

Security and Safety of Assets in Business Processes

Ganna Monakova
SAP Research Karlsruhe
Vincenz-Priessnitz-Str. 1
76131 Karlsruhe, Germany
ganna.monakova@sap.com

Achim D. Brucker
SAP Research Karlsruhe
Vincenz-Priessnitz-Str. 1
76131 Karlsruhe, Germany
achim.brucker@sap.com

Andreas Schaad
SAP Research Karlsruhe
Vincenz-Priessnitz-Str. 1
76131 Karlsruhe, Germany
andreas.schaad@sap.com

ABSTRACT

Business processes and service compositions are defined independent of the realizing systems. The visualization of security and safety constraints on the business process model level appears to be a promising approach to system independent specification of the security and safety requirements. Such requirements can be realized through business process annotation and used for communication or documentation, but they also can have an execution semantics that allows for automating the security and safety controls.

In this paper, we present a tool-supported framework that extends modeling and execution of business processes with specification, execution and monitoring of the security and safety constraints that are used to protect business assets. We illustrate our approach on basis of a case study modeling a supply chain for perishable goods.

Categories and Subject Descriptors

K.6.5 [Computing Milieux]: Management of Computing and Information Systems—*Security and Protection*

Keywords

Resource Modeling, Monitoring, Security, Safety, BPMN

1. INTRODUCTION

Workflow management systems are widely used for automating the day-to-day business of enterprises. The basis for this automation are abstract business process models, e. g., expressed in BPMN.

These abstract business process models are not only used on a technical level for automating the execution of processes. They are, as well, used for communication between the different stakeholders (e. g., the different parties involved in the process execution, business experts, compliance officers) of a business process. In both cases, security and safety of business process assets are important for all parties involved. For example, in a multi-party process the partners

want to be sure that assets sent to another party are treated correctly and that assets received from another party can be trusted. As many enterprises need to comply to regulations such as the European Food Safety regulations (e. g., see Regulation (EC) No 882/2004 of the European Parliament and of the Council of 29 April 2004 and related documents), there is strong demand to specify and communicate security and safety requirements. Furthermore, during the execution of a business process, the compliance with these security and safety requirements needs to be monitored and certified. Today, the business expert concentrates on the specification of the business aspects of a workflow, while security or safety aspects mostly remain neglected (if at all understood).

Security, safety, and compliance of a business process and involved assets are critical to any organization. For example, in a supply chain process the partners not only want to ensure that the purchase order data and the payment data are correct, but also want to be sure that the ordered good is treated according to various requirements. To obtain such an assurance, the partner must treat assets in a certain way, e. g., ensure that the temperature is correct, ensure correct packaging, or do contamination checks.

Together with retailers, freight carriers, and food manufacturers, we analyzed the security and safety requirements of food supply chains. As a result of this analysis, we identified the following objectives that must be supported by a framework for business level specification of the security and safety constraints:

- *Security and Safety Awareness*: a business process modeler should be aware of security and safety threats for the assets used in a business process.
- *Security and Safety Visibility*: a business user, modeling a process, should be able to communicate security and safety requirements through visual representations or annotations, similar to the business requirements.
- *Security and Safety Consistency*: requirements should be fulfilled in a consistent (best-practice) way.
- *Security and Safety Executability*: implementation of the identified countermeasures must be feasible.
- *Security and Safety Provability*: it should be possible to prove fulfillment of specified requirements.

In this paper, we present an approach for modeling and monitoring security and safety requirements that achieves the described objectives. Our approach built upon a generic framework supporting the specification as well as runtime enforcement and monitoring of a large class security and safety issues in consistent way. Concluding, our framework support the complete business process lifecycle.

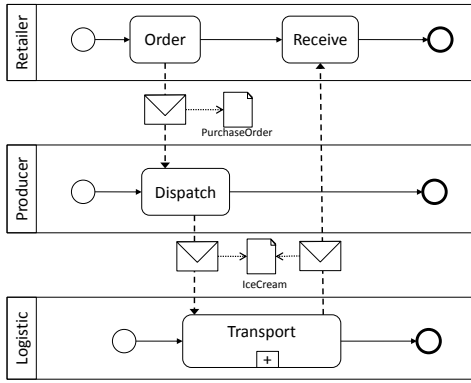


Figure 1: Example supply chain process

2. MOTIVATING EXAMPLE

According to [8], approximately one-third of all fresh fruit and vegetables produced worldwide is lost before it reaches consumers. In [9] the authors state that sometimes the losses and wastage of the food may even reach 50 percent between field and fork. Incorrect harvesting, transport, storage and packaging play an important role in these losses.

