Give me all I pay for — Execution Guarantees in Electronic Commerce Payment Processes

Heiko Schuldt Andrei Popovici Hans-Jörg Schek

Database Research Group
Institute of Information Systems
ETH Zentrum, 8092 Zürich, Switzerland
Email: {schuldt,popovici,schek}@inf.ethz.ch

Abstract

Electronic Commerce over the Internet is one of the most rapidly growing areas in todays business. However, considering the most important phase of Electronic Commerce, the *payment*, it has to be noted that in most currently exploited approaches, support for at least one of the participants is limited. From a general point of view, a couple of requirements for correct payment interactions exist, namely different levels of atomicity in the exchange of money and goods of a single customer with different merchants. In this paper, we identify the different requirements participants demand on Electronic Commerce payments from the point of view of execution guarantees and present how payment interactions can be implemented by transactional processes. Finally, we show how these execution guarantees can be provided for payment processes in a natural way by applying the ideas of transactional process management to an Electronic Commerce *Payment Coordinator*.

1 Introduction

Along with the enormous proliferation of the Internet, Electronic Commerce (E-Commerce) is continuously gaining importance. The spectrum of applications that are subsumed under the term E-Commerce leads from rather simple orders performed by Email to the purchase of shopping baskets consisting of several goods originating from different merchants by spending electronic cash tokens.

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. From the point of view of the database community, atomicity properties have been identified as one key requirement for payment protocols in E-Commerce [Tyg96, Tyg98]. The more complex interactions with consumers and merchants become, the more dimensions of atomicity have to be addressed. In the simplest case, only money has to be transferred atomically from the consumer to the merchant. However, considering complex shopping baskets filled with (electronic) goods from several merchants, atomicity may also be required for the purchase of all these goods originating from different possibly independent and autonomous sources, along with the atomic exchange of money and all goods.

Due to their distributed nature, protocols that have been suggested to support payment atomicity in E-Commerce impose high requirements on the participating instances (e.g., NetBill [CTS95]). However, with a centralized payment coordinator, the complex interactions of the various participants can be embedded within a payment process, thus reducing the prerequisites for merchants and customers to participate in E-Commerce. Transactional process management [SAS99] can then be exploited in order to provide the necessary execution guarantees for transactional E-Commerce payment processes in a natural way.

This paper is structured as follows: In Section 2, we provide a general framework for E-Commerce payment interactions. Based on this framework, we analyze the different atomicity requirements for E–Commerce payment (Section 3). Then, in Section 4, we summarize transactional process management and present the structure of a transactional payment process allowing the required execution guarantees to be provided by a Payment Coordinator. Section 5 finally concludes the paper.

2 Schema for Payment Protocols in E-Commerce

The description of sales interactions in non-electronic markets [Sch98] encompasses three phases: information, negotiation, and payment. During the information phase, a customer evaluates and compares the offers of several merchants. After selecting the best offer, she negotiates with the chosen merchant the conditions for the deal (negotiation). If they reach an agreement, the last step (the payment) involves the money transfer from customer to merchant and the service (the merchant fulfills his contract).

Most electronic payment systems focus only on the money transfer of the last phase. Our view of an *electronic payment scheme* also considers the systems and protocols for accomplishing both the money transfer and the service.

2.1 Participants

An electronic payment scheme involves participants originating from two distinct worlds: on the Internet side there are the customer, the merchant and a third entity, the payment server which coordinates the two. The other side is represented by the financial world with its proprietary network infrastructure and protocols. The participants are financial institutes and again the payment server, that has to consistently transform the data flow on the Internet side in corresponding “real world” money flow. The participants are depicted in Figure 1.

