



UNIVERSITÀ DEGLI STUDI
DI TRENTO

DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE
ICT International Doctoral School

SECURE BUSINESS PROCESS
ENGINEERING:
A SOCIO-TECHNICAL APPROACH

Mattia Salnitri

Advisor

Prof. Paolo Giorgini

Università degli Studi di Trento

April 2016

Abstract

Dealing with security is a central activity for today's organizations. Security breaches impact on the activities executed in organizations, preventing them to execute their business processes and, therefore, causing millions of dollars of losses. Security by design principles underline the importance of considering security as early as during the design of organizations to avoid expensive fixes during later phases of their lifecycle. However, the design of secure business processes cannot take into account only security aspects on the sequences of activities. Security reports in the last years demonstrate that security breaches are more and more caused by attacks that take advantage of social vulnerabilities. Therefore, those aspects should be analyzed in order to design a business process robust to technical and social attacks. Still, the mere design of business processes does not guarantee that their correct execution, such business processes have to be correctly implemented and performed.

We propose SEcure Business process Engineering (SEBE), a method that considers social and organizational aspects for designing and implementing secure business processes. SEBE provides an iterative and incremental process and a set of verification of transformation rules, supported by a software tool, that integrate different modeling languages used to specify social security aspects, business processes and the implementation code. In particular, SEBE provides a new modeling language which permits to specify business processes with security concepts and complex security constraints.

We evaluated the effectiveness of SEBE for engineering secure business processes with two empirical evaluations and applications of the method to three real scenarios.

Keywords

[Security, business processes, social/organizational security aspects, automated reasoning, automated implementation.]

Acknowledgements

This thesis would not have been possible without the help of several different people. I would like to thank Professor Paolo Giorgini, my advisor, for the continuous support of my PhD study, his precious comments and critics. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist.

Thanks to the professors who served on my thesis committee: Professor Achim Brucker (University of Sheffield), Professor Haralambos Mouratidis (University of Brighton), Professor Luca Spalazzi (Universita politecnica delle Marche) and Professor Angelo Susi (Fondazione Bruno Kessler).

A special tank to my main co-authors Dr. Elda Paja and Professor Fabiano Dalpiaz. You played an essential role during my PhD, I would not have reached this achievement without your contributions and help.

Many thanks to Mauro Poggianella, the developer of SecBPMN2 and STS-Tool software tools, without your work my thesis would have been only a theoretical study.

I would like to thank Professor Achim Brucker, Professor Emil Lupu, and Professor Raian Ali for offering me internship opportunities in their groups and leading me working on diverse exciting projects.

I am very thankful to the software engineering research group of Trento. The stimulating discussions and the seminars were central for the development of the initial idea in my thesis. I apologize for not listing your names, but I am afraid to miss someone.

Thanks to my parents, Giuseppe and Sonia, and my sister Iris, for being always proud and for the unconditional support. I can't thank you enough for encouraging me throughout this experience.

Last but not least thank you Elena, my love, for the great patient and unconditional love. Thank you for always believing in me and supporting me in the difficult parts of this experience.

To all of you, *Grazie mille.*

Contents

1	Introduction	1
1.1	Security and business processes	1
1.2	A method to engineer secure business processes	7
1.3	Research questions and success criteria	8
1.3.1	Research questions	8
1.3.2	Success criteria	10
1.3.3	Evaluation	12
1.4	Research contribution	13
1.5	Running example	15
1.6	Organization of the thesis	16
1.7	Published papers	18
1.7.1	Book chapters	18
1.7.2	International journals	18
1.7.3	International conferences	18
1.7.4	International workshops and forums	19
1.7.5	Under preparation	20
2	Related work	21
2.1	Approaches for business process engineering	24
2.2	Social/organizational approaches	27
2.3	Approaches for secure business processes	30
2.3.1	Business process modeling languages with security concepts	30
2.3.2	Query languages	32

2.4	Transformation approaches	35
2.4.1	From social/organizational to business process models	36
2.4.2	From business process models to implementation .	37
2.5	Chapter summary	39
3	Baseline	41
3.1	Representing social/organizational aspects with STS-ml . .	41
3.1.1	Multi views approach	41
3.2	Querying business processes with BPMN-Q	43
3.3	A reference model for security concepts	46
3.4	Representing business artifacts with River Definition Language	47
3.5	Chapter summary	48
4	A process for secure business process engineering	49
4.1	Phases	51
4.1.1	Phase 1	51
4.1.2	Phase 2	52
4.1.3	Phase 3	52
4.1.4	Phase 4	53
4.1.5	Phase 5	53
4.1.6	Phase 6	53
4.2	Chapter summary	54
5	SecBPMN2	55
5.1	SecBPMN2: principles	55
5.2	BPMN 2.0	56
5.3	Security annotations	61
5.4	SecBPMN2-ml	70
5.5	SecBPMN2-Q	73
5.6	Verifying security policies	75
5.7	Chapter summary	80

6	Connecting multiple perspectives with SEBE	83
6.1	Designing secure business processes	83
6.1.1	Mapping STS-ml concepts to SecBPMN2 concepts .	84
6.1.2	Transforming STS-ml models into SecBPMN2 models	85
6.1.3	Security policies generation	90
6.2	Implementing secure business processes	92
6.2.1	Mapping SecBPMN2 concepts to RDL concepts . .	94
6.2.2	Transforming SecBPMN2 models into RDL scripts	95
6.2.3	Enforcing SecBPMN2 security specifications	96
6.3	Chapter summary	102
7	Software support for SEBE	103
7.1	Consistency verification	105
7.2	Automated generation of secure business processes and se- curity policies	106
7.3	Procedural security policy verification	107
7.4	Generation of RDL code	112
7.5	Architecture	113
7.6	Chapter summary	115
8	Empirical evaluations	117
8.1	Evaluation of SecBPMN	117
8.1.1	Experimental design	118
8.1.2	Participants	120
8.1.3	Experiment results	120
8.1.4	Threats to validity	122
8.2	Evaluation of SEBE	124
8.2.1	Interviews	124
8.2.2	Empirical evaluation	125
8.2.3	Experimental design	126
8.2.4	Participants	127
8.2.5	Quantitative data analysis	128

8.2.6	Qualitative data analysis	131
8.2.7	Threats to validity	133
8.3	Chapter summary	134
9	Case studies and application scenarios	137
9.1	Case study settings	137
9.2	Ospedale Pediatrico Bambin Gesù'	139
9.2.1	Outcomes of the case study	140
9.3	Ministero dello Sviluppo Economico	141
9.3.1	Outcomes of the case study	142
9.4	Athens municipality	143
9.4.1	Outcomes of the case study	144
9.5	Comparison of case study results	145
9.6	Chapter Summary	146
10	Conclusions and future work	149
10.1	Fulfillment of success criteria	149
10.1.1	Success Criteria 1	149
10.1.2	Success Criteria 2	151
10.1.3	Success Criteria 3	151
10.1.4	Success Criteria 4	152
10.1.5	Success Criteria 5	152
10.1.6	Success Criteria 6	153
10.2	Conclusion and limitations	153
10.2.1	Limitations	155
10.3	Ongoing works	156
10.4	Future research work	157
	Bibliography	159
A	Transformation rules for STS-ml elements	179
B	Transformation rules for security requirements	183

List of Tables

1.1	Success criteria	10
1.2	Evaluation of success criteria	13
3.1	Security concepts covered by the RMIAS reference model [26]	46
5.1	Security annotations in SecBPMN2: predicates and their graphical syntax	64
6.1	Default SecBPMN2-ml transformation rules provided in SEBE	90
6.2	Default SecBPMN2-Q transformation rules provided in SEBE	93
7.1	List of transformations of SecBPMN2 concepts in \mathcal{K} predicates	111
8.1	GQM template for our experiments	118
8.2	Hypotheses of the experiments	119
8.3	GQM template for our experiments	125
8.4	Data about the operations executed by subjects during the modeling	130
8.5	Data about the usage of SEBE	130
8.6	Research questions outcomes	133
9.1	Elements of STS-ml and Secure BPMN 2.0 (SecBPMN2) used to specified the case studies	144
9.2	Case studies comparison	146
10.1	Thesis map	150

A.1	Default transformation rules provided in SEBE to transform Socio-Technical Security - modeling language (STS-ml) into Secure BPMN 2.0 - modeling language (SecBPMN2-ml) models	179
A.2	Default transformation rules provided in SEBE to transform STS-ml actors into SecBPMN2-ml models	180
A.3	Default transformation rules provided in SEBE to transform STS-ml goals and relation between goals and documents into SecBPMN2-ml models	181
B.1	Default transformation rules for security requirements on delegations	184
B.2	Default transformation rules for redundancy security requirements	185
B.3	Default transformation rules for security requirements on transmissions	186
B.4	Default transformation rules for authorizations	187
B.5	Default transformation rules for compatible and incompatible requirements	188

List of Figures

1.1	Components of SEBE.	15
3.1	STS-ml diagram representing social/organizational model of PE.	44
3.2	Example of BPMN-Q query	45
3.3	The SAP RDL Platform	47
4.1	Overview of the SEBE process.	50
4.2	The process proposed with SEBE.	52
5.1	Example of SecBPMN2 model.	58
5.2	Examples of Secure BPMN 2.0 - Query (SecBPMN2-Q) security policies	75
6.1	Mapping of STS-ml, SecBPMN2 and River Definition Language (RDL) concepts.	86
6.2	Example of a transformation rule and its application.	88
6.3	A SecBPMN2 model representing part of a PE business process	96
7.1	Screenshots of STS-Tool.	105
7.2	STS-Tool components	115

Chapter 1

Introduction

Dealing with security is an extremely important activity for today's organizations [17]. Security breaches impact on the activities and on the business processes of such organizations and, consequently, they cause millions of dollars of losses. But an analysis of the business processes executed in organizations is not enough for avoiding security issues. Security reports over the past decades demonstrate that more and more breaches occur because of trusted users, insiders [2, 77]. As such, dealing with security requires an analysis of the social and organizational aspects of organizations, along with a business process one. However, these analyses alone do not guarantee organizations to be secure, the implementation of business processes should enforce security constraints in order to avoid security issues.

In this chapter we discuss the problem of engineering secure business processes. Specifically, we propose a method to design secure business processes, considering social and organizational security aspects of the organizations, and to implement such processes.

1.1 Security and business processes

Security is a central aspect for many organizations. This is demonstrated by the interest and investments organizations put on security. According to PricewaterhouseCoopers's (PWC) 2014 report [1], 36,25% of the organizations examined invested between 50.000 and 1 million of dollars and

another 31% invested more than 1 million of dollars.

Such investments are justified by the impact of security breaches in the activities and the business processes, i.e., sequences of activities, executed by organizations to deliver their products or services. Although huge investments are made to deal with security, still during 2014 40.73% of the organizations, which reported security issues, lost between 50.000 and 1 million dollars, with another 31.61% reporting losses of more than 1 million dollars.

The consequences of security leaks are not limited to monetary losses due to inability of organizations to execute their business processes. Other negative impacts include: (i) loss of customers: since the loss of money or sensitive data may convince customers to move to other systems; (ii) legal consequences: some organizations must guarantee a level of security prescribed by the local legislation; otherwise in case of a security breach, offended parties might sue the organizations; (iii) loss of reputation: whenever there is a security leak the organization will lose the appealing for new users or customers.

For example, in 2008 Heartland Payment Systems (Heartland), a company with over 250,000 clients that process more that \$4.2 billion transactions annually, reported a security breach that was very costly for the company. It lost 50% of its market capitalization and it spent more than \$32 million on legal fees and other related settlement costs [25].

Security by design principles strongly encourage to consider security as early as possible [36] during the design of organizations, to minimize the costs of modifications that, otherwise, would be order of magnitudes more expensive [80]. Therefore, security shall be taken into account as early as during the design of business processes, instead of during their implementation.

Specifically, in order to avoid security issues and their consequences, business processes should be designed to be compliant with security requirements, which specify security needs and constraints, defined by stakeholders of organizations. Even if only one business process allows a com-

bination of execution of actions that violates one or more security requirements, the overall organization may become vulnerable to attacks. For example, Heartland, as an organization who manages sensitive data of its customers, must protect the data stored and not disclose them. If a business process executes a routine that publishes all the sensitive data collected, then such single execution will lead to the infringement of laws, with penal consequences and monetary losses. These consequences will prevent the execution of other business processes of the company, causing additional monetary losses and other consequences.

Multiple research work considered the design of business processes compliant with security requirements an extremely relevant problem [81]. One of the most promising research directions consists of representing security concepts with modeling languages based on business process. However, such modeling languages are focused on specific application areas since the interpretation of security concepts highly depends on the context where the business processes are executed. Examples of these languages are SecureBPMN [24] that extends Business Process Management and Notation (BPMN) [86] with concepts that permit to specify access controls constraints; and UMLSec [57] that extends activity diagrams [38] to represent security concepts. Unfortunately, no modeling language covers enough security concepts to be used as a general tool for designing secure business processes of organizations. Moreover, most of the languages have a complex syntax that generates diagrams difficult to understand when real scenarios, which are composed by tens of business processes with hundreds of activities, are modeled.

Social and organizational security requirements define complex constraints that cannot be expressed with atomic security concepts in business processes, instead, they correspond to patterns and security constraints on such processes, from now on security policies. Such security policies have to be defined and verified against business processes to avoid security violations. In the Heartland example, the security issue was caused by a SQL injection that may be prevented if a security policy, which constraints the

authentication process to be executed every time employees access customers' information, would have been specified and verified.

The verification of security policies is a time consuming and error-prone operation because organizations' security policies may be complex, with many interdependent constraints, that have to be verified against huge business processes with hundreds of elements. Therefore, the verification should be automated in order to avoid error, while checking very complex business processes.

Some approaches have been proposed for the representation and verification of security policies against business processes. For example, BPMN-Query (BPMN-Q) [9] is an intuitive modeling language for graphically expressing policies, but it has a limited expressiveness and it is not focused on security. Other works use a formal approach to formulate policies using logical and/or formal languages, such as the approach proposed by Liu et al. [68] or FPSPARQL [16]. Those approaches permit to specify a wide range of policies, however, the complexity of the languages prevents their application in real scenarios. There is no modeling language which is expressive enough to specify security policies of real scenarios that are easily understandable in such complex cases.

The design approaches we examined consider security concepts in business processes, such as confidentiality, availability and integrity, from a technical perspective, i.e., as security concepts related to security mechanisms, for example, the cryptographic protocol used to encrypt messages exchanged between two services of the organization, or the key length set for the hash calculated for a document. Those approaches do not consider the social context in which business processes are executed.

Organizations are composed of people and organizational units, aside from technical components, that interact and collaborate as autonomous actors to achieve common objectives. This forms what in the last years has been defined as socio-technical systems [30, 117]. For example, a payment system, as Heartland, is considered a socio-technical system because customers, banks and parts of the system which perform the payment

collaborate and exchange information, for the shared goal of transferring values.

In socio-technical systems part of business processes are executed by social actors, such as employees, or other organizational units. Therefore, a technical analysis is not enough to avoid security breaches: an analysis of the organization, which considers how executors of business processes relate with each other, and the social aspects, which takes into account why they interact with each other, plays a fundamental role in the design of business processes.

For what concerns security, the design of business processes compliant with social and organizational security requirements, from now on secure business processes, requires an analysis of the relations and objectives of actors within a organization. Referring to PWC report on Cybersecurity of 2015 [2], during last year 64% of all security risks are attributed to organizations employees (34% from current employees and 30% from former employees).

Most importantly the social and organizational (social/organizational from now on) analysis should be done as early as during the requirement engineering phase [37] to avoid expensive fixes and conflicts with existing business processes executed in the organization. In other words, first a social/organizational analysis should be conducted to identify stakeholders' security requirements of the organization, then business processes shall be designed.

The business processes, therefore, should fulfill the social/organizational security requirements, otherwise security problems will arise when they are implemented and then executed. The verification of security requirements against business processes requires the transformation of social/organizational concepts in business process terms. For example, the objectives of actors of organizations, specified in social/organizational analyses, are not considered in business processes, therefore, security requirements on objectives have to be transformed in security policies that can be verified in business processes.

Only few, preliminary, research approaches face the transformation of social/organizational security concepts to business process concepts. Argyropoulos et al. [4,5] proposed a method for deriving business processes from social/organizational models, however the security concepts in the source and target languages missed precise definition of the abstract syntax of security concepts.

The social/organizational perspective can be used to design secure business processes, however, if such processes are not correctly implemented, the organization will not benefit of the security analysis conducted. This may happen for multiple reasons. Developers, who will implement the automated part of business processes, might misunderstand the specifications or, in some cases, deliberately deviate from the blueprint. These deviations results in unexpected behaviors that may lead to security vulnerabilities.

For these reasons business process designers, i.e., the business process experts in charge of defining business processes of organizations, should guide the implementation of the business processes, and check if the behavior of the implementation deviated from what is expected.

One promising research direction consists in generating the software code from business processes, as proposed by Kim et al. in [58]. Francesco-marino et al. [46] analyze the logs of the system, and check if they are consisted with the specified business processes. Brucker and Hang [23] proposed a framework to statically check source code of the implementation against constraints defined in a procedural modeling language focused on security. However, all research work we analyzed are limited on the type of checks they verify against business processes or they are too complex to be used in real case scenarios.

Concluding, business process designers deal with many challenges when engineering secure business processes, the most relevant being:

- define business processes which take in consideration security concepts;
- use social/organizational perspective of the system, to design business

processes;

- verify that the business process fulfill security requirements;
- generate an unambiguous specification for the implementation of secure business processes.

No approach assists business process designers for the design of secure business process, using the outcome of social/organizational analysis, until their deployment.

1.2 A method to engineer secure business processes

In order to tackle the aforementioned problems, this thesis proposes SEBE (SEcure Business process Engineering), a method to engineer secure business processes that supports business processes designers in designing secure business processes and ensuring their correct development.

SEBE uses a social/organizational analysis to consider the organizational structure of the organization, the rationale behind its actors and their interactions. Such analysis of the organization supports stakeholders to specify the security requirements a secure business process should enforce. For example, in a complex systems such as Heartland, with thousands of customers, that was created with the acquisition of smaller organizations, hundreds of business processes are executed. This complexity caused an erroneous analysis of security requirements that led to monetary losses and penal consequences. A social/organizational analysis would have helped to specify relations between components, to highlight inconsistencies on security requirements and it would have led to a correct specification of the security requirements and, therefore, to a secure design of the Heartland system.

SEBE provides an expressive modeling language to specify secure business processes and security policies. The modeling language permits to specify an exhaustive range of security concepts, is not limited to specific

contexts, as many of the current approaches are. It is graphical, easy to use and scalable with the size of real case scenarios.

Moreover, the method is provided with a software tool, which permits an automated verification of the security policies against secure business processes. This allows business process designers to easily check whether business processes are compliant with procedural and security constraints specified in security policies.

In order to guarantee the enforcement of security requirements, SEBE generates specifications for the developers of the part of secure business processes that have to be implemented. Specifically, the method generates part of the the implementation of the automated task of the business processes, which already enforces the security specification defined in the processes. The implementation will then be completed by the developers on the missing parts.

Concluding, the thesis proposes SEBE: a comprehensive method that supports business process engineers in engineering secure business process. The method exploit a social/organizational analysis to design secure business processes and it generates part of the implementation of the secure business processes.

1.3 Research questions and success criteria

In the following we specify the research questions of this thesis, the success criteria we consider for each of them and the evaluation method we used to check the achievement of success criteria.

1.3.1 Research questions

On the whole, our objective is that of helping business process designers in engineering secure business processes. We, therefore, define the following Research Questions (RQs).

RQ1. How to represent secure business processes?

RQ1 consists in finding an appropriate graphical modeling language for business processes with security concepts. This includes the specification of the sequences of actions and security concepts.

RQ2. How to represent security policies?

RQ2 consists in finding a representation of security policies which should be expressive enough to be used by business process designers to specify security constraints on secure business processes.

RQ3. How to verify security policies against secure business processes?

RQ3 requires an approach that permits to verify constraints specified in security policies against secure business processes. The verification should be automated to increase the scalability and to allow to analyze real, frequently complex, scenarios.

RQ4. How the social/organizational perspective can be used to design secure business processes?

RQ4 is about the connection between social/organizational perspective secure business processes. The answers of this research question covers: (i) how social/organizational concepts are mapped to business process concepts; (ii) how social/organizational security requirements are enforced in secure business process models.

RQ5. How to generate an implementation that enforces security concepts specified in secure business processes?

RQ5 covers the generation of implementation of secure business processes. Specifically, it covers how the business process concepts are mapped to implementation concepts, and how security concepts of secure business process are enforced in the implementation.

RQ6. How can we guide business process designers in the creation of secure business processes and their implementation?

RQ6 asks for a process that guides the business process designers for engineering secure business processes. From our experience, business process designers needs rules and best practices in order to effectively use the results produced for the previous research questions.

1.3.2 Success criteria

We defined Success Criteria (SC) to determine when a research question can be considered achieved. Table 1.1 lists the success criteria associated to each research question.

Research question	Success criteria
<i>RQ1</i>	<i>SC1</i> Have a modeling approach that can be used to: <i>SC1.1</i> represent business processes <i>SC1.2</i> represent security concepts of business processes <i>SC1.3</i> formalize business processes and security concepts
<i>RQ2</i>	<i>SC2</i> Have a modeling approach that can be used to: <i>SC2.1</i> represent security policies <i>SC2.2</i> formalize security policies.
<i>RQ3</i>	<i>SC3</i> Have an analysis approach that: <i>SC3.1</i> verifies procedural patterns against business processes <i>SC3.2</i> verifies security constraints <i>SC3.3</i> automates analysis
<i>RQ4</i>	<i>SC4</i> Have a systematic approach that: <i>SC4.1</i> helps deriving business process from organizational concepts <i>SC4.2</i> checks the enforcement of security requirements
<i>RQ5</i>	<i>SC5</i> Have a systematic approach that: <i>SC5.1</i> provides transformation rules to generate implementation <i>SC5.2</i> automatically generate implementation
<i>RQ6</i>	<i>SC6</i> Have a systematic approach that: <i>SC6.1</i> provides a process to be followed by the user of the method <i>SC6.2</i> provides a software tool that supports the process

Table 1.1: Success criteria

SC1 specifies that SEBE should include a modeling approach which permits business process designers to graphically *represent business processes* (SC1.1) in order to specify secure business process models, *represent security concepts of business processes* (SC1.2) in order to specify such concepts in the graphical representation of secure business process models, and *formalize business processes and security concepts* (SC1.3) in order to detect inconsistencies and conflicts in secure business process models.

SC2 specifies that method should include a modeling approach that

represents security policies (SC2.1) in order to permit business process designers to specify constraints on the flow of activities and security aspects in terms of business process concepts. Moreover, the approach should permit to *formalize security policies* (SC2.1) in order to detect conflicts and inconsistencies.

SC3 indicates that SEBE should help business process designers to analyze various aspects of the business processes. Specifically, it should allow to *verify procedural patterns against business processes* (SC3.1), i.e., it should help designers to check if the constraints on the flow of activities specified in the security policies are respected in the business processes analyzed. The method should permit to *verify security constraints* (SC3.2): it should help to verify that security constraints specified in the security policies are satisfied in the business processes analyzed. Moreover, it should *automate analysis* (SC3.3), which consist in providing a software tool which permits to execute the analysis and, therefore, permits to examine complex models as the one that can be frequently found in real scenarios.

SC4 points out that SEBE should include a systematic approach that uses social/organizational perspective for the design of secure business processes. Specifically it should help business process designers in *deriving business process from organizational concepts* (SC4.1), in order to facilitate the creation of secure business processes and give business process designers a starting point, i.e., initial business process models, that can be enriched. Moreover, it should *check the enforcement of security requirements in business processes* (SC4.2): SEBE should verify if the security requirements of stakeholders, specified at a social/organizational perspective, are implemented in the secure business processes enriched by the business process designers.

SC5 indicates that the method should provide a systematic approach that generates at least part of the implementation of the socio-technical system as the one of organizations. Precisely, the method should *provide transformation rules to generate implementation* (SC5.1) that allows the creation of part of the implementation, and it should provide a software

toolset which implements those rules to *automatically generate the implementation* (SC5.2).

SC6 specifies that the method should include a systematic approach that guides business process designers with rules and best practices to engineer secure business processes. Specifically, it should *provide a process that help business process designers* in using the theoretical and software tool provided for the research questions described above. Furthermore, the method should *provide a comprehensive software tool that supports all the steps of the process* in order to facilitate the application of the methods.

1.3.3 Evaluation

In order to assess if the success criteria are met by SEBE, we examined the outcomes of two empirical studies, three case studies and the formal definition of the abstract syntax of the modeling languages provided with the method. Table 1.2 lists the evaluations we used to decide the achievement of the success criteria.

We conducted two empirical experiments, one consisted in a controlled experiment, in which subjects used the method to perform exercises and then answered questionnaires, the other empirical experiment consisted in a online survey. In particular, we evaluated how secure business process and security policies are graphically represented (SC1.1, SC1.2 and SC2.1); the generation of business processes and the verification of security policies (SC4.1 and SC4.2); the process and the tool which support the process proposed in this thesis (SC6.1 and SC6.2).

The method we proposed, provides a modeling language for secure business process and security policies with the abstract syntax formally specified. This provides answers to SC1.3 and SC2.2.

We used three case studies, one of a hospital, one of a ministry and one of a municipality, to evaluate SEBE as a whole. We strictly collaborated with domain experts of the three case studies and we collected information of how they used the method and we examined the diagrams

generated. In particular we evaluated how secure business process and security policies are represented (SC1.1, SC1.2 and SC2.1). We use the case studies to check if security policies are correctly verified against business processes, this includes the verification of the control flow (SC3.1), the verification of security part (SC3.2) and the automated software support (SC3.3). The same type of evaluation was used for checking if the method helps business process designers in designing business process starting from social/organizational information and in verifying the enforcement of security requirements (SC4.2). Moreover, the evaluation was used to check if the transformation rules used for generating the implementation are correct (SC5.1) and if they are correctly implemented (SC5.2).

	Empirical study	Formal syntax	Case study
SC 1.1	✓		✓
SC 1.2	✓		✓
SC 1.3		✓	
SC 2.1	✓		✓
SC 2.2		✓	
SC 3.1			✓
SC 3.2			✓
SC 3.3			✓
SC 4.1	✓		✓
SC 4.2	✓		✓
SC 5.1			✓
SC 5.2			✓
SC 6.1	✓		
SC 6.2	✓		

Table 1.2: Evaluation of success criteria

1.4 Research contribution

This thesis proposes SEBE, a method to engineering secure business processes using a socio-technical approach. It helps business process designers

in designing secure business processes, considering social and organizational perspectives, and in generating an implementation of such processes.

Figure 1.1 shows the components of the method. The method builds on existing components highlighted in yellow, i.e., STS-ml and RDL, while it extends the STS-Tool which is marked in light yellow. STS-ml [89] is a goal-based modeling language we used to analyze social/organizational security aspects of organizations, whereas RDL [109] is a script language we chose as implementation language for secure business processes. STS-Tool [106] is a software toolset, written in Java, which supports the specification of STS-ml diagrams.

The first contribution of this thesis is SecBPMN2, a modeling language we propose to specify secure business processes and security policies. The modeling language extends Business Process Management and Notation 2.0 (BPMN 2.0), the standard for modeling business processes, and it is provided with a formal definition of its syntax.

The second contribution consists in a set of transformation rules that permit to use the social/organizational perspective to design secure business processes and to generate part of the implementation of the socio-technical system of organizations from secure business processes.

The third contribution of this thesis is a process, which guides business process designers in engineering of secure business processes. The process is iterative and incremental and is used to direct method users from the social/organizational analysis of security, through the design of business processes, until their implementation.

The fourth contribution is an extension of STS-Tool that implements the transformation and verification rules, and verifies procedural security policies against business processes. The software extension provides full support to the method, including the process, the specification and analysis of SecBPMN2 and STS-ml diagrams and the coding of RDL applications.

The last contribution of this thesis consists in the evaluation of the success criteria with two empirical experiments and three case studies.

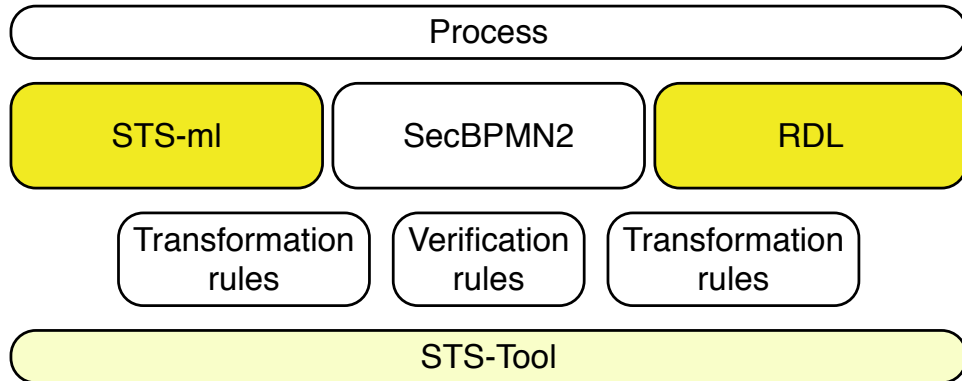


Figure 1.1: Components of SEBE.

1.5 Running example

SAP Payment Engine (PE) [108] is a software system created to perform payments for e-commerce shops or, more in general, for services that allow users to pay with electronic methods. Usually, to support the electronic payments, the e-shops implement interfaces to communicate with all the banks they intend to use as source/destination of the payments. But each bank requires a different set of protocols and security measures. Therefore the e-shops are forced to put a noticeable amount of effort to implement different interfaces, and for medium/small organizations it is not acceptable investing large quantity of time and money just to allow people to pay the goods/services they offer.

The PE minimizes such effort: it contains a set of pre-built interfaces with the most known banks in the world that can be used out of the shelf. E-shop programmers only have to create one interface to transmit the required data to the PE system.

Such system highly relies on security since the consequences of a security breach might be severe. Therefore, it is essential that each actor in such system maintains the level of security it committed, since a security breach in only one actor may compromise the entire system. For example, if a bank of the PA system transmits payment order with a channel not enough protected, all sensitive information in the transmitted documents

might be disclosed; moreover, the payment orders themselves might be modified. This will lead to catastrophic consequences for the PE system and the organization who runs it: customers will sue the organization, the organization will reimburse the modified transactions and the credibility of the system, and consequently of the organization, will drop.

We chose PE as a running example since is a real case study of a complex socio-technical system in which many autonomous actors interact, executing business processes, to achieve common objectives. The engineering of such system calls for a socio-technical approach, where social/organizational perspective is considered for the design and implementation of secure business processes executed to perform payment and handle sensitive data of customers and organizations.

1.6 Organization of the thesis

The thesis is organized as follows.

Chapter 2 describes the related work. We review the methodologies for business process engineering in Section 2.1, while we describe modeling languages for social/organizational security aspects in Section 2.2. Section 2.3 describes the most known modeling languages for secure business processes and security policies, while Section 2.4 reports the research approaches used to transform social/organizational models to business process models and the latter in implementation.

Chapter 3 presents the research work and the technologies on which SEBE is built. Section 3.1 provides a description of STS-ml, a modeling language for social/organizational security aspects. Section 3.2 introduces BPMN-Q, a query language for business processes, while Section 3.3 presents RMIAS: a reference model for security. Section 3.4 delineates RDL a script language for business artifacts and their business logic.

Chapter 4 presents the process provided with SEBE, specifically Section 4.1 describes all its phases.

Chapter 5 describes SecBPMN2 the modeling language for secure busi-

ness processes and security policies, provided with SEBE. Section 5.1 delineates its principles. Section 5.2 presents BPMN 2.0 the standard for representing business processes on which SecBPMN2 is based, while Section 5.3 describes the security concepts that are used in SecBPMN2. We describe the part of the modeling language for business processes in Section 5.4 and the part for modeling security policies in Section 5.5. Section 5.6 formally specifies how a security policies is satisfied in SecBPMN2 business processes.

Chapter 6 presents the transformation rules which permit to design SecBPMN2 business processes starting from a social/organizational modeling language (Section 6.1), and to generate part of the implementation from SecBPMN2 business processes (Section 6.2).

Chapter 7 describes STS-Tool, the software we created to support the users of SEBE. In particular, the chapter describes how the consistency of models is verified automatically (Section 7.1), how SecBPMN2 business processes are generated (Section 7.2) and how security policies are verified (Section 7.3). Moreover, Section 7.4 describes how part of the implementation is automatically generated, while Section 7.5 provides a description of the architecture of the software tool.

Chapter 8 provides a description of the empirical evaluations we conducted. Section 8.1 describes the empirical evaluation on SecBPMN while Section 8.2 reports the evaluation of the SEBE method as a whole.

Chapter 9 presents the application of SEBE to three case studies. Section 9.1 presents the settings of the case studies, while sections 9.2, 9.3 and 9.4 present the case studies and the results of the application of the method. Section 9.5 describes the results of the application of the three methods.

Chapter 10 concludes the thesis. Section 10.1 describe how the SC are fulfilled, while Section 10.2 highlights the limitations of the SEBE method. Section 10.3 delineates the ongoing work and Section 10.4 presents the future work.

1.7 Published papers

1.7.1 Book chapters

1. Erkuden Rios, Francesco Malmignati, Eider Iturbe, Michela D’Errico and *Mattia Salnitri* **From Consumer Requirements to Policies in Secure Services**. In *Secure and Trustworthy Service Composition: The Aniketos approach*. Pages 79 - 94, 2014.

1.7.2 International journals

2. Jennifer Horkoff, Tong Li, Feng-Lin Li, *Mattia Salnitri*, Evellin Cardoso, Paolo Giorgini and John Mylopoulos. **Using goal models down-stream: A systematic roadmap and literature review**. *International Journal of Information System Modeling and Design*. pages 1 - 42, 2015.
3. *Mattia Salnitri*, Fabiano Dalpiaz and Paolo Giorgini. **Designing secure business processes with SecBPMN**. *Software & Systems Modeling*, pages 1 - 21, 2015.

1.7.3 International conferences

4. Mohamad Gharib, *Mattia Salnitri*, Elda Paja, Paolo Giorgini, Haralambos Mouratidis, Michalis Pavlidis, Jose F. Ruiz, Sandra Fernandez, Andrea Della Siria. **Privacy Requirements: Findings and Lessons Learned in Developing a Privacy Platform**. To appear in proceeding of Requirement Engineering (RE), 2016.
5. Jennifer Horkoff, Fatma Basak Aydemir, Evellin Cardoso, Tong Li, Alejandro Mate, Elda Paja, *Mattia Salnitri*, John Mylopoulos, Paolo Giorgini. **Goal-Oriented Requirements Engineering: A Systematic Literature Map**. To appear in proceeding of Requirement Engineering (RE), 2016.

6. *Mattia Salnitri*, Achim Brucker and Paolo Giorgini **From Secure Business Process Models to Secure Artifact-Centric Specifications**. In proceeding of Business Process Modeling Development and Support (BPMDS) working conference. Pages 246 - 262, 2015.
7. Jennifer Horkoff, Tong Li, Feng-Lin Li, *Mattia Salnitri*, Evellin Cardoso, João Pimentel, Paolo Giorgini and John Mylopoulos **Taking Goal Models Downstream: A Systematic Roadmap In proceeding of Research Challenges**. In proceedings of Information Science (RCIS) conference. Pages 1 - 12, 2014.
8. *Mattia Salnitri*, Fabiano Dalpiaz and Paolo Giorgini **Modeling and Verifying Security Policies in Business Processes**. In proceeding of Business Process Modeling Development and Support (BPMDS) working conference. Pages 200 - 214, 2014
9. *Mattia Salnitri*, Fabiano Dalpiaz and Paolo Giorgini **Aligning Service-Oriented Architectures with Security Requirements**. In proceeding of On The Move (OTM) conference. Pages 232 - 249, 2012.

1.7.4 International workshops and forums

10. *Mattia Salnitri*, Elda Paja and Paolo Giorgini. **Maintaining Secure Business Processes in Light of Socio-Technical Systems Evolution**. To appear in proceeding of Model-Driven Requirement Engineering (MoDRE) workshop, 2016.
11. *Mattia Salnitri*, Elda Paja, Mauro Poggianella and Paolo Giorgini **STS-Tool 3.0: Maintaining Security in Socio-Technical Systems** In proceeding of Conference on Advanced Information System Engineering (CAiSE) Forum 2015 .
12. *Mattia Salnitri*, Elda Paja and Paolo Giorgini. **Preserving compliance with security requirements in socio-technical systems**. In Cyber Security and Privacy. Pages 49 - 61, 2014.