Figure 1 presents a simplified ice cream supply chain process involving three parties: *Retailer* sends an order to *Producer* in the *Order* activity; *Producer* dispatches the required amount of the product (*Dispatch* activity) and uses *Logistic* partner to deliver it (*Transport* sub-process) to *Retailer*. There are two data objects modeled in the process: *PurchaseOrder* object contains all information required to make an order, including required amount of the product and the delivery destination; *IceCream* data object represents the actual physical good that is passed between the supply chain participants.

PurchaseOrder contains sensitive information and must be protected against tampering. For instance, the *Producer* wants to be sure that the requested amount and the destination address have not be changed by an unauthorized party. *Retailer* wants to be sure that *IceCream* has been handled in a correct manner, e.g., temperature of the product was always in the region of -26°C to -25°C and that there was no unauthorized access to the product during transportation to ensure that product was not deliberately contaminated.

With respect to the four objectives we identified, the desired outcome, in our example, is as follows: support during business process design with identification of the potential threats for the *PurchaseOrder* and *IceCream* assets to achieve *security awareness*; automated help in identification of countermeasures for the identified threats to achieve *consistency* in applying security controls; for example suggestion to control *PurchaseOrder* integrity through usage of digital signature or control correct handling of *IceCream* through suggestion of usage of a temperature sensor to detect temperature violation; suitable tools for visualization of the security measures taken to protect *PurchaseOrder* against tampering and *IceCream* against contamination for *security visibility* and *provability* on the design level; extension of the business process engine that would allow to execute security measures; automated evidence collection at runtime, such as monitoring of the ice cream temperature, to achieve *provability* of the taken measures.

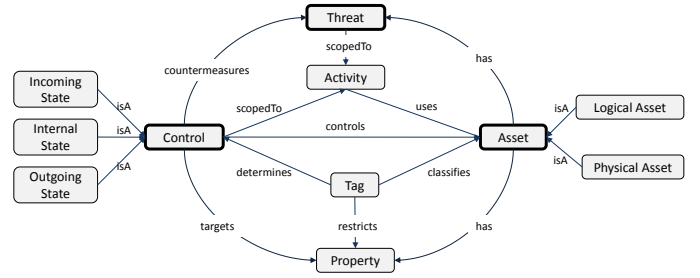


Figure 2: Conceptual Model

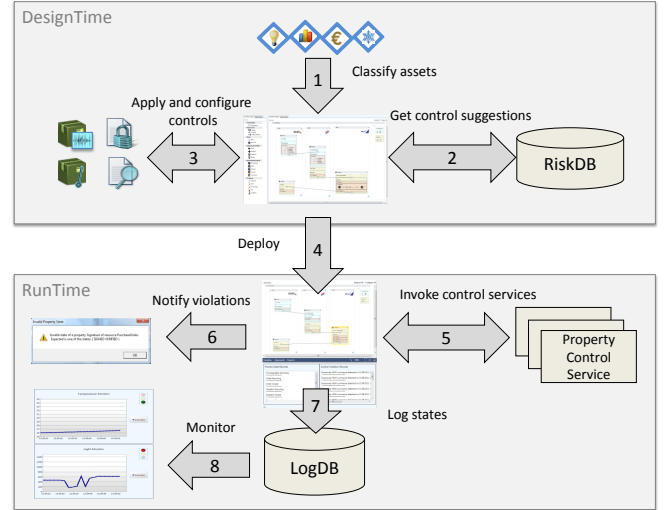


Figure 3: Approach in prototypical implementation

3. PROPOSED APPROACH

Figure 2 shows the conceptual model of presented approach, while Figure 3 gives an overview over the proposed approach. The three main concepts in the model are *Asset*, *Threat*, and *Control*. An asset has potential threats and certain controls can countermeasure these threats. The role of the rest of the model is to help identify which threats are applicable to which asset and which controls can be used to countermeasure these threats. The following describes the steps of our approach based on the conceptual model.

