Modelling and Transforming Security Constraints in Privacy-Aware Business Processes

Jutta Mülle, Silvia von Stackelberg and Klemens Böhm
Karlsruhe Institute of Technology (KIT) 76131 Karlsruhe, Germany

Abstract—**Security and privacy are essential for business processes (BPs). In particular, BPs dealing with personally-identifiable information require mechanisms to give data owners control over their data. Currently, business-process-management systems (BPMSs) lack security features important for BPs in SOA. We propose a language sufficiently broad to formulate security constraints. In addition, we considerably ease how data owners can control their security, privacy and trust preferences at process runtime. The BPMS extensions we have implemented transform security-enhanced BPMN schemas into executable secure processes in a versatile manner.**

I. Motivation

The support of security goals like authorization, authentication, and confidentiality of data is essential for business-process-management systems (BPMSs) in service-oriented architectures (SOA). In addition, BPMSs dealing with personally-identifiable information (PII) have to protect private data from misuse. Further, BPMS have to support explicit control of PII usage by the data owners, such as selecting trusted services to process PII. Currently, BPMSs do not support all this adequately [1]. However, BPMSs should offer security mechanisms which are easy to use.

Problem Statement: To represent rich security functionality, we build on existing approaches to specify constraints to BP models [2]–[4]. But the security vocabularies proposed so far are not sufficiently comprehensive. For instance, one cannot represent delegation for data-access rights in BP. Further, the vocabularies facilitate the specification of high-level security goals, but rarely allow for detailed constraints. To illustrate, some approaches cover the notion of confidentiality, but do not allow a specification of which realization alternative to use.

A further issue is that BPMSs must consider privacy and trust preferences of users, as follows: With conventional approaches, users specify preferences, such as “who is allowed to access which data” or the “level of trust” for service providers processing their PII, a priori. However, they should be able to do so at runtime, a feature we refer to as *user involvement*. This is because preferences vary for different instances of a BP, because their contexts are different. Think of a user selecting a service to process his PII. But availability of services for processing changes at runtime. This asks for language constructs that are easy to use instead of modelling user involvements explicitly.

Contributions: We propose a language sufficiently broad and deep to represent security aspects for BPs. A new feature of the language is to support delegation of access rights to BP data. A distinguished property is the specification of user involvements. To illustrate, process designers can now specify trust aspects of a service selection by one annotation term. Using this language for two different, complex real-world scenarios has shown that the language is sufficiently generic to cover important application needs. We also propose new forms of transformation of the various language constructs.

II. Requirements

In an earlier study, we have systematically collected security requirements for BPs [1]. They form the basis of our annotation language by representing the first class of requirements in the following. Another class says how to let users specify privacy and trust concerns during process execution.

A. Security for BP

Authorization regulates which users may execute activities or access the data. Respective mechanisms have to consider the process context, such as execution times of running activities. Further issues are the coordination of access control with the process flow, namely separation and binding of duties (SoD, BoD). Delegation mechanisms transfer execution rights for activities and access rights to data. We now focus on the latter ones, which existing literature hardly covers. They are needed if there are access restrictions to data, and someone who currently does not have access rights needs them. We distinguish two cases: First, the receiver of delegated rights can be a process participant. To illustrate, think of a sticky policy [5] that specifies that only the data owner has access rights to her PII. But another role holder needs access to the PII to perform an activity. Second, the receiver can be external to the process. We need delegation mechanisms if services called need access to data but are not authorized to.

Authentication functionality ensures that a system can rely on the identity and on attribute values of actors. An actor can be, for example, a human. Authentication is challenging in distributed, inter-organizational environments with mechanisms such as single sign-on (SSO). Process designers must be able to rely on federated identity-management systems where identity providers (IdPs) authenticate individuals and provide certified attributes about them. *Auditing* means that the system monitors the BP, to provide traceability. Typically, compliance rules frame auditing. Auditing can affect any aspect of BPs, such as execution of activities or calls of services. *Confidentiality* is the protection of data and service calls in BPs against misue. Authorization constraints are a first step towards confidentiality, but additional mechanisms are needed, such as encryption. *Data Integrity* ensures that a system must act in an expected way at each point in time regarding the transfer, processing or storage of data.

