An Interconnection Architecture for Micropayment Systems

R. Párhonyi, D. Quartel, A. Pras, L.J.M. Nieuwenhuis
University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science
P.O. Box 217, 7500AE Enschede, The Netherlands
+31 53 489 4880
{parhonyi, quartel, pras, l.j.m.nieuwenhuis}@ewi.utwente.nl

ABSTRACT

In the next years the market for low value online content, like music and videos, is expected to grow substantially. To allow "pay-per-use" of such content, micropayment systems are expected to play an important role. Since there are already many competing micropayment systems on the market, customers and merchants are forced to use multiple systems. To overcome the problems associated with using multiple systems, the research presented in this paper builds upon the idea of payment gateways that interconnect these systems. We introduce a generally applicable interconnection method such that the interconnection of these systems can be easily realized in a systematic way. This approach consists of (de)enhancing the existing micropayment services towards a uniform service level before the interconnection takes place. This paper presents the main functional characteristics of existing payment systems, and proposes modification strategies for existing micropayment systems to provide the uniform payment service. The modifications are required before the interconnection can take place.

Categories and Subject Descriptors

H.1.1 [Models and Principles]: Systems and Information Theory – General systems theory, Information theory

General Terms

Design, Economics, Standardization, Theory.

Keywords

micropayment system, micropayment interconnection architecture, uniform micropayment service, micropayment gateway

1. INTRODUCTION

In the next years the market for low value online content, like music and videos, is expected to grow substantially [1]. To allow "pay-per-use" of such content, micropayment systems are expected to play an important role. Although many micropayment systems already exist, none has obtained a dominant market position. In fact, existing systems are often used within restricted communities, usually within national borders. In the light of globalization, however, future micropayment systems must be useable across country borders, preferably at a world-wide scale, as the demand for cross-border payments is growing [2].

Currently, both customers and merchants use multiple micropayment systems to serve all their needs [3]. As a consequence, customers and merchants need to install multiple software packages and hardware devices, learn the usage of several systems, manage multiple accounts and e-wallets, remember multiple passwords, trust different payment system operators and so on. To overcome these problems, a hybrid micropayment system has been proposed in literature [4]. The idea behind that system is to allow customers and merchants to use their micropayment system of choice, while still being able to pay each other in a seamless manner regardless the choice of the other party. Such a hybrid system is able to interconnect existing micropayment systems, based on rules that define the mapping between the various systems. Due to the diversity of available micropayment systems, however, the interconnection is far from trivial and not solved yet. This paper therefore proposes an interconnection method, and presents a partial design of the hybrid system.

Figure 1 illustrates the proposed architecture of the hybrid payment system, which has two users: the *Consumer* and the *Provider*. The *Consumer* requests and consumes the chargeable products (e.g., online content, services), and pays for it via its *Payer* function. The precise operation of the *Payer* depends on the underlying micropayment system A; in fact the *Payer* can be seen as a user of, or consumer on top of micropayment system A. Also at the user's side two roles can be identified: *Provider* and *Payee*. The *Provider* offers and delivers the products requested by consumers, and receives payments via its *Payee* function. Like the *Payer*, the operation of the *Payee* depends on its underlying payment system. This system, however, need not be the same as the one used by the *Payer*. In fact, the *Payment Gateway* (*PG*) is responsible for interconnecting the different payment systems, the *Payment Gateway* is therefore the core component of the hybrid payment system. The novelty of this architecture is that consumers and providers can always pay and be paid, respectively, in the same way, independently of the specific payment systems. All payment specific tasks are delegated to the *Payer* and *Payee*, which are part of the hybrid payment system and provide the hybrid payment service to the *Consumer* and *Provider*.

This paper introduces and elaborates a method to interconnect existing micropayment systems. This method can be used in the design of the hybrid payment system. Since the *Payment*

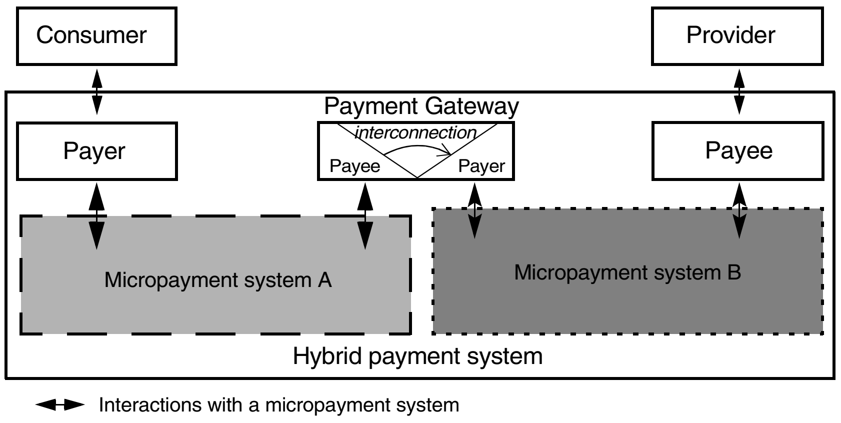

Figure 1: Hybrid micropayment system

Gateway within the hybrid payment system should be able to interconnect a variety of existing payment systems, this paper also identifies the main characteristics of such existing systems.

