Security Requirements Engineering for Secure Business Processes

Elda Paja[1], Paolo Giorgini[1], Stéphane Paul[2], and Per Håkon Meland[3]

[1] Universitá degli studi di Trento, Italy
paja@disi.unitn.it,paolo.giorgini@unitn.it
[2] Thales Research and Technology, France
stephane.paul@thalesgroup.com
[3] Sintef, Norway
Per.H.Meland@sintef.no

Abstract. Traditional approaches to business process modelling deal with security only after the business process has been defined, namely without considering security needs as input for the definition. This may require very costly corrections if new security issues are discovered. Moreover, security concerns are mainly considered at the system level without providing the rationale for their existence, that is, without taking into account the social or organizational perspective, which is essential for business processes related to considerably large organizations. In this paper, we introduce a framework for engineering secure business processes. We propose a security requirements engineering approach to model and analyze participants' objectives and interactions, and then derive from them a set of security requirements that are used to annotate business processes. We capture security requirements through the notion of social commitment, that is a promise with contractual validity between participants. We illustrate the framework by means of an Air Traffic Management scenario.

Key words: security requirements, business process, BPMN, social commitments

1 Introduction

Business processes are the combination of a set of activities within an enterprise following a structure that describes their operational order and dependence, to pursue a desired objective or result. Business process modelling enables a common understanding and analysis of a business process [1]. It can be used to describe complex interactions between business partners and to indicate related business requirements on an abstract level. With the growth of businesses, business processes have experienced considerable growth, not only in size but also in complexity. The evolution in the nature of organizational information systems into cross-organizational systems has exposed an organization's assets and resources in terms of business services [18]. These business services are modelled as

business processes reflecting the control and information flow, without considering any security related issues. But, security is an important aspect that needs to be considered early during the modeling phases [14]. Information systems are inevitably subject to threats [7] that may influence organizational assets. This might increase the vulnerability of the provided business services, hence that of business processes.

Current approaches to business process modelling lack a security focus in the early phases [4]. This is often due to the fact that business analysts are not security experts and assume that this will be bolted on later. Fortunately, this trend is changing and we are seeing examples where security requirements are integrated into business processes. For instance, Wolter et al. [18], describe an approach to integrate security goals and constraints in business process modelling together with a model-driven transformation that focuses on authorisation requirements. In a similar way, Rodriguez et al. [14] introduce an extension to the *Business Process Modeling Notation* (BPMN) to allow business analysts express security needs from their perspective. However, these approaches do not facilitate the creation of security policies in compliance with the modelled security properties. Moreover, they provide no rationale on how the business analyst should decide upon security requirements in the business process. In [10], Menzel et al. employ a model-driven approach to generate security policies based on security patterns. They provide an enhancement to BPMN to enable the assessment of risks based on the evaluation of assets and the trustworthiness of participants, and to enable the annotation of security requirements such as confidentiality or integrity.

Pavlovski and Zou [13] extend BPMN to capture non-functional requirements related to business process models, among which security policies that apply. Their extension involves two notations: *operating condition*, which refer to constraints over activities, and *control case*, which describes the risks associated to the operating condition together with mechanisms to mitigate or reduce business risks. Cardoso et al. [6] start from goal modeling to elicit business process models. Goals are considered as objectives to be achieved by the execution of a business process. The authors show how the elicitation process takes place starting from a preliminary phase to a supplementary one, which refines the goal models by using NFR (Non-Functional Requirements) catalogues. However, how goal models are related to business process models is left as future work.

For high-level business process modelling in UML, the approaches by Sindre and Opdahl [16] related to misuse cases and UMLSec by Jürjens [9], are well-known. In [15], Sindre proposes another technique, which complements misuse cases, to capture security issues throughout business process diagrams. The author extends UML activity diagrams by adding malicious activities and malicious actors to identify possible threats, and then adds defensive processes to mitigate the identified risks, suggesting where in the process the mitigation activities would be placed. These approaches are complementary to ours.

