Electron Commerce Res (2006) 6:305–335
DOI 10.1007/s10660-006-8677-7

Security requirement analysis of business processes

Peter Herrmann · Gaby Herrmann

Abstract Economic globalization leads to complex decentralized company structures calling for the extensive use of distributed IT-systems. The business processes of a company have to reflect these changes of infrastructure. In particular, due to new electronic applications and the inclusion of a higher number of—potentially unknown—persons, the business processes are more vulnerable against malicious attacks than traditional processes. Thus, a business should undergo a security analysis. Here, the vulnerabilities of the business process are recognized, the risks resulting from the vulnerabilities are calculated, and suitable safeguards reducing the vulnerabilities are selected. Unfortunately, a security analysis tends to be complex and affords expensive security expert support. In order to reduce the expense and to enable domain experts with in-depth insight in business processes but with limited knowledge about security to develop secure business processes, we developed the framework MoSS$_{BP}$ facilitating the handling of business process security requirements from their specification to their realization. In particular, MoSS$_{BP}$ provides graphical concepts to specify security requirements, repositories of various mechanisms enforcing the security requirements, and a collection of reference models and case studies enabling the modification of the business processes. In this paper, the MoSS$_{BP}$-framework is presented. Additionally, we introduce a tool supporting the MoSS$_{BP}$-related security analysis of business processes and the incorporation of safeguards. This tool is based on object-oriented process models and acts with graph rewrite systems. Finally, we clarify the application

P. Herrmann
Department of Telematics, Norwegian University of Science and Technology, 7491 Trondheim, Norway
e-mail: herrmann@item.ntnu.no

G. Herrmann
Institute of Computer Science and Business Information Systems, University of Duisburg-Essen
45141 Essen, Germany
e-mail: herrmann@wi-inf.uni-essen.de

of the MoSS$_{BP}$-framework by means of a business process for tender-handling which is provided by anonymity-preserving safeguards.

Keywords e-Commerce · Business process · MoSS$_{BP}$ · Object-oriented security analysis · Graph rewriting

1. Introduction

The evolution of the Internet from a net used predominantly by researchers to an instrument used by nearly everybody in industrial countries leads to the evolution of electronic commerce applications. These applications vary from business-to-business applications to business-to-consumer and administration-to-consumer applications. The growth of the number of e-commerce users, however, is weaker than expected. One argument for this development is that many potential users distrust e-commerce applications fearing personal damage due to real or assumed lack of security.

Companies consider this fear by designing secure business processes. When a company adapts its business processes to its IT-infrastructure in order to act with business partners (other companies or final consumers) electronically, the modification of the business processes have to fulfill certain security requirements. In particular, one has to reflect a new class of security aspects which are relevant to e-commerce-based but not to traditional business processes. For example the usage of signatures is a well established and legally unambiguous method to subscribe traditional "paper and pen" contracts. The use of digital signatures for signing contracts electronically is a new and not yet settled field in e-commerce. Moreover, due to either non-existing laws or laws containing impractical solutions, the legal consequences of electronic contract signatures are not yet clear.

A security analysis is a suitable method to address security aspects of a business process. The business process is audited for vulnerabilities and threats which may cause security risks. Based on this audit effective safeguards are selected, designed, and configured. In detail, an audit comprises a possibly iterated series of phases concerning the following subtasks (cf. [4]):

1. Identification of the business processes, their elements, and the related human principals,
2. valuation of the assets contained in the business processes and definition of their security levels,
3. identification of security requirements resp. vulnerabilities and threats,
4. assessment of resulting risks,
5. planning, design, and evaluation of suitable countermeasures.

Unfortunately, due to the complexity of real-life systems and their security requirements a security analysis tends to be complex and laborious. It is suited to well-trained security experts but not to experts in business application domains. Thus, the engineering of secure business processes is quite expensive since security experts have to be hired for this task. In the last years, however, new approaches were developed which reduce the expense and complexity of the analysis of computer systems. They utilize abstract formal models of the systems and of the security requirements (cf. [3, 11, 35, 39, 40]). The system model forms the basis for the introduction of problem

solutions which are described by model modifications. Finally, the abstract solutions are refined to implementable countermeasures. In this paper we adapt formal-based security analysis to the domain of business processes. In particular, we combine the two approaches MoSS$_{BP}$ and Object-Oriented Security Analysis in order to facilitate the automated realization of security requirements of business processes.

$MoSS_{BP}$ (**Mo**deling **S**ecurity **S**emantics of **B**usiness **P**rocesses, cf. [25]) is an approach to support domain experts which need not to be security experts. The security requirements, a business process has to fulfill in order to be secure, are modeled based on graphical design concepts provided by the framework. Moreover, MoSS$_{BP}$ introduces a procedure to handle modifications of business processes according to their security requirements. Therefore various existing enforcement procedures for security requirements as well as soft- and hardware tools realizing the corresponding safeguards are collected and reference models and case studies guide the modifications.

The approach *Object-Oriented Security Analysis* [27] reduces the efforts of an analysis further by using object-oriented description techniques and graph rewriting to facilitate the design of business process models and to enable automated model refinement. The corresponding tool-support is similar to object-oriented design tools which are well established in the field of software engineering (e.g., [47, 59]). The approach was successfully used for the security analysis of applications based on the middleware platform CORBA [28] and for information flow analysis of component-structured software [26], too.

In this paper, we apply Object-Oriented Security Analysis to the refinement of business processes in order to guarantee security requirements. The corresponding tool support is a useful complement to the MoSS$_{BP}$-framework. It supports the application of the MoSS$_{BP}$-methodology to modify business processes according to the required security requirements. The diagrams representing the business process and its security requirements, can be refined in a highly automated fashion by application of graph rewrite systems (e.g., [2]). A rewrite system consists of a set of graph rewrite rules. Each rule is a tuple of two graph patterns—a pre-pattern and a post-pattern—, an application condition, and an effect function. The rule can be applied to a graph if the graph contains a subgraph which is an instance of its pre-pattern. Moreover, the object attributes in the subgraph have to fulfill the application condition. By application of the rule, the subgraph is replaced by an instance of the post-pattern and the attributes of the replacement objects are set according to the effect function.

The paper is structured as follows: First an overview of related approaches is given. Section 3 provides a survey of security requirements and corresponding business process elements. Thereafter we outline the MOSS$_{BP}$-framework including the different perspectives of MoSS$_{BP}$-models, the architecture of the framework, and the process to utilize MoSS$_{BP}$ (Section 4). The Object-Oriented Security Analysis approach and the corresponding tool support is introduced in Section 5, followed by an application example where a tender-handling process is checked for anonymity (Section 6).

2. Related work

The importance of business process' security is accepted in general (cf. [32]). Many approaches adapt access control and authorization methods used in database and oper-

ation system areas to the domain of business processes and workflows (e.g., [1, 5, 9, 30, 53, 58]). But the handling of security requirements of these areas need a more broaden view. For example, two companies may interact by performing a common business process. The companies, however, may demand different, perhaps contradicting, security requirements from the common business process. A solution to this problem is introduced in [44]. The task management in business processes is addressed by Hung and Karlapalem [33] who use tokens for describing the capabilities and security clearances of human or computer agents performing tasks. A task is also provided by tokens and an agent may perform only a task if its tokens coincide with the tokens of the task.

A more comprehensive approach is SEMPER (Secure Electronic Marketplace for Europe, [38]) facilitating the construction of an open and secure electronic marketplace. SEMPER's main focus is the technical realization of activities fulfilling certain security requirements. The requirements are realized by means of security-related services which are classified by a four-layer architecture. The project COPS (Commercial Protocols and Services, [50]) has a broader view to security issues of electronic marketplaces than SEMPER. It enables the design of an infrastructure for marketplaces supporting all phases of a market transaction (i.e., gaining information, negotiation, completion). The security services offered by SEMPER and COPS can be assigned to the layer 1 of the MoSS_{BP}-architecture (cf. Section 4) while the support components to design and maintain activities based on the services are part of layer 2.

A lot of work was done in the field of security analysis. Baskerville delineates three generations of security analysis methods [4]. The first generation are methods based on checklists. Here, a system is scrutinized for the availability of safeguards by means of checklists. Examples are SAFE [36], the Computer Security Handbook [31], and AFIPS [7]. Tools based on this method comprise [3, 8, 10, 22, 29, 54, 60].

The main drawback of the first generation is the informal and non-structured way of analysis which is hardly scalable to more complex computer systems. This is addressed by the so-called mechanistic engineering method [4]. This method focusses on identifying and solving detailed function system requirements facilitating the reduction of a complex system analysis into easier manageable system requirement examinations by the five steps listed in the introduction. This method was introduced by Parker [43] and Fisher [20]. A well-known tool is CRAMM (e.g., [12]) provided by the UK Government. Here, examiners scrutinize a computer system for its assets by means of checklist-based interviews with the system owners. Based on the interview results, CRAMM develops further questionnaires to determine the threats on the assets and to introduce suitable safeguards. Other tools based on mechanistic engineering are RISKPAC [14], BDSS [42], and CBISA [19].

Unfortunately mechanistic engineering-based security checks tend to be laborious and expensive. The third generation of so-called logical transformational systems intends to overcome this shortcoming by introducing abstract models of systems and security requirements. The extension SSADM of the tool CRAMM [11] is an early solution of this idea. Here, abstract specifications of a system, its problems, the security requirements, and possible technical options guiding the reviewing process are developed in parallel to the CRAMM interviews. Another approach is Baskerville's logical control design method [3] where relevant assets of a system and the threats to them are modeled in a process like way and collected in a dictionary. More recent approaches

concentrate on formal modeling of processes and requirements. For instance, Kienzle and Wulf propose the use of hierarchical organized trees which are called Methodically Organized Argument Trees (MOAT) as a method to assess security of computer systems [35]. Here, security requirements are defined in the form of MOAT roots which can be refined or decomposed into subgoals resp. alternatives. Thereafter the leaves of the trees are justified either by formal verification or by informal plausibility checks. A similar method is the harmonizer approach of Leiwo and Zheng [39]. A major drawback of these approaches is that they root in abstract requirement descriptions. Thus they support the development of secure systems but are hardly suitable to the analysis of existing systems. The Risk Data Repository (RDR) approach of Kwok and Longley [37] centers on supporting security officers to maintain existing systems. The RDR consists of various domains describing relevant elements of a computer system, mappings between domains, and countermeasure diagrams. In the EU-project CORAS [40] a framework is developed combining various security analysis approaches as CRAMM, Hazard and Operability Analysis (HAZOP) as well as Markov Analysis.

