A Language for Modelling Secure Business Transactions

Alexander W. Röhm, Gaby Herrmann, Günther Pernul
Department of Information Systems
University of Essen, Germany

{roehm | herrmann | pernul}@wi-inf.uni-essen.de

Abstract

Among other areas electronic commerce includes the fields of electronic markets and workflow management. Workflow management systems are usually used to specify and manage inter- and intra-organisational business processes. Although workflow management techniques are capable to specify and conduct at least parts of market transactions, these techniques are not or very rarely used for this purpose yet. In both fields users demand security and integrity to protect for example their privacy, their property rights or digital payments. To satisfy these security demands a variety of existing security services, mechanisms, protocols, and organisational measures are existent and may be used. At one hand side, to encourage using these techniques it is necessary to have a tool which enables a firm's executive to formulate market transactions security demands at a high abstraction level. On the other hand executing market transactions needs a more formal, machine readable description of the transaction and its security requirements. In this paper we present a methodology to specify secure protocols, which are usable to automatically conduct business processes as well as market transactions.

1 Introduction

By using the Internet for commercial purpose the significance of electronic commerce increases due to Internet's openness, which offers new opportunities and therefore changes our way of doing business. Electronic commerce is "... the sharing of business information, maintaining business relationships, and conducting business transactions by means of telecommunications networks" [14]. Electronic commerce reduces transaction costs due to the use of information technology (IT) and electronic markets in the Internet are therefore supposed to produce lower prices and bigger margins than traditional markets [8]. Why do many executives and consumers still hesitate to conduct their business activities in the Web? Our hypothesis is that users want to have integrated tools guaranteeing security and fair trade. The tools must be embedded in a legal system which protects from fraud and larceny. Executives who are responsible for trouble free and optimal execution of electronic business transactions want secure and fair trade and they therefore need possibilities to analyse and specify their security needs by using appropriate techniques.

In recent years a boom in the research field of electronic commerce is taking place. Mainly through the rapid adoption of the Internet by the users today nobody doubts the fact that electronic commerce is a strategic sector. This trend also inspired the research fields of IT-security, workflow management, business (re)engineering and others. The outcome of this research is that we have solutions in many areas like cryptographic mechanisms, secure payment protocols, workflow management systems and new promising business models. Up to now the integrated view of abstract specification and enabling technologies are rarely addressed. The relation between these areas is displayed in figure 1.

Some work about security for workflows, business processes, and market transactions has already been done. But most of them focus on authorisation mainly, for example Thomas and Sandhu [13], Bertino et al. [1], and Bußler [2]. For an appropriate integrated view on secure business transactions we need a broader view on security [7]. For example, non-repudiation of a message containing a document or originality of a payment token may be demanded for business transactions.

There are also a lot of single solutions for electronic commerce that allow fair exchange, secure payment, signing of digital contracts, and so on. But if you consider yourself to be in the position to realise a business model for the production, offering and selling of a digital product in the Internet your best choice would be hiring a software expert with a security expertise. comes visible by focusing on virtual enterprises which execute distributed business processes over open communication networks. By using the same underlying technologies the same security requirements may be taken into account. In this paper we use the concept of business transaction to describe both market transactions and business processes.

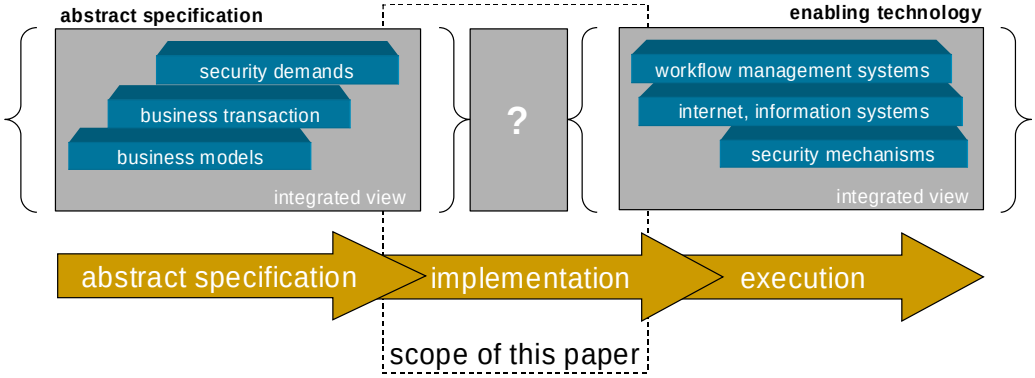

Figure 1: Scope of modelling secure business transactions

Taking into account that business models change relatively often the integration gap between the left and the right side of figure 1 shows its importance.

