Ontology-based Security Modeling in ArchiMate

Ítalo Oliveira[1,2*], Tiago Prince Sales[2], João Paulo A. Almeida[3], Riccardo Baratella[4], Mattia Fumagalli[1] and Giancarlo Guizzardi[2]

[1*]Conceptual and Cognitive Modeling Research Group (CORE), Free University of Bozen-Bolzano, Bolzano, Italy.
[2]Semantics, Cybersecurity & Services Group, University of Twente, Enschede, Netherlands.
[3]Ontology & Conceptual Modeling Research Group (NEMO), Federal University of Espírito Santo, Vitória, Brazil.
[4]Dipartimento di Antichità, Filosofia, Storia (DAFIST), University of Genoa, Genoa, Italy.

*Corresponding author(s). E-mail(s): idasilvaoliveira@unibz.it;
Contributing authors: t.princesales@utwente.nl; jpalmeida@ieee.org;
riccardo.baratella@edu.unige.it; mfumagalli@unibz.it; g.guizzardi@utwente.nl;

Abstract

Enterprise Risk Management involves the process of identification, evaluation, treatment, and communication regarding risks throughout the enterprise. To support the tasks associated with this process, several frameworks and modeling languages have been proposed, such as the *Risk and Security Overlay* (RSO) of ArchiMate. An ontological investigation of this artifact would reveal its adequacy, capabilities, and limitations w.r.t. the domain of risk and security. Based on that, a language redesign can be proposed as a refinement. Such analysis and redesign have been executed for the risk elements of the RSO grounded in the *Common Ontology of Value and Risk*. The next step along this line of research is to address the following research problems: what would be the outcome of an ontological analysis of security-related elements of the RSO? That is, can we identify other semantic deficiencies in the RSO through an ontological analysis? Once such an analysis is provided, can we redesign the security elements of the RSO accordingly, in order to produce an improved artifact? Here, with the aid of the *Reference Ontology for Security Engineering* (ROSE) and the ontological theory of prevention behind it, we address the remaining gap by proceeding with an *ontological analysis* of the security-related constructs of the RSO. The outcome of this assessment is an ontology-based redesign of the ArchiMate language regarding security modeling. In a nutshell, we report the following contributions: (1) an ontological analysis of the RSO that identifies six limitations concerning security modeling; (2) because of the key role of the notion of prevention in security modeling, the introduction of the ontological theory of prevention in ArchiMate; (3) a well-founded redesign of security elements of ArchiMate; (4) ontology-based security modeling patterns that are logical consequences of our proposal of redesign due to its underlying ontology of security. As a form of evaluation, we show that our proposal can describe risk treatment options, according to ISO 31000. Finally, besides presenting multiple examples, we proceed with a real-world illustrative application taken from the cybersecurity domain.

Keywords: Security Modeling, Ontological Analysis, Ontological Patterns, Enterprise Architecture, ArchiMate, Reference Ontology for Security Engineering, Unified Foundational Ontology

1 Introduction

Enterprise architecture refers to principles, methods, and models that are used in the design and implementation of an enterprise's organizational structure, business processes, information systems, and infrastructure [1]. Risks are pervasive throughout the activities of any enterprise, so it is important to create security mechanisms to control those that are particularly threatening to an organization's objectives. Enterprise risk management deals with the process of identification, evaluation, treatment, and communication regarding these risks, as described by ISO 31000, an international standard for risk management [2]. The TOGAF Series Guide to "Integrating Risk and Security within a TOGAF Enterprise Architecture" [3] states that the Security Architecture is a cross-cutting matter, ubiquitous throughout the entire Enterprise Architecture. It is understood as a coherent collection of views, viewpoints, and artifacts, including security, privacy, and operational risk perspectives, along with related topics like security objectives and security services. The Security Architecture affects and informs the Business, Data, Application, and Technology Architectures [3]. Because of that, Enterprise Risk Management has, naturally, become a key aspect of Enterprise Architecture, as seen by the *Risk and Security Overlay* (RSO) of ArchiMate [4], an attempt to introduce risk and security concepts into the ArchiMate language—the Open Group's conceptual modeling language for Enterprise Architecture [5].

Though the RSO is based on risk and security frameworks (COSO, ISO, TOGAF, and SABES) [4], it has already been shown to have some limitations concerning its conceptualization of risk concepts [6], including ambiguity and missing modeling elements that negatively impact its capabilities to support enterprise risk and security modeling. Through an ontological analysis founded upon the *Unified Foundational Ontology* (UFO) [7] and the *Common Ontology of Value and Risk* (COVER) [8], earlier work has revealed, for example, the presence of *construct overload* on the Vulnerability construct, which collapses actual vulnerabilities with assessments about them, and the presence of *construct deficit* in the representation of Threat Capabilities [6]. Based on the results of this analysis, an ontologically well-founded redesign of RSO was proposed to overcome the identified problems in the risk-related elements [6].

Given this literature, the natural next step along this line of research is to address the following research problems: what would be the outcome of an ontological analysis of security-related elements of the RSO? That is, can we identify other semantic deficiencies in the RSO through an ontological analysis? Once such an analysis is provided, can we redesign the security elements of the RSO accordingly, in order to produce an improved artifact?

Here, by employing a similar methodology of ontological analysis (tracing back to [7, 9]), we investigate the modeling capabilities of the *security* elements of RSO: namely, the concepts of Control Objective, Security Requirement, Security Principle, Control Measure, and Implemented Control Measure [10]. Our analysis is grounded in the *Reference Ontology for Security Engineering* (ROSE) [11], which is a UFO-based core ontology for safety and security; particularly, ROSE provides an elucidation of the notion of security mechanism. Then, based on this ontological analysis, we propose a redesign of the concerned language fragment, taking advantage of the improved risk-related elements by the previous work [6].

In addition to that, we advance this proposal even further by showing several ontology-based security modeling patterns that are logically implied by ROSE and embedded into our redesign of ArchiMate. These modeling patterns can be useful for modeling concrete scenarios in Enterprise Risk Management since they serve as blueprints for risk treatment options, that is, they describe possible ways of executing risk treatment in a general fashion. In a nutshell, we report the following contributions:

1. An ontological analysis of the RSO that identifies six limitations concerning security modeling;
2. because of the key role of the notion of prevention in security modeling, the introduction of the ontological theory of prevention in ArchiMate;
3. a well-founded redesign of security elements of ArchiMate;

4. ontology-based security modeling patterns that are logical consequences of our proposal of redesign due to its underlying ontology of security.

As a form of evaluation, we show that our proposal is able to describe risk treatment options, according to ISO 31000 [2]. To illustrate our proposal, besides presenting multiple application examples, we proceed with an elucidative study case from the cybersecurity domain, representing a recent breach with the LastPass password manager.

The remainder of this paper is structured as follows:

- In Section 2, we provide some methodological considerations for our work, explaining how we employ *ontological analysis*;
- In Section 3, we present the baseline of our work, namely the ontology of prevention and the ontological foundations of value, risk, and security;
- In Section 4, we present the original proposal for modeling risk and security in ArchiMate–the Risk and Security Overlay; in addition to that, we present the well-founded risk elements of ArchiMate that form the basis for our work;
- In Section 5, we proceed with an ontological analysis of the security elements of RSO by showing its semantic shortcomings, according to the methodology described in Section 2;
- In Section 6, as a result of the previous analysis, we redesign ArchiMate RSO accordingly. This Section includes a novel presentation about how to represent prevention in ArchiMate;
- In Section 7, we show the multiple ontology-based patterns of security modeling in ArchiMate that are entailed by our proposal–also, something completely novel compared with our previous work;
- In Section 8, we show that our proposal is able to represent risk treatment options of ISO 31000;
- In Section 9, we illustrate our proposal by means of a detailed study involving a real-world security incident and the enterprise's reaction to it;
- We conclude with an extended discussion on related work in Section 10 and final remarks in Section 11.

2 Methodology: Ontological Analysis

Our general approach is aligned with "Design Science Research" [12] in that there is a focus on an artifact and on its cycles of (re)design and evaluation. The artifact is justified through its *relevance* to a certain context of application (Enterprise Risk Management), and required *rigor* is employed in artifact (re)design (ontological foundations and ontological analysis).

Having established the necessity of security modeling for Enterprise Risk Management purposes (*problem identification*), we start by evaluating an existing artifact, the RSO of ArchiMate, through an ontological analysis (*assessment of a current solution*). After identifying several ontological limitations of the RSO, we propose its redesign, accordingly, that is, a novel improved artifact to overcome these shortcomings (*an innovative intervention*, *solution*). Then, we show the capabilities of our proposal, including ontology-based security modeling patterns (*development of our solution*). Finally, as *a form of assessment*, we show that our proposal is capable of representing risk treatment options defined by ISO 31000 [2]. In what follows in this section, we explain the method of ontological analysis.

ArchiMate is a modeling language for Enterprise Architecture. The RSO enriches ArchiMate with risk and security elements to support Enterprise Risk Management and security. It is known that one of the key success factors behind the use of a modeling language is its ability to provide its target users with a set of modeling primitives that can directly express important domain abstractions [7]. In other words, the more the grammar of a domain-specific modeling language corresponds to the ontology of the domain, the more capable the language is of modeling domain scenarios accurately. An ontological analysis is "the evaluation of a modeling grammar, from the viewpoint of a predefined and well-established ontology" [9], which is, in our case, ROSE [11] concerning the security domain. Ideally, according to Rosemann *et al.* [9], the modeling grammars should be isomorphic to their underlying ontology, that is, the interpretation from the modeling constructs to the ontology concepts should be bijective. This is a desirable characteristic because

it prevents certain types of issues that affect the modeling capability of the language: (a) *ontological incompleteness* (or *construct deficit*), which is the lack of a grammatical construct for an existing ontological concept; (b) *construct overload*, which occurs when one grammatical construct represents more than one ontological construct; (c) *construct redundancy*, which happens when more than one grammatical construct represents the same ontological construct; (d) *construct excess*, when there is a grammatical construct that does not map to any ontological construct [9]. With the support of this framework, summarized in Figure 1, we identify shortcomings concerning the security modeling capability of the RSO in Section 5.

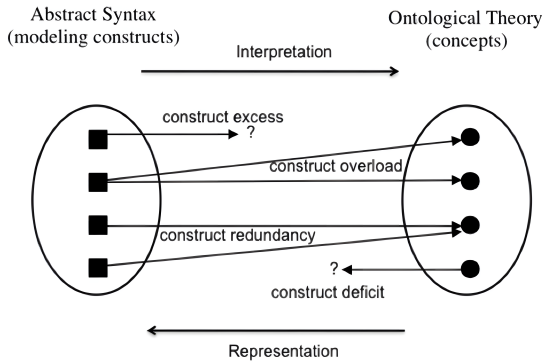

Fig. 1 The illustration from [13] of the relation between modeling constructs in a language's syntax and ontological concepts

3 Ontological Foundations

Understanding security requires understanding concepts of value and risk, as security involves protecting valuable assets from threats and losses. However, the very notion of protection is related to the notion of prevention: the idea of avoiding or counteracting the occurrence of certain types of events or processes. This is why some of us previously proposed an ontology of prevention [14] as a general foundation for our ontology of security [11]. This theory of prevention implies the existence of multiple patterns explaining why certain types of events are prevented. In this section, we present those ontological foundations that support our analysis and redesign of ArchiMate security elements.

3.1 An Ontology of Prevention

Prevention is about blocking an effect before it happens or stopping it as it unfolds. Prevention may occur as a natural phenomenon or as a result of intentional human intervention, which is a key aspect of the security domain. For example, vaccines prevent the unfolding of diseases; seat belts prevent events causing serious injuries; and circuit breaks prevent the manifestation of overcurrents. Prevention may occur with a certain degree of likelihood and with a certain level of effectiveness. The ontology of prevention [14], grounded in UFO, says that an event e prevents certain types of events ET if e brings about a situation of a type that is incompatible with the types of situations that would be necessary to trigger the events of type ET. Figure 2 summarizes this idea on the type level, that is, in terms of regularities.

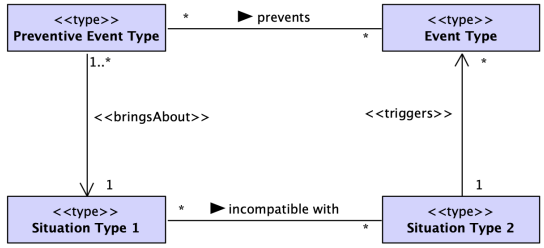

Fig. 2 Prevention schema: certain types of events bring about situations of a given type, such that other types of situations are impossible, resulting in the prevention of the types of events that are triggered by these situations

The key point from which several patterns emerge as logical consequences is that this incompatibility may be the result of changes in multiple different entities. Once we have in mind that events are manifestations of *interacting* dispositions (capabilities, vulnerabilities, liabilities, etc.), which inhere in objects, we conclude that prevention may occur if:

1. a given disposition is altered in the scene (situations);
2. its mutual partner dispositions (the ones necessary to its manifestation) are altered in the scene;
3. the object, bearing one of these dispositions, is altered in the scene.

Note that what we call "patterns of prevention" are simply one of those changes in the state

of affairs, explaining why the phenomenon of prevention happened according to the theory. These patterns correspond to what Blomqvist and Sandkuhl [15] called "semantic patterns": language-independent description of a certain concept, relation, or axiom. For each change in situations, these patterns can be grouped into "design patterns" (collection of semantic patterns [15]) and, then, arranged according to domain entities, forming "architecture patterns" [15] to achieve a goal (for instance, preventing certain kinds of attacks).

Consider the following illustrative scenario: a certain computer software contains a vulnerability, whose exploitation may happen due to the threatening capabilities of certain malware that are present in the same device. Removing the vulnerability by updating the software or destroying the malware before the manifestations of its capabilities are two different ways of preventing the events of exploitation. This is so because either the prevention event (software update or malware destruction) would bring about a situation where that vulnerability is now absent or where the threatening capabilities were removed from the scene (since they depend on the presence of the malware that was destroyed). This reasoning is applied to develop the ontology of security described below as security involves the prevention of risks.