13. *Mattia Salnitri*, Paolo Giorgini **Transforming Socio-Technical Security Requirements in SecBPMN Security Policies**. In proceeding of IStar workshop 2014.
14. *Mattia Salnitri*, Paolo Giorgini **Modeling and Verification of ATM Security Policies with SecBPMN**. In proceeding of Security and High Performance Computing and Systems (SHCPS) workshop. Pages 588 - 591, 2014.

1.7.5 Under preparation

15. *Mattia Salnitri*, Elda Paja and Paolo Giorgini. **From Socio-Technical Requirements to Technical Security Design**.

Chapter 2

Related work

The literature offers many approaches that share part of the objectives with this thesis. This chapter analyzes the most relevant and prominent works.

Criteria and methods

We used a simple but effective method for describing the related work of SEBE. Initially we defined inclusion criteria as list of questions which covers the main aspects of SEBE. Then we classified the included papers in areas of interest.

Inclusion criteria

1. Does the approach cover security?
2. Does the approach cover business process engineering?
3. Does the approach cover the modeling of social/organizational concepts?
4. Does the approach cover the modeling of business processes?
5. Does the approach cover enforcement of requirements?

6. Does the approach cover the transformation of social/organizational concepts in business processes or the latter in the implementation code?

The first question allows to include approaches on security, we consider this question as a necessary but not sufficient condition, it applies with second, third, fourth and fifth questions. This means that it is not sufficient that an approach is about security to be considered a related work.

The second question includes approaches that propose methods to help the design, implementation and deployment of business processes.

The third question permits to include research work that propose or extend a modeling language which permits to specify social/organizational aspects.

The fourth question covers research work which proposed or extends a modeling language that permits to specify business processes.

The fifth question covers all research work which are focused on the implementation of requirements.

The sixth questions includes approaches that permit to transform models containing social/organizational aspects in models that contains business processes, or they permit to create at least part of the implementation of the system to be.

Areas of interest

We classified the papers we considered in areas of interest that we defined taking in consideration the inclusion criteria specified before. The areas of interest are:

1. approaches for engineering business processes;
2. approaches for modeling security concepts in social/organizational models;
3. approaches for modeling security concepts in business process models;

-
4. approaches for transforming social/organizational models in business process models or the latter in implementation code, and approaches for enforcing social/organizational security requirements or business process security specifications.

The first area of interest includes research work focused on providing methods and/or guidelines for helping the design, implementation and deployment of secure business processes. This area of interest is derived from inclusion criteria 1 and 2.

The second area of interest comprehends research work focused on modeling security concepts in social/organizational models, i.e., that match inclusion criteria 1 and 3.

The third area of interest includes all research work that are concerned about the modeling of security concepts in business process models, i.e., that match inclusion criteria 1 and 4.

The fourth area consists of approaches that transform social/organizational models in business process models or that they transform business process models in the implementation code. Specifically, this area comprehends research work focused on the realization of constraint of security concepts, i.e. research work that allow to actualize abstract security requirements. It includes approaches that enforce social/organizational security requirements in business process models, and business process security concepts in the implementation code. This area of interest is derived from inclusion criteria 5 and 6.

Selection criteria

The selection criteria consists in selecting approaches which permit a certain degree of scalability. This led to exclude non-graphical approaches for the specification of social/organizational and business processes. For example, textual approaches such as Based Requirements Analysis Method (GBRAM) [3], cannot be used in real scenarios where there are tens of actors, tens of social/organizational relations and hundreds of business

process. The language would lead to errors in the specification of such complex models, moreover, the specifications would be too difficult to read in these complex scenarios.

The rest of the chapter describes the research works we examined, organized by area of interest.

2.1 Approaches for business process engineering

In this section we describe the most known approaches for engineering business processes with a focus on research works on security.

The most know field is Business Process Management (BPM) [121]. This research area covers all the life cycle of a business process, which is split in four phases: design, implement/configure, run and adjust. The first phase consists in creating the business processes, the second phase in create a system which follow the specification of the business processes, the third phase consists in deploying the system and, therefore, executing the actions specified by the business process, the last phase consists in modifying the business processes to adapt in to new exigencies or changes in the environment.

In BPM area many approaches were proposed. One of the most known is Architecture of Integrated Information Systems (ARIS) [111,112] an approach to enterprise modeling [61]. The main objective of ARIS consists in implement business processes in information system. It is based on Event-Process Chain (EPC) [125] modeling language for business process. This method use an approach similar to the one proposed in this thesis: it divides the information used for the design of business process in separate perspectives: organizational, data, control and function. The organizational perspective is used to specify the organizational hierarchy of entities and the relations between them. The data perspectives includes the data object definition with the function used to manipulate them. The function perspectives is used to specify the processes executed in the organization analyzed. The control perspectives is used to connects the other perspec-

tives. ARIS is similar to our approach but it miss the security part in all perspectives, moreover, it considers only the organization hierarchy of actors leaving out the social part, i.e., the rationale behind actors processes and their interactions.

Business process design is a phase of BPM and consists in the definition of business processes. Numerous research work focus in business process design.

Bleistein et al. [19], for example, proposed a method to design business process starting from goal models. Their idea is based on problem frames [55], which separates in two *moods* the design of a system: *indicative*, which indicates the status of the world as it is, and *optative*, which indicates how to change the world. Bleistein et al. used a goal-based modeling language for the indicative mood and Role Activity Diagram (RDA), a business process language, for the optative mood. This approach is focused on the design of business process using social/organizational information (defined in the goal modeling language), but it is not focused on security and it miss the generation of a specification for the developers who will implement the defined business processes.

Backes at al. [11] present an approach for integrating security requirements to the development of business processes. Their proposal is focused on the connection between the business processes and the cryptography solution that can be used in the implementation. Even if the propose guidelines for the implementation of many security requirements, the approach is focused on cryptography. Moreover, it does not take into consideration the social/organizational information but only high-level of abstraction security requirements.

Gruhn and Laue [49] propose a heuristic approach for finding common design errors in business process models, represented using EPC [125]. They defined a set of rules to check if a business process is not sound or it matches some bad design patterns. Security experts could adopt this approach to verify the compliance of the business process model against a fixed set of rules. However, using a fix set of rules is a major limitation

when dealing with security policies, as it forces security experts to adopt an interpretation of security policies which may not fit the original policy.

Blanc et al. [18] propose an incremental inconsistency checker. Such framework is based on the hypothesis that the definition of a business process is an incremental task, and, thus, inconsistency checking shall be done incrementally. They offer a software tool, based on Prolog [27], which checks if a fixed set of well-formedness rules are satisfied by a business process model. The framework can be applied to any modeling language that can be translated in Prolog. The fixed set of queries is a major limitation, as it inhibits custom security policies.

BPM is an evolution of a research area called business process re-engineering, focused on the enhancement, and maintenance of already existing business processes. In this field few approaches of security were proposed but, mostly, they do not take into consideration social/organizational information, since their starting point is already existing business processes.

Herrmann and Herrmann [50] propose Modeling Security Semantics of Business Processes (MoSSBP) a method which guides security experts in reengineering businesses processes through graph rewriting techniques. The method allow to refine high-level of abstraction business processes through 4 levels where, at each level, more information are added. In the last level the implementation code is generated. The method takes into account procedural security policies and help the business process designer in checking if, in each level, the security policies are enforced correctly. This method lacks in the specification of the social/organizational information. Even if they authors claim that high level business process can capture organizational information, they still miss to model the social part of the system.

2.2 Social/organizational approaches

In this section we describe the most prominent approaches for modeling social and organizational aspects of socio-technical systems. We analyze modeling languages that capture security along organizational aspects such as business objectives, business strategies, social dependencies and organizational structures.

This perspective hides most of the technical details of socio-technical systems but offers an executive dashboard which permits to immediately identify conceptual errors or misbehaviors. From the security point of view this perspective is central since permits to specify abstract security requirements that can be understood by not-technical people [118].

McDermott and Fox [75] extend use cases modeling language with abuse cases. Their modeling language, called Abuse Case, allows to specify interactions between one or more malicious actors and the system where the results of the interactions are harmful to the system, one or more actors or the stakeholders of the system. However, the interactions are not well defined and they are more oriented to describe a sequence of high-level malicious activities. Moreover, this language is not focused on security, therefore, is not possible to specify security requirements.

Similarly, Sindre and Opdahl [116] extend use cases with misuse case, i.e., use case whose execution threat a use case, and two relations, threats and mitigates. The relations connect a misuse case to a use case; the former specifies that the use case is hindered by the misuse case, while the latter relation specifies that a use case is a countermeasure to a misuse case. The approach proposed by the authors is systematic and permits to specify misuse cases graphically and then with a table, to define more details. However, the authors did not focus on security aspects and, therefore, is not possible to specify security concepts.

UMLsec [57] extends Unified Modeling Language (UML) [20] with security requirements focusing mostly on authenticity, secrecy, and integrity. It permits to express security information within UML diagrams, and to ver-

ify if the security concepts are coherent with the UML design of the system. Security issues are analyzed by representing the behavior of potential attackers, and modeling specific types of attackers. While the expressiveness of UML permits to define a large amount of details of the system analyzed, the set of security requirements that can be expressed with UMLSec are limited to technical aspects of the systems and therefore, they cannot be used for security requirements on social/organizational aspects.

Goal-based modeling languages allow to specify the rationale behind the actions and interactions between actors of the systems, i.e., they permit to specify what are the objectives of the actors and why they interact. The expressiveness of these languages, focused of on the objectives of actors, permits to easily specify social/organizational concepts of the systems analyzed.

Lamsweerde [122] proposes a method for elaborating security requirements by building two models: (i) a goal model of the system-to-be, which covers both the software and its environment and inter-relates their goals, agents, and requirements; (ii) an anti-model, derived from the goal model, that that describe goals of attackers. Both models are specified using an extension of Knowledge Acquisition in autOmedated Specification (KAOS) [33], a goal based modeling language. However, this approach is not focused on the security requirements but rather on the objectives of a possible malicious person therefore, it uses a limited set of security requirements that are formally specified but too limited for a real case scenario.

SI* [48] is a goal-based modeling language for the acquisition of security requirements for systems where actors socially depends on each other. It extends i^* [129], a goal based modeling language, adding the possibility of specifying different type social relations between actors, such as delegation of goals, transmission of authorizations and trust relations. With SI* is not possible to specify security requirements directly but only using ownership of goals, delegation of goals and permissions on delegations, therefore, it is not possible to specify well-known security concepts such as confidentiality or integrity.

Elahi et al. [40] propose an extension of i^* with with vulnerability, attacker and countermeasure concepts, for security tradeoff analysis. They based their approach on the idea that security requirement may not be fully satisfied but the can be partially satisfied and business process designers have to balance the satisfaction of such goal with other, contrasting goal, such as usability goal, costs, etc. The authors focused on creating a modeling language highly expressive, which can be used to specify many concepts. The resulting diagrams are very complex and difficult to read, therefore, the modeling language will be hardly usable in real scenarios. Moreover, security requirements are specified a set of goals with a textual description of few words for each goals, this may leads to misinterpretation and confusion.

Secure Tropos [79] is an extension of Tropos [22] method which enable to model security concerns throughout the whole development process. The modeling language used for Secure Tropos, which share the name with the method, extends i^* with security constraints over goal dependencies and with security capabilities of actors of the system. The language permits to specify security aspects in social/organizational perspective using a textual description; this choice prevent any limitation on the set of security requirements that can be detailed but, on the other hand, it may generate confusion and misunderstanding when the diagrams are read by different people.

STS-ml [31] is a goal-based modeling language for the specification of social/organizational security requirements in socio-technical systems. STS-ml uses three views to hide details of the model and, therefore, it permits to define richer models maintaining their readability higher than other similar approaches. It extends Secure Tropos with the possibility of defining security requirements as social commitments between actors and authorizations relations between actors.

2.3 Approaches for secure business processes

This section describes the most relevant approaches for modeling business processes with security concepts and for modeling security policies.

During last years a relevant number of modeling languages for secure business processes were proposed. But most of these languages are focused on only specific areas and so they fail to provide adequate coverage of the security domain [72].

As far as we known only few approaches were proposed in the last years for modeling security policies and none of them are graphical. The second part of this section analyzes query languages: graphical modeling languages for the specification of patterns (constraints) of business processes that are not focused on security.

2.3.1 Business process modeling languages with security concepts

A natural solution to represent the security concepts of business process is to create or extend a modeling language. Such languages are easy to learn and to use [78], thereby requiring a moderately low effort for security designers to specify a secure business process.

Menzel et al. [76] propose security extension of BPMN that enables generating security specifications for Service-oriented Architecture (SoA). The modeling language proposed by the authors specify security concepts as patterns defined in business process terms. However this approach limits the definition of security concepts in terms of business processes. Moreover, it is focused on a specific technical domain, i.e., SoA.

Rodriguez et al. [95] extend a subset of BPMN to express a predefined set of security requirement types. However, this approach permits to define a limited set of security concepts (non repudiation, attack harm detection, integrity, privacy and access control), and it does not take into account the flow of messages between business processes.

Saleem et al. [98] extend BPMN with security concepts for SoA applications. They include a set of security concepts in BPMN: confidentiality, integrity, availability, traceability, and auditing. The language permits to model a limited set of security concepts, moreover, their work is specific for the SoA domain.

Wolter and Schaad [128] propose an extension of BPMN for specifying task-based authorization constraints. Their approach includes a graphical extension of BPMN as well as a formalization of task-based authorization constraints. Their approach permits to specify dynamic resource allocation such as dynamic separation of duty and role-based resource allocation. Their approach is focused on authorization constraints of executors of tasks, and it is not possible to use it to specify other security concepts, such as availability or integrity.

SecureBPMN [24] extends BPMN with access control and information flow constraints. It uses the hierarchic structure of the organization, in which the business process will be executed, to help security designers to define security properties such as, for example, binding of duty [67] and separation of duty [67, 115]. However, SecureBPMN is limited in that it is not possible to specify other central security concepts such as, for instance, confidentiality or availability.

UMLSec [57], as described above, is a security-oriented extension of the UML [85]. In particular, for want concerns business processes, the extensions of UML activity diagrams, sequence diagrams and communication diagrams can be used to define business process with security choices. However, this part of UMLsec has a limited expressiveness in term of business process concepts that can be specified, moreover, the usage of three different modeling languages (diagrams) makes the specification of business process difficult since details are divided in three different diagrams.

Atluri and Warner [7] suggested a list of security concepts that shall be taken into consideration, and therefore specified, when modeling workflows. Their proposal is limited to the taxonomy, they did not propose a modeling language to specify such security concepts with business processes.

Other research works [6, 92] use extensions of Petri nets to define business processes with security concepts. Petri net modeling language is simple and easy to use but it does not include all the graphical constructs of BPMN. This influence negatively the understandability of models about medium-size or large business processes, limiting the applicability to only small-size business processes.

2.3.2 Query languages

Security policies can be seen as patterns, and the their verification against business processes corresponds to the problem of checking if a pattern holds in a business process. Query languages and their software tooling can be used to solve this type of problems, as they allow the creation of queries (patterns), and compliance verification against a business process model [100].

Dolman et al. [34] propose a pattern matching approach for conceptual models. Such approach consists in algorithms for solving the relaxed graph isomorphism problem, i.e., verifying if the nodes of a labeled graph match with a given pattern (isomorphism problem), and the existence of a path among the graph nodes as indicated in the pattern (homeomorphism problem). They created a tool that implements their algorithms to verify the compliance of a graph with a pattern. The approach is not specific to any modeling language, being rooted in labeled graph theory. However, such approach has to be extended to support the verification of security concepts in a business process.

Beeri et al. [15] propose Business Process Query Language (BP-QL), a pattern-based graphical query language for business processes. They also provide software tooling to determine the compliance of a business process—defined using WS-BPEL [83]—with a pattern. The decision of using Web Service - Business Process Execution Language (WS-BPEL), a machine-readable standard, hinders the readability of the business process, especially with real case scenarios, where business process easily reach

hundreds of elements.

A Process model Query Language (APQL), proposed by Hofstede et al. [51], is a textual query language, based on 20 predicates that can be composed to create complex queries. This approach suffers of scalability issues: the definition of complex queries is a challenging task that will lead to errors due to the complexity of the task. Moreover, as far as our knowledge goes, this approach is not supported by a software framework.

Visual Model Query Language (VMQL) [119] is a graphical query language based on UML activity diagrams [38]. It permits to define custom properties, which are evaluated when the compliance of a query is verified against a business process. But VMQL was not created for security purposes: even if the custom properties can be used to represent security concepts, the VMQL software engine can not interpret them limiting their usage only as a representation of security concepts.

Business Process Query Language (BPQL) [35] permits to graphically define both queries and business process models using the same language. Unfortunately, BPQL is not based on BPMN, hence the learning process is likely to be slower than that with by BPMN-Q. Moreover, BPQL does not include security concepts.

Ribeiro and Guedes [93] presented an analyzer for automatically verifying the consistency between workflow specifications and organizational security policies. But the language they provided for expressing security policies can be used for expressing separation of duty, information flow and access control constraints. This limits considerably the expressiveness of the language that cannot be used to specify well-known security concepts such as confidentiality, availability or integrity.

Wolter et al. [127] propose a modeling language for business processes and business security concepts, to be used to graphically define security specifications. They also develop a framework which transforms security goals in security policies specified in eXtensible Access Control Markup Language (XACML) [84] and Rampart [120]. The framework automatically extracts specifications of security mechanisms which enforce the se-

curity goals, but it does not permit security experts to compose security goals and, therefore, to create complex security policies.

Schmidt et al. [113] propose two ontologies for defining quality constraints and for defining service processes, respectively. Such ontologies are used to check if a service process complies with the imposed quality constraints. The main drawback of this approach consists in the fixed set of constraints that can be specified and checked.

Folder-Path Protocol and RDF Query Language (FPSPARQL) [16] is a query language which permits to define queries using a formal textual language. The proposal focuses on analyzing business process generated from action logs, hence is not possible to directly define a process, and moreover is not focused on security.

Sadiq et al. [97] propose to use a Formal Contract Language (FCL) to express normative specifications. Their approach includes a modeling language to visualize business processes as well as normative constraints. They also define a compliance distance, which denotes the extent to which the process model has to be changed to become compliant with the declared constraints. The limitation of this approach is the complexity of the language, despite the provision of a tool to graphical represent normative requirements and business processes. In future work, it would be interesting to compare the usability of the SecBPMN2 framework with the FCL-based approach.

Liu et al. [68] propose a language and a framework which statically verifies a business process against a formally expressed regulatory requirements. The framework accepts as input a business process specified in WS-BPEL [83] and a set of regulatory requirements, expressed with a temporal logic language called “Business Process Specification Language”. While powerful, this approach is hardly usable for large scenarios, due to the complexity of expressing regulatory requirements.

Ghose and Koliadis [47] enrich BPMN with annotations, and calculate how much a business process deviates from another business process. Differently from our approach, theirs focuses only on the structural difference

between processes with no consideration of security requirements.

2.4 Transformation approaches

This section describe the most known research approaches that permit to transform social/organizational approaches in business processes and business processes in part of the implementation.

The focus of this thesis is on business processes, but other research work focused on the generation of the implementation specification directly from a social/organizational perspective. Still, we believe that for nowadays' organizations, where tasks are executed by both technical component and people, business processes are central and the specification of the code is not enough to avoid security breaches.

For example, Alexei et al. in [62] propose a systematic method to generate executable business processes directly from a social/organizational model of the system, which is represented with a goal-based modeling language. Since their approach does not use any intermediate language between the social/organizational representation and the implementation, such as SecBPMN2 for our approach, the goal-base language they proposed is used to specify both social/organizational information and information on the execution of business processes. This syntax of this language is complex and we believe it will not scale well on real, huge, scenarios. Moreover, the approach does not take into account security concepts.

Mouratidis and Jurjens [80] propose a method to engineer the design of a system, using UMLsec, starting from social/organizational security requirements, using Secure Tropos. The guidelines defined for transforming Secure Tropos concepts in UMLsec concepts are not formally specified and the part on the transformation of security concepts consists in only one rule, therefore, a lot of freedom, and effort, is left to the user of the method. Moreover, the choice of UMLsec permitted the authors to include in the design also business processes, using sequence diagrams. But UMLsec sequence diagrams defined with the method are meant to be exe-

cuted by software and not by people. This limit the scope of the method only to technical system or, however, systems where the human part is not central and cannot compromise the security of the system.

Brandozzi and Perry [21] proposed a method to transform goal models in architecture. They based the transformation on a formal framework which minimizes the number of revisions of the goal model and the architectural model. They used KAOS [33] as goal-based modeling language and APL (Architecture Prescription Language) [21] as modeling language for the architecture. Both modeling languages does not take into account security perspectives, therefore, the proposed approach misses the transformation of security concepts.

2.4.1 From social/organizational to business process models

We surveyed the most cited papers on transformation of goal-based modeling language [52, 53], and even if the research field is active, i.e., many papers are published on the topic, only few of them are centered in the transformation of such modeling language in business processes with a focus on security aspects.

Lopez et al. [71] proposed a method to re-engineer business processes using SI*. Their proposal consists in creating an SI* diagram from an old, secure business process, create a new business process and verify if the new business process is compatible with what have been modeled in the SI* diagram. They used traces for the comparison: they create a trace from the two business model and a set of trace from the SI* model. If the two traces of the business process are compatible with the set of trace of the goal model, then the business process are goal-equivalent, i.e., the achieve the same goals. The approach is comparable to what is proposed in this thesis but the Lopez et al. used plain BPMN, i.e. without security concepts. This limit the approach since security concepts are limited to an a social/organizational perspective.

Knorr and Roehrig in [59] presented a framework for enforcing security

objectives in e-business processes, i.e., business processes which use information and communication technologies in support of all the activities of business. They focused on four main security objectives: confidentiality, availability integrity and accountability; but they claim their proposal can be easily extended to other security objectives. This proposal is focused to specific type of business processes, and offers solutions that are not easily generalizable to other scenarios, for example they integrate four fixed phases each business process analyzed shall execute: information, negotiation, payment, delivery. Even if this approach does not include directly social/organizational aspects we included it in this section because it offer a method to enforce high-level of abstraction security objectives (i.e. security requirements) in business processes.

2.4.2 From business process models to implementation

In the area of mapping or translating business process specifications to implementation code the number of existing proposals is surprisingly small.

Ouyang at al. [88] propose an approach to automatically transform BPMN 1.0 business processes in WS-BPEL [83] (Business Process Execution Language) code. The algorithm, and the software which implements the algorithm, proved that is possible to generate BPEL code from BPMN models and that, under certain conditions, the generated code is readable. This powerful approach was tested against four collections of real business processes that prove its reliability. But this approach used BPMN 1.0 witch has a very limited expressiveness, if compared to BPMN 2.0, moreover, it is focused on a subset of BPMN elements they called core elements. Therefore, many concepts, such as the ones that represents the executors of the task, are excluded from the transformation.

Backes at al. [11] propose a guideline for integrating security requirements in the development of business processes. The paper is focused on providing a formal trust model that shall be integrated in a formal specification of business processes to express arbitrary security goals. This will

provide concrete guidelines how a secure implementation can be achieved from secure specification. This approach does not use a model driven approach and, therefore, it will not be easily usable in real scenarios, where, frequently, there are tens of business processes with hundreds of activities. Moreover, the approach does not propose a formalization of the business process model to use.

Menzel et al. [76] propose security extension of BPMN that enables generating security specifications for SoA. Their proposed transformation rules generate machine-readable specification of such security properties in Rampart [120].

Wolter et al. [127], as described in a previous section, propose a modeling language for business processes and business security concepts. They also develop a framework which transforms security concepts in security policies specified in XACML [84] and Rampart [120]. The framework automatically extracts specifications of security mechanisms which enforce the security concepts, but it does not permit security experts to compose security concepts and, therefore, to create complex and custom security policies.

Other approaches focus on integrating security mechanism (e.g., access control infrastructures) into business process execution engines [24, 128].

Lohmann et al. [69, 70] discuss the integration of compliance aspects into artifact centric business processes. They integrate role-based access control, separation of duty and binding of duty properties in artifact-centric business processes. Similarly, they propose to use a petri-net based formalization to ensure the compliance to high-level separation of duty and binding of duty compliance requirements. This approach is related to other BPMN based process models, e.g., [29].

Kim et al. in [58] presented a framework for generating role-driven access security from business process models. Their work is based on information control net method, and permits to model business process, with an ad-hoc language provide with a formal definition of the semantic, and automatically derive a set of access control, that can be implemented with

any Role Based Access Control (RBAC) system. This proposal completely automate the generation of specifications, that can be implemented by the developer or directly in constraints for RBAC systems. But it is focused on a small part of security, which is the enforcement of authorizations, ignoring all other security concepts such as integrity or availability.

Damasceno et al. in [32] proposed SSC4Cloud, a framework which permits to model business processes in a shared environment, in order to allow the collaboration of different experts in the same model. The same framework provides an execution environment where the business processes created can be executed without waiting their implementations to be ready. The framework uses an ad-hoc language for the specification of business processes with security concepts that is used to generate the implementation code (WS-BPEL [83]) and a set of configurations, used for enforcing the security concepts. The implementation code generated can be executed only inside the execution environment provided with the framework, i.e., in a controlled environment where the security configuration are always applied correctly.

Other approaches focused on the verification of the implementation code. For example, Rushby [96] proposes a language and a framework which checks if the code of a software system diverges from specified behaviors. Software code can be considered as a specification of the behavior executed by a program, while the behaviors can be considered as the policies. This proposal suggests a new perspective on how to specify and check behaviors of software systems. But it can not be used on complex information systems because the transformation of business process to software is time consuming and it does not take in consideration the human perspective: one of the salient characteristic of business processes.

2.5 Chapter summary

This chapter describes the research work related to SEBE. While many approaches focused on part of the secure business engineering problem,

no approach offered an integrated method that supports business process designers in designing secure business processes, considering social/organizational information, and in generating the implementation code.

Many approaches are focused on narrow aspects of security, losing generalization and becoming too specified to be used as a general approach for security of business processes in organizations. Other approaches, instead, cannot be applied in real case scenarios because the complexity of the languages or operations to be executed prevents their execution/usage in complex, real, scenarios.

Currently, there is no method that permits to engineer secure business processes from a socio-technical perspective, that is applicable in real scenarios, with hundreds of processes executed by tens of actors that are involved in many (tens or even hundreds) of social relationships.

Chapter 3

Baseline

3.1 Representing social/organizational aspects with STS-ml

Social/organizational aspects of organizations define why components interact with each other and how they are related. Security requirements about such aspects, therefore, outline the security constraints they require to be met in each interaction. The rationale behind interaction is the first detail that should be determined when designing business processes, in order to understand the objective of actors and the results they expect from the interactions.

For this, we use the STS-ml [31, 89] modeling language. STS-ml is an actor and goal-oriented security requirements modeling language for socio-technical systems. It was chosen because: (1) it is specifically thought for socio-technical systems of organizations, relating security to interaction, (2) it supports a rich set of security requirements, while providing a clearer ontological foundation than existing approaches [48, 79].

3.1.1 Multi views approach

In STS-ml, requirements models are created through three views: (i) the *social view*—represents the main stakeholders (in terms of actors) together with their objectives (via goals) and the interactions they enter in the

socio-technical system; (ii) the *information view*—represents stakeholders’ informational assets and their representation via documents; and (iii) the *authorization view*—represents the authorizations that actors grant to others over their information. Figure 3.1 shows a partial STS-ml model of the motivating example.

Social view. Actors in STS-ml are modeled in terms of (i) agents—concrete entities that are already known at design-time (e.g., *Payment Engine*), and (ii) roles—abstract entities representing a class of participants (e.g., *Bank dst*). An actor’s rationale captures actors’ goals, and how they are achieved via and/or goal decompositions (e.g., the root goal of the *Payment Engine* is *Value transferred* that is and-decomposed in two subgoals *Transfer authorized* and *ID banks received*). Moreover, to achieve their goals, actors might need to *read* or *modify* documents, as well as create (*produce*) new documents (e.g., *Payment engine* reads document *Transfer order* to achieve goal *Value transferred*). Most importantly, the social view captures actors’ social interactions via two social relationships: *goal delegation* and *document transmission*. STS-ml allows actors to express their concerns about security (security needs) over the interactions they enter to then derive security requirements with respect to confidentiality, integrity, availability, accountability, reliability, and authenticity.

Information view. Information is a first class citizen in STS-ml, since most security issues are concerned with the protection of information. Information owners are the ones concerned with the protection of information. Therefore, STS-ml allows specifying information ownership via the relationship *own* that relates an actor to the information it owns. Information may be available in various forms, and thus, STS-ml distinguishes information from its representation in form of *documents*. Documents become relevant from a security point of view because of the information they represent. Thus, the purpose of the information view, apart from representing information entities and their respective owners, is to link together the documents actors use and exchange in the social view with their in-

formational content. This link is drawn through *Tangible by* relationships. In Figure 3.1, information *e-transfer details* is made tangible by document *Transfer order*.

Authorization view. An adequate representation of permissions and prohibitions is crucial to establishing whether information is used and exchanged in compliance with security requirements. The authorization view allows specifying the permissions and/or prohibitions on information that actors grant one to another. An authorization relationship details: (i) the permissions/prohibitions on the operations actors can perform over information (Read, Modify, Produce, Transmit) while manipulating documents for the achievement of their goals; (ii) information entities for which permissions/prohibitions are specified; (iii) the scope of authorization, referring to the goal(s) for the fulfillment of which permission/prohibition is specified; and finally, (iv) transferability, specifying whether permissions can be further granted to others (not applicable to prohibitions). In Figure 3.1, the *Payment Engine* authorizes the *Bank dst* to read *e-transfer details* in the scope of goal *Dst transfer authorized*, granting a transferable authorization. Security requirements are generated from authorizations whenever prohibitions are specified. For example, from the authorization to *Bank dst*, three security requirements are generated, one for each operation that is not authorized, namely: (i) non-modification, non-production and non-disclosure (i.e. not transmission) of information *e-transfer details*, as shown in Figure 3.1.

3.2 Querying business processes with BPMN-Q

While BPMN is adequate for expressing the interactions among the components in a complex socio-technical system, it does not natively support the verification of compliance with certain security properties that should hold in the model. For example, when modeling the landing procedure in ATM, one may want to verify that in the process it is always the case that

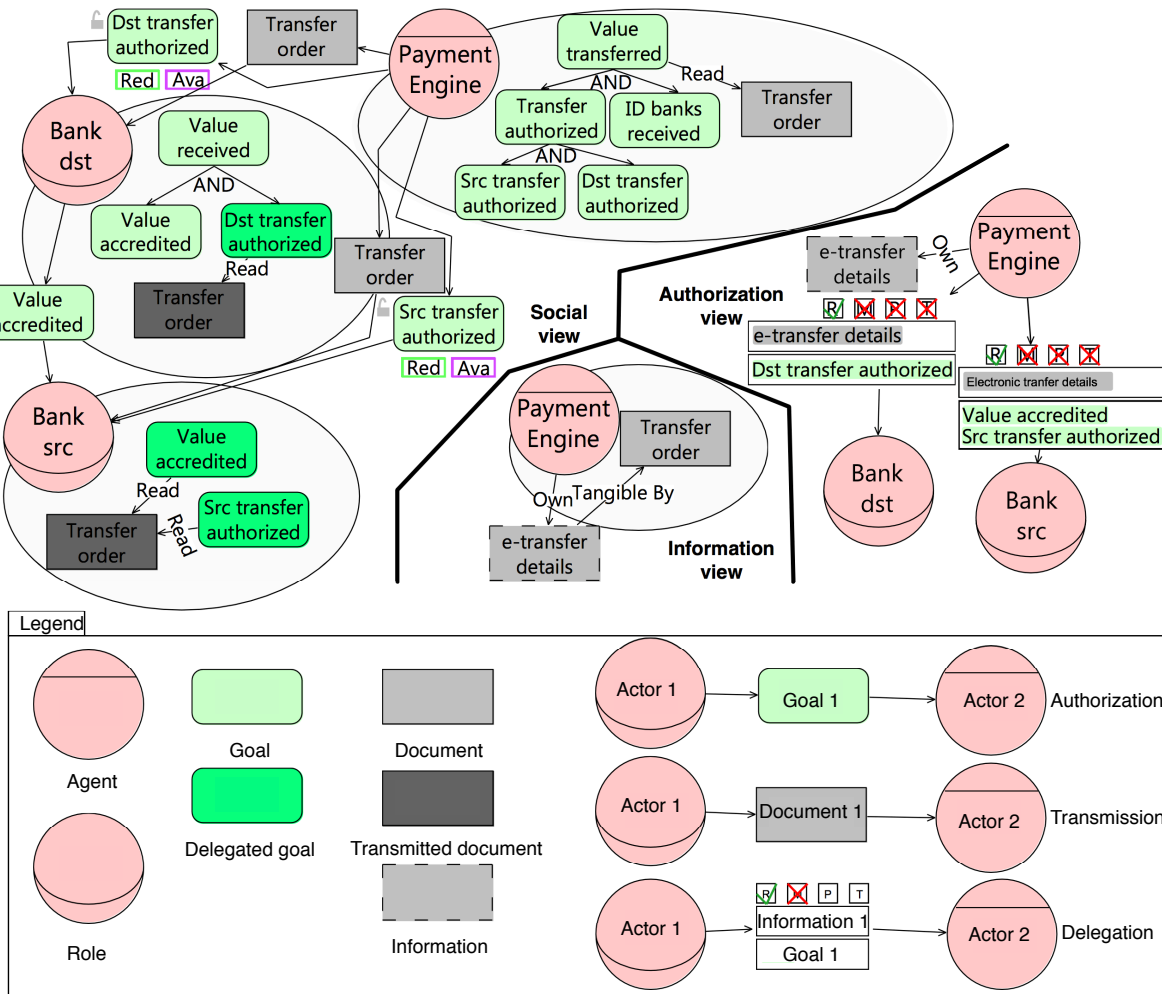


Figure 3.1: STS-ml diagram representing social/organizational model of PE.

pilots will confirm the landing trajectory of the plane.

Visual analysis of BPMN models works only for small scenarios, but it becomes ineffective when many models exist, or when they are as large as hundreds of elements. Moreover, when safety and security properties are at stake, relying on an informal analysis is not an option, due to the harmful effects of adopting a model that violates them.

BPMN-Q is a diagrammatic query language which partially overcomes this limitation, by expressing properties concerning business process models through graphical queries that can be checked against a BPMN model [10]. These queries can be seen as patterns that a given BPMN model should

comply with. BPMN-Q introduces relations that are functional to define the queries, i.e., the concepts of walk, negative walk and negative flow.

Figure 3.2 shows an example of a BPMN-Q query, on a security policy used in an airport [42]. The query enables checking whether the flight plan (Reference Business Trajectory or simply RBT) is approved and if the landing documents are checked at least once. The query will match against any business process model which: (i) contains an task labeled “Plane RBT generation service” and such task generates the data object “RBT [Proposed]” (the text within brackets denotes the state of the data object); (ii) contains a walk, i.e., a sequence of BPMN elements connected through a control flow, that connects the first task to a parallel gateway; (iii) contains a walk that connects the gateway to “Control Tower communication service” that generates the data object “RBT [Accepted]”; (iv) contains another walk that connects the gateway to “@Y” that reads the data object “Landing documents [Approved]”.

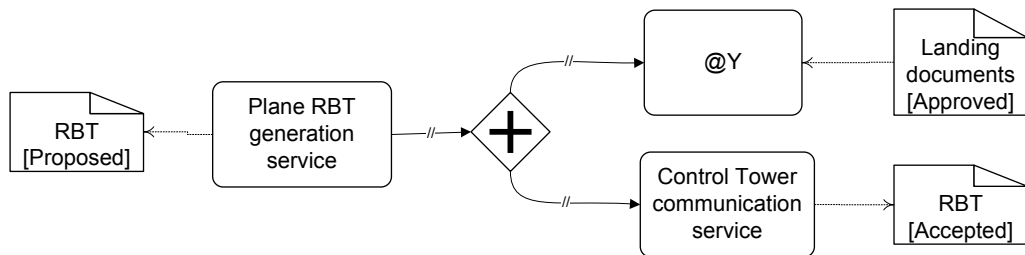


Figure 3.2: Example of BPMN-Q query

BPMN-Q enables expressing generic properties over BPMN elements, but does not provide any explicit modeling primitive for specifying security properties. We intend to overcome this limitation by defining security annotations that adhere with a state-of-the-art reference model for information security.