In this paper we present a language to specify secure business transactions which is integrated in a model, that describes the whole way of an electronic business transaction: from its abstract specification by an executive its implementation in machine readable form and finally to its execution. We see our work as a first step to close the gap in figure 1.

The structure of this paper is as follows: We first describe the integrated model in which ALMO$T is embedded in chapter 2. Chapter 3 contains the outline of the specification language ALMO$T (A Language for Modelling $ecure Business Transactions). ALMO$T can be used to specify protocols that realise security and integrity requirements of business transactions. The goal is to meet the users security needs. Chapter 4 gives an overview how the execution of business transactions is done with ALMO$T followed by the conclusion in chapter 5. In two appendixes we include the definition of ALMO$T with a BNF grammar and a real world example of a business transaction realised in ALMO$T.

2 Secure Electronic Business Transactions

Modelling methods for business processes are used to specify processes in firms. Anyway, they may be used to specify market transactions, too. Focusing on security requirements of electronic business processes and market transactions an alignment may be recognised. This be-

2.1 Electronic Business Transactions

Market transactions and business processes are interrelated. While business processes describe the activities inside the firm, market transactions describe the coordination of the (potential) business partners. More precisely, a business process is a general activity (or set of inter-related activities) with the intention to support an organisation to reach its (business) targets. Commonly, in each business process a relationship to business process partners exists. The interactions with business partners takes place on markets by market transactions. A market transaction is the process of a barter, where either tangible or intangible goods are exchanged between the different parties involved. Therefore, a market transaction is usually defined as a set of interactions between market participants in different roles having the goal to make and fulfil a contract concerning the exchange of goods.

Business transactions have immanent security requirements like confidentiality and legal binding of information, privacy, non-repudiation of having participated in a communication and so on. One of the most important security requirement for trade is to provide the integrity of traded goods, which means to conserve the value of traded goods by protecting their specific property rights. Integrity of a good may differ depending on the type of the good. As an example free tradable rights on electronic markets like emission permits (see example in appendix B) are valuable as original, only. To guarantee integrity of an emission permit the requirement originality has to be maintained. Looking on emission

permits additionally anonymity may be important for economic reasons [9].

We introduce this idea of digital goods' integrity by extending the traditional view of integrity of a message because we see the importance for secure electronic commerce. Integrity of goods depend on the circumstances of the application. In contrast the integrity of a message results from the conditions of the communication.

Another important security demand is fairness, which also is not possible to ensure on basis of messages and communications. Fairness means that either all of the business partners got everything they expected, or none got anything and none lost anything at the end of a transaction. It is obvious, that realisation of the integrity of the good is part of the realisation of fairness. Especially, this is true when coin-based electronic payments are viewed as a type of digital good. When a business partner has sent a coin it immediately lost its value for him.

Security requirements result from different circumstances from internal of the enterprise or from the environment the enterprise is acting in (e.g. laws or ethical requirements). Each security requirement may occur at different security levels, because different transactions may have different risks. Therefore, different security mechanisms with different strength are needed.

2.2 Model and Realisation

To solve the security problems mentioned in the previous subsection an infrastructure which realises the security services and mechanisms is needed. To make these realisations usable for the person who specifies the secure business transactions, a way to combine existing solutions with new ones must be provided. In this section we first describe a model that shows the whole way from the specification at a high level of abstraction to the realisation of secure business transactions. Then we describe the basic properties of the security infrastructure in which a specified secure business transaction can be conducted, followed by some remarks on security service's characteristics.

2.1.1 Specification Model

To support realisation of secure electronic business transactions we have developed a three-layered architecture (for more information see [10]). At the upper layer graphical concepts to specify security requirements of business transactions are offered. To fulfil a specified security requirement the corresponding business process must include functionality which guarantees it. Usually such functionality is neither included in business process models nor in workflow management software and thus modifications or extensions are necessary. Such modifications are supported by a repository of use-cases located at the highest layer of the architecture. A business process is described by different perspectives (cf. [3]) which produces an integrated, consistent, and complete view of it. Use-cases offer modifications of different perspectives of the business process model to realise the demanded security requirements.