Like us, Thoben concentrates on using security analysis for business systems. He developed an approach for the security and risk analysis of workflow based systems [57]. In contrast to our approach, he is interested mainly on the evaluation of attacks and risks which is performed by means of a fuzzy logic. The approach is not considered suitable to MoSS_{BP} since it does not support the selection of countermeasures against attacks. Moreover, it is considered too complex to be used by domain experts.

3. Business process elements and security requirements

To provide a useful definition of the security requirements for a business process, one has first to distinguish the various parts of the business process, the so-called *business process elements*. According to [16, 55] one can tell apart four main categories of business process elements:

- *Agents* represent people and machines performing activities,
- *Roles* represent rights and obligations, which are assigned to agents,
- *Artifacts* are material which is worked with,
- *Activities* represent tasks.

In order to reach a better correlation between business process elements and the security requirements to be fulfilled by them, we adjusted these categories. On the one hand, for the sake of simplicity we omitted the category *role* since roles are assigned to agents. Therefore we can represent the role of an agent by the category *agent* as well. On the other hand, we refined categories in order to get more specific element types which relate directly to security requirements. The refined categories are listed below:

- *Agents*:
  - *Executing agent:* Agent performing a certain task.
  - *Ordering agent:* Agent who instructs another agent to perform a task.
  - *Agent of record:* Agent who is instructed by another agent to perform a task.

- *Artifacts*:
  - *Procedure:* Agents may act according procedures (algorithms) to execute activities.
  - *End product:* After executing a business process (or parts of it, e.g. activities), security requirements may relate to the produced end products. These security requirements may differ from security requirements of security objects used in the executed activities.
  - *Information* is represented by data. This kind of artefact includes all information which is not in the sub-categories *procedure* or *end product.*
  - *Material:* This kind of artifact includes all material which is not an *end product.*
  - *Information flow:*[1] The information flow describes all information exchanged between agents as well as all agents participating in the exchange process.
- *Activities*: An activity describes tasks in their entirety. It includes the executing agents, the procedures used, and the information/material which is used and produced (end product).

Security of computer-based systems mostly concerns confidentiality, integrity, and availability aspects. Our approach, however, is centered on domain experts who need not to be security experts as well. Therefore, a domain expert has a possibly rudimentary perception of business process security requirements which, moreover, is based on the notice of security in the traditional run of business processes. For this reason, it seems better to make a more detailed distinction of security requirements for business processes. In [23] we identified the security requirements listed below.[2] Here, we call the objects, security requirements concern with (i.e., agents, artifacts, and activities), *security objects* and persons, who act as intruders, *security subjects.*

- Common security requirements:
  - *Confidentiality*
  - *Integrity*
  - *Availability*
- Protection of personality:
  - *Anonymity:* The true identity of a security object is hidden and no one is able to uncover it.
  - *Pseudonymity:* Here, anonymity of a security object is realized in principle, but may be uncovered by authorized subjects.
  - *Privacy:* According to [32], privacy "is the right of individuals and organizations to control the collection, storage and dissemination of their information or information about themselves."
- Bindings:
  - *Legal binding:* An information or a specific end product is legally binding if it contains an agreement which can be proven at court.

[1] Generally, in business process models the information flow is not specified explicitly. However, it is relevant in order to realize certain security requirements.

[2] The list is subject to changes since new business processes may call for new security requirements.

  - *Non-repudiation:* In contrast to legal binding this security requirement is devoted to activities which must not be successfully repudiated by an agent. In particular, two different views are possible: At first, it should not be possible for a specific agent to deny doing activities in principle. At second, it should not be possible for agents to deny doing certain specific activities.
  - *Mutual dependency:* Security objects are mutual dependent if activities or properties of a security object lead to activities or properties of a related object.
- Physical property:
  - *Authenticity:* A security object is authentic, if it is what it pretends to be. It may be a copy of the original object.
  - *Originality:* A security object is original, if it is what it pretends to be and it is not a copy.
  - *Rights to use:* The rights to use a specific security object specify, which agents are allowed to use the object and in which manner.
  - *Copyright:* The copyright of a security object is the right to reproduce it.
- *Hiding activities:* This security requirement refers to the guarantee of confidentiality in the performing of activities and, in particular, in the execution or delegation of a procedure. Three different views are possible: At first, an activity must only be invisible if it is carried out, delegated to, or delegated by a certain agent. At second, an activity has only be hiddenly performed if it is executed by means of a certain procedure. At third, the performing of an activity must not be visible at all.

Of course, not every security requirement is relevant for each business process element (e.g., the requirement *copyright* is not reasonable for an *activity*). The useful correlations between security requirements and business process elements are listed in Table 1 (cf. also [23, 24]). Moreover, in some relations we need further refinements of security requirements in order to address specific characteristics of security subjects or objects. For instance, in the example system introduced in Section 6, we have to distinguish the following characteristics of the agent involved in carrying out a tender handling process in a fashion preserving the agent's *anonymity:*

- Against whom anonymity of the agent is required,
- in which actions anonymity of the agent is required,
- for which information, end product, procedure, resp. information flow anonymity of the agent is required in the actions of the business process.

4. MOSS$_{BP}$: A framework to support security of business processes

Domain experts have in-depth knowledge of the specific security requirements of a traditionally running business process. Moreover, only a domain expert knows if the risks resulting from the vulnerabilities of a business process are bearable. If a traditional business process is refined to a modern computer-based process, the domain expert demands that the refined process fulfills the same set of security requirements as the original (e.g., a digital signature should be as legally binding as a traditional "paper and pen" -signature). But as denoted above, in general, domain experts are not computer security experts and they often have only a rudimentary perception of the

Table 1 Correlation between business process elements and security requirements

Business process elem. / Security requirement	procedure	end product	information	material	executing agent	ordering agent	agent of record	activity	information flow
Confidentiality	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆
Integrity	⋆	⋆	⋆	⋆					⋆
Availability	⋆	⋆	⋆	⋆	⋆		⋆		
Anonymity					⋆	⋆	⋆		
Pseudonymity					⋆	⋆	⋆		
Privacy			⋆						
Legal binding		⋆	⋆						
Non-repudiation	⋆				⋆	⋆	⋆	⋆	⋆
Mutual dependency	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆	⋆
Authenticity	⋆	⋆	⋆	⋆	⋆	⋆	⋆		
Originality		⋆	⋆	⋆					
Rights to use	⋆	⋆	⋆	⋆	⋆		⋆		
Copyright	⋆	⋆	⋆		⋆				
Hiding activities	⋆				⋆	⋆	⋆	⋆	

security requirements necessary in a computer-based business process. Therefore, with respect to the design of secure computer-based business processes, at first, a domain expert needs help in the exact definition of security requirements suitable to a business process. For example, it is not sufficient to demand, that a specific communication should be confidential, since this statement is ambiguous. It does not clearly express if the identity of the communicating agents, the content of the communication, or the mere existence of the communication should be confidential. At second, to support the design of a secure business process, a domain expert should be supported in performing a security analysis of a business process guaranteeing that he considers all relevant security requirements. At third, the domain expert must receive help in order to determine the risks for the business process and the selection of suitable safeguards enforcing security requirements.

The project MoSS$_{BP}$ [23] was developed to support domain experts to develop exact definitions of security requirements as well as to analyze and modify business processes in order to fulfill the security requirements. The business processes are modeled by diagrams in the popular Unified Modeling Language (UML, cf. [6]). To facilitate the understanding of the sometimes complex processes, we developed a view-oriented model describing certain perspectives of a business process separately (Section 4.1). Moreover, the framework architecture contains repositories containing various concepts to describe security requirements of a business process as well as means to guarantee the requirements which are introduced in Section 4.2. Furthermore, we defined a process to analyze business processes and to integrate suitable safeguards as shown in Section 4.3.

4.1. Perspectives

As outlined in Fig. 1, one can describe certain aspects of a business process in separate perspectives by means of different UML-diagram types (cf. [6]) in every perspective. In the selection of suitable views, we followed Curtis et al. [16] who define the following

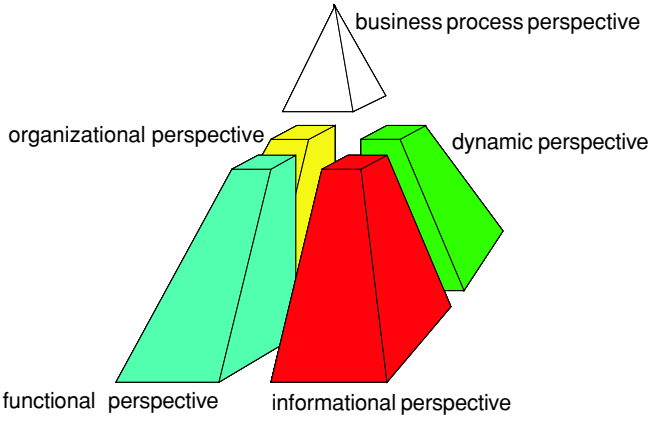

Fig. 1 MoSS$_{BP}$-perspectives

perspectives as necessary to produce an integrated, consistent, and complete view of a business process (cf. [25]):

- The *informational perspective* represents the information entities, their structuring and relationships between them. In our approach, we use UML class diagrams.
- The *functional perspective* shows which activities (processes) are performed and which data flow occurs between these activities. The functional perspective only represents the flow of data within the system. In our approach, we use UML activity diagrams.
- The *dynamic perspective* represents for each information entity all possible states and state transitions which may occur within the life cycle of the information entity. In our approach, we use UML state chart diagrams.
- The *organizational perspective* shows where and by whom activities are performed. This perspective corresponds to the organigram of an organization and to role models. In our approach, we use UML class diagrams.

While the combination of these very specific perspectives provides a complete description of a business process, the overall view is sometimes hard to achieve. Therefore, and in order to be able to analyze a complete business process, we added a fifth perspective which provides an integrated view on the other perspectives:

- The *business process perspective* offers an abstract and integrated view of the business process. In general, it is on a higher abstraction level in order to offer the domain expert an comprehensible image of the business process. Thus, the analysis of the security requirements, the business process should fulfill, is made easier. The business process perspective is similar to the functional perspective but less detailed. Additionally, it refers to the informational perspective and to the organizational perspective. It describes the assignment of the activities to departments by using the UML-construct swimlane (cf. Fig. 5).