2.2 Steps of an E-Commerce Transaction

Prior to the payment transaction, the participants are involved in an *initialization* phase, depicted in Figure 1 by dashed arrows. Both customer and merchant have to establish accounts within the financial institutes “issuer” (or “acquirer”, resp.). The transformation of electronic money into real money is performed using these accounts. Also in this phase the customer receives from his bank a *customer secret* which enables him to perform electronic payments. The customer secret is visible only for the customer herself, for the issuing bank and (eventually) for the payment server. The most common form of the customer secret is a credit card number, in electronic cash schemes (such as eCash^{TM} [Dig99]), the customer secret is an E-cash token. Because account operations are rather less often than payments, we can consider them as part of the initialization phase.

Almost all the payment schemes contain the five following steps, marked in Figure 1:

- Negotiation (1): the customer selects the desired service or merchandise she wants from the merchant, and negotiates with the merchant the price of the service. The result of this step is the *Order Information*. The Order Information is a protocol of the negotiation phase, including service (merchandise) and price specification.
- Payment order (2): the customer sends Payment Information (PI) and Order Information ($\mathrm{OI_c}$) to the merchant. The $\mathrm{OI_c}$ is the customer's view of the agreement with the merchant.
- Payment authorization (3): the merchant forwards PI, $\mathrm{OI_c}$, $\mathrm{OI_m}$ and additional data to the payment server. $\mathrm{OI_m}$ is the merchant's view of the agreement with the customer.

  The payment server directly or indirectly verifies the validity of the payment information, the consistency of the payment using $\mathrm{OI_c}$ and $\mathrm{OI_m}$. It eventually triggers the real world money transfer using its role on the non-Internet side.

  At the end of the payment authorization, the merchant receives a confirmation message C from the payment server (4).
- Purchase response (5): The merchant sends himself a confirmation to the customer. In case of electronic (non-tangible) goods, the purchase response can be immediately followed by the merchandise or the service itself.

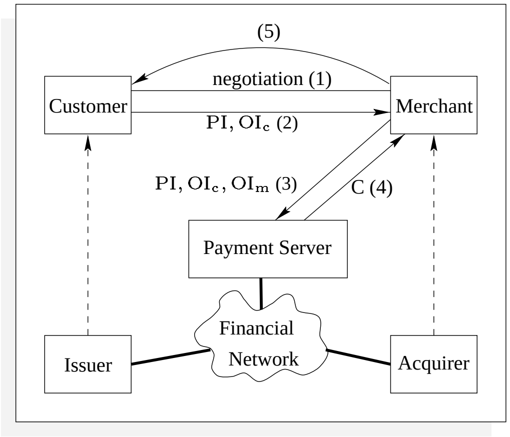

Figure 1: Generic payment steps

In most existent payment protocols, the payment server is invoked by the merchant. This is no intrinsic restriction, and communication between customer and payment server is also possible.

2.3 Characteristics of Payment Protocols

Several criteria serve as classification models of electronic payment schemes. Starting from the moment of transformation of real money into electronic money, payment protocols can be split in *pre-paid systems* and *pay-by-instruction* ones. *Atomicity* is another item, which will be discussed in detail later. Some protocols introduce the notion of *provability*, which is the ability of each party to prove their correct interactions. *Anonymity* is especially addressed by cash-based-systems. There are also implementation issues like *scalability*, *flexibility*, *efficiency*, *ease of use* and *off-line operation*, which are also important because of the large number of users expected.

3 Atomicity in Electronic Commerce

One key requirement in E-Commerce is to guarantee atomic interactions between the various participants in E-Commerce payment. As E–Commerce and thus also payment takes place in a highly distributed and heterogeneous environment, various aspects of atomicity can be identified: aside of money and goods atomicity [Tyg96, Tyg98], also the atomic interaction of a customer with multiple merchants is needed. In what follows, we analyze and classify these different atomicity requirements in detail.

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* [Tyg96]. As no viable E-Commerce payment solution can exist without supporting this atomicity property, multiple solutions have been proposed or are already established [MV96, Dig99]. However, the atomicity property is tightly coupled with the protocol architecture and design.

