Free University of Bozen-Bolzano

Doctoral Thesis

An Ontological Approach to Security Modeling

Author:
Ítalo Oliveira

Supervisor:
Giancarlo Guizzardi,
Enrico Franconi,
Tiago Prince Sales

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science

KRDB research centre for Knowledge-based Artificial Intelligence
Faculty of Engineering

May 27, 2024

Declaration of authorship

I, Ítalo Oliveira, declare that this thesis titled, "An Ontological Approach to Security Modeling" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

There's no sense in being precise when you don't even know what you're talking about.

John von Neumann

FREE UNIVERSITY OF BOZEN-BOLZANO

Abstract

Computer Science
Faculty of Engineering

Doctor of Philosophy in Computer Science

An Ontological Approach to Security Modeling

by Ítalo Oliveira

Risk Management involves complex relations among objects and agents, their capabilities and vulnerabilities, the events they are involved in, and the value and risk they pose to the stakeholders. There are patterns involving these relations that crosscut many domains, ranging from information security to public safety. Understanding and forming a shared conceptualization and vocabulary about these notions is fundamental for modeling the corresponding scenarios and devising security countermeasures. Ontologies are instruments developed to address these issues of conceptual clarification and terminological systematization. Although several ontologies have been proposed over the years for risk management purposes, they display many limitations concerning their generality, expressivity, adequacy, and interoperability. To bridge this gap, we investigate those patterns with the support of the *Unified Foundational Ontology* (UFO) and the *Common Ontology of Value and Risk* (COVER). However, we immediately observe that the phenomenon of *prevention* is crucial to understanding and modeling the security domain. Prevention is about blocking an effect before it happens or stopping it as it unfolds. It may occur as a natural phenomenon or as a result of intentional human intervention– 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. Therefore, an ontological theory of prevention is necessary to build an adequate security ontology. Since the theory of events of UFO lacks a characterization of prevention, this happens to be our first challenge and contribution: (1) *an ontological theory of prevention* based on UFO. This theory will ground a (2) *Reference Ontology for Security Engineering* (ROSE), a proposed ontology of the security domain from a risk treatment perspective, according to ISO 31000. We report three other major contributions by applying ROSE to (3) specializing it in a *Phishing Attack Ontology* (PHATO); (4) proceeding with *an ontological analysis of D3FEND*, an OWL cybersecurity ontology; we uncover numerous modeling mistakes and propose recommendations of improvement; (5) executing *an ontological analysis and redesign of security elements of ArchiMate* in the context of Enterprise Risk Management. By doing so, through an ontological approach, we provide a network of novel solutions for security modeling.

Acknowledgements

We are obliged to be born[1]. It has been a long complicated journey for me but experientially wealthy. I am so grateful for all the people who have supported me and believed in me, even when I have not believed in myself. I had the opportunity to meet incredible and lovely people throughout my path.

I thank my supervisors Giancarlo Guizzardi, Enrico Franconi, and Tiago Prince Sales who taught me how to do research in computer science. By accepting me for the Ph.D. program at Unibz, they opened many opportunities for me and changed my life completely. And Ana Ozaki, who said "You can learn" to me when I doubted I could pursue this Ph.D.

I thank Glenda Amaral, Claudenir Fonseca, João Paulo A. Almeida, Cristine Griffo, and Isadora Valle Sousa for their help, discussions, and collaboration.

I am immensely grateful for the friendship of Francesco Corcoglioniti, Marina Andric, and Mohamed Sabri Hafidi. I will never forget our fantastic hikes in the Dolomites.

I am grateful to the adorable South Tyrolean Anna Kostner and her mother Ulli as well as to my former flatmates Thomas Borsani and Julien Frédéric Mertens. My Italian teacher, Barbara Calvi. People who made my period in Bozen much more enjoyable.

I cannot forget Theo Abgrall, Onur Yolcu, and Nony(elum) Ndefo. They were always supportive and kind to me.

