In: Proceedings of the 1^{st} International Conference on Web Information Systems Engineering (WISE'2000), Hong Kong, China, June 2000.

Automatic Generation of Reliable E–Commerce Payment Processes*

Heiko Schuldt Andrei Popovici Hans-Jörg Schek
Institute of Information Systems
Swiss Federal Institute of Technology (ETH)
ETH Zentrum, CH–8092 Zürich, Switzerland
{schuldt,popovici,schek}@inf.ethz.ch

Abstract

The most important phase in E–Commerce interactions is the payment, due to the transfer of sensitive information (e.g., credit card numbers). A couple of requirements exist both from the point of view of a customer and from the merchant's perspective. This set of requirements is even enlarged when complex interactions are considered in which a customer purchases goods originating from different merchants within one single E–Commerce transaction.

In this paper, we show how all these different requirements of payment interactions can be seamlessly integrated in transactional payment processes. These processes are generated automatically based on the customer's specification of the E–Commerce transaction (involved participants, means of payment, etc.). We present the basic structure of such payment processes, how the requirements are mapped into these processes and how they can be generated automatically. Furthermore, we present the architecture of a payment coordinator that has been implemented within the INVENT *project. This payment coordinator controls the execution of transactional payment processes, thereby keeping track of the interactions with the various participants.*

1 Introduction

The accomplishment of payments is the crucial part in business to customer Electronic Commerce (E–Commerce) interactions since this phase does not only determine the success or failure but also requires sensitive information to be transferred. From a general point of view, aside of the basic prerequisite of secure communication, a couple of requirements for correct payment interactions exist, such as different levels of atomicity in the exchange of money and goods between customers and merchants [25, 26]. Furthermore, since fraudulent behavior of participants has to be considered, the ability to legally prove the processing of a payment transaction is required.

Remarkably, E–Commerce is a very interdisciplinary research area. As existing approaches are powered by different communities (i.e., cryptography, networking, etc.), they are very heterogeneous in nature and thus always focus on different special problems. However, it has to be noted that in currently exploited approaches, support for at least one of the above mentioned problems is limited. All requirements are even more important when a customer interacts not only with one merchant but with multiple merchants within one single E–Commerce transaction. In these cases, distribution and heterogeneity additionally have to be considered. Thus, traditional bilateral communication between the participants does no longer provide a feasible solution. The notion of atomicity, for instance, has to be extended when such more complex interactions are supported in that atomicity may be required for the purchase of several goods originating from different, possibly independent and autonomous sources (w.r.t. the exchange of money and all goods). However, all currently existing payment protocols lack support for simultaneous atomic purchases from multiple sources.

To this end, we propose to implement business to customer E–Commerce payments by a transactional process encompassing the diverse interactions between all participants [20]. A stringent requirement of such processes is that they are reliable in that they provide the basic guarantees (e.g., atomicity) for all participants; since our approach concentrates on complex distributed purchases and macro payments, the support of these guarantees is extremely important. A payment coordinator then controls the execution of payment processes and enforces their inherent execution guarantees even in the presence of failures and concurrency. The payment coordinator has to be located at the site of a trustworthy and reliable instance. In the Internet age, however, this does not impose unsolvable restrictions since such institutions are already established, for instance in the form of certification authorities or clearing houses.

* Part of this work has been funded by the Swiss National Science Foundation under the project INVENT (**In**frastructures for **V**irtual **Ent**erprises).

Aside of the reliability of the payment coordinator, the possibility to validate the correctness of each payment process is also required. Therefore, we introduce a mechanism to generate processes at the payment coordinator's site based on a generic payment process template that is filled with the customer's description of her special E–Commerce transaction. Furthermore, we present the payment coordinator developed within the INVENT project encompassing a transactional process management system, a generator for payment processes, and a client application that allows the combination of different interactions into one E–Commerce transaction which is invoked at the payment coordinator.

This paper is structured as follows: In Section 2, we present related work addressing E–Commerce transactions and payment interactions. Then, we provide an overview of the basic requirements imposed in E–Commerce payments (Section 3). After an introduction of the basic concepts of transactional process management (Section 4), the structure of the generic payment process which is used to encompass all steps of payment interactions is presented (Section 5). In Section 6, we discuss the payment coordinator and identify the key aspects in which this payment coordinator exceeds the standard functionality of a transactional process support system. Section 7 concludes the paper.