Certified Atomic Delivery Aside of money, also goods have to be transferred. Therefore, a further requirement is that the delivery takes place atomically. This can even be reinforced in that both associated parties –customer and merchant– require the necessary information in order to prove that the goods sent (or received, resp.) are the ones both parties agreed to in the initial negotiation phase (*certified atomic delivery*, encompassing the goods atomicity and the certified delivery described in [Tyg96]). This strengthened requirement results from the fact that –in contrast to traditional distributed database transactions where only technical failures have to be addressed– in E–Commerce also fraudulent behavior of participants has to be coped with. Especially when dealing with goods that can be transferred electronically, the combination of money atomicity and certified delivery is an important issue. In [CHTY96], this is realized by a customized Two-Phase-Commit protocol [GR93].

Distributed Purchase 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, however, 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 both products or none. *Distributed purchase atomicity* addresses the encompassment of interactions with different independent merchants into one single transaction.

Most currently deployed payment coordinators support only money atomicity while some advanced systems address also distributed purchase atomicity. However, all three dimensions are –to our best knowledge– not provided by existing systems and protocols although the highest level of guarantees would be supported and although this is required by a set of real-world applications.

This lack of support for full atomicity in E-Commerce payment is addressed by our current research activities where we apply transactional process management (section 4) to realize an E-Commerce Payment Coordinator.

4 Transactional Processes for E-Commerce Payments

In this section, we introduce the theory of transactional process management that provides a criterion for the correct execution of processes with respect to recovery (when failures of single processes have to be considered) and concurrency control (when multiple parallel processes access shared resources simultaneously) and we point out how this theory can be applied for payments in E–Commerce.

4.1 Transactional Process Management

In conventional databases, concurrency control and recovery are well understood problems. Unfortunately, this is not the case when transactions are grouped into entities with higher level semantics, such as *transactional processes*. Although concurrent processes may access shared resources simultaneously, consistency has to be guaranteed for these executions.

Transactional process management [SAS99] has to enforce consistency for concurrent executions and, at the same time, to cope with the added structure found in processes. In particular, and unlike in traditional transactions, processes introduce flow of control as one of the basic semantic elements. Thus, it has to be taken into consideration that processes already impose ordering constraints among their different operations and among their alternative executions. Similarly, processes integrate invocations to applications with different atomicity properties (e.g., activities may or may not be semantically compensatable).

The main components of transactional process management consist of a coordinator acting as top level scheduler and several transactional coordination agents [SSA99] —one for each subsystem participating in transactional processes— acting as lower level schedulers. Processes encompass *activities* which are invocations in subsystems scheduled by the coordinator. The coordinator's task is to execute transactional processes correctly with respect to concurrency control and recovery. Firstly, the execution guarantees to be provided include guaranteed termination, a more general notion of atomicity than the standard all or nothing semantics which is realized by partial compensation and alternative executions. Secondly, the correct parallelization of concurrent processes is required and thirdly, by applying the ideas of the composite systems theory [ABFS97], a high degree of parallelism for concurrent processes is to be provided.

The key aspects of transactional process management can be briefly summarized as follows: The coordinator acts as a kind of transaction scheduler that is more general than a traditional database scheduler in that it i.) knows about properties of activities (compensatable, retriable, or pivot, taken from the flex transaction model [MRSK93, ZNBB94]), ii.) knows about alternative executions paths in case of failures, and iii.) knows about semantic commutativity of activities.

Based on this information, the coordinator ensures global correctness but only under the assumption that the activities within the processes to be scheduled themselves provide transactional functionality (such as, for instance, atomicity, compensatability, order-preservation, etc.).

4.2 Transactional Payment Processes

According to [MWW98], trade interactions between customers and merchants can be classified in three phases: pre-sales, sales and post-sales. While the sales phase has a well-defined structure (especially the payment processing, see section 2), this is in general not the case for the pre-sales and the post-sales phase. Due to this well-defined structure, processes are a highly appropriate means to implement the interactions that have to be performed for payment purposes. Furthermore, all atomicity requirements for payments in E–Commerce can be realized in an elegant way by applying the ideas of transactional process management in an *E-Commerce Payment Coordinator*.