In this paper we will support security requirements engineering in the context of cross-organizational business processes. We present a novel framework that elicits a set of security specifications, analyzing first the organizational objectives

of different roles and analysing security from an organizational perspective. As cross-organizational business processes capture collaborations and interactions among different organizations or partners, it is important to provide a level of abstraction on which partners first agree on the business goals of their collaboration [8]. We adopt an interaction-oriented perspective to identify and express security needs. We analyse social interactions in the organization, responsibilities of relevant actors, information flow constraints, and rules actors should comply with. *Social commitments* are a powerful formalism to model actors' interactions [17] in the pursuit of achieving their objectives. A social commitment stands for a promise from a debtor (actor) to a creditor (actor) that if the antecedent is brought about, the consequent will be brought about (antecedent and consequent are propositions, promises actors exchange). Formally, a social commitment is represented as a quaternary relation C(debtor, creditor, antecedent, consequent). Commitments are created and evolve according to the messages actors exchange. These social abstractions are rooted in interaction, therefore they are very effective to capture security needs, as most of the security issues arise during interaction. They have contractual validity: their violation might lead to further commitments by the violator. The contractual validity of commitments enables the development of robust interactions, wherein violations eventually result in penalties and loss of reputation. The derived commitments serve as security specifications while modelling business processes.

In a nutshell, our approach is to *security-annotated-BPMN* [14], what BMM [1] is to BPMN: it provides the justification for the security requirements by capturing security needs. This statement is particularly true for market social structures, as defined in [12], that is, loci of interaction between participants who are peers of one another. It however remains valid for enterprise-type social structures, as demonstrated in this article's running example, where an enterprise is an organization with identifiable officers and with internally established goals that reflect the purpose of the organization.

The paper is structured as follows. Section 2 introduces our modelling framework, including the three operational views of the modelling language (SecCo) that allow one to model and express security needs and the specification of security requirements via commitments. Section 3 shows how SecCo requirements can be transferred to BPMN. Section 4 discusses the approach and future directions, whereas section 5 makes some final remarks.

2 Modelling Security via Commitments

Figure 1 outlines our modelling framework, namely *SecCo*, which stands for *Sec*urity via *Co*mmitments. The figure shows how security requirements for the business processes under design are derived from the security needs expressed by the stakeholders.

[1] Business Motivation Model Version 1.1 `http://www.omg.org/spec/BMM/1.1/`

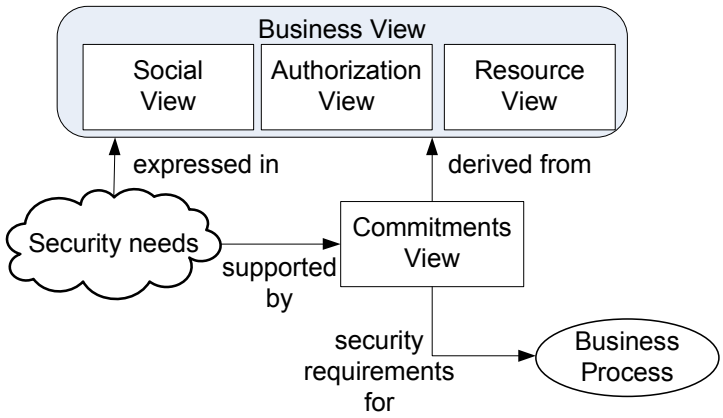


Fig. 1: Outline of our approach: from security needs to security requirements

Security needs are *expressed in* the *business view*. In the current version of the framework the *business view* consists of three different views: *social view*, *authorization view*, and *resource view* (Fig. 1). These views provide different perspectives over the considered setting. More detail about the views is provided in Section 2.2.

Together, these views give a comprehensive picture of the organization addressing at the same time both business concerns and security aspects. *Security needs* are *supported by* the *commitments view*, which consists of a set of commitments between actors. The *commitments view* is a high-level specification of the security requirements for the system-to-be. As long as the actors do not violate those commitments, the security needs in the setting are ensured. The commitments view can be automatically derived from the business view using a dedicated modelling tool.