The structure of this paper is as follows. Section 2 presents the interconnection method. Section 3 identifies the main functional characteristics of existing micropayment systems. Section 4 discusses the design of the uniform payment system. Section 5 presents a basic interconnection scenario, and discusses trust and security issues within the hybrid payment system. The conclusions are presented in Section 6.

2. INTERCONNECTION METHOD

The question how to interconnect existing (sub-)systems to create a larger or global system, is a traditional one and has been studied in literature before. Within the context of computer networks, for example, the embedding of heterogeneous, especially incompatible sub-networks into a global network was such a question. In literature [5] two approaches have been identified to answer this question. One of them is to interconnect available sub-networks just as they are. The other approach is to use the sub-networks as building blocks, adapt their functionality such that their services[1] became compatible with each other, and then to interconnect the compatible systems with gateways. The adaptation is achieved by applying "*wrapping*" functions to the incompatible services. The latter approach was also proposed by the ISO 8648:1988 standard, which is intended for use in the design of network layer protocols, in cases where multiple "real networks" are to be interconnected and used [6].

Because of the similarities between the interconnection problem of networks and of micropayment systems, we identify two similar methods that can solve the latter problem. The first method is to take different payment systems and interconnect them in an ad-hoc manner. This approach requires the definition of all mappings between each pair of interconnected payments such that the interconnection must be bi-directional. Looking at the number of current micropayment systems and at their relatively short longevity (i.e., their availability on the electronic payments market), and considering future developments (e.g., appearance of new systems, increasing volume of micropayments), this approach would only work if the number of payment systems is small, hence a more generic method is required.

[1] The service defines the external behaviour of a system, as experienced by its users.

The second interconnection method is to harmonize the payment services of existing systems to a uniform level, called the *uniform payment service*, and interconnect these uniform payment services. A prerequisite for this method is that the harmonization of existing and future micropayment systems to the uniform payment service is possible. We call a system that provides the uniform payment service a *uniform payment system*, and a money transfer that is performed by such a system a *uniform payment*.

Figure 2 depicts the interconnection of two uniform payment systems that wrap the existing micropayment systems *A* and *B* from Figure 1. Each *Payer* entity is decomposed into (i) entity *HPayer*, which is a user of *Uniform payment system A* and provides the hybrid payment service to the *Consumer* or the PG's interconnection function, and (ii) entity *UPayer A*, which is a user of *system A* and provides the uniform payment service to the *HPayer* or the PG's interaction function. Similarly, the *Payee* is decomposed into entities *HPayee* and *UPayee B* such that the *HPayee* provides the hybrid payment service to the *Provider* or the PG's interconnection function, and *UPayee B* provides the uniform payment service to the *HPayee* or the PG's interconnection function, respectively. The composition of internal entities *HPayee*, *HPayer* and *interconnection* of the Payment Gateway is called the *Hybrid Payment Gateway* (*HPG*) in the sequel.

An advantage of this method is that only one set of mapping rules has to be defined for each existing payment system. Furthermore, the mapping rules between the uniform and the hybrid service (as realized by the *Payer* and *Payee*) and the interconnection rules between different instances of the uniform payment service have to be defined only once.

In this paper, we use the second method and show that the services of most existing payment systems can be harmonized into uniform payment systems that can be interconnected.

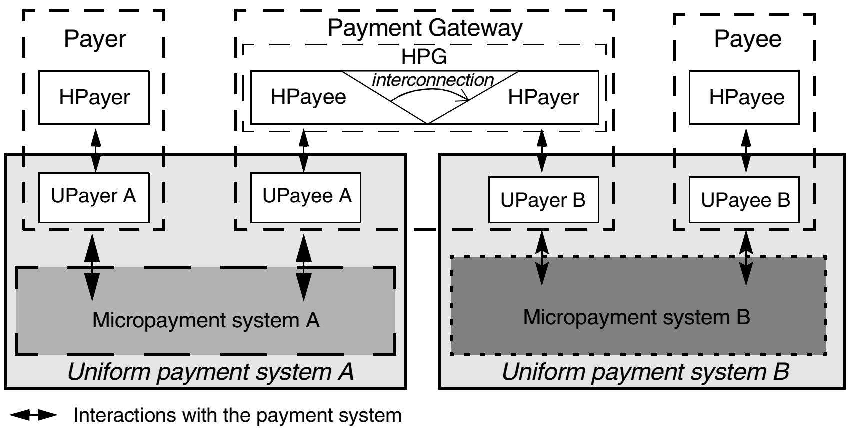

Figure 2: Interconnection of uniform payment systems

3. FUNCTIONALITY OF EXISTING MICROPAYMENT SYSTEMS

This section presents the main functional characteristics of existing payment systems. To be able to harmonize these systems, we need to determine their common functionality and decide whether this functionality provides a basis for the design and implementation of the uniform payment service. The remaining differences should be solved in the interconnection layer on top of the uniform payment systems.