Related Work: [2], [3], [4], [6], [7] have presented work on modelling security for BPs. [2] comprises authorization, integrity and confidentiality constraints in BPMN annotations, but does not consider delegation of data-access rights. [3] proposes a broad security vocabulary for BPMN artifacts. However, it lacks authorization constraints, such as SoD. [8] does not focus on security facets other than authorization, role assignment and policy checks at the enforcement level. Security- and privacy-aware BPs call for additional properties beyond these functional requirements to let users specify security, privacy and trust aspects.

B. Involving Data Owners at Process Runtime

Privacy laws state that providers have to ask data owners for agreement to use their PII in certain situations. Further, data owners might want to control more flexibly who should be allowed to have access to their data, and which trust level services processing their PII must have. But BP context, such as service availability and process participants, changes over time. Thus, data owners must be able to specify their security, privacy and trust preferences at runtime. To illustrate, a data owner has to modify at runtime her specified level of trust that service providers accessing her data must have, because there are no services fulfilling her specifications. Modelling security-specific user involvements results in complex models. For example, to specify the selection of a service deemed trustworthy by the user, we had to model 11 activities, including user interactions. Thus, re-usable mechanisms to represent complex, but recurring user involvements are needed.

Related Work: [9] lets data owners specify their privacy preferences for PII in BPs as well. They keep it open how and when users specify their preferences, and they do not take into account that user preferences change with BP context. We protect personal data by access-control and delegation mechanisms. [10] focuses on privacy aspects by restricting purpose, recipient and retention obligation of data in a sticky policy.

III. Annotation Language

Our language is embedded in BPMN 2.0 as structured text annotations, i.e., conforms to BPMN at the syntax level. This reduces our implementation effort. We now give an overview of the language and describe selected constraints. [11] provides further information.

The annotation types of our language correspond to the different requirements, namely authorization, authentication, auditing, confidentiality, integrity, and security-, privacy- and trust-related user involvements. The first groups are well-known in literature, but not as broad and deep as required, see II.A. Annotations for privacy- and trust-related user involvements are a new notion.

Annotations have the following generic structure:

≪*AnnotationTerm: list(ParameterName=“Value”)*≫

Each annotation term is defined for a set of BPMN elements, e.g., for activities, lanes, data stores, or message flows.

A. Authorization Constraints

Authorization constraints (Table I) specify who has which rights under which restrictions. Dots indicate parameters we have omitted here. To manage authorizations, one typically specifies the holders of roles which may perform or access annotated elements.

Authz Constraint	Syntax
Role Assignment	≪Assignment: type=“Role” name=“$rolename”≫
Mechanism Assgm.	≪Assignment: type=“Mechanism” ...≫
User Assignment	≪Assignment: type=“User” name=“$username”≫
Separation of Duty	≪SoD: role=“$rolename” number=“$nr” ...≫
Binding of Duty	≪BoD: spec=“weak”≫
Adaptation	≪Adaptation: rights=“$rightsname”...≫
Task-Delegation	≪T-Delegation: target=“$rolename”...≫
Data-Delegation	≪D-Delegation: rights=“$rightsname”...≫

Table I
Overview of Authorization Constraints

Assignment annotations allow to bind roles or users to BPMN elements, such as to activities or to lanes. This means that only role holders or the users specified are allowed to execute the activities contained therein. Further, Mechanism Assignment allows to specify security-specific ressource-allocation mechanisms, to assign individuals to perform BPMN activities. The annotations for separation of duty (SoD) and binding of duty (BoD) constrain the execution of a group of activities or of multiple instances of an activity, to be performed by different individuals (SoD) or by the same one (BoD). To avoid process blocking because, say, process participants are absent, we allow to ease BoD constraints by explicit transfer of responsibility, i.e., a weakening of the BoD constraint, by the optional parameter $spec =$“$weak$”. To facilitate ad-hoc process changes in a controlled way, we allow the specification of rights to adapt process instances.

