# Towards a Methodological Tool Support for Modeling Security-Oriented Processes

Jacob Geisel, Brahim Hamid, David Gonzalez, Jean-Michel Bruel

# Towards a Methodological Tool Support for Modeling Security-Oriented Processes

Jacob Geisel[1], Brahim Hamid[1], David Gonzales[2] and Jean-Michel Bruel[1]

[1] IRIT, University of Toulouse, Toulouse, France
{geisel,hamid,bruel}@irit.fr
[2] Ikerlan, Mandragon, Spain
DGonzalez@ikerlan.es

**Abstract.** Development processes for software construction are common knowledge and widely used in most development organizations. Unfortunately, these processes often offer only little or no support in order to meet security requirements. In our work, we propose a methodology to build domain specific process models with security concepts on the foundations of industry-relevant security approaches, backed by a security-oriented process model specification language. Instead of building domain specific security-oriented process models from the ground, the methodology allows process designers to fall back on existing well established security approaches and add domain relevant concepts and repository-centric approaches, as well as supplementary information security risk management standards (e.g., Common Criteria), to fulfill the demand for secure software engineering. Supplementary and/or domain specific concepts can be added trough our process modeling language in an easy and direct way. The methodology and the process modeling language we propose have been successfully evaluated by the TERESA project for specifying development processes for trusted applications and integrating security concepts into existing process models used in the railway domain.

**Keywords.** Process Modeling, Secure Software Engineering, Model-Driven Engineering, MDE toolchain, Repository, Reuse.

## 1 Introduction

Development processes for software construction are common knowledge and mainstream practice in most development organizations. Unfortunately, these processes offer little support in order to meet security requirements and are rarely formalized. As a consequence, there are increased risks of security vulnerabilities that are introduced into software in various stages of development. Secure software (or software security) engineering aims to avoid security vulnerabilities in software by considering security aspects from the very beginning and throughout the life cycle. From another perspective, formalizing processes offers the ability to teach and communicate them and to reason about them.

The Semco project [15] aims at closing this gap by offering a framework for modeling and formalizing, on the one hand, modeling a set of artifacts (e.g., security patterns) and on the other hand to provide methodologies for model-based

development (e.g., pattern-based security-oriented development). The modeling is becoming a major paradigm in system engineering, engineering of embedded systems, and particularly in system software engineering [14], but also in process engineering with the appearance of process metamodels [5]. Model-Driven Engineering (MDE) offers tools to deal with the development of complex systems improving their quality and reducing their development cycles.

In this work, we propose a process modeling environment, which associates model-driven paradigms and established security engineering concepts, to support the design of repository-centric security-oriented process models. In this context, we propose a methodology to build domain specific process models with security concepts on the foundations of industry-relevant security approaches, backed by a security-oriented process model specification language. To enable reuse, common industry-relevant approaches for considering security aspects in process models are made available to process designers through process model skeletons. The methodology allows process designers to build domain specific security-oriented process models based on existing industry-relevant security-oriented approaches and potentially add supplementary information security risk management and/or repository concepts. As part of the assistance for the modeling of process models for secure applications, we implement a tool-chain based on the Eclipse platform to support the different activities of process modeling and a repository, providing a set of reusable process skeletons and process type libraries. The proposed solutions were evaluated in the TERESA project through a case study from the metrology domain.

The rest of this paper is organized as follows. Section 2 outlines existing work on (security-oriented) process metamodels and models. Section 3 outlines our approach on building security-oriented process models based on solid foundations. Section 4 introduces RCPM (Repository Centric Process Metamodel) and details the packages used for security. Section 5 describes the concrete syntax of the RCPM modeling language. Moreover, it presents possibilities of analyzing the process model under various points of view and describes our proposed toolset to support the methodology, including a textual process model editor and a repository of process artifacts. Section 6 concludes this paper, discussing the advantages and limits of our approach and giving an outlook on future work.

## 2 Related Work

We will give an overview on the existing approaches on formalizing process models, on industry-relevant process models as well as on approaches on taking into account security concepts.

*Process metamodeling.* Different process metamodels are proposed [8, 12] for modeling software engineering processes. These process metamodels are divided into different categories according to [7]. The viewpoint of process metamodel concentrates different aspects of methodologies that are used by these metamodels. In our context, process models will be created with the viewpoint of activity-oriented, as the development of security-oriented systems is more directly modeled in this viewpoint. SPEM2 (Software & Systems Process En-

gineering Metamodel) [12] was created by the OMG as a *de facto*, high-level standard for processes used in object-oriented software development. The scope of SPEM is purposely limited to the minimal elements necessary to define any software and system development process, without adding specific features for particular development domains or disciplines (e.g., project management, security). Other commonly used process metamodels like UMA or OPEN have similar characteristics.