2.1 Running Example

Air traffic management involves many critical processes between several organizations. We will illustrate the features of SecCo with the help of a running example developed from a report published by the Aniketos project [3].

Handover scenario: En-route air traffic controllers work in facilities called Area Control Centers (ACC). Each ACC is responsible for a vast airspace. As an aircraft reaches the boundary of an ACC, it is handed over to the Adjacent ACC. This transfer of responsibility involves electronic exchange of information between the ACCs and the aircraft, with the purpose of collaboratively managing the flight's Reference Business Trajectory (RBT). The sharing of RBT related information is carried out according to the Flight Object (FO) paradigm. The FO contains flight data, including RBTs. Today, handover is handled by voice

(radio), sometimes supported by dedicated point-to-point electronic means (data link). During the handover, the aircraft is given a new radio frequency and the pilot begins talking to the next controller. This process continues until the aircraft is handed over to a terminal controller. In the short future, Flight Handover will be enabled by the SWIM (System Wide Information Management) infrastructure. All the information exchange will be made possible through SWIM, an internet-like network for the aviation community. Like for the internet, SWIM will enhance communications, but will also be vulnerable to new threats. In our handover case-study, SWIM will be responsible for: managing the handover request it receives from the controlling ACC, check the eligibility of the ACC to handover the flight, determine the next ACC to be contacted, notify the handover request to the identified FO server, and finally change the unit's role, making the adjacent ACC the new controlling ACC.

We have modelled the scenario using SecCo as shown in Fig. 2 and Fig. 3.

2.2 Multi-view Modelling

A distinguishing feature of SecCo is to rely on multiple views of the same model. Each view represents a specific perspective on the business view. Multi-view modelling promotes modularity and allows modellers to focus on well-defined tasks, as opposed to building a single model representing orthogonal concerns.