3.3 A reference model for security concepts

The Reference Model on Information Assurance and Security (RMIAS) [26], which was assembled through an analysis and classification of security concepts proposed by the most known reference models on information assurance and security. As far as our knowledge goes, RMIAS proposes the most comprehensive set of security concepts, for it aggregates and classifies security concepts proposed in the most known reference models on security and information assurance, such as the Confidentiality-Integrity-Availability (CIA) triad and BMIS [54]. The security concepts proposed in RMIAS are listed in Table 3.1.

Table 3.1: Security concepts covered by the RMIAS reference model [26]

Name	Definition
Accountability	An ability of a system to hold users responsible for their actions (e.g. misuse of information).
Auditability	An ability of a system to conduct persistent, non-by passable monitoring of all actions performed by humans or machines within the system.
Authenticity	An ability of a system to verify identity and establish trust in a third party and in information it provides.
Availability	A system should ensure that all system's components are available and operational when they are required by authorized users.
Confidentiality	A system should ensure that only authorized users access information.
Integrity	A system should ensure completeness, accuracy and absence of unauthorized modifications in all its components.
Non-Repudiation	The ability of a system to prove (with legal validity) occurrence/non-occurrence of an event or participation/non-participation of a party in an event.
Privacy	A system should obey privacy legislation and it should enable individuals to control, where feasible, their personal information (user-involvement).

3.4 Representing business artifacts with River Definition Language

Well known business-process or workflow modeling languages such as BPMN 2.0 or WS-BPEL are based on task flows: data that is processed within the processes is often an afterthought. In contrast, artifact-centric business process modeling [28,82] puts the business artifacts (e.g., data, documents) into the center of the process modeling.

We chose RDL [109] as the implementation language. RDL is an executable specification language that allows specifying declaratively: (i) the artifacts (e.g., entities); (ii) the relationships between them (e.g., associations); (iii) the business logic (e.g., actions) on the artifacts. RDL implementations generated with the transformation rules we provide, can be directly deployed.

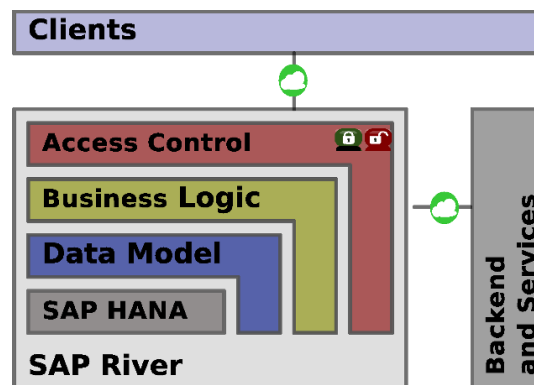


Figure 3.3: The SAP RDL Platform

Figure 3.3 shows an high-level overview of SAP RDL: SAP HANA provides the persistency layer as well as the container in which the enterprise applications are executed. Clients as well as back end systems or external services can communicate with the RDL platform using standard protocols. The business artifacts (i.e., the data model) and their behavior (i.e., the business logic) as well as the access control are specified in the RDL [109].

Listing 3.1 illustrates an example of a RDL application, which allows to specify entities (e.g., *customerAccount*), their attributes (e.g., *dateofBirth*),

```

1 type LocalDate { date : TimeStamp; state: String; }
2 application PaymentEngine{
3     role Consultant;
4     export entity customerAccount accessible by Consultant {
5         key element ID : String;
6         element dateOfBirth : LocalDate;
7     action updateInformation() { [...] } } }

```

Listing 3.1: An RDL application derived from the business process model in Figure 5.1

and custom types of the attributes (e.g. *LocalDate*). Besides this pure data modeling, i.e., definition of the data structure used to store information, RDL also supports the specification of the actions (e.g., *updateInformation*) in a declarative style. Finally, RDL supports the specification of role-based access control restrictions: the actions of the entity *customerAccount* are only accessible by members of the role *Consultant*.

By default, artifacts in RDL are private. To enable access outside of their scope, they need to explicitly marked with the export keyword. Moreover, the annotation @OData enables remote access using the OData protocol (www.odata.org). Such a remote access is controlled by the same access control restrictions of internal access.

3.5 Chapter summary

In this chapter we introduced the technologies and research approaches on which SEBE is based. STS-ml is used for the specification of social/organizational security aspects and security requirements. BPMN-Q is used to specify security policies, while RMIAS is used as reference for the security concepts that SEBE supports. RDL is our choice as implementation language for the implementation of business processes.

Chapter 4

A process for secure business process engineering

SEBE aims at helping business process designers to design secure business processes, using a social/organizational analysis of socio-technical systems of organizations, and to create part of the implementations of these socio-technical systems, which enforce security concepts specified in such secure business processes. The wide scope of the method calls for a flexible process that will guide the users of the method on its application,

Figure 4.1 shows an overview of the process provided with the SEBE method. The process receives as input a description of the social/organizational aspects of the socio-technical system of the organization analyzed, and it produces part of the implementation. The process consists in three main activities: (i) the definition of the social/organizational model and business process models (**Modeling** in the figure); (ii) transition from the social organizational model to business process model (**Transformation/analysis** in the figure); (iii) generation of the code (**Code generation** in the figure). The first and the second activities are repeated until the users consider the social/organizational models and the business process models complete.

Given the complexity and size of socio-technical systems, the first and the second activities, i.e., the definition and analysis of social/organizational and the business process models are repeated to create the models

iteratively and incrementally. The modeling activity is alternated with a transformation and analysis activity which helps in defining the models and gaining knowledge on the domain analyzed. When business process designers define business processes using information of the the social/organizational model, they gain more knowledge on the domain, therefore, if the new the social/organizational aspects are discovered, the related model will be updated.

This iteration can be used for the security aspects as well. If new requirements are found after the business processes are define, they are added in the social/organizational model and, then, a new iteration is needed. In other cases social/organizational security requirements may be to strict and, therefore, it is impossible to specify a business process which satisfy them and they have to be relaxed. Those interactions will lead to the creation of secure business processes that, once complete can be transformed in part of the implementation.

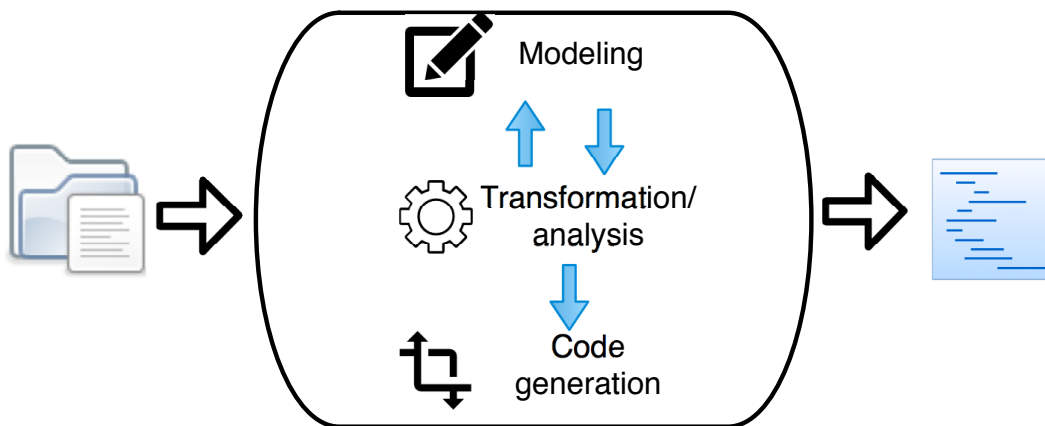


Figure 4.1: Overview of the SEBE process.

Roles. SEBE is meant to be used by business process designers, a role that encapsulates the expertise of a requirements analyst and a security engineer or expert. Such role requires a remarkable set of skills in modeling and security, therefore we suggest to execute the process by a team of experts. Ideally, such team should be supported by at least a domain expert, since

the creation of diagrams required by SEBE need deep knowledge of the socio-technical system analyzed.

Inputs/outputs. The method receives as input the requirements document (the result of elicitation activities), with a focus on security requirements. This document is the result of elicitation activities, including stakeholders' needs and information from regulatory rules, laws, and other related documents. Such document can be used as a starting point, whose information can be enriched by the domain experts involved in the process. The final output of the process is part of the implementation code of the technical systems (of the socio-technical system) that enforces security requirements.

The method can be adopted in the context of broader methods for system engineering and for software engineering. Broadly, the adoption of the method is appropriate for open and large-scale systems, for the social/organizational analysis is justifiable for socio-technical systems, and not for software systems (e.g., compilers) that do not include external autonomous actors.

4.1 Phases

Figure 4.2 shows the process provided with SEBE. In the following, we describe each phase, highlighting its importance.

4.1.1 Phase 1

In phase 1 STS-ml is used to model the social/organizational aspects of the socio-technical systems under consideration. Most importantly, the models capture security requirements of stakeholders. This phase results in: (i) the creation of a diagram representing the social/organizational model (STS-ml diagram, Figure 4.2); (ii) a list of social/organizational security requirements (STS-ml security requirements, Figure 4.2).

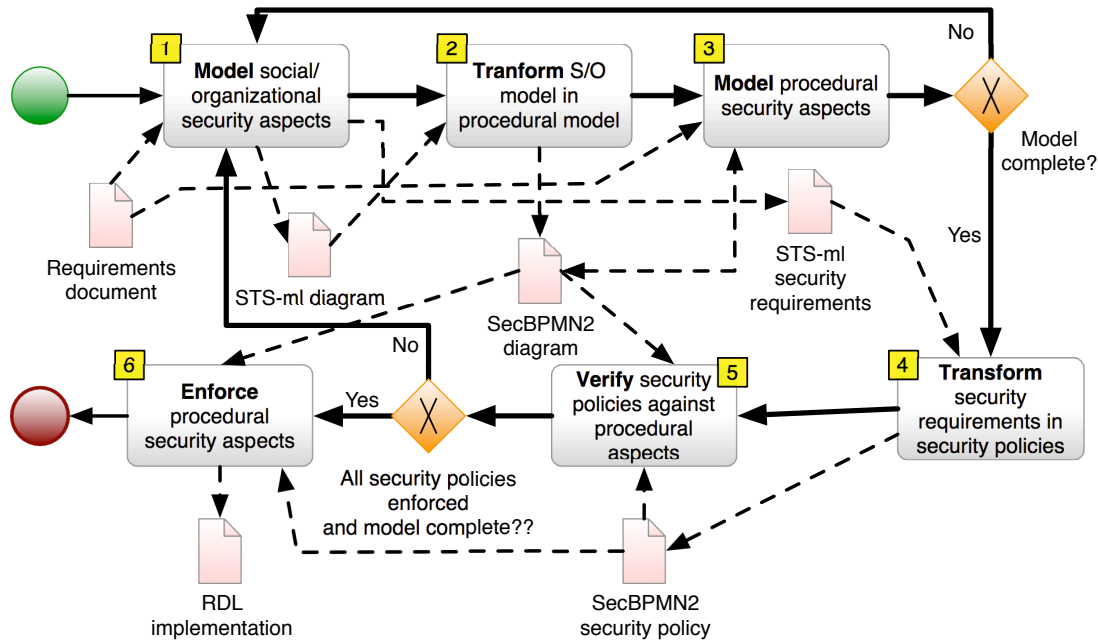


Figure 4.2: The process proposed with SEBE.

4.1.2 Phase 2

Phase 2 deals with the transformation of social/organizational models to business process ones, receiving as input a social/organizational diagram and generating a business process diagram specified using SecBPMN2, a modeling language for business processes with security concepts provided with SEBE (SecBPMN2 diagram, Figure 4.2).

4.1.3 Phase 3

Social/organizational models do not contain the information required to generate complete business process models. Therefore, phase 3 is executed to enrich the business process models generated by phase 2 with details such as the temporal aspects and security choices on the executed processes.

Phases 1-3 are repeated until business process designers decide they have captured all important (security) details and the social/organizational and business process models are complete and accurate.

4.1.4 Phase 4

Phase 4 specifies security policies. It generates security policies from social/organizational security requirements. The generation requires the list of social/organizational security requirements (STS-ml security requirements, Figure 4.2) and generates procedural security policies (SecBPMN2 security policy, Figure 4.2). It permits to translate security requirements, defined in terms of social/organizational concepts, in more operational constraints, as the ones specified in procedural security policies.

Optionally, users can specify procedural security policies using SecBPMN2 modeling language. This permits to define custom security policies that are not linked to any social/organizational security requirement.

4.1.5 Phase 5

Phase 5 deals with the verification of procedural security policies, generated in phase 4, against the business process model, enriched in phase 3. This phase validates the business process models by verifying, using automated reasoning tools such as DLV [39], if they enforce the security requirements specified in the social/organizational model. If all security policies are enforced, then the business process designers have the proof that the business process model meets the security requirements. If the business process model does not satisfy all procedural security policies, the business process designers will start the process from the beginning, refining the social/organizational or the business process models.

4.1.6 Phase 6

Phase 6 consists in generating part of the implementation from the enriched and verified business process model. The resulting application will enforce the social/organizational security requirements, because social/organizational security requirements define the security policies that are verified against the business process model. Therefore, because the transformation

enforces all the security concepts defined in the business process model, the implementation attends all security requirements as specified in social/organizational model.

4.2 Chapter summary

In this chapter describes the process provided with SEBE method. It is composed by 6 phases which forms an iterative and incremental process that supports business process designers in applying the method and use it efficaciously. The process meets SC6.1, which requires a process to be followed by the users of the SEBE.

Chapter 5

SecBPMN2

This chapter describes SecBPMN2 the modeling language for business processes with security concepts and procedural security policies. We designed it with the objective of creating a modeling language that can be used to specify business processes, with hundreds or thousands of elements, used in real scenarios. SecBPMN2 is expressive enough to be applied in most of the scenarios and with a formal definition of the abstract syntax that can be used for checking conflicts, incoherences and for the verification of security policies against secure business processes.

5.1 SecBPMN2: principles

SecBPMN2 sits on two main objectives: (i) the modeling language should be expressive enough to be used to define business processes with security concepts and security policies; (ii) however it should be as easy as possible to be understood even when business process experts deal with real, complex, scenarios.

Given those objectives we decided to extend BPMN 2.0 [87], a well known standard for modeling business process, with: (i) security concepts, in order to allow the specification of security concepts that are necessary for designing secure business processes; (ii) a query language, to specify security policies as constraints on the sequences of activities and on the security concepts.

The outcome of this research work is SecBPMN2, a modeling language composed in two parts: SecBPMN2-ml, a graphical modeling language for business processes with security concepts, and SecBPMN2-Q, a graphical query language for specifying security policies in terms of SecBPMN2-ml elements.

We extend BPMN 2.0 with security annotations to cover security concepts derived mainly from RMIAS reference model. Every annotation has a graphical syntax and has to be linked with an existing element of a BPMN 2.0 model: a task, a data object, a message flow or a pool.

We limited SecBPMN2-ml to the subset of BPMN 2.0 that serves for specifying processes and collaborations, i.e., the current version of the language does not support conversations and choreographies. This decision is a consequence of the objective of creating a language as simple as possible. Still, SecBPMN2-ml is not less expressive than BPMN 2.0 because conversations and choreographies are generalization collaborations, that are meant to be used for specify an overview of the system analyzed. Moreover, from our experience, conversations and choreographies models are far less used than process and collaboration models.

Other modeling languages have been proposed for specifying business processes and security concepts, such as those proposed by Menzel et al. [76], Rodriguez et al. [95] and Saleem et al. [98]. However, such modeling languages are either not designed to express security policies, e.g., the approach proposed by Menzel et. al [76], or they permit to specify only a subset of the security aspects that can be expressed with SecBPMN2, e.g., the approaches of Rodriguez et al. [95] and Saleem et al. [98].

5.2 BPMN 2.0

Figure 5.1 shows an example of a SecBPMN2-ml diagram. If we exclude the security annotations (the orange circles) the rest of the diagram is a legal BPMN 2.0 coordination model. We introduce briefly BPMN 2.0 using an example, since is a well known modeling language and on-line there are

plenty of resources such as tutorials and guides.

Figure 5.1 represents two business processes delimited by a start event and an end event. Each business process is executed by a participant, namely **Customer** and **PE**, and contains at least one task, e.g., **Generate Payment order**. BPMN 2.0 specifies 16 types of activities, but for the purposes of this thesis we introduce only two of them, the general task, which represent an atomic execution of a set of activities, for example **Convert values**, and **Call Activities**, which represent another business process that is called when the task is executed. For example, **Perform payment** is a call activity which refers to another business process that, its execution consist in the execution of another business process. When the linked business process ends, the call activity is considered concluded and the execution of the main process can continue.

Each participant may own one or more data objects that represent physical entities in which information is stored. For example, **PE** owns data object **Payment order**. Data objects are read/written by one or more activities. The arrow from **Payment order** data object to task **Perform payment** specifies that the task reads the data object when it is executed. A arrow in the opposite direction, i.e., from an task to a data object, specifies that the task writes the data object during its execution.

Communications between two participants are represented with message flows (thick dashed arrows), while the contents of the communications are represented by the **message** elements. For example, the execution of the task **Ask transfer** creates a communication channel from the **Customer** to the **PE** where the **Transfer Req.** message is sent.

The order of execution of activities is represented with the control flow relation that links two SecBPMN2 elements and specifies that the source element is executed before the target element. Call activities, e.g., **Store failure report**, are special type of activities that are linked to an external business process. The execution of such task consists in the execution of the linked business process.

Gateways represent branches in the control flow, for example, the di-

among shape with an “X” symbol is an exclusive gateway. The gateway executed after the task **Analyze transfer order** in Figure 5.1 specifies that the control flow can take two different paths, based on the evaluation of the condition **Transfer order correct?**. The exclusive gateway on the right merges the two branches of the control flow.

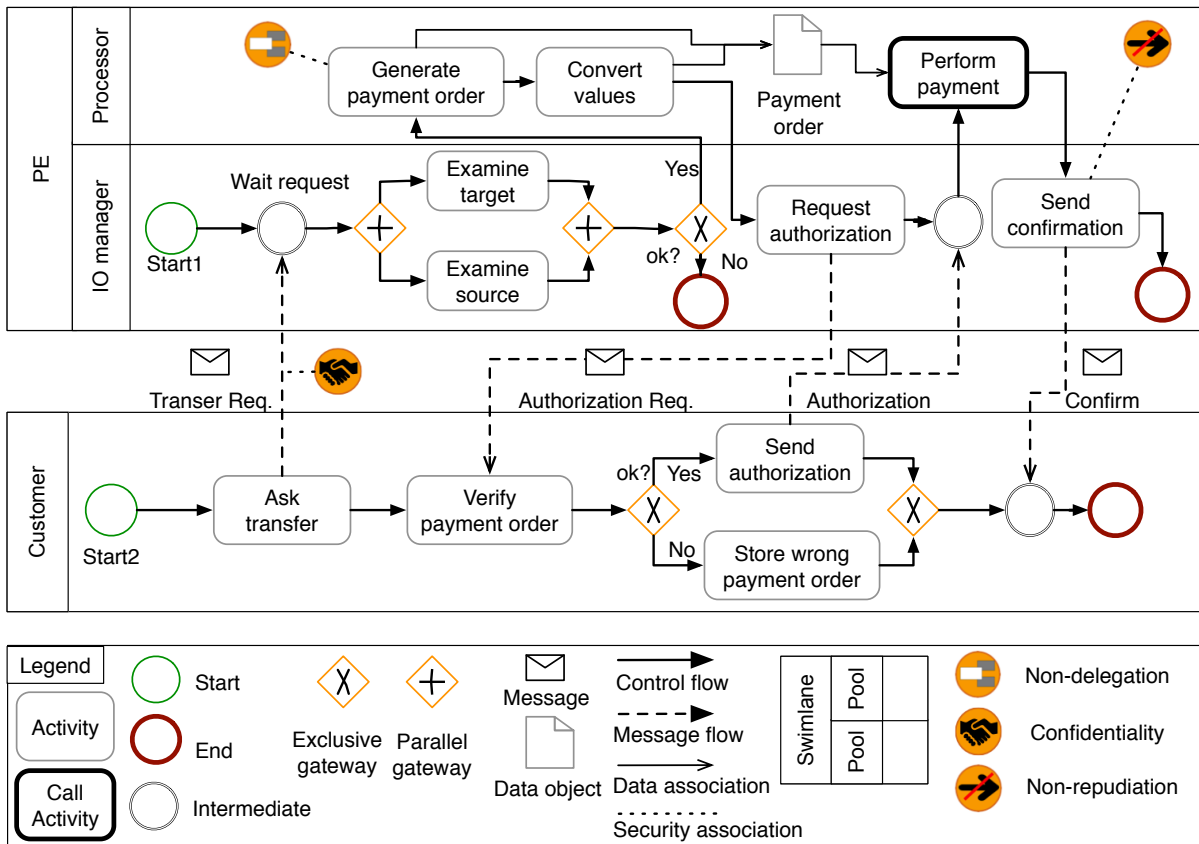


Figure 5.1: Example of SecBPMN2 model.

BPMN 2.0 standard does not provide any formal definition of its abstract syntax, actually, the standard is, in some parts, ambiguous. For example, it is not clearly defined the difference between swimlanes and pools, and there are contradictions in the instantiation of call activities [60]. We, therefore, provide a formal specification of the abstract syntax of BPMN 2.0 needed for the specification of SecBPMN2, and the specification of how the security policies are verified against secure business processes.

We define a BPMN 2.0 business process as tuple that contains the sequence of elements that can be executed, data objects, participants and association between them.

Definition 1 (BPMN 2.0 Business Process). *A BPMN 2.0 business process is a tuple $(\mathcal{A}, \mathcal{E}, \mathcal{G}, \mathcal{D}, \mathcal{P}, \text{controlFlow}, \text{dataAssociation}, \text{executor})$ where:*

- (a) \mathcal{A} is a finite set of activities,
- (b) $\mathcal{E} \subseteq \mathcal{E}^s \cup \mathcal{E}^e \cup \mathcal{E}^i$ is a finite set of events, $\mathcal{E}^s \cap \mathcal{E}^e \cap \mathcal{E}^i = \emptyset$,
- (c) \mathcal{G} is a finite set of gateways,
- (d) \mathcal{D} is a finite set of data objects,
- (e) $\mathcal{P} \subseteq \mathcal{P}^{pool} \cup \mathcal{P}^{swimlane}$ is a finite set of participants, $\mathcal{P}^{pool} \cap \mathcal{P}^{swimlane} = \emptyset$,
- (f) $\text{controlFlow} \subseteq (\mathcal{A} \cup \mathcal{G} \cup \mathcal{E} \setminus \mathcal{E}^s) \times (\mathcal{A} \cup \mathcal{G} \cup \mathcal{E} \setminus \mathcal{E}^e)$ is the control flow association,
- (g) $\text{dataAssociation} \subseteq \mathcal{D} \times \mathcal{A} \times \{\text{input}, \text{output}\}$ is the data association,
- (h) $\text{executor} \subseteq \mathcal{P} \times (\mathcal{A} \cup \mathcal{E} \cup \mathcal{G})$ is the executor association.

Definition 1 formalizes the definition of the abstract syntax of BPMN 2.0 business processes. The tuple is composed by: (a) a set of activities \mathcal{A} ; (b) a set of events \mathcal{E} that contains: a set of start events \mathcal{E}^s , that specifies where the business processes start; a set of end events \mathcal{E}^e , that indicates when the business processes finish; a set of intermediate events \mathcal{E}^i , that specifies when the business processes wait events to happen. The set \mathcal{G} , point (c), contains the gateways, i.e., elements used to create variants in the sequence of activities executed. The set \mathcal{D} , point (d), contains the set of data objects, i.e., the documents needed as input or outputted by activities, while the set \mathcal{P} , point (e), contains the participants, i.e., the actors who execute the activities. Participants are divided in pools \mathcal{P}^{pool} , i.e., independent actors, and swimlanes $\mathcal{P}^{swimlane}$, i.e., sub-division of pools. The controlFlow association, point (f), connects executable elements, i.e., \mathcal{A}, \mathcal{E} and \mathcal{G} ; from

now on $\mathcal{AEG} = \mathcal{A} \cup \mathcal{E} \cup \mathcal{G}$. The association `dataAssociation`, point (g) connects data objects with activities, and specifies if the data objects are used as input or are outputted by the activities. The association `executor` connects executable elements to the participant who executes them.

Following Definition 1, the business process executed by the participant PE in Figure 5.1 is defined as a tuple with $\mathcal{A} = \{\text{Examine target, Examine source, ...}\}$, $\mathcal{E}^s = \{\text{Start1}\}$, $\mathcal{E}^I = \{\text{Wait request, ...}\}$, $\mathcal{E}^e = \{\text{End1}\}$, $\mathcal{G} = \{\text{ok?}\}$. $\mathcal{D} = \{\text{Payment order}\}$ and $\mathcal{P}^{pool} = \{\text{PE}\}$ $\mathcal{P}^{swimlane} = \{\text{IO manger, Processor}\}$. In Figure 5.1 `controlFlow` connects, for example, `Generate payment order` with `Convert values`; `dataAssociation` connects `Payment order` with task `Convert values` and the value `output`. `Association executor` correlates, for example, `IO manager` with `Examine target` task, meaning that the `IO manager` executes the task `Examine target`.

Definition 2 specifies the conditions for a well-formed business process.

Definition 2 (Well-formed business process). *A BPMN 2.0 business process*

$(\mathcal{A}, \mathcal{E}, \mathcal{G}, \mathcal{D}, \mathcal{P}, \text{controlFlow}, \text{dataAssociation}, \text{executor})$ *is well-formed iff:*

- (a) $|\mathcal{E}^s| = 1$,
- (b) $|\mathcal{P}^{pool}| = 1$, *and*
- (c) $\forall x. x \in \mathcal{AEG} \rightarrow \exists! p \in \mathcal{P}^{pool}. (x, p) \in \text{executor}$.

Definition 2 specifies that a business process is well-formed when: (a) it has one start event; (b) it has one pool; (c) all executable elements are executed by only one pool. For example, the business processes in Figure 5.1 are well formed because they have 1 start event each, 1 pool each, and all activities are associated to a single pool.

Collaboration models are the core part of BPMN 2.0, they permit to model the communications between business processes. In such models communications are represented with message flows (thick dashed arrows), while the contents of the communications are represented by the message

elements. For example, the execution of **Ask transfer** creates a communication channel from the **Customer** to the **PE** where the **Transfer Req.** message is sent.

Definition 3 formally specifies a collaboration model: a set of business processes that exchange messages through message flows.

Definition 3 (BPMN 2.0 collaboration model). *A BPMN 2.0 collaboration model is a tuple $(\mathcal{BP}, \mathcal{M}, \mathcal{MF})$ where:*

- (a) $\mathcal{BP} = \{bp_1 \dots bp_n\}$ is a finite set of business processes in which $bp_i = (\mathcal{A}_i, \mathcal{E}_i, \mathcal{G}_i, \mathcal{D}_i, \mathcal{P}_i, \text{controlFlow}_i, \text{dataAssociation}_i, \text{executor}_i)$ is the i -th bp in \mathcal{BP} and $1 \leq i \leq n = |\mathcal{BP}|$,
- (b) \mathcal{M} is a finite set of messages,
- (c) $\mathcal{MF} \subseteq (\mathcal{A}_i \cup \mathcal{P}_i^{\text{pool}}) \times (\mathcal{A}_j \cup \mathcal{P}_j^{\text{pool}} \cup \mathcal{E}_j) \times (\mathcal{M})$ is the message flow relation, where $i \neq j$ and $1 \leq i \leq |\mathcal{BP}|$ and $1 \leq j \leq |\mathcal{BP}|$, and
- (d) $\mathcal{L} \subseteq \mathcal{AEG} \cup \mathcal{D} \cup \mathcal{P} \cup \mathcal{M} \times l$ is a labeling relation.

Definition 3 specifies a collaboration model as tuple composed of: (a) a set of business processes; (b) a set of messages; (c) a message flow relation that represents the communications between business processes. A message flow links a pool or a task of a business process to a pool, a task or an event of another business process, to a message that is exchanged. In the collaboration model a labeling relation, (d), associates a string to executable elements, data objects, messages and participants. In Figure 5.1 two business processes are connected with four message flows that transmit the messages **Transfer Req.**, **Authorization Req.**, **Authorization** and **Confirm**.

5.3 Security annotations

BPMN 2.0 is an expressive, well known standard for modeling business processes, but it is not focused on security and, therefore, it lacks on the security concepts that are needed for specifying secure business processes.

Given that, we extend BPMN 2.0 with security annotations: graphical annotations that specify constraints on the execution of the business process and their activities.

Before choosing the security concepts to introduce in BPMN 2.0 we carefully examined the available reference models on security, i.e., security-specific ontology that clearly defined security concepts that are produced by experts.

A prominent family of reference models relies on the Confidentiality Integrity Availability (CIA) triad [90]. However, their adequacy has been questioned for they characterize a too limited set of properties of a system [91]. Richer models exist, such as McCumber's cube [74], which conceives system security from three perspectives: information states, critical information characteristics, and security measures. The Business Model for Information Security (BMIS) [54] addresses four interconnected elements: organization design and strategy, people, process, and technology.

We choose to extend BPMN 2.0 with the concepts proposed in the Reference Model on Information Assurance and Security (RMIAS) [26]. As far as our knowledge goes, RMIAS proposes the most comprehensive set of security concepts, for it aggregates and classifies security concepts proposed in the most known reference models on security and information assurance, such as the CIA triad and BMIS [54]. Still, during the iteration with security experts, we realized that RMIAS has some limitations: some security concepts, required by the experts to model common scenarios, were missing. We, therefore, extended the reference model with three security concepts. The final list of security concepts are listed in Table 3.1, the first eight are defined in RMIAS the last three are our extension of the reference model.

We designed the graphical syntax of the primitives following Moody's guidelines for increasing the usability and comprehensibility of modeling languages [78]. Moody classifies the visual differences between graphical elements of modeling languages in 8 visual variables: horizontal and vertical position, shape, size, color, brightness, orientation and texture. Graphical

elements that represent similar concepts should share as many variables as possible, while they should be easily distinguishable among themselves, i.e., they have at least one visual variable not in common. For example, the BPMN 2.0 start and end graphical elements in Figure 5.1 have a visual distance of 3. They share the same shape, i.e., a circle, the same size, brightness, orientation and texture, but they have different colors, i.e., start events are green while end events are red, and different vertical and horizontal positions

Security annotations share three common visual variables: they all have an orange fill color, a solid texture, and a circular shape; they differ for the icon in the middle of the circle. Every security annotation has a visual distance of 3 from non-security annotations, i.e., they can be easily recognized from other elements of the modeling language, and a visual distance of 1 (the texture, i.e., the icon) from other security annotations.

We decided to use icons instead of abstract symbols because icons are deemed as easier to remember and faster to recognize [78]. For example, the icon for the availability security annotation is a clock face, which should recall the concept of “time” and, therefore, should be easily linked to the definition of availability security concepts (see Table 3.1). Leitner et al. [63–65] conducted empirical studies to propose guidelines for representing a set of security concepts. We did not apply these suggestions because they conflict with the recommendation by the security experts that helped us define the security annotations and, moreover, the set of security concepts Leitner et al. took into account covers only partially the security concepts proposed in RMIAS.

Security annotations can be linked to activities, data objects, message flows and pools, but cannot be linked to events, which represent changes in the environment of a socio-technical system. Moreover, security annotations have attributes that are used to specify detailed information on the security mechanisms¹ that enforce the policy. Attributes are optional

¹The low level (software and hardware) functions that implement the controls imposed by the policy [107]

except for the one linking the annotation with a BPMN 2.0 element.

In the rest of the section all security annotation are defined. Each of them is formalized in terms of one or more predicates, one for every type of BPMN 2.0 element that the annotation can be linked with.












AccountabilityAct (a: task, enfBy: {SecMechanisms}, monitored: {Users})	
AuditabilityAct (a: task, enfBy: {SecMechanisms}, frequency: Time) AuditabilityDO (do: DataObject, enfBy: {SecMechanisms}, frequency: Time) AuditabilityMF (mf: MessageFlow, enfBy: {SecMechanisms}, frequency: Time)	
AuthenticityAct (a: task, enfBy: {SecMechanisms}, ident: Bool, auth: Bool, trustValue: Float) AuthenticityDO (do: DataObject, enfBy: {SecMechanisms})	
AvailabilityAct (a: task, enfBy: {SecMechanisms}, level: Float) AvailabilityDO (do: DataObject, enfBy: {SecMechanisms}, authUsers: {Users}, level: Float) AvailabilityMF (mf: MessageFlow, enfBy: {SecMechanisms}, level: Float)	
ConfidentialityDO (do: DataObject, enfBy: {SecMechanisms}, readers: {Users}, writers: {Users}) ConfidentialityMF (mf: MessageFlow, enfBy: {SecMechanisms}, readers: {Users}, writers: {Users})	
IntegrityAct (a: task, enfBy: {SecMechanisms}, personnel: Bool, hardware: Bool, software: Bool) IntegrityDO (do: DataObject, enfBy: {SecMechanisms}) IntegrityMF (mf: MessageFlow, enfBy: {SecMechanisms})	
NonRepudAct (a: task, enfBy: {SecMechanisms}, execution: Bool) NonRepudMF (mf: MessageFlow, enfBy: {SecMechanisms}, execution: Bool)	
PrivacyAct (a: task, enfBy: {SecMechanisms}, sensitiveInfo: {Info}) PrivacyDO (do: DataObject, enfBy: {SecMechanisms}, sensitiveInfo: {Info})	
SoDPool (p1: Pool, p2: Pool, enfBy: {SecMechanisms}, dynamic: Bool)	
BoDPool (p1: Pool, p2: Pool, enfBy: {SecMechanisms}, dynamic: Bool)	
NonDelegationAct (a: task, enfBy: {SecMechanisms})	

Table 5.1: Security annotations in SecBPMN2: predicates and their graphical syntax

Accountability. It applies only to activities—thus, only one corresponding predicate called `AccountabilityAct` exists—, and expresses the need of monitoring a set of users when executing the task. The predicate has three parameters: the task `a` that is being monitored, a set of security mechanisms `enfBy` used to enforce accountability for the task, and the set of users `monitored` which are monitored.

If the task is executed by a user that is not in `monitored`, the security property is satisfied without using the enforcement mechanism. This situation would typically occur with trusted users that do not need to be monitored. Security designers can specify the keyword `ALL` in `monitored`, to indicate that all users are held for their actions.

Consider, for example, the predicate `AccountabilityAct`(“Generate payment order”, `{RBAC}`, `{Processor}`). The first attribute details the task linked with the security annotation, the second one indicates that RBAC (Role-Based Access Control) [43] will be used to enforce accountability, while the third one specifies that only `Processor` have to be monitored while executing that task.

Auditability. This security annotation comes in three variants, expressing different types of auditability in a business process: (i) `AuditabilityAct` indicates that it should be possible to keep track of all the actions performed by the executor of the task `a` when trying to execute that task; (ii) `AuditabilityDO` indicates that it should be possible to keep track of all the actions (e.g., write, read, store) concerning a data object `do`; (iii) `AuditabilityMF` indicates that it should be possible to keep track of all the actions executed to handle the communication (send/receive actions) within a message flow `mf`.

The predicates share two parameters: `enfBy` to express the set of security mechanisms to be used, and `frequency` to specify how often the security checks are performed. If `frequency` is set to zero, continuous verification is required.

For instance, consider the predicate `AuditabilityAct`(“Perform payment”,

{}, 10d). It applies to task **Perform payment**, it does not require a specific technology for checking auditability, and requires audits to be performed every 10 days.

Authenticity. It comes in two versions, depending on which BPMN elements the annotation applies to. **AuthenticityAct** imposes that the identity and/or authenticity of the users of task **a** are verified. The attribute **enfBy** is the set of security mechanisms to be used, while **trustValue** is the minimum level of trust [56] the executor of task **a** must have. If attribute **ident** is true, anonymous users should not take part in the execution of the task, while if **auth** is set to true, the identity of users should be verified. **AuthenticityDo** indicates that it should be possible to prove the data object **do** is genuine: the fact that **do** was not modified by unauthorized parties, and it contained proofs of the identity of the entities who generated and/or modified it.

For example, consider the predicate **AuthenticityDO** (“Payment order”, {TLS, X.509}). The predicate specifies that the authenticity of **Payment order** data object should be guaranteed using Transport Security Layer (TLS) and X.509 security mechanisms.

Availability. It applies to three BPMN elements, hence we defined three different versions: (i) **AvailabilityAct** specifies that the task **a** should be ready for execution whenever the task is encountered in the control flow of the business process; (ii) **AvailabilityDO** specifies that the data object **do** should be available when required by the authorized users specified in the attribute **authUsers**; (iii) **AvailabilityMF** specifies that it should always be possible to communicate through the message flow **mf**.