My dear Israeli collaborators, Gal Engelberg and Dan Klein, from the Accenture Lab Israel. Mattia Fumagalli and Riccardo Bartella who, like me, jumped into computer science coming from another field.

I need to mention Fred Freitas, Cleyton Rodrigues, Torquato Castro Júnior, Ivan Vazinzack, and Ruy de Queiroz. They introduced me to the fascinating world of mathematical logic and knowledge representation before I dreamed about doing a Ph.D. in computer science when I was just a lawyer.

Of course, my mom Sibéria Maria de Oliveira, without whom I would not have reached anywhere since the beginning. ❤️

[1] Ólafur Arnalds, *Undone*.

Contents

List of Figures

List of Tables

List of Abbreviations

BFO	**B**asic **F**ormal **O**ntology
COSO	**Co**mmittee of **S**ponsoring **O**rganizations of the Treadway Framework
COVER	**C**ommon **O**ntology of **V**alu**e** and **R**isk
D3FEND	**D**etection, **D**enial, and **D**isruption **F**ramework **E**mpowering **N**etwork **D**efense
gUFO	**G**entle **U**nified **F**oundational **O**ntology
DL	**D**escription **L**ogics
DOLCE	**D**escriptive **O**ntology for **L**inguistic and **C**ognitive **E**ngineering
FAIR	**F**indability, **A**ccessibility, **I**nteroperability, and **R**eusability
HUFO	**Hu**man **F**actors **O**ntology
IDO	**I**nfectious **D**isease **O**ntology
IEC	**I**nternational **E**lectrotechnical **C**ommission
ISO	**I**nternational **O**rganization for **S**tandardization
MAP	**M**utual **A**ctivation **P**artnership
OSCO	**O**ntology of **S**ecure **C**yber **O**perations
OWL	**W**eb **O**ntology **L**anguage
PHATO	**Ph**ishing **A**ttack **O**ntology
RDF	**R**esource **D**escription **F**ramework
ROSE	**R**eference **O**ntology for **S**ecurity **E**ngineering
RSO	**R**isk and **S**ecurity **O**verlay of ArchiMate
SABES	**S**herwood **A**pplied **B**usiness **S**ecurity **A**rchitecture
SECCO	**Sec**urity **C**ore **O**ntology
TOGAF	**T**he **O**pen **G**roup **A**rchitecture **F**ramework
UFO	**U**nified **F**oundational **O**ntology
UML	**U**nified **M**odeling **L**anguage
W3C	**W**orld **W**ide **W**eb **C**onsortium

Dedicated to all those who no matter what keep finding something to fight for.

Chapter 1

Introduction

This thesis contributes to the definition of general ontological foundations for *security modeling*, that is, conceptual modeling activities within the security domain. In this chapter, Section 1.1 introduces the context regarding conceptual modeling and risk management. Section 1.2 motivates the relevance of our research in the context of security modeling. Section 1.3 describes our research objectives. Section 1.4 presents how we address these objectives through an ontological approach. Section 1.5 presents an overview of the thesis structure. Section 1.6 concludes with a list of publications.

1.1 Context: modeling risk and security

Conceptual Modeling is described by Mylopoulos, 1992 as "the activity of formally describing some aspects of the physical and social world around us *for purposes of understanding and communication*". Conceptual models are intended to be used by humans, not machines (Mylopoulos, 1992), although they can support computational tasks, such as data model design and automated reasoning. The need for conceptual models is, therefore, a human need to understand the environment by a form of intersubjective representation (very often a diagrammatic form). Because of that, conceptual models arise in different disciplines and domains, notably in Databases and Software Engineering, Artificial Intelligence, Information Systems Engineering, Object-Oriented Models, and Business Process Management. (Guarino, Guizzardi, and Mylopoulos, 2020)