2 Related Work

Although E–Commerce over the Internet is a rather new but heavily evolving area, a couple of approaches addressing this new form of trade exist. In general, a distinction is made between business to business (b2b) and business to customer (b2c) E–Commerce. While b2b focuses on component based integration of existing services into business processes of (virtual) enterprises (e.g., [2, 8]) and data exchange [1], b2c E–Commerce mainly addresses correctness issues w.r.t. the interactions of the various participants.

In the b2c area, initial work by Doug Tygar provides a first and general overview of requirements of E–Commerce from a customers' perspective [4, 25, 26], mainly addressing atomic interactions but with the restriction that only one customer and one merchant at a time is considered. According to [15], trade interactions between customers and merchants can be classified in three phases: pre-sales, sales and post-sales. The sales phase has a well-defined structure, which is in general not the case for the pre-sales and the post-sales phase. Therefore, processes are a highly appropriate means to implement the interactions that have to be performed for payment purposes. In the context of E–Commerce payments, also several agent-based approaches exist, e.g., [12, 27]. However, all these approaches either lack support for distribution (which is in general present in the negotiation phase but not during payment [27]) or they do not provide appropriate support for transactional execution guarantees in distributed environments [17].

3 Requirements of E–Commerce Payments

Several criteria serve as characteristics of electronic payments. We distinguish between security aspects, technical aspects and economical aspects. Security aspects encompass payment atomicity, provability, anonymity, and cryptographic security. Technical aspects consider the issues of scalability, efficiency, hardware independence, and flexibility. Economical aspects finally focus on the fitness of a payment protocol to penetrate in the E–Commerce market. These are, for instance, transaction costs and the interaction between the client and the payment software (simplicity).

The critical aspects of E–Commerce payments are the security aspects. Therefore, we will discuss these properties in more detail.

3.1 Atomicity

One key requirement in E–Commerce payments is to guarantee atomic interactions between the various participants. As E–Commerce and thus also payments take place in a highly distributed and heterogeneous environment, various aspects of atomicity can be identified.

3.1.1 Money Atomicity

The basic form of atomicity in E–Commerce is associated with the transfer of money from the customer to the merchant. This is denoted by the term *money atomicity* [25, 26]. As no viable E–Commerce payment solution can exist without supporting this atomicity property, multiple solutions have been proposed or are already established. For all account-based protocols (such as FirstVirtual [23], or NetCheque [16]) and for all protocols based on payments by credit card (e.g., SET [22], or CyberCash [6]), money atomicity is guaranteed by the participating institutions of the financial world. For payments based on electronic cash (e.g., eCash^{TM} [7], or MilliCent [14]), money atomicity is tightly coupled with the protocol architecture and design.

3.1.2 Goods Transfer Atomicity

Aside of the money transfer, also the transfer of the merchandise has to be performed atomically. Except for [25], all currently exploited payment protocols do not address *goods transfer atomicity*. However, they do not treat goods transfer atomicity independently from the money transfer. Both aspects are rather captured jointly under the notion of goods atomicity, although they are not related in general.

3.1.3 Distributed Payment Atomicity

In many E–Commerce applications, interaction of customers is not limited to a single merchant. Consider, for instance, a customer who wants to purchase specialized software from a merchant. In order run this software, she also needs an operating system which is only available from a different merchant. As both goods individually are of no value for the customer, she needs the guarantee to perform the purchase transaction with the two different merchants atomically in order to get all products or none. The problem of *distributed purchase atomicity* in general goes along with the fact that different heterogeneous interfaces are involved and different communication protocols are supported by the participating merchants. Although there is a real need to support distributed E–Commerce transactions, this aspect of atomicity is not yet considered by any of the existing payment protocols. Our approach addresses atomicity in distributed payments, yet in combination with money atomicity and goods transfer atomicity.

3.2 Anonymity

A customer wants to remain unknown against the merchant or a bank after an electronic purchase. Protecting the customer and her spending patterns against other participants in the payment is addressed by anonymity. We distinguish between partial anonymity and total anonymity. In the case of partial anonymity, the merchant does not get any personal information about the customer. Total anonymity, in contrast, is characterized by the impossibility of both merchant and payment server to capture any information about the customer. Furthermore, the payment server is not able to map any purchase to any customer identity. Since total anonymity imposes very strict limitations and may even violate legislation, we will only consider partial anonymity.