The predicates share two parameters: **enfBy**, described above, and **level**, i.e., the minimum time percentage that the resource (i.e., task, data object or message flow, depending on the variant of availability annotation) should be available. In **AvailabilityDO**, security designers can specify that all users are authorized to request the data object, simply specifying the keyword **ALL** in the attribute **authUsers**.

For instance, the predicate `AvailabilityAct("Examine source", {SAVE}, 99.5)` specifies that `Examine source` has to process at least 99.5% of the total requests, using the Source Address Validity Enforcement (SAVE) [66] protocol to prevent denial of service attacks.

Confidentiality. It has two variants: `ConfidentialityDO` specifies that the data object `do` can be accessed only by authorized users, and `ConfidentialityMF` specifies that only authorized users can use (i.e., send or receive through) the message flow `mf`. Both predicates share three parameters: `enfBy`, already described; `readers`, i.e., the set of users that are authorized to read the data object (or receive from the message flow); `writers`, i.e., the set of users that are authorized to write the data object `do` (or send through the message flow). The attributes `readers` and `writers` allow the usage of the keyword `ALL` to specify that all the users are authorized.

For example, consider the predicate `ConfidentialityMF (mf("Ask transfer", "Wait request"), {TLS, RBAC}, {IO manager, Processor, PE}, {Customer, Bank})`, which details one of the confidentiality annotations in Figure 5.1. It specifies that only the users `IO manager`, `Processor` and `PE` can receive from the message flow between `Ask transfer` and `Wait request`, and only `Customer` and `Bank` can send messages through that channel. This security annotation must be enforced using both TLS and RBAC security mechanisms.

Integrity. It comes in three variants: (i) `IntegrityAct` specifies that the functionalities of task `a` should be protected from intentional corruption. Attributes `personnel`, `hardware` and `software` determine which entities—involved in the execution of the `a`—are protected from intentional corruption [45]; (ii) `IntegrityDO` specifies that the data object `do` should be protected from intentional corruption; (iii) `IntegrityMF` specifies that every message exchanged through `mf` should be protected from intentional corruption. All the predicates share the attribute `enfBy`.

For instance, the predicate `IntegrityAct("Convert values", {}, false, true, true)` indicates that software and hardware used to execute `Convert values`

will be protected from intentional corruption, e.g., unauthorized modifications of the software or hardware robbery.

Non-Repudiation. It comes in two variants, depending on the element it applies to: `NonRepudiationAct` and `NonRepudiationMF`. The former indicates that the execution (or non-execution) of task `a` should be provable, while the latter specifies that the usage (or non-usage) of the message flow `mf` should be verifiable. Both predicates share two attributes: `enfBy`, already described before, and `execution`. The latter specifies that: (i) a proof of execution of task `a` or a proof of usage of the communication channel `mf` shall be provided, if set to true; (ii) a proof of non-execution for `a` or a proof of non-usage for `mf` shall be provided, if set to false.

For example, the predicate `NonRepudiationAct("Send confirmation", {}, false)` defines one of the non-repudiation annotations in Figure 5.1. It specifies that it should be possible to prove that `Send confirmation` has never been executed. There are no constraints on the security mechanisms that have to be implemented because the parameter is an empty set.

Privacy. It has two variants: (i) `privacyACT` specifies that task `a` should be compliant with privacy legislation, and it should let users to control their own data; (ii) `privacyDO` is similar to the former one, but is targeted to a specific data object `do`. Both predicates share two parameters: `enfBy`, already described, and `sensitiveInfo`, i.e., the set of sensitive information to protect.

For example, consider the predicate `PrivacyDO("Payment order", {}, {name, surname, CreditCardNumber})` specifies that, if the content of `Payment order` is published, name, surname and credit card number information shall be anonymized as required by law, e.g., only partial information can be published.

Separation of duties is a security principle used to formulate multi-person control policies, requiring that two or more different people be responsible for the completion of a task or set of related activities [115]. If the set of people changes during the execution of the system, separation of

duties is dynamic, otherwise, separation of duties is static.

It can be linked to two pools and it is defined by one predicate `SoDPool`, which specifies that `p1` and `p2` cannot be the same person at the same type. The parameter `enfBy` specifies the security mechanisms, while if `dynamic` is true, it specifies that `p1` and `p2` can be played by different people, if it is false the pool identifies one person for the entire execution of the business process.

For example, the predicate `SoD("PE", "Customer", {RBAC}, false)` defines a separation of duties security annotation linked to `PE` and `Customer` and specifies that a person cannot execute simultaneously the activities of `Customer` and `PE`. In this case separation of duties is not `dynamic` because, `PE` does not change and `Customer` is the same for all the execution of the business process.

Binding of duties requires the same person to be responsible for the completion of a set of related activities [123]. Binding of duties can be either static or dynamic. Such security concept is represented by the `Bind of duties` security annotation and it shares the same security property of separation of duties security annotation.

Similarly to Separation of duties annotation, it can be linked to two pools and it is defined by one predicate `BoDPool`, which specifies that `p1` and `p2` must be the same person. The two parameters `enfBy` and `dynamic` share the same semantic of the ones specified for separation of duties.

Non-delegation requires that a set of actions shall be executed only by the users assigned.

It is specified using the `NonDelegationAct` predicate which specifies that it is not possible to delegate part or the whole task `a` to any other participant. The parameter `enfBy` specifies the list of security mechanisms used for enforcing the security annotation.

For example, `NonDelegation("Generate payment order", {})` specifies the Non-delegation annotation in Figure 5.1: it is linked to `Generate payment order`, the task will be executed only by the `Processor`, and no one else.

5.4 SecBPMN2-ml

The integration of the security annotations, that we specified before with BPMN 2.0 creates SecBPMN2-ml [102, 103]. Figure 5.1 shows an example of a SecBPMN2-ml diagram.

Please note that the modeling language does not permit to graphically specify the parameters of the security annotations. This is because we wanted to maximize the readability of the language in case of huge diagrams for real scenarios. However, STS-Tool, the software tool which permits to create SecBPMN2 diagrams, allows to specify those properties.

In the following we formally specify the abstract syntax of SecBPMN2-ml. Definition 4 specifies a SecBPMN2-ml collaboration model as an extension of a BPMN 2.0 collaboration model with security annotations.

Definition 4 (SecBPMN2-ml collaboration model). *A SecBPMN2-ml collaboration model is a tuple $(CM, \mathcal{SA}, \text{SecAss}, \text{SAType})$ where:*

(a) *CM is a collaboration model,*

(b) *\mathcal{SA} is a finite set of security annotations,*

(c) $\text{SecAss} \subseteq (\mathcal{SA}) \times \left(\bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i \cup \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{D}_i \cup \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{P}_i^{\text{pool}} \cup \mathcal{M} \cup \text{MF} \right)$, and

(d) $\text{SAType} \subseteq \mathcal{SA} \times \{ \text{Accountability, Auditability, Authenticity, Availability, Confidentiality, Integrity, NonRepudiation, Privacy, BindOfDuties, SeparationOfDuties, NonDelegation} \}$ is a relation that associates a type to each security annotation.

Definition 4 defines a SecBPMN2-ml collaboration model as a tuple containing: (a) a BPMN 2.0 collaboration model; (b) a set of security annotations; (c) a security association relation that connects each security association with an element among activities, data objects, pools, message flows and messages; (d) an association which is used to determine the type of the security association. For example, in Figure 5.1, \mathcal{SA} is composed

of three security annotations and **SecAss** contains the link between the security associations and the elements of the collaboration, i.e., the links between the non-delegation association and **Generate payment order** task, between the confidentiality security annotation and the message flows that transmits **Transfer Req.** message, and between the non-repudiation security annotation and the **Send confirmation** task.

We consider a SecBPMN2-ml collaboration model well-formed when all BPMN 2.0 business processes in the model are well-formed, see Definition 2, and all security annotations are correctly associated to BPMN 2.0 elements.

Definition 5 (Well-formed SecBPMN2-ml collaboration model). *A SecBPMN2-ml collaboration model $(CM, \mathcal{SA}, \text{SecAss})$ is well-formed iff:*

- (a) $\forall \text{bp}. \text{bp} \in \mathcal{BP} \rightarrow \text{bp}$ is well formed,
- (b) $\forall (sa, t_1). ((sa, t_1) \in \text{SecAss} \wedge \text{SAType}(sa) \neq \text{BindOfDuties} \wedge \text{SAType}(sa) \neq \text{SeparationOfDuties}) \rightarrow / \exists (sa, t_2). (sa, t_2) \in \text{SecAss}. t_1 = t_2$, and
- (c) $\forall (sa, t). (sa, t) \in \text{SecAss} \rightarrow$

$$(i) \text{ if } \text{SAType}(sa) = \text{Accountability} \text{ then } t \in \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i,$$

$$(ii) \text{ if } \text{SAType}(sa) = \text{Auditability} \text{ then } t \in \left(\bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i \cup \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{D}_i \cup \mathcal{MF} \right),$$

$$(iii) \text{ if } \text{SAType}(sa) = \text{Authenticity} \text{ then } t \in \left(\bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i \cup \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{D}_i \cup \mathcal{M} \right),$$

$$(iv) \text{ if } \text{SAType}(sa) = \text{Availability} \text{ then } t \in \left(\bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i \cup \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{D}_i \cup \mathcal{MF} \right),$$

$$(v) \text{ if } \text{SAType}(sa) = \text{Confidentiality} \text{ then } t \in \left(\bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{D}_i \cup \mathcal{MF} \right),$$

$$(vi) \text{ if } \text{SAType}(sa) = \text{Integrity} \text{ then } t \in \left(\bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i \cup \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{D}_i \cup \mathcal{MF} \right),$$

(vii) if $\text{SAType}(sa) = \text{NonRepudiation}$ then $t \in (\bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i \cup \mathcal{MF})$,

(viii) if $\text{SAType}(sa) = \text{Privacy}$ then $t \in (\bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i \cup \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{D}_i)$,

(ix) if $\text{SAType}(sa) = \text{BindOfDuties}$ then $t \in \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{P}_i^{\text{pool}}$,

(x) if $\text{SAType}(sa) = \text{SeparationOfDuties}$ then $t \in \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{P}_i^{\text{pool}}$, and

(xi) if $\text{SAType}(sa) = \text{NonDelegation}$ then $t \in \bigcup_{i=1}^{|\mathcal{BP}|} \mathcal{A}_i$.

Definition 5 specifies that a SecBPMN2-ml collaboration model is well-formed if: (a) all the business processes in the SecBPMN2 collaboration model are well formed; (b) each security annotation, that is not bind of duties or separation of duties, is linked to one and only one element of the business processes; (c) if the security annotations are associated to a subset of BPMN 2.0 elements. Specifically, point (c) specifies that:

- (i) accountability annotation can be linked only to activities;
- (ii) auditability annotation can be linked only to activities and data objects;
- (iii) authenticity annotation can be linked only to activities and data objects;
- (iv) availability annotation can be linked only to activities, data objects and message flows;
- (v) confidentiality annotation can be linked only to data objects and message flows;
- (vi) integrity annotation can be linked only to activities, data objects and message flows;

- (vii) non-repudiation annotation can be linked only to activities and message flows;
- (viii) privacy annotation can be linked only to activities and data objects;
- (ix) bind of duties annotation can be linked only to pools;
- (x) separation of duty annotation can be linked only to pools;
- (xi) non-delegation annotation can be linked only to activities.

5.5 SecBPMN2-Q

SecBPMN2-Q is a modeling language expressly thought for the specification of security policies. It extends SecBPMN2-ml with BPMN-Q [9], a query language for business processes, based on BPMN. Specifically, BPMN-Q allows to graphically specify queries: sets of constraints on sequence of activities that can be checked against BPMN business processes. We extended BPMN-Q for BPMN 2.0 models and then we integrated it in SecBPMN2-ml. We called the SecBPMN2-Q queries can be used to specify security constraints on business processes, we, therefore, call them security policies.

SecBPMN2-Q extends SecBPMN2-ml with three relations: **negative control flow**, **walk** and **negative walk**. All of them connect two SecBPMN2-ml elements among activities, events or gateways.

Negative control flow relation matches all business processes in which the target element is never executed right after the source element.

Walk relation matches all business process in which the target element is executed after the source element.

Negative walk relation is the negation of walk relation, i.e., it matches all the business processes in which the target element is never executed after the source element.

SecBPMN2-Q, similarly to BPMN-Q, assigns a specific semantic to the label “@” of BPMN 2.0 elements: when an element in a security policy has

“@” in its label, it will match all elements of the same type irrespective of the label.

Definition 6 specifies a SecBPMN2-Q security policy as SecBPMN2-ml collaboration model with negative control flow, walk and negative walk relations.

Definition 6 (SecBPMN2-Q security policy). *A SecBPMN2-Q security policy is a tuple $(SecCM, NCF, Walk, NWalk)$ where:*

- (a) *SecCM is a SecBPMN2-ml collaboration model,*
- (b) *$NCF \subseteq (\mathcal{AEG}_i) \times (\mathcal{AEG}_i)$ with $1 < i < |\mathcal{BP}|$,*
- (c) *$Walk \subseteq (\mathcal{AEG}_i) \times (\mathcal{AEG}_i)$ with $1 < i < |\mathcal{BP}|$, and*
- (d) *$NWalk \subseteq (\mathcal{AEG}_i) \times (\mathcal{AEG}_i)$ with $1 < i < |\mathcal{BP}|$.*

Definition 6 specifies a SecBPMN2-Q security policy as a tuple which contains: (a) a SecBPMN2 collaboration model; (b) a Negative Control Flow (NCF) relation; (c) a walk relation; (d) a negative walk (NWalk) relation. Negative control flow, walk and negative walk relations connect two elements of the same business process, this is indicated with the same variable in the subscript of the elements of the ordered couples. A security policy may contain all SecBPMN2-ml collaboration model elements and 0 or more walk, negative walk and negative control flow relations.

Figure 5.2 shows two examples of a SecBPMN2-Q procedural security policy. Security policy in Figure 5.2(a) has a green continued border that specifies the policies is a pattern, i.e., it must be verified in all business processes of the system. The arrow with two heads between **Wait request** and **Request authorization** is a **Walk** which matches all the SecBPMN2-ml models where the first and the second activities are connected through an arbitrary long list of elements. The “@” wildcard, followed by any string, is used to match any SecBPMN2-ml elements of the same type. For example, the business process model of Figure 5.1 matches the procedural security policy of Figure 5.2, because the task **Ask transfer** (matches by the “@X”

task in the security policy) sends the message **Transfer Req.** to the event **Wait request**, which is connected with **Request authorization** through three gateways and three activities.

Security policy in Figure 5.2(b) has a red dashed border that specifies the policy is an anti-pattern, i.e., it must *not* be verified in all business processes of the system. It is composed of two activities connected with a **Walk** relation which matches all the business processes where the source task is connected to the second task through a walk. Since no pools are specified, no constraints are put on the participants who execute the process. The business process model shown in Figure 5.1 does not satisfy the security policies. Since there is a walk between **Convert values** and **Send confirmation** that has a non-repudiation security annotation linked, and since the security policy is an anti-pattern, it is not satisfied.

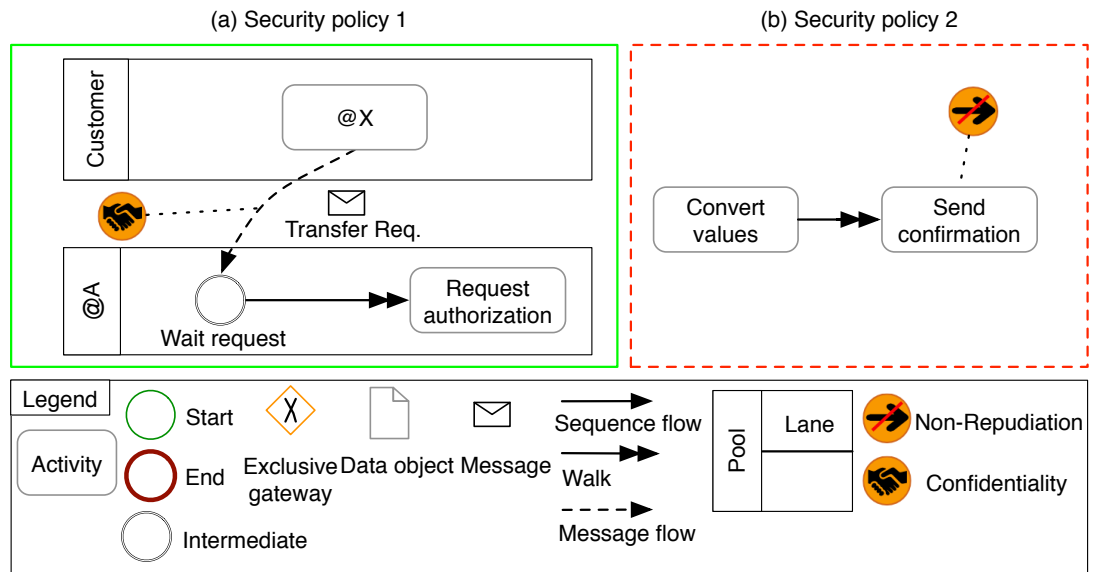


Figure 5.2: Examples of SecBPMN2-Q security policies

5.6 Verifying security policies

In the following, we formally specify the verification of SecBPMN2-Q security policy against a SecBPMN2-ml business process.

Definition 7 specifies the `correspond` function, which checks if two elements of SecBPMN2 have compatible labels and they are of the same type.

Definition 7 (Label correspondence check).

`correspond` : $(\mathcal{AEG} \cup \mathcal{D} \cup \mathcal{M} \cup \mathcal{P}) \times (\mathcal{AEG} \cup \mathcal{D} \cup \mathcal{M} \cup \mathcal{P}) \rightarrow \{\mathbf{true}, \mathbf{false}\}$
 $elem_{pol}, elem_{coll} \mapsto \mathbf{true}$ iff:

- (a) $(elem_{pol}$ and $elem_{coll})$ are elements of the same type, and
- (b) $[\mathbf{L}(elem_{pol}) = \mathbf{L}(elem_{coll})] \vee [\mathbf{L}(elem_{pol})$ starts with “@”].

Definition 7 checks if: (a) two elements are of the same type; (b) if the labels of the two elements are identical or if the label of the security policy starts with a “@” and, therefore, it matches every elements.

The `oneWalk` function in Definition 8 checks if there exists at least one walk between two elements in a collaboration model.

Definition 8 (Check walk existence).

`oneWalk` : $Walk \times SecBPMN2 - ml \rightarrow \{\mathbf{true}, \mathbf{false}\}$
 $(s, t), coll \mapsto \mathbf{true}$ iff:

- (a) $\exists x. x \in \mathcal{AEG}_{coll}. \mathbf{correspond}(x, s)$,
- (b) $\exists y. y \in \mathcal{AEG}_{coll}. \mathbf{correspond}(y, t) \wedge x \neq y$, and
- (c) exists a walk from x to y in $coll$.

Definition 8 specifies `oneWalk` function, which returns true if: (a), (b) there exists two distinct, correspondent elements in the SecBPMN2 collaboration model; (c) the elements are connected by at least a walk. The walk concept is not specified since is a well-known concept in graph theory [124].

Definition 9 defines a function that verifies if the security annotations of the policy specify stricter constraints than the security annotations of the business process.

Definition 9 (Security annotation enforcement verification).

$\text{enforces} : \text{SecBPMN2} - Q \times \text{SecBPMN2} - ml \rightarrow \{\text{true}, \text{false}\}$

$pol, coll \mapsto \text{true}$ iff:

$\forall (sa_{pol}, elem_{pol}). (sa_{pol}, elem_{pol}) \in \text{SecAss}_{pol} \rightarrow$

$\exists (sa_{coll}, elem_{coll}). (sa_{coll}, elem_{coll}) \in \text{SecAss}_{coll}.$

$\text{correspond}(elem_{pol}, elem_{coll}) \wedge \text{SAType}(sa_{pol}) = \text{SAType}(sa_{coll}) \wedge$

$\text{enfBy}(sa_{pol}) \subseteq \text{enfBy}(sa_{coll}) \wedge$

(a) $\text{SAType}(sa_{pol}) = \text{Accountability} \rightarrow \text{monitored}(sa_{pol}) \subseteq \text{monitored}(sa_{coll})$

(b) $\text{SAType}(sa_{pol}) = \text{Auditability} \rightarrow \text{frequency}(sa_{pol}) \leq \text{frequency}(sa_{coll})$

(c) $\text{SAType}(sa_{pol}) = \text{Authenticity} \rightarrow$

(i) $\text{identity}(sa_{pol}) \rightarrow \text{identity}(sa_{coll})$

(ii) $\text{authenticity}(sa_{pol}) \rightarrow \text{authenticity}(sa_{coll})$

(iii) $\text{trustValue}(sa_{pol}) \leq \text{trustValue}(sa_{coll})$

(d) $\text{SAType}(sa_{pol}) = \text{Availability} \rightarrow$

(i) $\text{level}(sa_{pol}) \leq \text{level}(sa_{coll})$

(ii) $\text{authUsers}(sa_{pol}) \supseteq \text{authUsers}(sa_{coll})$

(e) $\text{SAType}(sa_{pol}) = \text{Confidentiality} \rightarrow$

(i) $\text{readers}(sa_{pol}) \supseteq \text{readers}(sa_{coll})$

(ii) $\text{writers}(sa_{pol}) \supseteq \text{writers}(sa_{coll})$

(f) $\text{SAType}(sa_{pol}) = \text{Integrity} \rightarrow$

(i) $\text{personnel}(sa_{pol}) \rightarrow \text{personnel}(sa_{coll})$

(ii) $\text{hardware}(sa_{pol}) \rightarrow \text{hardware}(sa_{coll})$

(iii) $\text{software}(sa_{pol}) \rightarrow \text{software}(sa_{coll})$

(g) $\text{SAType}(sa_{pol}) = \text{NonRepudiation} \rightarrow \text{execution}(sa_{pol}) \leftrightarrow \text{execution}(sa_{coll})$

(h) $\text{SAType}(sa_{pol}) = \text{Privacy} \rightarrow \text{sensitiveInfo}(sa_{pol}) \subseteq \text{sensitiveInfo}(sa_{coll})$

- (i) $\text{SAType}(sa_{pol}) = \text{BindOfDuties} \vee \text{SAType}(sa_{pol}) = \text{SeparationOfDuties} \rightarrow$
- (i) $\text{dynamic}(sa_{pol}) \leftrightarrow \text{dynamic}(sa_{coll})$, and
- (ii) sa_{pol} and sa_{coll} are connected to the same elements;

Definition 9 specifies that all security annotations in a security policy are enforced if, for all of them, there exists at least one security annotation in the SecBPMN2-ml collaboration model that is connected to a correspondent element, see Definition 7, and has stricter security properties. In the definition we used a set of functions which returns the value of the attribute with the same in the correspondent security annotation, specified in Table 5.1. Specifically `enfBy`, `frequency`, `level`, `readers`, `writers`, `execution`, `sensitiveInfo`, `dynamic` return the value specified in the correspondent attributes and:

- `identity` returns the value specified in `ident`, false if it is not defined
- `authorization` returns the value specified in `auth`, false if it is not defined
- `trustValue` returns the value specified, 0 otherwise
- `authUser` returns the value specified, empty set otherwise
- `personnel` returns the value specified, false if it is not defined
- `hardware` returns the value specified, false if it is not defined
- `software` returns the value specified, false if it is not defined

Points (a-i) use these functions to specify, for each security annotation type, how the attributes of the security policy are more restrictive of the attributes of the security annotation of the collaboration model.

- *Accountability* (a): are all the monitored users specified in the policy also monitored by the business process?
- *Auditability* (b): is the frequency of the checks specified in the business process higher than or equal to the one specified in the business process?

- *Authenticity* (c): if the attribute **ident** is true in the security annotation specified in the security policy (every user has to be identified), then is the same attribute specified in the business process also true? The same criteria is used also for attribute **auth**. The **trustValue** defined in the security annotation of the security policy has to be less or equal that the value defined in the one specified in the business process, since the security annotation is satisfied when the trust provided by the executor of the task is higher than that required by the policy.
- *Availability* (d): is the value specified in the business process higher than the value specified in the policy?
- *Confidentiality* (e): is the set of authorized users specified in the business process a subset of the set of authorized users of the security annotation of the security policy?
- *Integrity* (f): if the **personnel** attribute (of **IntegrityAct**) is true in the security policy, is it also true in the business process? The same criteria applies for **hardware** and **software**. The other two variants of integrity do not need special criteria because they are characterized only by the attribute **enfBy**, that is already checked in the first two lines of the algorithm.
- *Non-repudiation* (g): is the attribute **execution** set to the same value in both security annotations?
- *Privacy* (h): is the set of sensitive information specified in the security policy included in the set specified in the business process?
- *Bind of duty/separation of duty* (i): is the attribute **dynamic** set to the same value in both security annotations? Are the security annotations connected to the same elements?

Definition 10 specifies a function that verifies a SecBPMN2-Q security policy against a SecBPMN2-ml collaboration model.

Definition 10 (Verify security policy).

$\text{verifySecurityPolicy} : \text{SecBPMN2-Q} \times \text{SecBPMN2-ml} \rightarrow \{\text{true}, \text{false}\}$
 $\text{pol}, \text{coll} \mapsto \text{true}$ iff:

- (a) $\forall x. x \in \mathcal{AEG}_{\text{pol}} \cup (D)_{\text{pol}} \cup (M)_{\text{pol}} \cup (P)_{\text{pol}} \exists y \in \mathcal{AEG}_{\text{coll}} \cup (D)_{\text{coll}} \cup (M)_{\text{coll}} \cup (P)_{\text{coll}}. \text{correspond}(x, y)$,
- (b) $\forall (x, y). (x, y) \in \text{controlFlow}_{\text{pol}} \rightarrow (x, y) \in \text{controlFlow}_{\text{coll}}$,
- (c) $\text{enforces}(\text{pol}, \text{coll})$,
- (d) $\forall \text{walk}. \text{walk} \in \text{Walk}_{\text{pol}} \rightarrow \text{oneWalk}(\text{walk}, \text{pol}, \text{coll})$,
- (e) $\forall n\text{Walk}. n\text{Walk} \in \text{NWalk}_{\text{pol}} \rightarrow \text{oneWalk}(n\text{Walk}, \text{coll}) = \text{false}$, and
- (f) $\forall (x, y). (x, y) \in \text{NCF}_{\text{pol}} \rightarrow \nexists (x_1, y_1). (x_1, y_1) \in \text{controlFlow}_{\text{coll}}. \text{correspond}(x, x_1) \wedge \text{correspond}(y, y_1)$.

A SecBPMN2-Q security policy is verified in a SecBPMN2-ml collaboration model if: (a) for all the elements of the security policy there exists at least one element with a correspondent label; (b) for all control flow relations in the security policy there exists a control flow in the collaboration model between two correspondent elements; (c) all security annotations of the security policy are enforced; (d) for each walk relation of the security policy there exists a walk in the collaboration model; (e) for each negative walk relation of the policy no walk connects the two elements; (f) for each negative control flow there is no correspondent control flow in the collaboration model.

5.7 Chapter summary

In this chapter we proposed SecBPMN2 a modeling language for business process with security concepts and for security policies.

SecBPMN2-ml meets SC1, it is used in SEBE to specify business processes (SC1.1) and security concepts in business processes (SC1.2). More-

over, the definition of its abstract syntax is formally specified and this permit to specify business process and security concepts precisely (SC1.2).

SecBPMN2-Q meets SC2, it is used in SEBE to represent security policies (SC2.1) with a precise syntax, formally defined (SC2.2).

Moreover, the formal specification of how the security policies are satisfied against business processes, specified in section 5.6, meets SC3.1, which requires an analysis that permits to verify procedural patterns against business processes, and SC3.2, which requires an analysis that permits to verify security constraints against business processes.

Chapter 6

Connecting multiple perspectives with SEBE

This chapter describes how the social/organizational business process and implementation perspective are integrated by the SEBE method. This key feature of the method is the central contribution and what it distinguishes SEBE from other security methods, which cover only partially security when engineering secure business processes. To integrate those perspectives we defined a set of transformation and verification rules to permit the transition from STS-ml to SecBPMN2 models and from SecBPMN2 to RDL, the implementation language we chose.

6.1 Designing secure business processes

The design of secure business process is one of the distinguish features of the SEBE method. The design consists in deriving secure business processes from the social/organizational model and verifying if the security requirements specified in the social/organizational model are satisfied by the business processes.

The derivation of secure business processes is semi-automated: an initial business process is generated using a set of transformation rules and then it is modified by business process experts. The complete automation, as far as our knowledge goes, cannot be achieve because the information present

in the social/organizational model are not enough. For example, in STS-ml the concept of sequence of actions is missing, but it is central in SecBPMN2.

The verification of security requirements plays a central role in this context [94,105], because business process designers will enrich the models generated, and these modifications may lead to business processes that do not satisfy the social/organizational security requirements.

6.1.1 Mapping STS-ml concepts to SecBPMN2 concepts

The design of secure business processes is a complex operation, especially in real-world socio-technical systems, which are frequently large and complex, with tens of actors and social interactions and hundreds of objectives to achieve [104]. An analysis of the social/organizational perspective can be used to ease the specification of the business processes by specifying security requirements from a high-level of abstraction point of view and by organizing the knowledge of domain experts and business process designers. Still, social/organizational models of real scenarios may be complex and the derivation of business processes can be a difficult operation. To facilitate this derivation, we propose a set of transformation rules to generate SecBPMN2 business processes that are coherent with the STS-ml models.

The transformation is based on the mapping shown in Figure 6.1(a,b). Social/organizational concepts (Figure 6.1(a)), represented by STS-ml's concepts, are linked to business process concepts (Figure 6.1(b)), represented by SecBPMN2's.

In particular, STS-ml concepts of **role** and **agent** are mapped to the SecBPMN2 concepts of **lane** and **pool**. Roles and agents represent autonomous active entities; similarly pools and lanes represent autonomous participants that execute processes.

The concept of **document** in STS-ml is mapped to the concepts of **data object** and **messages** because documents, data objects and messages represent physical objects in which information is stored.

The STS-ml concept of **goal** is linked to the concept of **process**, because

the execution of a process can achieve one or more goals.

Read, **modify** and **produce** relations are linked to **process** because they can be modeled as a process that reads and/or writes a data object, that is, in turn, linked to a document.

And-decomposition and **or-decompositions** (And-dec. and Or-dec in the figure), are mapped to **process**, because a decomposition can be seen as a process which calls the sub-processes that are the decomposed goals. For example, the and-decomposition in Figure 3.1 of **Value received** goal, can be mapped to a business process which calls the processes mapped to **Value accredited** and **Dst transfer authorized**.

The **transmission** of a document is linked to **process** because it can be modeled as a process that transmits a message, that is in turn is linked to the STS-ml document.

Delegation represents transfer of a responsibility of a goal between actors. In STS-ml a delegation always corresponds to a commitment of the delegatee to achieve the goal, i.e., the delegatee cannot reject the delegation. We opted for a more realistic specification in SecBPMN2, therefore a delegation can be modeled as a business process used, to negotiate the terms of the delegation and possibly, reject it. For example, the delegation of **Dst transfer authorized** in Figure 3.1, can be mapped to a process that is executed to negotiate the quality of the service or the amount of money requested to achieve the goal. In a delegation relation the goal delegated is a secondary element, that is linked to a call activity. This permits to insert a call to the process, linked to the delegated goal, in the process for the delegation.

6.1.2 Transforming STS-ml models into SecBPMN2 models

Such mapping permits to define complex transformation rules that allow transforming parts of STS-ml models to SecBPMN2-ml business process models. Unfortunately, there is not a commonly shared definition of how social/organizational concepts can be transformed in business processes.

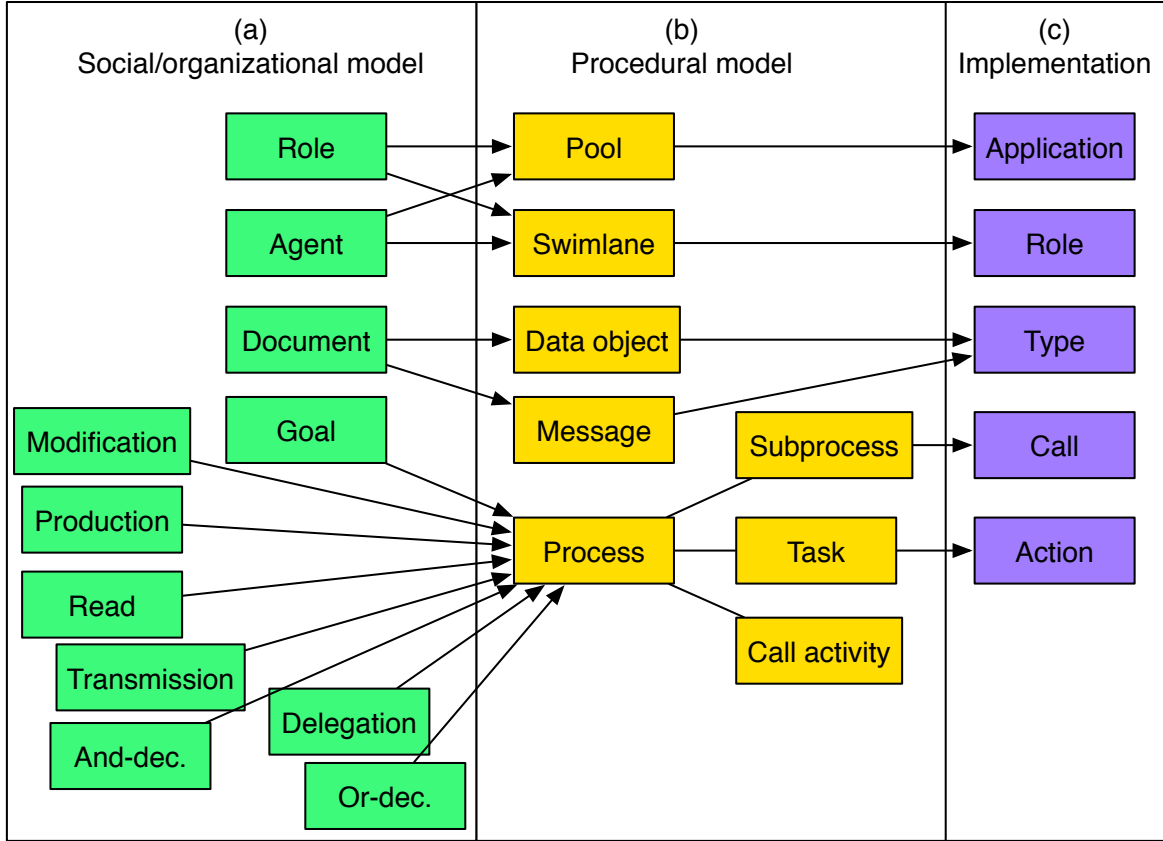


Figure 6.1: Mapping of STS-ml, SecBPMN2 and RDL concepts.

Such transformation rules change with respect to contexts, stakeholders, and business process designers. For these reasons, we defined a language that permits to specify custom transformation rules. However, we defined a generic transformation rule for each mapping, which can be found in Appendix A, in order to help unexperienced users.

Figure 6.2(a) exemplifies transformation rule. The upper part of the figure shows part of an STS-ml model, namely a transmission, that will be transformed. Each element's label starts with the keyword \$ followed by a unique string to identify the element. The rest of the figure contains the SecBPMN2-ml model that will be generated and a list of mapping relations that links the STS-ml elements with the SecBPMN2 elements. The name of the elements in the SecBPMN2 part may contain references to STS-ml

elements, identified with a string between $\{$ and $\}$, that will be substituted with the name of the STS-ml elements, when the transformation rule is instantiated. For example, the message $\{D\}$ will take the name of the document $\{D\}$.

The transformation rule generates two business processes, where the task of the upper one sends a message to the task of the lower one. The execution of the processes transmits the message from participant $\{B\}$, that is mapped to agent $\$B$, to participant $\{A\}$, which is mapped to role $\$A$. The STS-ml document transmitted is mapped to the message $\{D\}$, while the two processes are mapped to the transmission relation.

Figure 6.2(b) shows an example of the application of this transformation rule. The upper part shows the transmission of document *Transfer order* from the *Payment Engine* to the *Bank src*. The middle part of the figure shows the generated business process model with *Payment engine* instead of $\{B\}$, *Bank src* instead of $\{A\}$ and *Transfer order* instead of $\{D\}$. The instance of the mapping, i.e., the bind between STS-ml and SecBPMN2 elements, is shown in the lower part of the figure and it will be stored to be used in the fifth activity of the process in Figure 4.2.