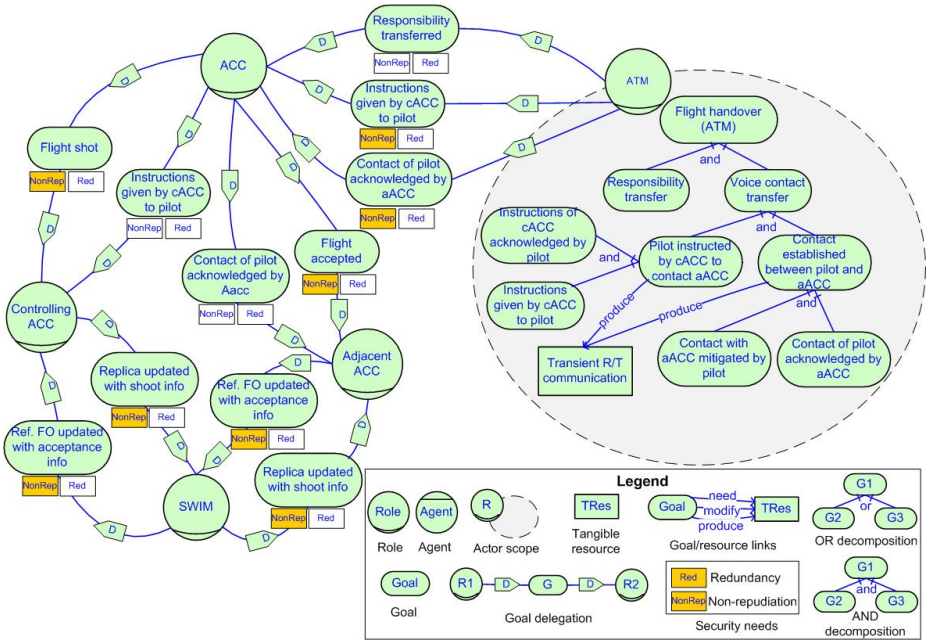
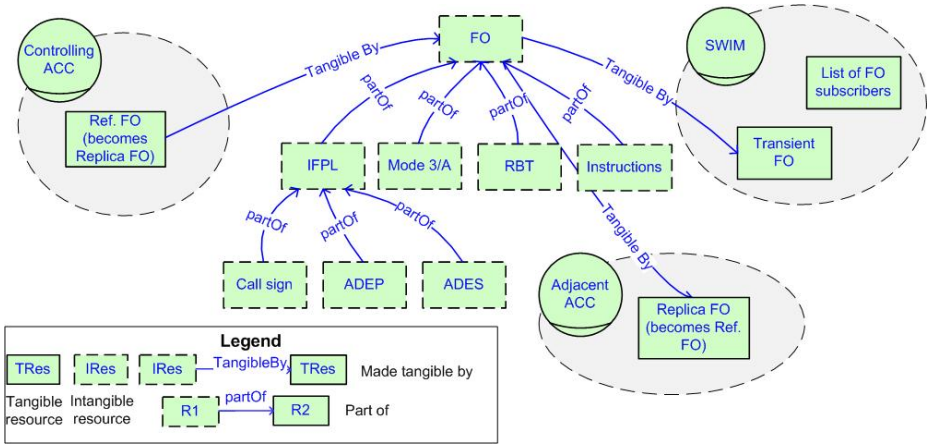
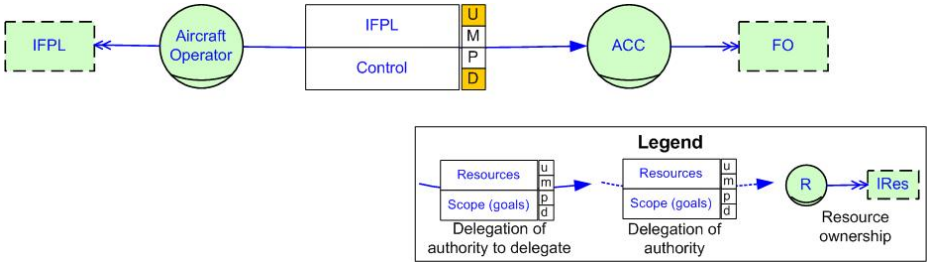


Fig. 2: Multi-view modelling for the handover scenario: *Social view*



(a) *Resource view*



(b) *Authorization view*

Fig. 3: Multi-view modelling for the handover scenario

SecCo currently includes three views:

1. *Social view (Fig. 2)*: builds on top of traditional i^*-based frameworks [19, 5], extending them to provide support for cross-organizational settings. In this view we can represent actors as intentional and social entities, capturing the objectives they want to achieve and their interactions respectively.
   There are two types of actors: *agent* and *role*. An agent refers to the actual participants who are going to adopt certain roles at runtime. As a consequence an agent can play multiple roles. At design-time, we do not know most actors, so they are modelled as roles, except for the agents that are already present in the setting and are known since requirements time. Actors, as intentional entities, are characterized in terms of *goals*, their desired objectives. Referring to our running example, ATM for instance, is a role, and it has the goal of handing over the flight ("Flight handed over"). Goals can be refined by *AND/OR-decompositions*. For example, the goal "flight handed over" is further AND-decomposed into "responsibility transferred" and "Voice contact transferred". The latter goal is also further refined by

AND-decompositions. Goals are linked to *tangible resources*—information represented by some support means—in various ways: a goal can *read* (*use*), *produce*, *modify*, or *distribute* a resource. In our example, we show how ATM *produces* the tangible resource "Transient R/T communication" while achieving the goal "Contact established between pilot and aACC". Resource *possession* indicates that an actor has or possesses a certain resource without the need of interacting with others. In the social view there are two social relationships: *resource provision* and *goal delegation*. Resource provision captures the distribution and exchange of information, whereas goal delegation captures how an actor (delegator) transfers the responsibility of achieving a goal (delegatum) to another actor (delegatee). Goal delegation indicates that the delegator expects the delegatee to achieve the delegated goal. In the handover scenario, there are several goal delegations, such as Adjacent ACC delegates the goal "Ref. FO updated with acceptance info" to SWIM.