1. *Asset identification*. In this step we analyzed what are the assets used in a business process that we want to target in our work. We identified two types of assets: *logical asset* is the data that contains certain sensitive information, such as purchase order details or credit card number, while *physical asset* is a real world object that is used in the business process, such as *Good* in the supply chain process described in Section 2. In general, these assets will be represented through data objects in BPMN or through variables in BPEL. Any asset can be described by a set of *Properties* it possesses. Thereby any logical asset can be described by a set of the same properties, such as *signature*, *content* and *encryption* properties. Similar, any physical asset can be described by the set of the same properties, such as *temperature*, *location* and *size*. Each property is defined by a set of states it can adopt. For exam-











Physical Asset		Logical Asset	
Deep-frozen		Private	
Light-sensitive		Financial	
Explosive		Legal	
Poisonous		Confidential	
Radioactive		Audit-relevant	

Figure 4: Tags for logical and physical assets

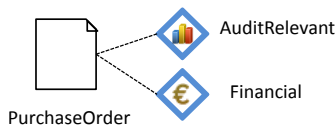


Figure 5: Possible BPMN data object annotation

- ple, temperature property can be in a state -18°C or $+5^{\circ}\text{C}$, while signature property can be in a state *Signed*, *Unsigned*, *Verified* or *Unknown*.
2. *Asset classification*. Different threats are applicable to different assets depending on asset classification. Thereby it is not sufficient to distinguish between logical and physical assets. For example, two logical assets can have different threats: the first logical asset might contain *private* information about a customer with a threat of information disclosure, while another logical asset might contain *financial* data, which has threat of unauthorized modification. Similar, a *frozen* physical asset might have threat of being, defrozed, while a *fragile* physical asset has a threat of being broken. To allow business process designer to classify business assets, a concept of *Tag* has been introduced. A tag attached to an asset identifies a certain characteristic or classification of this asset. Figure 4 shows an example set of tags that can be used to classify logical and physical assets. As an asset is represented through a data object in a BPMN business process, tags can be attached to the corresponding visual elements of the BPMN diagram for example as shown in Figure 5. In the supply chain example, *IceCream* can be annotated with tags *DeepFrozen* and *LightSensitive*, while *PurchaseOrder* can be annotated with tags *AuditRelevant* and *Financial*.
 3. *Controls identification*. To provide a generic methodology for relating controls to the assets, we classify controls based on the asset properties it can control. For example, a *temperature* property, can be monitored by a *temperature sensor* control. Similar, a *signature* property can be controlled by a *signature service* that can identify whether the document is signed and whether the signature is valid (monitor), or sign the











Physical Asset		Logical Asset	
Light		Signature	
Temperature		Encryption	
Concussion		Audit-Controls	
Pressure		Privacy-Policies	
Location		Separation of Duties	

Figure 6: Controls for logical and physical assets

document (enforcer). We distinguish between state-properties and range-properties, and correspondingly state-controls and range-controls. State properties are specified by a list of states a property can take and a state control contains specification of valid states for this property for a given asset at certain time. Range properties are defined by the range of the values it can take, and a range control contains the border specifications for the property values of an asset. Each tag attached to an asset can be viewed as a restriction on the certain asset properties. A tag puts restrictions on a property by restricting the set of valid states for this property for the asset. For example *DeepFrozen* tag puts constraints on the temperature property of a physical asset by restricting valid temperature values to under -18°C . Based on tags and implied property restrictions, controls can be identified. For example for the *DeepFrozen* tag a temperature control will be suggested. To achieve *consistency* in control identification, the required controls are identified based on the rules stored in a database. The rules derive required controls for each activity that uses an asset annotated with certain tags. Thereby controls can depend on multiple tags as well as on the type of activity that uses the asset. For example an asset that is *flammable* and *explosive* might require different controls than only *flammable* assets. Controls are implementations of a certain functionality that can control a certain property. For example, a temperature sensor can control *temperature* property, while a service that can sign and validate digital signatures is able to control *signature* property. Figure 6 gives an overview over sample controls for logical and physical assets. As mentioned above, controls are related to a certain asset property rather than to an asset, which allows to use the same controls with different assets that have the same property. A signature or encryption service can be used with multiple logical assets, as well as sensors can be used with multiple physical assets. A control “understands” a certain property and can be configured with the valid states for the property and the given asset. The role of the control is to ensure that

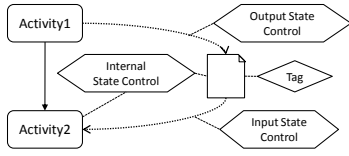


Figure 7: BPMN data object annotation

the asset property the control is responsible for is in a valid state. For example, for a deep-frozen pizza we need controls to ensure that the pizza temperature is under -18°C . Controls are scoped to activities, therefore different activities that use the same assets can have different controls applied to the same assets. Controls can be divided into three main categories: monitors, enforcers and auditors.