*Process models.* The V-Model [3] development process, also called verification & validation model, is suggested by the standard IEC61508. It is a trustworthy software development model, which aims at taming the complexity of project management, and which is used by big companies. The Rational Unified Process (RUP), an implementation of the Unified Process, is a comprehensive process framework that provides industry-tested practices for software engineering [8]. It is an iterative software development process framework, providing prototypes during each iteration.

*Security engineering.* The focus is put on three forefront representatives, namely Microsoft's Security Development Life cycle (SDL), OWASP's Comprehensive, Lightweight Application Security Process (CLASP) and McGraw' Touchpoints, as they are recognized as the major players in the field. Microsoft's Security Development Life (SDL) cycle [10] is probably the most rigorous, most tool-supported and more oriented towards large organizations (e.g., Microsoft uses it internally). Microsoft defined this process in 2002 to address security issues frequently faced in development. It contains an extensive set of (security oriented) activities, which can be used as supporting activities in development process models. These activities are often related to functionality-oriented activities and complement them by adding security aspects. Proposed activities are grouped into classical development phases (i.e., Education, Design, Implementation, Verification, Release) to ease the introduction into existing approaches. Vast guidance, such as detailed description of methods and tool support, is available, enabling even less qualified practitioners to achieve the required outcome. These guidance go as far down as to give coding and compiling guidelines, which do not map to process model activities any more.

The Comprehensive, Lightweight Application Security Process [13] by the OWASP Consortium is a lightweight process containing 24 main activities. It can be customized to fit different projects (activities can be integrated) and focuses on security as the central role of the system. CLASP also offers a rich set of security support resources.

McGraw's work [9] is based on industrial experience and has been validated over time. It provides a set of best practices regrouped into 7 so-called touchpoints. The activities focus on risk management and flexibility and offer white-hat and black-hat approaches to increase security.

Common Criteria for Information Technology Security Evaluation [1] is a ISO/IEC standard for computer security certification. It is defined as a generic framework offering process designers and project managers can specify security functional requirements through protection profiles.

## 3 Approach

The methodology we propose is based on a repository of modeling artifacts. Once the repository is set up and populated with process model skeletons and process type libraries, the (end-user) process engineer begins building domain specific process models. The central idea of our methodology consists of building on existing security-oriented process models, which then are extended by the process designer to meet the domain demands and additional security-oriented concepts. The methodology is illustrated in Fig. 1.

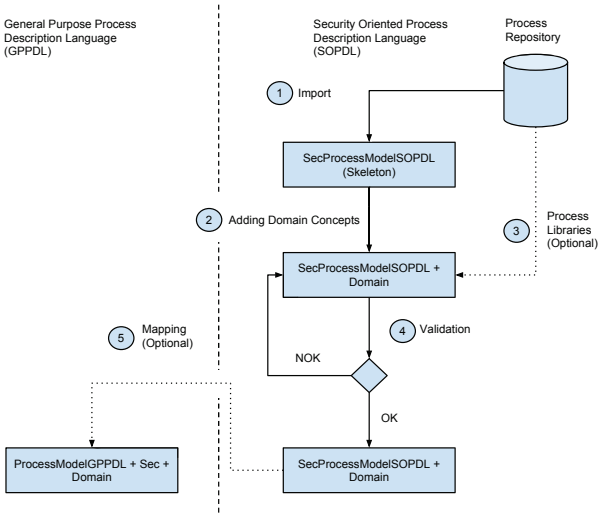


**Fig. 1.** From Scratch Methodology Overview

In the following we detail the different steps of this methodology and describe the alternatives the process designer has.

1. In a first step (**Step1**) the process designer chooses a security model skeleton as a template for the process model. The designer then imports the process model from the repository and creates a local copy of it in the process model design environment. The proposed framework makes available some process skeletons, based on the aforementioned secure development approaches, and offers process engineers to deposit process skeletons (as well as complete process models) to the repository.
2. In a second step (**Step2**) the process designer uses this process skeleton and adds domain specific phases, activities, roles, etc. to the process model coming from project-related recommendations e.g., in-house guidelines, non-formalized domain standards. In this step the process designer customizes the process skeleton, which is a rather generic (security-oriented) approach to fit the approaches used in the application domain.