These processes are extensions of anonymous atomic transactions described in [CHTY96], they rely on electronic cash token as means of payment, and are primarily designed for the purchase of electronically available goods that are transferred in an encrypted way to the customer prior to the payment. Furthermore, the idea of transactional payment processes is to encompass all interactions between the participants (customer, merchants and bank). To this end, and in contrast to the currently applied payment schemes, the payment has to be initiated by the customer by invoking a payment process at the Payment Coordinator[1]. The structure of a transactional payment process can be seen in figure 2. The precedence orders are depicted by solid arcs while for the preference order, dotted arcs are used. For each activity, the associated termination property (c̲ompensatable, p̲ivot, r̲etriable) is also given.

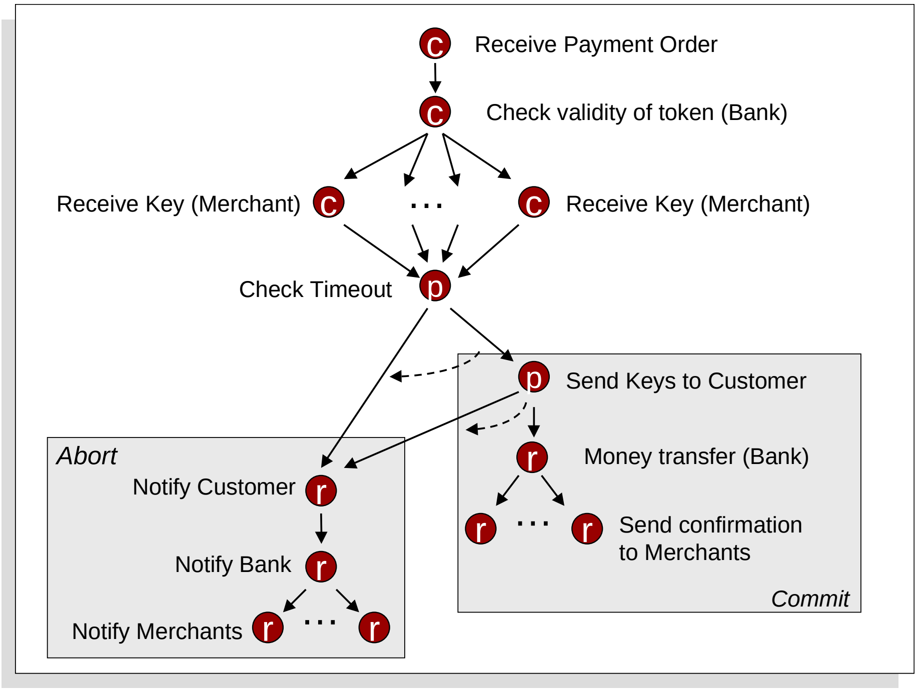

Figure 2: Structure of Payment Process

When a payment process is invoked, the customer first has to specify the payment information PI and all n bilaterally agreed order information (and thus also all different merchants) that have to be encompassed within one single payment transaction. Therefore, a tuple $(OI_c, M)_j$ with order information OI_c and merchant identifier M for each product j with $1 \leq j \leq n$ has to be sent to the Payment Coordinator

[1] Like in the traditional case, the customer has in the initial negotiation phase to agree upon the way the payment is processed with all merchants.