The functional characteristics we consider are the interactions between the payment system and its users, the information that is exchanged in these interactions, and the relationships between these interactions and the exchanged information. For the interconnection problem only interactions need to be considered that have end-to-end significance, i.e., interactions in which different (end-)users are involved and that are related. These interactions are the initiation and acknowledgment of payments. Other interactions such as registration, login and reviewing the payments history have only local significance, and therefore do not need to be considered further. In addition, we abstract from the different ways in which the initiation and acknowledgement of payments can be implemented, because we consider that only the result of these interactions is important for solving the interconnection problem. The result of the initiation of a payment is that the payment is accepted by the underlying payment system. The result of an acknowledgement is that a user receives a confirmation that the underlying payment system completed a payment. Generally, an initiation is followed by an acknowledgement. However, due to error situations, a payment may not be processed after an initiation has occurred. In these situations, the general rule is that the payer bears the loss of money, but only up to a certain limit, as determined by the European Central Bank [2]. The probability that these situations occur is low since existing payment systems claim high reliability and money loss may only occur rarely (e.g., only one out of every million micropayment fails).

In our studies we analysed a large number of existing micropayment systems and classified them based on their main functional characteristics. We determined these characteristics based on information available on the web sites of these systems, implementation documents, performing payments with the systems, inspecting the source code of provider web sites, and capturing network traffic using Ethereal. The following sections present these characteristics and give examples of payment systems grouped in different categories.

3.1 Payment Initiations

We identified that three alternative ways are theoretically possible to initiate a payment, as depicted in Figure 3. The vertical lines represent the interaction points or service access points (SAPs) between a micropayment system and its users, arrows represent payment initiations, and the direction of the arrows indicates the direction of the information flow. We also give examples of the information exchanged in a payment initiation and search for common elements.

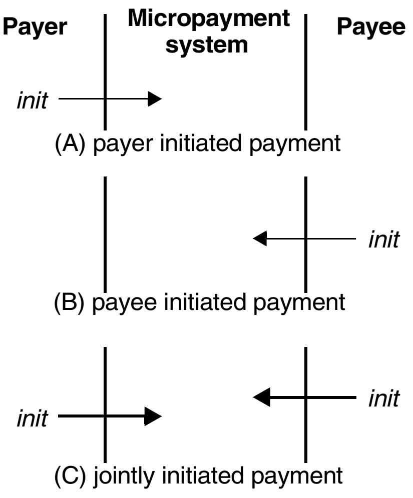
Payer
Micropayment system
Payee
init
(A) payer initiated payment
init
(B) payee initiated payment
init
init
(C) jointly initiated payment

Figure 3: Payment initiations

First, Figure 3 *A* represents the case in which the payer initiates the payment, called a *payer initiated payment*. Payment systems that fall into this category are, e.g., Minitix, Way2Pay, Wallie, Micromoney, Secoin, Teletik, Softpay, PayPal and Peppercoin. A Minitix initiation, for instance, contains the payer identifier (user name and password), the payee identifier, name and web site, the order identifier and description, amount and currency, validity period of the order, a return and an error URL to which the consumer will be sent to if the payment is completed or rejected, respectively. A Wallie payment initiation requires the account number of the payer, the identifier of the payee, the shopping cart identifier and the amount of money. A Way2Pay payment initiation requires the payer identifier (user name and password), the payee identifier and name, the product transaction identifier and item name, the amount of money, and the success and failure URLs.

Second, Figure 3 *B* represents the case in which the payee initiates a payment, called a *payee initiated payment*. No micropayment systems were found with this type of initiation. In case of bank payments, however, the automatic direct debit uses this type of initiation.

Third, Figure 3 *C* represents the case in which both users initiate a payment, which we call a *jointly initiated payment*. Because both users initiate the same payment, they both have to provide payment information such that the payment system can correlate the initiations, and then process the payment. Systems that fall in this category are, for instance, Bitpass, click&buy, PaySafeCard, PayNova and PayStone. In case of Bitpass and click&buy, payees initiate the transactions for each product only once at the moment the product is offered online. After that, payers initiate a payment for each product. A click&buy payment initiation requires a payer to provide the user name and password, and the URL of the content (this is used to identify the payee and make the correlation). A payee needs to supply the content name and description, the price and the period for how long the product will be available after the payment. A PayNova initiation requires the users to provide the session key (used for correlation), the payer identifier, the payee identifier, the order description, amount of money and currency, and the return URL. A PaySafeCard payment initiation requires the account identifier of the payer, the payee identifier, the provider transaction identifier, the amount and currency, account balance limit, OK and NOK URLs, and configuration information.

In conclusion, payers initiate the payments for the majority of existing systems. Regardless of the initiation type, the most common parameters of initiations are the payer and payee identifiers, the product transaction (or shopping cart, order) identifier, and the amount of money (value and currency). The other parameters of the initiations are related to the implementation of the payment systems. Because the amount of money is present in every initiation, and existing systems perform payments of different values and currencies, we will investigate what ranges of payments are supported in Section 3.4.

3.2 Payment Acknowledgements

Completed payments can be acknowledged to the payers and payees in two ways:

- the payment system provides an acknowledgement to the payer and/or payee;
- the payment system provides no explicit acknowledgement, instead it retrieves the paid product from the provider, and delivers it to the consumer (e.g., click&buy).

In the first case we can say that the payment system "performs only payments", while in the second case, the payment system "does more than payments". Confirmation information of completed payments is only provided in the first case and it differs from system to system.