Example: Figure 1 shows two role assignments for lane “Placement coordinator” and lane “Learner”. This means that these lanes represent roles. The BoD annotations in Figure 1 mean that the same individuals have to perform the activities. A learner cannot transfer activities to other role holders of learner, while a placement coordinator can do so.

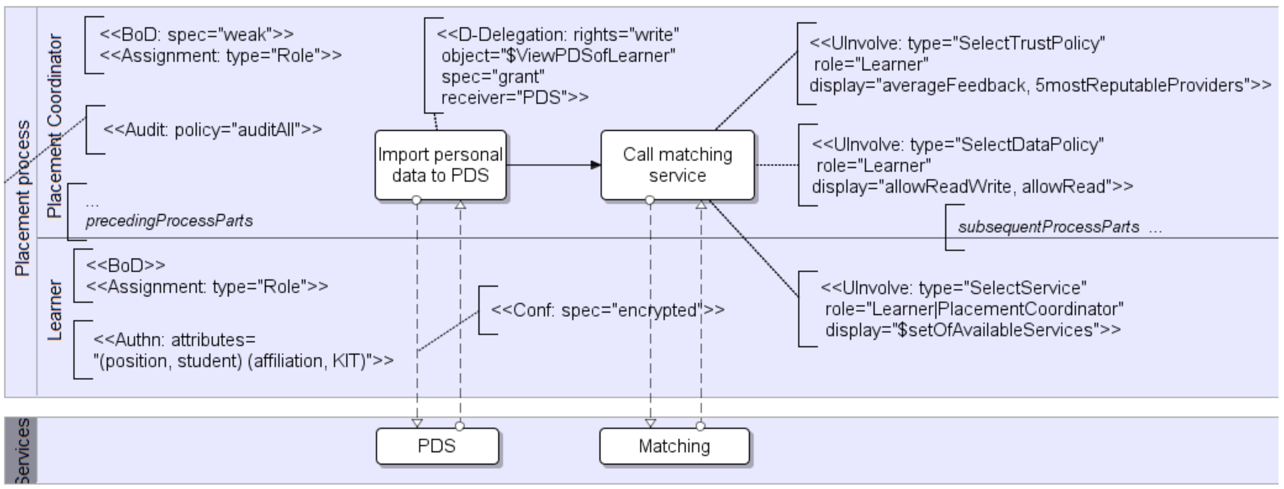
Placement process
Placement Coordinator
Learner
Services
<<BoD: spec="weak">>
<<Assignment: type="Role">>
<<Audit: policy="auditAll">>
...
precedingProcessParts
<<D-Delegation: rights="write" object="$ViewPDSofLearner" spec="grant" receiver="PDS">>
Import personal data to PDS
Call matching service
<<UInvolve: type="SelectTrustPolicy" role="Learner" display="averageFeedback, 5mostReputableProviders">>
<<UInvolve: type="SelectDataPolicy" role="Learner" display="allowReadWrite, allowRead">>
subsequentProcessParts ...
<<BoD>>
<<Assignment: type="Role">>
<<Authn: attributes= "(position, student) (affiliation, KIT)">>
<<Conf: spec="encrypted">>
<<UInvolve: type="SelectService" role="Learner|PlacementCoordinator" display="$setOfAvailableServices">>
PDS
Matching

Figure 1. Example Security Annotations

Delegation: Delegating access rights may apply to activities, so-called *task-based delegation*, but also to data. We now describe *data-based delegation*.

The annotation

≪D-Delegation: rights="list ($rightsname)"
object="$objectname"
interval="($activityname1,$activityname2)|$group"
owner="$rolename"
receiver="$rolename"|"$webservicename"
spec="$specification"≫

specifies the delegation of access rights to data, with $read$, $write$, $delete$, or $policy$ as possible values for rights. Delegation varies depending on the annotated BPMN context and the category of the receiver and therefore requires different parameters. E.g., a delegation annotated to an activity requires an $object$ parameter, but the delegation for annotated data does not.