3.3 Verification and Provability

All participants, customer and merchants, must be able to *prove* that the goods sent (received) are those both parties agreed upon in the initial negotiation phase (certified delivery [25]). This requirement stems from the fact that, in contrast to traditional distributed database transactions where only technical failures have to be addressed, also fraudulent behavior of participants may arise in E–Commerce.

Secondly, all participants have to be supported by appropriate mechanisms to *verify* the properties of a payment protocol with respect to the previously identified characteristics. Such a verification can either positively report that a particular protocol does not allow any of the participants to cheat, or, in the negative case, provides information about which participant may act fraudulent (that is, violate one of the previously mentioned requirements). In existing approaches, verification is not present in this form but is rather hidden in extensive protocol specifications. Aside of provability, verification is a main feature of our approach.

3.4 Cryptographic Security

The transfer of sensitive data, such as electronic cash or credit card numbers, requires cryptographic mechanisms and algorithms in order to avoid third party attacks. Furthermore, appropriate authentication mechanisms have to be provided. Since the Internet is the medium connecting customer and merchants in E–Commerce, a temporarily established fraudulent store-front with a nice web interface cannot be distinguished from reliable merchants. Thus, each customer must be able to validate the identity of the merchants she interacts with (e.g., by signing all messages that are transferred [24]). While all previously identified requirements are inherently part of our process-based E–Commerce payment transactions, cryptographic security is orthogonally provided by the underlying infrastructure.

4 Transactional Process Management

In this section, we introduce the concepts of transactional process management which will be exploited for the definition and the enactment of payment processes.

4.1 Process Model

A process is a partially ordered collection of activities, which correspond to invocations of application services. These activities can be characterized in terms of their termination guarantees: they are either *compensatable, retriable*, or *pivot* [13, 28]. Compensatable activities can be semantically undone after they have committed, pivot activities are those which are not compensatable (when no appropriate compensation is available or when compensation is too expensive and thus has to be avoided), and retriable activities are the ones that are guaranteed to successfully terminate. Furthermore, the added structure of a transactional process is reflected by different orders: a precedence order specifies the regular execution of activities and a preference order indicates alternative executions in case of failures [19].

Based on the different termination properties of activities and the precedence and preference orders, it can be validated whether a single processes is defined correctly. This is the case when only compensatable activities precede the first pivot activity and when an alternative execution is associated with each pivot consisting only of retriable activities [28]. For these processes, all possible failures can be handled by either undoing all activities (when only compensatable activities have committed) or by executing a safe alternative consisting only of retriable activities. Thus, they are called *processes with guaranteed termination*. This inherent correctness property of transactional processes is a generalization of the “all-or-nothing” semantics of traditional

ACID transactions since it ensures that one of eventually many valid executions (specified by alternatives) is effected.

4.2 Process Coordinator

Each activity of a process corresponds to a service invocation in a subsystem. The main components of transactional process management thus consist of a process coordinator controlling the process execution and several transactional coordination agents (TCAs), one for each subsystem [21], adding transactional properties to service invocations.

Starting with the correct specification of single processes having guaranteed termination property, the process coordinator's task is to enforce the correct execution of transactional processes even in the presence of failures and concurrency. The key aspects of the transactional process coordinator can be briefly summarized as follows: it acts as a kind of transaction scheduler that is more general than a traditional database scheduler in that it i.) knows about semantic commutativity of activities, ii.) knows about the termination properties of activities, and iii.) exploits the regular precedence order of processes when executing activities and knows about alternative executions paths in case of failures.

5 Transactional Payment Processes

The payment processes we consider are extensions of anonymous atomic transactions described in detail in [4]. They rely on electronic cash token as means of payment and are primarily designed for the purchase of digital goods. These are transferred prior to the payment in an encrypted way. Due to this encryption, the merchandise is not utilizable until the key transfer –which is an integral part of the payment process– is effected successfully. However, the purchase of "traditional" goods that are shipped by regular mail is also possible. In this case, a legally binding digital contract is subject to the exchange (again, the contract is sent in an encrypted way and the keys required for decryption are transferred within the payment processing).

5.1 Structure of Transactional Payment Processes

The various interactions that have to take place between all participants in (distributed) E–Commerce payments are combined within one single entity, a transactional payment process. Each payment process is invoked by the customer at a central payment coordinator, i.e. the process coordinator specialized to payments. The overall structure of a transactional payment process can be seen in Figure 1. The precedence order is depicted by solid arcs; dotted arcs are used for the preference order.