We identified three alternative ways to acknowledge a payment explicitly, as depicted in Figure 4. We also give examples of the information exchanged in a payment acknowledgement and search for common elements.

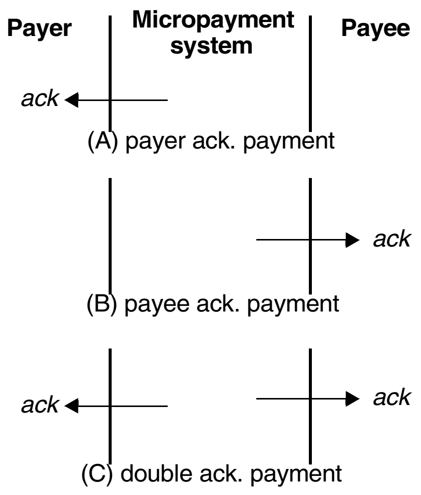

Figure 4: Payment acknowledgements

First, Figure 4 *A* represents the case in which the payer receives an acknowledgement, called a *payer acknowledged payment*. Way2Pay and PaySafeCard fall into this category. A Way2Pay acknowledgement contains the name and e-mail of the provider, the amount of money transferred, the content description, the return URL, the transaction identifier (set by the provider), a unique Way2Pay transaction number (of the payment), an error code and description (if the payment is completed this value is null). A PaySafeCard acknowledgement for a payer contains a return (or OK) URL of the provider, which was specified during the initiation and contains information, e.g., product transaction identifier, needed by the provider to deliver the content.

Second, Figure 4 *B* represents the case in which the payee receives an acknowledgement, called *payee acknowledged payment*. Bitpass falls into this category, because it sends the payee a ticket, which describes the context of the payment and the URL of the paid content. The payee verifies the ticket and delivers the content, if the ticket is valid.

Third, Figure 4 *C* represents the case in which both the payer and payee receive acknowledgements, called *double-acknowledged payment*. Systems that fall into this category are Minitix, Wallie, Centipix, and PayNova. A Minitix acknowledgement for a payer contains the amount of money paid and the return URL of the provider. A Minitix acknowledgement for a payee contains the payee identifier, the order identifier, the paid amount of money and currency, a stack and ticket identifiers (which identify the payment within Minitix).

In conclusion, considering systems that support acknowledgements, in most cases the payees receive payment acknowledgements. The most common parameters of acknowledgements are the product transaction identifier (or context of payment) and payment identifiers (or transaction number, or ticket identifier). The other parameters of the acknowledgements are specific for the implementation of the payment systems.

3.3 Usage Conditions

The primary condition for using a payment system is the condition the system imposes on the way it is paid by its users. Concerning this usage condition, micropayment systems can be divided into non-credit and credit systems [7].

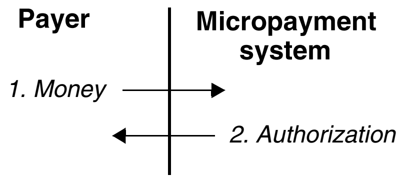

Figure 5: Pre-paid system

A non-credit system requires payers to transfer money to the system before they can use it (Figure 5). The amount of money transferred to the system is stored in the form of electronic money and the payer receives authorization to use it for payment. Payment initiations are accepted and authorized by the system until the account balance becomes insufficient. A non-credit system is called a *pre-paid system* [7]. Examples of pre-paid systems are Minitix, Wallie, Way2Pay, PaySafeCard, Microeuro, Centipix, PayNova and Softpay.

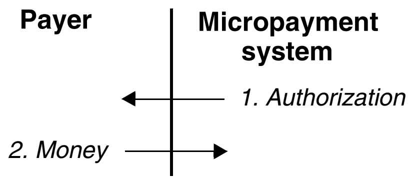

Figure 6: Post-paid system

A credit system authorizes payers to initiate payments before they transfer money to the system (Figure 6). Before payers receive such an authorization, they need to present the system a money source, e.g., a bank account, from which money is transferred to the system. If the money source is valid, the payer is authorized to use the system. At the same time, the system may set a credit limit for the payer. Payment initiations are accepted until the total value of payments is below the credit limit, if applicable. Periodically or when the limit is reached, a money transfer occurs from the specified source to restore the account balance. A credit system is called *post-paid system* [7]. Examples of post-paid systems are click&buy, Peppercoin and PayPal.

Other conditions for payees to make use of a payment system is to register themselves. Because this is a general requirement for all payees and the registration hardly differs, we do not consider it for the interconnection problem.

3.4 Supported Currencies and Payment Values

Payment systems may support different currencies and different minimum and maximum payment values. Minitix, for instance, supports payment values between €0.10 and €10 and no other currencies. PayNova supports payment values between €0.10 and €1000, and also supports Great Britain Pounds, Swedish Crowns, Danish Crowns, Norwegian Crowns and US Dollars. PayStone supports payment values between US$0.25 and US$1000, and also supports Canadian Dollars.

4. UNIFORM PAYMENT SERVICE DESIGN

This section presents the design of the uniform payment service, which forms the basis for the interconnection approach proposed in Section 2. The uniform payment service is defined by the interactions that can occur between a uniform payment system and its users, including the information exchanged in these interactions, and the relationships between the interactions. But first we explain the design decisions underlying the uniform payment service.