The representations of business processes have relationships to an infrastructure for electronic markets. That's why to realise a business process an enterprise may act on an electronic market, e.g. to offer its products or to use some services offered by other market participants (e.g. notary service). An electronic market is represented as a three-dimensional polygon which is structured in three layers correspond to the phases of a market transaction (information phase, negotiation phase, execution phase1); the edges represent different participants of the market. Besides the economic parties supplier, demander and intermediaries also parties, who realise trusted services2 and information services are needed to build a secure electronic marketplace. Trusted services are important for contracting and for digital goods which often need authenticated time, originality, and similar properties. For future electronic markets we also expect a lot of new tasks for trusted third parties. Information services provide technical information about the market infrastructure and the network. Examples are certificate directories or a special host which processes inquiries like: "What is the network address of a trusted third party issuing secure time stamps?". Business transactions are specified to describe the processes that are executed locally at each party and the protocols which are steering the co-operation between the parties.

The middle layer of the proposed architecture offers a repository of already modelled solutions of basic security elements and of activities linked with security requirements. Basic security elements are abstract descriptions of security mechanisms which enclose all information for their realisation. For example, verify digital signature R of alleged signatory S. An example of an activity linked with security requirements is the activity "deliver a licence anonymously under consideration of its originality".

These solutions are created by security experts located in the involved enterprises or employees of a third party of the market (enterprise) offering solutions of security problems. The ALMO$T language is the main

1 Sometimes instead of the execution phase two phases are define: the settlement and the adjustment phase.

2 Parties which realise trusted services are usually called trusted third parties.

means to bridge the gap between basic security elements and activities lined with security requirements respectively and soft- and hardware modules used for their realisation. It is located at the middle layer. For example, to realise basic security element "verify digital signature of signatory" the software building blocks "request certificate of signatory" and "proof digital signature using MD5 and RSA" are needed. Specifications on the middle layer should be built as kind to execute automatically.

At the lowest layer of the architecture a repository with hard- and software building blocks is included. They are combinable to realise security services.

Protocols specified with ALMO$T, as each other kind of protocol too, may be used in a business transaction only if all participants of that business transaction accept it and act according this protocol. If no agreement about a procedure to realise a certain degree of required security is reached, at least one participant of the business transaction must modify its security requirements or the business transaction may not be executed with these participants.

2.1.2 Characteristics of Security Services

In many cases the local usage of security services and mechanisms is not sufficient, and an infrastructure to offer additional services is necessary. In this subsection we discuss characteristics of such services.

In the example of an application of the digital signature the trusted service public-key-certification and the less trusted service public-key-directory are necessary. Theses services are offered by third parties, which can be part of the global environment, the local environment (e.g. the enterprise), or in the private environment of a particular participant. For example, a directory may offer its services globally to all participants or only to participants inside a corresponding enterprise in an address book, which contains public keys of communication partners and which is stored in the enterprise's local environment. Moreover, a participant of a business process may realise his/her private directory.

Using third parties of different environments may lead to different results. For example, usage of a third party from the local environment to obtain a certificate of a public key may lead to an obsolete public key because the corresponding certificate is declared invalid in the global environment but the local directory was not updated.

In most business processes third parties are involved, especially in secure business processes. Services of third parties may be used online which means the service is produced at the moment the service is needed, which includes a communication between the business participant and the party offering the service during usage of the service. In some cases the demanded service is produced in advance and the result is stored e.g. in the local environment of the business participant. For example, to register the originality of a document the third party must be involved in the business process online. But to obtain a certified public key an online service is not always necessary. If the corresponding certificate is stored locally the service may be produced offline. For security requirements the distinction between offline and online is relevant, because the use of online services may result in less availability and higher risk of vulnerability.

To sum it up, services may be established in different environments (private, local, and global) and the produced services may be integrated into business transactions either in advance (offline) or simultaneously (online).

3 ALMO$T

In this section we explain the overall design of ALMO$T. What we present is an approach which uses object-oriented ideas and notations for two reasons. First, because this results in a paradigm, which is easy to understand and straightforward to use and second, the realisation of the infrastructure is almost completely written in the object-oriented language Java. The syntax of ALMO$T-statements are given in appendix A.

3.1 Special Objects, Classes and their Interfaces

Because certain objects and classes occur in almost any business process, we include them as predefined classes and objects such as `doc`, `kCrypto` and `net`. For example, `doc` is the base class for documents, `kCrypto` the base class for key-based cryptography. The underlying communication network is a special object called `net`. If a special net should be used for communication (e.g. Internet) it can be derived from the class net as an additional object.

All classes are part of the same class hierarchy that is shown in figure 2. The root of this hierarchy is the abstract base class `object`. All other objects in ALMO$T are extensions of this class. The interface of `object` includes the method for assignments `set` and methods for comparisons, which are denoted by the corresponding mathematical relation combined with a question mark (e.g. =?, ≤?).