When a payment process is invoked, the customer first has to specify the payment information PI she is willing to use within this E–Commerce payment transaction (*Receive Payment Information*). The payment information can either be a credit card number or some eCash token. In the next step, the validity of this payment information is checked by invoking an appropriate service at the corresponding bank (*Check Validity of Payment Information*). This check is required in order to ensure that either the balance of the credit card account covers the amount of payment or, in the case of eCash token, that double spending is prevented. After positive validation, all merchants are requested to send the keys needed to decrypt the previously encrypted goods to the payment coordinator (*Receive Key*). When non-digital goods are subject to a purchase (which are in general available with a limited quality-on-hand), by transferring the key, the merchant guarantees the subsequent shipment after the commit of the transactional payment process. After all keys from all merchants have been received, a timeout-check is performed (*Check Timeout*). This activity checks whether all time frames of all merchants are respected (they are in general not willing to hold reservations on their goods for unlimited periods).

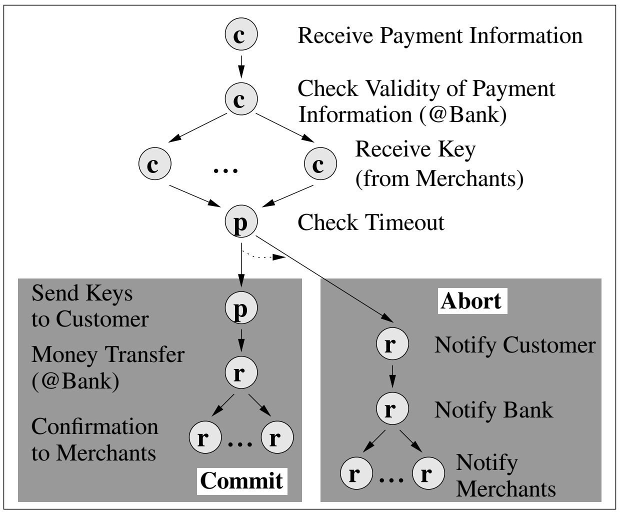

Figure 1. Structure of a Payment Process

If all constraints are satisfied, the payment coordinator determines the success of the payment transaction. To this end, all keys are sent to the customer (*Send Keys to Customer*). These keys are required to decrypt the previously delivered encrypted goods and make them usable for the customer. Subsequently, a money transfer activity is issued in the bank application in order to credit the merchant's accounts (*Money Transfer*), and finally a confirmation on the successful termination of the payment is sent to all merchants (*Send Confirmation to Merchant*). If the timeout check fails, an abort of the payment process is issued. To this end, an alternative execution path is taken which encompasses notifications to all participants about the failure of the process. This alternative execution path has also to be taken when the payment coordinator decides to commit the process but is not able to transfer the keys to the customer.

5.2 Properties of Transactional Payment Processes

Reasoning about the correctness of payment processes has to consider two different aspects: firstly, each process itself has to be correct with respect to the guaranteed termination property. Secondly, correctness also encompasses the compliance of all previously discussed guarantees required for payments (Section 3).

5.2.1 Guaranteed Termination of Payment Processes

In order to verify the inherent property of guaranteed termination required for each transactional payment process, the termination properties of all activities have to be identified. Since the reception of the payment order and the reception of the keys can be trivially undone by deleting these information, all these activities are compensatable. Additionally, the validation of payment information can also be undone (e.g., by wiping out reservations on the eCash token used). The timeout check, however, where the overall outcome of the payment transaction is determined (either commit or abort) has to be treated as pivot which enforces the process to terminate in forward direction. All subsequent steps are guaranteed to succeed (retriable) except for the key transfer to the customer which is pivot. The unavailability of the customer must also lead to an abort of the whole process by issuing the alternative execution labeled with abort. The termination property of each activity (**c**ompensatable, **p**ivot, **r**etriable) is also depicted in Figure 1. Based on the precedence and preference orders as well on the termination properties of each activity, it can be shown that this transactional payment process is correctly defined and thus provides guaranteed termination.

5.2.2 Compliance of Payment Process Requirements

The inherent correctness of single processes has to be extended about the requirements imposed by E–Commerce payments. Therefore, it has to be shown that the desired semantics of payment interactions with respect to all requirements is provided in all correct terminations of processes.

