USING AN ENTERPRISE ARCHITECTURE FOR IT RISK MANAGEMENT

Frank Innerhofer–Oberperfler, Ruth Breu

Research Group Quality Engineering
Institute of Computer Science – University of Innsbruck
Technikerstr. 21a, 6020 – Innsbruck, Austria
{frank.innerhofer-oberperfler, ruth.breu}@uibk.ac.at

ABSTRACT

In this paper we propose a novel approach for the systematic assessment and analysis of IT related risks in organisations and projects. The approach is model-driven using an enterprise architecture as the basis for the security management process. Using an enterprise architecture it is possible to provide an integrated description of an organisation's structure, processes and its underlying IT landscape. That way we want to bridge the technical and business oriented views on information security. The proposed approach provides a detailed process of security management and defines the necessary responsibilities and roles of the participating stake-holders.

1 INTRODUCTION

The management of information security in an enterprise or even a project is a challenging endeavour. The complex interrelationships between technical, business oriented and organisational factors combined with the pace of change in todays organisations make the analysis of the enterprise information systems in general and with regards to its security a non trivial task.

The need for a more business oriented view to justify investments in information security and to report and analyse the related risks for the enterprise is recognised by a variety of authors [SS05, NS01, CM03, Car04]. To support a business oriented approach to information security management a variety of stake-holders in the enterprise have to cooperate and communicate. This poses additional problems to be solved as the various stake-holders have a differing background and view on these issues.

The problems that have to be addressed in security management can be summarised as follows:

- handling the complexity of business supporting information processes and understanding their dependencies and interrelationships,
- supporting stake holders (IT responsibles, IT security officers, management) in their cooperation and accomplishment of tasks,
- providing security relevant information at the right level of abstraction,
- establishing a continuously executed security management.

In this paper we propose a novel approach to the management of information security risks in organisations and projects. Our approach tries to utilise the advantages offered by the discipline of enterprise architecture to support an enterprise-wide holistic information security risk management. The motivation for the approach is to show if and how enterprise architecture can be used to overcome the previously discussed problems. Enterprise architecture is an approach that provides an integrated description of an organisation's structure, processes and its underlying IT landscape and reduces complexity by providing specific viewpoints on an integrated entire model. By combining the enterprise architecture with a risk driven security management process we are able to address the previously discussed problems.

The security process itself contains classic actions like requirements elicitation, threat and risk analysis and choice of appropriate countermeasures. All these actions are utilising the information contained in the enterprise architecture as input and are producing as a result changes in the architecture.

The following sections are structured as follows. Section 2 discusses related work. In section 3 our notion of Enterprise Architecture and the related meta-model is introduced. In section 4 the security relevant information that is captured by our approach is presented. The core of the paper is section 5 presenting the security analysis process. Section 6 discusses the potential of our approach for monitoring and reporting tasks and finally in section 7 a conclusion is drawn.

2 BACKGROUND AND RELATED WORK

Enterprise architecture is generally defined as a model-based management and planning instrument for the evolution of enterprise-wide information systems. A variety of authors and institutions have developed frameworks for enterprise architectures. Among the most important ones are the Zachman Framework [Zac99], the Department of Defense Architecture Framework [DoD04], the Open Group Architectural Framework (TOGAF) [The03] and Architecture of Integrated Information Systems (ARIS) [Sch00].

Most of these frameworks are used as an instrument to support planning and management of information systems in organisations. The Zachman framework for enterprise architecture has furthermore been evaluated and used for security engineering by several authors [Hen96, DeL01, LE05]. The authors emphasise the advantages of requirements traceability along the different layers of the enterprise architecture. The Open Group has also released a guide to describe how the TOGAF Architecture development can be used to create a security architecture [The05].

For our purposes we have developed a simple enterprise architecture using a meta-model-based approach for the definition of the necessary concepts and interrelationships. By using UML as a modelling language to describe the architecture artifacts in a meta-model we are able to formalise the concepts and interrelationships between elements and layers. This is particularly useful to propagate and update security relevant information throughout the architecture.

From the discipline of information security risk management a variety of approaches underline the importance of managing information security from a business point of view. These approaches are based on the concept of risk that in general determines the impact of an adverse event in terms of its likelihood and its potential damage.

An approach that recognises this fact is the OCTAVE approach [AD02]. The difference to our approach is that we employ modelling techniques to visualise the relevant assets and their dependencies using the viewpoints an enterprise architecture provides.

Another interesting concept can be found in Peltiers FRAP (Facilitated Risk Analysis Process) methodology [Pel01]. The FRAP method is identifying stake-holders and puts information assets under their ownership [Vid04]. We share the idea of involving the various stake-holders to utilise their specific knowledge.

An approach that is following a model-based risk analysis is CORAS [BDG$^+$02]. While the CORAS approach uses models mainly for descriptive purposes we employ models to systematically understand and analyse the dependencies and interrelationships of business and technical objects of an enterprise. In addition the CORAS method does not provide for reassessment and continuous risk management [BL04].

Suh and Han [SH03] use a business model to identify business functions in order to evaluate the relative importance of information assets for these functions. Suh and Han focus solely on the security requirement of operational continuity. Our approach is in contrast not restricted to a single set of requirements.

The German IT Baseline Protection Manual [BSI03] follows an approach that is very similar to our approach. The main focus of the approach is on the analysis and assessment of the IT infrastructure, although also organisational aspects are generically considered. Our approach is different in that we are using an enterprise architecture to systematically define the dependencies of business related artifacts with technical artifacts.