Similar to the class `object` there are predefined interfaces for the classes `boolean`, `doc`, `net`, `kCrypto`. Objects for key-based cryptography offer the methods `encrypt`, `decrypt`, and `verify`. These methods need the parameters `doc` (representing the data to process) and the key to use (e.g. `RSA.encrypt(key,data)`,

where RSA is an extension of kCrypto). The methods send and receive of a network object net refer to the sent/received documents. They are used in combination with the communication net on which the data should be sent or received. Assigning a new value to an object the method set is used (e.g. rating.set(clear accept)). A document is an object which consists of a set of attributes. To access an object that is an attribute of a document, the get_attr method of the class doc is applicable. doc.get_attr(tip) delivers the value of the attribute tip of the object doc. Predefined classes are summarised with their interface in table 1.

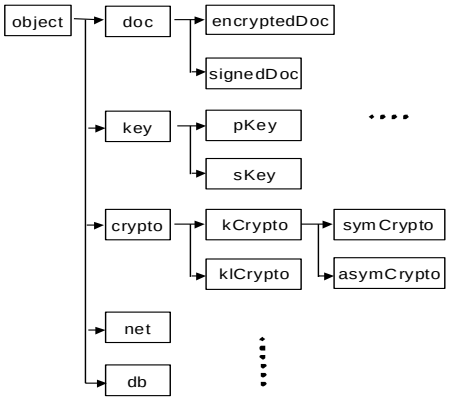

Figure 2: Class hierarchy of ALMO$T

object	abstract base class f. objects
create(object)	standard constructor
<rel>?(object)	compares two objects according to <relation>∈ {=,≠ ...}
set(object)	assigns a value to the object
boolean	objects of boolean type
<op>(boolean)	<op>∈ boolean operators
doc	base class for all documents
attach(doc)	attach doc to the document
detach(doc)	detach doc from the document
get_attr(id)	selector for attributes
set_attr(id, object)	equivalent to get_attr(id)set(object)
kCrypto	base class for key-dependent cryptography
encrypt(key,doc)	encrypt doc by using key
decrypt(key,doc)	decrypt doc by using key
verify(key,doc)	verify doc by using key
net	communication network
send(doc,id)	send doc to id
receive(doc,id)	receive doc from id
db	database object
remove(object)	removes object from database
add(object)	adds object to database
get(object)	get object from database

Table 1: Classes and their interface

3.2 Multiparty Business Transactions and Environments

In general more than one party may be involved in a business transaction. Therefore, ALMO$T allows to specify business transactions for single parties involved as well as together for all parties part of the business transaction from in- and outside a company. A specification of a business transaction is an object and the actions performed are its methods. To specify a multiparty business transaction for *n* different parties a table with *n* rows allows parallel specification of the business transaction for all participants. Because different parties are specified in one view an environment structure must be part of ALMO$T. This has an important impact on security due to the fact that the public services as well as the private-key must have its own definite place in the environment. For that reason a cascading environment structure has been chosen. Listed from the inner environment level to the outmost the different environments are:

- the *temporal* environment of the method,
- the *private* environment of the party,
- the *local* environment, to which a set of parties have access, and
- the *global* environment.

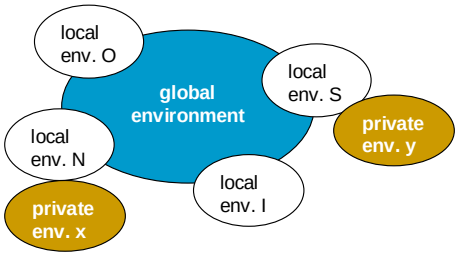

Figure 3: Environments

Used objects may be located at different levels. For example, an object may be located global to each business party or local to a single business party. Assume two objects with the same identifier directory are located at two different environment levels, e.g. global and local environment. Using one of these objects may lead to a different result in comparison to the use of the other one. Therefore, the priority of the environments is given by the following search order for identifiers: At first the temporal environment is searched for an identifier and if found the other levels are not been searched anymore. Otherwise the same procedure goes on until the outmost environment level (the global environment) is reached.

It is possible to define explicitly the environment in which an object is used. If a search strategy is explicitly given in a statement, as in the example

`env.local(pKeyring(Boss))`, the standard search strategy (temporal → private → local → global) is not applied to the identifiers. For example: `env.global(db.get(Boss))` looks for our boss in the database of the global environment.