4.1 Design Decisions

Section 3 presented the commonalities and differences between existing payment systems. Concerning the differences, it has to be decided whether these differences can be solved at the interconnection layer, or within the uniform payment system by adding some extra functionality to the existing payment systems. In the latter case, an assessment should be made as to whether all, or at least most, existing payment systems can be (de)enhanced to provide the uniform payment service.

4.1.1 Uniform Payment Initiations

Design decision: *the uniform payment service supports payer initiated payments.* The motivation for this decision is that existing payment systems should change as little as possible, and most payment systems support payer initiated payments. Furthermore, the consequence of supporting jointly initiated payments would not only be that the majority of existing payment systems must be changed, but the interconnection of systems that support jointly initiated payments is more complicated than of systems that support the interconnection of payer initiated payments.

For existing systems that support jointly initiated payments we distinguish two cases, based on the time difference between the initiations. In the first case, both the payer and the payee initiate the payment within a few seconds. This case can easily be adapted to support user initiated payments, because the *UPayer* and *UPayee* (see Figure 2) can exchange the payment information received from an *HPayer*, and subsequently request jointly the payment from the existing system.

In the second case, the payer may initiate a payment weeks or months after the payee has initiated the payment. Typically, a payee initiates only one payment for every product the provider sells. A payment is processed each time the payer provides the remaining part of the payment information. Such systems (e.g., Bitpass, click&buy) cannot provide the uniform payment service and should change such that they support payer initiated payments or fall into the first case of jointly initiated payments.

In addition to deciding on the type of payment initiation, it has to be decided which information is required in a payment initiation. The required information is defined in Section 4.2. This information should be kept as minimal as possible, since it impacts the amount of information that has to be stored in the *HPG* for auditing purposes, as imposed by the European Central Bank [2].

4.1.2 Uniform Payment Acknowledgements

Design decision: *the uniform payment service supports payee acknowledged payments.* The motivation for this decision is similar to the one on payment initiations: existing payment systems should change as little as possible, and most systems support this acknowledgement type.

In Section 3.2 we identified two ways for acknowledging payments and we grouped the payment systems accordingly. In the first group, there are payment systems that perform only payments and acknowledge the payments to both payers and payees or only to payees. These systems can easily comply to the uniform service by making only the *UPayee* (see Figure 2) to pass an acknowledgement on to the *HPayee*.

In the second group, there are systems that do more than payments and provide no acknowledgements to their users (e.g., click&buy). This means that these systems cannot provide the uniform service unless acknowledgement functionality is added, such that at least the payer or the payee gets an acknowledgement. In case an acknowledgement is provided to the payee, then the *UPayee* (see Figure 2) must pass this acknowledgement on to the *HPayee*. In case an acknowledgement is provided to the payer, then the *UPayer* must exchange this acknowledgment with the *UPayee*, which passes it on to the *HPayee*.

Similarly to the requirement on payment initiation information, the amount of information provided in a payment acknowledgement should be kept to a minimum. This information is stored in the *HPG*, and used to interconnect the uniform payments in which an hybrid payment is decomposed.

4.1.3 Usage Conditions

Design decision: *the uniform payment service supports pre-paid or post-paid payments*. From the payer's point of view, one of the most important differences between pre-paid and post-paid systems, is how it obtains the authorization to use a payment system for paying payees. Because of that difference, the financial risks and the responsibilities of users and systems involved, it is unlikely that a pre-paid system can function as a post-paid system, or the other way around. That is why we define two types of uniform payment systems: pre-paid (denoted as UPS_{pre}) and post-paid (denoted as UPS_{post}). In the sequel, the term *UPS* is used in case the pre-paid or post-paid characteristic of a uniform payment system is considered irrelevant.

4.1.4 Supported Currencies and Payment Values

Design decision: *the uniform payment service does not prescribe a fixed minimum or maximum payment value, nor prescribes a fixed currency*. Fixed limits on payment values are not imposed, because of the variance in minimum and maximum values (and currencies) supported by existing systems. In case global minimum and maximum payment values would be imposed, the range of supported hybrid payment values would be reduced significantly. Instead, the uniform payment service inherits the minimum and maximum payment values and the supported currencies from the underlying payment systems.

If two interconnected UPSs support different currencies and payment values, the interconnection layer will be responsible to solve these differences.

4.2 Interactions

Uniform payments are initiated by *HPayers* and are acknowledged to *HPayees*. To model these payment interactions, two service primitives (SPs) are introduced: the *UPayRequest* primitive, which occurs at the SAP between the *HPayer* and the *UPS*, and the *UPayConfirm* primitive, which occurs at the SAP between the *HPayee* and the *UPS*.

4.2.1 UPayRequest Service Primitive

An *HPayer* executes an *UPayRequest* SP to initiate a uniform payment. Before the *UPS* accepts the initiation, it authenticates the *HPayer* and authorizes the payment initiation or not. The authorization procedure depends on the type of the *UPS*, which is pre-paid or post-paid. In case the *UPS* accepts the initiation, the payment initiation cannot be reversed or cancelled. We note that the *UPayRequest* SP abstracts from a number of interactions that are commonly found in implementations of this service primitive, e.g., log in to the system and confirmation of the payment initiation.