Multiple transformation rules can be defined for the same STS-ml element. In SEBE we define a set of default transformation rules that covers most of the STS-ml elements. Transformation rules can be applied only to the social view of STS-ml diagrams, because they contain STS-ml elements that can be transformed to one or more SecBPMN2-ml models. Instead, the concepts used in the authorization and information views are used to specify security requirements.

Table 6.1 shows the relevant default transformation rules provided in SEBE, the complete list can be found in the appendix. Each transformation rule contains a STS-ml element or relation, a SecBPMN2-ml diagram and a mapping, which links STS-ml elements with SecBPMN2-ml elements.

Agent element is transformed into a business process where the participant, who executes the business process (represented by a pool), is mapped to the agent. The name of the participant is the string $\{A\}$ which is the

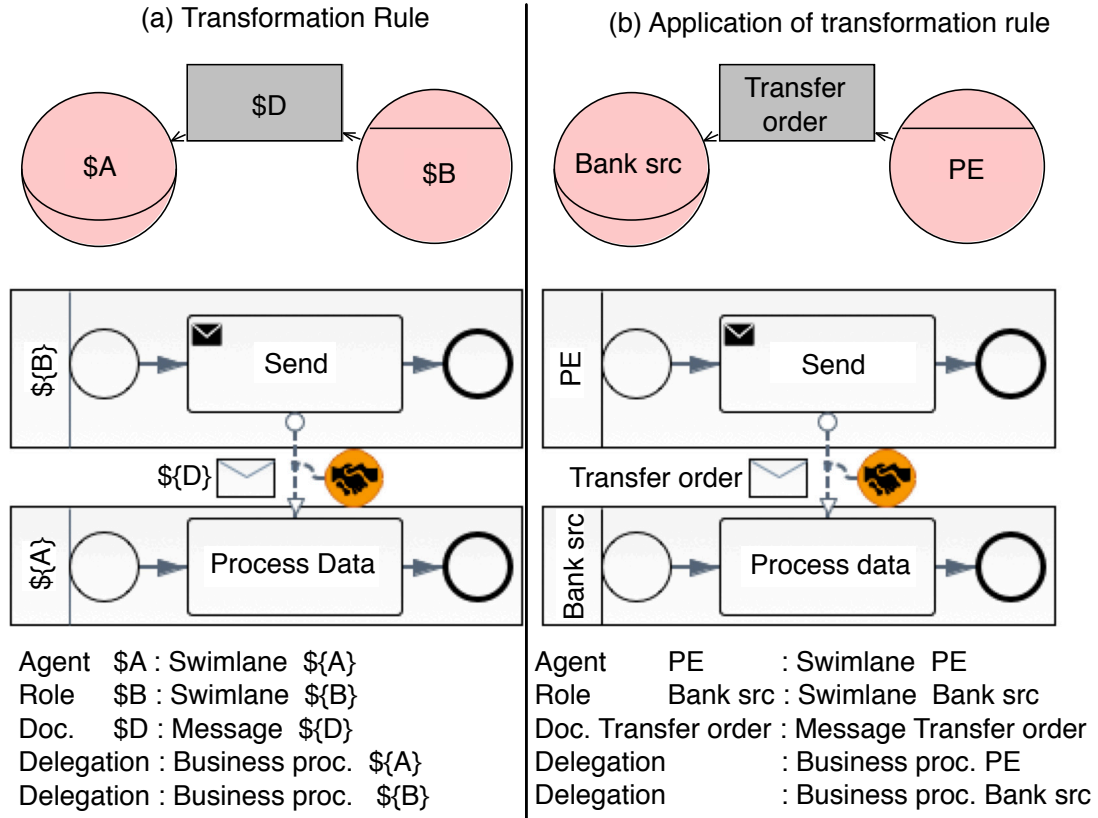


Figure 6.2: Example of a transformation rule and its application.

variable that refers to the name of the agent, therefore, when the transformation rule is used to generate a business process, the participant takes the name of the agent. We define a similar transformation rule for role element.

The **goal** element is transformed into a business process where the actor, who achieves the goal, is mapped to the participant that executes the process. The process is mapped to the goal. Goals are always assigned to one actor therefore this transformation can always be applied.

The **read** relation is transformed into a business process with a data object that is used as input by the task of the process. The document is mapped to the data object, the goal is mapped to the process, while the actor is mapped to the participant that executes the process.

We defined similar transformation rules for modify and produce rela-

tions. The difference consists in the data associations between the activities and the data objects. For the **produce** relation the data object is used only as output of the task, since to produce a data only the write operation is performed. For the **modify** relation the data object is used both as input and output, because modification consists in both reading and producing (writing) operations.

We consider a **delegation** as a negotiation that, if successfully executed, leads the delegatee (i.e., the actor who receives the delegation) to achieve the delegated goal. In business process terms, a set of business process are executed to negotiate the terms of the delegation. If the negotiation is successful, it will leads the delegatee to execute the process mapped to the delegated goal.

We, therefore, defined a transformation rule for the delegation where it is transformed in two business processes one executed by a performer mapped to the delegator (i.e., the actor that initially owns the goal) and one executed by the performer mapped to the delegatee. The delegator, sends a request to the delegatee through a message flows. As soon as the delegatee receives the message, it will execute the process mapped to the goal. We use a call activity element to represent the call to such business process.

AND-decompositions are used to create a hierarchy of business processes. The semantic of an AND-decomposition consists in specifying that all the subgoals have to be achieved, in order to consider the decomposed goal achieved. Since goals are mapped to processes, whenever a goal is AND-decomposed we insert in the associated process a call to each process that achieves a subgoal. For example, in Figure 3.1 the goal **Value transferred** is AND-decomposed in **ID bank received** and **Transfer authorized**. Such goals are transformed in three processes, one for each goal, and in the process associated to the decomposed goal, i.e., **Value transferred**, two call activities are inserted and linked to the processes linked to the subgoals, i.e., **ID bank received** and **Transfer authorized**. The user then specify the business process interpretation of an AND-decomposition, e.g., as parallel calls to

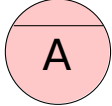

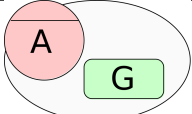
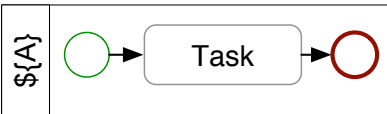
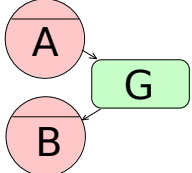
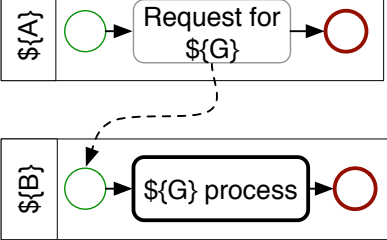
Name	STS-ml element	SecBPMN2-ml business process	Mapping
Agent/Role			Agent/Role : Pool
Goal			Actor : Pool, Goal : Process
Delegation			Actor A: Part. $\{A\}$, Actor B: Part. $\{B\}$, Delegation : Proc.

Table 6.1: Default SecBPMN2-ml transformation rules provided in SEBE

the processes linked to the subgoals, or as sequential calls.

We use the same approach for **OR-decompositions**. The user will specify their business process interpretation, e.g., using exclusive gateways to call a subset of processes linked to the subgoals.

We defined generic business process transformation rules because they highly depend on the context in which they are used. However, the users of SEBE can modify them, or create a completely new ones, using the modeling language for the transformation rule definition described above in this section.

6.1.3 Security policies generation

We provide default transformation rules for all the social/organizational security requirements, making this transformation transparent to the users of SEBE. The STS-ml PE model in Figure 3.1 specifies 12 security requirements, 4 specified in the social view, and 8 specified in the authorization view. Each security requirement can be transformed in one procedural security policies, using the default transformation rules or a customized one.

Table 6.2 shows three examples of security policy transformation rules we provide with SEBE. For the complete list of the transformation rules is shown in Appendix B. **Fall-back multi-actor redundancy** security requirements requires the executor to adopt a primary strategy to fulfill the goal, while other strategies are maintained as backup, and are used only if the primary strategy fails; the strategies must have been executed by different actors [89]. We consider such security requirements enforced in a SecBPMN2 process if (i) the goal targeted by the security requirements is linked to two call activities executed by different actors and (ii) the two call activities are linked and executed sequentially. We verify these properties by creating a security policy in which: (i) two call activities are mapped to the process that will be executed to achieve the goal, and are executed by two different participants. We used a separation of duty security annotation to specify that the participants cannot be the same entity.

Availability requires the transmission to be accessible whenever is required [101]. We consider such security requirement enforced in a SecBPMN2 process if the channel used to send the messages linked to the document delivers all messages. We verify this property with a security policy in which two pools exchange a message through a message flow, which linked to an availability security annotation. The source pool and the destination pool are mapped to the source agent and the destination agent, while the document is mapped to the message. The availability security requirement can be relaxed by specifying the percentage of delivered messages as a property. Similar property can be specified in the availability security annotation. In this case we consider the security requirement enforced if the percentage specified in the security annotation is higher than the one specified in the security requirement. For more information please refer to [101].

We transform the lack of authority to **modify** an information, which we interpreted as the security requirement of not modifying that information, in a security policy. But the STS-ml concept of information is not linked to

any SecBPMN2 concept, instead it is linked, via the made-tangible-by relation, to the document concept that, in turn is mapped to the SecBPMN2 concept of data object. Therefore, we consider the security requirement enforced in a SecBPMN2 model when the data objects mapped to the documents, that make tangible the information of the security requirement, are never modified. We verify this property with an anti-pattern (red dashed border) security policy where a data object, mapped to one of the documents that make tangible the information targeted by the security requirements, is modified (the data association that goes to the data object) by any task that is executed by a pool mapped to the actor targeted by the security requirement.

6.2 Implementing secure business processes

The generation of the implementation of a socio-technical system is one of the main objectives of SEBE. But generating a complete implementation of a system from a SecBPMN2 model is not possible, because the information contained in the models is not sufficient. Each task in the business process diagrams is identified with a short string whose interpretation changes with respect to the context and the reader of the diagram. As far as our knowledge goes, it is not possible to generate executable code from such short strings. For example, the task **Store failure report** in Figure 5.1 can be interpreted as the action of saving the report in a data base, or creating a backup in a different storage unit, etc.

For such reasons, we chose to generate part of the implementation that is still central for most socio technical systems, but minimally relies on the functional part: the business artifacts, i.e., entities, data and documents [99].

We chose River Definition Language (RDL) [109] as the implementation language. RDL is an executable specification language that allows specifying declaratively: (i) the artifacts (e.g., entities); (ii) the relationships between them (e.g., associations); (iii) the business logic (e.g., actions) on

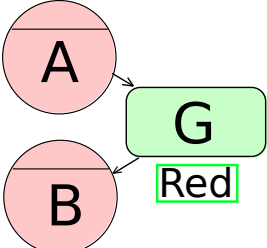
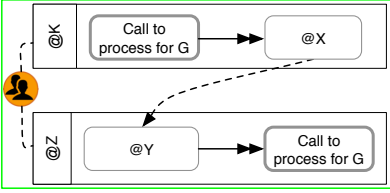
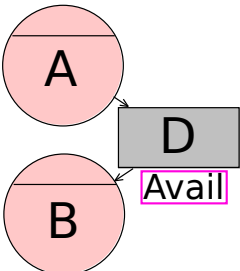
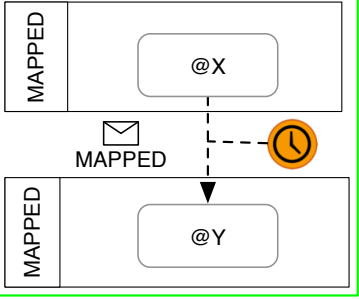
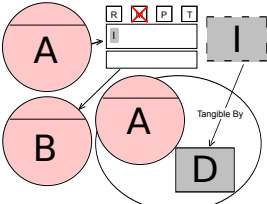
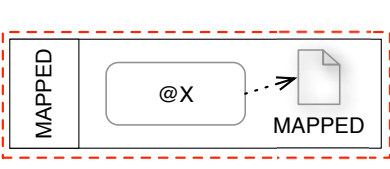
Security requirement	SecBPMN2-Q transformation rule	Mapping
<p>Redundancy</p> 		<p>Goal: Process called</p>
<p>Availability</p> 		<p>Agent/Role : Pool Document: Message</p>
<p>Non-reading</p> 		<p>Agent/Role A : Part., Doc. : Data object</p>

Table 6.2: Default SecBPMN2-Q transformation rules provided in SEBE

the artifacts. We opted for RDL because it is focused on the business artifacts, it includes an authorization system as first class citizen, and it is based on SAP HANA¹, a well known and widely used relational database management system. RDL implementations generated with the transformation rules we provide, can be directly deployed. For further details please refer to [99].

6.2.1 Mapping SecBPMN2 concepts to RDL concepts

Figure 6.1(b,c) shows the mapping between SecBPMN2 concepts and RDL concepts, which we used to define the transformation rules to generate RDL applications from SecBPMN2 models.

A **Data object**, which represents a set of information, is mapped to RDL **Type**, which represents the structure of the information of an element in entity. A **Message**, that represents a set of information sent between pools, is linked to **Type**, which in this case represents the structure of a message sent in RDL.

A **Pool**, which defines a organization or an actor such as a buyer or a manufacturer, is linked to an **Application**, that represents a set of business artifacts, which can be accessed only using the APIs, and their business logic. Both pools and a RDL applications are used to identify organizations or well-defined parts of them.

An **Task** represents an operation performed by a participant; similarly, an **Action** represents the business logic linked to a data structure, i.e., they are the operations executed to set/maintain some properties of the business objects.

A **Lane** represents a participant, i.e., a person, a service or a set of them and it is mapped to **Role**, which represents any entity that can receive an authorization.

A **Sub-process** is a task that encapsulates a business process, which contains a set of SecBPMN2 elements. It is linked to a **Call** that is a reference

¹<https://hana.sap.com>

to another RDL application, which, in turn, contains a set of business artifacts.

6.2.2 Transforming SecBPMN2 models into RDL scripts

We defined in [99] a set of transformation rules, based on the mapping described above, to generate RDL applications that enforce the security choices specified in business process models. If the business process models verify security policies generated from social/organizational security requirements, the generated applications meet the initial security requirements.

The creation of RDL implementation is based on generation rules that follow the mapping relations defined above. Events and gateways elements are, however, not part of the model transformation since they are used to define the control flow. The main generation rule specifies that a RDL entity and a RDL type are generated for each data object and for each message in the SecBPMN2 model. Each entity contains one element of the type generated together with the entity. Each task linked to the data object is transformed in an action, that is placed in an ad-hoc namespace, created for the data object.

Figure 6.3 shows an example of generation. The name of the application reflects the name of the pool and two roles, which correspond to the lanes in the business process, are specified. From the data object **VCNUM** are generated: (i) a type **VCNUM** that contains the structure of the data that makes tangible the information; (ii) an entity **VCNUMEntity** that contains the actual information; (iii) a namespace **VCNUMnamespace** that contains all the actions derived from the SecBPMN2 activities linked to the **VCNUM** data object. The structure of **VCNUM** is retrieved from the SAP repository, indeed **VCNUM** is a transformation rule for the information related to credit cards.

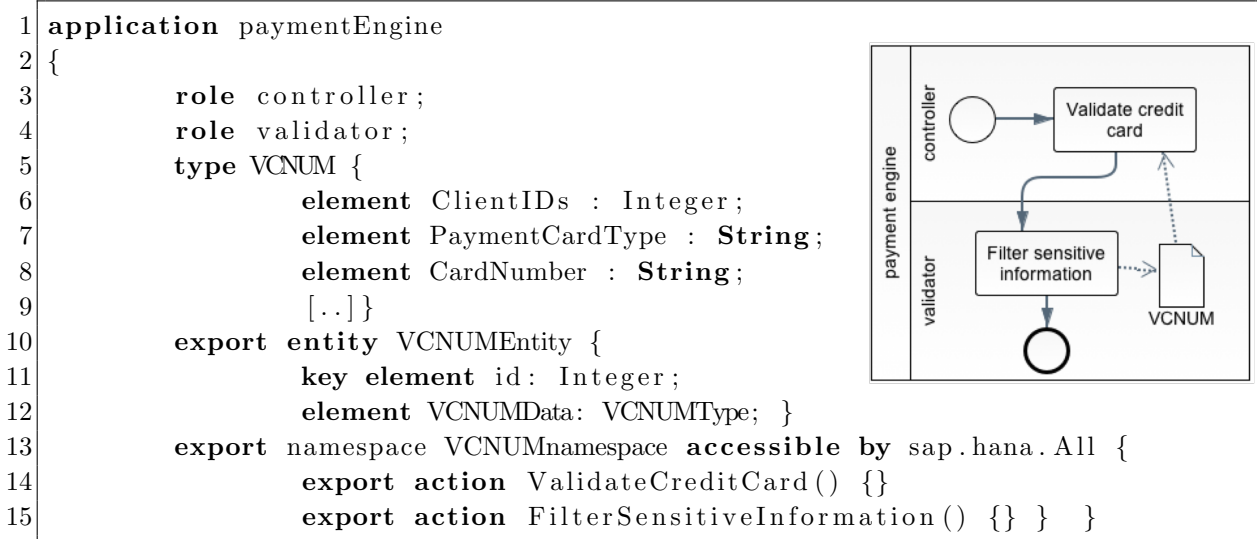


Figure 6.3: A SecBPMN2 model representing part of a PE business process

6.2.3 Enforcing SecBPMN2 security specifications

In this section, we present a set of rules that are used to enforce the security choices of SecBPMN2 into RDL applications. For each SecBPMN2 security annotation, we briefly describe in the following its meaning and the corresponding Enforcement Rule (ER).

ER1: Integrity. It requires a system to ensure completeness, accuracy and absence of unauthorized modifications in all its components [101]. It can be linked to one task, data object or message flow. Although, it can be partially enforced by filtering the users who can access the RDL entities (i.e. using authentication and access control), backup mechanisms should be used to avoid losing potentially precious information (when linked to a data object), or losing functionalities offered by the system (when linked to the message flow or to a task). Since, such configurations cannot be specified in a RDL application, they are enlisted in the security specification document.

ER2: Authenticity. It is defined as the ability of a system to verify

identity and to establish trust in a third party and in information it provides [101]. It can be connected to one task or data object. When it is linked to `#Task`, it can be enforced with an authenticity security mechanism that verifies the identities of users who execute the action generated from `#Task`. When it is linked to `#DO`, an element which contains the hash of the type generated from the `#DO` will be included in the type itself.

ER3: Accountability. It is defined as the ability of a system to hold users responsible for their actions (e.g., misuse of information) [101]. It can be linked to a task. It is enforced using the signature security mechanism, which stores the private key of the user who performs the action generated from `#Task`. We used private key to unequivocally identify users that performed the action.

Listing 6.1 shows part of the RDL transformation rule used to generate the RDL code for the signature security mechanism. A RDL transformation rule is a piece of RDL application with placeholders, marked with a `#`, that are substituted with an appropriate string. For example, in Listing 6.1, `#Pool` will be substituted with the name of the pool in `SecBPMN2`. The entity in lines 2–4 contains the private keys associated to the users, while the entity defined in lines 5–8 stores the link to the signature of the user who executed the action, the date in which the action is performed and the link to the entity that contains the action performed. Lines 10–15 show how the signature security mechanism is implemented in each action generated from `#Task`: a new entry is inserted in the entity `SignatureLogs`.

ER4: Non-repudiation. It is defined as the ability of a system to prove (with legal validity) occurrence/non-occurrence of an event or participation/non-participation of a party in an event [101]. It can be connected to one task or one message flow. We use the signature security mechanism to enforce the non-repudiation security choice. If the security annotation is linked to a message flow, every time send and receive actions are executed, the information about the execution is inserted in the signature entity. If the

```

1 application #Pool {
2   entity repositorySignatures {
3     signature : String;
4     user : String; }
5   export entity SignatureLogs {
6     element signature: Association to repositorySignatures;
7     element date: UTCTimestamp;
8     element #DO : Association to #DO; }
9   export namespace #DONamespace accessible by sap.hana.All {
10    export action #Task() {
11      let newSignatureLogs : SignatureLogs = SignatureLogs{
12        date : sap.hana.utils.dateTime.currentUTCTimestamp(),
13        signature : SELECT ONE repositorySignatures FROM
14          repositorySignatures
15          WHERE user = sap.hana.services.session.getUserName(), #
16          DO : this };
17      Add newSignatureLogs to SignatureLogs; } } }

```

Listing 6.1: Enforcement of accountability, implementing signature security mechanism

security annotation is linked to `#Task`, the information is inserted whenever the action, generated from `#Task`, is executed.

ER5: Auditability. It is defined as the ability of a system to conduct persistent, non-by-passable monitoring of all actions performed by humans or machines within the system [101]. It can be linked to one task, data object or message flow. It is enforced with the logging security mechanism, which stores information of the actions performed. If the security annotation is linked to `#DO`, all actions in the entity that contains the type generated from `#DO` are logged; if it is linked to `#Task`, only the calls to the action that is generated from `#Task` are stored; if it is linked to a message flow only the actions send/receive, generated from the message flow, are stored.

Listing 6.2 shows part of the RDL transformation rule for the logging security mechanism. The type `ActionType` (line 2) defines the type of actions. The entity in lines 3–6 contains: type of the action, date of execution and user who performed the action. Lines 9–13 show how the information about the execution of an action is stored in the entity `actionLog#DO`. If


```

1 application #Pool {
2   type ActionType : enum { READ; WRITTEN; SENT; RECEIVED; }
3   export entity ActionLogs#DO {
4     element date: UTCTimestamp;
5     element actionType : ActionType;
6     element user : String; } [...]
7   export namespace #DONamespace accessible by sap.hana.All {
8     export action get#DO(idEntity: Integer) : #Pool.#DOType { [...]
9       let log:ActionLogs#DO = ActionLogs#DO{
10        user : sap.hana.services.session.getUserName() ,
11        actionType : ActionType.READ,
12        date : sap.hana.utils.dateTime.currentUTCTimestamp() };
13      log.save(); }
14     export action #Task() { [...] } } }

```

Listing 6.2: Enforcement of auditability, implementing logging security mechanism

the security annotation is linked to `#DO`, information about the execution is inserted in `actionLog#DO` every time an action, defined in an entity that contains the type generated from `#DO`, is performed; if the security annotation is linked to `#Task` then the information is stored every time the action, generated from `#Task`, is performed; if the security annotation is linked to the message flow, then the information is stored every time the send and receive actions, generated from the message flow, are executed.

ER6: Confidentiality. It requires a system to ensure that only authorized users access information [101]. It is a security annotation that is linked to one message flow or one data object. We enforced it using authentication, access control and implementing encryption security mechanism.

Listing 6.3 shows part of the RDL transformation rule for the encryption security mechanism. In lines 5 and 6 the encryption and decryption functions are defined. For the sake of brevity, the algorithms used to encrypt and decrypt data are not shown. Lines 8–11 show how the functions are used to enforce confidentiality: the content of the entity/message is decrypted when retrieved/received and encrypted when is stored/sent. Therefore, the content of entity/message will be visible only in the RDL application. The encryption/decryption functions are inserted in the send/re-

```

1 application #Pool {
2   type #DO { [..] }
3   export entity #DOEntity {
4     [..]
5     action encrypt(data :#DO): #DO { [ENCRYPTION ALGORITHM] }
6     action decrypt(data :#DO): #DO { [DECRYPT ALGORITHM] } }
7   export namespace #DOnamespace accessible by sap.hana.All {
8     export action get#DO(idEnt: Integer) : #DOEntity {
9       let #DOInst: #DOEntity = SELECT * FROM #DOEntity WHERE id =
10        idEnt;
11        #DOInst.#DOData = #DOInst.decrypt(#DOInst.#DOData);
12        return #DOInst.#DOData; } } }

```

Listing 6.3: Enforcement of confidentiality, implementing encryption security mechanism

ceive actions when the security annotation is linked to a message flow, while are inserted in getters and setters of the entity which contains the type generated from #DO.

ER7: Privacy. It requires a system to obey privacy legislation and it should enable individuals to control, where feasible, their personal information (user-involvement) [101]. It is linked to one message flow or one data object. With authentication and access control, we restrict the access to authorized users. We further enforce it, encrypting the content of the entity that contains the type generated from #DO (when the security annotation is connected to #DO), or encrypting the entities sent and received (when the security annotation is linked to a message flow).

ER8: Binding of duties. It requires the same person to be responsible for the completion of a set of activities [123]. It is linked to two pools. It is enforced using authentication and access-control, ad-hoc security mechanisms. Listing 6.4 shows the transformation rule for the enforcement of binding of duties. Element `BoDUser` in line 3 contains the first user who accesses the entity and, therefore, the only one that is authorized to access the entity(s) contained in the RDL applications generated from #Pool and #Pool2. In lines 4–14 the function `CheckBoD` is defined: it checks if the variable `BoDUser` is set locally and in the application generated from

```

1 application #Pool{ [...]
2   export entity #DOEntity {
3     element BoDUser : String;
4     action checkBoD(): Boolean {
5       if (BodUser is null && getBodUser('#poolURL',id) is null) {
6         BodUser = sap.hana.services.session.getUserName();
7         setBodUser('#poolURL', id, sap.hana.services.session.getUserName())
8           ;
9         return true; }
10      else
11        if (BodUser == sap.hana.services.session.getUserName() &&
12            getBodUser('#poolURL',id) == sap.hana.services.session.
13              getUserName())
14          return false;
15        else
16          return true; } }
17   export namespace #DONamespace accessible by sap.hana.All {
18     export action get#DO(idEnt: Integer) : #DOEntity {
19       let #DOInst: #DOEntity = SELECT * FROM #DOEntity WHERE id == idEnt;
20       if (!#DOInst.checkBoD()) return null;
21       return #DOInst.#DOData; }
22     export action setBoDUser(urlPool: String, idEnt: String, BodUser:
23       String) {[.]}
24     export action getBoDUser(urlPool: String, idEnt: String): String {[.]}
25   } }

```

Listing 6.4: Implementation of dynamic binding of duties

#Pool2. If the variable is not set, it sets the variable both local and remotely, otherwise it checks if the user who is executing the action in which the **CheckBoD** method is called, is the same as the one memorized in the variable. The **CheckBoD** method will be called in any action of the entities contained in the applications generated from **#Pool** and **#Pool2** (lines 16–19).

ER9: Separation of duties. It requires two or more different people to be responsible for the completion of a set of activities [115]. It is linked to two pools. Static separation of duties [44] is enforced using authentication and access control, while dynamic separation of duties [44] is enforced with authentication, access control and ad-hoc security mechanisms. The transformation rule for enforcing dynamic separation of duty is similar to

the one for binding of duty Listing (6.4).

ER10: Availability. It requires a system to ensure that all its components are available and operational when they are required by authorized users [101]. It cannot be enforced in a RDL application because it requires configuration of the system (e.g., the configuration of the data-base management system), so the specification of using backup mechanism for `#DO` will be added to the security specification document.

ER11: Non-delegation. It requires the system to ensure that the actions are performed only by indicated actor(s). It can be linked to one task. It is enforced using access control: when `#Activity` is transformed in an action in RDL, it is executed by the roles authorized to access the tenancy/server where the RDL application is deployed. Once the action is implemented, it will not be anymore delegated to a third party.

6.3 Chapter summary

This chapter integrates three different perspectives of organizations: the social/organizational perspective, the business process perspective and the implementation.

It meets SC4 since SC4.1 requires a systematic approach to derive business processes from social/organizational concepts, and SC4.2 requires a systematic approach that permits to verify the enforcement of security requirements. In particular, SC4.2. requires the contribution of this chapter and Chapter 5 because the transformation of security requirements in security policies alone is not sufficient, the latter have to be verified and this is possible thanks to the verification specified in Chapter 5.

Moreover, this chapter meets SC5.1 which asks for systematic approach that provides transformation rules to generate the implementation code.

Chapter 7

Software support for SEBE

SEBE is provided with STS-Tool [106], a software tool¹ that fully supports the method.

STS-Tool constitutes a fundamental part of SEBE, since business process designers will use it through all the phases of the SEBE process. In particular the tool will help with the usage of the modeling languages, the application of transformation and verification rules and generation of the implementation.

STS-Tool was developed to be as easy as possible: it is provided with manuals and tutorials, and application of transformation and verification rules are transparent to the users. Still, STS-Tool permits a complete customization of all the transformation rules used to generate business processes and of the mapping between STS-ml and SecBPMN2 elements in order to allow expert users for a detailed control of the transformations.

STS-Tool builds on a previous version of the same software which permits to model STS-ml diagrams. We extended the tool with a plug-in system, and we created two plugins: (i) one plugin supports the creation of SecBPMN2 diagrams, the transformation of STS-ml in SecBPMN2-ml secure business processes, of STS-ml security requirements in SecBPMN2-Q security policies and the verification of security policies against secure business processes; (ii) another plugin supports the creation of RDL applications and their modification. The extension of STS-Tool was developed by

¹STS-Tool can be downloaded from www.sts-tool.eu

Mauro Poggianella following the design specified by Mattia Salnitri (the author of this thesis), with the help of Elda Paja and Paolo Giorgini.

Specifically STS-Tool offers the following features.

- Graphical editor for STS-ml and SecBPMN2 modeling languages, and a textual editor for RDL.
- Consistency verification of STS-ml and SecBPMN2 models.
- Automated generation of SecBPMN2-ml business processes from STS-ml elements.
- Automated generation of SecBPMN2-Q security policies using default transformation rules.
- Integration of STS-ml and SecBPMN2:
 - the tool permits to easily change between STS-ml and SecBPMN2 diagrams;
 - element highlights: the tool permits to highlight which STS-ml elements are linked to SecBPMN2 business processes;
 - mapping visualization: the tool visualizes the mapping between STS-ml and SecBPMN2 elements;
 - security policy generation wizard: the tool offers the possibility to use a wizard to generate ad-hoc security policies.
- Automated verification of security policies: the tool enables the automated verification of SecBPMN2-Q security policies against SecBPMN2-ml business processes
- Automated generation of security reports: STS-Tool generates a security report which contains information about all STS-ml and SecBPMN2 models and RDL applications.

Figure 7.1 shows three screenshots of STS-Tool.

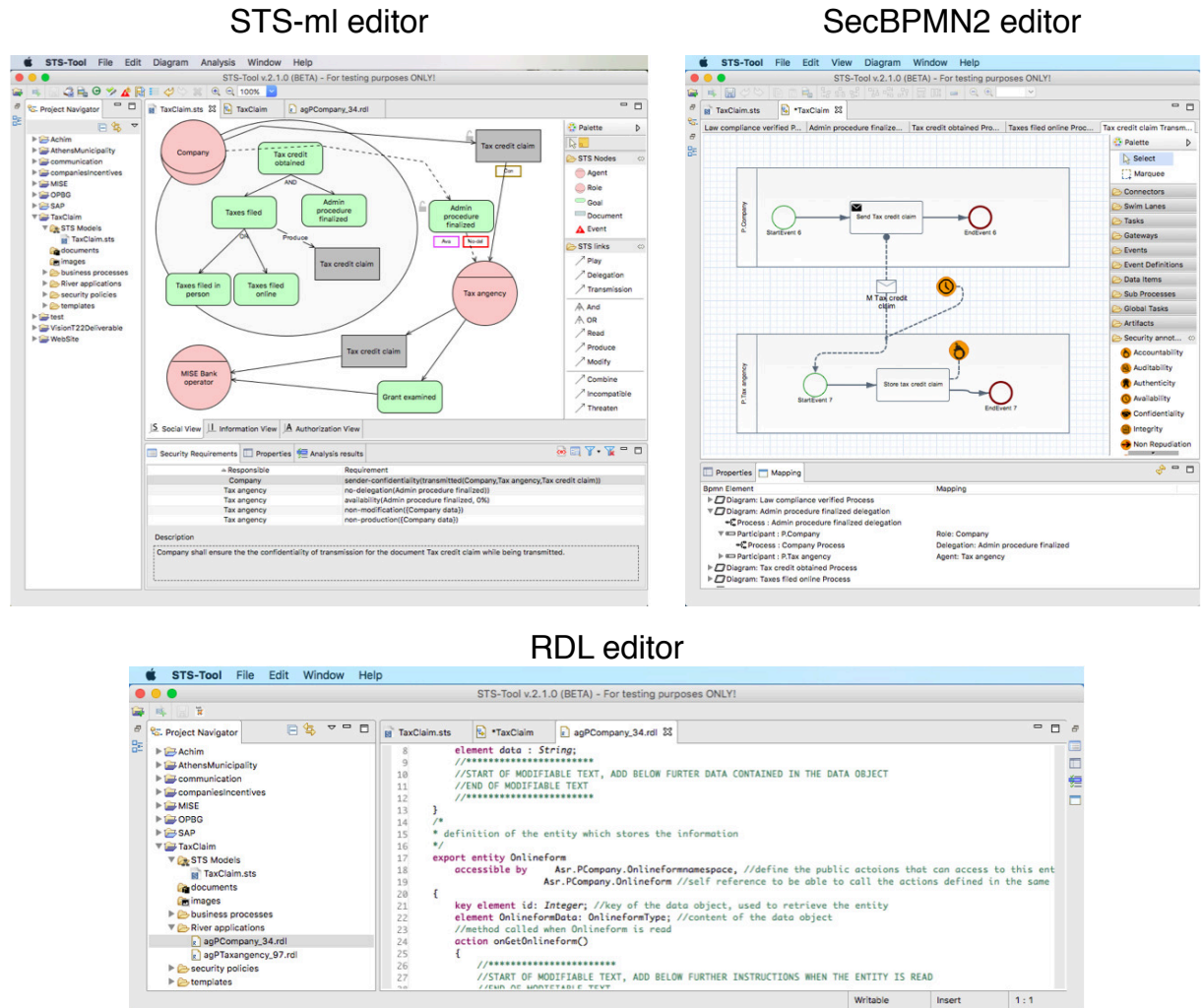


Figure 7.1: Screenshots of STS-Tool.

7.1 Consistency verification

Consistency verification comprehend the verification of the syntax of the model and the verification of social/organizational security requirements against the social/organizational model. If security requirements are not verified, the business process models derived from the social/organizational models will inherit security inconsistencies leading to an implementation that, as a consequence, will not enforce the security requirements.

Consistency verifications may be executed online and offline. The latter consists in verification explicitly executed by users of STS-Tool or when

some execution trigger the analysis, for example, before the generation of secure business process the social/organizational model consistency is verified. The former is executed every time the users modify a diagram, in order to highlight immediately the errors committed.

The online consistency verification is preferable when the verification algorithm is lightweight and the result does not prevent the users to continue drawing the diagram. Otherwise, the offline verification is preferable in case of heavy verification algorithms or when the interpretation of the results is not immediate and requires the users to stop modeling to analyze them.

The execution of verification algorithms of the syntax of STS-ml diagrams and the security requirements against social/organizational models with medium size models takes, on average, few seconds we, therefore, decided for an offline verification. Such algorithms and their implementation are not part of this thesis and were provided in the previous version of STS-Tool. Instead, the verification of the syntax of SecBPMN2 models is lightweight and we implemented it as an online verification.

With respect to the SEBE process, phases 2, 4, and 6, are executed only if the diagrams they receive in input are consistent, in order to generate respectively consistent business process models, security policies, and the implementation.

7.2 Automated generation of secure business processes and security policies

The STS-Tool automates the generation of business process models, security policies and the implementations, i.e., phases 2 and 4 of the process provided with the SEBE method proposed in this thesis. The implementation is based on the transformation rules described in Section 6, for phases 2 and 4.

The software fully supports the transformations and it permits to modify

the transformation rules used for the generation of business processes and security policies. Still, the tool is provided with default transformation rules, listed in Appendix B, that are used transparently by not-expert users.

7.3 Procedural security policy verification

For the purposes of SEBE secure business process models should satisfy security policies, derived from the social/organizational security requirements. STS-Tool includes a verification engine which permit to easily perform such verification and visualize the results.

Few verification software engines for business process models are available in literature, e.g., [6,92]. But such approaches specify and verify only the control flow of processes. STS-Tool verifies STS-ml security policies, which can be used to specify constraints on both control flows and the message flows.

For this, we created a verification engine based on \mathcal{K} [39], which permits to check SecBPMN2-Q procedural security policies against SecBPMN2-ml business process models.