3.2 Ontological Foundations of Value, Risk, and Security

Taking into consideration risk treatment options defined by ISO 31000, the *Reference Ontology for Security Engineering* (ROSE) [11] describes the general entities and relations of the security engineering domain, making use of an adapted version of the *Common Ontology of Value and Risk* (COVER) to capture the value and risk-related notions[1]. ROSE understands the domain of security as the *intersection* between the domain of value and risk, understood under the terms of the Common Ontology of Value and Risk (COVER) [8], and the dispositional theory of prevention presented in [14]. The latter extends UFO to explain how certain types of events are prevented or interrupted due to the occurrence of other events of certain types. From this perspective, a Security Mechanism is an Object that was intentionally designed to create value by protecting certain goals from Risk Events (encompassing Threat Events and Loss Events) in a systematic fashion.

In COVER[2], whose fragment is depicted in Figure 3, *value* is a relational mode that emerges from the relations between the capacities (Dispositions) of certain objects and the goals of an Agent. The manifestations of these capacities are Events that *bring about* a Situation that *impacts* or satisfies the goal of a given Agent (a Value Subject) – a goal is understood as the propositional content of an Intention [17]. Risk is the anti-value: Risk Events are the manifestations of (threat) capacities, Vulnerabilities, and, sometimes, Intentions that inhere in an Agent; these Events *bring about* a Situation that *hurts* the goal of a given Agent (a Risk Subject), as shown by Figure 4. Analogous to value, security (Figure 5) is also a relational mode that emerges from the relations between the (control) capabilities of Objects and the goals of an Agent, particularly a Protected Subject; however, manifestations of these capabilities *bring about* a Situation that *impacts* the goal of an Agent in a very specific way: *preventing* Risk events [11].

Using the prevention theory described in [14], ROSE understands that Threat Capability, Vulnerability, and, sometimes, Intention are dispositions associated with types whose instances maintain a *mutual activation partnership* [18] to each other. This means that a Threat Object can only manifest its Threat Capability if a Vulnerability can be exploited; if the Threat Object participates in an Attack (an Action, an intentional Event), then the Intention is also required. Analogously, a Vulnerability is only manifested in the presence of a Threat Capability. From a security point of view, the importance of this *generic dependence* relation

[1] Files related to ROSE can be found in the following public repository: https://purl.org/security-ontology.

[2] The OntoUML stereotype connects types and relations in these models to ontological categories of monadic and relational universals in UFO, respectively. For their ontological justification and semantics, one should refer to [16]. Moreover, the colors in these diagrams represent a color convention used by the OntoUML community: object types are represented in pink, intrinsic aspect types in blue, situation types in orange, event types in yellow, and higher-order types in darker blue.

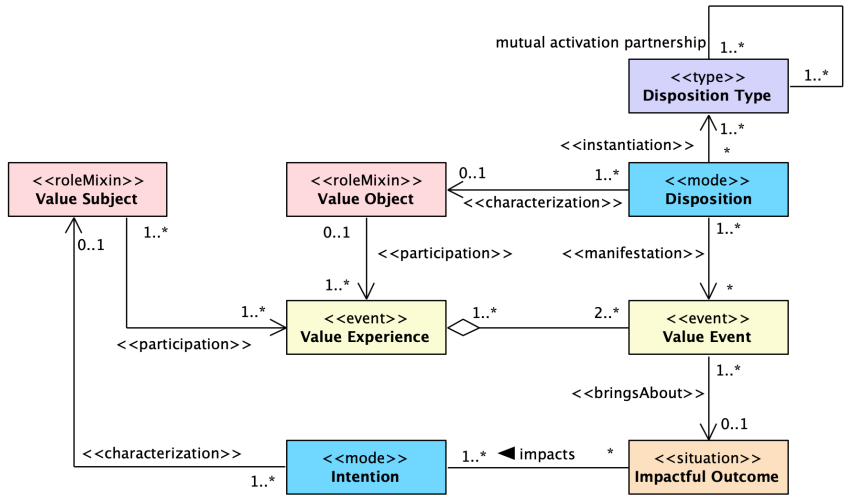

Fig. 3 Value Experience, adapted from [8, 11]

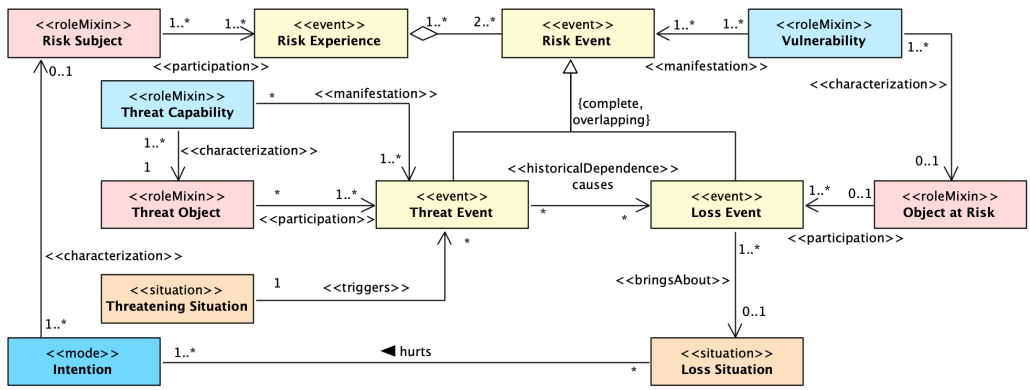

Fig. 4 Risk Experience, adapted from [8, 11]

among these entities is that it determines multiple ways by which security measures can work: the removal of any of them from the situation that could activate them all together implies the prevention of the associated Risk Event. In general, mutual activation partners compose the conditions of activation of any Disposition, as shown by Figure 3. This relation generalizes the role of enabler objects (Value Enabler, Risk Enabler, Threat Enabler, and so on), which aggregate ancillary Dispositions with regard to Threat Capability, Vulnerability, etc.

A Security Mechanism is always designed by an Agent called the Security Designer to be a *countermeasure to* events of a certain type (Risk Event Type) [11, 14]. When an Object is made to be a countermeasure to certain types of events, it aggregates capabilities whose manifestations systematically prevent these events. The Agent who designs a Security Mechanism is not necessarily the one protected by its proper functioning, i.e., the Protected Subject. However, both agents have Intentions that are positively impacted by this proper functioning. For example, the government designs policies for public safety, and the functioning of such policies satisfies some goals the government had when designing them but also satisfies the goal of people who want to be safe. Sometimes, the Protected Subject is the same Agent as the Security Designer, such as when a person places an electric fence surrounding their own house.

A Security Mechanism is an object, which may be a simple physical object like a wall, a

high-tech air defense system like the Israeli Iron Dome, an Agent like a policeman, a social entity like a security standard or anti-COVID-19 rules, that bears capabilities called Control Capability. The manifestation of this kind of capability is a Control Event (or Protection Event), which may come in a form of a chain of events that ultimately causes the Control Event. The Control Event is of a type (Control Event Type) that prevents, directly or indirectly, events of a certain type (Risk Event Type). This is so because the Control Events bring about a Controlled Situation, which is of a type that is *incompatible with* the types of Situations (Risk Trigger Type) that trigger Risk Events of certain types.

Notice that Control Capabilities may characterize not only a Security Mechanism but also other objects. This means that a Control Event can be, for instance, a single action that prevents certain types of Risk Events, although not in a systematic fashion. For instance, when someone puts herself away from dangerous machines in a factory, she is manifesting her Control Capabilities by avoiding the danger and, therefore, generating value for herself, even though she is not a Security Mechanism. This is important to draw a distinction between a Security Mechanism whose actions are systematic and a Control Event that may be the manifestation of a Control Capability that does not inhere in a Security Mechanism.

4 Risk and Security Modeling in ArchiMate

Risk and security modeling was not initially supported by ArchiMate. This is why some approaches emerged to address this gap by customizing the ArchiMate language accordingly. The main proposal is the RSO [4]–the target of our analysis. A well-founded redesign regarding risk elements of RSO has been proposed [6] by our previous work. As security modeling requires risk modeling, this is our natural starting point.

4.1 The Original ArchiMate Risk and Security Overlay

The latest version of the RSO was developed by a joint project of The Open Group ArchiMate Forum and The Open Group Security Forum [4], accommodating changes to the ArchiMate language in Version 3.1 of the standard. The RSO was designed through ArchiMate language customization mechanisms; in particular, the specialization of both ArchiMate Core and Motivation and Strategy elements, and additional risk and security-specific attributes [4].

The RSO supports the representation of Threat Agents as those responsible for Threat Events, which are events that trigger Loss Events. Both Threat and Loss Events are associated with Vulnerabilities, which in turn are associated with Resources. Loss Events influence Risk assessments, which can motivate Control Objectives. These are then realized in Security Requirements and Control Measures, which are in turn realized in Implemented Control Measures.

The RSO defines a Threat as "a possible danger that might exploit a vulnerability to breach security and thus cause possible harm". Admitting this the term is ambiguous, the authors distinguish between the events that have the potential of harming the organization, which they call Threat Events, from the entities responsible for intentionally or unintentionally causing them, which are labeled Threat Agents. Because this element can be applied to groups or objects, such as a machine or an organization, a Threat Agent may be represented by any Active Structure Element. A Threat Event is represented by a specialized Business Event, whereas a Loss Event is defined as "any circumstance that causes a loss or damage to an asset" and is triggered by a Threat Event. It is also mapped to a Business Event in ArchiMate.

Vulnerability is given two definitions. In one definition, a Vulnerability is "the probability that an asset will be unable to resist the actions of a threat agent". The second defines a Vulnerability as "a weakness which allows an attacker to threaten the value of an asset". Vulnerabilities are mapped as ArchiMate Assessments, which "represents the result of an analysis of the state of affairs of the enterprise concerning some driver.". A Vulnerability can be associated with both Threat Events and Loss Events as well as with resources and other core elements.

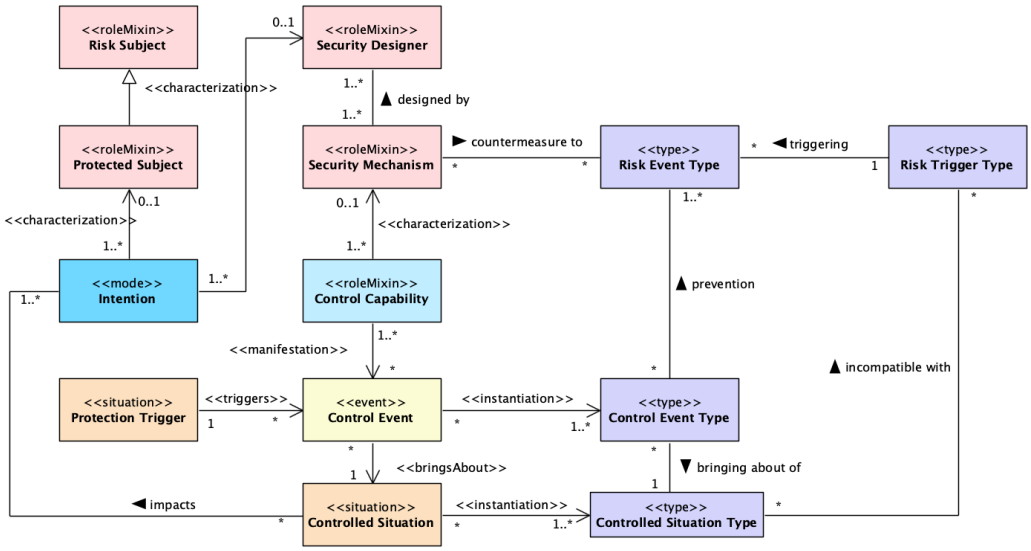

Fig. 5 Security Mechanism, adapted from [11]

Risk is defined as "the probable frequency and probable magnitude of future loss", following the definition proposed in the Open FAIR Risk Taxonomy. But other definitions are provided: "the potential of loss (an undesirable outcome; however, not necessarily so) resulting from a given action, activity, and/or inaction, foreseen or unforeseen". A third definition, namely that "a risk is a quantification of a threat" is invoked to justify the representation of Risk using a specialization of the Assessment construct in ArchiMate.

In the RSO, risks are usually represented by focusing on a particular entity the organization desires to protect–an Asset at Risk. This notion of asset accounts for any kind of object, tangible or intangible, that can be owned or controlled by the organization to create value. This is why it can be applied to a Resource or any Core Element in ArchiMate (including Business Actors and Business Processes).

The RSO proposes five elements in the Security domain: Control Objective, Security Requirement, Security Principle, Control Measure and Implemented Control Measure.

Control Objectives (or security objectives) are defined according to the outcome of Risk Assessments. Control Objectives are high-level goals that define what the organization plans to do about an identified risk. For instance, if the Risk of employees getting injured in work-related accidents is considered unacceptable, the organization might decide to reduce it (e.g. by changing safety procedures) or to transfer it (e.g. by purchasing a broader insurance policy). In any case, the result of this decision is captured by a Control Objective, which is mapped as an ArchiMate Goal.

A Control Objective should be realized by a Security Requirement (or control requirement), which is defined as "formalized needs to be fulfilled by means of a control in order to face an identified threat". A Control Measure is simply a more specific Security Requirement: "during the risk analysis process, a specification of an action or set of actions that should be executed or that must be implemented as part of the control, treatment, and mitigation of a particular risk" [4]. Both Security Requirement and Control Measure are represented as specializations of the ArchiMate's Requirement. Given the lack of details in the white paper, the two aforementioned definitions may be equally applied to Security Requirement and Control Measure.

An Implemented Control Measure is the deployment of a Control Measure. Depending on the kind of control, almost any core concept or combination of core elements of ArchiMate can be used to model the implementation of a Control Measure. This is so because an Implemented

Control Measure can be an "action, device, procedure, or technique that reduces a threat, a vulnerability, or an attack by eliminating or preventing it, by minimizing the harm it can cause, or by discovering and reporting it so that corrective action can be taken". A Control Measure may also be realized by a grouping of a set of core elements as its implementation [4].

The notion of Security Principle is less developed in the RSO white paper [4]. A Principle in ArchiMate represents a statement of intent defining a general property that applies to any system in a certain context in the architecture [5]. Similarly to requirements, principles defines the intended properties of systems. But principles are wider in scope and more abstract than requirements. For example, the principle "Information management processes comply with all relevant laws, policies, and regulations" is realized by the requirements that are imposed by the actual laws, policies, and regulations that apply to the specific system under design [5]. A Security Principle is related to the notion of policy and ArchiMate Motivation elements, though the RSO offers neither an explicit definition of it nor its usage in an example. The white paper also notes that the ArchiMate language does not have the concept of operational policy [4].