3. In the following optional step (**Step3**) the designer has the possibility to select additional process type libraries to augment the process model (either domain specific process model libraries derived from standards or process type libraries for specific purposes, such as repository-centric or pattern-based libraries). The integration of the process libraries will be manual or semi-automatic, depending on the complexity of the process model and the library.
4. As next step (**Step4**) the designer validates the process model and iterates over the second step until validation passes. The designer has the choice between different validations, on the hand conformance validation towards the process metamodel and on the other hand supplementary validations, such as validations concerning correct implementation of the process type libraries.
5. Finally, the process designer can choose in a optional step (**Step5**) to map the created process model to a General Purpose Process Description Language (GPPDL) or to stay in a Security-Oriented Process Description Language (SOPDL). This allows the process designer to take advantages of both modeling environments, and to either use existing tools and frameworks to analyze and enact the process model, in the case of the initial *GPPDL* or any other *GPPDL*, or to take advantage of the proposed framework, in the case of the *SOPDL*, or to use both.

In the next section we will detail a metamodel to define a security-oriented process description language and concepts for reusing knowledge in process engineering, allowing process designers to follow the proposed methodology. In addition to the concepts of metamodel and the concrete syntax, we offer a repository with process model skeletons and process type libraries to give an infrastructure to build on to process designers. This repository includes process skeleton, like the SDL skeleton, and process type libraries for security (e.g., CLASP, Touchpoints) and repository-centric development approaches (e.g., PBSE)

## 4 RCPM concepts for security and reuse

The RCPM is a metamodel defining a new formalism for security process modeling based on a repository of modeling artifacts. The concepts of the metamodel, which are only briefly outlined have been presented in previous work [4, 6]. The complete description of the abstract syntax of the RCPM are available online via `http://www.semcomdt.org/semco/resources/RCPM.pdf`.

To illustrate the concepts presented in this paper, we will use the working example described in the following.

### 4.1 Working example: Simplified V-Modell XT

The *V-Modell XT* [2] is a high level framework and model for planning and realizing projects developed by the German government. It is the successor of the established *V-Modell 97*. The *V-Modell XT* allows to be tailored to specific needs of projects (e.g., size, budget, time constraints).

For demonstration purposes and better understandability, we simplify the process and focus on the software development part. Decision making and management phases and activities up to specifications upstream to the system development, as well as product maintenance are not treated.

### 4.2 Metamodel description

The Repository-Centric Process Metamodel (RCPM) is divided into the following six sub-packages: 1.) Core, regrouping basic concepts; 2.) Process, for concepts related to process engineering, based on Core; 3.) Safety, for safety related concepts, based on Process; 4.) Security, for security phases, activities, based on Process and Security; 5.) Repository, for interactions with a pattern repository, based on Process and Core and 6.) Types, for typing process elements and enforcing reuse of process elements, based on Core.

**Core concepts.**

*Core package.* The Core Package contains the elements which are used as top-level elements throughout the other packages and contain the basic attributes of all elements. These concepts include basic concepts (e.g., Element, Association) and their attributes (e.g., name, description).

*Process package.* The Process Package contains all the concepts used for process engineering, the basic concepts, like Process Model, concepts of a work breakdown structure (e.g., Phase, Activity, Task) and a breakdown structure (e.g., Role, Tool, WorkProduct) and concepts needed for detailing activities (e.g., Steps and Relationships as Responsible, Workdirection, Performer). This package is largely inspired by either existing process metamodels, such as SPEM2.0, UMA and/or OPF as well as by industry used process models such as the V-Modell XT.

**Safety engineering package.** Based on the Process Package, the Safety Engineering Package regroups recurring Safety Engineering Concepts and extends and enhances process concepts. The safety concepts of this package are derived from process models which are safety oriented, such as the V-Model XT.

**Security engineering package.** The Security Engineering Package regroups recurring Security Concepts, like Activities, Phases or Checkpoints. It is based on the Process and the Safety Package and reuses their concepts to express security-oriented concepts.

- **Recurring Elements**
  - *SecurityEngineer.* A Role describing a Security Engineer of the system-under-development.

- *ThreatModeling.* Recurring Activity to define sets of possible attacks on system assets.
- *SecurityReview.* Checkpoint targeting the entire system with a focus on highest-risk components. This is normally a checkpoint at the end of each phase.
- *AttackSurfaceReduction.* Activity to minimize attack surface (reduction of privileges and/or access points).
- *RiskAssessment.* Checkpoint to determine the quantitative or qualitative value of risk related to a concrete situation and a recognized threat.

