Using SecureBPMN for Modelling Security-Aware Service Compositions

Achim D. Brucker

SAP SE, Vincenz-Priessnitz-Str. 1, 76131 Karlsruhe, Germany
achim.brucker@sap.com

Abstract. Today, many systems are built by orchestrating existing services, custom developed services, as well as interaction with users. These orchestrations, also called composition plans, are often described using high-level modelling languages that allow for simplifying 1) the implementation of systems by using generic execution engines and 2) the adaption of deployed systems to changing business needs. Thus, composition plans play an important role for both communicating business requirements between domain experts and system experts, and serving as a basis for the system implementation.
At the same time, ICT systems need to fulfil an increasing number of security and compliance requirements. Thus, there is a demand for integrating security and compliance requirements into composition plans.
We present SecureBPMN, a language for modelling security properties that can easily be integrated into languages used for describing service orchestrations. Moreover, we integrate SecureBPMN into BPMN and, thus, present a common language for describing service orchestration (in terms of business process models) together with their security and compliance requirements.
Keywords: SecureBPMN, BPMN, Access Control, Confidentiality.

1 Introduction

Today, many systems are built by orchestrating existing service offerings, custom developed services, as well as human-centred tasks. These orchestration models, also called composition plans, are often described using high-level modelling languages, such as the Business Process Modelling Language and Notation (BPMN) [21] or the Business Process Execution Language (BPEL) [20]. Using such high-level description languages allows for simplifying:

1. the implementation of systems by using generic execution engines and
2. the adaption of deployed systems to changing business needs.

Thus, high-level composition plans play an important role both for communicating business requirements between domain experts and system experts as well as a basis for the system implementation.

Since several years, enterprise systems need to fulfil an increasing number of security and compliance requirements. One reason for this is that the number

A.D. Brucker et al. (Eds.): Secure Service Composition, LNCS 8900, pp. 110–120, 2014.

of businesses that operate in regulated markets, i. e., that need to comply to regulations such as HIPAA [15] in the health care sector or Basel III [4] in the financial sector, is increasing. Such compliance regulations along with the increased awareness of IT security result in need for modelling, analysis, an d execution techniques that treat security, privacy, and compliance properties as first class citizens.

Consequently, the demand for an integrating means for specifying security and compliance requirements into languages that fulfil the need of business experts, system experts, and security experts, is increasing. Fulfilling the needs of business experts *and* system experts at the same time is already challenging—bringing the security experts to the same table, makes it even more challenging.

To meet this challenge, we developed SecureBPMN [8]: a security modelling language for expressing high-level security and compliance requirements such as role-based access control (RBAC), break-glass, separation-of-duty (SoD), delegation, or variants of the need-to-know principle. SecureBPMN is defined using a metamodel which makes it particularly suitable for integration into business process modelling languages that are themselves defined by a metamodel.

In this paper, we present SecureBPMN and its integration into BPMN. As BPMN is used in Aniketos for specifying service composition plans, this integration provides a language that allows for specifying, analysing security properties on the level of service compositions plans. Thus, SecureBPMN provides the foundation for the secure and compliant execution of service compositions.

2 Using BPMN for Modelling Service Orchestrations

The modelling in BPMN is done by expressing business processes through business models. A BPMN model is an executable specification of the workflow, i. e., a flowchart based diagram that captures the basic structure and flow of activities and data within a business process. From a high-level perspective, the development of a system using BPMN is divided into two major phases:

1. During the *design phase*, a service developer—together with domain or business experts—designs the process model, i. e., the *service composition plan*. This process model comprises both automatic services and human interactions with these services.
2. During the *deployment phase*, the process model is deployed in a business process execution engine, which can act as a service orchestrator.

This high-level view does not include several other tasks involved in system development such as the implementation of actual services and the design of the user interface.