Figure 6 summarizes how RSO proposes to represent risk and security elements in ArchiMate [4]. An Implemented Control Measure is associated with an Asset at Risk, which can be a Resource or a core element of ArchiMate. An Implemented Control Measure influences negatively a Vulnerability as an Assessment, in the sense that it makes the emergence of a Threat Event and the consequent Loss Event associated with that Vulnerability less probable.

To exemplify how the RSO can be used, we present two examples extracted from [4], highlighting the assumptions that the white paper calls "common characteristics shared by entities in risk management domains". The examples refer to the case of the Coldhard Steel company, illustrating the stereotyping of ArchiMate Motivation elements as risk elements. Figure 7 represents the risk of losing production due to machine failure. A power supply assembly is an Asset at Risk that fails when the power fluctuates (a Threat Event). This power assembly failure causes the failure of other machines, characterizing a loss for the organization (a Loss Event), associated with the Risk of production loss. Then, the Control Objective is defined as an adequate peak power supply capacity, which means that the organization seeks to reduce this risk, which should be done by the Control Measure of replacing the power supply assembly. By this example, we notice some of the aforementioned characteristics: the asset is exposed to a threat or a risk due to its vulnerability, but, at the same time, the asset posses a control requirement and, indeed, participates in the realization of its own Control Measure.

The second example (Figure 8) illustrates a risk mitigation approach – continuous improvement of machine reliability – applied across the entire Coldhard Steel risk management domain. The implementation of control measures is grouped by Risk Mitigation Domain, aimed at negatively influencing the vulnerability of inadequate power supply. This implementation involves several core elements of ArchiMate, such as Contract, Outcome, Business Process, and Equipment.

Table 1 lists the risk and security elements according to ArchiMate elements they specialize, including their definitions from [4].

4.2 Ontology-based Risk Modeling in ArchiMate

In [6], we performed an ontological analysis of the risk aspects of the RSO based on the Common Ontology of Value and Risk (COVER), proposing a redesign of part of the RSO to address the limitations identified by the analysis. Figure 9 shows the proposal of [6] for evolving the RSO, while Table 2 shows the full representation of risk concepts in ArchiMate based on COVER. This representation will be the basis of our own proposal concerning the security aspects of ArchiMate.

A Hazard Assessment, proposed to represent UFO situations that activate Threat Capabilities, is an identified state of affairs that increases the likelihood of a Threat Event and, consequently, of a Loss Event. The occurrence of these events depends on the Vulnerabilities of an Asset at Risk or of a Threat Enabler and the Threat Capabilities involving Threat Agent. All of this forms the Risk Experience

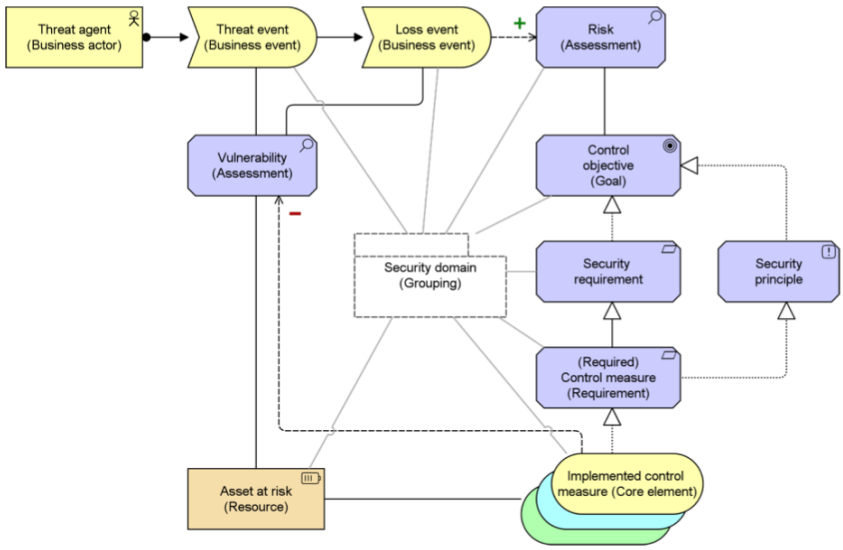

Fig. 6 Mapping of Risk and Security Elements to the ArchiMate language [4]

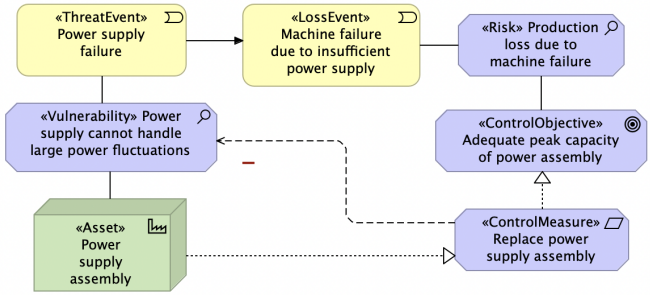

Fig. 7 Example from the case of the Coldhard Steel company [4]

of a Risk Subject, whose intention or Goal is harmed by a Loss Event. This experience may be assessed by a Risk Assessor (who may be the same subject as the Risk Subject) through a Risk Assessment (e.g., that determines that the Risk is unacceptable).

5 Ontological Analysis

Taking ROSE as a reference, our ontological analysis relies on concepts described in Section 2, namely: (a) ontological incompleteness (or construct deficit); (b) construct overload; (c) construct redundancy; (d) construct excess. Then, in Section 6, this analysis supports the redesign of the RSO of ArchiMate by the introduction or elimination of elements.

5.1 Redundant Intentions and Lack of Clarity

The notions of Control Objective, Security Requirement, Control Measure, and Security Principle, all reflect a desired state of affairs that guides the actions of some agent. As we interpret the RSO, there are two relevant aspects among these distinctions: (1) a distinction between an end and a means to this end; that is the meaning behind, for example, the statement that a Security Requirement (a means) realizes a Control Objective (an end); and (2) the generality and abstractness of these intentions, in the sense that, for example, Control Objective is more general than Control Measure; concerning this generality and abstractness, it is not clear where Security Principle should

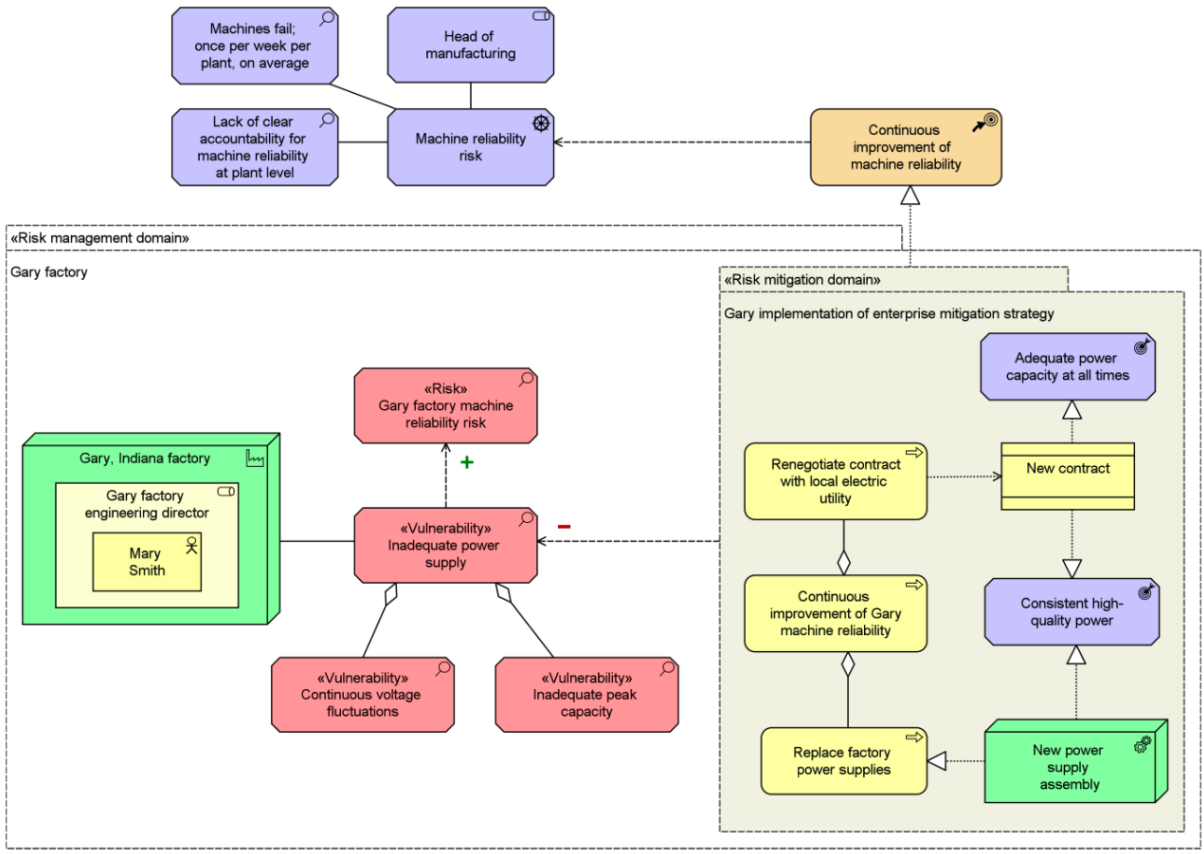

Fig. 8 Mitigation of Machine Failure Risk at Coldhard Steel Gary Factory [4]

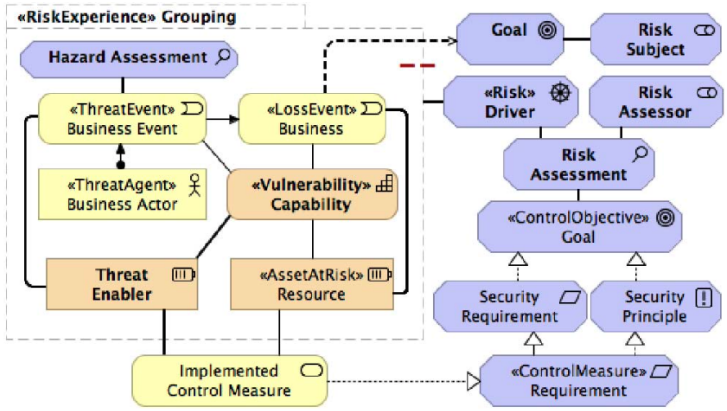

Fig. 9 Proposal of [6] for evolving the Risk and Security Overlay

be placed, since in Figure 6 Security Principle realizes Control Objective, though the documentation of ArchiMate suggests Principle has a higher level of generality and abstraction, which means the realization relation should be the inverse. The white paper [4] does not provide an example employing Security Principle or even Security Requirement, making use solely of Control Objective, Control Measure, and implemented control measure. Furthermore, no distinction is made regarding how Control Measure specializes Security Requirement.

Table 1 Summary of risk and security modeling elements in ArchiMate's Risk and Security Overlay (RSO)

RSO Element	ArchiMate Element	Definition
Threat Agent	Active Structure Element	Anything that is capable of acting against an asset in a manner that can result in harm.
Threat Event	Business Event	Event with the potential to adversely impact an asset (including attacks).
Loss Event	Business Event	Any circumstance that causes a loss or damage to an asset.
Vulnerability	Assessment	D1: The probability that an asset will be unable to resist the actions of a threat agent. D2: A weakness that allows an attacker to threaten the value of an asset.
Risk	Assessment	D1: The probable frequency and probable magnitude of future loss. D2: The potential of loss resulting from an action, activity, or inaction, foreseen or not.
Asset at Risk	Resource, Core Element	D1: Anything tangible or intangible that can be owned or controlled to produce value. D2: Any data, device, or other components of the environment that supports information-related activities.
Control Objective	Goal	A high-level goal that should be realized by a Security Requirement (e.g. reduction, transfer, sharing).
Security Requirement	Requirement	A formalized need to be fulfilled by means of a control in order to face an identified threat.
Security Principle	Principle	A principle that has something to do with policy, which is defined as a set of rules that governs the behavior of a system.
Control Measure	Requirement	In a risk analysis process, a specification of an action or set of actions that have to be performed or that should be implemented as part of the control, treatment, and mitigation of a particular risk.
Implemented Control Measure	Core Element	D1: An action, device, procedure, or technique that reduces a threat, a vulnerability, or an attack by eliminating or preventing it, by minimizing the harm it can cause, or by discovering and reporting it so that corrective action can be taken. D2: The deployment of a set of security services to protect against a security threat.