\mathcal{K} is a logic-based planning language supported by DVL^{K2} , an inference engine created to generate the set of actions to achieve a goal. In a \mathcal{K} program, actions are defined with inputs, prerequisites and the effects they have once executed. DVL^K generates a sequence of such actions to achieve a state specified in the goal. For example, in an environment where an actuator can move bricks three actions may be specified: (i) **move** the actuator; (ii) **take** the brick; (iii) **release** the brick. In this environment, we can set the goal of moving the brick from a position to another. DVL^K will generate a plan which may consists in four actions: (i) **move** the actuator over the brick; (ii) **take** the brick; (iii) **move** the actuator in the new position; (iv) **release** the brick. This sequence of action will achieve the

² DVL^K can be downloaded from <http://www.dlvsystem.com/k-planning-system/>

goal.

We encoded business processes as a set of constraints in \mathcal{K} so the plans that can be generated will follow the control flow of the business processes, i.e., DVL^K only generates plans compatible with the business processes. We encoded the procedural security policies as \mathcal{K} goals, that is, a desirable state of the world to be reached with the execution of a plan. Therefore, if DVL^K generates a plan (a set of activities that follows a business process model) that achieves a goal (a procedural security policy), the business process model verifies the security policy.

If a security policy is a pattern, STS-Tool checks if the security policy is satisfied in all business processes modeled by the business process designer, if it is an anti-pattern it checks if it is *not* satisfied in all business processes.

Table 7.1 shows the transformation rules between SecBPMN2 and \mathcal{K} . In particular, any **Task** of SecBPMN2 is transformed in a predicate `task(<Element name>)` with a parameter which is the name of the element.

An **Exclusive gateway** is transformed into `gatewayExclusive(<Element name>)` predicate while all other gateways are transformed into `inclusiveGateway (<Element name>)` predicate, since parallel, event based and complex gateways behave as inclusive gateways, i.e., is possible to execute more than one outgoing control flow. The difference between those gateways is in the semantic assigned by the business process designer to the type of decision they represent, but it does not affect the possible executions of the business process.

All **Events** are transformed to `event(<Element name>)` predicate. We do not distinguish between different types of events since all type of events have the same impact in terms of sequence of elements executed in a business process.

Data objects, **Data reference**, **Data input** and **Data output** are transformed in `Data object(<Element name>)` predicate, which represents the physical storage of information. **Messages** are transformed in the predicate `message(<Element name>)`.

Sub-processes and **Call activities** are not transformed in any predicate.

Instead, the business processes contained (in case of sub-processes) or referred to (in case of call activities), are merged to the business process which executed the call activity or the sub process. Pragmatically, we ignore the call activity, and the start and end event of the called process, and we integrate the called process by linking the element before the call activity to the first element of the called process and the its last element to the element after the call activity or sub process. Similar operations are executed for sub-processes.

Global tasks are similar to tasks, but they are use as anchor points for call activities. In term of execution global tasks can be considered as plain tasks and, therefore, they are transformed in the same predicate we used for tasks, i.e., `task(<Element name>)`.

Participants, both lanes and pools are transformed to `participant(<Element name>)` predicate.

The **Control flow** relation is transformed in a predicate `controlFlow(<Source>, <Destination>)` where the first parameter is the element source of the control flow, and the second element the target of the control flow.

The **Message flow** relation is transformed in the predicate `messageFlow(<Source>, <Destination>, <Message>)`. Similarly to the control flow predicate, this predicate has two parameters that specifies the source and the target of the relation, moreover it requires a third parameter which specifies which is the message sent with the message flow.

SecBPMN2 uses one **Data association** relation to specify if a data object is used by another element and the direction of the association (incoming or outgoing from the element) to specify if the data object is read or written. We generate two predicates depending if the data object is read, `readDO(<Data object name>, <Element name>)`, or written, `WriteDO(<Data object name>, <Element name>)`. The two predicates use the same parameters: the name of the data object and the name of the element which use the data object.

When an element is placed inside the scope of a participant an **Ownership** relation is specified between the two elements. We generate a similar

predicate, i.e., `owns(<Name pool>, <Element name>)`, to specify the same concept in \mathcal{K} . The ownership relation is not limited to elements such as tasks and data objects, but binds together also lanes specified inside pools.

For what concerns the security annotations we generate predicates as specified in Table 5.1. Such predicates already contain references to the SecBPMN2 element associated to the security annotation, (specified in SecBPMN2 with a `security association`), therefore, we do not generate a predicate for such association.

We do not generate predicates for `Artifacts`, such as group and textual annotations, since, they are irrelevant for the execution of the business process.

The predicates we defined in \mathcal{K} identify the elements of the business processes by their names, in other words two elements of the same type and name cannot be differentiated in \mathcal{K} . Therefore, we imposed a constraint that prevent to create two elements of the same type with the same name in SecBPMN2 diagrams.

Listing 7.1 shows an example of a \mathcal{K} representation of a SecBPMN2 business process model. In \mathcal{K} strings that start with an upper case character are considered variables, while strings that start with a lower case character are constants. The first one represents any task and the latter represents specific activities. Lines 1-3 define the message and the control flow fluents. The control flow fluent requires two activities as parameters and represents the control flow between two elements. The message flow fluents require as parameters two activities and a message, and represents the message flow between two activities. Lines 4-6 represent part of the business process model in Figure 5.1. In particular, the control flow between the `generate transfer order` and `send transfer order` and the message flow containing the message `transfer order` between the latter task and `analyse transfer order`.

SecBPMN2 Elements	\mathcal{K} elements
Task	task(<Element name>).
Exclusive gateway	gatewayExclusive(<Element name>).
Inclusive gateway	gatewayInclusive(<Element name>).
Parallel gateway	
Event Based gateway	
Complex gateway	
Event	event(<Element name>).
Data object	dataObject(<Element name>).
Data reference	
Data input	
Data output	
Message	message(<Element name>).
Sub process	X
Call activity	X
Global tasks	task(<Element name>).
Participant	participant(<Element name>).
Control flow	controlFlow(<Source>,<Destination>).
Message flow	messageFlow(<Source>,<Destination>,<Message>).
Data association	readDO(<Data object name>,<Element name>). WriteDO(<Data object name>,<Element name>).
Ownership	owns(<Participant name>,<Element name>).
Security annotations	see chapter
Security association	X
Artifact	X

Table 7.1: List of transformations of SecBPMN2 concepts in \mathcal{K} predicates

```

1 fluents:
2   controlFlow(T1,T2) requires task(T1), task(T1).
3   messageFlow(T1,T2,M) requires task(T1), task(T2), message(M).
4 initially:
5   controlFlow(generate transfer order, send transfer order).
6   messageFlow(send transfer order, analyse transfer order, transfer
   order).

```

Listing 7.1: A \mathcal{K} representation of part of the process in Figure 5.1

Listing 7.2 shows the \mathcal{K} goal generated from the procedural security policy in Figure 5.1(b). Line 1 specifies that a \mathcal{K} goal is defined. Lines 2-3 contain the predicate `executed` that indicates the task specified in the first

parameter must be executed by the participant indicated by the second parameter. For example the predicate Line 2 instructs the planner to find a plan in which the task `analyze transfer order` is executed by `bank src`. Line 4 specifies that the plan must contain a message sent by any task to `analyze transfer order` and transmitting the message `transfer order`. Line 5 instructs the planner to find a plan where `walk1` is executed. `walk1` is defined as a set of constraints (Lines 7-8), where `walk1_1` becomes true after `analyze transfer order` is executed and `walk1` becomes true if `accredit value` is executed and `walk1_1` is true. Line 6 specifies how many actions the generated plan shall contain maximum, we set this as the number of all SecBPMN2 elements in the business process. This parameter does not influence the generated plans, if it is high enough, i.e., as high as the longest possible execution, which in the case of this thesis is the number of elements in the business process.

If DVL^K generates at least a plan that respects the constraints defined in Listing 7.1, which achieves the goal defined in Listing 7.2, then:(i) if the policy is a pattern, the business process model used to generate the constraints is verified against the procedural security policy used to generate the goal; (ii) if the policy is an anti-pattern, the business process model is not verified against the security policy.

```

1 goal :
2   executed(analyze transfer order , bank src) ,
3   executed(accredit value , bank src) ,
4   sent(X, analyze transfer order , transfer order) ,
5   walk1
6   ? (10) .
7 caused walk1_1 after exec(analyze transfer order) .
8 caused walk1 after exec(accredit value) , walk1_1 .

```

Listing 7.2: Example of a \mathcal{K} goal generated from Figure 5.1(b)

7.4 Generation of RDL code

STS-Tool takes as inputs a SecBPMN2 model, a repository of business artifact definitions and, optionally, a set of enforcement rules. Using the gen-

eration rules described earlier, the prototype generates the RDL implementation from transformation rules using Freemarker (<http://freemarker.org>): a Java template library that permits to parse text files with placeholders and insert text only by specifying the placeholder to substitute.

Algorithm 1 shows the generation of the RDL implementation. It follows the generation and enforcement rules described in chapters 5 and 6. It uses the function **Generate** that retrieves the Freemarker templates and instantiate them using the information contained in the SecBPMN2 model. For each pool of the SecBPMN2 model (line 1), the algorithm creates a new RDL application (line 2) and it adds, to the application generated, all roles generated from all lanes contained in the pool (lines 3–5). For each data object, it creates a RDL type, entity and namespace, and add them to the application (lines 6–10). After that, for each task in the pool, it generates the corresponding action and adds it to the entity(ies) that is(are) generated from the data object that is linked to the task (lines 11–16). The **Retrieve** function checks for this link. If no data object is linked the task, the generated action is added to the application. The last part of the algorithm is for the enforcement of the security annotations: for each security annotation in the pool, **GenerateSC** instantiates the Freemarker template for the corresponding security mechanism(s) and after that **GenerateSP** generates the security specifications that are added to the security specification document.

STS-Tool is currently limited to public and private process models and collaboration models. We do not foresee any fundamental problem in extending the prototype to support SecBPMN2 choreography models.

7.5 Architecture

STS-Tool is a stand-alone software written in Java 7, based on Eclipse RCP Framework³. The software is available for MAC OS, Microsoft Windows

³<https://eclipse.org>

Algorithm 1 Algorithm for generation of RDL applications

```

GENERATERIVERAPPLICATIONS(SecBPMN2 model)
1  for each pool ∈ model
2  do riverApplication ← GENERATE(pool)
3    for each lane ∈ pool
4    do riverRole ← GENERATE(lane)
5      riverApplication.ADD(riverRole)
6    for each dataObject ∈ pool
7    do riverType, riverEntity, riverNamespace ← GENERATE(dataObject)
8      riverApplication.ADD(riverType)
9      riverApplication.ADD(riverEntity)
10     riverApplication.ADD(riverNamespace)
11   for each task ∈ pool
12   do riverAction ← GENERATE(task)
13     if LINKEDDO(task)
14       then riverEntity ← RETRIEVE(DataObject)
15         riverEntity.ADD(riverAction)
16       else riverApplication.ADD(riverAction)
17   securitySpecificationDoc ← NEW()
18   for each securityAnnotation ∈ pool
19   do securityMechanisms ← GENERATESC(securityAnnotation)
20     riverApplication.ADD(securityMechanisms)
21     securitySpecifications ← GENERATESP(securityAnnotation)
22     securitySpecificationDoc ← ADD(securitySpecification)

```

(32 and 64 bits), and Linux (32 and 64 bits).

Figure 7.2 shows the main architectural components of STS-Tool with a layered architectural style diagram. The components we created for STS-Tool have bold borders, external components we used for the extension have dashed borders, while components of the previous version of STS-Tool have thin borders.

STS-Tool is based on OSG Equinox and Eclipse RCP plugin, a well known Integrated Development Environment (IDE) for Java. BPMN2 modeler is an Eclipse plugin which permits to model BPMN 2.0 diagrams. We extended such plugin in order to (i) model secure business processes, with SecBPMN2-ml editors; (ii) model security policies, with SecBPMN2-Q editor, (iii) model secure business process transformation rules, with SecBPMN2-ml transformation rule editor; (iv) model security policy transformation rules, with SecBPMN2-Q transformation rule editor. The RDL editor per-

mits to edit RDL files. The **Security policy generator** creates security policies using security policy transformation rules. The **Transformation module** generates SecBPMN2-ml business processes from STS-ml models, using SecBPMN2-ml transformation rules, and it generates RDL applications using SecBPMN2-ml models. SecBPMN2-Q security policies are verified against SecBPMN2-ml business processes by the **Security policy verifier**, which uses the DLV- \mathcal{K} plugin.

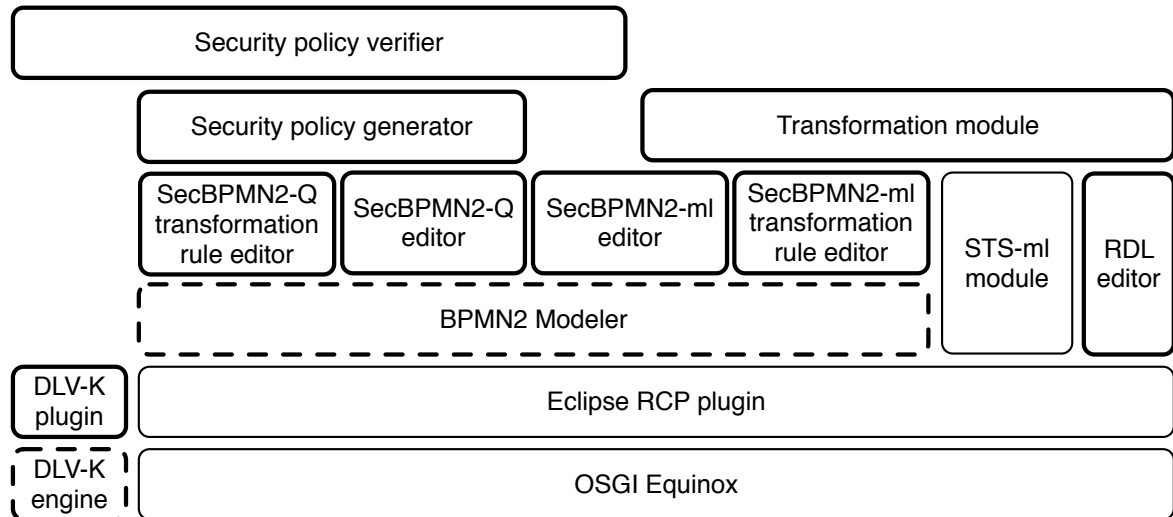


Figure 7.2: STS-Tool components

7.6 Chapter summary

This chapter describes STS-Tool, a software tool provided with SEBE. STS-Tool meets SC3.3 since it verifies security policies against secure business processes in an automated fashion, SC5.2 since it automatically generates part of the implementation, and SC6.2 because it provide a full support to the process, integrating the modeling languages and offering a rich set of functions for each phase of the process.

Chapter 8

Empirical evaluations

This chapter describes two empirical evaluations we performed to verify if the success criteria, defined in Chapter 1, are met. The first empirical evaluation focuses on SecBPMN while the second one focuses on the method as a whole.

Empirical evaluations use experiments and observation to collect empirical evidence that can be used to verify or dismiss certain hypotheses [126]. We use an engineeristic approach to reach conclusions as objective as possible, to collect impression and feedback from a sample of the possible users of SEBE.

8.1 Evaluation of SecBPMN

We designed and conducted an experiment to test the understandability and the perceived graphical complexity of SecBPMN-ml and SecBPMN-Q; we define the latter concept as semiotic clarity [78] and diagrammatic complexity [78]. Our experiment was conducted through an online survey as a way to maximize the number of subjects.

The experiment was conducted using the first version of SecBPMN. The differences between SecBPMN2 and SecBPMN are, mainly, on the possibility of further specify concepts already present in SecBPMN. For example, in SecBPMN is possible to specify only tasks, while in SecBPMN2 is possible to specify 8 types of tasks such as **Manual task**, **Script task** or **Send**

task. We believe that the evaluation is still valid for SecBPMN2 because the latter is an extension of SecBPMN which did not change the nature of the language. Indeed, we believe that a person trained for SecBPMN does not need any further training for using SecBPMN2.

8.1.1 Experimental design

The design of our experiment was conducted following Wohlin’s guidelines [126]. We used the Goal, Question, Metric (GQM) template [14] to define the scope and objectives of the survey; in particular, the GQMtemplate specifies: (i) the focus of the experiment; (ii) the objective of the experiment; (iii) the variables to test; (iv) the subjects; and (v) the context of the experiment.

Table 8.1 shows the two GQM templates for our experiment. The first part of the experiment analyzes SecBPMN-ml for evaluating its perceived graphical complexity and understandability. The experiment targets the main roles involved in the process provided with SEBE: security experts and business process designers. The latter category is a prominent subset of business analysts. The evaluation is performed by asking the subject to read SecBPMN models. The second part of the experiment compares SecBPMN-Q with a formal approach for expressing policies, i.e., Computational Tree Logic (CTL) [41] formulas.

Table 8.1: GQM template for our experiments

<p>Analyze SecBPMN-ml for the purpose of evaluation with respect to their perceived graphical complexity and understandability from the point of view of the security experts and business process modelers in the context of reading SecBPMN models.</p>
<p>Analyze SecBPMN-Q for the purpose of compare it with CTL formulas with respect to their perceived graphical complexity and understandability from the point of view of the security experts and business process modelers in the context of reading security policies.</p>

Table 8.2 shows the hypotheses that we tested with the experiments. We divide hypothesis into two sets, one per experiment: the first set compares SecBPMN-ml with BPMN models with a textual description of the security policy (BPMNts); the second set compares SecBPMN-Q with CTL.

We chose BPMNts, instead of BPMN, in order to compare SecBPMN with a modeling language with the same expressiveness. Other languages, such as SecureBPMN, can express only a part of the security concepts and, therefore, the comparison would be unfair. For the same reason we chose to compare SecBPMN-Q with CTL: they can express the same type of time patterns.

Table 8.2: Hypotheses of the experiments

Experiment 1: SecBPMN-ml vs. BPMNts
H0-1.1: SecBPMN-ml is more complex than BPMNts
H1-1.1: SecBPMN-ml is less complex than BPMNts
H0-1.2: SecBPMN-ml is less understandable than BPMNts
H1-1.2: SecBPMN-ml is more understandable than BPMNts
H0-1.3: SecBPMN-ml is more complex and less understandable than BPMNts
H1-1.3: SecBPMN-ml is less complex and more understandable than BPMNts
Experiment 2: SecBPMN-Q vs. CTL
H0-2.1: CTL is preferable to SecBPMN-Q for communication with stakeholders
H1-2.1: SecBPMN-Q is preferable to CTL for communication with stakeholders

We opted for convenience sampling as a means to recruit subjects: we did spread the word about the survey through mailing lists, used by security experts and business process modelers. We left the survey available on line for 20 days, and then we analyzed the answers.

To evaluate the perceived complexity and the readability of SecBPMN and BPMNts, we created three pairs of diagrams, each consisting of a SecBPMN-ml diagram and BPMNts diagram. To ensure a fair comparison, both diagrams modeled the same business processes, with the same security choices. We also use the same layout, except of the message flow that was colored differently (we discuss the implications in Section 8.1.4).

The survey was structured in different parts (for the details see [114]):

- General questions concerning the background of the subject;
- An introduction to SecBPMN-ml;
- Questions on a small-size business process (SecBPMN-ml/BPMNts);
- Questions on a slightly bigger business process (SecBPMN-ml/BPMNts);
- Questions on a medium-size business process (SecBPMN-ml/BPMNts);
- An introduction to SecBPMN-Q;
- Questions on security policies (SecBPMN-Q/CTL).

8.1.2 Participants

The survey was completed by 30 subjects; the large majority (96%) were familiar with at least one business process modeling language, 60 % were familiar with BPMN standard. The majority of the subjects (60%, $N = 18$) declared to have good or wide knowledge of security, while 40% of the subjects ($N = 12$) are not security experts ($\bar{x} = 3.13$ on a scale from 1 to 5, $\sigma = 1.41$).

8.1.3 Experiment results

The original results are publicly available on [114]. Let us review some key results:

- For small diagrams (respondents $N = 30$), SecBPMN-ml is largely considered more understandable and less complex than BPMNts: 80% preferred it to BPMNts, 13% rated both diagrams understandable, 7% found none of them understandable, no-one preferred BPMNts.
- For slightly larger diagrams ($N = 27$), the preference for SecBPMN-ml is confirmed, even though a smaller percentage (67%); 11% of the respondents opted for BPMNts diagrams, 18% of the respondents found

both of them understandable, and 4% of the sample found no notation understandable.

- For medium-size diagrams ($N = 25$), the majority of the subjects found SecBPMN-ml more useful to define secure business processes (80%), 8% preferred BPMNts, 8% of the subjects would choose either, and 4% of the sample would use none. When asked about which language would be more effective to communicate with stakeholders, 60% chose SecBPMN-ml, 12% chose BPMNts, 20% chose both of them and, 8% wouldn't use neither SecBPMN-ml nor BPMNts.

An interesting result is about the perceived level of security of business process modeled with SecBPMN-ml, that we tested with the second couple of diagrams. While both diagrams expressed the same security information, the majority of the subjects (74%, $N = 27$) thought that the SecBPMN diagram represents a more secure business process, 15% chose BPMNts diagram, and 11% thought that both diagram represent a business process with the same level of security. This seems to indicate that SecBPMN is more understandable than BPMNts, in the sense that it is easier to identify the security choices.

For what concerns the comparison between SecBPMN-Q and CTL, the former is preferred for communication with customers (88%), none of the subjects chose the CTL formula, 4% would use either of them, and 8% none of them. Regarding the use for verifying compliance, a CTL formula was chosen by the majority of the subjects (54%), 21% would use SecBPMN-Q, 21% would use either of them and 4% would use neither SecBPMN-Q nor CTL. This result shows that the subjects think that SecBPMN-Q is not expressive enough for the verification of security policies. However, SecBPMN-Q is based on temporal logics, and our verification engine fully supports it. It is probably the case that the respondents' opinion is due to the common knowledge about the expressiveness of temporal logic formulas, and had no knowledge on the expressiveness of SecBPMN-Q.

The last question investigated the usefulness of highlighting those walks

in a model that satisfy a given security policy; this is a key feature of our modeling and verification toolset. We asked whether this could help security experts to find causes of non-compliance; the respondents ($N = 23$) rated this feature positively, even though not extremely positively: on as scale from 1 (not useful at all) to 5 (extremely useful), we obtained $\bar{x} = 3.78$ and $\sigma = 1.00$.

With the results collected in the survey is possible to refute H0-1.1, H0-1.2, and H0-1.3 and hence confirm H1-1.1, H1-1.2, and H1-1.3. For the set of hypotheses we defined for SecBPMN-Q, it is possible to refute H0-2.1 and to confirm H1-2.1. In other words, the results of the survey constitute a preliminary evidence of the fact that SecBPMN-ml is more understandable and has a lower perceived complexity than BPMNts, and that SecBPMN-Q is preferred to CTL for communicating security policies with stakeholders.

8.1.4 Threats to validity

We report the main threats to the validity of our experiment, using Wohlin's categorization [126].

Threats to conclusion validity. The relevant threats in this category are the following: (i) low statistical power, as we cannot determine the size of the mailing lists and how many respondents in advance; (ii) random irrelevancies in the experimental setting, for we opted for an on line survey, thereby having no control on external factors which could affect the results of the experiment; (iii) random heterogeneity of subjects, as we distribute the survey on line and we were not able to select adequate participants. Looking at the obtained results, the statistical power threat is only partially addressed: while 30 respondents do not yield strong statistical power, the number is in the average for PhD studies [73]; concerning the third threat, most of the participants had knowledge in business process modeling (28 out of 30 had experience in either BPMN, Petri nets, or UML activity

diagrams), and with the subjects having a knowledge above average in information security average $\bar{x} = 3.13$ (range from 1 to 5), standard deviation $\sigma = 1.41$.

Threats to internal validity. The only relevant threat is mortality. We mitigated such threat by allowing subjects to interrupt the survey at the end of each part. 10% of the subjects interrupted the experiment after the questions about small-size business process, 7%, of the remaining subjects interrupted it in the following part, and 3% in the part about medium-size business processes. Overall, we collected results on the small-size processes from all subjects, on slightly larger models from 90% of the subjects, and on medium-size models from 80% of the sample.

Threats to construct validity. This type of validity is threatened by the restricted generalizability across constructs. In other words, some constructs (diagrams) can influence the valuation of other diagrams. In our case, the danger is that by looking at a diagram in one notation, the user already gets a sense about its meaning, and is facilitated in understanding the alternative notation. We mitigated this threat in different ways. First, since we aimed to assess SecBPMN, we presented our notation first, so that most of the cognitive effort was put on understanding the process using our languages. Second, we modeled the same business processes throughout the survey but with different level of details. For the same threat, we avoid to influence subjects with factors that are not tested in the survey using for each part the same layout and the same detail. Construct validity is also threaten by a difference in the coloring of the diagrams: while we did our best to keep the layouts as similar as possible between business processes of the same pairs, the message flows of the SecBPMN-ml diagrams are colored in blue while the same message flows in BPMN diagrams are black. Another threat to construct validity is hypothesis guessing, where the subjects can be conditioned by the results they are providing. We mitigated this threat by carefully formulating questions

as much impartially as possible, and by clearly stating the purpose of the questionnaire.

Threats to external validity. External validity is threatened by the interaction of setting and treatment. In our case, this would occur with business process diagrams that are not an accurate representation of the real process. Due to time constraints, the first two proposed business processes were relatively small; the third one, however, is medium-sized, and constitutes therefore a fair representation of a real-world business process for the chosen domain.

8.2 Evaluation of SEBE

We evaluated SEBE in two ways. First, we checked the actual interest practitioners have about such method. This was done through a series of interviews with industry experts. Second, we evaluated the method conducting an empirical study with modelers.

8.2.1 Interviews

We interviewed industry experts to evaluate the interest of organizations about SEBE. The interviews consisted in a description of the method and 10 questions. They were conducted through Voice over IP (VoIP) calls or telephone calls and lasted between 10 and 20 minutes. We interviewed 11 subjects from 10 different companies such as HP, Tecnia, ATOS, Eurocontrol, and Thales group.

64% of the subjects declared to be security experts and to perform regularly systematic security analysis, while 36% of them have a superficial knowledge on security. The subjects interested in a usage of SEBE are 64%, in particular 86% of the security experts showed high interest, while only 14% of them declared not to be interested. The subjects not interested in the method are 36% but, interestingly, 75% of them are not security

experts and only 25% of them are. None of the subjects were aware of methods similar to SEBE.

This data suggested that there is a good interest about SEBE and, if we focus on security experts, the appreciation is much higher. The interviews confirmed the lack in the market of comprehensive methods as the one proposed in this thesis and that it would be more than welcomed by security experts.

8.2.2 Empirical evaluation

We designed and conducted an empirical study to evaluate the effectiveness of the process supporting SEBE, the automated transformations, and the usability of STS-Tool.

As for the previous empirical evaluation, we used the GQM template [14] to define the scope and objectives of the survey. Table 8.3 shows a GQM templates for our empirical evaluation. The evaluation analyzes SEBE for evaluating its effectiveness and usability. We defined the effectiveness as how much SEBE helps in enforcing security requirements, while usability is how easy is to apply SEBE with the aid of the extended STS-Tool. The experiment targets business process designers, while they are enforcing security requirement, i.e. while they are using SEBE.

Table 8.3: GQM template for our experiments

<p>Analyze SEBE for the purpose of evaluation with respect to effectiveness and usability from the point of view of business process designers in the context of enforcing security requirements.</p>
--

From the GQM template in Table 8.3, we elicited the following Evaluation Question (EQ):

EQ1: Does the process effectively help business process designers? The criteria taken into account for evaluating the process are: (criteria i) how much its execution helps business process designers in specifying

security requirements; (criteria ii) how well it adapts with the software engineering process used by the subjects; (criteria iii) how easy it is to follow the process.

EQ2: Do the automated transformations effectively help business process designers? We evaluated how much automated transformations help in: (criteria i) specifying business processes; (criteria ii) specifying and enforcing security policies; (criteria iii) specifying and enforcing security policies in large scenarios.

EQ3: Is the software tool usable? We evaluated the following criteria: (criteria i) how well STS-Tool integrates STS-ml and SecBPMN2; (criteria ii) how well STS-Tool supports the process proposed in this thesis; (criteria iii) the usability of STS-Tool. We evaluated the usability with the standard System Usability Scale (SUS) [110]. The original SUS [110] was used with a slight adaptation: for each item of the scale, the word *system* was replaced with *tool*, to ease the understanding of the scale. The scale for assessing the tool consisted of 10 items, like in the original SUS. Apart from the tool usability, we asked about how well the software tool supports SEBE.

8.2.3 Experimental design

The experiment was divided in three phases, described in the following list.

1. Training phase:

- subjects attended an introduction about the SEBE;
- subjects executed simple exercises on the modeling languages and transformation between them;
- subjects attended an introduction to the extended STS-Tool.

2. Application:

- subjects performed alone an exercise that requires to follow the process and using the automated transformations with the help

of STS-Tool. During the execution of the exercise subjects navigated between social/organizational and business process models, to generate the latter from the former, and to modify the business process models;

- at the end of the application phase, subjects delivered a report with information about the execution of the exercise.

3. Evaluation:

- subjects compiled an on-line questionnaire, they had the choice to compile it right after the application phase, or in the following days.

We provided to the subjects with a cheat sheet for each modeling language (STS-ml and SecBPMN2) that consists in a table with images and descriptions of all the elements of the modeling languages.

8.2.4 Participants

20 subjects participated to the empirical study. Subjects were Ph.D. students (40%) and master students (60%), all of them expert modelers. Most of the subjects (60%) have a good knowledge on goal-based modeling languages, 35% are experienced goal modelers and only 5% have low knowledge on such modeling languages. For what concerns business process models, 15% are experienced modelers, 60% have a good knowledge on the business process and only 5% have a low knowledge on the topic. 33% of the subjects have an average security knowledge, 27% are security experts, while 40% have a low or no knowledge on security. The sample contains a heterogeneous representation of possible users of SEBE.

The exercises in the second part of the empirical study were executed to assess if the subjects correctly understood the concepts introduced in the first part. 50% of the subjects executed the exercises without errors, 40% did 1 error, while 10% did 2 errors. No subjects did more than two errors. This indicates that the baseline concepts, that was explained in the first

part of the empirical study, were well understood; a key point to properly follow the hand-on section, execute the exercise in the fourth section and, therefore, obtain significant results.

8.2.5 Quantitative data analysis

We collected data on the execution of the exercises performed during the empirical study, shown in Table 8.4. We measured the number of times the subjects transformed an STS-ml element in SecBPMN2 diagram (**# Transformations**) and the number of time the subjects modified the the generated business processes (**# Changes**) in order to check if the subjects understood the rationale behind the transformations. The higher the number of transformations and changes, the higher the probability the users comprehended how the transformation works.

STS-Tool provides to its users some functionalities that ease the transaction between STS-ml and SecBPMN2 diagrams. In particular it provides a built-in mechanism to easily change perspective (from now on, jump) between STS-ml ad SecBPMN2 diagrams. We measured the number of times the subjects used the jump mechanism (**# Jumps**) to check if they considered the two modeling language part of the same model.

The mapping of SecBPMN2 task to and-decomposition and the or-decomposition of STS-ml permits business process designers to replicate the goal structure in business processes. In other words, a goal that is decomposed in two or more business processes it corresponds to a business process with call activities that refer to the business processes mapped to the subgoals. For example, consider a goal G that is decomposed in two sub goal G_a and G_b . In this case the replicated hierarchy in business process consists in a business process B which contains, among other elements, two call activities that refer to two business processes B_a and B_b , that are mapped respectively to G_a and G_b .

We checked how many times subjects replicated the hierarchy of goals decompositions of the STS-ml models, in business process (**# Hierarchy**

usage). We used this metric to check if this, not automated, guideline is used by the subjects or if they prefer other organization of the business processes. This metric will confirm if the hierarchy defined in the goal model is functional to the organization of the business processes.

Transformations, jumps and the usage of the hierarchy are meant to integrate STS-ml and SecBPMN2 modeling languages. Therefore, the higher the number of these metrics, the higher the possibility subjects consider the two modeling languages integrated.

Table 8.4 shows quantitative data of the operations on the models. we collected such data by directly asking the subjects, before the experiment, to count the metric we described.

The subjects transformed, on average, 3 (with $\sigma = 1.56$) STS-ml elements in SecBPMN2 business processes and they modified, on average, 9 (with $\sigma = 12.4$) elements of such business processes. These data suggest the subjects created few fairly complex business processes. We believe they did not create more business process because of time constraints. The standard deviation of the number of modifications is high because three subjects did many modifications (more than 20) to the generated SecBPMN2 business processes. They put their effort in customizing and improving the generated models, instead of generating more, less accurate, business processes.

The subjects, during the experiment, jumped between diagrams 6 times on average, with $\sigma = 6.62$. This indicates the subjects used this mechanism frequently to ease the transition between diagrams.

The subjects replicated the hierarchy of goal models in business processes 3 times on average, with $\sigma = 3.31$. This metric shows that subjects applied the guideline almost on all processes, if we compare it with the number of transformations. This demonstrates that the hierarchy of goal model is a valid guideline to organize business processes.

As part of the quantitative analysis we collected data on the usage of the process and the execution of the empirical experiment, shown in table 8.5. We checked the number of steps, of the process provided in SEBE executed

	# Transformations	# Changes	# Jumps	# Hierarchy usage
Mean	3	9	6	3
Std. Dev	1,57	12,4	6,62	3,31

Table 8.4: Data about the operations executed by subjects during the modeling

by the subjects during the experiment (**# Steps**), to check if subjects used the process.

We checked the number of iterations (**# Iterations**), i.e. the number of time the process is executed from the beginning or, in other words, how many times the gateways are evaluated negatively. We used such metric to check if the iterative, incremental aspect of the process was exploited by the subjects.

The **Execution time** indicates the time the subjects used to complete the experiments.

We checked how many times the documentation we provided to the subject was used (**# Documentation usage**) to check how much the modeling languages were understood.

Table 8.5 shows data we collected about the usage of the process the execution time and the usage of the documentation we provided during the empirical evaluation. The first two columns indicate that the subjects executed the process once, on average. While the usage of the documentation indicates, as expected, that the modeling languages were not completely learnt in the training phase. The standard deviation of execution time is high because we fixed a minimum of ten minutes for the last exercise, but we did not put an upper bound.

	# Steps	# Iterations	Execution time	# Documentation usage
Mean	3	1	19 min	3
Std. Dev	0,96	0,83	12,1 min	1,73

Table 8.5: Data about the usage of SEBE

Tables 8.4 and 8.5 show that the subjects used the features of SEBE we wanted to test in the experiment, therefore, the results reflect the real

experience of users.

8.2.6 Qualitative data analysis

After finishing the modeling task (fourth activity of the empirical study), the participants filled in an online questionnaire to evaluate the research questions we identified. Table 8.6 shows the results of each criteria and the overall results, for each research question. In the following we describe how each criteria has been evaluated.

EQ1 For what concerns the help offered by the process, 66% of the subjects confirmed that the process eases the definition of business processes, 33% do not have a strong opinion. On the matter, 59% of the subjects believe the process is useful for enforcing security requirements. These results confirm that the process significantly helped subjects during the experiment (criteria (i)). For what concerns criteria (ii), most of the subjects (65%) will consider using the process to enforce security requirements in other projects, 39% do not have any strong opinion and only 5% will not use the process. We believe this is a clear indication the process is flexible enough to be used in other contexts and with different software engineering process. Results also confirm the process is easy to follow (criteria (iii)): 65% of the subjects consider the process easy to apply, 35% do not have any strong feeling, and no one of the subjects consider the process difficult to apply.

EQ2 The subjects appreciated the automated transformation. In particular, 78% of them agree on the fact that it helps in modeling and in enforcing security requirements (criteria i), 16% do not have any opinion, while 6% disagree. 61% of the subjects believes the automated transformation helps the definition of SecBPMN2 models (criteria ii), 33% do not have strong opinion and only 6% believe it does not help in the definition of SecBPMN2 models. When asked whether the automated transformations help business process designers in case of large scenarios (criteria iii), the answers were even more positive: 66% of the subjects agree, 33% do

not have any opinion, and 0% disagree. This emphasizes that subjects believe in the usefulness of automated transformation, especially in case of real-world scenarios, with wide and complex models. Subjects' comments confirm such conclusion, but they also highlight that it may be difficult to properly use the automated transformation, especially for users who do not have a previous knowledge on STS-ml and SecBPMN2.