Figure 1 shows a BPMN diagram modelling a service composition that provides a travel booking service to customers. First, customers enter their flight and hotel preferences into the system (such kind of user interactions are modelled by user tasks in BPMN). Next, two web services (modelled as service tasks) are executed and connected via parallel gateways. These web services can be operated by different service providers and, in our example, provide functionalities

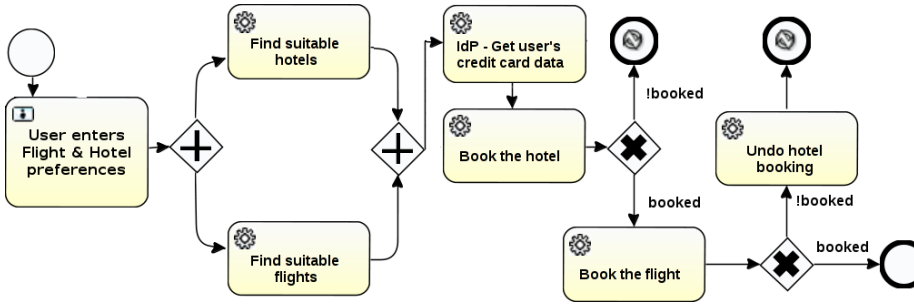

Fig. 1. A composed service for booking flights and hotels.

for finding suitable hotel and flight information respectively. Here the parallel gateways ensure that the service which queries customer's credit card data will only be executed if both the Find suitable hotels and Find suitable flights tasks are terminated successfully. By using exclusive gateways the service developer is able to indicate that the Book the hotel task might fail. In case the booking is failed (!booked), an error boundary event will be reached.

3 Security in Service Orchestrations

Our motivating example, represented in Figure 1, requires already a surprisingly large number of security and compliance requirements; for example:

- While all users should be able to search for hotels and flights, certain offers should only be available to premium customers. Moreover, travel arrangements that are above a certain limit (e. g., cost more than 5 000 Euro) might require additional approval steps to avoid credit card fraud. Thus, already this simple scenario requires a *fine-grained access control* that cannot be modelled using a simple role-based access control model. Moreover, we want to ensure that the person booking the travel and the credit card holder are the same (*binding of duty*).
- To avoid fraud or price-fixing agreements, we demand that the services for finding hotels (flights) and the booking service, are from different service providers. Of course, such a strict application of the *separation of duty* principle may hinder some travel agencies and we might want to relax this requirement such that it only holds for travels that costs more than 1000 Euro. Thus, separation of duty (as well as complementary binding of duty) should restrict individual permissions to execute an action on a task and not whole tasks (actions).
- The credit card company needs to know the price for the flight and hotel but it needs to know neither the travel destination nor the exact travel dates. Applying the principle of *need to know* or *least privilege*, can ensure such confidentiality requirements.

– Applying the discussed security and compliance requirements strictly may harm the business, e. g., if travel requests are done by an assistant to the holder of the credit card. Thus, a controlled way for transferring rights such as through *delegation*, is essential. To ensure that a delegation of tasks does not violate more important compliance rules, we also need to be able to specify restrictions on delegations (e. g., certain tasks might not be delegable at all or only delegable to persons that already possesses the necessary access rights).

Even this simple scenario shows that describing the non-functional security and compliance requirements is a significant part of the overall business process design. In real-world scenarios, the effort for specifying and implementing the non-functional requirements can easily outgrow the effort for specifying and implementing the functional requirements.

4 SecureBPMN

Security and compliance should be modelled together with the service orchestration this is while building the service composition, instead of addressing them as an after-thought. To address this need, we used a metamodel-based (Brucker and Doser [9] discuss the details of metamodel-based language extensions) approach for defining SecureBPMN. Overall, SecureBPMN is a security language that easily can be integrated into business process modelling languages or work flow modelling languages. Figure 2 shows the (slightly simplified) metamodel of SecureBPMN and its integration into BPMN. SecureBPMN allows for describing the following security and compliance requirements:

– *Access Control:* the core of SecureBPMN is a hierarchical role-based access control (RBAC) [3] language supporting constraints (AuthorizationConstraint) on the permissions. The constraints can be used to express requirements like "a credit card payment shall be approved only if it is requested by the card holder." A Subject in SecureBPMN can be an individual User or a Group of subjects. Subjects are mapped to a Role hierarchy. SecureBPMN allows to permit (Permission) the actions (Action) on resources (Resource). In case of BPMN, resources are instances of the BPMN meta-classes Process, Activity, or ItemAwareElement. This part of SecureBPMN is, conceptually, very close to SecureUML [7].
– *Delegation:* SecureBPMN supports delegation with (TransferDelegation) and without (SimpleDelegation) transferring *all* (including access to data or back-end systems) access rights that are necessary to execute a task. The former only allows to delegate tasks to subjects that already possess the necessary rights. The latter allows to delegate tasks to arbitrary subjects that, then, can act on behalf of the original subject (Delegator). The number of delegations can be restricted by maxDepth: a maxDepth of 'zero' forbids any delegation explicitly, and value of 'one' forbids a delegatee to delegate a task further. A delegation can be negotiable, i. e., the delegatee can refuse to do a delegated