The means-end distinction is relational: an end targeted by a means may be a means to another end. For example, protecting the technical infrastructure from damage may be an end targeted by control measures, but it may also be a means

Table 2 Representation of risk concepts in ArchiMate based on COVER [6]

COVER Concept	Representation in ArchiMate
Vulnerability	Capability stereotyped with «Vulnerability»
Threat Object	Structure Element stereotyped with «ThreatAgent»
Threat Event	Event stereotyped with «ThreatEvent»
Hazard Assessment	Assessment stereotyped with «HazardAssessment»
Loss Event	Event stereotyped with «LossEvent»
Intention	Goal
Risk Subject	Stakeholder associated with a Goal that is negatively impacted by a «LossEvent»
Object at Risk	Structure Element stereotyped with «AssetAtRisk»
Threat Enabler	Structure Element associated with a «ThreatEvent» or a «LossEvent»
Risk Experience	Grouping stereotyped with «RiskExperience»
Risk	Driver stereotyped with «Risk»
Risk Assessment	Assessment associated with a «Risk»
Risk Assessor	Stakeholder associated with a Risk Assessment

to achieve mandatory legal requirements. Because of all that, those distinct notions of the RSO seem to be a case of construct redundancy, since different security modeling constructs represent the same ontological concept. The redundant constructs (particularly, Security Requirement and Security Principle) do not seem to play any practical role in security modeling[1]. We refer to this as *Limitation L1.*

5.2 Underspecification of Implemented Control Measures

An Implemented Control Measure can be any ArchiMate core element or multiple core elements grouped in a cluster, as seen in Figure 8. This would look like a construct overload since a single construct collapses the object, its capability, and the event that is the manifestation of this capability. However, it is actually a strategy of representation via a supertype, so it is not an ontological problem by itself. The issue relies on the fact that this strategy offers no guidance to the modeler on what the implementation of a control measure should look like. In other words, the device of Implemented Control Measure is too generic and suffers from underspecification. In contrast, ROSE unfolds the notion of security mechanism in a general pattern that distinctively shows the difference between objects (Protected Subject, Security Designer, Security Mechanism), their modes and capabilities (Intention, Control Capability), the associated events (Control Event) and situations (Protection Trigger, Controlled Situation). The lack of this richness of the domain may be better classified as a construct deficit. This is aggravated by the assumption that the asset itself realizes its own control measure (see Figure 7), suggesting confusion between the Object at Risk and elements of the pattern of Security Mechanism. We term this issue *Limitation L2.*

5.3 Lack of Distinction Between Baseline Architecture and Target Architecture

The implementation and migration concepts of ArchiMate are used to describe how an architecture will be realized over time through changes [1], providing the means to represent a baseline and a target architecture. The existence of these concepts in ArchiMate is justified by the importance of accounting for changes in the process of evolution of an enterprise. The introduction of a security mechanism is one of these changes. However,

[1]Actually, we can wonder whether the distinction of several of ArchiMate's Motivation Elements is (or not) redundant, such as goal, outcome, requirement, and principle, but this issue is outside the scope of our paper.

the RSO does not make use of this characteristic of ArchiMate, simply showing that security entities have a negative influence on Vulnerability. The redesigned RSO (see Figure 9) connects Implemented Control Measure to Threat Enabler and Asset at Risk, in order to express the impact on the threat event or the loss event. Still, no account of change is provided, as it would be expected from the capabilities of ArchiMate language by the means of constructs showing different plateaus from the baseline architecture to the target architecture. We call this lack of use of temporal aspects of ArchiMate *Limitation L3*.

5.4 Modeling the Subjects in the Security Domain

ROSE highlights there is a subject whose Intention is positively impacted by the effects of a Security Mechanism, the Protected Subject. Considering the risk domain, it is clear that this subject must be a proper subtype of the Risk Subject, which appears in the redesigned version of the RSO, as seen in Figure 9. In addition, another subject has not only his or her intentions positively impacted by the effects of a security mechanism, but is also responsible for the creation or introduction of the mechanism – often due to legal or contractual reasons, such as when someone is hired to install an electric fence. This is what ROSE calls the Security Designer. Sometimes the Protected Subject and the Security Designer are the same individuals, while sometimes this is not the case. The original RSO presents none of that, whereas these subjects are not part of the scope of the redesigned version of the RSO. In summary, a case of construct deficit. We call this *Limitation L4*.

5.5 Triggering Conditions of Protection Events

The manifestation of the capability of a Security Mechanism occurs due to a Protection Trigger, a certain state of affairs that activates that capability. This represents environmental conditions that affect the manifestation of a Control Capability. For instance, a circuit breaker manifests its capability of interrupting a current flow when a fault condition is detected (heating or magnetic effects of electric current). In the redesigned RSO, there is an analogous notion for Threat Event, a threatening circumstance mapped as an assessment called Hazard Assessment [6]. They are particular configurations of the world that allow or increase the probability of the occurrence of a Threat Event. The advantage of explicitly accounting for the situations that trigger the Protection (Control) Event is that we can represent how several environmental factors increase the effectiveness of the Security Mechanism, assuming its effectiveness is directly connected to how likely it works properly, manifesting the Protection Event. This whole dimension is neglected by the RSO, a case of construct deficit – *Limitation L5*.

5.6 Interdependence Relation Among Risk Capabilities

As shown by ROSE, in its risk aspects (Figure 4), the manifestations of threat capabilities, vulnerabilities, and, sometimes, intentions depend on the presence of each other. From this perspective, for example, it makes no sense to say that there is an ongoing threat without the simultaneous participation of a vulnerability. More importantly, from the security modeling point of view, recognizing this generic dependence relation among these entities allows for different strategies of protection or mitigation, since the removal of any of these capabilities or intentions would result in the prevention of the threat or loss event. Again, this dimension is not considered by the RSO, which refers to the efficacy of the control measure as simply influencing negatively a vulnerability. Doing so, the RSO says nothing about the multiple patterns of prevention uncovered by ROSE. Therefore, a case of construct deficit, *Limitation L6*. Table 3 summarizes the ontological limitations.

6 A Well-Founded Security Overlay in ArchiMate

Once we identified the ontological limitations of the RSO, we can proceed with a redesign of ArchiMate security constructs in such a way that those shortcomings are solved. To do this, first, we need to formulate the ontology of prevention in ArchiMate's terms, then we will use ROSE to address each limitation accordingly.

Table 3 Summary of Ontological Limitations

Ontological Limitation
L1. A *construct redundancy* and lack of clarity of Control Objective, Security Requirement, Control Measure, and Security Principle, which all reflect a desired state of affairs that guides the actions of some agent, that is, a UFO Intention.
L2. Underspecification of Implemented Control Measure, a *construct overload* in the sense that this construct is a supertype of multiple different security elements.
L3. Lack of distinction between baseline architecture and target architecture, that is, *lack of use of temporal aspects of ArchiMate*.
L4. A *construct deficit* since the RSO does not represent the subjects whose goals are affected by the introduction of a security mechanism, that is, the Security Designer and Protected Subject.
L5. A *construct deficit* due to the absence of a construct to represent the conditions related to the activation of a security mechanism.
L6. A *construct deficit* regarding the representation of the interdependence among risk-related capabilities.

6.1 Representing Prevention in ArchiMate

Because ArchiMate does not clearly distinguish the instance level from the type level, the representation of the theory of prevention in ArchiMate requires adaptation. The context can clarify whether we are speaking about a type or an individual, but there is no sense in assigning a probability to an individual event [8]. There are different ways of representing prevention, according to what the modeler wants to make explicit and due to the lack of formal semantics of ArchiMate. With the support of the «Likelihood» stereotype introduced in [19], we can say that a prevention event (type) is one that decreases the likelihood of the occurrence of events of a certain type (Figure 10). For instance, by adopting a two-factor authentication policy an organization decreases the chances of occurrence of a data breach. A broader view would take into account a previous event (type) that causes another one, so the prevention event (type), in this case, decreases the probability of an event causing another (type of event), as depicted in Figure 11. For example, phishing attacks are causally connected to data breaches but implementing cybersecurity training decreases the chances of a phishing attack causing a data breach. In both cases, the shown likelihood corresponds to the current state of affairs, that is, the likelihood affected by the prevention event.

Representing prevention the way expressed by Figure 10 and Figure 11 – taking into account only events, types of events, and their likelihoods – may suffice for certain needs. However, as shown by UFO and ROSE, events are always existentially dependent on their participants (objects) with interacting dispositions. Moreover, the dimension of time was neglected, that is, how things changed due to the introduction of a new element. Following [19] regarding the representation of dispositions in ArchiMate, we can say prevention in ArchiMate occurs due to the introduction of a new object whose capabilities are manifested as prevention events that decrease the likelihood of events of a certain type, as depicted in Figure 12. Now that we have a representation of a before and after the state of affairs, including objects and their properties, we can explain the effect of prevention as a change predicted by one or more patterns described in Section 3.1. We also see different likelihood values in different configurations (plateaus). In the case where events cause other events, we may have the prevention event as a continuation of the causal chain, instead of the prevented event - this is represented by ArchiMate's *or* junction, shown in Figure 13. For instance, as a result of acquiring the new capability of cybersecurity awareness, a company's employees avoid data breaches by behaving accordingly. This "right behavior" is the direct prevention event, whereas we can think of cybersecurity awareness training as the event that introduced the new capabilities

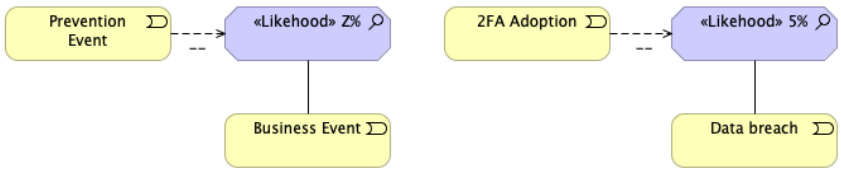

Fig. 10 A representation of a prevention event that decreases the likelihood of the occurrence of events of a certain type

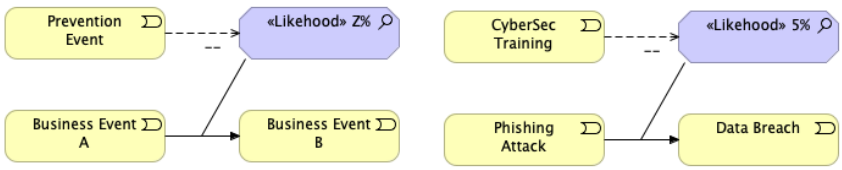

Fig. 11 A representation of a prevention event that decreases the likelihood of an event causing events of a certain type

– and, therefore, it has a prevention role too. We could also say that the cybersecurity awareness training is an implementation event that removes certain employees' vulnerabilities, a type of scenario represented by Figure 14. Another example of this kind would be the prevention implementation event of removing permission to commit to a repository so that the new architecture would not contain this permission.

6.2 Redesigning the Security Elements of ArchiMate

Since *L1* concerns a case of construct redundancy, we retain only the required constructs. So we retain Control Objective as a goal and Control Measure as a required means to achieve this goal. Considering this distinction from ROSE's perspective, we can conclude that the former is associated with a Protected Subject, while the latter is associated with a Security Designer, the one responsible for introducing the Security Mechanism. For example, a company has a Control Objective of protecting customers' data from cyberattacks. Based on an assessment, a series of Control Measures should be implemented by the company's cybersecurity team, playing the role of Security Designer; both the company and the customers may be regarded as Protected Subjects since they have assets at risk that should be protected.

L4 is the absence of these two subjects, so we propose to introduce them, respectively, as a Stakeholder and a Business Role. The Protected Subject specializes Risk Subject, though some Risk Subjects might not be Protected Subjects due to lack of protection. Similarly, *L6* is the absence of a dependence relation among Threat Capabilities, Vulnerabilities, and Intentions (Goal in ArchiMate), a limitation that is easily solved by adding ArchiMate's associations among these entities. To address *L5*, the introduction of ROSE's concept of Protection Trigger follows the previous work [6], which uses Assessment to represent Threatening (or Hazardous) Situations. So Protection Trigger becomes Control Assessment.

Limitations L2 and *L3* are treated together: the baseline architecture reflects the state of the organization before the implementation of a security mechanism, and the target architecture shows the impact of the implementation of the security mechanism. At baseline, following a proposal for a pattern language for value modeling in ArchiMate [19], there is a Likelihood associated with the causal emergence of a Threat Event and a Loss Event. The dependence relations among risk entities are also shown so that it should be clear that interfering in one of them would affect the likelihood of happening events like these. This is exactly what a Security Mechanism does in a systematic fashion, following the ROSE and the theory of prevention [11, 14]. But the implementation of a Security Mechanism is carried out by a Security Designer through the Work Package device of ArchiMate's migration layer, oriented by an identified gap in the baseline architecture. Once a Security Mechanism is implemented, the target architecture may

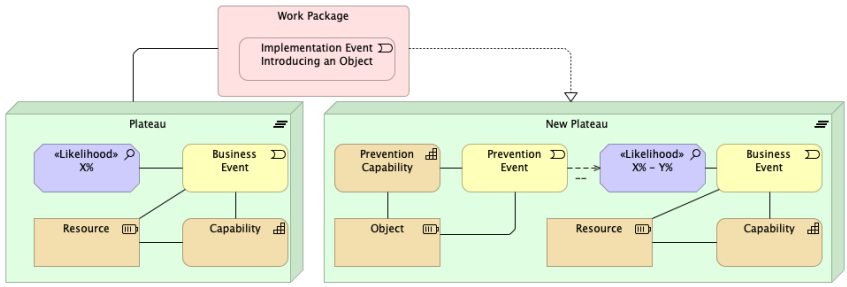

Fig. 12 A representation of prevention in ArchiMate that includes objects, their capabilities, and temporal changes

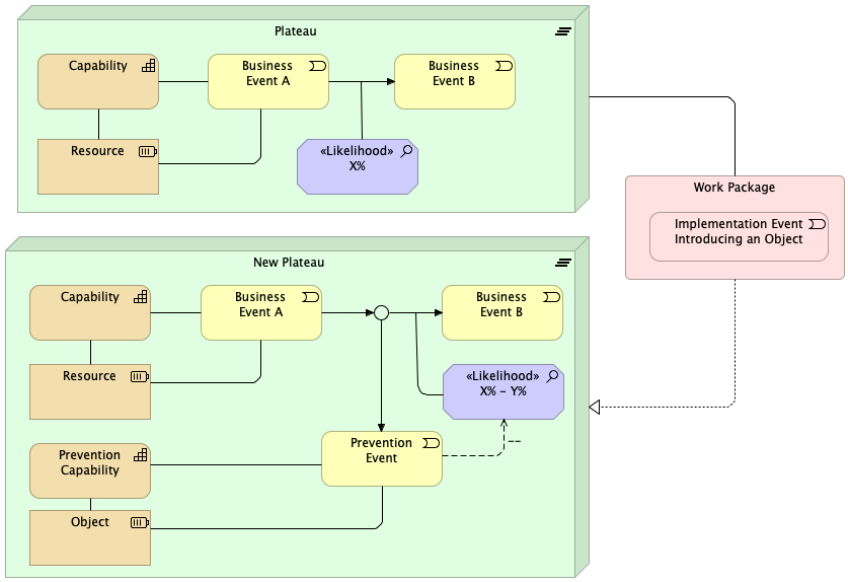

Fig. 13 A representation of prevention in ArchiMate that includes objects, their capabilities, and temporal changes, throughout a causal chain

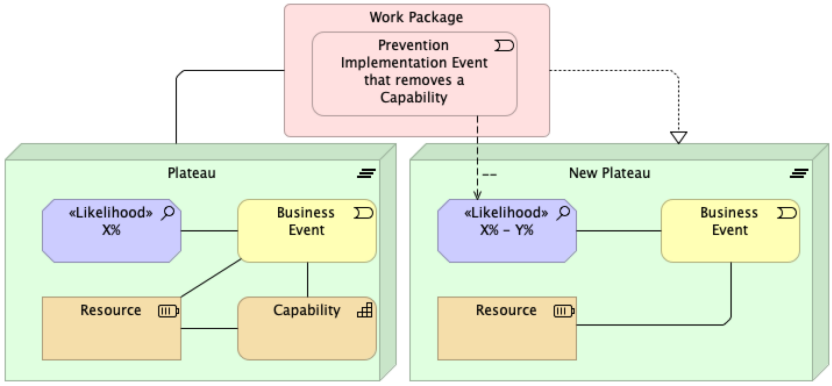

Fig. 14 A representation of prevention in ArchiMate where an implementation event removes a capability, therefore decreasing the likelihood of the associated events in the new plateau

show a different configuration of the risk entities that are interdependent, as well as a decreased

likelihood concerning the emergence of a Threat Event or a Loss Event. Because of that, Risk Assessment may also be different, maybe evaluating the risk is now acceptable. Similarly, the required Control Measure might change. The pattern of Security Mechanism from ROSE is translated in ArchiMate as a Structure Element that holds a capability whose manifestation is an event that negatively influences the likelihood of Threat Event or a Loss Event. This pattern follows the value modeling pattern in ArchiMate proposed by [19] since security is a matter of specific creation of value through the prevention of risks. Figure 15 shows our proposal to evolve the security aspects of the RSO, highlighting in bold the constructs and relations we propose. Table 4 shows our proposal of the representation of security concepts in ArchiMate based on ROSE.