– **Education and Project Inception**
  - *Education Phase.* Commonly used Phase for learning of security aspects. This phase also allows to create common security knowledge in the team.
  - *SecurityTeamBuilding Activity.* Addressing the set-up of security-oriented responsibilities, project or company-wide.
  - *SecurityLogistics Activity.* Addressing logistic aspects (e.g., tools, type of security bugs to be handled).
  - *SecurityMetrics Activity.* Assessing the security posture of the product as well as enforcing accountability of security issues.
– **Analysis and Requirements**
  - *AnalysisLevelThreatModeling.* A refinement of ThreatModeling, using different approaches on threat modeling, such as use case driven, resource driven and/or knowledge driven, assessing whether known attacks can be valid and useful
  - *SecurityRequirements.* An Activity specialized on defining the security requirements of the system-under-development. These requirements contain legal, financial, contractual and functional security requirements. This activity also resolves deficiencies and conflicts between requirement sets.
– **Architecture**
  - *ThirdPartyRiskAssessment.* Refinement of RiskAssessment to analyze weaknesses that arise by using third party software such as off-the-shelf components.
  - *RequirementsAudit.* Checkpoint to audit security (and non-security) requirements in order to assess their completeness.
  - *ArchitectureLevelThreatModeling* A refinement of ThreatModeling focussing on threat identification and risk assessment where risks in the system are identified and mitigated
  - *SecurityArchitecture.* Activity to take into account the security requirement in the architecture of the system-under-development.

**Type package.** The Types Package is used to define libraries for reuse of process blocks and to create constraints on Breakdown, Work Breakdown and Association Elements. The type elements correspond to the existing process elements and associations in the other packages (i.e., process, safety, security, repository).

– *TypeLibrary.* Library enabling reuse of process blocks (e.g., Phases, Activities), containing types and links among types.
– *ProcessElementTypes.* Generic Process Element Type.
– *AssociationTypes.* An Element allowing to type different kinds of associations.
– *WorkBreakdownElementTypes.* An Element (and especially its derived Elements) allowing to build a Work Breakdown structures. for reuse
– *BreakdownElementTypes.* An Element (and especially its derived Elements) allowing to build a Breakdown structures for reuse.

## 5 Security-Oriented Process Modeling

### 5.1 Concrete Syntax

Most metamodels and/or abstract syntaxes offer one or more concrete syntaxes to instantiate their concepts. The standards UML and SPEM2, for example, provide concrete syntaxes with diagrams for different viewpoints, in a graphical manner with icons and links. Other metamodels and especially domain-specific modeling languages often come with a textual syntax. We provide a tree-based concrete syntax, derived automatically from the metamodel, but which is not as convenient as using a well domain-adapted concrete syntax. A text-based syntax offers process modeling engineers a common and accustomed way to model their processes. We choose to use an EBNF grammar to define a concrete syntax for the RCPM language.

### 5.2 Process model analysis and documentation

In this section we detail the possibilities of analyzing the process model under various points of view. We allow process designers to check the conformance of their process model to the metamodel, helping to find concepts and relationships breaking the conformance. Another analysis approach is to extract metrics on the process model. Process Model Metrics is an important tool to analyze and understand process models. By these metrics it is possible to point out problems and find ways to improve the process model (e.g., reduce complexity, error probability).

The metrics offered by the framework are (1) *Size*, the number of nodes (and/or arcs) in the process model or the number of nodes referenced by or nested in an element (approximately equivalent to LoC metric, (2) *Diameter*, longest path from start to end, exploring the process model from the beginning to the end, checking the different alternatives and taking into account different parameters for computation, (3) *Depth*, depth of nesting of elements in a process model and listing these according to a root element shows a depth metric for taking the process model as root element) and (4) Control Flow Complexity, summing up all the choices in a process model. In addition to these quality-oriented metrics we offer also metrics used by project managers helping evaluate time and resources consumption for the aimed project. These metrics include estimations on resource consumption (e.g., a Security Engineer intervenes in a certain number of Activities and Tasks, Security Documentation is made up of a certain number of Work Products) and time consumption (e.g., the longest path, regarding estimated time, from possible alternatives from ActivityA to ActivityB).

In addition to giving analysis approaches on process models to process engineers and project managers, we allow through generation of documentation to extract process information and guidelines for enacting practitioners of the process model from different viewpoints and for different parts of the process model.

### 5.3 Tool Support

Using the proposed metamodels and the Eclipse Modeling Framework, ongoing experimental work is done with SemcoMDT as a MDE tool-chain supporting the proposed approach metamodels. We build a set of software tools, for designing process models, for populating and for retrieval from the repository. Moreover, we provide tools to support the management of the repository (Semco Gaya Repository), the generation of documentation and the transformations for refinement and analysis. We choose to derive a text-based syntax to create instances of the metamodel using the Xtext Framework (Semco Naravas Process Model Editor). For the description of the model transformations, the QVT Operational language is used and for metrics generation the Acceleo transformation engine [11] is used to build static HTML pages based on the Bootstrap Framework. An Example of the output is given in Fig. 2.