3 ENTERPRISE ARCHITECTURE

This section is split in two subsections: The subsection *Meta-model* describes the concepts and associations that can be modelled in the architecture. The subsection *Definitions and notions* delivers a small glossary of terms that are used throughout the rest of the paper to describe the enterprise architecture artifacts on an abstract level.

3.1 Meta-model

Figure 1 depicts the meta-model of our enterprise architecture. In the Enterprise Architecture Meta-model we specify the relevant concepts of an enterprise together with their interrelationships. The Enterprise Meta-model is divided in four main layers focusing on different levels of abstraction: *business*, the *application* layer, the *technical* layer and the *physical* layer. The different layers are interconnected by the associations of the meta-model that cross the layer boundaries.

Furthermore it is possible to provide the various stake-holders with different views on the enterprise architecture, that show only specific types of artifacts. For example by just visualising organisational

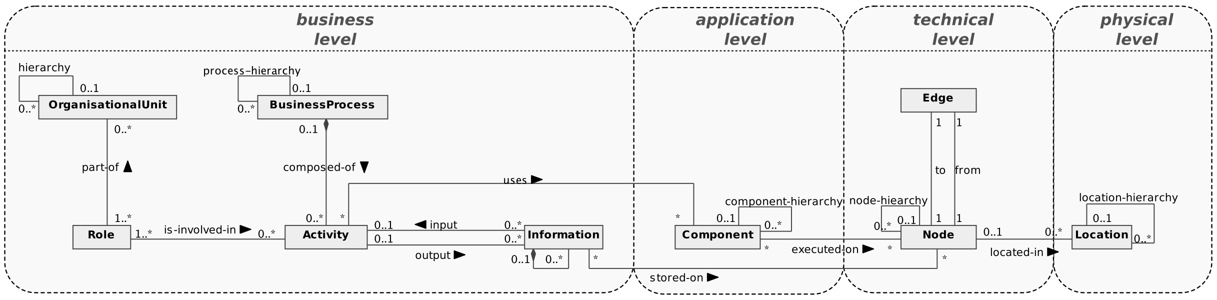

Figure 1: Enterprise Architecture Meta-model

units and roles we can provide an organisational view. By combining the views of the application layer and the technical layer we can visualise the dependencies of information systems on the IT infrastructure.

In the following the concepts of the enterprise architecture meta-model will be briefly discussed:

Business Layer The business layer contains business oriented artifacts like organisational units, roles, business processes, activities and information objects. An *Organisational Unit* represents the enterprise as a whole, its business units and organisational departments. Due to the possibility of hierarchical structuring a fine grained modelling of the organisational entities is possible.

Associated with an organisational unit is the concept of *Role*. A Role is involved in executing the actions of business processes.

The concepts *Business Process* is used to describe the main business processes. To allow a more detailed modelling the concept can be hierarchically structured to model sub-processes. A business process is not directly linked with an organisational unit but indirectly through the attribution of roles to specific activities of a business process. A business process is composed of either sub-processes or at the lowest level of a set of activities that describe the dynamic behaviour.

The concept *Activity* plays a central role in our architecture as it connects the organisational artifacts with the concept information. An activity may require information as input and can produce new information objects or information objects with a changed state as output. Furthermore an activity can be supported by a component (e.g. an information system).

Finally the concept *Information* is used to model information objects or artifacts (e.g. credit card information, order information). The concept information can be hierarchically composed to describe the different parts of an information object. Information is processed by activities through the use of information systems, may be stored on nodes and is transferred between nodes via edges (e.g. a network connection).

Application Layer On the application layer the information systems and its components are modelled. An *component* can be a application that supports the execution of an activity and is typically supported by one or more nodes (e.g. a dedicated server or workstation).

Technical Layer On the technical layer the physical hardware and communication systems are modelled. A *Node* therefore represents technical or physical objects that are either used to store information objects (e.g. file server, usb stick, plain paper) or are supporting the execution of a component. The concept node may be hierarchically composed to provide a more detailed model of a specific node. That way the critical technical components of a server can be modelled (e.g. hard drives, network adaptors).

An *Edge* is a connector between different nodes. It may be used for transmitting information objects from one node to another. Edges can be communication links like LANs, WLANs, the Internet and also classic communication lines like the telephone cable.

Physical Layer The physical layer serves the purpose of modelling the different physical locations of the enterprise. The concept *Location* can be hierarchically structured to distinguish different sites of an organisation and the buildings, floors and rooms at the more detailed level. A location can host specific nodes.

At the instance level the enterprise architecture is modelled using a set of UML diagrams and some additional own shapes to depict the various elements, Figure 2 gives an example of a simple enterprise architecture.