4. *Control points identification.* A control can be applied at different stages of an activity execution. If applied on activity initialization, it can control the incoming states of the asset properties; if applied on activity execution, it can control the internal states of the asset properties; if applied on activity completion, it can control outgoing states of the asset properties. Depending on the type of activity, different control types are applicable. Figure 7 shows how different control points can be represented in BPMN.

Incoming state controls and outgoing state controls can be enforced by the workflow engine—it can invoke control services to verify that the asset properties are in a correct states and can for example suspend a workflow (or execute any other activities that are defined as part of a reactive process) if a violation has been detected. The internal controls on the other side can be viewed as the requirements on the activity implementation with regard to the asset handling.

4. ARCHITECTURE

Figure 8 gives an overview over the architecture in a SOA environment. At the design time, the RiskDB is consulted to identify threats and countermeasures for the business process assets, that have been classified with the tags.

At the runtime, process execution engine invokes control services at the specified control points through the control service broker. All controls are available as *property control services* that subscribe to the property they can control in the RiskDB, specifying the type of the control (monitor, enforcer, or assessor) and the assets it can handle. The Business process engine sends the asset or asset reference and the property to control to the control service broker, which then looks up available services in the RiskDB and finds a service that can evaluate or change the state of the given property for the given asset.

All property states, as well as process execution states are stored in a LogDB, which feeds data into the dashboard and allows offline analysis of the completed instances and improvement of the rules specified in RiskDB.

5. IMPLEMENTATION

Our prototype is based on the Windows Workflow Foundation (WF 4.0). Figure 9 shows the prototype architecture, where the bold elements represent our extensions to WF.

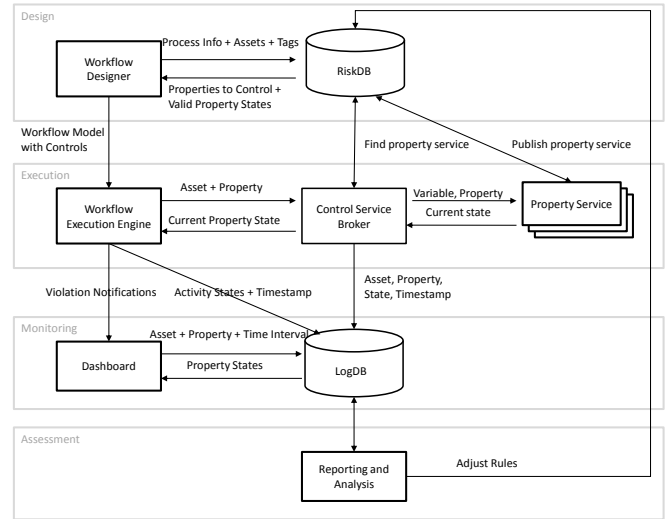


Figure 8: Architecture

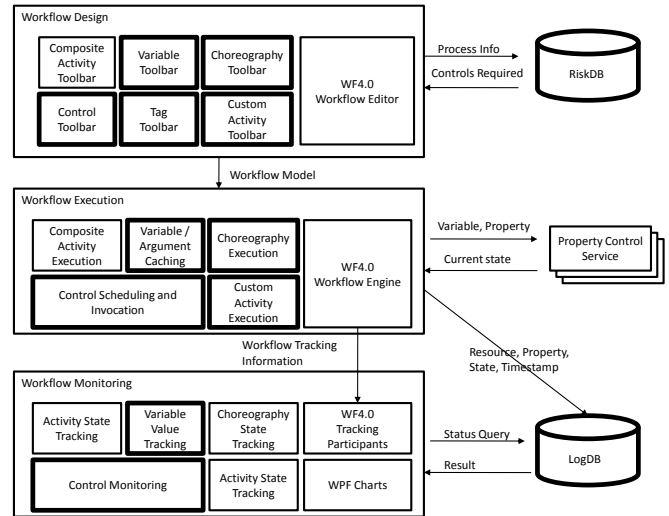


Figure 9: Implemented Extensions and Architecture

5.1 Workflow Design

WF uses variables to represent data used in a business process, however the variables are defined in a variable tab and are not visible in the designer. To advocate security awareness, we extended existing workflow modeling constructs with two visual elements for logical and physical assets. Furthermore, we added an asset (or variable) panel to the business process, which contains all assets used in the process. To add a new asset (variable) to the process, the user needs to drag & drop the corresponding visual element into the asset/variable panel of the workflow.