Figure 16 exemplifies our proposal using the same example from the RSO involving a Loss Event of production loss caused by a Threat Event of power fluctuation with intermediate steps in between. Notice that there is a certain likelihood associated with the causation between the power fluctuation and the power supply failure. The business owner is the Risk Subject, and the Risk Assessment is that the risk of production loss is unacceptable. Considering this risk experience in the baseline architecture, therefore before the introduction of a prevention implementation event, which is a Control Event, the Control Objective is defined to be an adequate peak capability of power assembly, realized by a Control Measure of replacing power supply assembly. This is the responsibility of a technician, the Security Designer. In the target architecture, we see some changes concerning the risk entities: the new power supply assembly can handle large power fluctuations, so the likelihood of power supply failure is lower; the original power supply assembly was removed from the scene, which means its vulnerability was also removed from the scene. This is one of the ways of prevention [14]. Now, the risk of production loss is acceptable, because this interference in the risk causal chain ultimately decreases the chances of happening production loss. Finally, Control Measure turned into checking the capability of the new power supply assembly.

Note that nothing prevents us from designing multiple Security Mechanisms for the same type of Threat Event or Loss Event. Multiple Control Events (or Protection Events) can realize a single «ControlObjective». This is aligned with the idea of the Swiss cheese model in risk management [20].

We provide the resulting files with related information in a public repository[3]. Our proposal is well-documented on a dedicated website[4].

7 Ontology-based Security Modeling Patterns

In Section 3.1, we described general patterns of prevention according to an ontological theory grounded in UFO [14]. In Section 6.1, we proposed ways of introducing this ontology of prevention in ArchiMate, particularly displayed in Figure 12, Figure 13, and Figure 14. In Section 6.2, we proposed a redesign of ArchiMate according to ROSE (Figure 15), which assumes the ontology of prevention. Now, we will develop those patterns of prevention implied by this redesigned artifact.

Gangemi and Presutti [21] state that an "ontology design pattern" is a modeling solution to solve a recurrent ontology design problem. Considering this meaning, our patterns of prevention, once applied to the security domain as it is in ROSE, represent modeling solutions to address the task of modeling risk treatment measures. Then, we conclude there are at least the following ways of action of a Control Event, so that Threat Events or Loss Events are ultimately prevented:

1. The Threat Agent can be disabled by losing its Threat Capability. For example, when tranquilizer darts temporarily disable the threatening capacities of large animals.
2. The very Threat Agent can be destroyed or moved away from the scene. For instance, when missiles intercept dangerous projectiles or when inspections enforce regulations about the replacement of defective components.
3. The Threat Agent can be dissuaded from its Goals. For example, warnings, security cameras, and walls that demotivate thieves from starting their criminal activities against a facility. Obviously, this is only possible if

[3] See DOI: https://doi.org/10.5281/zenodo.10005209.
[4] See https://unibz-core.github.io/security-archimate/.

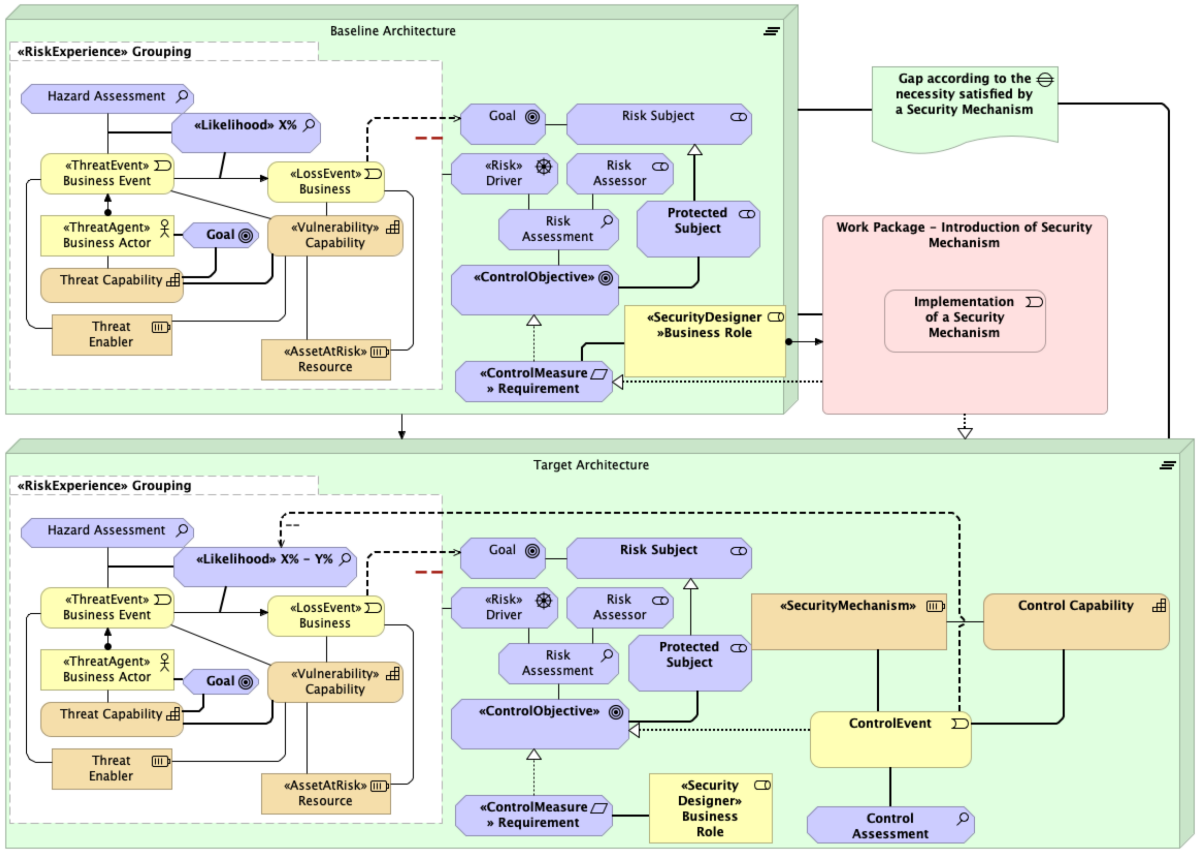

Fig. 15 Proposal for evolving the security aspects of the Risk and Security Overlay of ArchiMate

Table 4 Representation of security concepts in ArchiMate based on ROSE

Ontology Concept	Representation in ArchiMate Element
Protected Subject	A specialization of Risk Subject associated with a «ControlObjective»
Security Designer	Business Role stereotyped with «SecurityDesigner» (normally, it is associated with «ControlMeasure» and assigned to the implementation of a Security Mechanism)
Security Mechanism	Structure Element (Business Agent, Resource) stereotyped with «SecurityMechanism»
Control Capability	Capability associated with Control (Protection) Event
Protection Trigger	Assessment stereopyed with «ControlAssessment»
Control Event	An event that realizes «ControlObjective», and negatively influences the «Likelihood» associated with «ThreatEvent» or «Loss Event»

the Threat Agent is a person, a potential criminal, not a purposeless object.

4. The Assets At Risk can be hardened, that is, their Vulnerabilities can be removed. Say, when a piece of software provides updates for a given program by removing potentially problematic code.
5. The very Asset At Risk can be moved away from the scene. For instance, when customers

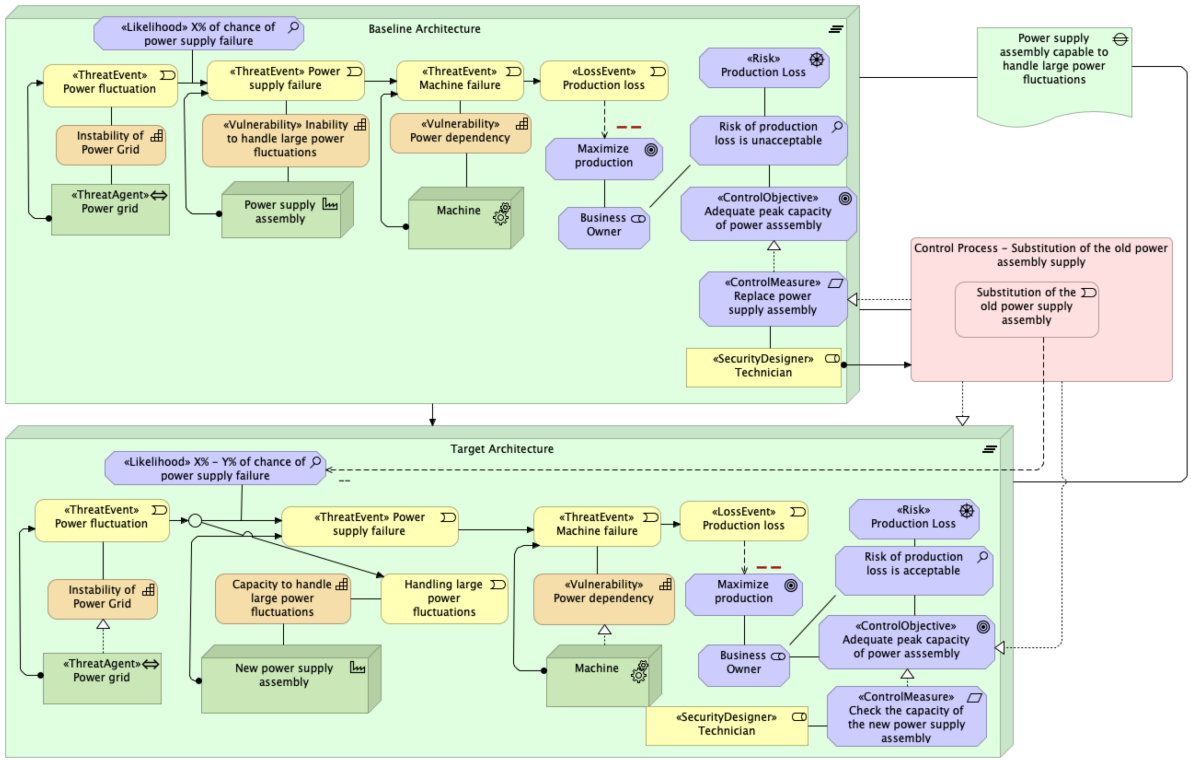

Fig. 16 Example of modeling the introduction of a security mechanism

and employees are blocked from accessing certain dangerous spaces in a factory.

There are other ways by which a Control Event can affect the architecture because this depends on partner dispositions that can be removed. For example, a Threat Enabler [6] simply aggregates partner dispositions with regard to the dispositions involved in a Threat Event or a Loss Event. This means that the removal of those partner dispositions (or their bearers) can also prevent those kinds of events. However, once we keep in mind other possible partner dispositions (in case of the necessity of further investigation), it makes sense to focus on those security-specific entities under the ROSE framework.

Note that all cases of patterns below are actually examples that instantiate those ontology-based security modeling patterns. We opt for this to produce more meaningful and useful material in the context of security, considering that there are infinitely many patterns according to the removal of partner dispositions.

7.1 Removing a Threat Capability

Figure 17 exemplifies the case where a Threat Capability is removed from the target architecture by a Control Event that is a prevention implementation event, as described in Figure 14. A software team has to deal with the common situation of inexperienced developers committing bad code, which sometimes leads to software failures. As a Control Measure, they remove this permission to commit to the project's repository for those developers. The target architecture reflects this change. In natural language, the removal of a Threat Capability is usually associated with the idea of *disabling* an agent or an object.