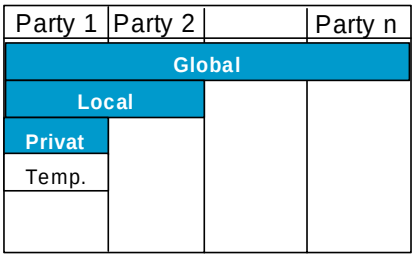

Figure 4: Multiparty business transactions

Figure 4 illustrates the relationship between different environments. The global environment is relevant for each participant of a business transaction. The whole infrastructure together with its objects that realise trusted or non-trusted services is located in the global environment. Each local environment is relevant for one enterprise and each private environment represents the objects internal to a party. Private environments must not overlap any other environment. Here are the most sensible objects located (such as secret keys or original objects like digital coins).

3.3 Constructs

In business processes it must be possible to change the control flow depending on certain conditions. For example, in signing a contract different roles are used depending on contract's value. That is why the if_else-construct is introduced. Additionally, a construct to model iteration loops is needed, which is realised in ALMO$T by the while_do-construct. The set of production rules (see appendix A) must be extended by the following rules:

if_else → `if` <statement> <sequence> `if_end` |
`if`<statement> <sequence> `else`
<sequence> `if_end`

while_do → `while` <statement> `do` <sequence>
`while_end`

The extended production-rule of statement must is

statement → if_else | while_do

3.4 Conventions

For a better understanding of grammar expressions we introduce some naming conventions. As shown above third parties may be trusted or not. Because this distinction is important for the selection of third parties, this characteristic should be denoted. We use the prefix "t_" to denote the feature trusted and we use no prefix to denote non-trusted third parties. The prefix "t_" may be used to express the characteristic "trust" of other objects, too. The identifiers of all objects representing a database start with "db_". Table 2 summarises these conventions.

`t_<object>`	<object> must be trusted
`db_<object>`	db_<object> is a database

Table 2: Summary of naming conventions

In business transactions there are objects which may be used in many cases. Such commonly used objects in secure business transactions are for example the public-key pairs of participants. For these cases identifiers are reserved and predefined in ALMO$T. For example, `pKey` represents the public key and `sKey` the secret key of the party executing an activity located in the private environment. `pKeyring` represents a collection of public keys.

`pKey, sKey`	public key respectively secret key belonging to the party conducting the business transaction
`pKeyring`	list of previously obtained public keys of business partners
`env`	environment selection object
`global(object)`	<object> from global environment
`local(object)`	<object> from local environment
`private(object)`	<object> from private environment

Table 3: Summary of predefined objects

To denote a special environment to search for an object we use the object `env` which has the selectors `private`, `local`, and `global` for the respective environment level. Table 3 summarises some predefined objects.

3.5 Using ALMO$T

Although our model bridges the technical part of the gap described in the introduction (figure 1), it is not yet an integrated approach. More than in other areas in IT-security the aspect of "proper use" is a very sensible and vital one. On the other hand there is the demand of efficiency from the business point of view. Therefore, we have to discuss some issues of the way how ALMO$T is intended to be used.

Because the task of achieving a generic, flexible, efficient electronic commerce solution that also reaches some degree of security is a very complex task, we are aware that the ideas we describe are only a first step towards the goal.

Up to now we identified four properties that will promote security and efficiency:

- having a visual concept
- having different levels of abstraction
- having a reuse concept and
- transparency

Therefore, we are realising a graphical editor providing this properties. With this editor it is possible to edit ALMO$T specifications at different levels of abstraction. These different levels of abstraction are also represented in the class hierarchy in figure 2. For example the user may abstract from the security mechanism. Instead of using the object RSA it is possible to use the abstract class `asymCrypto`. The person who specifies a business transaction in this abstract way does not specify the concrete security mechanisms. This may be done later, supported by the editor. This might be useful to react on changes on the security policy of the company or discovered weaknesses of security mechanisms. Additionally, these abstract specifications serve as design pattern for further specifications. For example, looking at the first statement of T2 of activity negotiations in appendix B:

```
net.send(RSA.encrypt(pKeyring.get(D),draft.
modified()),D)
```

If the used cryptographic method should not be fixed at specification time, but the usage of any asymmetric cryptographic method that is to be selected, the statement may specified as follows:

```
net.send(asymCrypto.encrypt(pKeyring.get(D)
,draft.modified()),D)
```

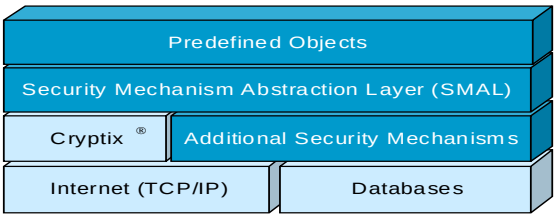

Figure 5: Realisation of objects

A class browser component of the editor will support reusing by allowing to store and browse security mechanisms, protocols and even already specified business transactions by their security properties. When selecting a solution the user simply drag-n-drop a graphical symbol that identifies the solution he/she intends to use.