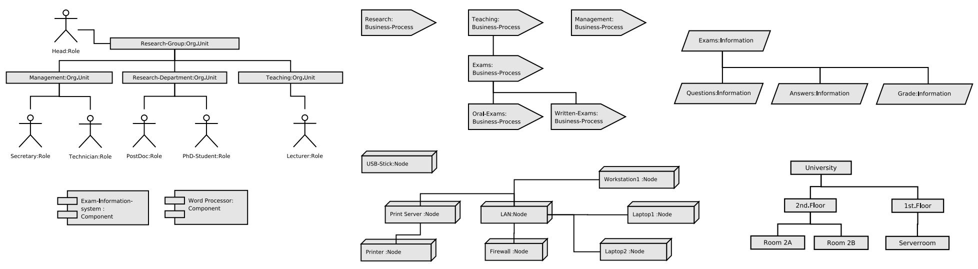

Figure 2: Schematic Example of an Enterprise Architecture

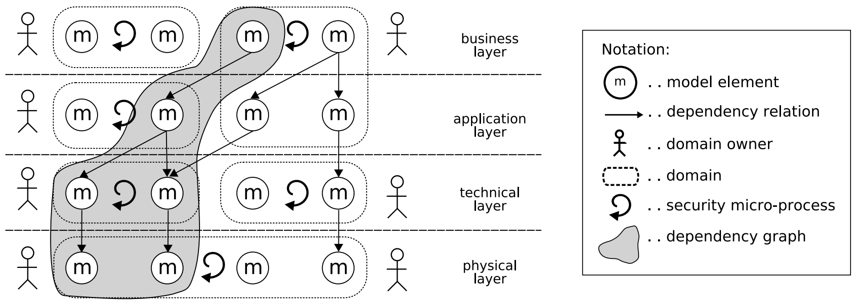

Figure 3: Abstract definitions and concepts

3.2 Definitions and notions

In the following sections we abstract from the concepts and interrelationships of the Enterprise Architecture and use the following notions that are depicted in Figure 3. We talk of

- model elements m
- a dependency relation between model elements

$$m_1 \to m_2$$

  if m_1 and m_2 are linked by any interrelationship of the meta model.

- For any dependency $m_1 \to m_2$ we assume

$$layer(m_1) \geq layer(m_2)$$

  where the abstraction layer $layer(m)$ of some model element is either *business*, *application*, *technical* or *physical*.

- a parent-child relation between model elements

$$m_p \rhd m_c$$

  if m_p and m_c are of the same type defined in the meta-model and a hierarchical association is defined.

- a *dependency graph* as the set of all model elements that are linked by a dependency relation and/or by a parent-child relation. Dependency graphs may intersect or overlap each other.
- a *root element* as the model element that serves as the root node to identify a dependency graph in a top-down-approach.
- a *domain* as the set of model elements that are put under the responsibility of one stake holder. The whole enterprise architecture is divided in domains that are non-overlapping.
- a *domain-owner* as any stake holder who is actively engaged in the security management process and has the responsibility for a domain of the enterprise architecture.
- a *security micro-process* as the actions of the entire security management process that are carried out by the respective domain-owners in the context of their domain.
- a *view* as a specific perspective on the enterprise architecture that shows all model elements of one or more defined types including only the dependency and parent-child relations of the selected elements.

4 SECURITY INFORMATION META-MODEL

The enterprise architecture that is described in the previous sections allows to model the necessary elements and their dependencies to describe the universe of discourse for the security management process. However that information is not sufficient for analysing the significance of the model elements regarding information security risks. For that purpose we have extended the enterprise architecture with security relevant information that reflects the status of the entire security process and that connects the model elements with security artifacts such as threats, requirements, risks and countermeasures.

Information security is generally defined by properties like confidentiality, integrity, availability, non-repudiation and authenticity of information [ISO05, Pel01, AD02]. In our model all basic security concepts like security objectives, security requirements, threats, risks and countermeasures are bound to model elements, thus enabling systematic tracing along dependencies.

The basic concepts of security information are depicted in Figure 4 and informally described below. The explanation of the status attributes and the rules for updating them follow in the section "Security Management Process".

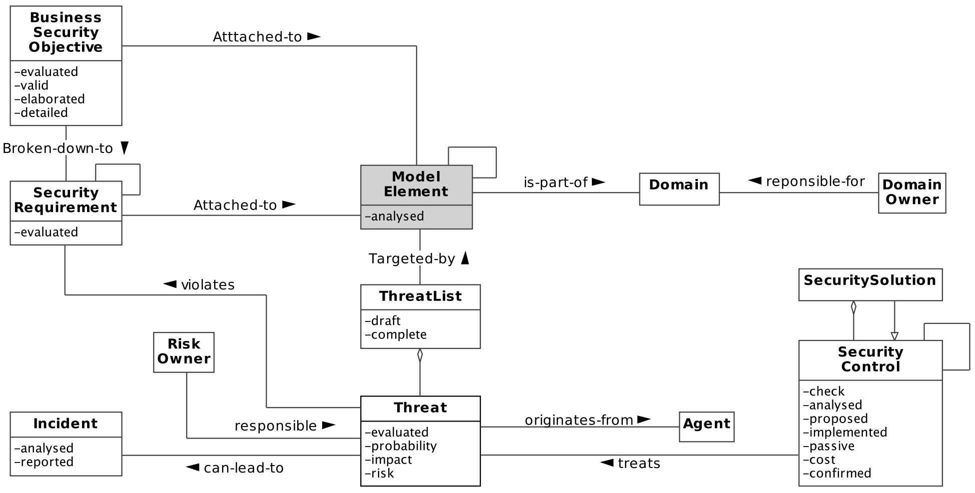

Figure 4: Security Information Meta-model

Model Element The central element of the security information meta-model is the concept model element. Model element acts as a place-holder for all elements of any type of the enterprise architecture meta-model, and builds therefore the bridge between the two meta-models. A model element is part of a defined **Domain** that is attributed to a specific stake-holder called **Domain Owner**.

Business Security Objective The concept Business Security Objective (BSO) is used to define an abstract high level definition of one goal of the security management effort. A BSO is attached to a specific model element, typically to one of the business layer. A BSO serves the purpose of communicating a goal for all stake-holders. A BSO furthermore defines the root node of a new dependency graph that is identified by tracing all dependencies of related elements of the enterprise architecture (see Figure 5, a). The BSO has significance for all elements of the dependency graph.