In order to provide business processes with safeguards, in general, more than one perspective has to be discussed. For instance, if one changes a tender handling process with authentic tenders from traditional to electronic execution, digital signatures have to be introduced. Thus, the modified business process has to reflect new artifacts as public key certificates guaranteeing the authenticity of the digital signatures as well as new activities like the checking of digital signatures for validity. In particular, the following modifications have to be performed in the different perspectives:

- Functional perspective: A further activity has to be added which checks that a digital signature is valid.
- Informational perspective: An additional class for public key certificates must be added.
- Dynamic perspective: Since the validity of certificates is time restricted, the provableness of digital signatures is only guaranteed for a certain amount of time. Therefore a life cycle model of certificates has to be specified.
- Organizational perspective: An agent managing the validity of digital signatures has to be incorporated.

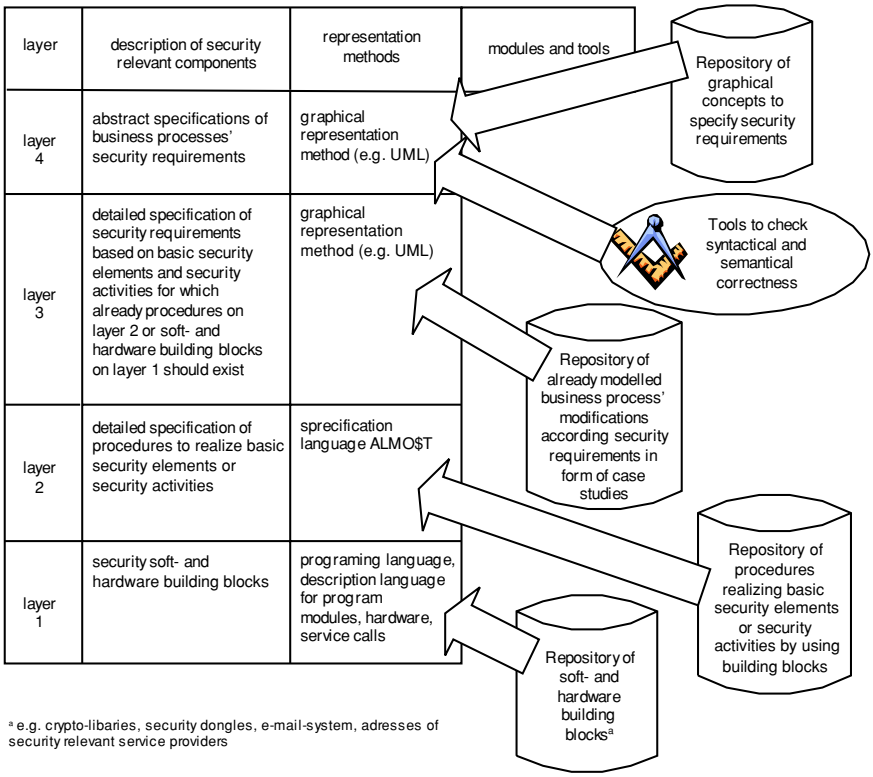

Fig. 2 MOSS$_{BP}$-architecture

4.2. Architecture

We already pointed out that a task of the MoSS$_{BP}$-framework is to support domain experts with a rudimentary knowledge of security aspects to provide business processes with security mechanisms. As a means to facilitate the comprehensibility of the necessary security requirements and the selection of safeguards, we added a layered set of repositories consisting of graphical concepts representing security requirements relevant to business process elements as well as case studies and reference models showing their application in order to facilitate business process modifications. Furthermore, software- or hardware building blocks as well as procedures to create safeguards are collected in repositories in order to clarify the design of the countermeasures. Nevertheless, as discussed in Section 4.3, often the additional support by a security expert is still necessary. The repositories are organized in an architecture of four layers as depicted in Fig. 2 (cf. [25]):

Layer 4: This layer supports the development of an abstract UML-based business process specification by means of a repository of graphical concepts describing typical business process elements and security requirements. The UML-diagrams modeling the five perspectives of a business process are created by application and adaptation of these concepts. In addition, this layer contains a tool checking the

correct assignment of security requirements to business process elements (i.e., a feature of the application SEMBA which is introduced in Section 5). If the tool recognizes a fault, it provides the person utilizing MoSS$_{BP}$ with an explanation of the security requirement, information for which business process elements the security requirement may be correlated, and information for which security requirements the business process element in question may be linked.

Layer 3: To facilitate the modification of business processes, a set of reference models and case studies[3] is included describing sub-processes enforcing security requirements. The sub-processes contain basic security elements and security activities. *Basic security elements* are abstract descriptions of security mechanisms which enclose all information for their realization (e.g., "verify digital signature *sig* of alleged signatory *White*"). An example of a *security activity* is the activity "deliver a licence anonymously under consideration of its originality".

The case studies of this repository are specified by means of the same UML diagram types as the business process models at layer 4 and therefore most domain experts should be able to understand them. A case study describes for each MoSS$_{BP}$-perspective how to act while realizing certain security requirements assigned to a specific security object. It contains basic security elements and security activities which may be realized by procedures on layer 2 or soft- and hardware building blocks on layer 1.

Layer 2: This layer contains procedures to realize the basic security elements and the security activities of layer 3 (e.g., a procedure checking if the digital signature can be decrypted by means of the public key of the contract partner; a procedure checking the originality of the contract partner's public key by contacting a trusted third party acting as a certification authority). To describe these procedures and their combination in an easy and comprehensible fashion, we use the specification language ALMO$T (**A** **L**anguage for **Mo**deling **S**ecure Business **T**ransactions, cf. [49]) which was developed in cooperation with the project COPS [50].

Layer 1: Soft- and hardware building blocks realizing security requirements directly, basic security elements included in case studies, or the procedures of layer 2 are collected in this layer (e.g., a hardware encryption and decryption chip resp. a distributed application enabling communication with certification authorities).

4.3. Proceeding

To protect a computer-based business process against malicious attacks, one has to create a model of the business process, to identify the necessary security requirements, and to search the repositories for corresponding building blocks, procedures, or case studies. If suitable elements are not available, they have to be designed or procured and added to the repositories which, of course, requires the support of security experts. If an addition of new elements is not possible, the domain expert has either to relax the security requirements or the business process cannot be carried out at all. The overall MoSS$_{BP}$-process is outlined in Fig. 3. Here, by different shades of gray we describe

[3] For the sake of simplicity we will consider only case studies below.

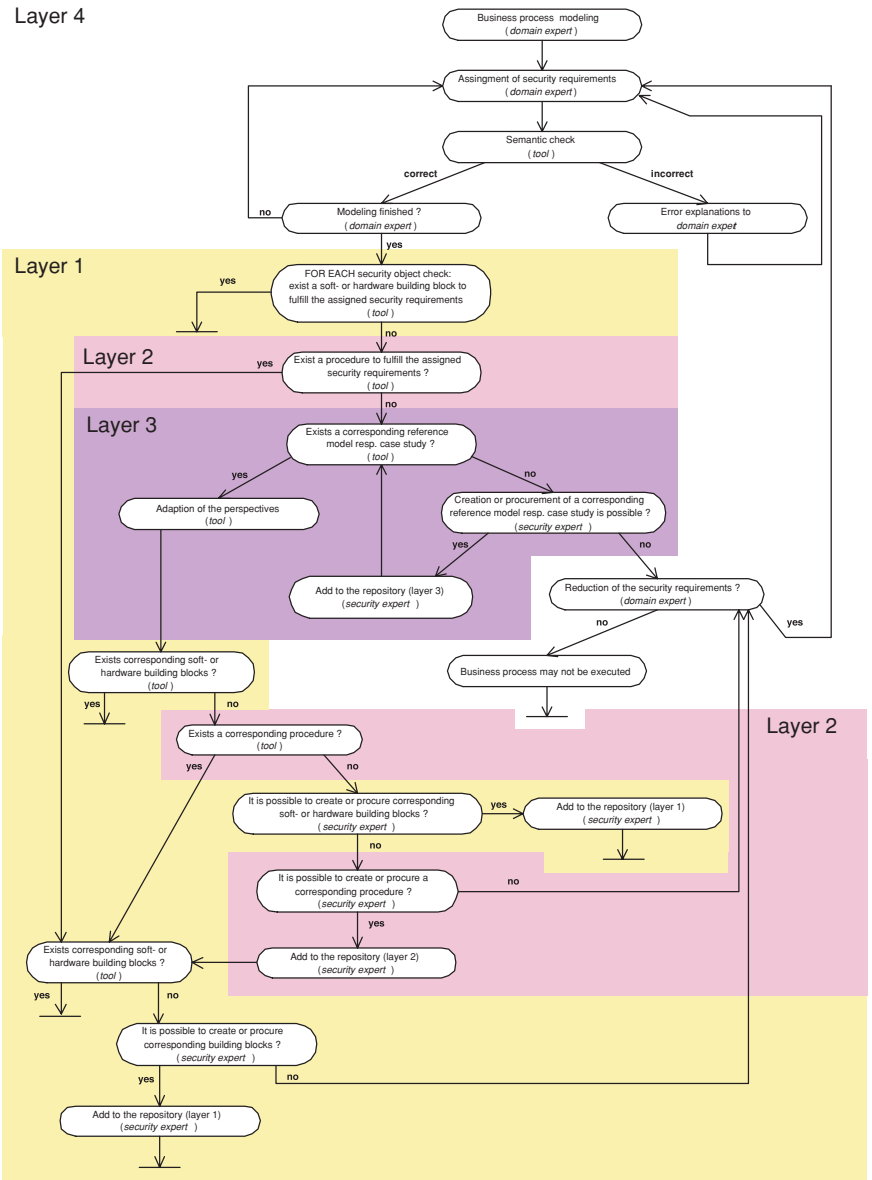

Fig. 3 MoSS$_{BP}$-process

which layer of the MoSS$_{BP}$-architecture is addressed by a particular process step. The process consists of four phases which are introduced as follows:

Phase 1: The domain expert has to identify security requirements and to assign them to the business process elements. Afterwards, a semantic check by the tool support takes place testing if the assignments are correct (layer 4). For each wrong assignment an explanation is delivered and the domain expert has to decide what he really

meant. If, for example, the security requirement *anonymity* is falsely assigned to an *activity*, the semantic checker outputs the following information:

- Explanation of the term *anonymity*,
- information, which security requirements are relevant for the security objects of category *activity*,
- information, for which categories of security objects the security requirement *anonymity* may be relevant.