To enable asset classification we provide a tag toolbar: the user can drag & drop the corresponding tag from the toolbar onto the visual asset specification being present in the asset panel. By combining different tags, a user can specify different characteristics of an asset.

Figure 10 shows a screenshot of the ice cream supply chain process modeled using our tool. It contains two variables that can be seen in the right panel: IceCream variable

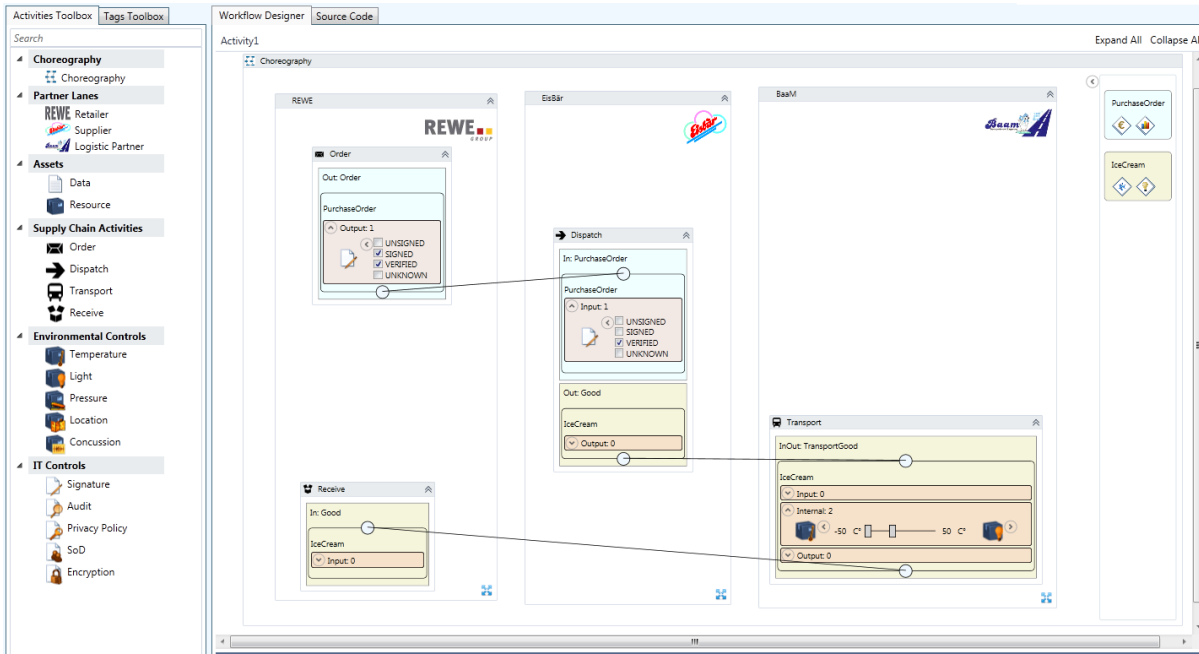


Figure 10: Design of the supply chain process

annotated with a *DeepFrozen* and *LighSensitive* tags, and *PurchaseOrder* variable annotated with *Financial* and *AuditRelevant* tags.

Figure 10 shows four activities: *Order*, *Dispatch*, *Transport*, and *Receive*. Activity *Order* outputs *PurchaseOrder*, which is then passed as input argument to *Dispatch* activity. *Dispatch* activity then outputs *IceCream*, which is passed to *Transport* activity and then through *Transport* activity to *Receive* activity. Depending on the type of argument (In, Out or InOut), we can see different types of control points available for each asset in each activity. This allows user to define input state controls on the incoming asset states (*PurchaseOrder* in *Dispatch* activity) output controls on outgoing asset states (*PurchaseOrder* in *Order* activity), and internal controls on data that exists all the way through activity execution (*IceCream* in *Transport* activity).