Atomicity The atomicity requirement jointly includes money transfer, goods delivery, and distributed purchase atomicity. It is present when all possible outcomes of a process guarantee that either money and goods are transferred jointly for all purchases that are part of a payment or that neither money nor goods are exchanged for any of them.

Whenever a failure occurs prior to the timeout check, all previous steps are compensated and no information is transferred. Similarly, in the case of an abort, payment information is released and no keys are transferred to the customer. In case of successful termination, the process structure guarantees that real money flow is performed only in the case where all keys have been sent to the customer. Therefore, money transfer is initiated jointly for all purchases; the basic requirement of enforcing money atomicity must also be provided by all participating financial institutions (as it is the case in traditional payment interactions).

Goods transfer atomicity is met by the combination of the transfer of encrypted goods and the key transfer which is performed together with the money transfer. Since the transfer of keys and money is performed via the payment coordinator, no party is able to cheat (which could be the case, for instance, when the customer sends payment information to a malicious merchant who then refuses to ship the ordered goods). Instead of keys, the payment coordinator could also support the transfer of the merchandise itself; however, this would not only increase the communication overhead but would also decrease the degree of anonymity (note that the merchandise is in general not known to the payment coordinator).

Distributed purchase atomicity, finally, is present for two reasons. Firstly, the separation of initial negotiation and subsequent payment processing provides the basis for this kind of atomicity. When the initial step would already consider legally binding orders (these orders would have to be performed serially when several merchants are considered), then, when some merchandise would not be available, the all-or-nothing semantics of distributed purchase atomicity would be violated. Secondly, the combination of key and money transfer is enforced for each purchase of a distributed payment within the payment process.

Anonymity Since the identity of a customer (e.g., the IP address of the host she is using) may not be revealed to the merchant by applying anonymizing techniques (e.g., [3]), anonymity is given – at least in the partial form. In order to hide the identity of the customer to the bank when issuing electronic cash token, cryptographic blinding techniques [5] can be applied. Total anonymity of a customer against the payment coordinator is not possible since the latter one needs to contact the customer for the key transfer.

Verification and Provability Due to the combination of all interactions within one centrally controlled process, its correctness can be verified by all participants. This would not be the case in an execution of a payment where parts of the protocol are distributed among all participants (e.g., in agent-based approaches). A basic prerequisite is, however, that all participants trust in the reliability of this payment coordinator which is supposed to reside at the site of a clearing house or certification authority, respectively.

Since the process coordinator stores all process information persistently, the participation of a customer in a transaction and the service ordered in this transactions can a posteriori be proven (this corresponds to certified delivery described in [4]). Apparently, a conflict between anonymity and provability occurs here. Generally speaking, there is no reason for the customer to provide real identity information in a payment transaction as long as it completes successfully. If the customer rejects the outcome of the transaction, she has to give up her mask to prove her correct interactions. Therefore, anonymity can no longer be protected in case a payment transaction fails.

Cryptographic Security This aspect is not reflected in the payment process but is provided by the underlying process support infrastructure (i.e., the payment coordinator).

Handling of System Failures Reliability does not only include the correct specification of payment processes but also encompasses high availability of the process coordinator. To this end, mechanisms have to be considered to increase availability, e.g., by replication techniques applied to process management systems [10].

System failures at the merchant's site, however, cannot be covered. In these cases, the customer is only able to prove the merchant's participation in a distributed E–Commerce transaction and also –based on the persistent log managed by the payment coordinator– to prove the merchant's violation of the initial agreement which allows her to take legal steps.

6 INVENT: A Process-Based Payment Coordinator

In this section, we present the architecture of our prototype system, the INVENT payment coordinator and describe the main components that are needed for the generation and execution of transactional payment processes.

6.1 Architecture

A feasible infrastructure for process based E–Commerce payments must be able to integrate arbitrary applications established by merchants and banks, respectively, and must also minimize the requirements imposed on customers to participate. The goal of the INVENT payment coordinator is to make existing banking applications or merchant store fronts available by small adapters (coordination agents) and to provide a client side application to define and initiate distributed E–Commerce transactions. The main components of the whole payment system together with the interactions between the various participants are depicted in Figure 2. All parts belonging to the payment coordinator are colored in dark grey while the remaining components provided by the other participants are depicted in light grey.