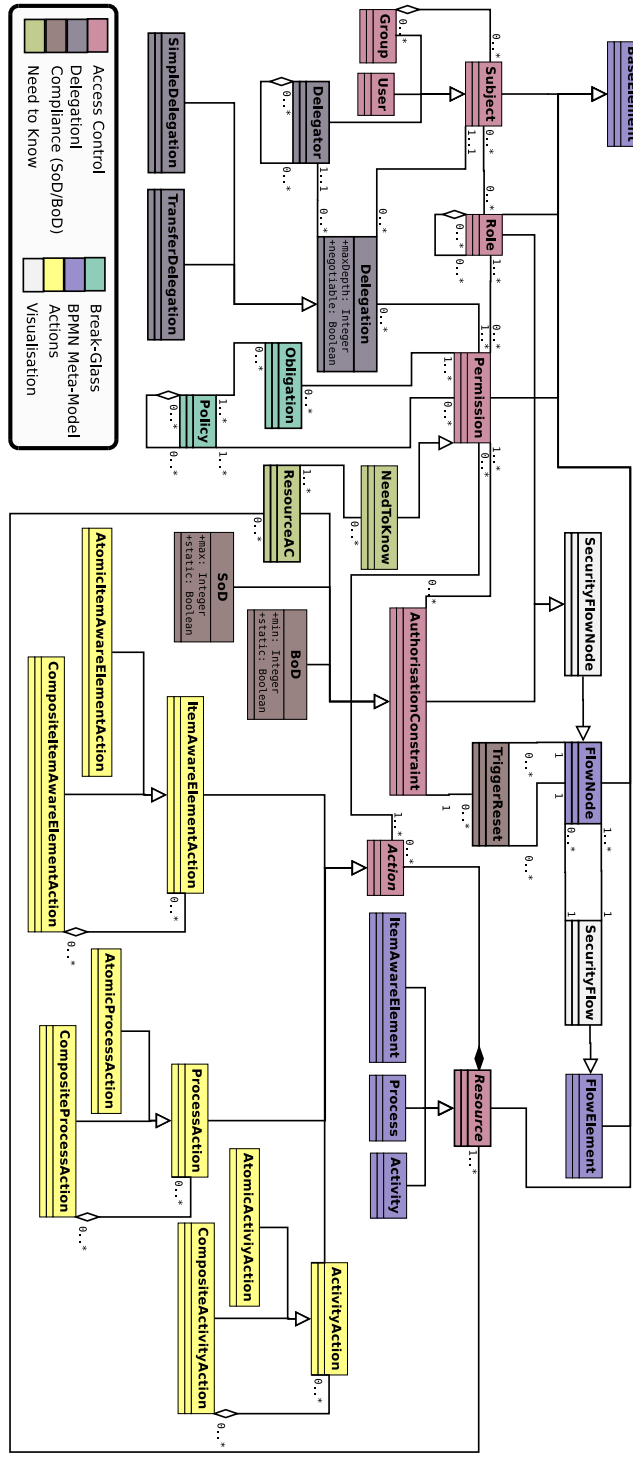

Fig. 2. The SecureBPMN metamodel (simplified) and its connection to the BPMN metamodel

task. If a delegation is not negotiable, it is an order and the delegatee has to do this task.