To identify controls required to countermeasure potential threats, we developed a Risk Database (RiskDB). The RiskDB stores relations between asset tags, threats these tags imply for different activities, and controls that should be applied to such assets in each activity. When a user annotates an asset with a new tag, a query is sent to the RiskDB that selects the necessary protection measurements (or controls) for each activity that uses this asset. After this the tool checks if the controls are already present in the model and if not, shows an error with the information about missing controls. This enforces business designer to model secure processes with respect to the rules stored in the RiskDB.

To enable control specification, we provide a control toolbar. To identify at which point of activity execution a control must be applied, the user needs to drop a control into the corresponding container. In Figure 10 we can see an output signature control applied to the *PurchaseOrder* variable in *Order* activity. This control specifies that the data must be signed when it leaves this activity. In *Dispatch* ac-

tivity we can see an example incoming state control, that states that *PurchaseOrder* signature property must be in state *verified* to be used by this activity. In the *Transport* activity internal temperature and light controls are specified, which define that *IceCream* temperature must be between -50°C and -25°C and light must be under 200 lm. Additional controls could be added as input and output controls. In general, any number of controls can be applied to each asset in each activity.

Design time extensions of the workflow foundation gives security visibility and awareness by providing tag and control toolbars, security awareness and consistency through connection to the RiskDB that consistently applies the same rules in similar situations and gives errors if any controls are missing, and security provability on design level by showing that there are no errors in the model with respect to the RiskDB rules.

5.2 Workflow Execution and Monitoring

To enable execution of the extensions, the visual assets have been mapped to the variables and passed as arguments into the corresponding activities.

Each control knows the property it targets. When a control is scheduled, it invokes the corresponding property control service. Such a service can be an internal implementation, such as automatic signature implementation, but can also be a remote service, such as sensor control that monitors resource temperature. In general, all property-specific actions are done by the property control services. This allows for a general model of the controls in the business process: a business process control knows the asset it needs to control, the property it targets, the service that can evaluate the state of this property, the point when the state need to be evaluated, and the states that are allowed at this point. The property control service knows how to find out the current state of the resource and how to modify the state, but

it is unaware of the business related semantics or the valid states of this property. At the specified execution point, the control asks a property service to evaluate the current state of an asset, logs results into the *LogDB*, compares it with the set of valid states and notifies user if the state is invalid.

All input controls scoped to an activity are evaluated before this activity starts its execution. If any violations are detected during these checks, the process terminates. After each activity execution, all output controls scoped to these activity are evaluated and if any violations are detected, the activity is reiterated until all assets have the valid property states. For the internal controls, monitors are triggered at the beginning of the activity execution and stopped when activity completes. Monitors observe and log the states of the corresponding properties during the activity execution with the specified frequency. Collected evidence can then be used to prove fulfillment of the specified restrictions. If a violation of an internal control is detected during activity execution, business process partners are notified. Process *termination*, activity *reiteration* and partners *notification* have been implemented as examples of the possible reactions to the control violations. In general, any customer-defined actions can be used as reactions to restriction violations.

Figure 11 shows an example monitoring screenshot taken during the simulation of the *Transport* activity in the supply chain process. On the right side we can see a chart representing the values logged by the temperature control. Light monitor has the red light, notifying that the violation has been detected. At the bottom left of the screen we can see the tracking information about the current state of the process execution and logged violations, while the currently active activity is highlighted on the top left part of the screen.

6. RELATED WORK

Related work can be divided in two main categories, business process visualization approaches that aim at improving the understanding of the model, and specification of security constraints on a business process.

Much work exists that investigates how a better process visualization can enhance understanding of the processes. Several use cases motivating the need for a proper business process visualization as well as a business process graph layouting approach were introduced in [12]. In [3] an approach for personalized visualization of processes and process data is presented. The approach is based on the independent definition of process symbols from process model data. In [10] the authors investigate how usage of icons representing task types improves understanding of the process model.

In [11] a 3D visualization approach is presented, that allows analysis of business process constraints and dependencies between different process dimensions. The BPMN provides a standard way for a business process to be graphically represented. Wide acceptance of BPMN shows the importance of business process visualization. While BPMN is widely used, the intention of the notation is to support visual exchange of knowledge between different parties, rather than to support protection of the assets used in the business process. In [6] authors propose a perspective oriented process modeling approach. They define different perspectives for a modeling construct, which then enables the generation of different views on a business process. The main perspectives identified are functional, data, operational, organiza-

tional and control flow perspectives. A layered meta model supports extensionality of the identified perspectives.