As the outcomes of conceptual modeling activity, conceptual models can be seen as *information objects* that are explicit descriptions of subjective *mental models*. The former can be regarded as *abstractions* of sensory inputs according to an inventory of what there is in someone's view, i.e., according to a *conceptualization*. A (conceptual) *modeling language* allows for interpreting conceptualizations by describing mental models as conceptual models, concrete information objects, which are located in the public (intersubjective) arena of discourse. (Guarino, Guizzardi, and Mylopoulos, 2020)

Thus, the more a language signature interprets conceptualizations of a given domain, the better the produced domain conceptual models tend to be because the language users will be capable of making their mental models explicit. The *domain appropriateness* of a language is a measure of the suitability of a language to model phenomena in a given domain, that is, how accurate the language modeling primitives are about abstractions of a given domain reality. Figure 1.1 depicts relations between conceptualizations, mental models, conceptual models, and modeling languages. (Guizzardi, 2005)

In the area of *Risk Management*, conceptual models and conceptual modeling languages play an essential role: they allow for the evaluation, communication, and documentation of risks and their treatments. According to the ISO 31000 (ISO, 2018),

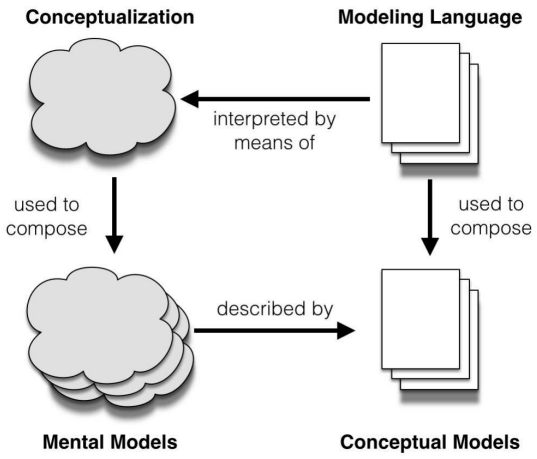

FIGURE 1.1: Relations between conceptualizations, mental models, conceptual models, and modeling languages (Guarino, Guizzardi, and Mylopoulos, 2020)

the *risk management process* (Figure 1.2) involves the systematic application of policies, procedures, and practices to the activities of communicating and consulting, establishing the context, and assessing, treating, monitoring, reviewing, recording, and reporting risk. *The purpose of risk management is the creation and protection of value.* (ISO, 2018) For that, it is necessary to formulate a proper conceptualization of risk and security entities, so that adequate modeling languages can be designed to support risk management activities. Complex relations among objects and agents, their capabilities and vulnerabilities, events and goals, assets and risks, security mechanisms, and safety measures, all that occurs transversely in multiple domains: information security, public safety, occupational safety, national security, financial risk management, aviation safety and security, and many others. Understanding and forming a shared conceptualization and vocabulary about these notions and relations is fundamental for modeling the corresponding scenarios and designing countermeasures.

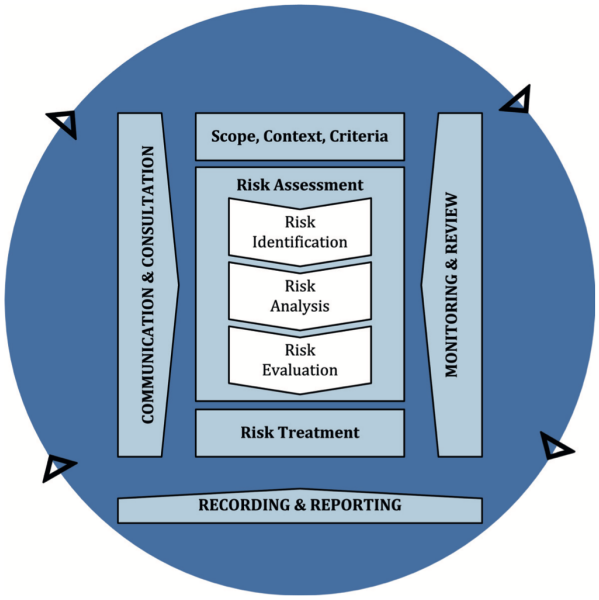

FIGURE 1.2: Risk Management Process (ISO, 2018)