7.2 Removing a Threat Agent

Figure 18 exemplifies the case where the very Threat Agent is destroyed and, therefore, events associated with the Threat Agent's capabilities are prevented. More specifically, the baseline architecture reflects the risks to a factory's integrity caused by rocket attacks. An air

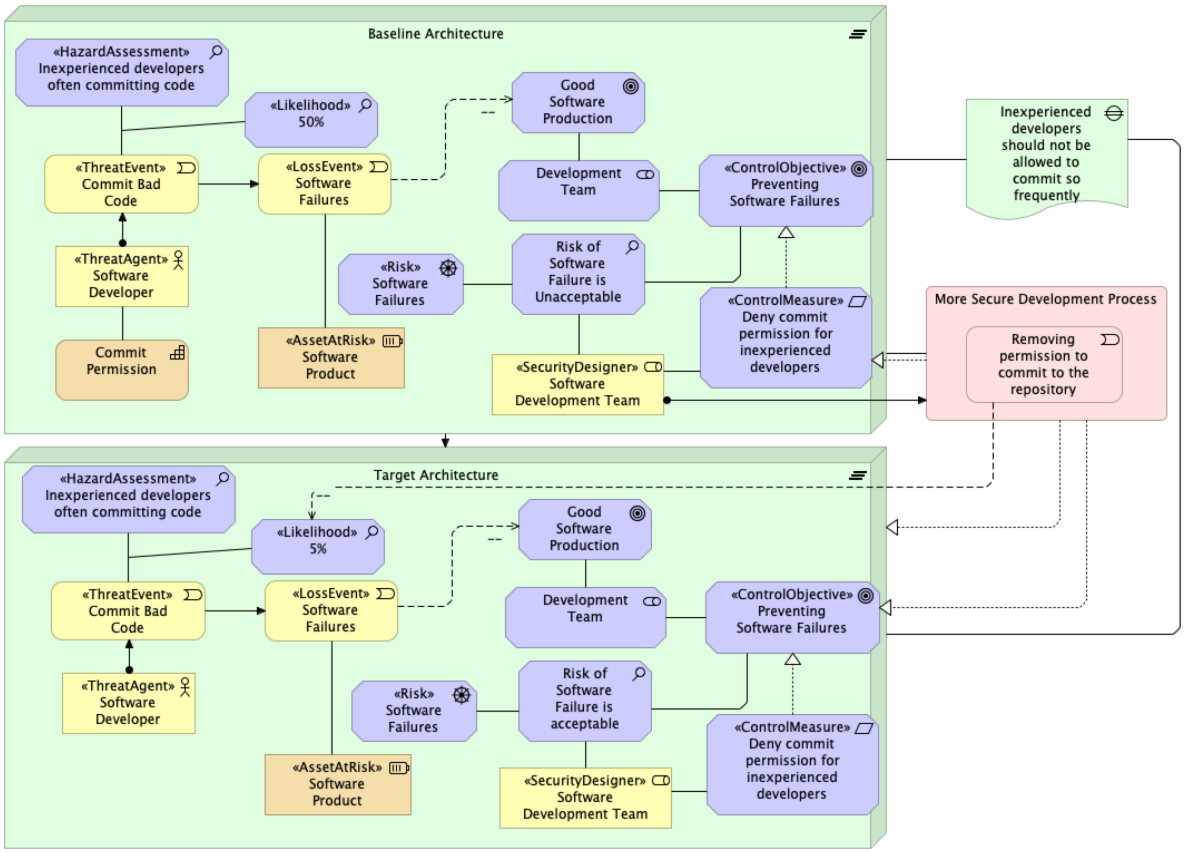

Fig. 17 Example of removal of the Threat Capability from the target architecture by a Control Event that is a prevention implementation event. Therefore, the associated Threat Event cannot occur

defense system is a Security Mechanism presented as a solution to diminish those risks. At first, the Threat Agent (the rocket) and its destructive capability are associated with both the Threat Event (the rocket attack) and Loss Event (the damage to the factory). In the target architecture, however, the Control Event is the outcome of the Control Capability of the air defense system, which is able to intercept the rocket, then avoid damage to the factory. To represent the likely destruction of the rocket, we disconnect it from the Loss Event. Rocket attacks can still end up damaging facilities because the effectiveness of the air defense system is not 100%.

7.3 Removing a Threat Agent's Goal

Removing a Threat Agent's Goal means dissuading the agent thanks to the deterrent capabilities of a given object. Warnings signs, security cameras, and walls can be considered Security Mechanisms that bearer Control Capabilities of this kind, among others. Figure 19 shows an example of this by depicting a scenario where the introduction of warning signs ultimately decreases the likelihood of work accidents and possible consequent death of employees.

7.4 Removing a Vulnerability

Figure 20 depicts an example of the case where a Vulnerability is removed from the scene, that is, some object undergoes a hardening process, so to speak. Since Vulnerability and Threat Capability are partner dispositions, the removal of the former results in the prevention of the associated Threat Events. Figure 20 describes a scenario of risks of exploitation of Vulnerabilities in a software code due to the lack of updates that users should perform. In this case, we can see that the Threat Agent (a hacker) must bear

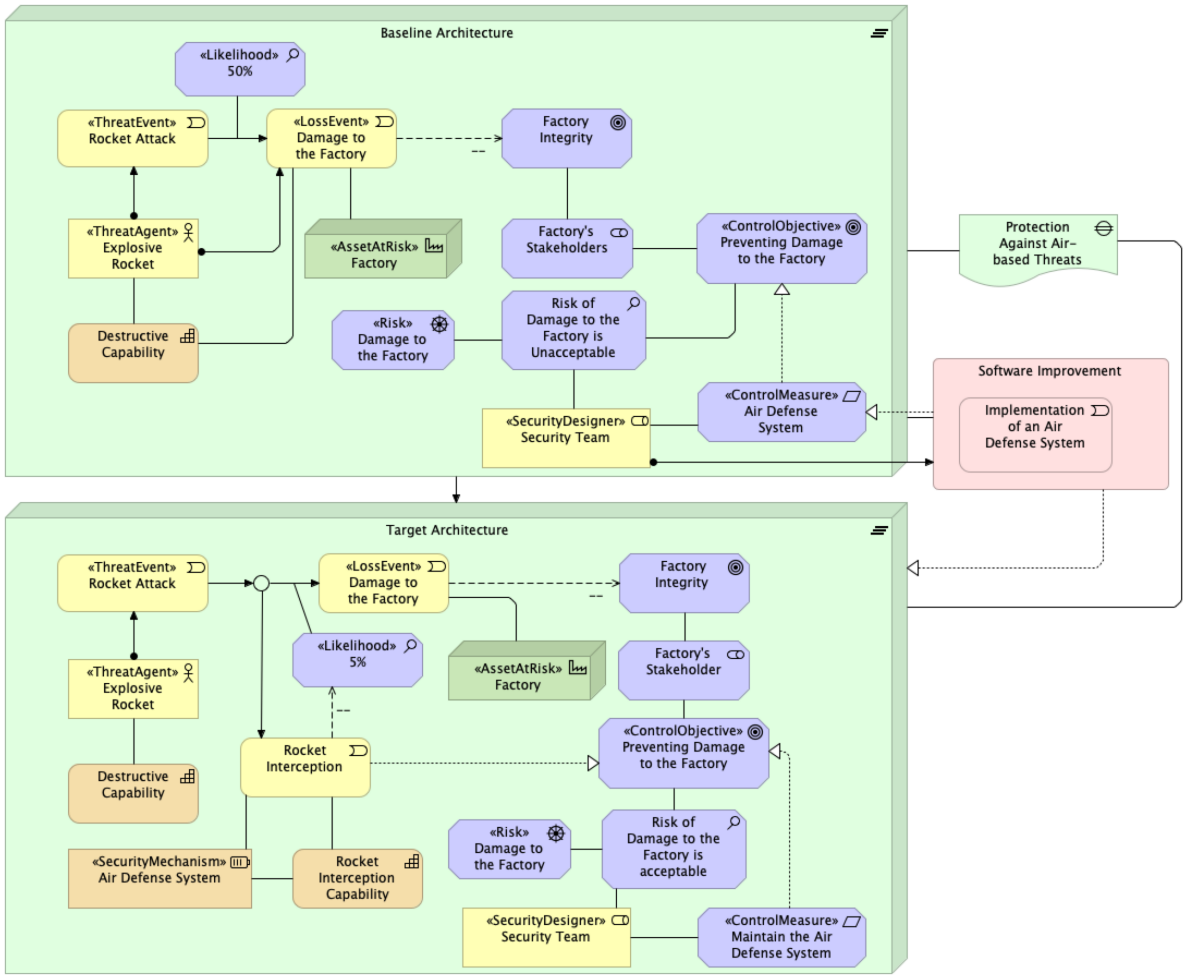

Fig. 18 Example of removal of the Threat Agent from the scene by the introduction of a Security Mechanism that is capable of destroying the agent and, therefore, its capabilities, so preventing the associated events

certain Intention (goal) and its Threat Capability, and they must meet the Vulnerability, so that Threat Events become possible with a given likelihood under the situation specified by «HarzardAssessment». The exploitation of vulnerabilities can eventually trigger data breaches. The IT Security Team understands this risk is unacceptable, then proposes autoupdates as a «ControlMeasure» that realizes the goal of preventing data breaches. Once the software code is changed accordingly, receiving an autoupdate feature, it loses the former weakness, which ultimately decreases the chances of data breaches.

7.5 Removing an Asset At Risk

Figure 21 depicts an example of the pattern where the very Asset At Risk is removed from the target architecture by the introduction of a new regulation (no-logs policy) as a Security Mechanism. The assets in this case are the logs produced by the customers of a VPN company.

Removing the Asset At Risk can potentially create other types of risks. Actually, any change in the baseline architecture promoted by Control Events or by the implementation of a Security Mechanism can create or increase other risks. Nevertheless, ROSE says that a particular Control Event prevents a particular type of Risk Events, so other risks are not touched unless addressed by other means. Moreover, the

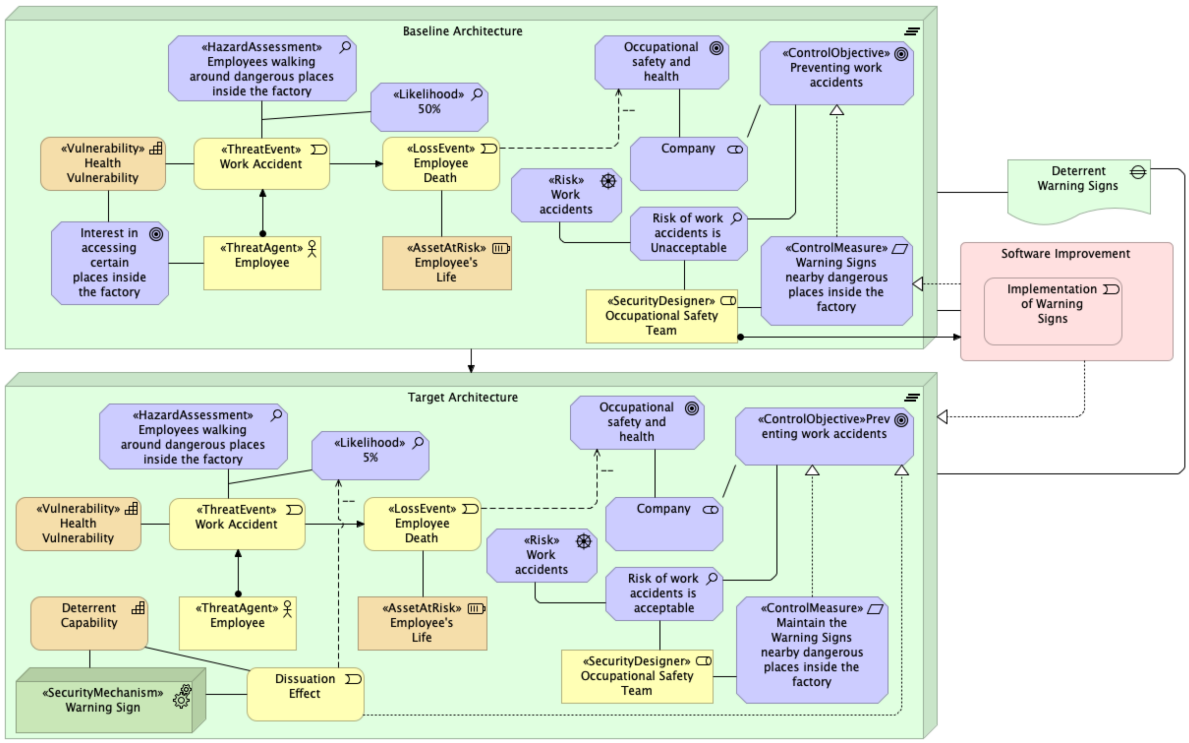

Fig. 19 Example of removal of the Threat Agent's Goal from the target architecture thanks to the introduction of a Security Mechanism that has a deterrent capability

very Security Mechanism may hold its own Vulnerabilities.

8 Evaluation: Representing Risk Treatment Options

As shown by our ontological analysis of the RSO, our proposal is clearly more expressive w.r.t. security modeling since it adds some domain-specific elements and explains how to make use of pre-existing elements of ArchiMate for this purpose (for instance, employing baseline and target architecture in the context of the implementation of a Security Mechanism).

As stated in Section 3.2, the *Reference Ontology for Security Engineering* (ROSE) takes into account the risk treatment options defined by ISO 31000 [11]. Our overlay, based on ROSE, inherits similar features, being able to represent those options in ArchiMate. The list below shows each option [2] and the description of its respective interpretations according to our proposal:

a) "avoiding the risk by deciding not to start or continue with the activity that gives rise to the risk": In this case, the motivation elements of ArchiMate can support statements that the risk is unacceptable; the target architecture should reflect the absence of risk-related entities displayed in the baseline architecture; it is possible that this case corresponds to the pattern of removing an Asset At Risk explained in Section 7.5;

b) "taking or increasing the risk in order to pursue an opportunity": Similarly, motivation elements of ArchiMate can state that the risk is acceptable; the target architecture may show a higher likelihood of certain Threat Events or Loss Events but some additional Value Objects.

c) "removing the risk source": As shown in [11], the notion of "risk source" is ambiguous since there can be multiple risk-related entities with this label. So this case may correspond to patterns of removing Threat Capabilities (Section 7.1), removing Threat Agents (Section 7.2), removing Vulnerabilities (Section 7.4), or even removing ancillary entities with partner dispositions;