Based on this information, the domain expert can correct the wrong assignments.

Phase 2: In the next phase, one checks for each security object if for all security requirements assigned to the particular object a corresponding soft- or hardware building block (layer 1) resp. a procedure (layer 2) exists. According to the result of this check, the process proceeds as follows:

- If a building block realizing the security requirements exists (layer 1), the process finishes with a positive output for the particular security requirement.
- If not a building block but a procedure exists (layer 2), one has to proceed to phase 4 checking if this procedure can be realized.
- If neither a building block nor a procedure exists for a security requirement, one checks if a corresponding case study is available in the repository at layer 3 and modifies the perspectives of the business process accordingly. If a case study fits, the perspectives of the business process model must be adopted to fulfill it and one may proceed to phase 3. Otherwise, a security expert is notified in order to obtain or create a case study and to include it into the repository at layer 3. If this is not possible, the domain expert is informed. He must decide if the security requirements of the business process can be reduced. If a reduction is not acceptable, the business process cannot be executed. If the domain expert modifies the security requirements, the MoSS_{BP}-process has to step back to phase 1 performing the semantic check (layer 4).

Phase 3: In this phase, one checks if each security activity and each basic security element of a selected or newly developed case study is realized by a soft- or hardware building block (layer 1) or a procedure (layer 2).

- If a soft- or hardware building block exists, we can terminate the process for the particular element.
- If no building block but a procedure exists, we can continue the process with phase 4.
- If neither a soft- or hardware building block nor a procedure exists, the security expert is informed. He tries to procure or develop corresponding soft- or hardware building blocks realizing the particular element of the case study. If this fails, he tries to create or procure an adequate procedure which is added to the repository at layer 2. If that is also not possible, he notifies the domain expert who similarly to phase 2 either relaxes the security requirements and steps back to phase 1 or gives up the business process.

Phase 4: If a procedure is used to fulfill the security requirements or a security activity resp. a basic security element of a case study, one has to check which soft- and hardware building blocks are needed to realize the procedure. If not all building

blocks are available at layer 1, the security expert is notified. If he cannot add them, the domain expert again has to relax the security activities or give up the business process.

By our proceeding, expensive security experts have to be consulted only if the repository does not contain a sufficient solution for a particular security problem. Moreover, any new development or procurement by a security expert will be added to the repositories and leads to a growing set of building blocks, procedures, and case studies. Thus, for a particular domain of security processes, the necessity to fall back on security experts will decrease in time.

To support the modeling of business processes in UML, a graphical syntax editor seems to be necessary to enable syntactical and semantical correct business process specifications (layer 4). This editor, moreover, should help to identify relevant security requirements. Furthermore, the modification process of layer 3 should also be facilitated by a tool since due to the large spectrum of different business processes one needs a confusing high number of different reference models and case studies. For these two tasks a security analysis tool seems helpful since it enables the highly automated modification of business processes by integration of suitable basic security elements and security activities as well as the realizing of this elements by basic building blocks. Therefore we extended MoSS$_{BP}$ by an adaption of the object-oriented security analysis tool SEMBA introduced in Section 5 which can be used as a graphical syntax editor and for the security analysis of business processes.

5. Object-oriented security analysis of business processes

The security analysis of IT systems is standardized by ISO/IEC in the so-called set of *Common Criteria* (CC, cf. [34]) providing a methodology for vulnerability detection, risk assessment, and countermeasure integration. The terminology with respect to security issues in the CC is more technical than those used in MoSS$_{BP}$ to describe security purposes of business processes. Therefore, in this section we give a short introduction to the CC and its terminology to support the understanding of the approach for more technical oriented readers. Moreover, the relationship between the CC and the business oriented terminology of MoSS$_{BP}$ is mentioned, too.

Figure 4 delineates the main security classes and associations defined by the CC. The security relevant parts of a system are assets for their owners which, unfortunately, are constantly exposed to threats by intruders, called threat agents, who exploit the vulnerabilities of the assets for attacks. Therefore, the assets underly security risks. In order to minimize these risks, the asset owners impose countermeasures reducing the vulnerabilities of the assets. In our context, the security objects (i.e., business process elements) correspond with the assets while the security subjects describe the intruders attacking an asset.

Our object-oriented approach [26, 27] facilitates the design of CC-compliant business models by providing a library of basic asset classes like networks, stations, applications, and data as well as associations between the classes. Moreover, more specialized classes are inherited from the basic classes in order to support modeling of business processes. We designed classes specifying the activities, agents, and artifacts

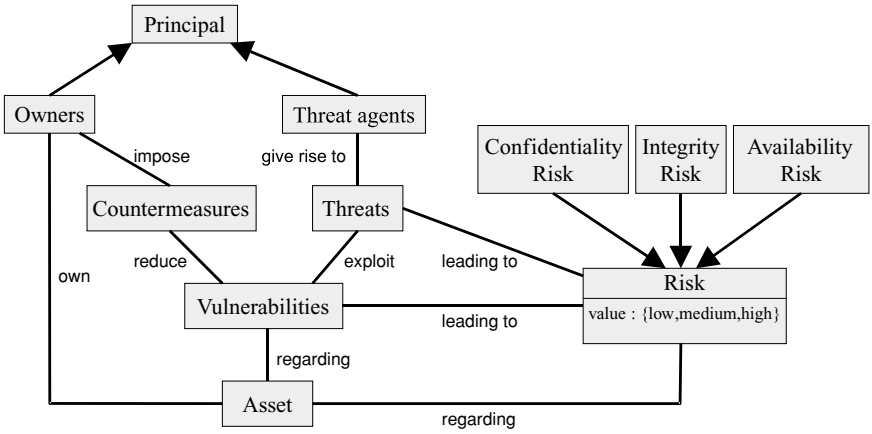

Fig. 4 CC security classes

(cf. Section 3) participating in a business process. Utilizing the class libraries, our tool SEMBA based on the toolset ARGO [59] facilitates the modeling of business processes and sub-processes in the form of UML object diagrams (cf. [6]).

Class attributes are used to describe the amount of protection, a business process needs to fulfill a certain security requirement. Each class contains attributes for all security requirements relevant for the modeled business process element (cf. Table 1). While there are various methods to describe the amount of protection for an asset, our approach refers to the seven security levels corresponding to the evaluation assurance levels defined in Part 3 of the CC [34]. For instance, level 7 shall be assigned to the legal binding property of a contract if a breaking of this contract without successful legal action leads to total collapse of the institution.

According to the CC, in the next analysis phase vulnerabilities and threats on the assets are identified. Furthermore, one has to estimate the seriousness of the vulnerabilities (i.e., the likelihood that they are in fact exploited to attack an asset). This seriousness, of course, depends on the safeguards used to protect the security requirement. For instance, a successful appeal against the repudiation of a contract is more likely if the subscription was witnessed by a notary. The seriousness is modeled by a class attribute (threat seriousness level) which, similarly to the security levels of the assets, may contain seven values.

Vulnerability and threat identification, however, tends to be laborious and complicated and therefore is not well suited to domain experts with limited knowledge on security. Therefore and in order to be consistent with MoSS_{BP}, we altered the procedure in this place. Instead of adding vulnerabilities and threats, the domain expert may identify security requirements and assign them to the business process elements. The security requirements are also modeled as classes and instances of these classes are added to the business process model. The tool can support the analysis process by suggesting useful security requirements itself. Here, we apply a graph rewrite system which modifies the UML object diagram by adding security requirement objects. The graph rewrite system consists of graph rewrite rules (cf. [2]) each consisting of a pre-pattern, a post-pattern, an application condition, and an effect function. The

pre-pattern contains a UML diagram describing a business process sub-system which, to be secure, requires a certain security requirement. The post-pattern describes the sub-system extended by an object modeling the security requirement and edges linking the new object with certain sub-system objects. Thus, by executing the rule, the security requirement object is added to each part of the UML model corresponding to the pre-pattern. The application condition may be used to restrict the execution of a rule to certain object attribute settings while by the effect function attributes may be altered. The security requirement objects are also provided with seriousness levels describing the likelihood of a successful attack on the system spoiling the corresponding requirement.

As an example, we will sketch the rule adding a security requirement object *authenticity* to an object of the class *information* or a derived class. The pre-pattern of this rule consists of an information object which is linked by an object of the class *authenticity*. The authenticity object, however, is marked as inhibitory stating that the graph rewrite rule must only be executed for information objects not yet provided by an authenticity object. The corresponding post-pattern consists also of the two objects but without any inhibitory marks. Thus, by executing this rule on a model of a business process, objects of class *information* or derived classes are provided by exactly one security requirement object of class *authenticity* each.

After introducing security requirements, another graph rewrite system is used for determining the risks on the assets. For each pair of a business process element and a security requirement a risk object is created stating the risk that the business process element may not be used correctly due to a violation of the corresponding security requirement. Moreover, the tool calculates the risk level which is modeled by a class attribute, too. According to Courtney [15], the risk level depends on the security level of the asset and on the seriousness level of the security requirement. Currently, we apply the matrix[4] in Table 2 reflecting that the risk level depends on the average of the security level and the seriousness level as intimated by Courtney. Of course, the exact mode to evaluate these two levels for computing the seriousness of risk is worth discussing and should be a part of an enterprise's security policy. The domain expert has to assess the risks which is also supported by a graph rewrite system. If all risks are bearable, the security analysis can be terminated at this place.

Table 2 Matrix to calculating risk values

Security level	Threat seriousness level						
	1	2	3	4	5	6	7
1	0	0	1	1	2	3	3
2	0	1	1	2	3	3	4
3	1	1	2	3	3	4	5
4	1	2	3	3	4	5	5
5	2	3	3	4	5	5	6
6	3	3	4	5	5	6	7
7	3	4	5	5	6	7	7

[4] The risk level 0 states that no risk is assumed and the risk object is removed.

If the risks for the business process cannot be accepted, the security analysis proceeds to the safeguard assignment phase in order to enforce the security requirements and, in consequence, to reduce the risks. The countermeasures are defined in another class library and may be added to the model. Since countermeasures may contain vulnerabilities themselves, the analysis iterates the vulnerability and threat identification phase as well as the risk evaluation phase. If the newly calculated risks can be accepted as bearable, the analysis terminates. Otherwise, new countermeasures are suggested and further iterations take place. After terminating the security analysis a real business process can be modified based on the resulting UML model.