2. *Resource view (Fig. 3a)*: represents the resources in the given setting, providing a structure of how the various resources are interconnected. We distinguish between tangible resources, which denote the representation of information by some means (already introduced in the social view) and *intangible resources*, which denote information irrespective of its representation. For instance, "FO" (flight object) and its constituent information, such as "IFPL", "RBT", etc., are intangible as long as they are not represented by any tangible resources. Resources can be hierarchically structured via the *part-of* relation, which relates homogeneous resources (intangible to intangible, tangible to tangible). Intangible resources are *made tangible by* tangible resources. For example, "FO" is made tangible by "Transient FO".
3. *Authorization view (Fig. 3b)*: represents the flow of permissions or authorizations, how they are delegated from one actor to another. We distinguish between delegation of *authority* and delegation of *authority to delegate*.
   Delegation of *authority* shows authorizations given by an actor to another for one or more intangible resources, on specific *operations*: *modify* (M), *produce* (P), *use* (U), and *distribute* (D). An authorization can be limited to a *scope*—a set of goals—that determines the purpose for which authorization is passed. We assume that the authorizations start from the owner of the resource(s). Ownership relation is represented as a double-headed arrow from the role to the intangible resource. For instance, in the handover scenario, the Aircraft Operator owns the resource "IFPL" and authorizes ACC to read (use: U) and distribute (D) IFPL in the scope of the goal "Control".
   The delegation of *authority to delegate* implies that the delegatee can further delegate the received authorization, and subsumes the first type of authorization.

2.3 Expressing Security Needs

SecCo supports the following security needs:

– *Non-repudiation*: in a goal delegation, the delegator wants to prevent the delegatee from challenging the validity of the delegation.

- *Redundancy*: in a delegation, the delegator wants the delegatee to adopt redundant strategies for the achievement of the delegated goal. He can either use different internal capabilities, or can rely on multiple actors.
- *No-delegation*: the delegator expresses a security need over the delegation that requires the delegatee not to further delegate goal fulfilment. Such requirement is closely related to *trust*: the delegator actor trusts *that* specific delegatee actor for some goal, and does not trust other actors the delegatee might want to involve.
- *Non-disclosure*: when authority over a resource is granted without transferring authority to delegate.
- *Need-to-know*: when delegation of the authority to delegate is restricted to a goal scope. The actor granting the authority enables the second actor to delegate permission to others as long as other actors conduct operations on the resource within the specified scope.
- *Integrity*: when an actor does *not* delegate the authority to modify a resource.

Considering the description of security needs, we can say that they are expressed either over delegations or authorizations. This is in compliance with our view of taking an interaction-oriented approach for identifying security issues. The first group represents constraints actors might want to impose over their interactions, especially when one is relying on another to get things done (goal delegations). The second group considers security issues that arise due to permission flows and information exchanges (authorizations and resource provisions).

Referring to our running example, we can illustrate how some of these security needs are supported by SecCo. In the new SWIM environment, **(Ex1)** SWIM infrastructure would want to ensure *non-repudiation* of "Replica updated with shoot info" when delegating it to the adjacent ACC. On the other hand, as the information contained within "FO" is critical, the Aircraft Operator would express an *integrity* security need when providing ACC with "IFPL" **(Ex2)**. Finally, the Aircraft Operator wants to ensure that "IFPL" information (intangible resource) is used to "Control" by the ACC, expressing in this way a *need-to-know* security need **(Ex3)**.

2.4 Deriving Security Requirements in Terms of Commitments

SecCo represents security requirements as commitments between actors. We are reasoning on a role-based perspective, since we do not know who the actual participants at run-time are going to be. Therefore the commitments are between roles, implying that at run-time the actual agents playing those roles, are expected to make and comply with those commitments. Whenever a security need is specified over an interaction, say over a goal delegation, by the delegator, the delegatee is expected to make a commitment on the opposite direction for that security need, promising it will fulfill it (similarly for resource provision, or granting authorization). If all agents playing those roles comply with their commitments, the security needs will be guaranteed.