The specified rights (e.g., $read$, $policy$) describe which use of the $object$ is delegated. The validity of a delegation can be specified for the execution time of activities in the interval [$\$activityname1$, $\$activityname2$], i.e., the part of the process starting with $\$activityname1$ and ending with $\$activityname2$, or in the $\$group$ of activities. The owner of the rights is specified using a $rolename$. The receiver of the rights can be a holder of a role or a web service.

Example: The annotation ≪D-Delegation ...≫ in Figure 1 means to give access rights to the specified object "$View-PDS-of-Learner" from the holder of role "Placement coordinator" to the web service of type "PDS".

B. Authentication

The annotation *≪Authn: attributes=list("attributename", "value") idp="$identityProvider"≫* means that process participants must be authenticated. The task of an IdP is to certify attributes of process participants. For this purpose, we allow for specifying a list of attributes which an IdP has to certify. For instance, any holder of role "Learner" in Figure 1 must be authenticated as a student at KIT.

C. Auditing

The annotation ≪*Audit:* $policy$ ="$\$policyname$"≫ enforces a monitoring of the execution of an activity, a group of activities, etc. Legal requirements call for different kinds of logging. For example, if a process handles personal data, it must be logged, among others, who has performed which action upon which data, and at which time. Thus, we assume that an auditing policy "$\$policyname$", stored in the BPMS, specifies the objects to be logged and the kind of logging. To illustrate, we have annotated the pool "placement process" with the annotation term $Audit$ in Figure 1. It specifies a monitoring of all task executions and all security tasks, e.g., the delegation or the encrypted call of the matching service.

D. Confidentiality and Integrity

Authorization mechanisms support a basic level of confidentiality, as only authorized users or services may perform activities and access data. Protecting message flows improves confidentiality. This can be expressed by ≪*Conf: spec="$value"*≫. Possible values of "spec" are "encrypted" or "signed". For example, the data flow from activity "Import personal data to PDS" to activity "PDS" in Figure 1 must be encrypted. We allow the annotation ≪*Integrity*≫ for data.

E. User Involvements

User involvements are important, because users have to specify and agree to security, privacy and trust preferences for BPs that are context-specific at runtime. We have identified the following user involvements: giving consent, selecting services in line with the trust levels required, specifying data-access policies or give trust feedback.

To support the representation of user involvements, we introduce annotation terms to specify them declaratively. One can annotate activities and certain events of a process model with user involvements. Thus, the process designer simply specifies them, instead of implementing them explicitly. The representation is as follows:

$$\ll \textit{UInvolve: list(parametername=“value”)} \gg$$

The parameter *type* (see Table II) specifies the intention of a user, e.g., *type=“Consent”* means giving user consent. The involvements to let users set their preferences typically are invoked right before the annotated activity takes place.

User Involvement	Syntax
Give Consent	≪UInvolve: type=“Consent” ...≫
Service Selection	≪UInvolve: type=“SelectService” ...≫
Select Data Access Policy	≪UInvolve: type=“SelectDataPolicy” ...≫
Select Trust Policy	≪UInvolve: type=“SelectTrustPolicy” ...≫
Give Trust Feedback	≪UInvolve: type=“TrustFeedback” ...≫

Table II
OVERVIEW OF USER INVOLVEMENTS

Select Trust Policy. It is intuitive to characterize service providers by trust levels, e.g., their reputation gained in the past. We allow process participants to select providers by means of a trust policy:

$$\ll \textit{UInvolve: type=“SelectTrustPolicy”} \\ \textit{display=“list(option)”} \gg$$

Example: The user involvement of type “SelectTrustPolicy” in Figure 1 means that a learner has to select a trust policy from the options given.