4 Execution

In appendix B we give an impression of our formalism by specifying the example of the purchase of a digital good (a digitally represented emission permit). The specified example uses the following basic elements: encryption, decryption, and verification of the object RSA, the `add`, `remove` and `get` methods of a database, and the `send` and `receive` methods of the networks.

For executing this specification we have implemented a software library shown in figure 5, which contains basic elements, especially security mechanisms (lowest layer of the proposed architecture, subsection 2.2.1). The modules are implemented in JAVA by using the Cryptix 1.0 library [12] which provides basic public-key crypto-

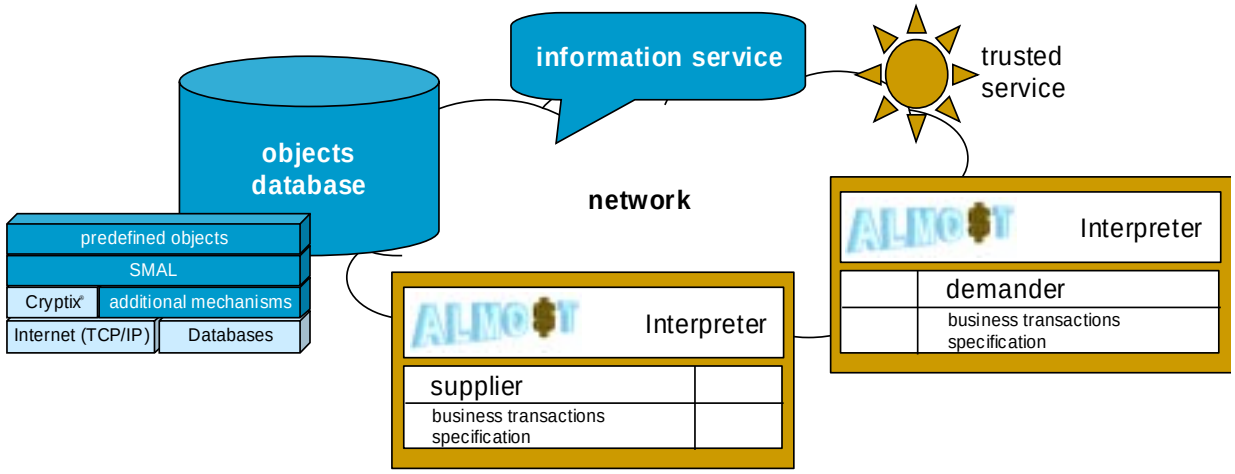

Figure 6: Execution of business transactions in an electronic commerce infrastructure

graphic mechanisms (e.g. RSA) and a collection of cryptographic mechanisms such as DES, IDEA, Blowfish, MD5, MD4, and SHA. All further cryptographic mechanisms in the SMAL are implemented by our own. In the first prototype we will use PGP 2.6.3i message and file formats for compatibility reasons [15]. For the certificate directory we use an Oracle 7.2 database with a jdbc Java interface and CORBA compatible object request broker3. The certification authority which currently is under development will provide X.509v3 [6] certificates and support the ISO certification infrastructure [5].

The realised objects are loaded from the different libraries during the time, when the specified business transaction is started in the ALMO$T-Interpreter. Each object or at least its interface (in the case the object is realised externally) is stored in a database and can be accessed according to the environment layer where it is defined. For example, to get public key of the supplier S in the version of global environment `env.global(pKeyring.get(S))`, communication with a component offering a directory service is necessary (example for an information service in figure 6). Other external objects are for example certification authorities, who are offering trusted services.

5 Conclusion

In this paper we introduced the specification language ALMO$T for specification of secure business transactions, which is usable for the specification of business processes in workflow management as well as for electronic markets transactions. The way to do that is to combine services provided in private, local and global environments that are represented as objects of which methods can be called from within a business transaction. Together with the specification model the whole way from abstract business transaction specification to the realisation and execution in an infrastructure was considered. An example was given in Appendix B.

In our future work we will address the usability issues by research in the way how an abstract security policy may be achieved with concrete mechanisms and protocols as we sketched in section 3.5. Additionally, the tasks of employees in an enterprise may be typed. An approach to define the authorisations and duties of such a type is the role-concept [11]. We consider to include the role-concept in ALMO$T.