6. Example

Business-to-business transactions between companies by means of electronic procurement (e-procurement) get more and more popular and, meanwhile, standards for the transactions exist. For instance, the OBI consortium issued a set of specifications for **O**pen **B**uying on the **I**nternet (OBI, [41]). In this standard, an architecture for electronic procurement of goods and a corresponding business-to-business model are defined. The architecture introduces a buying organization, selling organizations, a payment authority, and a requisitioner. In behalf of the buying organization, the requisitioner carries out orders at the selling organizations and the orders are paid by means of the payment authority. An ordering process starts when the buying organization decides to procure a certain good. At first, it provides the requisitioner with seller addresses. Thereafter the requisitioner sends requests for tenders to the sellers, receives tenders, decides about a winning seller based on the tenders, and sends an order to the winning seller. Afterwards the order is fulfilled and paid by means of the paying authority. The tenders and orders are compatible to the EDI standard [17].

As an example for the MoSS$_{BP}$-support including a SEMBA-based security analysis, we concentrate on the request for and the delivery of tenders which are realized by a partial business process. SEMBA supports the design process by a library of process element classes and edges to link the class instances. In the first step of the corresponding MoSS$_{BP}$-based process, the domain expert creates a UML activity diagram modeling the partial business process as depicted in Fig. 5. The swimlanes describe the departments which are responsible for the various activities. The requisitioner is an employee of the purchase department which is responsible for the tender handling. In later steps of the e-procurement, the dispatch department will be involved in the

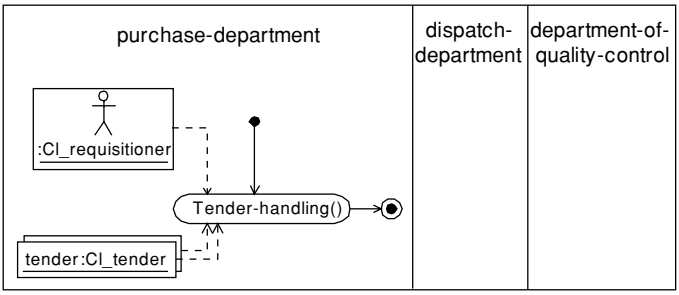

Fig. 5 Tender handling process

reception of the procured goods whereas the department of quality control will control their quality. In our example, we use three activity objects instantiated from the classes `Cl_process-start`, `Cl_tender-handling`, and `Cl_process-end`. The objects are linked by two solid arrows describing the order of the activities. The unnamed object of class `Cl_requisitioner` models the requisitioner who executes the activity `Tender-handling`. It is linked with the activity by a broken arrow of class `execute`. Moreover, the business process deals with tenders which are represented by artifact objects of the class `CL_tender`. The corresponding broken arrows depict that the tenders are used in the activity `Tender-handling`. For simplicity, we model only two tenders being sent from two different selling organizations.

In the following subsections, we will show the integration of new mechanisms into the tender handling process in order to guarantee that the requisitioner may carry out orders without disclosing the identity of his selling organization and himself to third parties. This will be achieved by a MoSS$_{BP}$-based analysis of the business process for the security requirement *anonymity* and the corresponding integration of suitable safeguards. This is outlined in the next three subsections. Moreover, we will sketch the analysis of the business process for the *authenticity* of the tenders.

6.1. Phase 1

The domain expert decides that anonymity is important for the business process and decides for simplicity to add the security requirement object of type *anonymity* to the model and to link it with the activity `Tender-handling` without applying SEMBA. This link, however, is incorrect (cf. row 'anonymity' and column 'activity' in Table 1) which is detected by a semantic check providing the domain expert with the following information:

- Explanation of the term *anonymity*,
- information, for which business process elements a correlation with anonymity is possible,
- information, for which security requirements a correlation to the business process element *activity* is possible.

The domain expert recognizes that *anonymity of the agent* is what he really wants and models the process accordingly. Figure 6 depicts the business process model after the assignment of the security requirement *anonymity*. The corresponding security

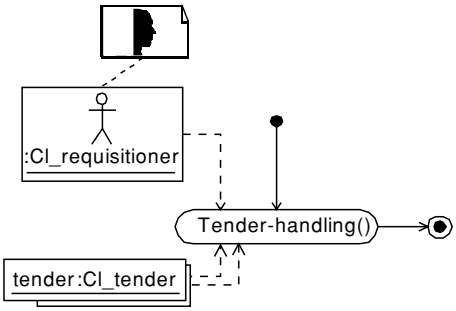

Fig. 6 Secure tender handling process (1)

requirement object is represented by a rectangle with a dog-eard corner including the icon representing the specific security requirement. The silhouette head represents *anonymity*. Again a semantic check is carried out and the tool notifies the domain expert that more information how to relate the security requirement *anonymity* of the agent to the objects of the business process is needed (cf. Section 3):

- Against whom anonymity of the agent is required,
- in which actions anonymity of the agent is required,
- for which information, end product, procedure, resp. information flow in the business process actions anonymity of the agent is required.

The domain expert decides to assign the following objects to the *anonymity* of the agent:

- Anonymity against the seller,
- anonymity based on the activity `Tender-handling`,
- anonymity based on the whole information transmitted between the buying and the selling organization during the execution of the activity `Tender-handling`. Information from the buyer to the seller is sent within tender requests while information on the way back is transferred within the tenders.

Unfortunately, in the current model representations of the seller and the request for tenders are still missing. To overcome this, the domain expert adds an *agent* object of the class `Cl_seller` modeling the sellers and the *information* object `customer_request` from the class `Cl_request` describing a tender request. Furthermore, he adds links describing the various aspects of the security requirement *anonymity*. In the model[5] in Fig. 7 these links are specified by thick broken arrows.

After the integration of the security requirement *anonymity* to the tender handling process, the domain expert has to rate the security level for the anonymity of the business process (cf. Section 5). The anonymity of the buying organization is of relative high relevance since a selling organization might use illegal methods like bribery of buying organization members to gain an order if it knows their identities. Moreover, an intruder disclosing the buyer's identity by eavesdropping request for tenders might supply competing selling organizations with information which worsens the business relation of these companies with the buying organization. As a consequence of these reflections, the domain expert assigns the security level 5. Additionally, the seriousness of a successful attack on the security requirement *anonymity* has to be determined. To provide this, the tool support SEMBA includes special policies. In the current case, the seriousness of anonymity attacks without any countermeasures is estimated as high, since the identity of the requisitioner is included as a sender address in the text message and may as well be contained in the request data. Therefore, SEMBA assigns the seriousness level 6 to the corresponding security requirement object.

As outlined in Section 5, we can also use the graph rewriting capability of SEMBA to determine security requirements. Here, in a first step SEMBA adds security requirement objects correlated with a business process element (see Table 1). In a second

[5] For the sake of clarity only one `tender`, one `customer_request`, and one unnamed object of class `Cl_seller` are represented while the passport symbol connected with tender is added later.

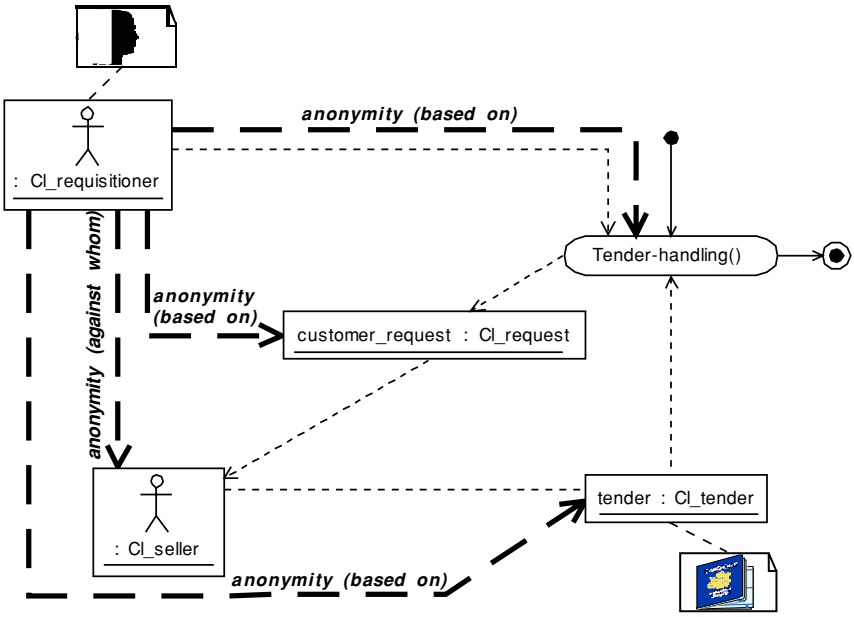

Fig. 7 Secure tender handling process (2)

step, the domain expert decides about accepting or rejecting security requirements and SEMBA removes the rejected elements.

In this example, we will look on providing security requirement objects of the business process element class *information* (i.e., the objects `tender` and `customer_request`). According to Table 1, the two process elements are linked with the security requirement objects *confidentiality, integrity, availability, privacy, legal binding, mutual dependency, authenticity, originality, right to use,* and *copyright*. Each addition of a security requirement object to the tender (and other *information* objects) is performed by a separate graph rewrite rule.

Afterwards, the domain expert decides for each security object which security requirements are necessary. Here, we assume that he selects for the tender objects only the security requirement *authenticity* guaranteeing that the tenders are authentic. This security requirement, however, is not seen as relevant for the `customer_request` objects since the requests are issued by the own buyer organization. Moreover, besides of the already selected security requirement *anonymity* for the requisitioner, all other requirements are discarded and one receives the model depicted in Fig. 7. Here, the security requirement object of type *authenticity* is modeled by the passport symbol.

In the next step the authenticity security level of the business process and the seriousness level of the security requirement have to be selected. Since not authentic orders may spoil the ordering decision and therefore damage the buying organization significantly, the value 6 is assigned to the security level. Moreover, without any safeguard it is not too difficult for an intruder to fake tenders if the request for tenders was previously wiretapped and SEMBA selects the seriousness level 4.

Thereafter, based on the selected security and seriousness levels, SEMBA creates objects describing the risks that the anonymity of the requisitioner resp. the authenticity

of the tender are successfully violated. According to Table 2 the risk levels are set to the values 5 for both the *anonymity* and the *authenticity* risks. The domain expert decides that these risks are too high to be accepted. Thus, one has to modify the business process adding safeguards in order to guarantee anonymity of the requisitioner and authenticity of the tenders. The steps adding the safeguards are outlined in the following.