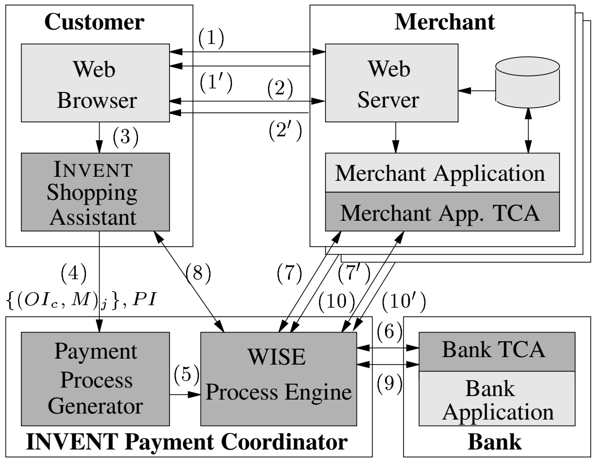

Figure 2. INVENT Payment System

6.2 Execution of Distributed Payment Transactions

The execution of distributed E–Commerce transactions consists of two phases: the first phase is based on bilateral negotiation between a customer and multiple merchants which are offering their services, electronic goods, or digital contracts (in the case of non-digital goods) through their electronic store fronts (steps 1 and $1'$ in Figure 2). In this initial phase, also encrypted goods are transferred (steps 2 and $2'$). For the generation of cryptographic keys, arbitrary merchant applications can be exploited. A realistic assumption is that these keys are stored in a database on the merchant's site. After the customer has received all encrypted goods, she can decide which ones to encompass into one E–Commerce transaction using a special client tool, the *Shopping Assistant* (step 3).

The second phase is the actual payment coordinated by the INVENT payment coordinator. For this, the customer sends a description of the E–Commerce transaction to the INVENT payment coordinator (step 4). Based on this information, a payment process is generated by the *Payment Process Generator (PPG)* which is then loaded into the *Process Engine* (step 5). This process follows the structure of a transactional payment process as presented in Section 5.1. All following steps (6 to $10'$) then correspond to the activities of this payment process executed by the core process engine.

6.3 Payment Coordinator

In what follows, we discuss all components of the payment coordinator, that is, the Shopping Assistant, the Payment Process Generator, and the Process Engine in detail.

6.3.1 Shopping Assistant

All payment information captured during the initial negotiation phase is collected in the Shopping Assistant. The Assistant runs at the customer's site and is plugged into her web browser. Data is exchanged by special MIME types that are tailored to the kind of information gathered in the negotiation phase. Figure 3 shows a screen shot of the Shopping Assistant during a distributed purchase encompassing two merchants. The Shopping Assistant also contains a wallet to manage electronic cash tokens and stores –after receipt– the keys needed for decryption.

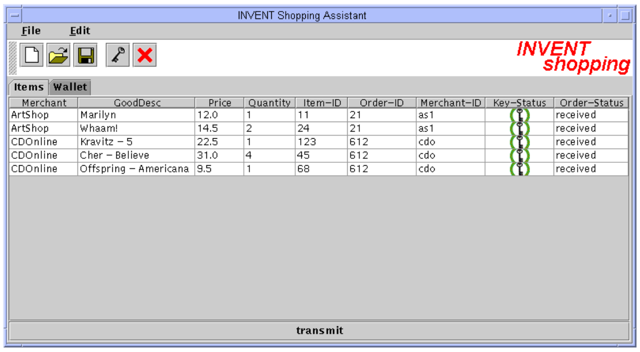

Figure 3. INVENT Shopping Assistant

In the traditional case where a customer interacts only with one merchant, she has to send all payment information directly to this merchant who then issues a verification and a subsequent real money transfer. In the case of distributed E–Commerce transactions encompassing multiple merchants, the customer is responsible to initiate the subsequent steps. This difference is, however, not of fundamental nature since the transfer of information is performed by the Shopping Assistant and is hidden to the customer.

6.3.2 Payment Process Generator (PPG)

A crucial aspect in the execution of E–Commerce payment process is the generation of payment processes which have to reflect the outcome of the initial negotiation phase (e.g., number and addresses of merchants, information about all goods, etc.). Since these processes have to be reliable, they have to be generated by the payment coordinator, the only instance trusted by all participants. Each process has to contain the information gathered by the customer during the negotiation phase. Therefore, she has to provide a typed representation of her E–Commerce transaction which can then be translated into a process. The transformation task is handled by the *Process Payment Generator* which is plugged into the process engine.