(*receive payment order*). Then, the value and validity of the payment information PI is checked (*check validity of token*). Given the validity of the payment information, the Payment Coordinator contacts all merchants, asks them to validate the order information $(OI_c)_j$ and in the case of successful validation, collects for each product j the key needed for decryption (*receive keys*). When all keys arrive within a given period of time (*check timeout*)[2], the Payment Coordinator sends all keys to the customer, sends a money transfer order to the bank in order to credit the merchant's accounts, and sends a confirmation about the successful termination of the payment to all merchants (*commit* of payment). Otherwise —when the customers view on the order information $(OI_c)_j$ and the merchants view $(OI_m)_j$ do not match for some j, when some keys are not available, when the timeout is exceeded, or when the validation of the payment information PI fails— no exchange will take place (*abort* of payment) but appropriate notifications are sent to all participants.

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. Furthermore, it has to be shown that by all correct terminations, the desired semantics of atomic payment interactions (with respect to all three dimensions of atomicity) is provided. To this end, all possible executions have to be considered. Whenever some failure occurs prior to the termination of the *check timeout* activity, all previously executed steps are semantically compensated by sending a notification about the failure of the payment process to all participants (since this notification is also sent to the customer, she does not lose her payment information but can spent it later within other payments). After the successful transfer of the keys to the customer, the payment process is also terminated correctly since the real-world money transfer has previously been ensured by the bank (in the *check validity of token* step). Finally, when the transfer of keys to the customer fails (e.g., since she cannot be contacted), also appropriate notifications are sent to all participants and no real-world money transfer takes place (again, the payment information can be used by the customer for further payments).

This transactional payment process now provides money atomicity, certified atomic delivery and distributed purchase atomicity simultaneously. Since it is guaranteed that the payment information is only transferred in real-world money flow when the process terminates correctly and since no merchant receives this payment information directly, the customer is able to spent it again in the abort case of a payment process without being accused of double-spending. For certified atomic delivery, the same arguments as given in [CHTY96] hold: the Payment Coordinator persistently stores process information and is thus in the case of customer complaints able to verify whether the order information matches the goods delivered. Finally, since the process only terminates correctly when all merchants agree to commit, distributed purchase atomicity is also provided.

Aside of atomicity, also anonymity of the customer and provability have been identified as security aspects of payment protocols. Transactional payment processes do not provide total anonymity (since the Payment Coordinator needs to contact the customer in order to transfer the keys needed to decrypt all goods), but at least they provide partial anonymity. The customer may hide her identity (e.g., the IP address of the host she is using) to the merchants by applying anonymizing techniques (such as, for instance, [Ano99]). In order to hide the identity of the customer to the bank when issuing electronic cash token, cryptographic blinding techniques [CFN88] can be applied. Since the Payment Coordinator stores all process information (including the order information) persistently, the proof of the participation of a customer in a transaction and the service ordered in this transactions is possible (total provability).

By executing payment processes by a centralized Payment Coordinator, the monitoring of the state of a payment interaction is facilitated compared to the distribution found in current payment protocols. However, all participants (and especially the customer) have to trust this centralized Payment Coordinator. But since in the case of these payment processes only information about the merchants involved in a deal and the prizes of goods is available to the Payment Coordinator but no information about the single goods, this is equivalent to the amount and kind of data credit card organizations collect when customers perform payments with their credit cards.

[2] This activity only generates a log entry making the decision persistent; although it can technically be compensated, it is treated as pivot since compensation of the process is no longer allowed.

5 Conclusion

This paper provides a detailed analysis of requirements participants in E-Commerce payment impose with respect to atomicity issues. Different levels of atomicity can be identified which, however, are not simultaneously provided by existing approaches. Using the notion of processes, it has been shown that all payment interactions can be embedded into a single payment process where all possible levels of execution guarantees can be provided while at the same time the prerequisites of the participants are reduced. Finally, by applying the ideas of transactional process management, it has been shown how a Payment Coordinator supporting atomic and provable payment processes can be developed.

This process-based Payment Coordinator is currently being implemented within the WISE system [AFH+99]. Based on this implementation, we will in our future work extend the analysis of payment processes to further properties (such as, for instance, anonymity, scalability, or flexibility). Our goal is to decouple these properties, to identify the building blocks needed to realize them and to flexibly generate payment processes with user-defined properties by plugging together the building blocks needed.