EQ3 For what concerns the evaluation of the software tool, 83% of the subjects believe that it well integrates the modeling languages (criteria i), 11% have no strong feeling, while only 6% of the subjects believe that the tool does not do a good job in integrating the different modeling languages. For what concerns the support to the process (criteria ii), 88% of the subjects believe the software tool facilitates the application of the process, 22% do not have any strong opinion, while no one disagreed. Furthermore, 50% of the subjects believe the tool is sufficiently fast, 22% have no opinion and 28% of the subjects believe the software is too slow.

As an overall result, the calculated mean SUS score (criteria iii) of the tool is 64,4 ($n=8$, $\sigma = 9.7$). SUS scores are normalized and range from 0 to 100, with 0 for least usable systems and 100 for best usable systems. On average, systems achieve approximately a SUS score of 68 [110]. Thus, the SUS score of the tool is close to this average, indicating that the tool is neither among the worst systems nor among the best with regard to usability. When the descriptive adjectives of the study of Bangor, Kortum and Miller [13] are applied to the SUS score from this evaluation, the tool can be labeled as between "good" and "ok". Consequently, the current usability status of the tool can be considered "acceptable". However, there is room for improvement with regard to its usability.

Table 8.6 summarizes the outcomes of the quantitative evaluation for each criteria of each research question. We consider all criteria of **EQ1** satisfied: the large majority of the subjects consider the process proposed in this thesis helpful and only a strict minority, at most 5%, is against its adoption. For what concerns **EQ2**, we consider criteria i and criteria iii

satisfied, since the large majority of the subjects believe in the automated transformations are useful for the definition and enforcement of security policies. We obtained similar results for criteria ii, but the percentage of subjects that agreed is lower than the order other criteria (61%) and the subject who disagreed it is higher than most of the other criteria (6%). We marked this criteria with a yellow tick, because it is the less strong result, but, still, is positive. We, therefore, consider EQ2 satisfied. We consider criteria i and criteria ii of **EQ3** satisfied since a large part of subjects consider the STS-Tool a valid help for using SEBE. However, we consider criteria iii not completely satisfied, marked in yellow, because the result of the SUS score is “acceptable”. Still we consider EQ3 satisfied, since all criteria are satisfied.

Research Question	Criteria i	Criteria ii	Criteria iii	Result
EQ1				
EQ2				
EQ3				

Table 8.6: Research questions outcomes

8.2.7 Threats to validity

We report the main threats to the validity of our experiment, using Wohlin’s categorization [126].

Threats to conclusion validity. The relevant threats in this category are the following: (i) low statistical power, the number of subjects is low, since we selected only subjects with experience in modeling social/organizational or business process models; (ii) fishing, we performed the training phase being as neutral as possible, without expressing opinion about SEBE; (iii) reliability of the measure, we used simple and short sentences in the questionnaires, most of the answers consisted in a likert scale; (iv)

reliability of treatment implementation, we provided to the subjects a social/organizational model partially transformed in business process models, this minimized the deviations and focused the attention of the subjects on SEBE, rather on the modeling languages, (v) reliability of irrelevancies in experimental setting, we created a quiet environment as possible.

Threats to internal validity. (i) maturation, the whole experiment lasted less than two hours, to avoid having tired subjects; (ii) instrumentation, we designed the questionnaire to be as simple as possible, we avoided complex sentences and the time requested for the filling-in was less than 15 minutes; (iii) selection, the participation was volunteer-based, therefore subjects might be enthusiastic, we mitigated this threat being as neutral as possible about SEBE.

Threats to construct validity. (i) interaction of testing and treatment, during the experiment we abstained from stressing how much security is central in socio-technical systems, in order to do not alter the perception of subjects about security problems; (ii) hypothesis guessing, during the experiment we suggested the subjects to answer honestly and to avoid to guess the best answer, since there were no wrong answers; (iii) evaluation apprehension, during the experiment we underlined that the subjects were not evaluated and the questionnaires are anonymous.

Threats to external validity. (i) interaction of selection and treatment, the set of subjects was heterogeneous: it included people with a different expertise in security, in goal-based modeling language and business process modeling language; (ii) interaction of setting and treatment, the exercise in the application phase was based on a real case study about the payment system.

8.3 Chapter summary

This chapter described two empirical evaluations we used to check if the SC defined in Chapter 1 are met by SEBE.

In particular, both empirical validations confirmed that SEBE provides

a modeling language (SecBPMN2) which meets SC1.1, SC1.2 and SC2.1. In other words SecBPMN2 is an modeling language that can be used to specify business process with security concepts and security policies, that is easy to use and can be effectively used in real scenarios.

The second experiment permits also to confirm that SC4 and SC6 are met by SEBE. For what concerns SC4, SEBE provides a systematic approach that generates secure business process from social/organizational models and it permits to enforce security requirements. For what concerns SC6, the method provides a process that can be easily followed by the users to use SEBE and it is well supported by STS-Tool, the software tool provided with the method.

Chapter 9

Case studies and application scenarios

This section describes the results of the application of SEBE to three case studies. Differently from the empirical evaluations, these three applications permit to observe how SEBE is applied.

9.1 Case study settings

The case studies are part, as pilots, of VisiOn¹, an european project in which the author of this thesis is involved. The objective of VisiOn consists in increasing the citizens awareness on privacy. The final outcome of the project will be a platform that can be used by public administrations and companies to design their systems, using privacy as a first class requirements, and to monitor, after the deployment, how they satisfy such privacy requirements. SEBE is used in VisiOn as the main method for designing the system of the organization.

The VisiOn platform will be validated with three case studies, consequently, SEBE, as the main design method adopted by the project, is evaluated as well.

For each case study at least two domain experts, who works in the companies analyzed, are involved.

¹<http://www.visioneuproject.eu/>

In order to elicit as more information as possible, we structured the empirical experiment as it follows:

1. SEBE method is presented to the subjects;
2. the subjects write a document with a general description of the case study. We prepare a template to guide the experts in describing the elements needed for the social/organizational ad business process elements;
3. we analyze the description and we create an initial social/organizational model. We create the initial models to let the subjects to focus on the evaluation of SEBE as a whole, instead of evaluating the modeling languages;
4. the subjects, during a workshop of 3 hours, refine the initial social/organizational models and, once complete, they create the secure business process models;
5. the subjects complete the business process models. Conference calls are used to clarify doubts on the method;
6. we collect feedback of subjects on the application of SEBE;
7. the subjects generate part of the implementation and modify it to their exigences;
8. we collect feedback of subjects on the generation of the implementation.

Unfortunately this experiment is not concluded because it is linked with the VisiOn european project which is still running and it is not yet on the implementation phase. Therefore, the last two steps are not executed yet. However we believe the partial results are still valid and can be used for the evaluation of the success criteria.

9.2 Ospedale Pediatrico Bambin Gesu'

Ospedale Pediatrico Bambin Gesu' (OPBG)² is a pediatric Italian hospital with more than 2200 employees, among doctors, nurses and technicians, and 600 beds. It dismisses more than 25000 patients every year and has 7 branches in Italy, the headquarter is located in Rome. OPBG offers web services to patients and their family through its website, and it manages information of patients and its employee with internal information systems. The hospital is connected to other hospitals for tele-consultations, i.e., external doctors may visit the patients hosted in OPBG, and for sharing patients' information with other organizations.

In such complex system secure business processes are extremely important because they guide the execution of most of the activities, from surgical operations, to the negotiation for the data shared between the ethical committees of the hospital and another organizations. Therefore, it is central to design and execute secure business processes that assure that patients, OPBG and external organizations security requirements are correctly enforced.

OPBG is a socio-technical system because it is composed by humans, such as the patients, doctors, nurses, technical services such as the information systems used to manage data of patients and doctors, and other organizations such as other hospitals, the catering company, the cleaning company, etc..

We collaborated with this organization for the creation of a platform that ensures privacy of patients that need a tele-consultation or need to move their data from the hospital to an external, accredited, organization.

Specifically two subjects, a doctor and a technician, used SEBE for engineering secure business processes executed for the management of patients that are not physically present in the hospital. The two subjects followed the process proposed in the method, used the modeling languages and applied the transformation rules by using STS-Tool.

²<http://www.ospedalebambinogesu.it>

9.2.1 Outcomes of the case study

Table 9.1 shows the number of elements defined in the case studies, specifically the second column shows the elements specified by the OPBG subjects to specify the social/organizational model and business process models of the part of OPBG system examined for the case study.

For what concerns the STS-ml models 11 actors and 44 goals were defined. Most of the actors, though, are not associated to any goal but they represent different agents that play the same role. 32 documents were defined and used/exchanged by the actors in the OBG system; they made tangible 10 pieces of information. The subjects identified 68 security requirements at the social/organizational perspective.

Subjects defined 46 business processes, executed by 11 unique participants. They defined a total of 80 activities. The average number of activities per business process is 1,74: some business processes were composed by only one task, because they are out of scope but still their execution is important for other processes. For example, the usage of the localization service, is not part of OPBG but is needed for the tele-consultation. Therefore, its usage is represented with a SecBPMN2 call activity, which is linked to a business process with one task that represents the execution of the localization service. This permits to consider external services, the external actors that execute the services and it permits to specify security constraints, in terms of annotations, on the called external service. The subjects specified 40 security annotations.

This case study underlined how essential the connection between social/organization and business process models is. The specification of the organizational elements helped the subjects to organize their knowledge and, therefore to be more effective in the specification of the business process. Indeed, during the workshop and during the conference calls, the subjects discovered incoherences in the social/organizational model and in the business processes that would not have been discovered with a separated analysis of the models.

On the other hand the subjects in their feedback underlined the fact that SEBE modeling languages are complex to master. Only after the workshop and some hours spent using the modeling languages the experts had been able to autonomously use SEBE.

9.3 Ministero dello Sviluppo Economico

Ministero Italiano Sviluppo Economico (MISE)³ is an Italian ministry in charge of helping and guiding the economic development of Italy. One of the main activities of MISE consists in helping the development of Italian companies, therefore, it is permanently in contact with many Italian firms.

Due to an Italian law on transparency, MISE is obliged to publish details about its employees and about the organizations it deals with. Business processes play a central role because they are used to check and publish data of organizations and employees. Therefore, it is extremely important to design and execute secure business processes which enforce security requirements of citizens and organizations.

MISE is a socio-technical system because it is composed by human actors such as the employees, technical services such as the information systems used to manage organizations data or publish data to citizens, and other organization such as the organizations helped by the ministry.

We collaborated with MISE for creating a platform which ensures the enforcement of privacy requirements of organizations and MISE for the publication of data to fulfill Italian transparency law for ministries. In particular, four subjects used SEBE for engineering business processes. They executed the process provided by the method and the collaborated to define the social/organizational model and the business process models.

³<http://www.mise.gov.it>

9.3.1 Outcomes of the case study

The third column of Table 9.1 shows the elements used by MISE subjects. For what concerns the social/organizational model, the subjects defined 23 actor and 94 goals. They also specified 43 documents and 36 information. The resulting STS-ml diagram is complex and do not focus only on the part of the MISE system used for the evaluation. They specified 131 security requirements.

For what concerns the business process perspective they specified 12 business processes, executed by 3 participants. The number of business processes is considerably lower, respect of the dimension of the STS-model, if we compare the OPBG case study, because the subjects used only a part of the social/organization model they specified to generate the business processes. This shows a slightly different usage of SEBE: they exploited the social/organizational model not only for gathering information for the generation of the business process but for eliciting contextual information. The subjects specified 3 security annotations. This is due to the fact that many of the security requirements relevant for the business processes generated from the social/organizational model were about authorization constrains and that were enforced with the proper access to data objects.

The subjects confirmed that the SEBE process was easy to execute and that the transformation rule were easy to apply and that were extremely useful to generate the business processes and check if the security requirements were satisfied by the business processes. In particular the latter were essential to check if only non sensitive information are published. In a latter stage of the case study, i.e., after the workshop, the domain expert underlined the possibility of customizing the transformation rules was essential to generate ad-hoc security policies.

9.4 Athens municipality

Athens municipality⁴ is the main public administration of the city of Athens. It provides numerous services to the citizens, such as birth registration, road misconstructions and fees payments. Most of those services are automated and offered through municipality websites. Dimos Athinaion Epicheirisi Michanografisis (DAEM)⁵ was the first greek company to manage services for Athens municipality and, currently, is the largest company in the city which provides the municipality the infrastructure and manages the services for the citizens of Athens. Security is central for DAEM, since most of services provided to citizens require the management of sensitive data and are critical for the citizens and the municipality.

Business processes are central in this system because they are used to manage data and to specify the steps, most of which are imposed by law, to provide the Athens services.

Athens municipality is a socio-technical system because is composed of: (i) humans such as citizens and employees of the municipality; (ii) autonomous technical component such as services of other public administration; and organizations such as private company that collaborate, or interact with the municipality.

For this case study we did not collaborated directly with Athens municipality, instead, we collaborated with DAEM to create a socio-technical system to manage data of citizens and organizations in order to provide services as birth registration, road misconstruction report and fees payment.

In particular two subjects, one expert of the PA and one DAEM technician, used SEBE. They executed the process provided with the method, specifying the social/organizational model generating and enriching the business process models.

⁴<https://www.cityofathens.gr>

⁵<http://www.daem.gr>

9.4.1 Outcomes of the case study

The fourth column of Table 9.1 shows the elements used by DAEM subjects to define secure business processes. They specified 8 actors and 23 goals and, for the data part, they specified 11 documents and 26 information. The social/organizational model defined by DAEM experts is smaller of the models created by MISE and OPBG experts. The number of security requirements, 15, are lower too. This does not mean the case study is smaller, instead the experts create the social/organizational model at a higher level of abstraction. It is a choice that was due to multiple factors such as the effort available and the information available for the analyzed part of the system.

For what concerns the procedures, 40 business processes were specified, executed by 8 participants. A total of 63 activities were specified. SEBE leded to the creation of small business processes, that are focused on the achievement of the goals specified in the social/organizational model. In this case study DAEM experts preferred to focus their effort in the creation of more business processes, instead of specifying in social/organizational models, as preferred by MISE subjects. This choice did not influence the final outcome, since in both cases the subjects specified the secure business processes needed for the achievement of their core goals.

	OPBG	MISE	Athens municipality
Actors	11	23	8
Goals	44	94	23
Documents	32	43	11
Information	10	36	26
Security requirements	68	131	15
Business processes	46	12	40
Participants	11	3	8
activities	80	15	63
Security annotations	40	3	0

Table 9.1: Elements of STS-ml and SecBPMN2 used to specified the case studies

Subjects appreciated the intuitive generation of business processes from

social/organizational model and the easiness of generating security policies from security requirements and the verification of the latter against business processes. On the other hand, they underlined the initial difficulties in learning the STS-ml and SecBPMN2 modeling languages, because of the richness of their syntaxes.

9.5 Comparison of case study results

The three case studies analyzed in this chapter highlighted different usages of SEBE. MISE focused on the social/organizational model while OPBG focused on the design of business processes. We believe this difference was due to the different nature of the case studies and in the amount of information available during the modeling phases. Table 9.2 shows the distinctive characteristics of the case studies reported in this chapter. OPBG is a private company and, therefore, this may have influenced the focus of the subjects. On the other hand MISE is public administrations with, probably, less control on the business process are executed.

Subjects of Athens municipality applied a different approach since they used a general social/organizational model to specify the business processes. The conceptual gap between the social/organizational model and the business process model in this case was marked, because business processes were associated to very generic goals, but this did not prevent the application of the transformation rules.

In all three case studies it was possible to define business processes that satisfied all the social/organizational security requirements. This was permitted by the incremental process provided by SEBE, which allowed subjects to modify the the social/organizational and business process models in order to create secure business processes, i.e., business process models that satisfies social/organizational security requirements.

In the MISE case study the security requirements, elicited from the part of the social/organizational model was not used for the generation of the business processes, were ignored. Indeed, STS-Tool automatically ignores

	Type of organizations	Type of actors	Type of information
OPBG	Private	Patients, services, organizations	ID info, health info, organization info.
MISE	Public administration	Citizens, services, organizations	ID info, Taxation info, organization info
Athens municipality	Public administration	Citizens, services, organizations	ID info, birth info, building site info

Table 9.2: Case studies comparison

security requirements that refer to elements of STS-ml but that are not mapped to any SecBPMN2 elements.

9.6 Chapter Summary

This chapter describes the application of SEBE to three case studies: two public administrations and a company, which used SEBE to engineer secure business processes in complex, real scenarios, using a socio-technical approach. In particular the case studies permit to verify that SC3, SC1.1, SC1.2 and SC2.1.

For what concerns SC3 they confirmed that SEBE can be used to verify procedural patterns against SecBPMN2 business processes (SC3.1), verify security concepts against SecBPMN2 business processes (SC3.2), and that STS-Tool offers a valid support for the analysis (SC3.3).

These case studies also confirmed that SecBPMN2 is an effective modeling language for business processes with security concepts (SC1.1 and SC1.2) and for security policies (SC2.1), and that the transformation rules provided for generating business processes and enforcing security policies can be effectively used by business process designers in real scenarios. These SC were already demonstrated in Chapter 8, their validation has

been crosschecked in this chapter.

As specified before, VisiOn, and consequently the evaluation using the case studies, is not yet concluded. Therefore, this chapter describes only partial results of the applications of SEBE. With the partial results we cannot evaluate how subjects perceive the usefulness of SC5, which requires a method that provides transformation rules that can be effectively used by business process designers to generate part of the implementation (SC5.1) and that these transformation rules are supported by the extended version of STS-Tool (SC5.2). However, from preliminary results and discussions with security and domain experts we believe that this part of the method will be more than welcomed, since it support business process designers in the last phase of the engineering of secure business processes, i.e, their implementation.

Chapter 10

Conclusions and future work

This thesis proposes SEBE: a method for engineering secure business processes that considers socio-technical aspects of organizations.

Table 10.1 shows a map of the thesis, with the Research Questions (RQs), the corresponding Success Criteria (SC), the chapters in which are described the parts of SEBE that meets the SC, and the methods/contributions used to evaluate the SC, namely, empirical evaluations, the formal definition of the abstract syntax and applications to case studies.

The rest of the chapter describes how SEBE meets the SC, conclusions and limitations of this thesis, ongoing research works and future lines of research.

10.1 Fulfillment of success criteria

10.1.1 Success Criteria 1

SC1 requires a modeling approach that can be used for: representing business processes (SC1.1), representing security concepts (SC1.2) and for formalizing business processes with security concepts (SC1.3).

To achieve these requests, SEBE provides SecBPMN2: a modeling language we designed for modeling secure business processes and security policies. SecBPMN2 is composed of two parts: SecBPMN2-ml and SecBPMN2-Q. In particular SecBPMN2-ml extends BPMN 2.0 with a rich set of secu-

RQ	SC	Chapters	Empirical eval.	Formal syntax	Case studies
RQ1	SC 1.1	5	✓		✓
	SC 1.2		✓		✓
	SC 1.3			✓	
RQ2	SC 2.1	5	✓		✓
	SC 2.2			✓	
RQ3	SC 3.1	5			✓
	SC 3.2				✓
	SC 3.3	7			✓
RQ4	SC 4.1	6	✓		✓
	SC 4.2	5/6	✓		✓
RQ5	SC 5.1	6			?
	SC 5.2	7			?
RQ6	SC 6.1	4	✓		
	SC 6.2	7	✓		

Table 10.1: Thesis map

rity concepts. It meets SC1.1, since it permits to specify business processes, while it achieves SC1.2 because it allows to specify security concepts and to associate them to business process elements..

SecBPMN2-ml is provided with a formal specification of its abstract syntax that meets SC1.3, since it permits to avoid incoherences and ambiguities and it allows business process designers to specify business processes diagrams with a formally specified abstract syntax.

The fulfillment of SC1.1 and SC1.2 is verified with empirical evaluations and with applications of SEBE to case studies. In particular the empirical evaluation described in Section 8.1 confirms the fulfillment of SC1.1, since the subjects underlined the the usefulness of the modeling language and they preferred it instead to other, less expressive and/or more complex, modeling languages. The applications to case studies, presented in Chapter 9, confirm that SecBPMN2 is an effective language for modeling secure business processes. The formalization of SecBPMN2-ml satisfies SC1.3.

10.1.2 Success Criteria 2

SC2 requires a modeling approach that permits to specify (SC2.1) and to formalize (SC2.2) security policies.

SEBE provides SecBPMN2-Q, a graphical query language, with an abstract syntax formally specified, that extends SecBPMN2-ml and allows to specify security policies. The language permits to define patterns, i.e. constraints on the sequence of activities of the business processes that must or must not be executed. It also permits to specify constraints on the security concepts.

SecBPMN2-Q fulfills SC2.1, since it permits to specify security policies, while its formal abstract syntax satisfies SC2.1.

Empirical evaluations, described in Chapter 8, and applications of SEBE to case studies, presented in Chapter 9, confirmed the utility of SecBPMN2-Q in defining security policies. In particular, in the empirical experiment described in Section 8.1 subjects preferred SecBPMN2-Q instead of other languages for defining security policies. While, for the the applications of the method to case studies, subjects confirmed the utility of SecBPMN2-Q and the easiness of usage with large, real scenarios.

10.1.3 Success Criteria 3

SC3 requires an automated approach (SC3.3) that permits to verify procedural patterns (SC3.1) and security constraints (SC3.2) against business processes.

SC3.1 and SC3.2 are fulfilled respectively by the formalization of SecBPMN2-ml and SecBPMN2-Q, and by the formal specification of how the security policies are satisfied by secure business processes, described in Chapter 5. SC3.3 is achieved thanks to the implementation of the verification of security policies in STS-Tool.

The fulfillment of SC3 is verified with the applications of SEBE to case studies. In particular, the feedback from the subjects who applied the method confirmed that SC3.1 and SC3.2 are achieved since the verification

outputs matched the expectations of the subjects. While SC3.3 is achieved because the subjects, during the experiments, used the implementation of the verification numerous times and they underlined, with positive feedbacks, the usefulness of the functionality.

10.1.4 Success Criteria 4

SC4 asks for a systematic approach that supports the derivation of business processes from a social/organizational models (SC4.1) and the verification of the enforcement of social/organizational security requirements against secure business processes (SC4.2).

SC4.1 is fulfilled with the transformation rules described in Chapter 6 while SC4.2 is achieved thanks to the same transformation rules and the verification rules described in Chapter 5. In particular, the transformation rules permits to transform security requirements in security policies, while the verification rules permit to verify if security policies are satisfied against business processes.

The fulfillment of SC4 is checked with the empirical evaluations and the applications of SEBE to case studies. The results of empirical evaluations described in Section 8.2 confirm that the users of SEBE find the generation of business process and the verification of security policies relevant and useful functionalities for engineering secure business processes. These results are confirmed by the feedback of the subjects who applied SEBE to case studies.

10.1.5 Success Criteria 5

SC5 requires a systematic approach that permits to generate part of the implementation of secure business processes (SC5.1) in an automated fashion (SC5.2).

SC5.1 is fulfilled by the transformation rules provided in Section 6.2, since they specify how to generate RDL code from SecBPMN2 business

processes, while SC5.2 is achieved thanks to the implementation of such rules in STS-Tool.

Our internal tests confirm that the generation algorithm and implementation work as expected and, therefore, the fulfillment of SC5.1. The verification of the utility of the implementation is an objective of the evaluation using the application of SEBE to case studies. Unfortunately the evaluation is not finished because it is part of a two year european project, VisiOn¹, which has not yet reached the development phase and, therefore, it is not possible to fully evaluate SC5.2. However, from preliminary discussions with the subjects, we believe that this feature is more than welcomed and it will be appreciated.

10.1.6 Success Criteria 6

SC6 asks for a process that guides business process designers, during the engineering of secure business processes (SC6.1), that is supported by a software tool (SC6.2).

SC6.1 is achieved thanks to the process described in Chapter 4 while SC6.2 is fulfilled with the extension of STS-Tool we provided with SEBE.

The fulfillment of SC6 is verified with the empirical evaluation described in Section 8.2, which confirms that users of SEBE find the process easy to follow and helpful, especially when they use the method for the first times. The same evaluation confirmed that STS-Tool provided a valid support for the process, allowing a natural execution of all its phases.

10.2 Conclusion and limitations

Dealing with security is an extremely important activity for today's organizations. Security breaches impact on the activities and on the business processes of such organizations causing millions of dollars of losses, as described in PWC reports of last years [1, 2].

¹www.visioneuproject.eu

The same documents report that, nowadays, more and more security breaches occurs because of insiders, which uses the organizational structure and social aspects to harm organizations. Therefore, the design of business processes should enforce both social and organizational security aspects.

Still the mere design of business processes does not guarantee to avoid security issues. Business processes have to be implemented and executed correctly. Their implementation should employ security controls that enforce security aspects defined in secure business processes.

This thesis proposes SEBE a method for engineering secure business processes using a socio-technical approach, namely, it supports business process designers in designing secure business processes using social/organizational security aspects, and in generating part of the implementation that enforces security aspects defined in secure business processes.

In the following we summarize the contribution made by this thesis.

A process for secure business process engineering. SEBE is provided with a process which supports business process engineers in designing secure business processes and generating implementation from such business processes. The process can be described in three macro steps: (i) the definition of a social/organizational models that define social/organizational security requirements; (ii) the definition of secure business processes that enforce social/organizational security requirements; (iii) the generation of the implementation that follows what is specified in the business processes.

The process is iterative and incremental and it permits a continuous refinement of the social/organizational models the business process models and the implementation.

SecBPMN2 modeling language. SecBPMN2 is an expressive, graphical modeling language that extends BPMN 2.0 with a rich set of security annotations and a set of relations that permits to specify secure business processes and security policies. The formalization of the abstract syntax of the language resolves many ambiguities of BPMN 2.0 and specifies how security policies are satisfied in secure business processes.

Transformation rules. SEBE is provided with a set of transformation rules which integrate social/organizational models with business process models with the implementations. The method provides a meta-language to change such transformation rules in order to customize them for specific contexts.

STS-Tool. SEBE is fully supported by STS-Tool, a software which helps business process designers in applying the method. The software supports the execution of the process in all its steps, with functionalities that implements the transformation rules, the verifications rules and that integrate the modeling languages.

Evaluation of SEBE. We evaluate SEBE with two empirical evaluations and with applications two three case studies. The results of the evaluations confirms that the method achieved all the success criteria.

10.2.1 Limitations

During the empirical evaluations and the applications of SEBE to case studies, we observed some limitations of the method.

Users underlined that the STS-ml and SecBPMN2 are expressive modeling languages but with a complex syntax that is not easy to learn and it takes hours to be mastered. STS-ml uses concepts new to many business process designers and, consequently, they have to learn how to model the social/organizational aspects of organization. SecBPMN2, on the other hands, is based on the intuitive idea of processes, but it builds on BPMN 2.0 which is an expressive and complex language with many concepts and relations to be learned and understood.

Some users, highlighted the lack of security concepts in SecBPMN2, for example anonymity and undetectability. Even if the set of security concepts in SecBPMN2 is rich and derivates from a reference model on security, in some specific context peculiar security concepts, that are not considered by SecBPMN2, may be needed.

Another limitation of the method regards the implementation language

we chose, RDL, for the generation of the implementation of secure business processes. We chose RDL because is a new programming language for business artifacts and their business logic that is built on top of a well-known and vastly used framework: HANA². Still, the choice of an implementation language inevitably reduces the scope of the applications of SEBE, however we defined the generation of RDL applications using templates that can be modified to adopt a similar language such as Advanced Business Application Programming (ABAP)³ or Oracle PeopleCode.

10.3 Ongoing works

While SEBE is a mature method, we still work to eliminate or reduce its limitations.

Evaluation of the applications to case studies. The applications to case studies will permit to check if users in real scenarios find the generation of the implementation useful to their purposes. Our preliminary evaluations and discussions with security experts confirm the usefulness of this feature of SEBE, we believe that the generated RDL applications will be of a great help to developers.

Security concepts in SecBPMN2. One of the limitations of SEBE regards the limited set of security concepts that can be used in SecBPMN2 diagrams. Even if the set is reach and it comprehends the most known security concepts, it will never have a full coverage. We, therefore, are working on another direction: instead of enriching the set, we will permit the users to define their security concepts and add them to the set.

Implementation consistency check. Currently SEBE has no control on the implementation, once it is generated. Therefore, developers may inject malicious code or remove security controls to create security vulnerabilities. To minimize this possibility we are extending SEBE in order to allow business process designers to verify the implementation code, after

²<https://hana.sap.com>

³<http://scn.sap.com/community/abap>

it is modified by the developers. This verification will check if the implementation code modified by developers enforces the security requirements that were enforced by the original implementation code. This check is composed by two parts. One part verifies if the security controls were modified, if it is true we will not guarantee the enforcement of the security requirements. The second part will allow to propagate security constraints through business artifacts data, i.e., business artifacts will transmit their security constraints to all the business artifacts that store their information. The second part of the consistency check will follow the information flow and it will check if a business artifacts declared by the developers, which contains protected information, is protected by the security control.

Improvement of STS-Tool. STS-Tool is a stable, complete software tool already released for the public. However, we plan to modify the graphical appearance of the tool to further facilitate the usage of SEBE. Moreover, we will extend the software to support the verification of the code as describe in the previous point.

10.4 Future research work

This thesis opens a number research lines, that are described in the rest of the section.

Beyond security. SEBE is a method that offers support for security during business process engineering. We focused on security because is an extremely relevant aspect for organizations, but it is not the only one. Other aspects can heavily influence an organization and they are, therefore, central to consider when developing business processes. For example, privacy, information quality and the next release problem [12]. These aspects cross the technological or social/organizational boundaries and, therefore, have to be analyzed using a SEBE-based approach.

Adaptive security. Another research line consists in using the method to create agent-based systems with adaptive security. Nowadays security is static, once security controls are implemented in a system, they are

rarely substituted/upgraded if not necessary because of security breaches. Adaptive security systems consist in autonomous components (agents) that react immediately to security breaches deploying the best security solution. SEBE can be used for designing and implementing these systems, since it permits to specify the social/organizational security goals, the plans executed by the agents that enforce the security requirements (i.e., the business processes) and then the implementation of such plans.

Open data. Open data is the idea that some data should be freely available to everyone to use and republish as they wish, without restrictions from copyright, patents or other mechanisms of control [8]. In this context the european commission requests companies and public administrations to be more and more transparent allowing users and citizens to access the data of the organizations⁴.

Still, sensitive data cannot be published, therefore, organizations' data have to execute processes to sanitize them. But such sanitization depends on the type of information and the requirements of the customers. A method based on SEBE can be used to define the privacy requirements and what are the sensitive information, define the business process used to sanitize data and check if they are applied in business process of organizations, and generate if their implementation that safely published sanitized data.

Business process re-engineering. SEBE opens the possibility of re-engineer secure business processes for complex socio-technical systems. The transformation and verification rules can be extended to derive secure business processes from the implementation of socio-technical systems of organizations, check if they are compliant with socio/organizational security requirements, modify them and re-implement them.

⁴<https://ec.europa.eu/digital-single-market/en/open-data>

Bibliography

- [1] Managing cyber risks in an interconnected world: Key findings from the global state of information security survey 2015. Technical report, PWC, 2014.
- [2] Turnaround and transformation in cybersecurity. Technical report, PWC, 2015.
- [3] A. I. Anton. Goal-based requirements analysis. *Proc. of International Conference on Requirements Engineering*, pages 136–144, 1996.
- [4] N. Argyropoulos, L. M. Alcañiz, H. Mouratidis, A. Fish, D. G. Rosado, I. G.-R. de Guzmán, and E. Fernández-Medina. Eliciting security requirements for business processes of legacy systems. In *The Practice of Enterprise Modeling*, pages 91–107. Springer, 2015.
- [5] N. Argyropoulos, H. Mouratidis, and A. Fish. Towards the derivation of secure business process designs. In *Advances in Conceptual Modeling*, pages 248–258. Springer, 2015.
- [6] V. Atluri and W. Huang. An Extended Petri Net Model for Supporting Workflows in a Multilevel Secure Environment. In *Database Security X*, pages 199–216, 1996.
- [7] V. Atluri and J. Warner. Security for workflow systems. In *Handbook of Database Security*, pages 213–230. Springer, 2008.
- [8] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. *Dbpedia: A nucleus for a web of open data*. Springer, 2007.

- [9] A. Awad. BPMN-Q: A language to query business processes. In *Proc.of Enterprise Modelling and Information Systems Architectures*, volume P-119, pages 115–128, 2007.
- [10] A. Awad. *A compliance management framework for business process models*. PhD thesis, 2010.
- [11] M. Backes, B. Pfitzmann, and M. Waidner. Security in business process engineering. In *Business Process Management*, pages 168–183. Springer, 2003.
- [12] A. J. Bagnall, V. J. Rayward-Smith, and I. M. Whittley. The next release problem. *Information and software technology*, 43(14):883–890, 2001.
- [13] A. Bangor, P. Kortum, and J. Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *The Journal of Usability Studies*, 4(3):114–123, 2009.
- [14] V. R. Basili, G. Caldiera, and D. H. Rombach. *The Goal Question Metric Approach*. John Wiley & Sons, 1994.
- [15] C. Beerli, A. Eyal, S. Kamenkovich, and T. Milo. Querying business processes with BP-QL. *Information Systems*, 33(6):477–507, 2008.
- [16] S. Beheshti, B. Benatallah, H. Motahari-Nezhad, and S. Sakr. A Query Language for Analyzing Business Processes Execution. *Business Process Management*, 6896:281–297, 2011.
- [17] M. Bishop. What Is Computer Security? *IEEE Security & Privacy*, 1(1):67–69, 2003.
- [18] X. Blanc, A. Mougnot, I. Mounier, and T. Mens. Incremental Detection of Model Inconsistencies Based on Model Operations. *Proc. of Conference of Advance Information Systems Engineering (CAiSE)*, pages 32–46, 2009.

- [19] S. J. Bleistein, K. Cox, and J. Verner. Requirements engineering for e-business systems: integrating jackson problem diagrams with goal modeling and bpm. *Proc. of Software Engineering Conference*, pages 410–417, 2004.
- [20] G. Booch, J. Rumbaugh, and I. Jacobson. *Unified Modeling Language User Guide, The (2nd Edition)*. Addison-Wesley Professional, 2005.
- [21] M. Brandozzi and D. E. Perry. Transforming goal-oriented requirement specifications into architecture prescriptions. In *Proc. of workshop From Software Requirements to Architecture*, pages 54–61, 2001.
- [22] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. TROPOS: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.
- [23] A. D. Brucker and I. Hang. Secure and compliant implementation of business process-driven systems. *Proc. of International Conference on Business Process Management*, pages 662–674, 2012.
- [24] A. D. Brucker, I. Hang, G. Lückemeyer, and R. Ruparel. SecureBPMN: Modeling and Enforcing Access Control Requirements in Business Processes. In *Proc. of Symposium on Access control Models and Technologies*, 2012.
- [25] J. S. Cheney. Heartland payment systems: lessons learned from a data breach. *Federal Reserve Bank of Philadelphia-Payment Cards Center Discussion Paper*, (10-1), 2010.
- [26] Y. Cherdantseva and J. Hilton. A Reference Model of Information Assurance and Security. In *Proc. of International Conference on Availability, Reliability and Security*, pages 546–555, 2013.
- [27] W. Clocksin and C. Mellish. *Programming in PROLOG*. Springer Science & Business Media, 2003.

- [28] D. Cohn and R. Hull. Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Engineering Bulletin*, 32(3):3–9, 2009.
- [29] L. Compagna, P. Guilleminot, and A. D. Brucker. In *Proc. of IEEE International Conference on Software Testing*, 2013.
- [30] F. Dalpiaz, P. Giorgini, and J. Mylopoulos. Adaptive Socio-Technical Systems: a Requirements-driven Approach. *Requirements Engineering*, 18(1):1–24, 2013.
- [31] F. Dalpiaz, E. Paja, and P. Giorgini. Security Requirements Engineering via Commitments. In *Proc. of workshop on Socio-Technical Aspects in Security and Trust*, 2011.
- [32] J. Damasceno, F. Lins, R. Medeiros, B. Silva, A. Souza, D. Aragão, P. Maciel, N. Rosa, B. Stephenson, and J. Li. Modeling and executing business processes with annotated security requirements in the cloud. *Proc. of IEEE International Conference on Web Services*, pages 137–144, 2011.
- [33] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed Requirements Acquisition. *Science of Computer Programming*, 20:3–50, 1993.
- [34] P. Delfmann, H. Dietrich, J. Havel, and M. Steinhorst. A Language-independent Model Query Tool. In *Proc. of international conference on Design Science Research in Information Systems and Technology.*, pages 453–457, 2014.
- [35] D. Deutch and T. Milo. Querying Structural and Behavioral Properties of Business Processes. In *Proc. of international conference on Database Programming Languages*, pages 169–185. 2007.