The information that is sent to the PPG via the Shopping Assistant. It consists of the customer's view of the order information OI_c and a merchant information M for each good to be encompassed in a payment transaction as well as the payment information PI. The PPG then creates a process description encompassing the appropriate number of *Receive Key*, *Send Confirmation*, and *Notify Merchant* activities which can at run-time then be instantiated with the parameters (merchant addresses, order information) given by the customer. Moreover, all activities interacting with the bank have to be parameterized according to the payment information specified by the customer.

After the generation phase, the process description is sent to the process engine for execution. The generation, however, only affects the build-time of a payment process; once a payment process is instantiated, no dynamic structural changes (e.g., [18]) are applied.

6.3.3 Process Engine

The execution of transactional payment processes must provide certain execution guarantees. Even in the presence of failures and concurrent access of processes to shared resources, correct process execution must be guaranteed. The fault-tolerant execution is required in order to enforce all inherent guarantees (atomicity, etc.) of single processes. But also concurrency control at process level has to be provided. Consider, for instance, two merchants A and B offering non-digital goods a and b, respectively, of limited availability and assume further that the quantity on hand (qoh) of both goods is 1. When two concurrent distributed E–Commerce transactions are executed by the payment coordinator and each of these transactions aims in buying both a and b, then exactly one process should succeed while the other one would have to fail. However, when concurrency control would not be respected by the payment coordinator, one process may first successfully issue a key request from merchant A (this key is required to decrypt the previously shipped digital contract and goes along with a legally binding agreement of A to subsequently ship the requested good) while the other process successfully performs a key request from B. In this case, both processes would wrongly fail since at least one merchandise would not be available.

These properties are provided by the WISE process engine (Workflow based Internet SErvices), a process support system which has been developed at ETH [2, 9]. This process engine implements appropriate protocols to support the correct parallel and fault-tolerant execution of transactional processes [19]. The integration of all external components is performed by so-called *Transactional Coordination Agents (TCAs)* [21] acting as advanced wrappers and allow the invocation of services in the merchant and bank applications. By exploiting cryptographic libraries [11] within the WISE system, communication between the process engine and all applications is made secure.

7 Conclusion

This paper provides a detailed analysis of requirements participants impose in distributed E–Commerce payments. Using the notion of transactional processes, it has been shown that all payment interactions can be embedded into a single payment process inherently providing aside of provability and partial anonymity also all aspects of atomicity guarantees. With the payment coordinator implemented within the INVENT project, we have presented a system allowing a customer to i.) encompass different independent interactions with different merchants into one single E–Commerce transaction, ii.) dynamically generate a process description reflecting the outcome of the initial negotiation with multiple merchants, and iii.) invoke this transactional payment process at the payment coordinator which controls the execution and enforces correct termination of the payment process. Furthermore, when orchestrating E–Commerce payment processes by a centralized payment coordinator, the monitoring of the state of a payment interaction is facilitated compared with the distribution and complexity found in current payment protocols.

Based on this initial work, we will extend and generalize the idea of dynamically generating payment processes by allowing to use different payment methods for different goods within one single process (e.g., payment by credit cards, transfer of account information, etc.). To this end, building blocks for each payment method have to be identified. Additionally, since the usage of different payment methods for different parts of a distributed payment transaction may have influences on the execution guarantees that can be provided, an additional component will be implemented that analyzes the degree of possible execution guarantees and presents them to the customer prior to the instantiation of a payment process.

References

[1] S. Abiteboul, B. Amann, S. Cluet, A. Eyal, L. Mignet, and T. Milo. Active Views for Electronic Commerce. In *Proc. of the 25th Int'l Conf. on Very Large Databases (VLDB'99)*, pages 138–149, Edinburgh, Scotland, Sept. 1999.

[2] G. Alonso, U. Fiedler, C. Hagen, A. Lazcano, H. Schuldt, and N. Weiler. WISE: Business to Business E-Commerce. In *Proc. of the 9th Int'l Workshop on Research Issues in Data Engineering (RIDE-VE'99)*, pages 132–139, Mar. 1999.

[3] Anonymizer.com, `http://www.anonymizer.com`.

[4] J. Camp, M. Harkavy, D. Tygar, and B. Yee. Anonymous Atomic Transactions. In *Proc. of the 2nd Usenix Workshop on Electronic Commerce*, pages 123–133, Nov. 1996.