Modeling approaches for risk management purposes include, among many others, the Goal-Risk framework (Asnar, Giorgini, and Mylopoulos, 2011) to support risk analysis in the context of requirements engineering; RiskML (Siena, Morandini, and Susi, 2014), which is an i*-based modeling language for representing risks intrinsic to the adoption of open-source software; the CORAS language (Lund, Solhaug, and Stølen, 2010) to describe threats, assets, scenarios, and treatments in an enterprise; Archimate (Band et al., 2015; Band et al., 2019), in which risks and security are expressed in the context of enterprise architecture models; bow-tie diagrams (Ruijter and Guldenmund, 2016; Saud, Israni, and Goddard, 2014) that describe an accident in terms of its initial causes, negative consequences, and safety barriers designed to prevent or control the associated hazards. The ISO/IEC 31010 (ISO/IEC, 2019) mentions more than 40 risk assessment techniques, all of which require, explicitly or implicitly, a conceptualization of risk and security entities. Hence, an important research question for risk management is: *How can we define an adequate conceptualization of risk and security?*

Ontologies are instruments developed to address exactly this kind of problem of *domain analysis*, *conceptual clarification*, and *meaning negotiation*. "An ontology is a formal, explicit specification of a shared conceptualization." (Studer et al, 1998) (For a formal development of this definition, we refer to Guarino, Oberle, and Staab, 2009). In general, information systems contain information structures representing abstractions of a given domain of interest. This means that all information systems make *ontological commitments* and the quality of an information system depends on how truthful its information structures are to the aspects of reality it purports to represent. (Guizzardi, 2020) Moreover, making information systems *interoperate* with one another requires identifying and preserving the (real-world) semantics of those information structures beyond syntactical resemblance. Because of that, the question "What does exist within risk and security?" has not only implications for understanding and modeling the domain but also for building information systems to support risk management activities.

To answer this sort of question, according to Guizzardi, 2005, a *reference domain ontology* should be constructed with the sole objective of making the best possible description of the domain in reality. In this sense, ontology engineering requires a study of what exists in a given domain or universe of discourse. In this view, ontologies should be more than knowledge representation artifacts. They should be *application-independent* and not biased toward a specific mathematical model or formal theory because what matters is to make domain shared conceptualizations explicit. Finally, such domain ontology should leverage the extensive philosophy literature on *Formal Ontology*[1], a field dedicated to formal aspects of objects irrespective of their particular nature (parthood theories, theory of relations, theory of types, theories of change, etc.). (Guizzardi, 2005)

Luckily, our investigation of risk and security domains can leverage three major tools that allow us to satisfy those requirements:

- The *Unified Foundational Ontology* (UFO) embodies several theories from Formal Ontology (taxonomic structures, a theory of events, social entities) and serves as a reference for building domain models; (Guizzardi, 2005; Guizzardi et al., 2022; Guizzardi, G. et al., 2013; Guizzardi et al., 2008)

[1] "For Husserl, therefore, formal ontology is the discipline that studies the formal constitution of the object in that it depends both on the possible unity of the object (elimination of counter-sense) and on the mere unity of sense (elimination of nonsense).". (Poli, 1993)

- The OntoUML general-purpose conceptual modeling language incorporates in its grammar many of the ontological distinctions of UFO, being an adequate choice for expressing an ontological conceptual model of risk and security; (Guizzardi, 2005)
- The *Common Ontology of Value and Risk* (COVER) is a reference domain model of risk and value represented in OntoUML, making it a perfect starting point for our investigation. (Sales et al, 2018; Sales, 2019)

As COVER deals with value and risk (what Figure 1.2 calls *Risk Assessment*), our research challenge involves reusing it to support a reference ontology of security capturing shared conceptualizations of *Risk Treatment*, according to ISO 31000 (see Figure 1.2, ISO, 2018). By doing so, we intend to add another piece of knowledge to the broader task of understanding and modeling risk management processes.