- *Permission-level separation and binding of duty:* SecureBPMN models separation of duty (SoD) and binding of duty (BoD) as a sub-type of AuthorizationConstraint. In contrast to existing works such as[18], which constrain all actions on a task or service, our approach results in a fine-grained notion of these properties on the level of single permissions. Moreover, SecureBPMN generalises the usually binary SoD and BoD constraints to n-ary constraints: an SoD constraints models, that a Subject is not allowed to "use" more than max permissions out of n ($\mathsf{max} < n$); BoD is generalised similarly. If a SoD (BoD) constraint is already guaranteed by the RBAC configuration, it is called static SoD (BoD). Additionally, SecureBPMN supports history-resets (TriggerReset) for SoD (BoD) for processes with loops (similar to the work of Basin et al. [6]). Such resets allow to model that a SoD (BoD) constraint only needs to hold for the last (successful) execution of a loop and, thus, avoid the risk of successively "consuming" all subjects and, eventually, resulting in a dead lock.
- *Need-to-know:* a strict application of the need-to-know principle (NeedToKnow) is another important security property. In the context of business process-driven systems, this mainly refers to restricting the access to process variables or data objects (instances of the BPMN meta-class ItemAwareElement) and, thus, the process model internal data-flow. To allow the fine-grained access to certain resources (e. g., access to the travel details is not allowed, if the travel takes longer than 14 days), we model the need to know principle as a specialised Permission that is associated with a specific authorisation constraint (ResourceAC).
- *Exceptional access control:* The strict enforcement of security and compliance requirements always bears the risk of hindering legitimate business transactions. Thus, an increasing number of enterprises implement break-glass or exceptional access control mechanisms that allow regular users to override access control decisions in a controlled manner, e. g., adhering to certain obligations (Obligation) that are either defined on the permission or policy level. SecureBPMN supports such a mechanism using a hierarchy of security policies (defined by the meta-class Policy) implementing the approach presented in [11].

Conceptually, the integration of the SecureBPMN metamodel into the metamodel of BPMN is straight forward: BPMN defines the resources and actions that are constrained by the SecureBPMN language. On a technical level, the need for diagrammatical representations of parts of the language as well as the fact that we can extend a metamodel only by subclassing (and not by introducing new superclasses) creates additional complexity:

- The SecureBPMN metamodel contains classes (SecurityFlowNode and SecurityFlow) that are not necessary for modelling security and compliance requirements. Their sole purpose is to provide a diagrammatic specification of certain requirements, e. g., SoD.

– Conceptually we would only like to specify a common hierarchy of actions, but technically this is impossible. To integrate SecureBPMN into BPMN, we need to define this hierarchy for each resource in BPMN (e. g., Activity) separately.

These parts of the metamodel (see Figure 2) are specific to BPMN and not part of the core of SecureBPMN.

5 Discussion and Future Work

We report a number of challenges and suggestions for future work that have emerged from discussions with product groups of SAP SE and our own experience in applying SecureBPMN in several case studies in the domains air traffic management (see Chapter 14) and e-government (see Chapter 15).