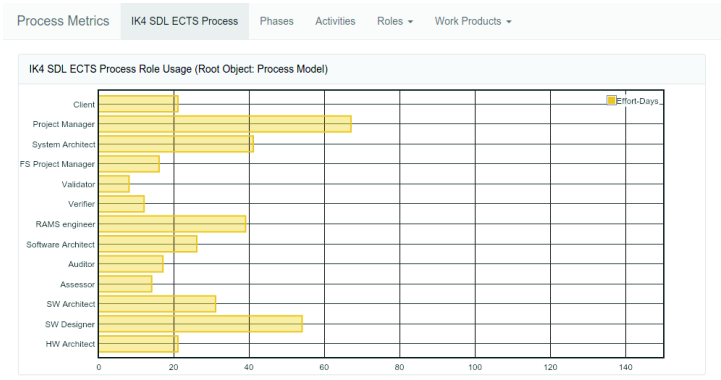


**Fig. 2.** NaravasX: Metrics on Security Engineer Effort Days

## 6 Conclusion

In this paper we propose a methodology to design process models with security aspects based on industry-relevant best-practices. This is realized by reusing and building upon security concepts from established security-oriented approaches. The security aspects of the modeling language are detailed and demonstrated on a working example through a text-based concrete syntax. The methodology and the security-oriented process modeling language are validated by a use case from the railway domain through the modeling of a process model for a Train Control System. The advantages of the approach are a more direct and intuitive way of building process models on existing security-oriented approaches and adding security concepts for domains having strong security requirements. In addition, to easing the process modeling from the ground up, assistance is

given to the process designer by model type libraries, guiding the designer to conform with domain specific guidelines and/or best practices. Despite the advantages of the approach and the modeling language, there are limits to the approach. Our security-oriented process modeling language is not able to represent all of the concepts given in SPEM2.0 or other GPPDLs (General Purpose Process Description Language), although this might not raise an issue, since the process concepts needed for security engineering are kept. Derived from this, the transformation from our process modeling language to a GPPDL might not be able to represent all the security concepts in the generic process description language in an explicit way.

## References


1. Common Criteria. Common criteria for information technology security evaluation v3.1r4. Technical Report CCMB-2012-09-001/002/003, Common Criteria, 2012.
2. Die Beauftragte der Bundesregierung für Informationstechnik. V-modell XT, 2005.
3. K. Forsberg, H. Mooz, and H. Cotterman. *Visualizing Project Management, A Model for Business and Technical Success*. Wiley, 2 edition, 2000.
4. J. Geisel, B. Hamid, and J.-M. Bruel. Repository-centric process modeling – example of a pattern based development process. In R. Lee, editor, *Software Engineering Research, Management and Applications*, number 496 in Studies in Computational Intelligence, pages 247–261. Springer International Publishing, 2014.
5. C. Gonzalez-Perez and B. Henderson-Sellers. Modelling software development methodologies: A conceptual foundation. *Journal of Systems and Software*, 80(11):1778–1796, Nov. 2007.
6. B. Hamid, J. Geisel, A. Ziani, and D. Gonzalez. Safety lifecycle development process modeling for embedded systems - example of railway domain. In P. Avgeriou, editor, *Software Engineering for Resilient Systems*, number 7527 in Lecture Notes in Computer Science, pages 63–75. Springer Berlin Heidelberg, 2012.
7. C. Hug, A. Front, D. Rieu, and B. Henderson-Sellers. A method to build information systems engineering process metamodels. *Journal of Systems and Software*, 82(10):1730–1742, Oct. 2009.
8. P. Kruchten. *The Rational Unified Process: An Introduction*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3 edition, 2003.
9. G. McGraw. *Software security: building security in*. Addison-Wesley Professional, 2006.
10. Microsoft. Microsoft security development lifecycle (SDL) process guidance - version 5.2, 2012.
11. OBEO. Acceleo. http://www.eclipse.org/acceleo/, 2014.
12. OMG. Software & systems process engineering metamodel specification (SPEM) version 2.0. Technical report, Object Management Group, Inc., 2008.
13. OWASP. *OWASP CLASP V1.2*. OWASP, Nov. 2007.
14. B. Selic. The pragmatics of model-driven development. *IEEE Softw.*, 20(5):19–25, 2003.
15. SEMCO. System and software engineering for embedded systems applications with multi-COncerns support, 2010.