In [7] the author presents a formalized approach based on UML for expressing security-relevant information within the diagrams in a system specification. The model allows MDA-like formal analysis and verification of security mechanisms and protocols. The presented approach focuses on stereotypes, tagged values and constraints of the UML extensions mechanisms without any systematical identification and description of the security goals that can be expressed with the proposed model. Similarly, [2] present a meta-model based extension of UML that allows for specifying RBAC properties. While this approach provides a systematical identification of the properties that can be specified, it is limited to access control specifications based on RBAC. A comprehensive overview of available security modeling methodologies is provided in [1]. In [5] the authors present workflow related security goals and study their possible assignment to main categories of business process elements such as agents, roles, artifacts, and activities. Further they present a tool which allows domain experts to define the inspected security requirements expressed through UML activity diagrams. Further, they discuss how the abstract security goals can be checked for syntactical and semantical correctness. However, they do not provide any description of the enforcing and realizing mechanisms. Graphical extensions, which allow the specification of auditing, integrity, access control, non-repudiation, privacy and attack harm detection are introduced in [13]. They also present to which Business Process Diagram these security properties can be applied. The presented approach focuses only on BPMN and takes only graphical aspects into consideration. Typical security goals, such as authentication or confidentiality are addressed in [14]. In their paper the authors propose a model-driven transformation approach from security requirements up to concrete security implementations. They further discuss a translation of security annotated business processes into XACML and AXIS2 security configurations. However, they do not define any graphical notation and provide no prototypical implementation to their approach.

7. CONCLUSIONS

We presented an approach that allows a business user to easily specify security and safety requirements on the business process level. The compliance to these requirements is monitored during the execution of the processes. Overall, this transfers the well known model-driven software development paradigm to workflow management systems that can execute the abstract process models directly. While our prototype is based on the Windows Workflow Foundations, we already demonstrated how the various security and safety properties can be visualized in the context of BPMN models. Similarly, to the monitoring environment our prototype provides for the Windows Workflow Foundations, existing BPMN or BPEL execution environments can be extended with the necessary monitoring infrastructure. Moreover, we plan to develop support for enforcement (e.g., by generating configurations for XACML-based access control infrastructures) and audit controls (e.g., by supporting post-hoc analysis infrastructures, such as [4]).

The prototype has been developed in the context of a German funded project that develops techniques for security- and safety-critical supply chains. This prototype has been