6.2. Phase 2 of anonymity

At first, the domain expert checks if for the unnamed class `CL_requisitioner` each security object of the tender handling process a soft- or hardware building block (layer 1), a corresponding procedure (layer 2), or a case study (layer 3) realizing the security requirement *anonymity* exists. Assuming that the MoSS$_{BP}$-repositories do not contain suitable solutions, the domain expert has to consult a security expert in order to create a case study for anonymous electronic tender handling. Before the security expert can decide if it is possible to develop a case study and on which techniques the case study should be based, he has to look on the relations between the involved objects since these relations may be the reason for vulnerabilities leading to the disclosure of identities.

Sources for vulnerabilities of an agent's anonymity are information units, the agent is dealing with, as well as the actions he is performing. Information usually contains a direct link with the identity of the information creator which can be utilized to disclose him. Actions involving the agent's environment (i.e., interactions) always take place by participating in a communication (i.e., by sending or receiving messages). Here, in particular the transmitter and receiver addresses can be exploited for attacks against an agent's anonymity. Moreover, the user data may contain so called documentation data (e.g., a declaration who is the creator of the user data) which is also a target for an attack.

The various relations between security objects cause a whole spectrum of different anonymity attacks. Gavish and Gerdes (cf. [21]) differentiate three anonymity types which have to be considered in order to prevent disclosures:

Environmental anonymity: A person, who wants to act anonymously, must not be disclosed by people of his direct environment. In our example, the buying organization has to guarantee that a potential intruder—which could also be an employee—must not be able to observe the activities of the purchase department.

Content-based anonymity: A user acting anonymously has not to disclose deliberately or inadvertently his own identity in user data created by himself. Therefore the identity of the buying organization must not be included in the user and documentation data of a request for tenders. Moreover, it has to be impossible that the identity can be deduced from the user or documentation data.

Procedural anonymity: If a user who wants to act anonymously, communicates electronically, the telecommunication protocol realizing the communication has not to contain identity information in the protocol control information. Thus, the protocol realizing the request for tenders (i.e., the HTTP-protocol according to the OBI-standard [41]) must not include data allowing the deduction of the buying organization's identity.

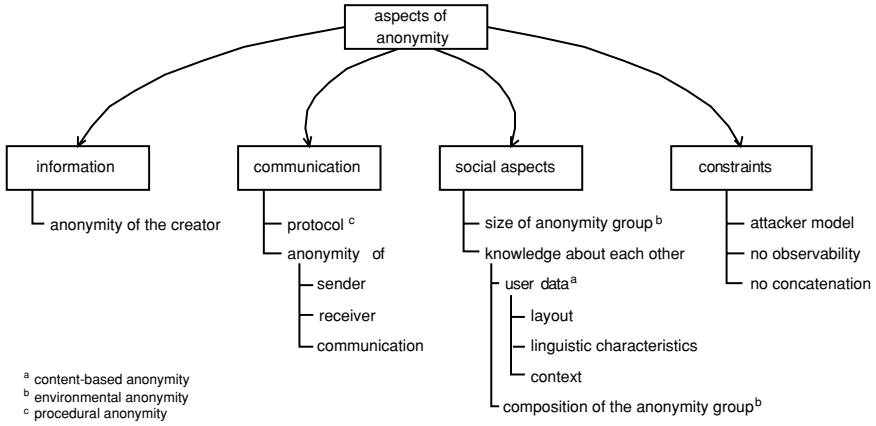

Fig. 8 Aspects of anonymity (taken from [52])

Another classification of the anonymity aspects was provided by Rubert [52]. As sketched in Fig. 8, like Gavish and Gerdes this model distinguishes between information and communication aspects. Moreover, it refers to social aspects which are based on so-called anonymity groups. An anonymity group hides an anonymously acting agent in a group of other agents. Intruders from outside the group must not be able to recognize which agent of the group is executing a certain action. Finally, the classification lists constraints which may influence the seriousness of the attacks. The constraints depend on an attacker model specifying the characteristics of a potential attacker (e.g., his computer capacity).

In our tender-handling example, the security expert decides in agreement with the domain expert that he mainly protects procedural anonymity of tender requests. In particular, he considers the following basic concepts to protect an agent's anonymity:

- Encryption of messages or message headers,
- removal of sender identifications,
- randomized transmissions which can be used to prevent traffic analysis of communications by comparing the lengths of messages or checking chronological orders,
- creation of dummy messages in order to prevent traffic analysis, too,
- broadcasting of messages.

Techniques to protect anonymity are based on these basic concepts. Below, we will outline some techniques to guarantee anonymity (cf., e.g., [13, 18, 45, 46, 48]):

Techniques to protect the receiver anonymity: Here, we can apply broadcasting of messages which do not contain the receiver's identity. This method, however, is not scalable to large numbers of potential receivers (e.g., the Internet). Nevertheless, it can be used for relatively small anonymity groups where the message is sent to the group and a copy is delivered to every group member.

Techniques to protect the sender anonymity: In this case, we can use the following basic concepts:

- By traffic analysis a third party may eavesdrop the amount of traffic transmitted between various parties. The results of a traffic analysis may be exploited by a statistical analysis guessing the transmitter of a certain message successfully. A means to prevent traffic analysis is the use of dummy messages distorting the statistical analysis.
- Removal of the sender identification which may be performed by so-called anonymity servers. The sender transmits a message to the anonymity server which forwards it without the sender identity to the receiver. Moreover, we call special anonymity servers which are able to send the replies to the original sender, as mediators.
- Encryption of messages in order to prevent man-in-the-middle attacks on the sender identity.

Techniques to protect the anonymity of the interconnection: In this case, only the communication partners should know the existence of a data exchange between the communication partners. To realize this technique, one has to combine various basic concepts:

- The sender identification has to be removed since, otherwise, the identity of the sender may be disclosed using protocol control information during the transmission.
- To prevent the disclosure of the sender identity by scrutinizing the user data, these data have to be sent encrypted to their receiver.
- To prevent attacks by performing a lexical analysis, the message has to be encrypted on the section between the sender and the anonymity server and also between the last router of the route and the receiver.
- A traffic analysis by observing the incoming and outgoing links of a router resp. anonymity server which can be used to conclude the whole route of a message can be prevented either by dummy messages or by randomized transmission.

Examples for techniques applying the listed methods in order to realize the anonymity of transmitted data comprise MIXes [46] and Onion-Routing [56].

Unfortunately, the techniques outlined above do not address disclosures based on content-based anonymity. Thus, the security expert cannot prevent that the buying organization (i.e., the requisitioner) discloses the own identity by including vulnerable data in requests for tenders. For instance, user data created by a word processing tool may contain additional information describing the creator of the data. If it is possible, the administrator of the word processor should turn off such features. Therefore, the example analysis leads also to modifications of the software installation processes in a company.

In our example, the anonymity of the buying organization against the selling organization in a request for tenders is based on the data transmitted between the requisitioner and the sellers in the activity modeled by the object `Tender-handling`. The transferred information is described by the information objects `customer_request` and `tender`.

In order to decide which techniques should be used to protect the buying organization's anonymity, an attacker model is necessary. As outlined above, both the receiver of a request for tenders and third parties wiretapping the interconnection links might gain by disclosing the identity of the buying organization. The domain

and security experts, however, agree that the likelihood of anonymity attacks by wiretapping messages is very low. Therefore, only countermeasures to protect the anonymity of the buyer against analysis of received tender requests are installed. In consequence, the case study developed by the security expert comprises the following countermeasures:

1. The requisitioner has to prevent hints of the buying organization's identity in the user and documentation data of the requests for tenders. Since for this problem no technical support is available, the system administration process has to be modified accordingly.
2. The sender identification has to be removed from the protocol control information by an anonymity server. Since the requisitioner needs to receive the answers of his request, a trustworthy mediator must be consulted who removes the identification data of the buying organization from the protocol control information of the tender request messages. Moreover, the mediator forwards the received tenders to the buying organization.

Figure 9 depicts the functional perspective of the adapted case study. If no trustworthy mediator is yet known, a lookup of suitable mediators is started and the access information of the detected mediators is stored in a list modeled by an unnamed object of class `List_mediator-anonymSenderBi`. If the requisitioner wants to request a tender, he selects a mediator, which is specified by the unnamed object of the class `Cl_mediator`, from the list and transmits the request for tenders to this mediator. The mediator removes the transmitter identification and forwards a request for tenders to the receiver. Moreover, the replied tenders are sent to the mediator which forwards it to the requisitioner.

After integrating the safeguards into the business process, the domain expert repeats the security requirement evaluation for the extended model. The application of a mediator removing the identification data and forwarding the received tenders reduces the danger of successful attacks on the anonymity of the buying organization. SEMBA

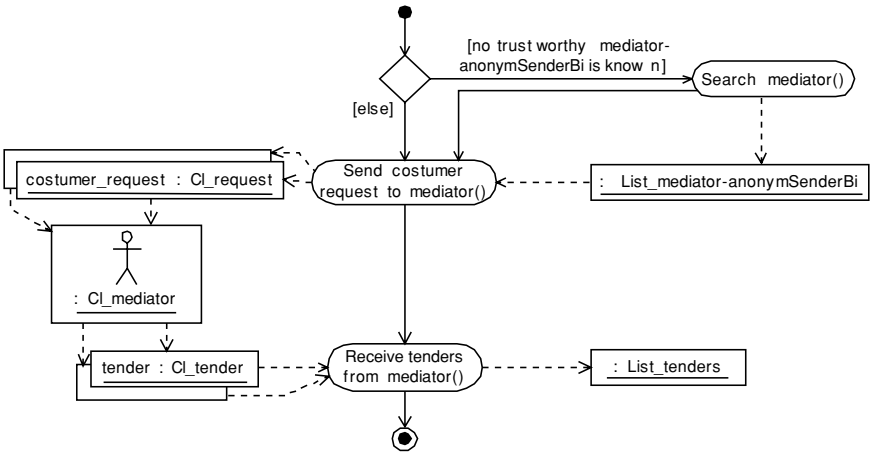

Fig. 9 Case study 'mediator'

enforces a policy stating that the applied safeguards reduce the seriousness level for the unnamed object of class `Cl_requisitioner` to 3 reflecting that it does not solve content-based anonymity. Since the object has the security level 5, the new risk of anonymity violations is set to 3 according to Table 2. The domain expert accepts this low to intermediate risk as bearable and terminates the security analysis, at this point.