Security requirements are automatically derived from the business view. We sketch some security requirements derived from the scenario in Section 2.3 related to the security needs Ex1-Ex3. In the commitments below, debtor and creditor are roles, whereas antecedent and consequent are propositions.

Ex1. The non-repudiation security need results in a commitment from the adjacent area control center (aacc) to SWIM infrastructure (swim) that, if goal "Replica updated with shoot info" is delegated to (aacc), it (aacc) will not repudiate the delegation:
C(aacc, swim, d_1 =delegate(swim,aacc,Replica updated with shoot info), non-repudiation(d_1))

Ex2. The integrity security need expressed by the Aircraft Operator (ao) results in an unconditional commitment made by ACC (acc) to (ao) that IFPL information will not be modified:
C(acc, ao, $\top$, integrity(IFPL))

Ex3. The need-to-know security need results in a commitment from the ACC (acc) to Aircraft Operator (ao) that it will not access the IFPL unless it is used for the goal "Control":
C(acc, ao, need-to-know(IFPL, Control, u $\wedge$ d))

3 Transferring SecCo Requirements to BPMN

The modelling presented in section 2 relates to business and/or operational modelling using a very simple language (in terms of concepts and notation) to express business/operational goals. It is easily understandable and accessible to decision makers who are not security experts. The modelling captures only what is important for the business or operation (i.e. goal-level), not how this business or operation needs to be conducted (i.e. process-level). Compared to BPMN, SecCo is therefore at a higher level of abstraction. We will now show how the derived commitments representing security requirements can be annotated into BPMN. We do this to guide the process modelling, but also to make these commitments a part of the specifications themselves.

BPMN 2.0 has four different diagram types, where the conversation diagram gives the most abstract view [11]. Its purpose is to give an overview of inter-company processes between several partners. Hence, we can annotate to which conversations and related participants the SecCo requirements apply. This is shown in figure 4, where SWIM, the flight pilot, the controlling ACC and the adjacent ACC need to co-operate in order to achieve a hand-over. Here, all three commitments (Ex1, Ex2 and Ex3) must be taken into consideration. BPMN participants can be mapped directly towards the SecCo actors, and conversations towards top goals. The security annotation has been manually added to the hand-over conversation guided by the ATM top-level goal. However, these security annotations might be too coarse grained in many situations, so we might have to dive a bit deeper to make the commitments more explicit.

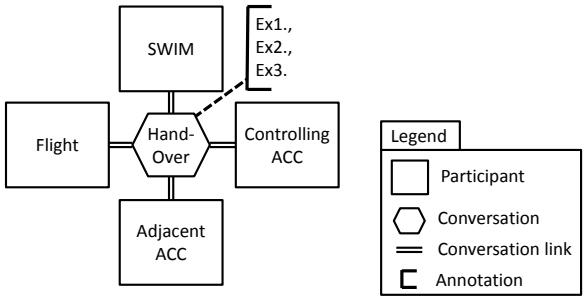


Fig. 4: A high-level conversation diagram for the handover scenario

The more detailed interactions can be represented using choreography or collaboration diagrams of BPMN. With choreographies, we are not interested in the partners' internal processes, but mainly the message exchange between them. According to Allweyer [2], choreographies are better than collaborations as a "basis for agreements and contracts between parties". In figure 5, we have created a choreography diagram for the hand-over scenario. First, the controlling ACC initiates a message exchange with SWIM when one of its flight is getting close to the airspace border and releases control over the FO. SWIM identifies which adjacent ACC to contact and relays the hand-over request to this site (Ex2 applies here). Notice that there are two logical outcomes of this last message exchange:

1. If the adjacent ACC refuses (this could be due to capacity problems), SWIM must inform the previously controlling ACC about this so that it can take back the control. The internal process of the controlling ACC (not shown in this figure) would then be to contact the flight and change the business trajectory of the FO. The process of requesting hand-over to another ACC would then be repeated.
2. The other outcome is that the adjacent ACC accepts the hand-over request. The following choreography would then be between the message exchange of these three parties related to the change of controlling ACC role.