References

[ABFS97] G. Alonso, S. Blott, A. Feßler, and H.-J. Schek. Correctness and Parallelism in Composite Systems. In *Proceedings of the ACM Symposium on Principles of Database Systems (PODS'97)*, Tucson, Arizona, May 12-15 1997.

[AFH+99] G. Alonso, U. Fiedler, C. Hagen, A. Lazcano, H. Schuldt, and N. Weiler. WISE: Business to Business E-Commerce. In *Proceedings of the 9^{th} International Workshop on Research Issues in Data Engineering. Information Technology for Virtual Enterprises (RIDE-VE'99)*, pages 132–139, Sydney, Australia, March 1999.

[Ano99] Anonymizer.com, 1999. `http://www.anonymizer.com`.

[CFN88] D. Chaum, A. Fiat, and M. Naor. Untraceable Electronic Cash. In *Proceedings of Advances in Cryptography (CRYPTO'88)*, pages 319–327. Springer, 1988.

[CHTY96] J. Camp, M. Harkavy, D. Tygar, and B. Yee. Anonymous Atomic Transactions. In *Proceedings of the 2^{nd} Usenix Workshop on Electronic Commerce*, pages 123–133, November 1996.

[CTS95] B. Cox, D. Tygar, and M. Sirbu. NetBill Security and Transaction Protocol. In *Proceedings of the 1^{st} USENIX Workshop on Electronic Commerce*, pages 77–88, July 1995.

[Dig99] DigiCash, 1999. `http://www.digicash.com/`.

[GR93] J. Gray and A. Reuter. *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann, 1993.

[HS98] A. Hermanns and M. Sauter, editors. *Management–Handbuch Electronic Commerce*. Vahlen, 1998. In German.

[MRSK93] S. Mehrotra, R. Rastogi, A. Silberschatz, and H. Korth. A Transaction Model for Multidatabase Systems. *Bulletin of the Technical Committee on Data Engineering*, 16(2), June 1993.

[MV96] MasterCard and Visa. *Secure Electronic Transaction Specification*. MasterCard and Visa, draft edition, June 1996. Book 1: Business Description, Book 2: Programmer's Guide, Book 3: Formal Protocol Specification (Slightly revised version of Book 3 appeared August 1, 1997).

[MWW98] P. Muth, J. Weissenfels, and G. Weikum. What Workflow Technology can do for Electronic Commerce. In *Proceedings of the EURO-MED NET Conference*, 1998.

[SAS99] H. Schuldt, G. Alonso, and H.-J. Schek. Concurrency Control and Recovery in Transactional Process Management. In *Proceedings of the ACM Symposium on Principles of Database Systems (PODS'99)*, pages 316–326, Philadelphia, Pennsylvania, USA, May 31-June 2 1999.

[Sch98] B. Schmidt. *Elektronische Märkte – Merkmale, Organisation und Potentiale*. In: [HS98]. 1998. In German.

[SSA99] H. Schuldt, H.-J. Schek, and G. Alonso. Transactional Coordination Agents for Composite Systems. In *Proceedings of the 3rd International Database Engineering and Applications Symposium (IDEAS'99)*, pages 321–331, Montréal, Canada, August 1999.

[Tyg96] D. Tygar. Atomicity in Electronic Commerce. In *Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing*, pages 8–26, May 1996.

[Tyg98] D. Tygar. Atomicity versus Anonymity: Distributed Transactions for Electronic Commerce. In *Proceedings of the 24th Conference on Very Large Databases (VLDB'98)*, New York, USA, August 1998.

[ZNBB94] A. Zhang, M. Nodine, B. Bhargava, and O. Bukhres. Ensuring Relaxed Atomicity for Flexible Transactions in Multidatabase Systems. In *Proceedings of the ACM SIGMOD Conference*, pages 67–78, 1994.