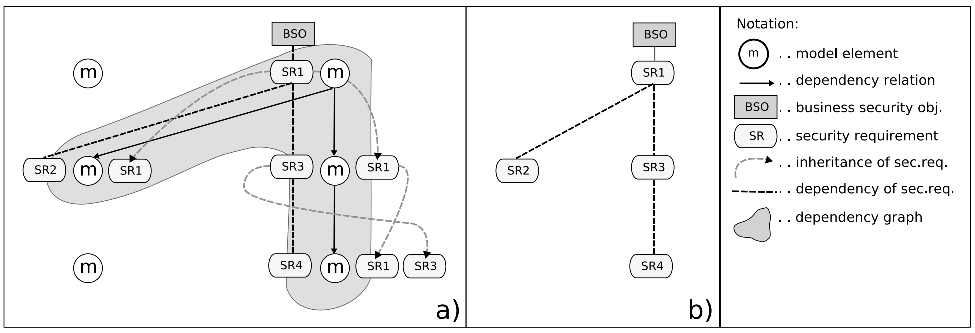

Figure 5: Dependencies of Security Requirements

Security Requirement A Security Requirement is a more detailed and concrete formulation of a BSO. The domain owners use security requirements to describe concrete subgoals of a BSO with regard to specific dependent model elements, reflecting their context and type. A Security Requirement must be linked with either the main Business Security Objective of a dependency graph or it has to be derived from another Security Requirement of an upper layer model element. Once a Security Requirement is defined it is inherited by all dependent model elements of the lower layers. That way we obtain a tree of Security Requirements that have a Business Security Objective as its root node (see Figure 5, b).

Threat The concept threat describes any event in the context of a specific model element, that can cause a violation of the defined and inherited security requirements. A threat is always related with a specific model element. A threat is evaluated measuring its probability and potential impact resulting in a measurement of its risk. A threat is also linked to the security requirements that it can violate. That way we can analyse what threats are targeting a specific security requirement. Furthermore it is possible to identify all threats that are related to a specific Business Security Objective. An identified threat can be put under the responsibility of a **Risk Owner**, who will typically be the domain owner of the related model element. Furthermore a threat is originating from a specific **Agent** (Nature, Systems, Human).

ThreatList A Threat List is a list that contains all the identified threats that are targeting one specific model element.

Incident An incident is the manifestation of an adverse event leading to a violation of security requirements. The modelling of an incident serves the purpose to analyse the causes of the event and to check if any of the security controls failed or if the related risk was properly identified.

Security Control A Security Control is any measure that addresses the identified risks to model elements and security requirements. That includes any controls that avoid, minimise, transfer or retain specific risks. The treatment can be of technical or organisational nature. Security Controls can be hierarchically structured in a tree and can be combined to deliver a security solution.

SecuritySolution A Security Solution is a bundle of atomic security controls that have to be implemented as a whole. The concept is needed to evaluate alternative bundles of countermeasures.

5 SECURITY MANAGEMENT PROCESS

The security management process is composed of the classical actions of security analysis: the elicitation of security requirements, the analysis of potential threats, the evaluation of risks and the definition of appropriate measures to mitigate the risks. The speciality of our approach is the execution of these actions on the basis of the modelled enterprise architecture. In the following subsections the single actions of the security management process will be described. Emphasis is put on clarifying how the enterprise architecture supports each of these action by providing the necessary context and how every action may itself result in changes of the enterprise architecture. Finally a simple example is provided alongside each action.

One requirement that our approach has to satisfy is supporting a continuous risk management process. Therefore it is not sufficient to provide a method and tool for analysing the security state at a specific point in time, but to allow the update and adaption of the enterprise architecture to reflect organisational changes and maintain an actual model. To achieve this requirement our security management process is continuously reiterated. A number of triggers that observe the enterprise architecture elements can cause a reiteration of the security management process. These triggers are described in the subsection 'Reiteration of the security management process'.

In our approach the security management process is realized by a set of concurrently executed *security micro-processes*, that are executed by the domain owners for their respective domains (see Figure 3). That way we are able to decrease the complexity of the overall security management process by reducing the scope of analysis for every stake holder, while maintaining an integrated model by using the enterprise architecture as a basis.

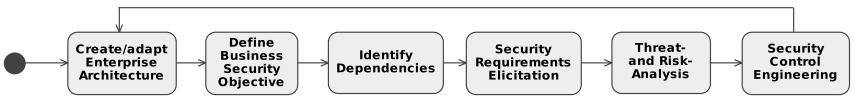
Create/adapt Enterprise Architecture
Define Business Security Objective
Identify Dependencies
Security Requirements Elicitation
Threat- and Risk-Analysis
Security Control Engineering

Figure 6: Activities of the Security Management Process

The structure of this section is as follows. The following subsections 5.1 to 5.5 are devoted to the actions of the security management process (see Figure 6). Finally in subsection 5.6 triggers causing a reiteration of the whole process and their relation to the status attributes are discussed.

5.1 Create or adapt Enterprise Model

The first step of the security management process is the creation of a simple and basic enterprise architecture that builds the starting point for the following steps. For instance, the main organisational units and business processes may be defined or the most important building blocks of the IT infrastructure are modelled. Additional elements required for more detailed modelling and analysis purposes may be added later at any time. During the first or initial iteration cycles, it is sufficient to just model a small set of elements in the enterprise architecture and extend and enrich the architecture later on.