4 Discussion and Further Work

The SecCo framework is meant to be an easy and intuitive way of obtaining security specifications in terms of formal commitments based on expressed security needs. It targets a non-scientific and non-technical population, such as commerce, marketing, pre-sale and business development staff of an organization. In this paper we have, through a scenario extracted from an industrial case study, shown the creation of commitments, and then (manually) transferred these commitments to BPMN models by system designers. Non-technical staff,

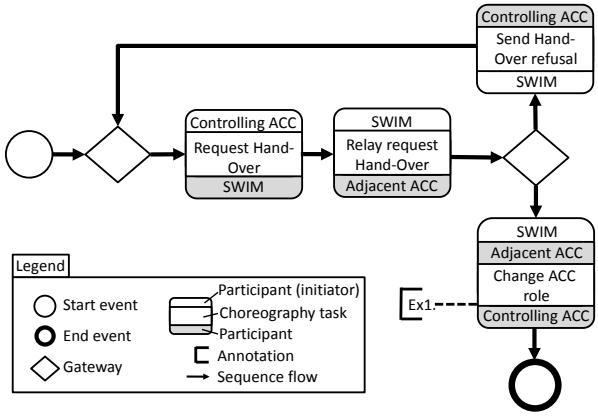


Fig. 5: A choreography diagram of the hand-over scenario

mainly business people or development engineers, offers support to identify the security properties using SecCo (see figure 6). At the time being, the process of mapping security specifications to security properties for business processes and the annotation of business process models with security properties is done manually.

In practice, our approach is not a one-way flow, as one often will identify new security goals when creating business processes (or the business processes might already exist beforehand). Another challenge is that of preserving a distinct separation between these two modelling activities. For instance, there is a danger that people try to express too much of the *process* with SecCo. This is not the intention here, and SecCo is not a process or business-oriented language, it operates at a higher level of abstraction and as a result does not contain enough constructs to have the same expressive power as for instance, BPMN. With SecCo we are stating what is important, and what needs to be protected.

Associated to the SecCo modelling notation is also a methodology that guides the modeller in eliciting and capturing the precise security criteria (including, but not limited to confidentiality, availability and integrity) that apply to the goals and/or resources that need protection.

We are currently working on the transformation of the security requirements as expressed with SecCo towards lower level languages aiming at service engineering. The commitments represent a powerful concept that should allow to enact the security at runtime through mechanisms such as security-by-contract. The transformations are not yet finalised as we are also analysing the necessity to include some risk assessment steps between the SecCo modelling and the lower level modelling, including BPMN. The BPMN examples shown in this paper are conversation and choreography diagrams, but for finer-grained commitments, it would be natural to also make use of collaboration and process diagrams, for instance to add commitments related to tasks or data object within a process.

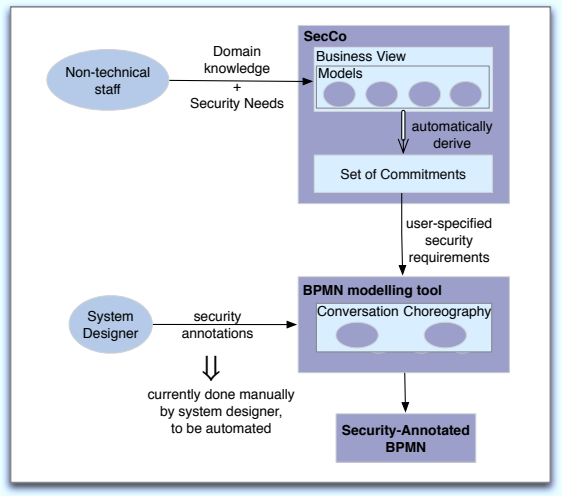


Fig. 6: Our approach to security-annotated BPMN

5 Conclusions

The need to take into account security issues when modelling business processes has been acknowledged by different research works, providing extensions to, for instance, BPMN [14, 18, 10] for security configurations. These approaches show the necessity to enrich BPMN annotation to support the specification of security requirements. However, existing approaches lack of the perspective of the security analyst as well as of a thorough analysis of the organizational setting. Security requirements are expressed considering traditional security properties and mechanisms.