However, we have observed that a deep comprehension of the general phenomenon of *prevention* is essential to understanding and modeling the security domain. Prevention is about blocking an effect before it happens or stopping it as it unfolds. It may occur as a natural phenomenon or a result of intentional human intervention, 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. In other words, an ontological theory of prevention is necessary to build an adequate ontology of security. Since the theory of events of UFO lacks a characterization of prevention, this happens to be our first challenge and theoretical contribution:

1. An *ontological theory of prevention* based on UFO, explaining the dynamics of events that prevent other events from happening.
2. Grounded in this theory, we construct a *Reference Ontology for Security Engineering* (ROSE), an ontology of the security domain from a risk treatment perspective, according to ISO 31000 (ISO, 2018).
3. To show how ROSE can be useful to model more specific domains, we build a *Phishing Attack Ontology* (PHATO).
4. As a practical application of ROSE, we proceed with an *ontological analysis of D3FEND*, an OWL cybersecurity ontology; we uncover numerous modeling mistakes and propose recommendations for improvement.
5. A second practical application of ROSE is an *ontological analysis and redesign of security elements of ArchiMate* in the context of Enterprise Risk Management. By doing so, through an ontological approach, we provide a network of novel solutions for security modeling.

1.2 Motivation: the need for security modeling

The relevance of this research work relies on the very need for security modeling, that is, *the necessity of understanding and modeling security-related entities in the context of risk management*. Having a clear understanding of the ontological nature of security is fundamental for performing risk treatment, and even more for developing modeling languages to support it (for instance, ArchiMate language). Security is still a heavily overloaded and conceptually unclear notion, despite the wide number of efforts to properly characterize it, including several standardization efforts (for

instance, ISO 31000). Over the years, several ontologies have been proposed in risk management and security engineering but they fall short in many respects, including a lack of generality and expressivity, which affect their interoperability capability. (Oliveira et al., 2021)

In the context of cybersecurity, according to the Open Worldwide Application Security Project (OWASP), a *threat model*[2] (Xiong and Lagerström, 2019) is a representation of all the information that affects the security of an application. *Threat modeling* is the process of using hypothetical scenarios, system diagrams, and testing to help secure systems and data. By identifying vulnerabilities, helping with risk assessment, and suggesting corrective action, threat modeling helps improve cybersecurity and trust in key business systems. Threat modeling works to identify, communicate, and understand threats and mitigations within the context of protecting something of value. A reference domain ontology of security could be beneficial for the threat modeling processes as it should be capable of showing the multiple general ways through which value can be protected or risk can be managed. (Oliveira et al., 2022a)

This research work addresses a gap left by COVER (Sales et al, 2018), contributing to a more complete ontological description of the domain of risk management. Furthermore, although our proposed ontological theory of prevention based on UFO is needed due to the connection between prevention and security, it also establishes a ground for applications in other domains than security, such as bioinformatics and health sciences, in which the concept of prevention plays a major role.

1.3 Research objectives

The general problem of this research work is understanding and modeling security-related entities in the context of risk management. This can be phrased as the question "*How can we define an adequate reference domain ontology of security?*". As security involves the notion of prevention, our first problem is defining this concept adequately: *How can we formulate a general ontological theory of prevention?* A further theoretical development of our ontology of security will be an adequate reference domain ontology of phishing attacks. This allows us to show how our ontology can be tailored for more specific domains. Once we establish those ontological foundations for the security domain, we report two major practical applications: an ontological analysis of the D3FEND cybersecurity model (Kaloroumakis and Smith, 2021) with recommendations for improvements, and an ontological analysis and redesign of the ArchiMate's Security Overlay (Band et al., 2019). Each chapter presents subquestions associated with the main general problem and goals. In summary, we define the following research objectives:

1. By identifying the state of the art in security ontologies, we intend to motivate and support our proposed security ontology.