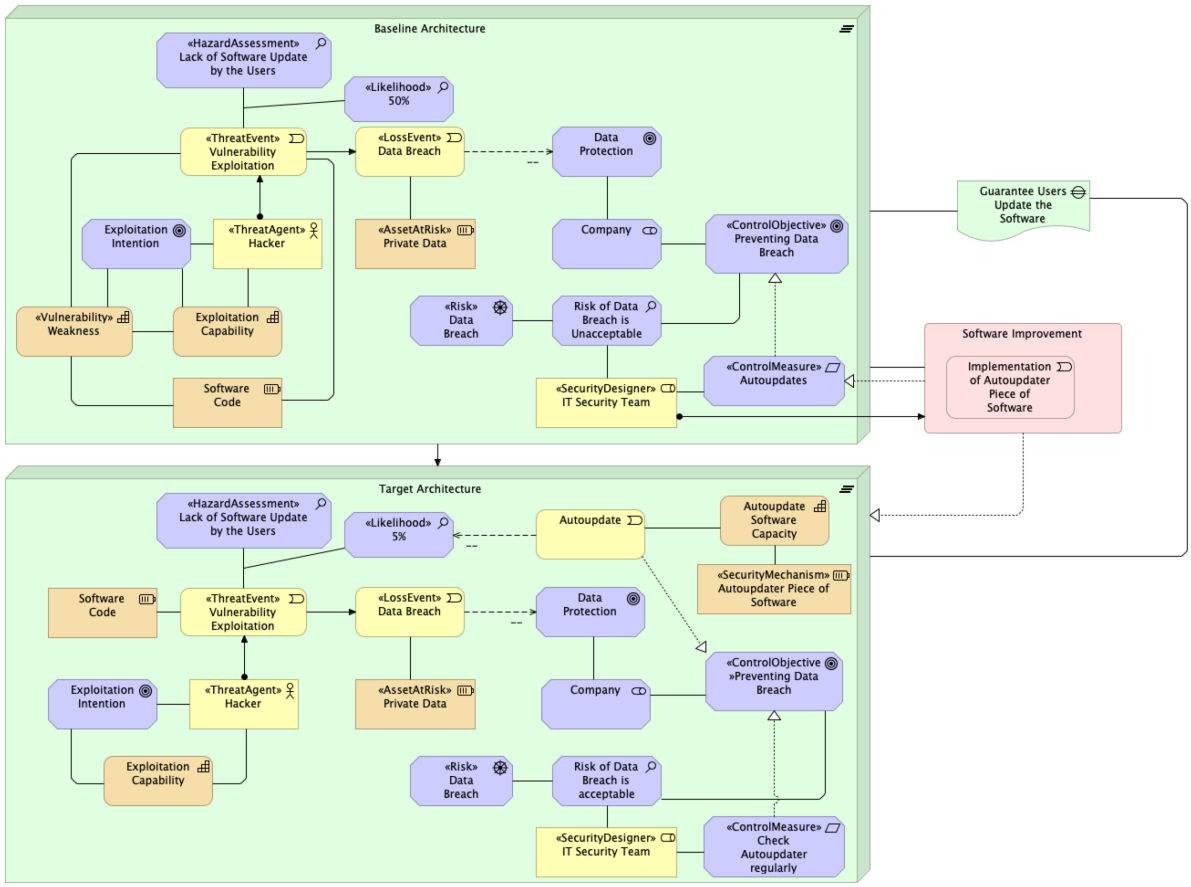

Fig. 20 Hardening, removing vulnerability. Example of removal of the Threat Agent from the scene by the introduction of a Security Mechanism that is capable of destroying the agent and, therefore, its capabilities, so preventing the associated events

d) "changing the likelihood": Thanks to the «Likelihood» construct, the representation of this case is straightforward, although not necessarily easy since there are multiple ways of assigning likelihood, as described in Section 6.1; more specifically, this case may refer to the likelihood of Threat Events;
e) "changing the consequences": This case is similar to the previous one but the change of likelihood now refers to Loss Events ("the consequences");
f) "sharing the risk with another party or parties (including contracts and risk financing)": This case implies that a given loss (say, production loss) triggers other losses (say, company bankruptcy). A Security Mechanism of this kind (say, insurance contracts) would be a countermeasure to those latter losses. Therefore, the representation of this case is a matter of choosing what are Threat Events and what are Loss Events.
g) "retaining the risk by informed decision": This case is simply the scenario where the risk is acceptable after the introduction of Security Mechanisms.

9 Illustrative Application: Security Breach

To illustrate our proposal in detail, we model a real-world security breach and the enterprise's reaction to it involving the LastPass password manager. Official blog posts, written by the company, support our description of the case.[5] Our

[5]The main sources describing what happened and LastPass's security reactions are the following blog posts: (1) https://blog.lastpass.com/2023/

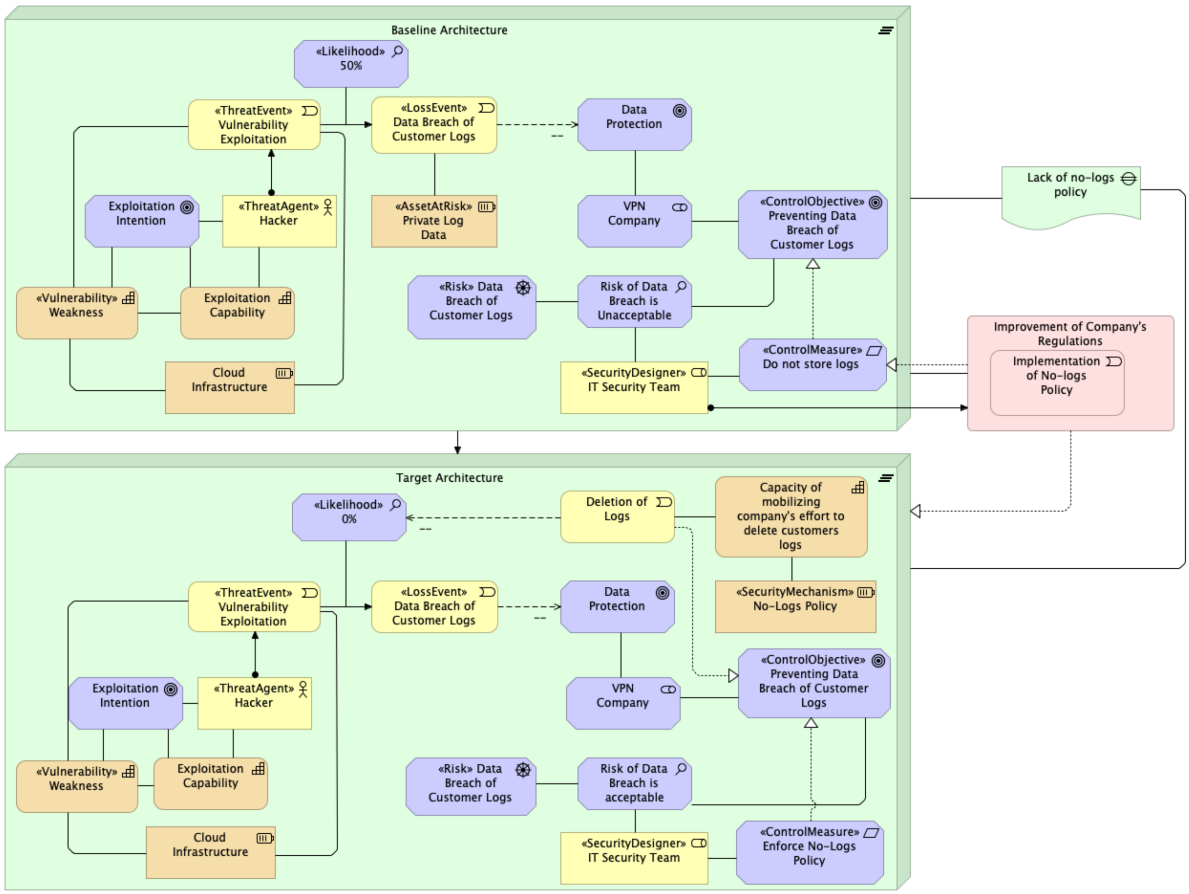

Fig. 21 Example of removal of the Asset at Risk from the scene by the introduction of a new regulation as a Security Mechanism. Therefore, the associated Loss Event cannot occur

model is not intended to cover the specific incident but a *type of incident* like this one.

In summary, according to the company, in August 2022 a software engineer's corporate laptop was compromised, allowing the unauthorized threat actor to gain access to a cloud-based development environment and steal source code, technical information, and certain LastPass internal system secrets. No customer data or vault data was taken during this incident, as there is no customer or vault data in the development environment. The information stolen in the first incident was used to identify targets and initiate the second incident, which is not addressed by our model. So we focus on the first incident. Moreover, the full description and representation of the incident are out of the scope of this paper, so we selected some important aspects of it. Given what happened, LastPass has implemented the following measures: (a) removed the development environment and rebuilt a new one to ensure full containment and eradication of the threat actor; (b) deployed additional security technologies and controls to supplement existing controls; (c) rotated all relevant cleartext secrets used by our teams and any exposed certificates. Our model details some of these measures and includes motivation elements, according to ArchiMate, as shown by Figure 22 and Figure 23.

In the baseline architecture, it is possible to see the chain of Threat Events that ultimately cause three different Loss Events (steal internal system secrets, technical documentation, and source code). The incident started when a Threat Agent gained access to the laptop

03/security-incident-update-recommended-actions/, (2) https://support.lastpass.com/help/incident-1-additional-details-of-the-attack.

to steal those Assets At Risk by using their hacking skills to exploit the laptop's Vulnerabilities. LastPass' reports do not specify how exactly the Threat Agent had access to the laptop but, for the sake of our study, let us say this happened through a Vulnerability of the employee's home network. A personal Control Measure this person implemented was hardening home networks by removing that Vulnerability, which corresponds to our pattern of Section 7.4. This is why the target architecture does not display the home network's Vulnerability anymore. Then, the attacker was able to use a third-party VPN to access the corporate network as if they were the employee. Although the corporate VPN was a Security Mechanism, it had its own Vulnerabilities that were exploited by threat actors. To mitigate this, the company implemented a more secure solution, ZTNA, allowing the employee to have secure remote access to the organization's applications. Inside the corporate network, the attacker had access to a cloud-based development environment and then they were able to achieve the targeted assets. The Control Event against this removed the access of engineers to the cloud platform. The aforementioned Loss Events impacted negatively the Password Manager Company's goal of maintaining confidentiality of internal code, documentation, and secrets. Although the implementation of the ZTNA solution is seen as more secure, it implies modifications in the architecture that can bring about new risks. The specific details regarding this change are, understandably, not specified by LastPass and, therefore, not addressed in our illustration. Finally, changes in the baseline architecture may have consequences for the compliance of regulatory requirements by the company, such as Zero Trust Architecture which is defined by NIST SP 800-207. This legal aspect, however, is outside the scope of LastPass' reports and our study.

10 Related Work

In [22], the authors analyzed the existing literature on Enterprise Risk Management frameworks, assessment models, and methods. Based on a systematic literature review, they found among 30 publications only one shows a conceptual model, which suggests a lack of attention towards this aspect in the field. Our paper addresses exactly this gap.

In [23], the authors carry out a systematic mapping study of the literature regarding the state-of-the-art surrounding incorporation of security aspects into enterprise architecture modeling languages, with a particular interest in the micro-mobility context. They identify a lack of research concerning the intersection of enterprise architecture modeling languages, security, and micro-mobility. According to them, only 14 primary papers were found; the vast majority highlight limitations in the existing security modeling, with ArchiMate being the most commonly used, but also the most criticized. The authors' conclusion states that there is a need for reference models for security aspects in ArchiMate, notably about transport and micro-mobility domains. Although our work does not address these specific areas, it does address the general gap regarding a reference model for security aspects in ArchiMate identified by [23].

In [24], the authors suggest that, from their experience cooperating with the Norwegian Armed Forces, there are two interconnected significant challenges for modeling risk and security in enterprise architecture: (1) modeling what is protected and why it is protected with sufficient detail whilst being simple enough to facilitate analysis, and (2) establishing automated support for analyzing and reasoning about the security models. In other words, a necessity for both an expressive modeling language and computational support attached to the resulting models. Our work provides contributions to the first challenge, whereas addressing the second one is among our future works.

A different version of the RSO, described in Section 4.1, was presented in a research paper [25], as shown in Figure 24. The risk and security concepts are linked to the phases of a typical Enterprise Risk and Security Management process, including an approach to show how the resulting model can be used as input for qualitative risk analysis and assessment of the impact of different control measures.

Some of the closest related works to ours are proposals for modeling Enterprise Risk Management and Security through ArchiMate, as seen by ArchiMate's Risk and Security Overlay. The

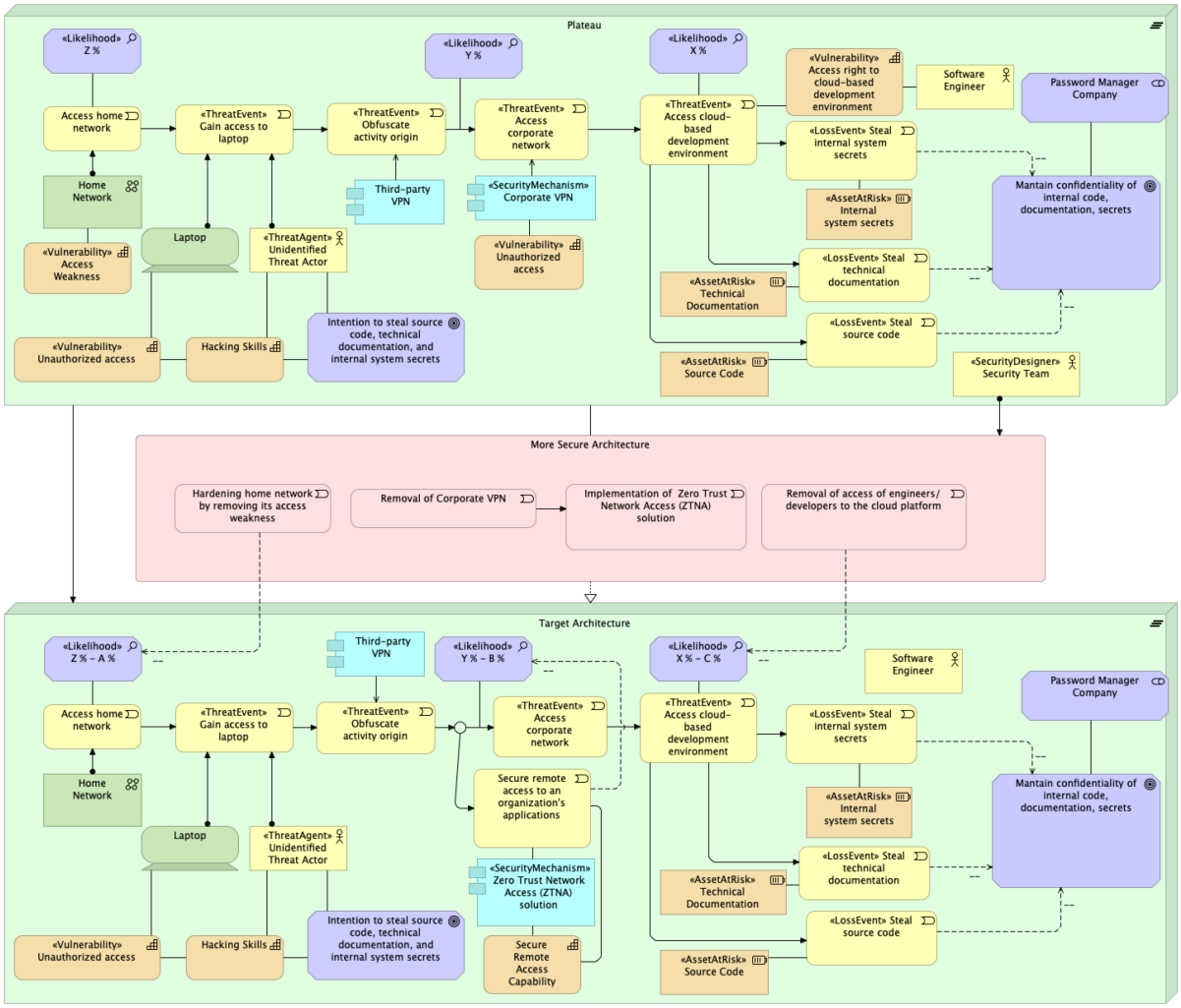

Fig. 22 A representation of incidents and security reactions of the type of LastPass's first incident in August 2022