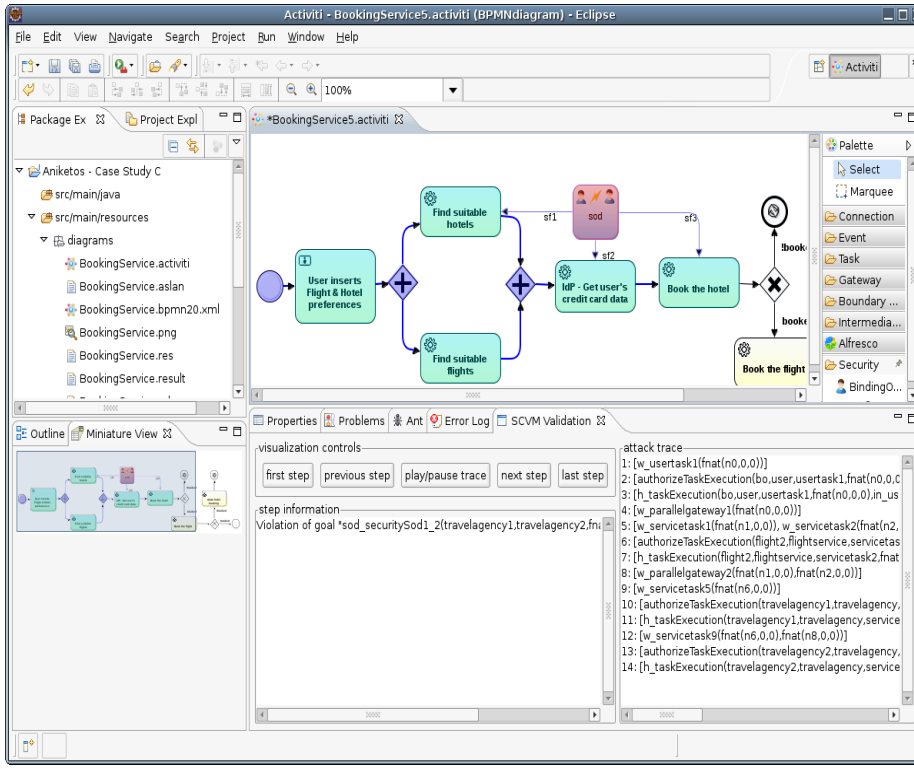

Fig. 3. Specifying security requirements diagrammatically with SecureBPMN as well as using specialised user interfaces.

5.1 Security and Compliance Properties

The selection of security and compliance properties supported by SecureBPMN is based on discussion with various experts at SAP SE as well as our own applications to case studies (e. g., see Chapter 14 and Chapter 15). In our experience, these properties cover the most important needs of business experts and, moreover, they can be expressed on the process level. Of course, there is a plethora of equally important security requirements and mechanisms (e. g., encryption as one means for realising confidentially) that need to be considered as well. In particular the security mechanisms are usually on a technical level and, thus, need to be defined during the implementation of a secure service composition. Nevertheless, the integration of technical properties into compositions plans (or business process descriptions) is an interesting line of future work.

5.2 Visualising Security Properties

One important property of BPMN is its support for describing business processes in a diagrammatic way that supports both the business experts and the system experts. Consequently, when extending such a language with a domain specific language for modelling security and compliance properties, it is tempting to provide visual representations for those properties as well. Figure 3 shows the user interface of our SecureBPMN modelling environment (which is based on Activiti BPMN Platform) in which we implemented a visual notation for SoD and BoD constraints (centre of the window). Applying this to larger case studies resulted quickly in over-populated diagrams that neither helped the business expert nor the security expert. Thus, we refrained from this approach and implemented dedicated property panes (lower part of the window). While such dedicated user interfaces provide the necessary tools for power users (i. e., security experts), they are not the best choices for increasing the awareness of business experts for security and compliance requirements. Thus, we still consider the question of finding a good (visual) representation of security and compliance requirements that can be easily understood by business experts, system experts, and security experts to be open.

5.3 Diagrams vs. Models

Many users of diagrammatic modelling languages identify the models with their visual representation, e. g., the business process diagram. This misconception is, sadly, also perceptible in most business process modelling tools: these tools present the process diagram in the centre of their user interface (see Figure 3 for an example) and provide no access to the underlying model. We argue, that a model is something much more fundamental than a diagram, i. e., a diagram is only a selected view on the model. Thus, often several diagrams, each of them visualising different aspects of a model, are necessary to capture the actual model. While this need for different views (including, e. g., an abstract, tree-like view of all model elements and their properties) is already prevalent for

modelling the functional aspects of a business process, it becomes inevitable when non-functional aspects such as security, compliance, or performance are added. Moreover, separating the model from its (visual) representation should also avoid the need for adding meta-classes purely for providing a visualisation (e. g., SecurityFlowNode and SecurityFlow in Figure 2).

5.4 Runtime Enforcement

While not the main scope of this chapter, we want to mention that modelling security and compliance requirements is only the beginning: these requirements need to be fulfilled at runtime, i. e., while executing the business processes in a business process execution engine. For example, in our prototype [13] we generate XACML [19] policies from the SecureBPMN models. An extended version of the Activiti BPMN runtime uses the generated XACML policies to enforce the access control, SoD/BoD, and the delegation requirements at runtime.

Within the Aniketos platform, we monitor compliance with various security and trustworthiness requirements using ConSpec [1], see Chapter 14 for details.

6 Conclusion and Related Work

We presented SecureBPMN, a security and compliance modelling language and its integration into BPMN. The integration of SecureBPMN, as a domain-specific language, into BPMN results in a modelling language that supports both security and compliance requirements as well as functional business requirements. Within the Aniketos platform, the security requirements are elicited using the socio-technical modelling language and tool (see Chapter 5, Chapter 6, and Chapter 7).