Parameters of the *UPayRequest* SP provide the necessary information to the *UPS* to be able to perform the initiated payment. We observed before that generally four parameters can be identified in each payment initiation (see Section 3.1): the payer identifier, payee identifier, product transaction identifier and amount of money. Because there are no products exchanged between the *HPayer* and *HPG*, or between the *HPG* and *HPayee*, we call the product transaction identifier a *context identifier*. Other parameters could be the provider name and URL, product description and URL, success and failure URLs, etc. But since this information is not vital for performing payments, we abstract from it.

As a result, the *UPayRequest* SP has the following parameters:

- *HPayer ID*: a unique identifier (within the *UPS*) issued by the *UPS* to the *HPayer*. This *ID* is used to authenticate the *HPayer*, and to determine the source account of the payment;
- *HPayee ID*: a unique identifier (within the *UPS*) issued by the *UPS* to the *HPayee*. This *ID* is used to identify the *UPS*, to determine the destination account of the payment and the address (SAP) where the payment should be confirmed;
- *Context ID*: a unique identifier (within the *HPayee*) of the context of the payment. It is generated by the *HPayer* or *HPayee*;
- *Amount of money*: specifies in terms of value and currency the amount of money that will be paid.

The accounts of *HPayers* and *HPayees* are stored and maintained by the existing payment systems, and the *HPayer ID* and *HPayee ID* identify these accounts uniquely within the *UPS*. Usually, the ID of a paying user of an existing payment system, i.e., the *UPayer*, consists of a <*user name, password*> couple. This identifier does not have to change with the introduction of the *UPS*, which means that *HPayer* on top of the *UPayer* can be identified using the same identifier. A paid user, i.e., the *UPayee*, is usually identified by a single <*payeeid*> field. Also this identifier can be reused to identify the *HPayee* on top of the *UPayee*.

The conditions on the execution of the *UPayRequest* service primitive are the following:

- the *HPayer* and *HPayee IDs* must exist and be known to the *UPS*, so the *UPS* can authenticate the users and identify their accounts;
- a UPS_{pre} verifies the source account balance of the *HPayer* to determine whether it allows the new payment. In case the account balance is too low, the *HPayer* needs to transfer some money into its accounts and initiates the payment again. How the money transfer can be performed does not need to be defined in this stage of the design;

- a UPS_{post} verifies the credit limit of the *HPayer*. In case this limit is not reached yet, or not set at all, the UPS_{post} accepts the initiation. Otherwise, the balance of *HPayer*'s credit account should be restored first, and then the payment can be initiated again;
- the amount of money must be between the minimum and maximum payment values supported by the *UPS*, and the currency must also be supported by the *UPS*.

4.2.2 UPayConfirm Service Primitive

The *UPS* executes the *UPayConfirm* SP after a *UPayRequest* SP has occurred. In this interaction the *UPS* indicates to the *HPayee* the completion of a uniform payment. To identify the SAP where the SP should be executed, the *UPS* uses the *HPayee ID* specified in the *UPayRequest* SP. The *HPayee* is not allowed to refuse a payment confirmation (acknowledgement).

Parameters of the *UPayConfirm* SP provide information to the *HPayee* to be able to initiate a second uniform payment or to acknowledge the hybrid payment to the *Provider*. We observed before that the product transaction and payment identifiers are always provided in acknowledgements. Again, we call the product transaction identifier a context identifier. We can argue whether or not the transferred amount of money should be indicated to the *HPayee*. Because the payment is initiated by the *HPayer*, the *HPayer* could modify (decrease) the amount of money to cheat on the *HPayee*. In case the amount of money is also indicated to the *HPayee*, the *HPayee* is able to verify that the correct amount of money is transferred. Other parameters could be the payee identifier, product description, date and time of payment, etc. Since this information is not essential for the *HPayee*, we abstract from it. In conclusion, the *UPayConfirm* SP has the following parameters:

- *Context ID*: a unique identifier (within the *HPayee*) of the context of the payment, which enables the *HPayee* to identify the reason of the completed payment;
- *UPay ID*: a unique payment identifier (within the *UPS*) generated by the *UPS*. This *ID* is stored in the *UPS*, and identifies the source and destination accounts and the payment value. It is used, for instance, to trace back uniform payments (e.g., in conflict situations between *HPayer* and *HPayee*) or to offer support for audit [2];
- *Amount of money*: specifies in terms of value and currency the amount of money that was paid.

The condition on the execution of this service primitive is the completion of the money transfer from the given source account to the destination account.

4.3 Interaction Occurrence Conditions

We distinguish two type of occurrence conditions: local and remote. Local conditions specify conditions local to a SAP, i.e., conditions for the occurrence of interactions at the SAP between the *HPayer* and the *UPS*, or the SAP between the *HPayee* and the *UPS*. Remote conditions define the relation between interactions at different SAPs, i.e., between payment initiations and acknowledgements.

HPayers can initiate one payment at a time, so a new payment can be initiated after the previous initiation is accepted by the *UPS*. *HPayees* receive payment acknowledgements from the *UPS* after the *UPS* successfully completed a payment. *HPayees* receive one acknowledgement at a time.

A payment initiation is usually followed by a payment acknowledgement. The time difference between an initiation and an acknowledgement varies depending on the time needed for verifying payment information and performing the payment.

