative discussions between the stakeholders about, for instance, technology usage, distribution paths, and key management. So the issue is to find possible solutions for these problems. Second, building on top of electronic-signature technology and the Hermes system, it is possible to extend it to support general-purpose documents. Then services are not limited to shopping, but can be extended to, for instance, contracting or signed-email services.

References

[1] Brokat. WWW Site. `http://www.brokat.com`.

[2] Paybox. WWW Site. `http://www.paybox.de`.

[3] Digital cellular telecommunications system (Phase 2+);Use of Data Terminal Equipment - Data Circuit terminating; Equipment (DTE - DCE) interface for Short Message Service (SMS) and Cell Broadcast Service (CBS). ETSI standard, 1998. Version 7.0.1, GSM 07.05.

[4] Technical Specification Group Terminals; AT Command Set for GSM Mobile Equipment. ETSI and 3GPP standard, 1998. Version 7.6.0, GSM 07.07.

[5] Austria. Bundesgesetz über elektronische Signaturen. Bundesgesetzblatt, Aug. 1999.

[6] M. Borcherding. Mobile elektronische Signaturen und das mSign-Protokoll. In *2001—Odyssee im Cyberspace? Sicherheit im Internet*, 7. Deutscher IT-Sicherheitskongress des BSI 2001, page 155 to 164. Bundesamt für Sicherheit in der Informationstechnik, SecuMedia Verlags-GmbH, Ingleheim, Germany, May 2001. (*Mobile Electronic Signatures and the mSign-Protocol* in 2001—Odysee in Cyberspace? Internet Security).

[7] European Parliament and European Council. Directive 1999/93/EC Of The European Parliament and Council of 13 December 1999 on a Communitiy framework for electronic signatures. *Official Journal of the European Community*, pages 13/13–13/20, Jan. 2000.

[8] S. Fischmeister, G. Hagleitner, W. Pree, and G. Pomberger. Symbolon—A Novel Concept for Secure E-Commerce. In *Proc. of The First IFIP Conference on E-Commerce, E-Business, E-Government*, Oct. 2001.

[9] J. Golic. Cryptanalysis of Alleged A5 Stream Cipher. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT'97: International Conference on the Theory and Application of Cryptographic Techniques*, LNCS1233, pages 239–255. Springer-Verlag, Berlin, Germany, 1997.

[10] R. Kazman, L. Bass, M. Webb, and G. Abowd. SAAM: A Method for Analyzing The Properties of Software Architectures. In *Proc. of 16th Int. Conf. on Soft. Eng.*, page 81 to 90, 1994.

[11] mSign Consortium. *mSign Protocol Specification*, first edition, Oct 2000.

[12] O. Oskarsson. *Mechanisms of Modifyability in Large Software Systems*. PhD thesis, Linköping Studies in Science and Technology Dissertations No. 77, 1982.

[13] G. Roman, G. Picco, and A. Murphy. Software Engineering for Mobility: A Roadmap. In *Proc. of FSE*, pages 243–242, 2000.