Initially the different elements defined serve the purpose of describing the static part of the enterprise architecture. The elements are not yet or only partially linked with each other and provide a small repository of elements needed to define the dynamic behaviour of the enterprise architecture (i.e. the activities of a business process and the interrelation of various elements working together). In the case of a reiteration cycle the enterprise architecture may be extended or adapted to reflect changes that occurred in the organisation.

Status changes. If a new element is added to the enterprise model its status is set to $analysed = false$.

Example. In the case study that is explained along each of the single actions we show the analysis of a specific business process of our research group. During this first step of creating the enterprise architecture we define the organisational units, the main business processes, important information objects and a simple model of the IT infrastructure and the locations of the organisation (see Figure 2).

5.2 Define Business Security Objective

The next step after having created a simple enterprise architecture is the definition of a high level Business Security Objective by the Chief Security Officer. Business Security Objective are the tactical goals, that the enterprise decides to reach. Security Objectives can be derived from a variety of sources: they can stem from legal requirements, from contractual or statutory obligations the enterprise assumed or are defined in the internal security policy.

The definition of a business security objective defines a separate scope of analysis for the security management process. This security scope is identified by tracing all the dependencies from the model element for which the security objective was defined. At this point the enterprise architecture is not yet linked throughout the layers and therefore the dependency graph may only contain the model element that is labelled with the security objective. However, the definition of a security objective requires the identification of dependencies throughout the layers of the architecture. One can say, that the security objective directs the more detailed modelling efforts to those areas of the enterprise architecture that are relevant for achieving the security goals.

Status changes. If a security objective is initially added to a model element of an enterprise architecture it has the states $evaluated = false$, $elaborated = false$ and $valid = false$ and $defined = false$. If the Chief Security Officer declares a security objective as valid for the security management its state is changed to $valid = true$.

Example. In our case study we identify the general Security Objective “Compliance with legal regulations regarding the exam process” to the business process “Exams”.

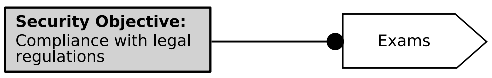

Figure 7: Sample Business Security Objective

5.3 Identify Dependencies

To achieve an integrated view of the different models of the layers of the enterprise architecture the business processes and the lifecycle of the used information objects are modelled in detail. For this purpose the single activities of a business process are modelled, describing the control flow and the information flow. That way we are able to define dependencies across the boundaries of the architectural layers.

The resulting business process model makes it possible to analyse what model elements are used to create, store, modify, delete, transfer information through the enterprise. The dependency graph is used to propagate security relevant attributes like security objectives, requirements and threats throughout the architecture.

The starting point for modelling the dynamic behaviour of a business process is the definition of a Business Security Objective for a specific model element. The respective domain owner begins to model the activities of the business process and by connecting the various model elements he identifies other domain owners that are then cooperating in the detailed modelling effort. It may be necessary to extend the enterprise architecture with additional model elements that have not been identified in the initial creation of the architecture.

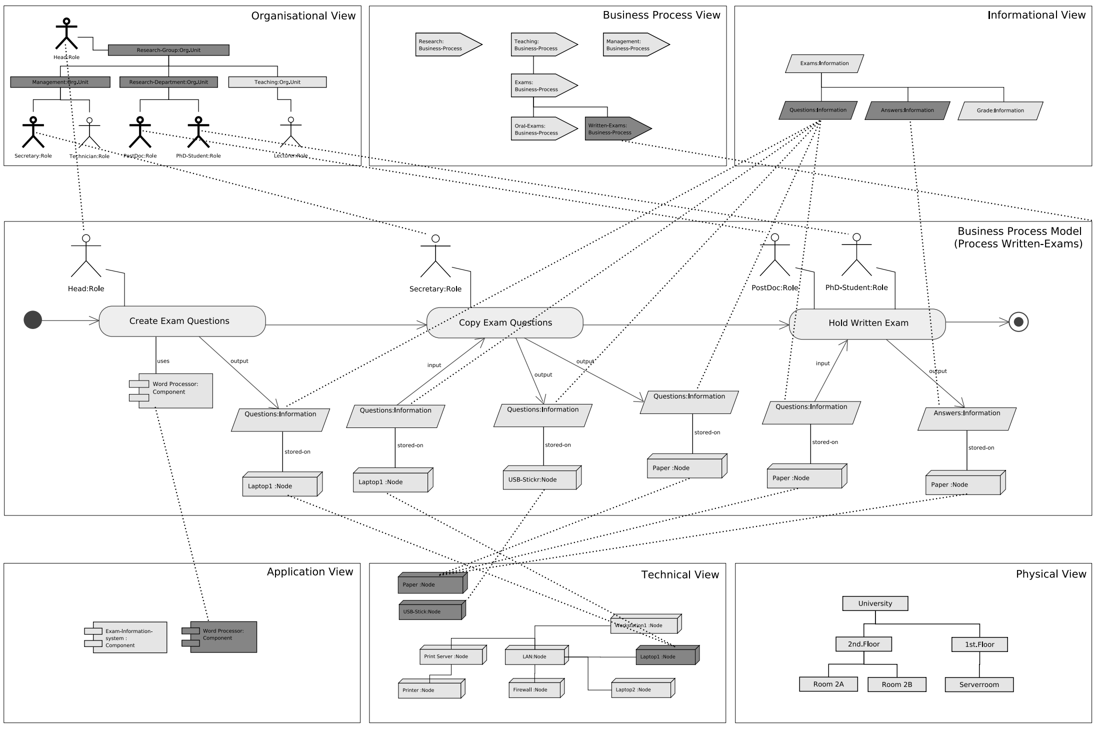

Figure 8: Identification of dependencies

Status changes. If the activity of identifying dependencies is concluded, that status of the Business Security Objective that identifies the dependency graph is set to $defined = true$. All the model elements of the dependency graph have their status reset to $analysed = false$ to indicate, that they have yet to be analysed and assessed with regard to the defined security objective.