The reliability of the *UPSs* is determined by the reliability of existing payment systems. These systems claim that payments do not fail, error situations can be traced back and corrected, so no money loss situations can occur. In the rare even that a payment initiation is accepted by a *UPS* but no acknowledgement follows, the *HPayer* bears the loss of money.

Figure 7 depicts a time sequence diagram of three successive uniform payments initiated by a *HPayer*. Two of the initiated payments are acknowledged to the specified *HPayee*, the third failed, which is an unusual situation. This figure also indicates that the time needed to complete a uniform payment may differ.

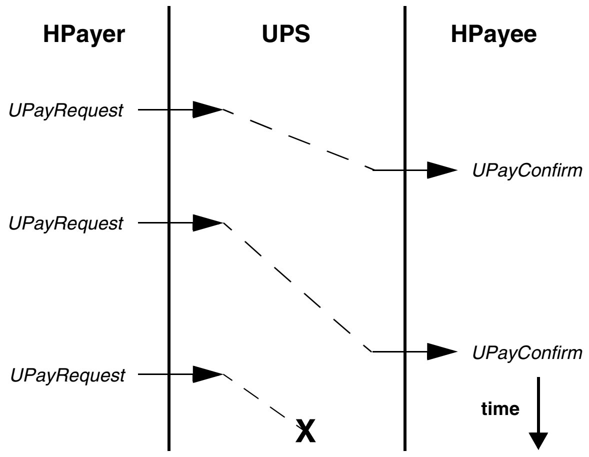

Figure 7: Uniform payment examples

5. BASIC INTERCONNECTION SCENARIO AND DISCUSSION

A basic interconnection scenario is illustrated in Figure 8. Consumer *John Doe*, user of the *Way2Pay* micropayment system, found a content provider *clipcollection.org*, which offers low priced music videos for sale and uses the *Minitix* micropayment system.

We assume that there are two uniform payment systems, *Uni4mpay* and *Uni4msys*, which wrap the *Way2Pay* and *Minitix* systems. Furthermore, we assume that the consumer and provider use the *PayAll* and *OneReceive* applications to initiate hybrid payments and receive acknowledgements of hybrid payments, respectively. These applications are implementations of the *HPayer* and *HPayee* entities, respectively. The *InterGate* hybrid payment gateway interconnects the *Uni4mpay* and *Uni4msys* systems.

In interaction (*A*) *John Doe* requests the content provider to deliver the latest video clip of a band called *Xperience*. The response (*B*) of *clipcollection.org* indicates to the consumer that he must pay €0.75 for the video using *Minitix* or the *Hybrid payment system*. Subsequently, *John Doe* initiates a

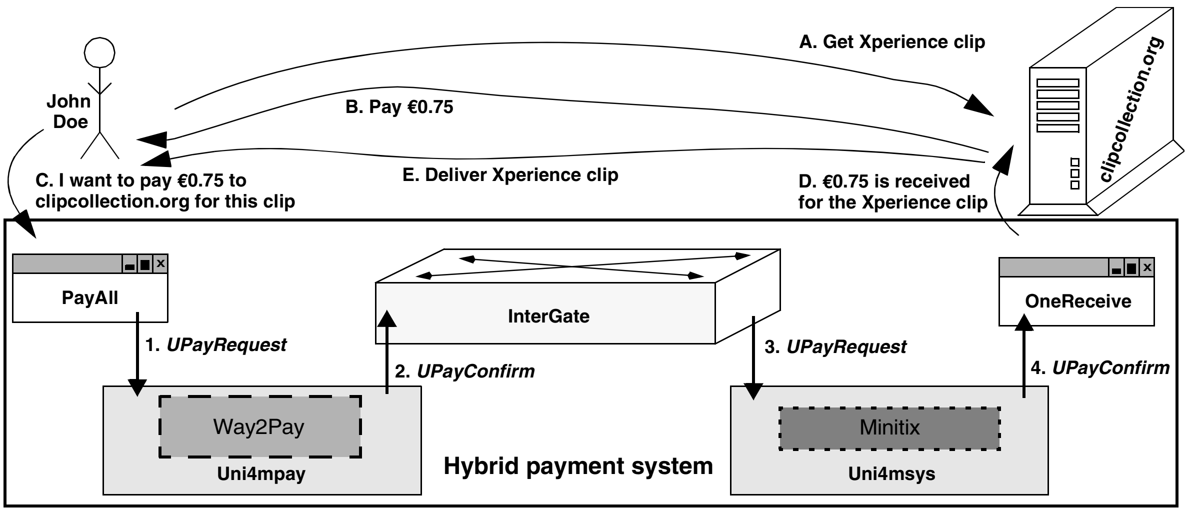

Figure 8: Basic interconnection scenario

hybrid payment (*C*) using *PayAll* to transfer €0.75 to the content provider. The initiation is accepted and *clipcollection.org* receives from its payment application an acknowledgement (*D*) that the requested amount of money is paid for the clip. Afterwards, the content provider delivers the *Xperience* clip (*E*).