6 References

[1] Bertino, E.; Ferrari, E.; Atluri, V.: A Flexible Model Supporting the Specification and Enforcement to Role-based Authorisations in Workflow Management Systems. Proceedings of Second ACM Workshop on Role-based Access Control, 1997.

[2] Bußler, Ch.: Access Control in Workflow Management Systems. Proceedings of IT Security'94, Oldenbourg-Verlag, 1995, pp. 165-179.

[3] Curtis, B.; Kellner, M.; Over, J.: Process Modeling. Communication of the ACM, vol.35, no.9, 1992, pp. 75-90.

[4] I-Kinetics Inc.: http://www.i-kinetics.com/ (last accessed 9/1997)

[5] International Organisation for Standardization (ISO): Information processing systems - Guidelines for the Use and Management of Trusted Third Parties - Part 2: Technical Aspects. International Standard ISO/IEC Working Draft 14516-2, Genf, 1995.

[6] International Telecommunication Union: Information Technology - Open Systems Inter-connection - The Directory: Authentication Framework. ITU-T Recommendation X.509, 1993.

[7] Karlapalem, K.; Hung, P.: Security Enforcement in Activity Management Systems. NATO ASI on Advances in Workflow Management Systems and Interoperability, Istanbul, 1997, pp. 166-194.

[8] Malone, T; Yates, J; Benjamin, R: Electronic Markets and Electronic Hierarchies. In, Communications of the ACM, vol.30, no.6, 1987, pp. 484-497.

[9] Röhm, A. W., Gerhard, M.: A Secure electronic market for Anonymous Transferable Emission Permits. In: Proceedings of Thirty-First Hawaii International Conference on System Sciences HICSS-31, 1998.

[10] Röhm, A.W.; Pernul, G; Herrmann, G.: Modelling Secure and Fair Electronic Commerce. Proceedings of the IEEE Annual Computer Security Application Conference ACSAC, 1998, pp. 155-164.

[11] Sandhu, R.S.; Coyne, E.J.: Role-Based Access Control Models. IEEE Computer, February 1996, pp. 38-47.

[12] Systemics Ltd: http://www.systemics.com/software/cryptix-java/ (last accessed 9/1997)

[13] Thomas, R.; Sandhu, R.S.: Task-based Authorization: A Research Project in Next-generation Active Security Models for Workflows. Proceedings of NSF Workshop on Workflow and Process Automation in Information Systems: State-of-the-art and Future Directions, Sheth, A. (ed.), Athens Georgia, 1996.

[14] Zwass, V.: Electronic Commerce: Structures and Issues. International Journal of Electronic Commerce, vol.1, no.1, Fall, 1996, pp. 3-23. http://www.cba.bgsu.edu/ijec/

[15] Zimmerman, P.: PGP User's Guide, Volume I: Essential Topics. October 1994.

3 product of i-kenetics [4]

Appendix A: Syntax of ALMO$T-Statements

Our language is based on the grammar G = (VN,VT,P,S) explained in the following. Non-terminals are VN = {bt, sequence, statement, identifier, object, method, arguments, digit, letters, integer, real, boolean}, where “bt” represents a business transaction. “statement” represents an activity. “object” represents an object, "method" represents a method of the object, and "arguments" are the objects passed to a method, when it is called. Additionally the boolean values true and false and also the integer and real constants are objects (7) and can be deduced to terminal symbols by the rules (1), (2), (3), and (4).

The set VT of terminals includes the identifiers of objects used in business transactions, produced by the rules (6), (1), and (5). It includes especially the object ϵ that represents an empty statement. The start-symbol S is defined by S = bt. The set P consists of the following production rules (where A°B means the concatenation AB of A and B):

(1)	digit	→	0|1 ... |9
(2)	boolean	→	true|false
(3)	integer	→	digit | integer°digit
(4)	real	→	integer, integer
(5)	letters	→	a|b|c...|z|A|B...Z|-|_| digit
(6)	identifier	→	letter | identifier°letter
(7)	object	→	identifier | integer | real | boolean
(8)	method	→	identifier
(9)	arguments	→	arguments, arguments | object | object.method(arguments) | ϵ
(10)	statement	→	object.method(arguments) | object
(11)	sequence	→	ϵ | sequence statement
(12)	bt	→	sequence

Appendix B: Example Specification

To explain our methodology we use as example the business process of purchasing an anonymous emission permit on an electronic market. To simplify our example, we leave out the role of the broker. An emission permit is a licence, which allows its owner to emit a certain amount of toxins into an ecological area. The model of emission permits is a successful tool for environmental policy. Its weakness are the high transaction cost, that can be reduced when digital emission permits are used in the way described in detail in [RöGe98]. This market transaction serves as a real world example, which we will realise in ALMO$T.