Example. In our example we show a simplified example of a model of the dynamic behaviour of the sub-process "Written-Exams" of the business process "Exam". The modelled activities are "Create Exam Questions", "Copy Exam Questions" and "Hold Exam". Figure 8 depicts the identified associations by connecting the model elements with dotted lines.

5.3.1 Requirements Elicitation

After the definition of a dependency graph by creating a model of the business process the Business Security Objective is further elaborated and translated in concrete requirements for the interconnected model elements. The elicitation of requirements, that have to be fulfilled to achieve the business security objective is conducted by the various domain-owners. Security Requirements Engineering is done in a top-down way starting with the root element of the dependency graph which has the Business Security Objective attached. Depending on the context and type of the model elements the security requirements are formulated.

Defined Security Requirements are inherited by lower layer elements of the dependency graph. One can say, that the defined requirements are propagated like tokens throughout the dependency graph (see Figure 5). A security requirement has to be associated with a parent security requirement or with the root security objective. That way we obtain a tree of security requirements that allow the traceability of all requirements to their parent objectives.

Status changes. If a security requirement is defined its initial state is set to $evaluated = false$. The status of all model elements that have a new security requirement attached or inherited is set to $analysed = false$. Once the elicitation of security requirements is concluded for a dependency graph the state of the root security objective is set to $elaborated = true$, indicating that all dependent security requirements have been formulated.

Example. In the example the Business Security Objective is translated into a concrete security requirement “Exam Questions must remain confidential before exam” that is attached to the model element “Written Exams”. As can be seen in Figure 9, a) the Security Requirement is propagated along the dependency relations to the related model elements. For the sake of simplicity in the example we focus only on a small section of the dependency graph.

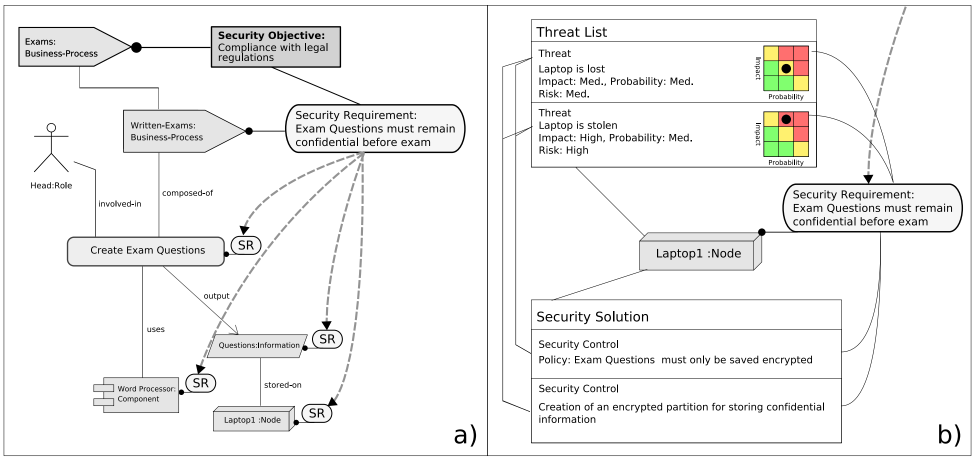

Figure 9: Schematic Example of requirements, threats and security controls

5.4 Risk and Threat analysis

The model elements of a dependency graph with clearly stated security requirements is now undergoing an analysis of possible events that can violate the security requirements. For this purpose existing security checklists or standards like the Baseline Protection Manual[BSI03] or EBIOS[DCS05] can be used. Similarly to security requirements every identified possible adverse event is linked to a model element of the dependency graph. Furthermore every threat must be related to the security requirements it may potentially violate. The set of identified threats element is contained in a threat list for the respective model element. Since every threat is associated with a model element and a security requirement we are able to trace every threat to a specific security objective.

At the moment we support only a qualitative assessment of threats by determining the impact and the probability using the classical (*high*, *medium*, *low*) rankings, that results in a 3x3 matrix (see Figure 9, b). We are aware of the shortcomings of a qualitative assessment [LV04] approach and are planning to extend our approach to include a quantitative approach as part of our future research.

Status changes. If the first possible adverse event is identified for a specific model element a threat list is created with the status $draft = true$ and $complete = false$ and the status of the threat is set to $evaluated = false$ and its attributes *probability*, *impact* and *risk* are empty. After the assessment of a threat by indicating its probability and impact we can calculate the associated risk and the threats status is set to $evaluated = true$. If the identification of threats for a model is concluded, the status of the threat list is set to $complete = true$. If all the threats of a completed threat list of a model element are evaluated the status of the model element is set to $analysed = true$. If all model elements that are subject to the same security requirement have the status $analysed = true$, then the state of the security requirement is set to $evaluated = true$. If all security requirements dependent on a security objective have reached the status $evaluated = true$, then the security objective’s status is also set to $evaluated = true$.

Example. As an example we show in Figure 9, b) how the model element “Laptop1” has an attached threat list that could potentially violate the inherited security requirement. The threats are evaluated regarding their risk and are linked with the targeted model element and the related security requirement.