SecureBPMN is supported by a BPMN modelling and execution framework [10, 13] that builds the back-bone of the Aniketos Secure Composition Framework (see Chapter 9). This Framework, in addition to the modelling and secure execution of service composition plans, supports the analysis of the consistency and correctness of the implementation of security and compliance properties.

There is a large body of literature extending graphical modelling languages with means for specifying security or privacy requirements. One of the first approaches is SecureUML [17], which is conceptually very close to the access control part of our BPMN extension. SecureUML is a metamodel based extension of UML that allows for specifying RBAC-requirements for UML class models and state charts. There are also various techniques for analysing SecureUML models, e. g., Basin et al. [5] or Brucker et al. [12]. While based on the same motivation, UMLsec [16] is not defined using a metamodel. Instead, the security specifications are written, in an ad-hoc manner, in UML profiles. In contrast, integrating security properties into business processes is a quite recent development, e. g., motivated by the work of Wolter and Schaad [24]. In the same year, Rodríguez et al. [22] presented a metamodel based approach introduction a secure business process type supporting global security goals. In contrast, our

approach allows the fine-grained specification of security requirements for single tasks or data objects. Similar to UMLsec, Mülle et al. [18] present an attribute-based approach (i. e., the conceptual equivalent of UML profiles) of specifying security constraints in BPMN models without actually extending BPMN. Similarly, Salnitri et al. [23] extend BPMN with means for specifying the security properties of RMIAS [14].

Besides the modelling of (rather technical) security and compliance requirements, integrating risk and attack models into business processes is an important line of research. For example, Altuhhova et al. [2] present an integration of the information security risk management model into BPMN. In what sense, such risk modelling and security requirement approaches can be combined, is still an open question. For example, one could try to use SecureBPMN for describing countermeasures for the risks and threats expressed in the information security risk management model.

References

[1] I. Aktug and K. Naliuka. Conspec - a formal language for policy specification. *Sci. Comput. Program.*, 74(1-2):2–12, 2008. doi: 10.1016/j.scico.2008.09.004.

[2] O. Altuhhova, R. Matulevicius, and N. Ahmed. Towards definition of secure business processes. In M. Bajec and J. Eder, editors, *CAiSE Workshops*, volume 112 of *Lecture Notes in Business Information Processing*, pages 1–15. Springer, 2012. ISBN 978-3-642-31068-3. doi: 10.1007/978-3-642-31069-0_1.

[3] *American National Standard for Information Technology – Role Based Access Control*. ANSI, New York, Feb. 2004. ANSI INCITS 359-2004.

[4] Basel Committee on Banking Supervision. Basel III: A global regulatory framework for more resilient banks and banking systems. Technical report, Bank for International Settlements, Basel, Switzerland, 2010.

[5] D. Basin, M. Clavel, J. Doser, and M. Egea. Automated analysis of security-design models. *Information and Software Technology*, 51(5):815–831, 2009. ISSN 0950-5849. doi: 10.1016/j.infsof.2008.05.011. Special Issue on Model-Driven Development for Secure Information Systems.

[6] D. Basin, S. J. Burri, and G. Karjoth. Separation of duties as a service. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '11, pages 423–429. ACM Press, 2011. ISBN 978-1-4503-0564-8. doi: 10.1145/1966913.1966972.

[7] D. A. Basin, J. Doser, and T. Lodderstedt. Model driven security: From UML models to access control infrastructures. *ACM Transactions on Software Engineering and Methodology*, 15(1):39–91, 2006. ISSN 1049-331X. doi: 10.1145/1125808.1125810.

[8] A. D. Brucker. Integrating security aspects into business process models. *it - Information Technology*, 55(6):239–246, Dec. 2013. ISSN 2196-7032. doi: 10.1524/itit.2013.2004.

[9] A. D. Brucker and J. Doser. Metamodel-based UML notations for domain-specific languages. In J. M. Favre, D. Gasevic, R. Lämmel, and A. Winter, editors, *4th International Workshop on Software Language Engineering (ATEM 2007)*. Oct. 2007.