IV. TRANSFORMATION OF SECURITY ANNOTATIONS

A BPMN model with security annotations is input of our transformation component. The main concern now is the transformation to representations which support the enforcement of the annotated constraints. To achieve these representations, we have developed a dedicated tool, the Security Modelling and Configuration Component for BPs (BP-SMC). To this end in turn, we have to deal with several system layers described by distinct models. Model transformation is essential for any model-driven software development (MDD). MDD approaches for security in BPs, see, e.g., [6], [12], have generic process models and security models as source, and XACML policies or UML models (e.g., for use case) as targets.

The source of our transformation is a BPMN process model with annotations representing the security model. We have discovered that different security constraints on BPs need to be transformed in different ways, i.e., have different outcomes depending on the security category of the annotation: generating or modifying a BP-access-control security policy, setting parameters to configure security components, or adapting the BP by canned process fragments. This leads to the three target models: XACML as policy language enhanced with BP-specifics, a data model of configuration parameters for the security components of our secure BPMS, and a BPMN process model adapting the model of the application process.

Our transformation approach is more differentiated than existing ones (e.g., [6], [4]) because we use a BPMN-specific security model, and we deal with user involvements. Annotated user involvements result in transformation for an adapted process model, in particular.

To give way to security-specific functionality, we have extended a WfMC-compliant BPMS architecture with BP-specific components according to an XACML security framework [13]. It enforces the security constraints during BP execution. We have implemented our transformation mechanism as part of this security-enhanced BPMS.

V. CONCLUSIONS

We have presented a rich language to represent security aspects as well as trust- and privacy-specific user involvements for BPs in SOA. Our extensions of a BPMS provide security support from the modelling to the runtime phase of a BP lifecycle. Using this infrastructure, we are able to execute BPs according to the specified constraints.

Acknowledgements: This research has received funding from the Seventh Framework Programme of the European Union (FP7/2007-2013) under grant agreement n^{o} 216287 (TAS^{3} - Trusted Architecture for Securely Shared Services)

REFERENCES

[1] J. Müller, J. Mülle, S. von Stackelberg, and K. Böhm, “Secure Business Processes in Service-Oriented Architectures – A Requirements Analysis,” in *ECOWS*, 2010.
[2] C. Wolter, M. Menzel, and C. Meinel, “Modelling Security Goals in Business Processes,” in *Modellierung*, 2008.
[3] A. Rodríguez, E. Fernández-Medina, and M. Piattini, “A BPMN Extension for the Modeling of Security Requirements in Business Processes,” *Trans. Inf. Syst.*, vol. E90-D, 2007.
[4] M. Menzel, I. Thomas, and C. Meinel, “Security Requirements Specification in Service-Oriented Business Process Management,” in *ARES*, 2009.
[5] M. Mont, S. Pearson, and P. Bramhall, “Towards Accountable Management of Identity and Privacy: Sticky Policies and Enforceable Tracing Services,” in *DEXA Workshop*, 2003.
[6] C. Wolter, A. Schaad, and C. Meinel, “Deriving XACML Policies from Business Process Models,” in *WISE*, 2007.
[7] M. Menzel and C. Meinel, “SecureSOA - Modelling Security Requirements for Service-oriented Architectures,” in *Services Computing*, 2010.
[8] E. Bertino, L. Martino, F. Paci, and A. Squicciarini, *Security for Web Services and Service-Oriented Architectures*. Springer, 2010.
[9] B. Alhaqbani, M. Adams, C. Fidge, and A. ter Hofstede, “Privacy-Aware Workflow Management,” BPM Center, Tech. Rep. BPM-09-06, 2009.
[10] S. Short and S. Kaluvuri, “A Data-Centric Approach for Privacy-Aware Business Process Enablement,” in *IWEI*, 2011.
[11] J. Mülle, S. von Stackelberg, and K. Böhm, “A Security Language for BPMN Process Models,” KIT TR, 2011.
[12] A. Rodríguez and et al., “Semi-formal transformation of secure business processes into analysis class and use case models: An MDA approach,” *Inf. Softw. Technol.*, vol. 52, 2010.
[13] J. Müller and K. Böhm, “The Architecture of a Secure Business-Process-Management System in Service-Oriented Environments,” in *ECOWS*, 2011.