5.5 Security Controls Engineering

The last step of the security management process is the definition of appropriate countermeasures to tackle the identified risks. We call the variety of countermeasures that are used to mitigate, transfer or avoid the security related risks security controls. Security controls can range from technical solutions to organisational measures. The possible security controls can be derived from best-practices or security checklists and have to be evaluated to determine their contribution to the reduction of the overall risk.

It is clear that the introduction of countermeasures can introduce new risks that require a further analysis and assessment. Countermeasures may also introduce new model elements in the enterprise architecture and alter existing or create new dependencies. To assess the effects of new countermeasures a snapshot of the current enterprise architecture can be created that is used to model the possible architectural changes introduced by the new countermeasure.

The first step is the proposal of a new security solution. A security solution is composed of several atomic security controls. Every atomic security control is directly related with the threats that are addressed. The proposed solution is modelled as a scenario that does not alter the existing enterprise architecture but operates on a duplicated temporary model of the enterprise architecture that can be used to derive the delta of architectural changes introduced by the countermeasure.

It is possible to define several different security solutions that tackle the same threats to provide alternatives for the decision making process. For this purpose the cost of the security controls has to be estimated. If a security solution is approved for implementation then the changes modelled in the scenario are applied to the enterprise architecture. This requires checking for potential conflicts if different countermeasure scenarios operate on the same model parts. These conflicts have to be solved in a coordinated effort by the concerned stake-holders.

Status changes. If a security solution is proposed its state is set to $proposed = true$, $passive = true$ and $check = true$. After the addition of atomic security controls a scenario describing the changes to the enterprise architecture and including an estimation of the associated costs the security is created. At this point the security solution reaches the state $analysed = true$ and $check = false$. If management decides to implement a specific security solution its state is set to $confirmed = true$. After the successful roll-out or deployment of the security the state of a security solution and all its atomic security controls is set to $implemented = true$. In the case of suspension or deactivation of a security solution its state is set to *passive*. If a reassessment of the security solution is scheduled due to the suffering of an incident or due to periodic checks the status attribute *check* is set to *true*, indicating that the domain-owner has to analyse again the security solution and its effects on the enterprise architecture.

Example. In Figure 9, b) a Security Solution containing two security controls is attached to the model element "Laptop1". The possible countermeasures include the encrypted storage of the confidential exam questions, that are stored on the laptop. The security solution and the security controls are connected with the model element, the security requirement and the threats that are handled.

5.6 Reiteration of the security management process

A variety of triggers can cause a reiteration of the security management process. One particularly important trigger are changes in the enterprise architecture. If a *new model element is added* and is connected with an existing dependency graph it automatically inherits all security requirements from upper elements connected by dependency relations. In this case the state of the related security requirements is set to $evaluted = false$ to indicate that not all elements have been assessed regarding their risk. In addition the root security objective's state is also set to $evaluated = false$. These status changes require the respective domain owner to accomplish the relevant tasks of the security process to ensure that the new model element is also considered in the risk analysis. The same situation occurs when a *new dependency relation is inserted* in the enterprise architecture. An existing dependency graph might be extended resulting in the propagation of its security requirements to new newly included model elements. Also in this case the related security requirements and security objective are reset to $evaluted = false$ requiring a new assessment cycle to include the new elements.

If a model element or dependency relation is deleted from the enterprise architecture already implemented security solutions might have become useless and could potentially constitute an overinvestment in security. Before a model element can be deleted the respective domain owner has to resolve all possible conflicts in his own or other domains. A successful deletion can reduce the dependency graph and therefore might reduce the scope for which a security objective is valid. Already implemented security solutions that are set to the state $passive = true$ and $check = true$ to reassess if they are still required.

Other triggers are the definition of a new security objective, a new security requirement, new threats and new security controls. For the related status changes we refer to the subsections that describe the activities of the security management process.

Therefore the status changes of the security related information reflect the status of the overall security management process. Reports and summaries of the status attributes can be used to govern the security management process and to define the task that have to be accomplished by the domain owners.

6 REPORTING AND MONITORING

Our approach offers various starting point for the provision of reports that can be used by the chief security officer and the domain owners to control the overall progress of the security management process.

One source for generating reports are the status attributes of the security related information. Queries that might be useful for the domain owners include calculating the number of model elements with the status $analysed = false$ to show to what extent a dependency graph has not yet been fully analysed. A similar result could be delivered by a query that counts all the security requirements of a domain that have a status of $evaluated = false$. A query of all security objectives with the state $detailed = false$ indicates which areas of the enterprise architecture require a detailed modelling to identify a dependency graph.

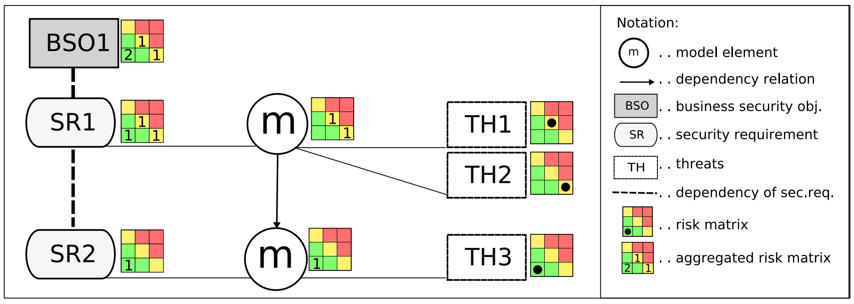

Figure 10: Aggregation of qualitative risk assessments

Another source for generating reports are the qualitative risk assessments of the identified threats. By querying the enterprise architecture for its risk ratings several combined risk reports can be generated that represent an accumulated view on different levels of abstraction and from different viewpoints. Figure 10 shows a simple example on which levels aggregated risk matrixes can be provided. An aggregated risk matrix just counts the risk ratings of the queried threats and shows the number of threats in a combined matrix.