- [36] P. T. Devanbu and S. Stubblebine. Software engineering for security: a roadmap. In *Proc. of conference on the Future of Software Engineering*, pages 227–239. ACM, 2000.
- [37] E. Dubois and H. Mouratidis. Guest editorial: security requirements engineering: past, present and future. *Requirements Engineering*, 15(1):1–5, 2010.
- [38] M. Dumas and A. H. M. Hofstede. UML Activity Diagrams As a Workflow Specification Language. In *Proceedings of UML*, pages 76–90, 2001.
- [39] T. Eiter, W. Faber, N. Leone, G. Pfeifer, and A. Polleres. Planning under incomplete knowledge. In *Proc. of Computational Logic*, volume 1861, pages 807–821. 2000.
- [40] G. Elahi and E. Yu. A goal oriented approach for modeling and analyzing security trade-offs. *Proc. of Conceptual Modeling*, pages 375–390, 2007.
- [41] E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computer and System Sciences*, 30(1):1–24, 1985.
- [42] Federal Aviation Administration. SWIM ATM case study, last visited April 2016. http://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techops/atc_comms_services/swim/, 2014.
- [43] D. Ferraiolo, J. Cugini, and D. Richard Kuhn. Role-Based Access Control (RBAC): Features and Motivations. 1995.
- [44] D. Ferraiolo and R. Kuhn. Role-based access control. In *Proc. of National Computer Security Conference*, pages 554–563, 1992.
- [45] D. Firesmith. Specifying Reusable Security Requirements. *Journal Of Object Technology*, 3(1):61–75, 2004.

- [46] C. Francescomarino, C. Ghidini, S. Tessaris, and I. V. Sandoval. Completing workflow traces using action languages. *Proc. of International Conference on Advanced Information Systems Engineering*, pages 314–330, 2015.
- [47] A. Ghose and G. Koliadis. Auditing Business Process Compliance. *Proc. of International Conference of Service-Oriented Computing*, pages 169–180, 2007.
- [48] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone. Modeling Security Requirements through Ownership, Permission and Delegation. *Proc. of international conference on Requirement Engineering*, pages 167–176, 2005.
- [49] V. Gruhn and R. Laue. A heuristic method for detecting problems in business process models. *Business Process Management Journal*, pages 806–821, 2010.
- [50] P. Herrmann and G. Herrmann. Security requirement analysis of business processes. *Electronic Commerce Research*, 6(3-4):305–335, 2006.
- [51] A. Hofstede, C. Ouyang, M. La Rosa, L. Song, J. Wang, and A. Polyvyanyy. APQL: A Process-Model Query Language. In *Proc. of AP-BPM*, volume 159, pages 23–38, 2013.
- [52] J. Horkoff, T. Li, F.-L. Li, M. Salnitri, E. Cardoso, P. Giorgini, and J. Mylopoulos. Using goal models downstream: A systematic roadmap and literature review. *International Journal of Information System Modeling and Design (IJISMD)*, 6(2):1–42, 2015.
- [53] J. Horkoff, T. Li, F.-L. Li, M. Salnitri, E. Cardoso, P. Giorgini, J. Mylopoulos, and J. Pimentel. Taking goal models downstream: a systematic roadmap. pages 1–12, 2014.

- [54] ISACA. An Introduction to the Business Model for Information Security. Technical report, 2009. <http://www.isaca.org/Knowledge-Center/Research/ResearchDeliverables/Pages/An-Introduction-to-the-Business-Model-for-Information-Security.aspx>.
- [55] M. Jackson. *Problem frames: analysing and structuring software development problems*. Addison-Wesley, 2001.
- [56] A. Josang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618 – 644, 2007.
- [57] J. Jurjens. UMLsec: Extending UML for Secure Systems Development. In *Proc. of UML*, pages 412–425, 2002.
- [58] H.-K. Kim, R. Y. Lee, and H.-S. Yang. Frameworks for secured business process management systems. pages 57–65, 2006.
- [59] K. Knorr and S. Röhrig. Security requirements of e-business processes. In *Towards the E-Society*, pages 72–86. 2002.
- [60] F. Kossak, C. Illibauer, V. Geist, J. Kubovy, C. Natschläger, T. Ziebermayr, T. Kopetzky, B. Freudenthaler, and K.-D. Schewe. A rigorous semantics for bpmn 2.0 process diagrams. In *A Rigorous Semantics for BPMN 2.0 Process Diagrams*, pages 29–152. 2014.
- [61] M. Lankhorst. *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer, 2009.
- [62] A. Lapouchnian, Y. Yu, and J. Mylopoulos. Requirements-driven design and configuration management of business processes. In *Business Process Management*, pages 246–261. Springer, 2007.
- [63] M. Leitner, M. Miller, and S. Rinderle-Ma. An Analysis and Evaluation of Security Aspects in the Business Process Model and Notation. *Proc. of International Conference on Availability, Reliability and Security*, pages 262–267, 2013.

- [64] M. Leitner and S. Rinderle-Ma. A Systematic Review on Security in Process-Aware Information Systems- Constitution, Challenges, and Future Directions. *Information Software Technology*, 56(3):273–293, 2014.
- [65] M. Leitner, S. Schefer-Wenzl, S. Rinderle-Ma, and M. Strembeck. An Experimental Study on the Design and Modeling of Security Concepts in Business Processes. *Proc. of The Practice of Enterprise Modeling*, pages 236–250, 2013.
- [66] J. Li, J. Mirkovic, M. Wang, P. Reiher, and L. Zhang. SAVE: Source address validity enforcement protocol. In *Proc. of IEEE International Conference on Computer Communications*, volume 3, pages 1557–1566, 2002.
- [67] N. Li, M. V. Tripunitara, and Z. Bizri. On mutually exclusive roles and separation-of-duty. *ACM Transactions on Information and System Security*, 10(2):5, 2007.
- [68] Y. Liu, S. Muller, and K. Xu. A static compliance-checking framework for business process models. *IBM System Journal*, 46(2):335–361, 2007.
- [69] N. Lohmann. Compliance by design for artifact-centric business processes. *Information Systems*, 38(4):606–618, 2013.
- [70] N. Lohmann and M. Nyolt. Artifact-centric modeling using BPMN. In G. Pallis, M. Jmaiel, A. Charfi, S. Graupner, Y. Karabulut, S. Guinea, F. Rosenberg, Q. Z. Sheng, C. Pautasso, and S. B. Mokhtar, editors, *International Conference on Service-Oriented Computing Workshops*, LNCS 7221, pages 54–65. Springer, 2011.
- [71] H. A. López, F. Massacci, and N. Zannone. Goal-equivalent secure business process re-engineering.

- [72] C. L. Maines, D. Llewellyn-Jones, S. Tang, and B. Zhou. A cyber security ontology for bpmn-security extensions. In *Proc. of International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/I-UCC/DASC/PICOM)*, pages 1756–1763. IEEE, 2015.
- [73] M. Mason. Sample size and saturation in PhD studies using qualitative interviews. *Forum: Qualitative Social Research*, 11(3), 2010.
- [74] J. McCumber. Information Systems Security: A Comprehensive Model. *Proc. of National Computer Security Conference*, 1991.
- [75] J. McDermott and C. Fox. Using Abuse Case Models for Security Requirements Analysis. In *Proc. of Computer Security Applications Conference*, pages 55–64. IEEE, 1999.
- [76] M. Menzel, I. Thomas, and C. Meinel. Security Requirements Specification in Service-Oriented Business Process Management. In *Proc. of International Conference on Availability, Reliability and Security*, pages 41–48, 2009.
- [77] K. Mickelberg, N. Pollard, and L. Schive. Us cybercrime: Rising risks, reduced readiness key findings from the 2014 us state of cybercrime survey. *US Secret Service, National Threat Assessment Center. Pricewaterhousecoopers*, 2014.
- [78] D. Moody. The Physics of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. *IEEE Transaction on Software Engineering*, 35:756–779, 2009.
- [79] H. Mouratidis and P. Giorgini. Secure Tropos: A Security-Oriented Extension of the Tropos methodology. *International Journal of Software Engineering and Knowledge Engineering*, pages 285–309, 2007.

- [80] H. Mouratidis and J. Jurjens. From goal-driven security requirements engineering to secure design. *International Journal of Intelligent Systems*, 25(8):813–840, 2010.
- [81] T. Neubauer, M. Klemen, and S. Biffl. Secure business process management: a roadmap. *Proc. of International Conference on Availability, Reliability and Security*, 2006.
- [82] A. Nigam and N. S. Caswell. Business artifacts: an approach to operational specification. *IBM Systems Journal*, 42(3):428–445, 2003.
- [83] OASIS. Web Services Business Process Execution Language. <http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html>, Apr 2007.
- [84] OASIS. eXtensible Access Control Markup Language (XACML) Version 3.0. <http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html>, Jan 2013.
- [85] OMG. OMG Unified Modeling Language (OMG UML), Infrastructure, V2.1.2. Technical report, 2007.
- [86] OMG. BPMN 1.1, Jan 2008.
- [87] OMG. BPMN 2.0, Jan 2011.
- [88] C. Ouyang, M. Dumas, W. M. Van Der Aalst, A. H. Ter Hofstede, and J. Mendling. From business process models to process-oriented software systems. *ACM transactions on software engineering and methodology*, 19(1):2, 2009.
- [89] E. Paja, F. Dalpiaz, and P. Giorgini. Managing Security Requirements Conflicts in Socio-Technical Systems. In *Proc. of International Conference on Conceptual Modeling*, pages 270–283, 2013.
- [90] D. Parker. Our Excessively Simplistic Information Security Model and How to Fix It. *Information Systems Security Association*, pages 12–21, 2010.

- [91] D. B. Parker. *Fighting computer crime - a new framework for protecting information*. Wiley, 1998.
- [92] J. L. Rasmussen and M. Singh. Designing a Security System by Means of Coloured Petri Nets. *Proc. of International Conference on Application and Theory of Petri Nets and Concurrency*, pages 400–419, 1996.
- [93] C. Ribeiro and P. Guedes. Verifying workflow processes against organization security policies. In *Proc. of International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 190–191. IEEE, 1999.
- [94] E. Rios, F. Malmignati, E. Iturbe, M. D’Errico, and M. Salnitri. From consumer requirements to policies in secure services. In *Secure and Trustworthy Service Composition*, pages 79–94. 2014.
- [95] A. Rodríguez, E. Fernández-Medina, and M. Piattini. A BPMN extension for the modeling of security requirements in business processes. *Institute of Electronics, Information and Communication Engineers, Transaction on Information and Systems*, 90(4):745–752, 2007.
- [96] J. Rushby. Using Model Checking to Help Discover Mode Confusions and Other Automation Surprises. *Reliability Engineering and System Safety*, 75:167–177, 2002.
- [97] S. Sadiq, G. Governatori, and K. Namiri. Modeling Control Objectives for Business Process Compliance. *Proc. of International conference on business process management*, pages 149–164, 2007.
- [98] M. Saleem, J. Jaafar, and M. Hassan. A Domain-Specific Language for Modelling Security Objectives in a Business Process Models of SOA Applications. *Advances in Information Sciences and Service Sciences*, 4(1):353–362, 2012.

- [99] M. Salnitri, A. D. Brucker, and P. Giorgini. From secure business process models to secure artifact-centric specifications. In *Enterprise, Business-Process and Information Systems Modeling*, pages 246–262. Springer, 2015.
- [100] M. Salnitri, F. Dalpiaz, and P. Giorgini. Aligning Service-Oriented Architectures with Security Requirements. *Proc. of OnTheMove Federated Conferences and Workshops*, pages 232–249, 2012.
- [101] M. Salnitri, F. Dalpiaz, and P. Giorgini. Modeling and Verifying Security Policies in Business Processes. *Proc. of Business Process Modeling, Development, and Support working conference*, pages 200–214, 2014.
- [102] M. Salnitri, F. Dalpiaz, and P. Giorgini. Designing secure business processes with SecBPMN. *Software & Systems Modeling*, pages 1–21, 2015.
- [103] M. Salnitri and P. Giorgini. Modeling and Verification of ATM Security Policies with SecBPMN. In *Proc. of International Workshop on Security and High Performance Computing Systems*, 2014.
- [104] M. Salnitri and P. Giorgini. Transforming Socio-Technical Security Requirements in SecBPMN Security Policies. In *Proc. of IStar*, 2014.
- [105] M. Salnitri, E. Paja, and P. Giorgini. Preserving compliance with security requirements in socio-technical systems. *Proc. of Cyber Security and Privacy forum*, pages 49–61, 2014.
- [106] M. Salnitri, E. Paja, M. Poggianella, and P. Giorgini. STS-Tool 3.0: Maintaining Security in Socio-Technical Systems. In *Proc. of Conference on Advanced Information Systems Engineering Forum*, pages 205–212, 2015.

- [107] P. Samarati and S. Vimercati. Access Control: Policies, Models, and Mechanisms. In *Foundations of Security Analysis and Design*, volume 2171, pages 137–196. 2001.
- [108] SAP Payment Engine Website. Last visited March '15. www.sap.com/services-support/svc/custom-app-development/cnsltg/prebuilt/payment-engine/index.html.
- [109] SAP SE. *SAP River Developer Guide*, 2014. Document Version 1.0 – 2014-08-21, SAP HANA SPS 08, revision 82.
- [110] Sauro, J. Measuring Usability with the System Usability Scale (SUS). Last visited April 2015. <http://www.measuringusability.com/sus.php>.
- [111] A. W. Scheer. *Business process engineering: reference models for industrial enterprises*. Springer Science & Business Media, 2012.
- [112] A. W. Scheer and M. Nüttgens. Aris architecture and reference models for business process management. In *Business Process Management, Models, Techniques, and Empirical Studies*, pages 376–389, 2000.
- [113] R. Schmidt, C. Bartsch, and R. Oberhauser. Ontology-based Representation of Compliance Requirements for Service Processes. In *Semantic Business Process and Product Lifecycle Management*, 2007.
- [114] SecBPMN Website. SecBPMN website, last visited April 2016. <http://www.secbpmn.disi.unitn.it>.
- [115] R. Simon and M. Zurko. Separation of duty in role-based environments. In *Proc. of Computer Security Foundations Workshop*, pages 183–194, 1997.
- [116] G. Sindre and A. L. Opdahl. Eliciting Security Requirements with Misuse Cases. *Requirements Engineering*, 10(1):34–44, 2005.

- [117] I. Sommerville, D. Cliff, R. Calinescu, J. Keen, T. Kelly, M. Kwiatkowska, J. Mcdermid, and R. Paige. Large-scale complex IT systems. *Communications of the ACM*, 55(7):71–77, July 2012.
- [118] L. Sterling and K. Taveter. *The art of agent-oriented modeling*. MIT Press, 2009.
- [119] H. Störrle. VMQL: A Visual Language for Ad-hoc Model Querying. *Journal of Visual Languages and Computing*, 22:3–29, 2011.
- [120] The Apache Software Foundation. Apache Rampart website, last visited April 2016. <http://axis.apache.org/axis2/java/rampart/>.
- [121] W. M. van der Aalst. Business process management: A comprehensive survey. *ISRN Software Engineering*, 2013.
- [122] A. van Lamsweerde. Elaborating Security Requirements by Construction of Intentional Anti-Models. In *Proc. of International Conference on Software Engineering*, pages 148–157. IEEE, 2004.
- [123] J. Wainer, P. Barthelmess, and A. Kumar. W-RBAC - A Workflow Security Model Incorporating Controlled Overriding of Constraints. *IJCIS*, 12, 2003.
- [124] R. Wilson. *Introduction to Graph Theory*. 1986.
- [125] W.M.P. van der Aalst. Formalization and Verification of Event-Driven Process Chains. *Information and Software Technology*, 41(10):639 – 650, 1999.
- [126] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*. 2000.
- [127] C. Wolter, M. Menzel, A. Schaad, P. Miseldine, and C. Meinel. Model-driven business process security requirement specification. *Journal of Systems Architecture*, 55(4):211 – 223, 2009.

- [128] C. Wolter and A. Schaad. Modeling of task-based authorization constraints in BPMN. *Proc. of Business Process Management conference*, pages 64–79, 2007.
- [129] E. S.-K. Yu. *Modelling strategic relationships for process reengineering*. PhD thesis, Toronto, Ont., Canada, Canada, 1996. UMI Order No. GAXNN-02887 (Canadian dissertation).

List of Acronyms

SEBE SEcure Business process Engineering

SecBPMN2 Secure BPMN 2.0

SecBPMN2-ml Secure BPMN 2.0 - modeling language

SecBPMN2-Q Secure BPMN 2.0 - Query

SecBPMN Secure BPMN 1.1

SecBPMN-ml Secure BPMN 1.1 - modeling language

SecBPMN-Q Secure BPMN 1.1 - Query

STS-ml Socio-Technical Security - modeling language

STS-Tool Socio-Technical Security - Tool

PWC PricewaterhouseCoopers

BPMN Business Process Management and Notation

BPMN 2.0 Business Process Management and Notation 2.0

BPMN-Q BPMN-Query

RQ Research Question

SC Success Criteria

RDL River Definition Language

PE Payment Engine

RMIAS Reference Model on Information Assurance and Security

CIA Confidentiality Integrity Availability

BMIS Business Model for Information Security

TLS Transport Security Layer

SAVE Source Address Validity Enforcement

RBAC Role-Based Access Control

ER Enforcement Rule

GBRAM Based Requirements Analysis Method

BPM Business Process Management

ARIS Architecture of Integrated Information Systems

EPC Event-Process Chain

RDA Role Activity Diagram

MoSSBP Modeling Security Semantics of Business Processes

WS-BPEL Web Service - Business Process Execution Language

SoA Service-oriented Architecture

XACML eXtensible Access Control Markup Language

UML Unified Modeling Language

BP-QL Business Process Query Language

APQL A Process model Query Language

VMQL Visual Model Query Language

BPQL Business Process Query Language

OCL Object Constraint Language

RBAC Role Based Access Control

FCL Formal Contract Language

FPSPARQL Folder-Path Protocol and RDF Query Language

KAOS Knowledge Acquisition in autOmated Specification

GQM Goal, Question, Metric

CTL Computational Tree Logic

BPMNts BPMN textual security

OPBG Ospedale Pediatrico Bambin Gesu'

MISE Ministero Italiano Sviluppo Economico

DAEM Dimos Athinaion Epicheirisi Michanografisis

EQ Evaluation Question

IDE Integrated Development Environment

ABAP Advanced Business Application Programming

Appendix A

Transformation rules for STS-ml elements

This appendix shows the complete list of default transformation rules provided in SEBE which permits to generate SecBPMN2-ml business processes from STS-ml elements. Table A.1 shows transformation rules for actors, Table A.2 shows transformation rules for goal and relations between goals and documents, while Table A.3 shows transformation rules for relations between actors.

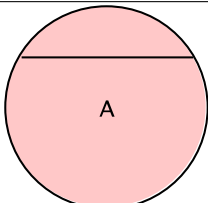
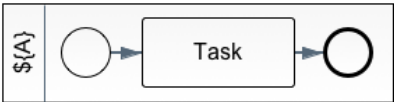
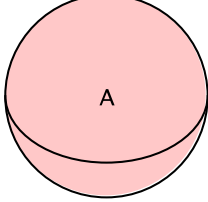
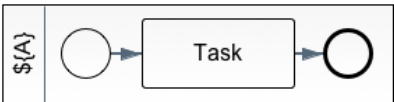
Name	STS-ml elements	SecBPMN2-ml business processes	Mapping
Agent			Agent : Pool
Role			Role : Pool

Table A.1: Default transformation rules provided in SEBE to transform STS-ml into SecBPMN2-ml models

APPENDIX A. TRANSFORMATION RULES FOR STS-ML ELEMENTS

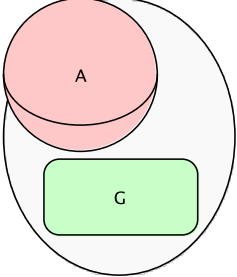
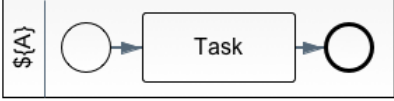
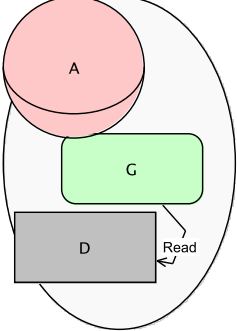
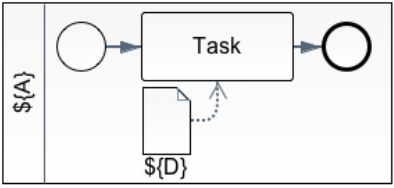
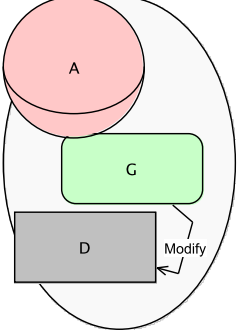
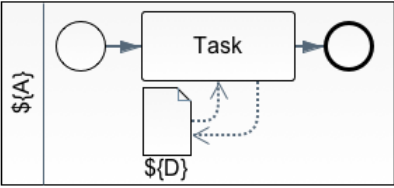
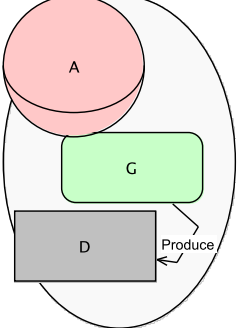
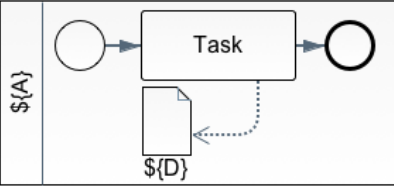
Name	STS-ml elements	SecBPMN2-ml business processes	Mapping
Goal			Actor : Pool, Goal : Process
Read			Actor : Pool, Read Rel.: Process, Document: Data object
Modify			Actor : Pool, Modify Rel.: Process, Document: Data object
Produce			Actor : Pool, Produce Rel.: Process, Document: Data object

Table A.2: Default transformation rules provided in SEBE to transform STS-ml actors into SecBPMN2-ml models

Name	STS-ml elements	SecBPMN2-ml business processes	Mapping
Transmission	<p>Diagram showing Actor A (red circle) sending document D (grey rectangle) to Actor B (red circle).</p>	<p>Diagram showing Actor A (red circle) sending message "Send \${D}" (envelope icon) to Actor B (red circle) via a "Task" (rectangle). A dashed arrow indicates the message flow from Actor A to Actor B.</p>	Actor A: Participant $\{A\}$, Actor B: Participant $\{B\}$, Document: Message, Transmission : Processes
Delegation	<p>Diagram showing Actor A (red circle) sending goal G (green rounded rectangle) to Actor B (red circle).</p>	<p>Diagram showing Actor A (red circle) sending message "Request for \${G}" (envelope icon) to Actor B (red circle) via a "\${G} process." (rectangle). A dashed arrow indicates the message flow from Actor A to Actor B.</p>	Actor A: Participant $\{A\}$, Actor B: Participant $\{B\}$, Delegation : Process

Table A.3: Default transformation rules provided in SEBE to transform STS-ml goals and relation between goals and documents into SecBPMN2-ml models

APPENDIX A. TRANSFORMATION RULES FOR STS-ML ELEMENTS

Appendix B

Transformation rules for security requirements

This appendix shows the transformation rules provided in SEBE to generate security policies from STS-ml security requirements. Table B.1 shows the transformation rules provided in SEBE for security requirements that can be specified on delegation of goals. Table B.2 shows the template for the four type of redundancy that can be defined in STS-ml. Table B.3 defines the template for security requirements that can be specified on transmission of documents. Table B.4 shows the template of security requirements that can be specified using authorizations in STS-ml. In this table is not shown the information perspective of the STS-ml models, which is common to all of them, and link the information I with the document D trough a **Made tangible by** relation. Table B.5 shows the template for compatible and incompatible security requirements.

APPENDIX B. TRANSFORMATION RULES FOR SECURITY REQUIREMENTS

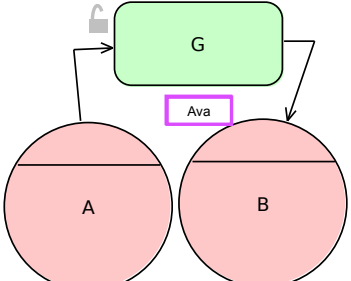
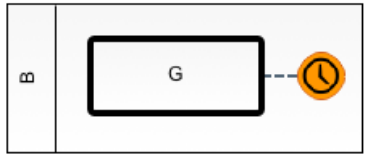
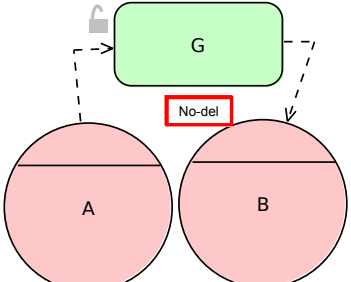

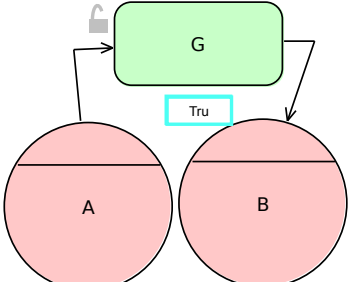

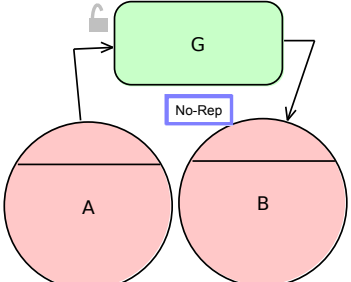

Security requirement	SecBPMN2-Q template	Mapping
<p>Availability</p> 		<p>Agent/Role : Pool Goal: Call activity</p>
<p>non-delegation</p> 		<p>Agent/Role : Pool Goal: Call activity</p>
<p>Trustworthiness</p> 		<p>Agent/Role : Pool Goal: Call activity</p>
<p>non-repudiation</p> 		<p>Agent/Role : Pool Goal: Call activity</p>

Table B.1: Default transformation rules for security requirements on delegations

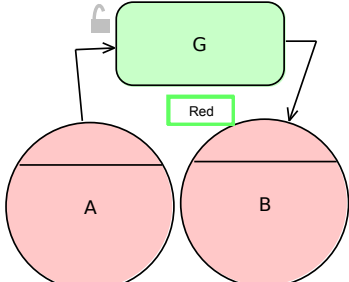
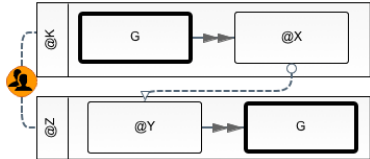
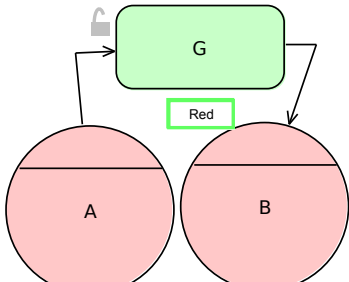
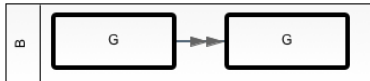
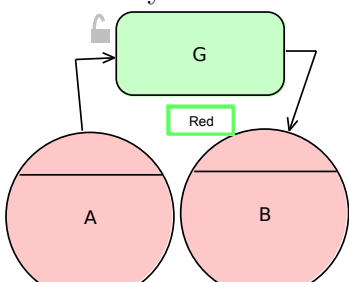
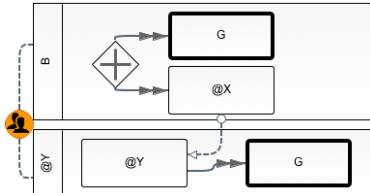
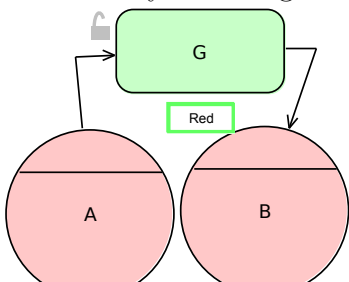
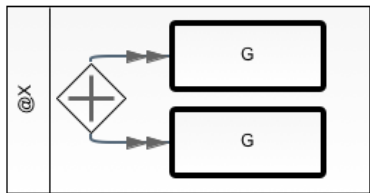
Security requirement	SecBPMN2-Q template	Mapping
<p>Redundancy fallback-multi</p> 		<p>Agent/Role : Pool Goal: Call activity</p>
<p>redundancy-fallback-single</p> 		<p>Agent/Role : Pool Goal: Call activity</p>
<p>redundancy-true-multi</p> 		<p>Agent/Role : Pool Goal: Call activity</p>
<p>redundancy-true-single</p> 		<p>Agent/Role : Pool Goal: Call activity</p>

Table B.2: Default transformation rules for redundancy security requirements

APPENDIX B. TRANSFORMATION RULES FOR SECURITY REQUIREMENTS

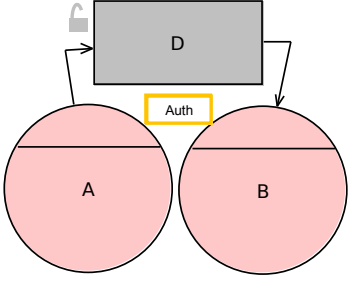
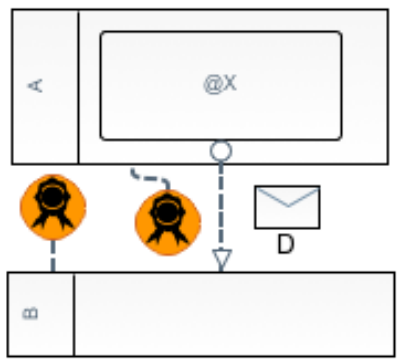
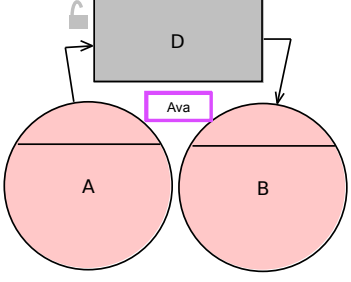
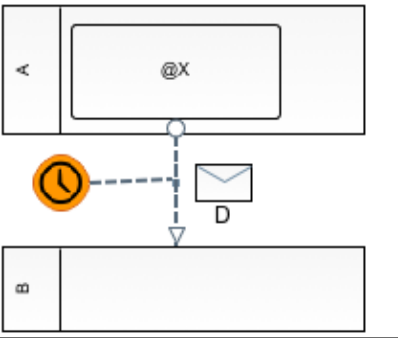
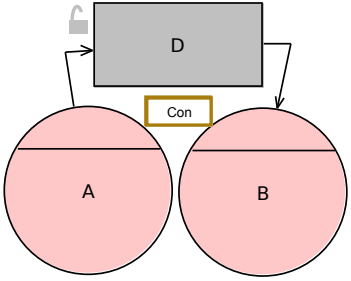
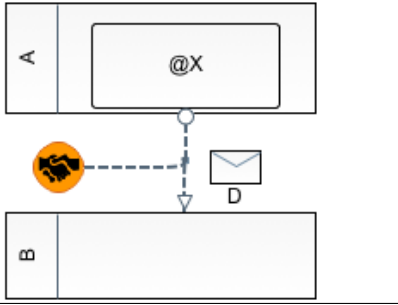
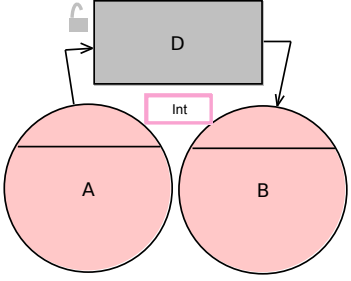
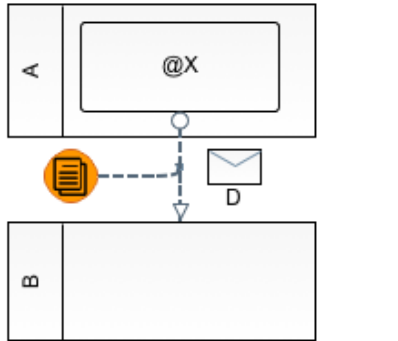
Security requirement	SecBPMN2-Q template	Mapping
<p>Authentication</p> 		<p>Agent/Role : Pool Document : Message</p>
<p>Availability</p> 		<p>Agent/Role : Pool Document: Message</p>
<p>Confidentiality</p> 		<p>Agent/Role : Pool Document: Message</p>
<p>Integrity</p> 		<p>Agent/Role : Pool Document: Message</p>

Table B.3: Default transformation rules for security requirements on transmissions

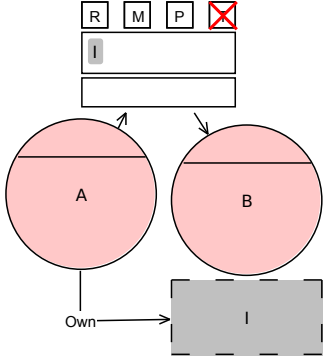
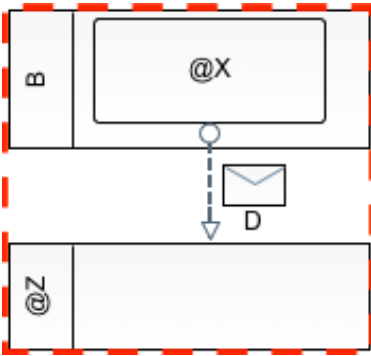
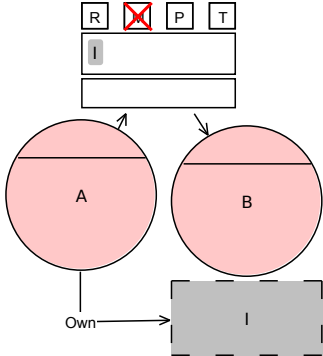

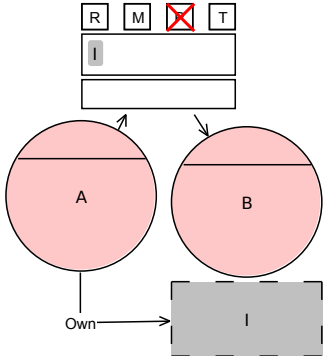

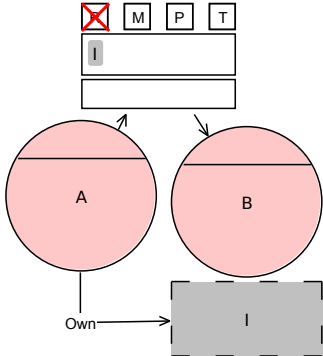

Security requirement	SecBPMN2-Q template	Mapping
<p>non-disclosure</p> 		<p>Agent/Role : Pool Document: Message</p>
<p>non-modification</p> 		<p>Agent/Role : Pool Document: Data object</p>
<p>non-production</p> 		<p>Agent/Role : Pool Document: Data object</p>
<p>non-reading</p> 		<p>Agent/Role : Pool Document: Data object</p>

Table B.4: Default transformation rules for authorizations

APPENDIX B. TRANSFORMATION RULES FOR SECURITY REQUIREMENTS

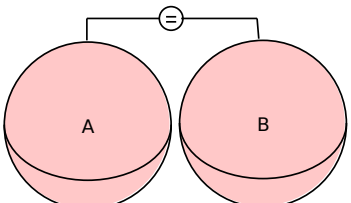
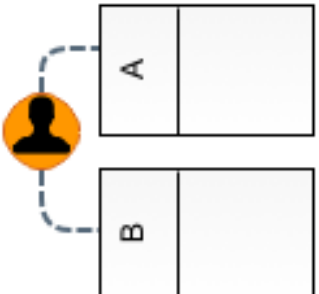
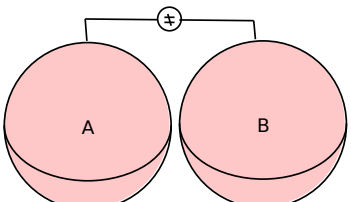
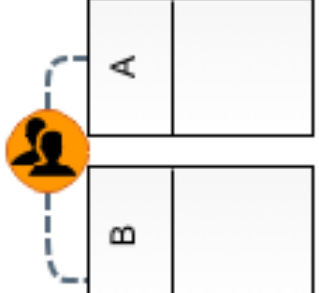
Security requirement	SecBPMN2-Q template	Mapping
<p>Compatible</p> 		<p>Agent/Role : Pool</p>
<p>Incompatible</p> 		<p>Agent/Role : Pool</p>

Table B.5: Default transformation rules for compatible and incompatible requirements