6.3. Phases 3 and 4 of anonymity

In phase 3, the domain expert has to check for each security activity and for each basic security element of the case study if a soft- or hardware building block (layer 1) or a corresponding procedure (layer 2) exists. The case study of the security requirement *anonymity* does not contain basic security elements and security-relevant activities (i.e., the two activities for sending requests for tenders and for receiving tenders do not implement security elements to be realized by certain building blocks). Therefore the domain expert does not need to take further action in phase 3.

Since we do not use procedures in phase 3, we can omit phase 4. Thus, by realizing the case study developed in phase 2 and applying the building block created in phase 3, we can modify the tender request process in order to fulfill the desired security requirement *anonymity*.

6.4. Phases 2 to 4 of authenticity of information

In this subsection the processing for the security requirement *authenticity* of the objects of class `Cl_tender` is shortly described. To ensure that a security object of the type information (cf. Section 3) is what it pretends to be, nobody must be able to modify the object after its creation. A useful method to guarantee authenticity is public key cryptography. In our example, the tender has to be digitally signed with the private key of the seller. If the check of the digital signature fails, the tender is not authentic

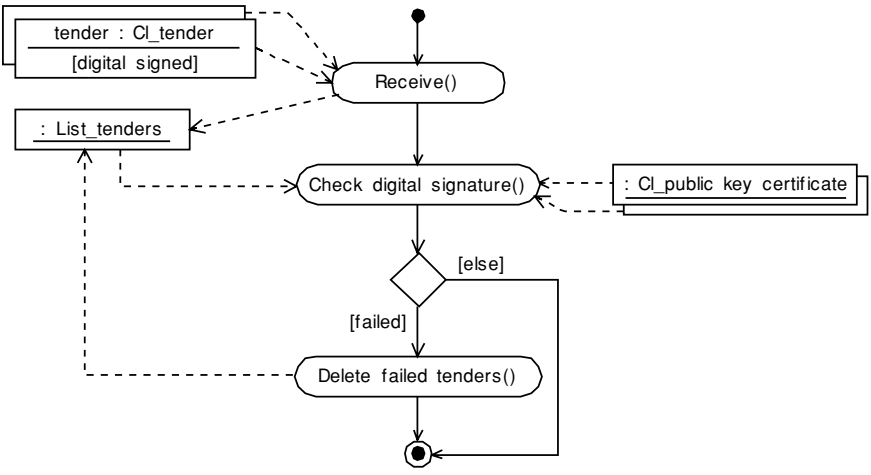

Fig. 10 Case study 'authenticity of a document'

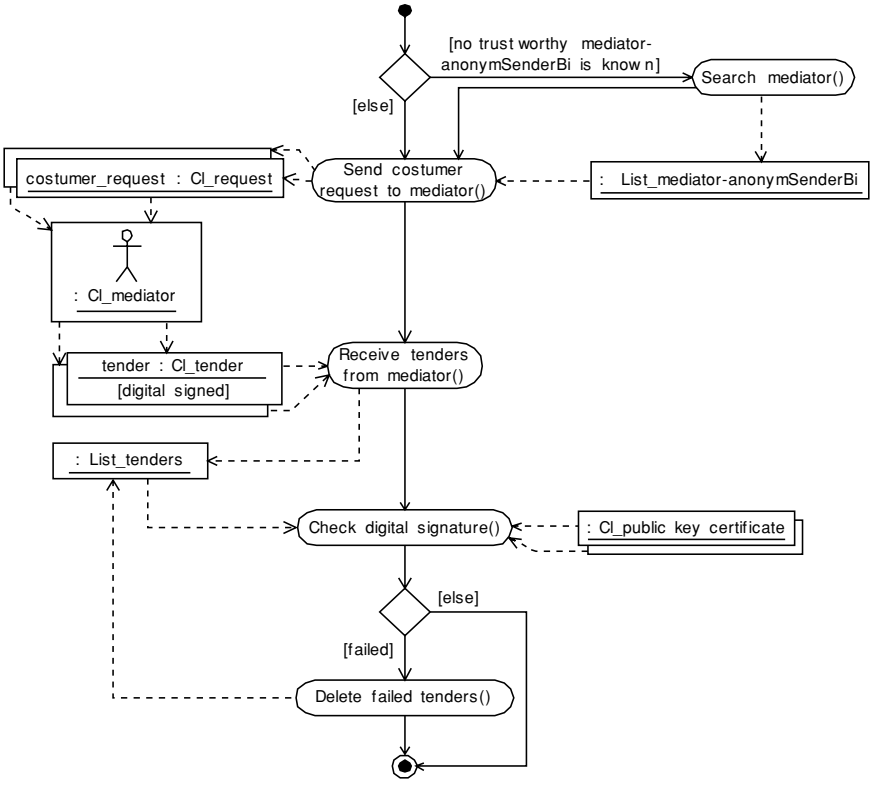

Fig. 11 Secure tender handling process (3)

and must be removed by the requisitioner. In the repository of layer 3 an adequate case study is stored modifying the following perspectives:

- In the *functional perspective* an activity `Check digital signature` of the relevant information and an activity `Delete failed tenders` are included, which will be executed if the check fails.
- The *informational perspective* contains an unnamed object of the class `Cl_public key certificate`.

In the repository of layer 2 an ALMO$T procedure (cf. [49]) is stored specifying how to realize the activity `Check digital signature`. In this procedure among other things the construct `RSA.decrypt(pKeyring.get(seller).tender)` is used. It guarantees that the document `tender` has to be decrypted by the method `RSA` using the public key of `seller` which is stored in `pKeyring`. Appropriate software tools to realize the decryption of a document using the method RSA are offered in the repository of layer 1.

Figure 10 shows the case study describing how to modify the functional perspective of the tender handling process according to the security requirement *authenticity* of the tenders. Figure 11 depicts the modified functional perspective of the tender handling process regarding the security requirements *anonymity* and *authenticity*. After sending

the costumer request to potential sellers, receiving their tenders via a mediator, and checking the digital signatures of the tenders, the tenders are evaluated and a winning seller is chosen (activity `Decision about tenders`). This safeguard protects the authenticity of the tenders very well and the resulting seriousness level of the security requirement is set to 1. In consequence, the risk level for successful attacks on the authenticity of tenders will be computed as 3 which is accepted by the domain expert.

7. Conclusion

In this paper, we introduced the MoSS$_{BP}$-framework supporting domain experts to define security requirements for business processes and to modify business processes in order to guarantee the requirements. The modeling of business processes, the identification of suitable security requirements, and the introduction of safeguards enforcing the requirements are supported by the object-oriented modeling tool SEMBA.

Currently, the graph-rewriting feature of SEMBA supports the selection and correct integration of security requirements into a business process model, the computation of seriousness levels, the addition of risk objects to a model, and the calculation of risk levels. In other application domains (cf. [26, 28]), we also use graph-rewriting for the selection and evaluation of safeguards. In particular, graph rewrite systems are used to introduce the objects modeling countermeasures to a system model. These objects contain attributes describing a protection level and the estimated costs of imposing the specified countermeasure. In a first step, this additional feature of SEMBA suggests for each pair of a system element (e.g., in our domain a business process element) and a risk object all countermeasures with a sufficient protection level (i.e., the protection level must be equal or higher than the risk level). Thereafter SEMBA compares the costs of the countermeasures and selects one with a good relation between costs and the level of protection. This feature can also be adapted to MoSS$_{BP}$ where the retrieval of the repositories for building blocks, procedures, and case studies may be automated.

Moreover, at the moment SEMBA models only functional aspects of business processes. A complete description of a business process, however, considers also at least two other perspectives (cf. [51]): At first, the *informational perspective* represents the information entities, their structure, and relationships between them. At second, the *organizational perspective* depicts in which place of an enterprise and by which agents activities are performed. For the selection of modifications in layer 3 of the MoSS$_{BP}$-architecture one has often to consider these perspectives as well. For instance, the modifications in Section 6.2 can only take place if mediators to remove sender identifications are known. The existence of mediators, however, can only be detected by considering the informational perspective. Furthermore the list of mediators (`List_mediator-anonymSenderBi`) must be administrated and an agent must be allowed to do this. This implies the necessity to modify the organizational perspective. For these reasons the informational and organizational perspectives must be modified accordingly. Therefore we plan to extend SEMBA further in order to support also UML diagrams modeling informational and organizational aspects. Moreover, the graph rewrite rules shall be extended in order to modify different diagrams simultaneously.