The information phase begins with the demander asking for offers (step 1). Afterwards he/she receives offers from potential suppliers (step 2). The negotiation phase consists of two activities: “negotiation” and “completion-of-contract”. In the negotiation activity the demander creates the first draft contract and then the two business partners exchange draft contracts until they agree and conclude. Step 3 represents the exchange of draft contracts from demander to supplier and step 4 represents the exchange vice versa. Activity “completion-of-contract” is responsible for legal binding of the contract. To realise legal binding of contract first the demander sends the signed contract to the supplier (step 5) then the supplier asks an information service for the public key of the demander (step 6) and verifies demander‘s digital signature. If the signature is valid, the supplier signs the contract, too and sends it back to the demander (step 7). The demander checks supplier‘s signature by using its public key received from an information service (step 8). The execution phase begins with step 9 where the demander generates a session key and sends it encrypted with the public key of the issuer (trusted third party) to the supplier. The supplier sends his/her original permit together with the encrypted session key to the issuer (step 10). The issuer generates a new original and encrypts it with the session key. Then he/she sends it to the supplier (step 11), who gives it to the demander (step 12), who pays electronically (step 13).

In figure 7 the secure delivery of the original, anonymous document representing the emission permit is modelled.

Explanations to the notation:

- The first row includes the roles involved in the corresponding activity.
- In the given example: supplier (S), demander (D), and issuer of emission permits (I).
- The second row (named as G) represents the global environment.
- The row named "L" represents the local environment of the firm the role at the top of the corresponding column belongs to.
- Row "P" represents the private environment of the roles given at the head of each column.
- Row "T" represents the temporal environment of the roles given at the head of each column.
- T1, T2 name task1 and task2.
- (Ti,Y) represents the actions of Y in task Ti ($Y \in \{S,D,I\}$)

The ALMO$T specification of activities “negotiation” and “completion-of-contract” representing the negotiation phase of the example are not included in this paper, but may be found on the web (www.wi-inf.uni-essen.de/~ifs/publikationen/)

The activity "delivery" guarantees the anonymity of the demander in the execution phase and the originality of the permission permit traded. To guarantee the originality of the permit, the demander generates a session key K for confidential communication with the issuer and send K encrypted with the public key of the issuer (`RSA.encrypt(pKeyring.get(I),K)),S)`) via the supplier to the issuer (T1 and (T2,S)). Additionally to the session K the supplier sends the permission permit (O) to the issuer (T2,S).

The issuer checks the originality of the permission permit[4], creates an new version number for the permission permit and sends it encrypted with the session key K via supplier to the demander (T3,I). The demander checks the encryption of the permission permit and stores it[5].

Activity delivery			
	D	S	I
G	net, RSA, IDEA, V		
L	db_disk		
P	pKeyring, sKey	pKeyring, O, sKey	db_originals, pKeyring, sKey
T	K, m, O	m	K, m

T1	K.set(V.random(128))
	net.send(RSA.sign(sKey,
	  RSA.encrypt(
	  pKeyring.get(I),K)),S)
		net.receive(m,D)
		while
		RSA.verify(pKeyring.get(D),m).=?
		(false)
		do net.receive(m,D)
		while_end	

T2		m.attach(RSA.encrypt(pKeyring.get(I),O)) net.send(RSA.sign(sKey,m),I)	net.receive(m,S) **while** RSA.verify(pKeyring.get(S),m).=? (false) **do** net.receive(m,S) **while_end** RSA.decrypt(sKey,m) O.set(detach(m))

T3	IDEA.decrypt(K,net.receive(O,S)
	)
	While
	RSA.verify(
	  pKeyring.get(I),O).=?(false)
	do IDEA.decrypt(
	  K,net.receive(O,S))
	do_end
	db_disk.add(O)
		net.receive(m,I)
		net.send(m,D)
			If O.=?(db_original.get(O))
			db_original.remove(O)
			db_original.add(
			  O.set_attr(ver,V.random(128)))
			K.set(RSA.decrypt(
			  sKey,detach(m)))
			net.send(RSA.sign(
			  sKey,IDEA.encrypt(K,O),S)
			if_end

Figure 7: Execution phase of the purchase of an emission permit

4 The case that the permission permit is not original is not specified in the example.

5 The case the verification fails is not specified in the example.