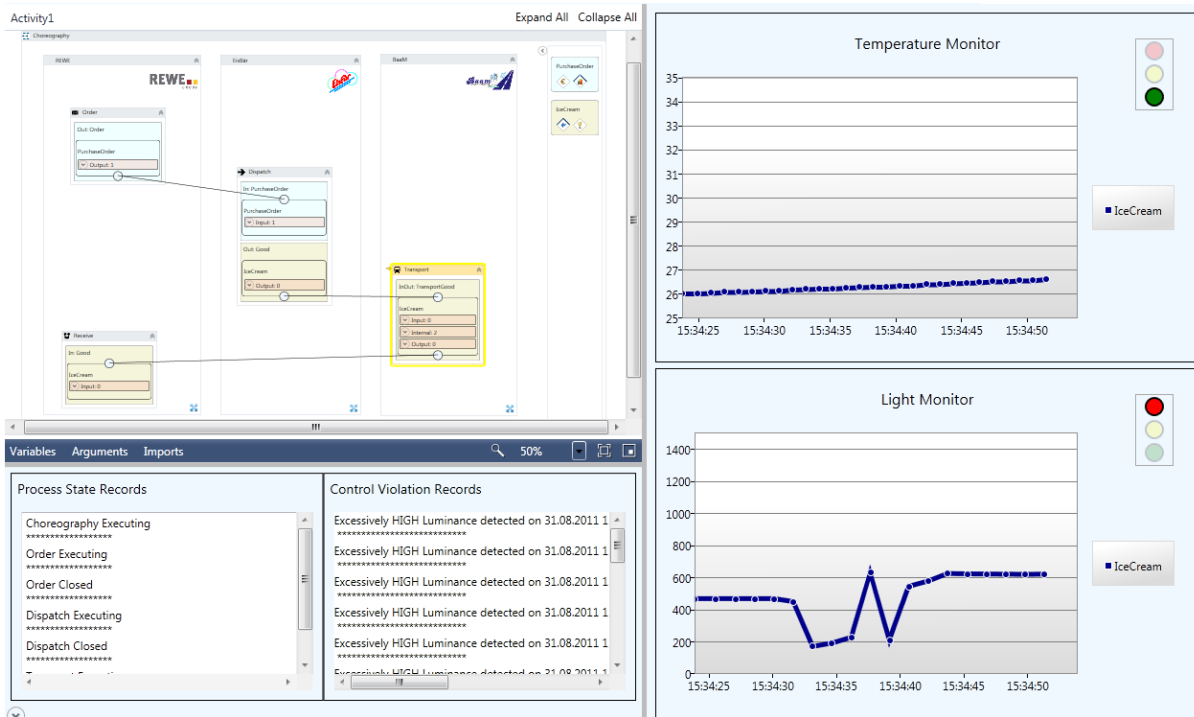


Figure 11: Execution of the supply chain process

showcased at various trade fairs and received positive feedback from the different parties involved in such supply chains. We found that even a non IT audience easily understands the visualization of security constraints (e. g., a signature symbol on a purchase order) as well as safety constraints (e. g., a temperature symbol on the purchased good).

Future work includes specification of the reactive actions that must be taken when a violation occurs. We also analyze how the specified security and safety measures can be used for contract generation between the choreography participants, as well as consider propagation of the high-level requirements on the internal processes.

Acknowledgments

The research leading to these results has received funding German “Federal Ministry of Education and Research” in the context of the project “RescueIT” and from the European Union Seventh Framework Programme (FP7/2007-2013) under grant no. 257930.

8. REFERENCES

- [1] C. Artelsmair, W. Eßmayr, P. Lang, R. Wagner, and E. Weippl. CoSMo: An approach towards conceptual security modeling. In A. Hameurlain, R. Cichetti, and R. Traummüller, editors, *Database and Expert Systems Applications (DEXA)*, volume 2453 of *Lecture Notes in Computer Science*, pages 557–566. Springer-Verlag, 2002.
- [2] D. A. Basin, J. Doser, and T. Lodderstedt. Model driven security: From UML models to access control infrastructures. *ACM Transactions on Software Engineering and Methodology*, 15(1):39–91, 2006.
- [3] R. Bobrik, T. Bauer, and M. Reichert. Proviado - personalized and configurable visualizations of business processes. In *EC-Web*, pages 61–71, 2006.
- [4] A. D. Brucker and H. Petritsch. A framework for managing and analyzing changes of security policies. In *IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY)*, pages 105–112. IEEE Computer Society, 2011.
- [5] P. Herrmann and G. Herrmann. Security requirement analysis of business processes. 6:305–335, 2006.
- [6] S. Jablonski and M. Götz. Perspective oriented business process visualization. In *Business Process Management Workshops*, pages 144–155, 2007.
- [7] J. Jürjens. *Secure Systems Development with UML*. Springer-Verlag, 2004.
- [8] A. Kader. Increasing food availability by reducing postharvest losses of fresh produce. In *V International Postharvest Symposium, International Society for Horticultural Science*, 2005.
- [9] J. Lundqvist, C. de Fraiture, and D. Molden. Saving water: From field to fork: Curbing losses and wastage in the food chain. In *SIWI Policy Brief*, 2008.
- [10] J. Mendling and J. Recker. Towards systematic usage of labels and icons in business process models. In *13th International Workshop on Exploring Modeling Methods for Systems Analysis and Design*, 2008.
- [11] G. Monakova and F. Leymann. Workflow ART. In R. Meersman, T. S. Dillon, and P. Herrero, editors, *OTM Conferences (1)*, volume 6426 of *Lecture Notes in Computer Science*, pages 376–393. Springer-Verlag, 2010.
- [12] S. Rinderle, R. Bobrik, M. Reichert, and T. Bauer. Business process visualization - use cases, challenges, solutions. In *ICEIS (3)*, pages 204–211, 2006.
- [13] A. Rodríguez, E. Fernández-Medina, and M. Piattini. A bpmn extension for the modeling of security requirements in business processes. *IEICE - Trans. Inf. Syst.*, E90-D:745–752, 2007.
- [14] 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. Secure Service-Oriented Architectures.