References

1. Atluri, V., Huang, W.-K., & Bertino, E. (1997). An execution model for multilevel secure workflows. In *Proceedings of the IFIP 11.3 Workshop on Database Security*.
2. Bardohl, R., Taentzer, G., Minas, M., & Schürr, A. (1999). Application of graph transformation to visual languages. In *Handbook on Graph Grammars and Computing by Graph Transformation, Volume 2: Applications, Languages and Tools*, Chapter 1. World Scientific.
3. Baskerville, R. (1988). *Designing Information Systems Security.* Chichester: Wiley & Sons.
4. Baskerville, R. (1993). Information systems design methods: Implications for information systems development. *ACM Computing Surveys, 25*(4), 375–414.
5. Bertino, E., Ferrari, E., & Atluri, V. (1997). A flexible model supporting the specification and enforcement of role-based authorizations in workflow management systems. In *Proceedings of the 2nd ACM Workshop on Role-Based Access Control*.
6. Booch, G., Rumbaugh, J., & Jacobson, I. (1999). *The unified modeling language user guide*. Addison-Wesley Longman.
7. Browne, P. (1979). *Security: Checklist for computer center self-audits*. Arlington: AFIPS Press.
8. Bui, T., & Sivasankaran, T. (1987). Cost-Effectiveness Modeling for a Decision Support System in Computer Security. *Computer Security, 6*(2), 139–151.
9. Bußler, C. (1995). Access control in workflow management systems. In *Proceedings of the IT Security'94 Conference* (pp. 165–179), Oldenbourg-Verlag Munich.
10. Carroll, J., & Maclver, W. (1984). Towards an expert system for computer facility certification. In *Computer Security A Global Challenge*, (pp. 293–306). Amsterdam: North-Holland
11. CCTA. (1991). SSADM-CRAMM, *Subject guide for SSADM version 3 and CRAMM version 2*. London: CCTA.
12. Chisnall, W. R. (1997). Applying risk analysis methods to university systems. In *Proceedings of the EUNIS 97 Congress*, Grenoble.
13. Clarke, R. (1999). Identified, anonymous and pseudonymous transactions: The spectrum of choice. In *IFIP WG 8.5/9.6 Working Concerence on User Identification & Privacy Protection*, Stockholm.
14. Computer Security Consultants, Ridgefield. (1988). *Using decision analysis to estimate computer security risk*.
15. Courtney, R. (1977). Security risk assessment in electronic data processing. In *AFIPS Conference Proceedings of the National Computer Conference 46* (pp. 97–104). Arlington:AFIPS.
16. Curtis, B., Kellner, M.I., & Over, J. (1992). Process modeling. *Communications of the ACM, 35*(9), 75–90.
17. Data Interchange Standards Association. (2001). *X12 Standard*, release 4050 edition, December.
18. Demuth, T., & Rieke, A. (2003). Bilateral anonymity and prevention of abusing logged web addresses. In *2000 Military Communications International Symposium*, Los Angeles.
19. Finne, T. (1996). Computer support for information security analysis in a small business environment. In Jan. H.P. Eloff, (Ed.), *Proceedings of the IFIP TC11 WG 11.2 on small systems security*, (pp. 73–88), Samos.
20. Fisher, R. (1984). *Information Systems Security*. Englewood Cliffs: Prentice-Hall.
21. Gavish, B., & Gerdes, J. (1998). Anonymous mechanisms in group decision support systems communication. *Decision Support Systems, 23*(4), 297–328.
22. Guarro, S. (1987). Principles and Procedures of the LRAM Approach to Information Systems Risk Analysis and Management. *Computer Security, 6*(6), 493–504.
23. Herrmann, G. (2002) *VerläUßlichkeit von GeschäUftsprozessen—Konzeptionelle Modellbildung und Realisierungsrahmen*. Logos Verlag, Published Version of Doctoral Thesis. In German.
24. Herrmann, G., & Pernul, G. (1998). Towards security semantics in workflow management. In *Proceedings of the 31st Annual Hawaii International Conference on System Sciences (HICSS-31)*. IEEE Computer Society Press.
25. Herrmann, G., & Pernul, G. (1999). Viewing business process security from different perspectives. *International Journal of Electronic Commerce, 3*(3), 89–103.
26. Herrmann, P. (2001). Information flow analysis of component-structured applications. In *Proceedings of the 17th Annual Computer Security Applications Conference (ACSAC'2001)* (pp. 45–54). New Orleans: ACM SIGSAC, IEEE Computer Society Press.
27. Herrmann, P., & Krumm, H. (2001). Object-oriented security analysis and modeling. In *Proceedings of the 9th International Conference on Telecommunication Systems—Modelling and Analysis* (pp. 21–32). Dallas: ATSMA, IFIP.

28. Herrmann, P., Wiebusch, L., & Krumm, H. (2001). Tool-assisted security assessment of distributed applications. In *Proceedings of the 3rd IFIP WG 6.1 International Working Conference on Distributed Applications and Interoperable Systems (DAIS 2001)* (pp. 289–294). Krakow: Kluwer.
29. Hoffman, L., Michelman, E., & Clements, D. (1978). SECURATE—Security evaluation and analysis using fuzzy metrics. In *AFIPS Conference Proceedings of the National Computer Conference 47* (pp. 531–540). Arlington. AFIPS.
30. Holbein, R., Teufel, S., & Bauknecht, K. (1996). The use of business process models for security design in organizations. In S. Katsikas & D. Gritzalis (Eds.). *Proceedings of the IFIP TC11 conference on information systems security* (pp. 13–22). London: Chapman & Hall.
31. Hoyt, D. (1973). *Computer security handbook.* New York: Macmillan.
32. Hudoklin, A., & Stadler, A. (1997). Security and Privacy of Electronic Commerce. In *Proceedings of the 10th International Bled Electronic Commerce Conference* (pp. 523–535). Moderna Organizacija.
33. Hung, P.C.K., & Karlapalem, K. (1997). A Paradigm for Security Enforcement in CapBasED-AMS. In *Proceedings of the 2nd IFCIS International Conference on Cooperative Information Systems (CoopIS'97)* (pp. 79–88).
34. ISO/IEC. (1998). *Common criteria for information technology security evaluation*. International Standard ISO/IEC 15408.
35. Kienzle, D.M., & Wulf, W.A. (1997). A Practical Approach to Security Assessment. In *Proceedings of the Workshop New Security Paradigms '97* (pp. 5–16). Lake District.
36. Krauss, L. (1972). *SAFE: Security audit and field evaluation for computer facilities and information systems*. New York: Amacon.
37. Kwok, L.F., & Longley, D. (1996). A security officer's workbench. *Computers & Security, 15*(8), 695–705.
38. Lacoste, G. (1995). *SEMPER: A Security Framework for the Global Electronic Marketplace* . SEMPER document 431LG042/Draft/25 August 1997/public.
39. Leiwo, J., Gamage, C., & Zheng, Y. (1998). Harmonizer—A Tool for Processing Information Security Requirements in Organization. In *Proceedings of the 3rd Nordic Workshop on Secure Computer Systems (NORDSEC'98)*, Trondheim.
40. Lund, M. S., den Braber, F., & Stølen, K. (2003). Maintaining Results from Security Assessments. In *Proceedings of the 7th European Conference on Software Maintenance and Reengineering (CSMR'2003)* (pp. 341–350). IEEE Computer Society Press.
41. OBI Consortium. (1999). *OBI Technical Specifications—Open Buying on the Internet*, draft release v2.1 edition.
42. Ozier, W. (1989). Risk Quantification Problems and Bayesian Decision Support System Solutions. *Information Age, 11*(4), 229–234.
43. Parker, D. (1981). *Computer security management*, Reston.
44. Pfitzmann, A. (1999). Technologies for Multilateral Security. In G. Müller, & K. Rannenberg, (Eds.), *Multilateral security in communications*, vol. 3: Technology, Infrastructure, Economy (pp. 85–91). Munich: Addison-Wesley.
45. Pfitzmann, A. & Köhntopp, M. (2001). Anonymity, Unobservability, and Pseudonymity—A Proposal for Terminology. In H. Federrath, (Ed.), *Anonymity 2000*, LNCS 2009, pages 1–9.
46. Pfitzmann, A., Pfitzmann, B., & Waidner, M. (1991). ISDN-MIXes: Untraceable Communication with Small Bandwidth Overhead. In *Kommunikation in Verteilten Systemen (KIVS'91)*, pages 451–463.
47. Quatrani, T. (2000). *Visual Modeling with Rational Rose 2000 and UML.* Addison-Wesley, 2 edition.
48. Roessler, T. (1999). Anonymization in data networks—extensive overview of anonymization services on the internet. In D. Fox, & H. Reimer, (Eds.), *Datenschutz und Datensicherheit 1999*. Vieweg.
49. Röhm, A., Herrmann, G., & Pernul, G. (1999). A Language for Modelling Secure Business Transactions. In *Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC'99)*, (pp. 22–31). IEEE Computer Society Press.
50. Röhm, A. & Pernul, G. (2000). COPS: A Model and Infrastructure for Secure and Fair Electronic Markets. *Decision Support Systems Journal, 29*(4), 343–355.
51. Röhm, A., Pernul, G., & Herrmann, G. (1998). Modelling Secure and Fair Electronic Commerce. In *Proceedings of the 14th Annual Computer Security Application Conference (ACSAC'98)*, (pp. 155–164). IEEE Computer Society Press.
52. Rubert, M. (1999). Anonymitat als Sicherheitsmerkmal von Geschäftsprozessen. Diploma thesis, Department of Business Administration, University of Essen. In German.
53. Shen, H., & Dewan, P. (1992). Access Control for Collaborative Environments. In *Proceedings of the CSCW'92 Conference*. ACM Press, New York.

54. Smith, S. & Lim, J. (1984). An Automated Method for Assessing the Effectiveness of Computer Security Safeguards. In *Computer Security A Global Challenge*, pages 321–328. North-Holland, Amsterdam.
55. Starke, G. (1994). Business Models and their Description. In G. Chroust & A. Benczur (Eds.), *Workflow Management: Challenges, Paradigms, and Products (CON'94)*, of *Schriftenreihe der österreichischen Computer Gesellschaft*, vol. 76, pages 134–147. Oldenbourg-Verlag Wien.
56. Syverson, P. F., Reed, M. G., & Goldschlag, D. M. (2000). Onion Routing Access Configurations. In *DISCEX 2000: Proceedings of the DARPA Information Survivability Conference and Exposition*, vol. 1, pages 34–40, Hilton Head, SC. IEEE Computer Society Press.
57. Thoben, W. (2000). *Wissensbasierte Bedrohungs-und Risikoanalyse Workflow-basierter Anwendungssysteme*. Reihe Wirtschaftsinformatik. B.G. Teubner-Verlag, Stuttgart. Published Version of Doctoral Thesis. In German.
58. Thomas, R. & Sandhu, R. (1997). Task-Based Authorization Controls (TBAC): A Family of Models for Active and Enterprise-Oriented Authorization Management. In *Proceedings of the IFIP WG11.3 Workshop on Database Security*. London: Chapman & Hall.
59. Tigris. (2000). *ArgoUML Vision*. argouml.tigris.org/vision.html.
60. Zviran, M., Hoge, J., & Micucci, V. (1990). SPAN—a DSS for Security Plan Analysis. *Computer Security, 9*(2), 153–160.

Peter Herrmann studied computer science at the University of Karlsruhe, Germany (diploma in 1990). Afterwards, he worked as a Ph.D. student (doctorate in 1997) and postdoctoral researcher in the Computer Networks and Distributed Systems Group of the Computer Science Department at the University of Dortmund, Germany. Since 2005 he is a full professor for formal methods at the Department for Telematics of the Norwegian University of Science and Technology (NTNU) in Trondheim, Norway. His research interests include the formal-based development of networked systems and the engineering of distributed services. Moreover, he is interested in security and trust aspects of component-structured distributed software.

Gaby Herrmann studied computer science at the University of Karlsruhe, Germany (diploma in 1991). Afterwards, she worked as a researcher in the Communication Group and the Information Systems Group at University of Duisburg-Essen (Doctorate in 2001, topic: security of business processes). Since 2000 she works as executive secretary at the Department of Economics, Business Studies and Computer Sciences at the same university.