[10] A. D. Brucker and I. Hang. Secure and compliant implementation of business process-driven systems. In M. L. Rosa and P. Soffer, editors, *Joint Workshop on Security in Business Processes (SBP)*, volume 132 of *Lecture Notes in Business Information Processing (LNBIP)*, pages 662–674. Springer, 2012. doi: 10.1007/978-3-642-36285-9_66.

[11] A. D. Brucker and H. Petritsch. Extending access control models with break-glass. In B. Carminati and J. Joshi, editors, *ACM SACMAT*, pages 197–206. ACM Press, 2009. ISBN 978-1-60558-537-6. doi: 10.1145/1542207.1542239.

[12] A. D. Brucker, J. Doser, and B. Wolff. A model transformation semantics and analysis methodology for SecureUML. In O. Nierstrasz, J. Whittle, D. Harel, and G. Reggio, editors, *MoDELS 2006: Model Driven Engineering Languages and Systems*, number 4199 in LNCS, pages 306–320. Springer, 2006. doi: 10.1007/11880240_22. An extended version of this paper is available as ETH Technical Report, no. 524.

[13] A. D. Brucker, I. Hang, G. Lückemeyer, and R. Ruparel. SecureBPMN: Modeling and enforcing access control requirements in business processes. In *ACM SACMAT*, pages 123–126. ACM Press, 2012. ISBN 978-1-4503-1295-0. doi: 10.1145/2295136.2295160.

[14] Y. Cherdantseva and J. Hilton. A reference model of information assurance amp;amp; security. In *Availability, Reliability and Security (ARES), 2013 Eighth International Conference on*, pages 546–555, Sept 2013. doi: 10.1109/ARES.2013.72.

[15] HIPAA. Health Insurance Portability and Accountability Act of 1996. `http://www.cms.hhs.gov/HIPAAGenInfo/`, 1996.

[16] J. Jürjens and R. Rumm. Model-based security analysis of the german health card architecture. *Methods Inf Med*, 47(5):409–416, 2008. ISSN 0026-1270.

[17] T. Lodderstedt, D. A. Basin, and J. Doser. SecureUML: a UML-based modeling language for model-driven security. In J.-M. Jézéquel, H. Hussmann, and S. Cook, editors, *UML*, number 2460 in LNCS, pages 426–441. Springer, 2002. ISBN 3-540-44254-5.

[18] J. Mülle, S. von Stackelberg, and K. Böhm. A security language for BPMN process models. Technical report, University Karlsruhe (KIT), 2011. URL `http://digbib.ubka.uni-karlsruhe.de/volltexte/1000023041`.

[19] OASIS. eXtensible Access Control Markup Language (XACML), version 2.0, 2005. URL `http://docs.oasis-open.org/xacml/2.0/XACML-2.0-OS-NORMATIVE.zip`.

[20] OASIS. Web services business process execution language (BPEL), version 2.0, Apr. 2007. URL `http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf`.

[21] Object Management Group. Business process model and notation (BPMN), version 2.0, Jan. 2011. Available as OMG document formal/2011-01-03.

[22] A. Rodríguez, E. Fernández-Medina, and M. Piattini. A BPMN extension for the modeling of security requirements in business processes. *IEICE - Trans. Inf. Syst.*, E90-D:745–752, March 2007. ISSN 0916-8532. doi: 10.1093/ietisy/e90-d.4.745.

[23] M. Salnitri, F. Dalpiaz, and P. Giorgini. Modeling and verifying security policies in business processes. In I. Bider, K. Gaaloul, J. Krogstie, S. Nurcan, H. A. Proper, R. Schmidt, and P. Soffer, editors, *BMMDS/EMMSAD*, volume 175 of *Lecture Notes in Business Information Processing*, pages 200–214. Springer, 2014. ISBN 978-3-662-43744-5. doi: 10.1007/978-3-662-43745-2_14.

[24] C. Wolter and A. Schaad. Modeling of task-based authorization constraints in BPMN. In G. Alonso, P. Dadam, and M. Rosemann, editors, *BPM*, volume 4714 of *LNCS*, pages 64–79. Springer, 2007. ISBN 978-3-540-75182-3.