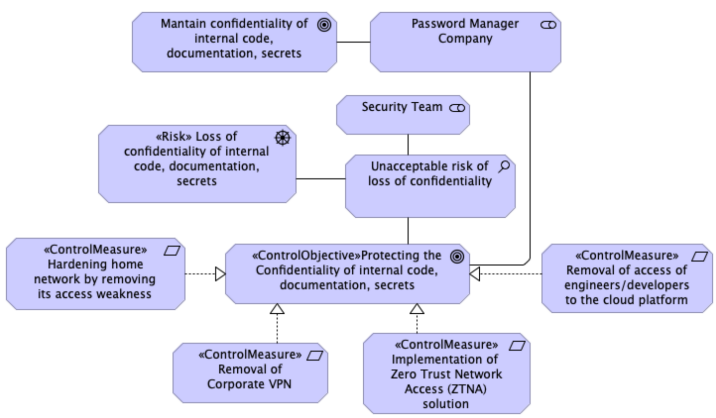

Fig. 23 Motivation layer containing security elements regarding LastPass's first incident in August 2022

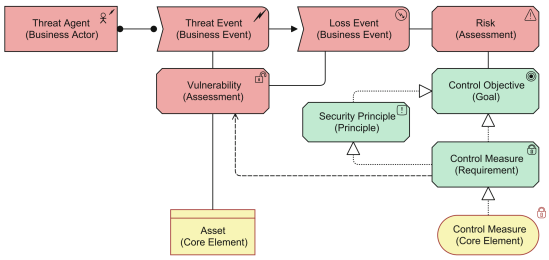

Fig. 24 Risk and security concepts as specializations of ArchiMate concepts, extracted from [25]

research conducted by Mayer and his collaborators [26] is one example of these proposals. They propose a conceptual model for Information System Security Risk Management, which is then integrated with enterprise architecture through ArchiMate's RSO. Their model contains four "risk treatment-related concepts": risk treatment, security requirement, and control. These concepts are mapped into the RSO metamodel without revision, which means that the problems we have shown remain untouched, such as construct redundancy and construct deficits.

Similarly, in [27], the authors make use of the works done by Mayer [26] to present how the concepts of an information system security risks management domain can be mapped into the ArchiMate enterprise architecture modeling language.

Another related proposal is the Master thesis by Sander van den Bosch [28]. Based on Zachman Framework and SABSA Model, he proposes a metamodel describing risk and security elements, which are the following: vulnerability, threat, risk, security mechanism, and security policy. Then he employs them to extend ArchiMate towards the "Secure Enterprise Architecture approach". The resulting language and the metamodel are validated by interviews with experts from both the enterprise architecture and the security discipline.

Teixeira et al. [29] goes in a similar direction, but it maps ISO 22301 and ISO 31000 concepts into ArchiMate concepts, then introduces risk and security concepts. For example, the concept *Risk Source* from ISO 31000 is defined as an "Element which alone or in combination has the intrinsic potential to give rise to risk". The authors understand that risk can come from every layer of the ArchiMate, and we can assume that all elements can be a source of risk, including Business Actor, Driver, and Resource. Although both proposals present interesting results, their analysis of security is not grounded in any well-founded ontology like ROSE, which is founded in UFO. As a consequence, their analysis suffers from a degree of informality, and certain modeling patterns and security elements are missing, such as the ones presented previously.

There is a white paper [30] that discusses how security architecture concepts can be expressed using ArchiMate by adding stereotypes to support the SABSA framework. This proposal seems to assume the official RSO and extends it further over different layers of ArchiMate, according to SABSA's needs. By doing so, it adds numerous other stereotypes, such as «Account», «Application Role», «Authorisation», «Credential», «Compliance Objective», «Standard», «Regulation». and so on. A full assessment of this interesting alignment between ArchiMate and SABSA is out of the scope of this paper, though we can mark that, once it assumes the RSO, it also inherits its ontological issues.

To improve the cybersecurity of critical infrastructure, the National Institute of Standards and Technology (NIST), in the United States, created a cybersecurity framework in 2014. It consists of five functions: identify, protect, detect, respond, and recover. The NIST Cyber Security Framework is known to be complex. Because of that, in the Master's thesis [31] the author introduces an enterprise architecture viewpoint that can assist organizations using enterprise architecture with the implementation of the NIST Cyber Security Framework. This proposal does not make use of security-specific stereotypes, except for some relations (v. g., «clean», «turns_off», «isolate»), and basically implements cybersecurity vocabulary through ArchiMate standard elements.

In [24], the authors suggest that, from their experience cooperating with the Norwegian Armed Forces, there are two interconnected major challenges for modeling risk and security in enterprise architecture: (1) modeling what is protected and why it is protected with sufficient detail whilst being simple enough to facilitate analysis, and (2) establishing automated support for analyzing and reasoning about the security models. In other words, a necessity for both an expressive modeling language and computational support attached

to the resulting models. Our work provides contributions to the first challenge, whereas addressing the second one is among our future works.

Lastly, there is a Master's thesis [32] that proposes an alignment between Mal-activity diagrams and ArchiMate in the context of Information System Security Risk Management, and another one [33] that compares the Secure Socio-Technical Systems models and ArchiMate's RSO.

11 Final Considerations

This paper considerably extends our previous work that analyses and redesigns security elements of ArchiMate [10]. We presented an ontologically-founded analysis of the security modeling fragment of ArchiMate's Risk and Security Overlay (RSO). This analysis, grounded in the *Reference Ontology for Security Engineering* (ROSE) [11], allowed us to clarify the real-world semantics underlying the security-related constructs of the overlay, as well as to unveil several deficiencies in its modeling capabilities, including both redundancy and deficit of constructs. We then addressed these issues by redesigning the security modeling aspects of the RSO, making it more precise and expressive. The proposed redesign supports the representation of several important elements of Enterprise Risk Management and security that the original RSO neglects, including ontology-based modeling patterns of Security Mechanism and Control Events, the subjects involved in it, the interdependence relations among risk entities, and the interaction between security and ArchiMate's baseline and target architecture. In doing so, we fill the gap left by a previous work that analyzed the risk and value aspects of ArchiMate [6, 19]. Among the elements of the novelty of this work, there is a detailed formulation of the ontology of prevention in ArchiMate, a list of ontology-based modeling patterns involving Control Events with numerous examples, an evaluation considering the expressiveness of our proposal w.r.t. risk treatment options of ISO 31000, an illustrative application, and an extended related work section.

Therefore, we expect to contribute to the ontology-based modeling of enterprise risk and security more comprehensively. In future work, we intend to provide support for computational simulations of scenarios in Enterprise Risk Management and security as well as address other aspects of security modeling, such as exception handling. Moreover, we plan to further validate our proposal by gathering systematic practitioners' feedback.

Acknowledgments. Work supported by Accenture Israel Cybersecurity Labs.

References

[1] Lankhorst M. Enterprise Architecture at Work: Modelling, Communication and Analysis. Springer; 2017.

[2] ISO.: ISO 31000:2018 - Risk management – Guidelines.

[3] The Open Group. Integrating Risk and Security within a TOGAF® Enterprise Architecture. The Open Group Guide white paper. 2019;.

[4] Band I, Engelsman W, Feltus C, Paredes SG, Hietala J, Jonkers H, et al. How to Model Enterprise Risk Management and Security with the ArchiMate Language. The Open Group; 2019. W172.

[5] The Open Group.: ArchiMate® 3.1 Specification. Available from: https://pubs.opengroup.org/architecture/archimate3-doc/.

[6] Sales TP, Almeida JPA, Santini S, Baião F, Guizzardi G. Ontological analysis and redesign of risk modeling in ArchiMate. In: 2018 IEEE 22nd International Enterprise Distributed Object Computing Conference (EDOC); 2018. p. 154–163.

[7] Guizzardi G. Ontological foundations for structural conceptual models. Enschede, The Netherlands: Telematica Instituut / CTIT; 2005.

[8] Sales TP, Baião F, Guizzardi G, Almeida JPA, Guarino N, Mylopoulos J. The Common Ontology of Value and Risk. In: Conceptual Modeling. ER 2018. vol. 11157. Cham: Springer; 2018. p. 121–135.

[9] Rosemann M, Green P, Indulska M. A Reference Methodology for Conducting Ontological Analyses. In: Conceptual Modeling. ER

2004. vol. 3288. Berlin, Heidelberg: Springer; 2004. p. 110–121.

[10] Oliveira Í, Sales TP, Almeida JPA, Baratella R, Fumagalli M, Guizzardi G. Ontological Analysis and Redesign of Security Modeling in ArchiMate. In: The Practice of Enterprise Modeling. PoEM 2022. vol. 456. Cham: Springer; 2022. p. 82–98.

[11] Oliveira Í, Sales TP, Baratella R, Fumagalli M, Guizzardi G. An Ontology of Security from a Risk Treatment Perspective. In: Conceptual Modeling. ER 2022. vol. 13607. Cham: Springer; 2022. p. 365–379.

[12] Dresch A, Lacerda DP, Antunes JAV. In: Design Science Research. Cham: Springer International Publishing; 2015. p. 67–102. Available from: https://doi.org/10.1007/978-3-319-07374-3_4.

[13] Azevedo CL, Iacob ME, Almeida JPA, van Sinderen M, Pires LF, Guizzardi G. Modeling resources and capabilities in enterprise architecture: A well-founded ontology-based proposal for ArchiMate. Information systems. 2015;54:235–262.

[14] Baratella R, Fumagalli M, Oliveira Í, Guizzardi G. Understanding and Modeling Prevention. In: International Conference on Research Challenges in Information Science. Springer; 2022. p. 389–405.

[15] Blomqvist E, Sandkuhl K. Patterns in ontology engineering: Classification of ontology patterns. In: International Conference on Enterprise Information Systems. vol. 4. SCITEPRESS; 2005. p. 413–416.

[16] Guizzardi G, Botti Benevides A, Fonseca CM, Porello D, Almeida JPA, Prince Sales T. UFO: Unified foundational ontology. Applied ontology. 2022;p. 1–44.

[17] Guizzardi G, de Almeida Falbo R, Guizzardi R. Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The case of the ODE Software Process Ontology. In: Ibero-American Conference on Software Engineering. CIbSE; 2008. p. 127–140.

[18] Mumford S. Dispositions. Clarendon Press; 2003.

[19] Sales TP, Roelens B, Poels G, Guizzardi G, Guarino N, Mylopoulos J. A Pattern Language for Value Modeling in ArchiMate. In: Advanced Information Systems Engineering. CAiSE 2019. vol. 11483. Cham: Springer; 2019. p. 230–245.

[20] Reason J. The contribution of latent human failures to the breakdown of complex systems. Philosophical Transactions of the Royal Society of London B, Biological Sciences. 1990;327(1241):475–484.

[21] Gangemi A, Presutti V. Ontology design patterns. In: Handbook on ontologies. Springer; 2009. p. 221–243.

[22] Fernandes AD, Ramalho D, Mira da Silva M. Enterprise Risk Management and Information Systems: a Systematic Literature Review. In: International Conference on Information Resources Management (CONFIRM). Association for Information Systems; 2022. .

[23] Ellerm A, Morales-Trujillo ME. Modelling security aspects with archimate: a systematic mapping study. In: 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE; 2020. p. 577–584.

[24] Grov G, Mancini F, Mestl EMS. Challenges for risk and security modelling in enterprise architecture. In: The Practice of Enterprise Modeling: 12th IFIP Working Conference, PoEM 2019, Luxembourg, Luxembourg, November 27–29, 2019, Proceedings 12. Springer; 2019. p. 215–225.

[25] Jonkers H, Quartel DAC. Enterprise Architecture-Based Risk and Security Modelling and Analysis. In: Kordy B, Ekstedt M, Kim DS, editors. Graphical Models for Security. vol. 9987. Cham: Springer; 2016. p. 94–101.

[26] Mayer N, Feltus C. Evaluation of the risk and security overlay of archimate to model information system security risks. In: 2017 IEEE 21st International Enterprise Distributed Object Computing Workshop (EDOCW). IEEE; 2017. p. 106–116.

[27] Grandry E, Feltus C, Dubois E. Conceptual integration of enterprise architecture management and security risk management. In: 2013 17th IEEE International Enterprise Distributed Object Computing Conference Workshops. IEEE; 2013. p. 114–123.

[28] van den Bosch SF. Designing secure enterprise architectures. A comprehensive approach: framework, method, and modelling language [Master's Thesis]. University of Twente. Enschede, The Netherlands; 2014.

[29] Almeida R, Teixeira JM, Mira da Silva M, Faroleiro P. A conceptual model for enterprise risk management. Journal of Enterprise Information Management. 2019;32(5):843–868.

[30] Bradley S. Modelling SABSA® with ArchiMate®. The SABSA Institute; 2021. T100.

[31] Hoogenboom C. An Enterprise Architecture Approach to Implementing the NIST Cyber Security Framework [Master's Thesis]. Leiden University. Leiden, The Netherlands; 2019.

[32] Tovstukha I. Management of Security Risks in the Enterprise Architecture using ArchiMate and Mal-activities [Master's Thesis]. University of Tartu; 2017.

[33] Artem Z. Comparison of STS and ArchiMate Risk and Security Overlay [Master's Thesis]. University of Tartu; 2018.