Within the *Hybrid payment system*, the *PayAll* and *InterGate* prepare the initiation of the uniform payments based on the hybrid payment information received by *PayAll*. This means that they verify whether the interconnection is necessary (because the consumer could make an error), determine the *HPayer* and *HPayee IDs*, generate *Context IDs*, authenticate the consumer and identify the content provider (to find out whether the source and destination accounts exist), and check whether the first uniform system supports the payment value and currency (if not, they perform the currency exchange). Should any of these checks fail, the hybrid payment initiation is rejected. Otherwise, *PayAll* initiates the first uniform payment (1), which is performed by the *Uni4mpay* system and acknowledged to *InterGate* (2). Afterwards, *InterGate* initiates the second uniform payment (3), which is performed by the *Uni4msys* system and acknowledged to the *OneReceive* application (4). *InterGate* also stores information of the two uniform payments for auditing and payment trace back purposes. Finally, *OneReceive* confirms the hybrid payment to *clipcollection.org*.

5.1 Trust Discussion

Intuitively it seems that the *(Hybrid) Payment Gateway* is a central component of the hybrid payment system in which all other components should trust. This gateway is not a single point of trust, however.

The *HPG* has payer and payee contracts with the micro-payment systems in order to perform their interconnection. Hence, the *HPG* can be controlled based on these contracts by the operators of these systems. Moreover, due to its role and special functions (e.g., exchanging currencies, providing credit to post-paid customers), the *HPG* falls under the supervision of financial authorities. The consequence of these control possibilities is that we do not see real trust problems. Therefore, the probability that the *HPG* misuses its position to create fraud is low. Actually, it is lower than of a fraudulent customer. Nevertheless, the impact of the *HPG*'s misuse would be much higher than of a fraudulent customer or merchant, because it affects the operation of the whole hybrid payment system.

5.2 Security Discussion

Because the hybrid payment system is an online system, it is likely to be attacked by malicious users (attackers) in order to create fraud, steal money or misuse the system. The hybrid payment system must therefore be secure enough to prevent and detect such attacks. We assume that each *HPayer*, *HPayee*, *HPG* and *UPS* is secured such that attackers cannot fraudulently access them to retrieve information or money, and concentrate only on so-called "*man-in-the-middle*" attacks, which are attacks on the interactions between components of the hybrid payment system. Such attacks can threaten the overall integrity and operation of this system.

An attacker could, for instance, capture and modify the information used to initialize or confirm a uniform payments to create fraud by modifying the *Context ID* and increase the *Amount of money*, so the *HPayer* will pay for the product(s) of the attacker, or by modifying the *HPayee ID* and increase the *Amount of money* to re-direct the a bigger money transfer to its own destination account, etc.

To prevent or detect such attacks the payment information needs to be secured during transmission and any modification needs to be detected. The Secure Socket Layer (SSL 3.0) or the Transport Layer Security (TLS 1.0, [8]) security protocols could be used to secure the communication channel. The SSL, for instance, provides the authentication of the end-points and communication privacy over the Internet using cryptographic algorithms like RSA, DES, MD5 and SHA, and prevents eavesdropping, altering or forgery of payment information. Because of the authentication, each component will always know whether the other party is really the party it claims to be or not. Existing systems commonly rely on the HTTPS communication protocol, which uses the above mentioned SSL or TLS protocols.

6. CONCLUSIONS

This paper presents a generic and systematic method to interconnect existing and future micropayment systems. This method requires the (de)enhancement of micropayment systems towards a uniform service level. In this way, the number of mapping rules and the amount of information that must be stored is limited, which makes this method highly scalable. We also describe the main functional characteristics of existing micropayment systems, and show how these systems can be (de)enhanced to provide a uniform payment service.

The advantage of the uniform payment service is that the design and realization of the Payment Gateway that actually interconnects existing payment systems becomes much easier. This paper should therefore be seen as a step into the direction of a globally accepted hybrid payment system. The uniform payment service could guide the design of future electronic payment systems such that new systems can be interconnected easily with existing systems. In this way, the uniform payment service, possibly extended with interactions that have only local significance, could become a de facto standard for micropayment systems.

7. REFERENCES

[1] Ulph Jennings, R. et al., Downloads: 13% of Europe’s music market in 2007, *Report of Forrester Research*, May 2003

[2] European Central Bank, Report on electronic money, Frankfurt am Main, August 1998 (http://www.ecb.int/pub/pdf/other/emoneyen.pdf)

[3] Verband Deutscher Zeitschriftenverleger and Sapient, Paid content market in Germany, Berlin, December 2002 (http://www.paidcontent.org/germarket1.ppt)

[4] Párhonyi, R., Pras A., Quartel, D., Collaborative micropayment systems, *In Proceedings of World Telecommunications Congress 2004*, Seoul, Korea, September 2004

[5] van Sinderen, M. and Vissers, C.A., An architectural model for network interconnection, *In the Proceedings of the EUTECO - European Teleinformatics Conference*, 1983

[6] ISO 8648:1988, Information processing systems -- Open Systems Interconnection -- Internal organization of the Network Layer, May 1999

[7] Chaffey, D., E-business and e-commerce management, *Pearson Education Limited*, ISBN 0 273 68378 0, 2004

[8] Dierks, T. and Allen, C., The TLS Protocol version 1.0, RFC 2246, Internet Engineering Task Force, January 1999