In this work, we presented an approach to derive security requirements by modelling the organizational objectives of the involved parties and the social interactions that emerge between them. Our modelling framework allows to make a thorough analysis of the organizational setting, following the perspectives of different participants, their business goals and the interdependencies among them. The framework allows the various interacting parties to constrain the interaction by expressing security needs, which are later operationalized in security requirements via social commitments. It is in this latter concept that resides the strength of our approach.

Acknowledgements This work has been partially supported by the EU-FP7-IST-IP-ANIKETOS and EU-FP7-IST-NOE-NESSOS projects.

References

1. R.S. Aguilar-Saven. Business process modelling: Review and framework. *International Journal of Production Economics*, 90(2):129–149, 2004.
2. Thomas Allweyer. *BPMN 2.0.* BoD, 2010.
3. Aniketos. Deliverable 6.1: Initial analysis of the industrial case studies, 2011.
4. M. Backes, B. Pfitzmann, and M. Waidner. Security in business process engineering. *Business Process Management*, pages 1019–1019, 2003.
5. P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.
6. E. Cardoso, J.P.A. Almeida, R.S.S. Guizzardi, and G. Guizzardi. A method for eliciting goals for business process models based on non-functional requirements catalogues. *International Journal of Information System Modeling and Design (IJISMD)*, 2(2):1–18, 2011.
7. D. G. Firesmith. Security Use Cases. *Journal of Object Technology*, 2(3):53–64, 2003.
8. U. Greiner, S. Lippe, T. Kahl, J. Ziemann, and F.W. Jäkel. Designing and implementing cross-organizational business processes-description and application of a modelling framework. *Enterprise Interoperability*, pages 137–147, 2007.
9. J. Jürjens. Umlsec: Extending uml for secure systems development. *UML 2002The Unified Modeling Language*, pages 1–9, 2002.
10. M. Menzel, I. Thomas, and C. Meinel. Security requirements specification in service-oriented business process management. In *2009 International Conference on Availability, Reliability and Security*, pages 41–48. IEEE, 2009.
11. OMG Document Number:formal/2011-01-03. Business process model and notation (bpmn) version 2.0, 2011.
12. OASIS. Reference Architecture Foundation for Service Oriented Architecture, Version 1.0, Organization for the Advancement of Structured Information Standards. 2009.
13. C.J. Pavlovski and J. Zou. Non-functional requirements in business process modeling. In *Proceedings of the Fifth Asia-Pacific Conference on Conceptual Modelling-Volume 79*, pages 103–112. Australian Computer Society, Inc., 2008.
14. A. Rodríguez, E. Fernández-Medina, and M. Piattini. A bpmn extension for the modeling of security requirements in business processes. *IEICE Transactions on Information and Systems*, 90(4):745–752, 2007.
15. G. Sindre. Mal-activity diagrams for capturing attacks on business processes. *Requirements Engineering: Foundation for Software Quality*, pages 355–366, 2007.
16. G. Sindre and A.L. Opdahl. Eliciting security requirements with misuse cases. *Requirements Engineering*, 10(1):34–44, 2005.
17. M. P. Singh. An Ontology for Commitments in Multiagent Systems: Toward a Unification of Normative Concepts. *Artificial Intelligence and Law*, 7(1):97–113, 1999.
18. C. Wolter, M. Menzel, and C. Meinel. Modelling security goals in business processes. *Modellierung 2008*, 127:201–216, 2008.
19. E. Yu. *Modelling Strategic Relationships for Process Reengineering.* PhD thesis, University of Toronto, Canada, 1996.