[5] D. Chaum, A. Fiat, and M. Naor. Untraceable Electronic Cash. In *Proc. of Advances in Cryptography (CRYPTO'88)*, pages 319–327. Springer, 1988.

[6] CyberCash, `http://www.cybercash.com`.

[7] eCash, `http://www.digicash.com/`.

[8] A. Geppert, M. Kradolfer, and D. Tombros. Market-Based Workflow Management. *Int'l Journal on Cooperative Information Systems (IJCIS)*, 7(4):297–314, Dec. 1998.

[9] C. Hagen. *A Generic Kernel for Reliable Process Support*. PhD thesis, ETH Zürich, 1999.

[10] C. Hagen and G. Alonso. Highly Available Process Support Systems: Implementing Backup Mechanisms. In *Proc. of the 18th Symposium on Reliable Distributed Systems (SRDS'99)*, pages 112–121, Oct. 1999.

[11] IAIK. Crypto-Toolkit. Technical University of Graz, Austria. `http://jcewww.iaik.tu-graz.ac.at/`.

[12] L. Kerschberg and S. Banerjee. An Agency-Based Framework for Electronic Business. In *Proc. f the Int'l Workshop on Cooperative Information Agents*, July 1999.

[13] S. Mehrotra, R. Rastogi, A. Silberschatz, and H. Korth. A Transaction Model for Multidatabase Systems. In *Proc. of the 12th Int'l Conf. on Distributed Computing Systems (ICDCS'92)*, pages 56–63, Yokohama, Japan, June 1992.

[14] MilliCent, `http://www.millicent.digital.com`.

[15] P. Muth, J. Weissenfels, and G. Weikum. What Workflow Technology can do for Electronic Commerce. In *Proc. of the EURO-MED NET Conference*, Mar. 1998.

[16] C. Neumann and G. Medvinsky. Requirements for Network Payment: The NetCheque^{TM} Perspective. In *Proc. of the 40th IEEE Computer Society Int'l Conf.*, pages 32–36, 1995.

[17] M. Papazoglou. The Role of Agent Technology in Business to Business Electronic Commerce. In *Proc. CIA'99*, 1999.

[18] M. Reichert and P. Dadam. ADEPT$_{\mathrm{flex}}$ — Supporting Dynamic Changes of Workflows Without Losing Control. *Journal of Intelligent Information Systems*, 10(2):93–129, 1998.

[19] H. Schuldt, G. Alonso, and H.-J. Schek. Concurrency Control and Recovery in Transactional Process Management. In *Proc. of the 18th ACM Symposium on Principles of Database Systems (PODS'99)*, pages 316–326, 1999.

[20] H. Schuldt, A. Popovici, and H.-J. Schek. Execution Guarantees in Electronic Commerce Payments. In *Proc. TDD'99*, Springer LNCS 1773, pages 193-202, Sept. 1999.

[21] H. Schuldt, H.-J. Schek, and G. Alonso. Transactional Coordination Agents for Composite Systems. In *Proc. of the 3rd Int'l Database Engineering and Applications Symposium (IDEAS'99)*, pages 321–331, Aug. 1999.

[22] SET, `http://www.setco.org`.

[23] M. Stolpmann. *Electronic Cash in the Internet – Basics, Concepts, Perspectives*. O'Reilly, 1997. In German.

[24] L. Tang. Verifiable Transaction Atomicity for Electronic Payment Protocols. In *Proc. of the 16th Int'l Conf. on Distributed Computing Systems*, pages 261–269, May 1996.

[25] D. Tygar. Atomicity in Electronic Commerce. In *Proc. of the 15th Annual ACM Symposium on Principles of Distributed Computing*, pages 8–26, May 1996.

[26] D. Tygar. Atomicity versus Anonymity: Distributed Transactions for Electronic Commerce. In *Proc. of the 24th Conf. on Very Large Databases (VLDB'98)*, Aug. 1998.

[27] X. Wang, X. Yi, K. Lam, C. Zhang, and E. Okamoto. Secure Agent-Mediated Auctionlike Negotiation Protocol for Internet Retail Commerce. In *Proc. CIA'99*, July 1999.

[28] A. Zhang, M. Nodine, B. Bhargava, and O. Bukhres. Ensuring Relaxed Atomicity for Flexible Transactions in Multidatabase Systems. In *Proc. ACM SIGMOD*, pages 67–78, May 1994.