It is therefore possible to show an aggregated risk matrix for one specific model element (aggregating all its threats), for a security requirement (aggregating all its related model elements), for a security objective (counting all ratings of a dependency graph) and finally for the entire organisation (aggregating the ratings of all security objectives).

7 CONCLUSIONS AND OUTLOOK

In this paper we discussed an approach that combines the disciplines of enterprise architecture and risk driven security management to allow a systematic elicitation and analysis of information security in an organisation. By modelling dependency graphs the relevant business and technical objects of the enterprise are interrelated and can be analysed as a separate scope.

We are able to decrease the complexity of the overall security management process by utilising the advantages offered through modelling the relevant artifacts with an enterprise architecture. By presenting each stake-holder with an appropriate view and by having defined a formal underlying meta-model we can ensure that the integrity of the overall model is maintained even if several domain-owners are working concurrently on their individual sub-domains.

The underlying enterprise architecture is the basis for the execution of the security management process and allows the traceability of dependencies and the propagation of security relevant information.

By capturing the progress and state of the security management process using status attributes in the enterprise architecture we can furthermore determine the relevant tasks of the security management that have to be accomplished. We have shown ideas of how the status attributes can be used to report the overall progress and state of security management and how the traceability of security relevant information can be used to provide aggregated risk reports for the stake-holders.

We are currently working on the implementation of a management tool that supports the outlined approach to security management. The tool is implemented as a plug-in on the basis on the open source Eclipse Framework (http://www.eclipse.org/). The prototype of the tool will be evaluated with industrial partners to provide input for improvements and to determine if the related effort of employing our approach is effective from a cost-benefit view. Future plans include the extension of the approach to provide a quantitative measurement method for calculating the risks.

References

[AD02] Christopher J. Alberts and Audrey J. Dorofee. *Managing information security risks: the OCTAVE approach.* Pearson Education, 2002.

[BDG+02] Folker den Braber, Theo Dimitrakos, Bjørn Axel Gran, Ketil Stølen, and Jan Øyvind Aagedal. Model–based risk management using UML and UP. In *Proc. Information Resources Management Association International Conference (IRMA'2002)*, pages 925–927, 2002.

[BL04] W. G. Bornman and L. Labuschagne. A Comparative Framework for Evaluating Information Security Risk Management Methods. In *Proc. ISSA 2004*, June 2004.

[BSI03] BSI (Federal Office for Information Security). IT Baseline Protection Manual, 2003. available online: http://www.bsi.bund.de/english/gshb/manual/index.htm.

[Car04] Richard A. Caralli. Managing for Enterprise Security. Technical Note CMU/SEI-2004-TN-046, Carnegie Mellon University, Software Engineering Institute, 2004.

[CM03] Robert S. Coles and Rolf Moulton. Operationalizing IT Risk Management. *Computers & Security*, 22(6):487–493, September 2003.

[DCS05] DCSSI. EBIOS: Expression des Besoins et Identification des Objectifs de Securite. URL: http://ebios.cases-cc.org/, June 2005. Version 2.0.

[DeL01] Lori L. DeLooze. Applying Security to an Enterprise using the Zachman Framework, Sep 2001. SANS Institute.

[DoD04] DoD Architecture Freamework Working Group. DoD Architecture Framework, Feb 2004.

[Hen96] R. R. Henning. Use of the Zachman Architecture for Security Engineering. In *Proc. 19th NIST-NCSC National Information Systems Security Conference*, pages 398–409, 1996.

[ISO05] ISO (International Organization for Standardization). ISO/IEC 17799:2005 Information technology – Code of practice for information security management, 2005.

[LE05] R. Sudarsanam L. Ertaul. Security Planning Using Zachman Framework for Enterprises. In *Proceedings of EURO mGOV 2005*, July 2005.

[LV04] Arjen Lenstra and Tim Voss. Information Security Risk Assessment, Aggregation, and Mitigation. In *ACISP: Information Security and Privacy: Australasian Conference*, 2004.

[NS01] Matunda Nyanchama and Paul Sop. Enterprise Security Management: Managing Complexity. *Information Systems Security*, 9(6):37–44, January 2001.

[Pel01] Thomas R. Peltier. *Information security risk analysis.* Auerbach, 2001.

[Sch00] A. W. Scheer. *ARIS: Business Process Modelling.* Springer-Verlag, Berlin, 2000.

[SH03] Bomil Suh and Ingoo Han. The IS risk analysis based on a business model. *Information & Management*, 41(2):149–158, December 2003.

[SS05] Basie von Solms and Rossouw von Solms. From information security to...business security? *Computers & Security*, 24(4):271–273, June 2005.

[The03] The Open Group. TOGAF (The Open Group Architectural Framework). http://www.opengroup.org/architecture/togaf/, 2003. Version 8.1, Enterprise Edition.

[The05] The Open Group. Guide to Security Architecture in TOGAF ADM, Nov. 2005.

[Vid04] Stilianos Vidalis. A Critical Discussion of Risk and Threat Analysis Methods and Methodologies. Technical Report CS-04-03, School of Computing, University of Glamorgan, Pontypridd, CF37 1DL, Wales, UK, July 2004.

[Zac99] John A. Zachman. A Framework for Information Systems Architecture. *IBM Systems Journal*, 38(2/3):454–470, 1999.