2. By proposing a general ontology of prevention based on UFO, we intend to ground our proposed security ontology.

3. By proposing a reference domain ontology of security from a risk treatment perspective (termed *ROSE*), according to ISO 31000, we address the main research problem of understanding and modeling security.

[2]The Open Worldwide Application Security Project (OWASP): https://owasp.org/www-community/Threat_Modeling.

4. By proposing a reference domain ontology of phishing attacks specializing ROSE's concepts, we show how ROSE can be tuned for more specific domains.

5. Based on ROSE, proceeding with an ontological analysis of the D3FEND cybersecurity model plus recommendations for improvements. This shows ROSE can support the evaluation of information artifacts within the security domain.

6. Proceeding with an ontological analysis and redesign of security-related elements of the ArchiMate's Risk and Security Overlay. This shows how ROSE can support an ontologically sound redesign of domain-specific languages related to the security domain.

1.4 Methodology: an ontological approach

Following the guidelines of the design science methodology (Hevner et al., 2004; Dresch, Lacerda, and Antunes, 2015), we propose (a) concrete artifacts, (b) to support technology-based solutions to important business problems; (c) both the construction and (d) the evaluation of the artifacts must follow systematic and theoretically sound methods (ontology-driven conceptual modeling). Finally, (e) the deliverable must be communicated to reach stakeholders.

To achieve those research objectives, we employ a combination of research methods, having the design science methodology underlying the whole thesis: (1) a *systematic mapping study*, following a protocol described by Petersen et al, 2008, to spot the state of the art in security ontologies; (2) an *ontological approach*, in the sense that we rely on *ontology-driven conceptual modeling* tools (Verdonck, M. et al., 2015; Verdonck and Gailly, 2016; Verdonck, M. et al., 2019), to construct well-founded reference domain ontologies; more specifically, we leverage the upper ontology UFO (Guizzardi et al., 2022; Guizzardi, 2005) and the reference domain ontology COVER (Sales et al, 2018) to support our conceptual models, which are represented through OntoUML language; (3) to evaluate the proposed artifacts, show their applications, as described in Chapter 7 and Chapter 8.

By an ontological approach, we also mean exploiting the method of *ontological analysis* (Rosemann, Green, and Indulska, 2004; Guizzardi, 2005) to evaluate and redesign other ontologies, following the design science methodology. One of the key success factors behind using a modeling language is its ability to provide its target users with a set of modeling primitives that can directly express important domain abstractions. (Guizzardi, 2005) 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". (Rosemann, Green, and Indulska, 2004) Ideally, according to Rosemann, Green, and Indulska, 2004, 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.

The general idea is to compare two ontologies (as descriptions of a domain or metamodels), assuming one is the reference to assess the other. As shown by Guizzardi, 2007a, this sort of analysis is more than a matter of direct comparison between the actual structures of these models; it is a matter of reconstructing the underlying intended conceptualizations of these models, i.e., about making their ontological commitments explicit.

a) *Ontological incompleteness* (or *construct deficit*), which is the lack of a grammatical construct for an existing ontological concept; when there is an element in the reference ontology that finds no representation in the evaluated ontology; a special case of ontological incompleteness is *under-specification*: when missing domain constraints allow for unintended models of the ontology.

b) *Construct overload*, which occurs when one grammatical construct represents more than one ontological construct; when two disjoint notions in the reference ontology are represented by the very same element in the evaluated ontology

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. (Rosemann, Green, and Indulska, 2004; Guizzardi, 2005)

With the support of this framework, summarized in Figure 1.3, we will analyze the D3FEND cybersecurity model in Chapter 7 and identify shortcomings concerning the security modeling capability of the Risk and Security Overlay of ArchiMate in Chapter 8, which also goes into details about the design science methodology.

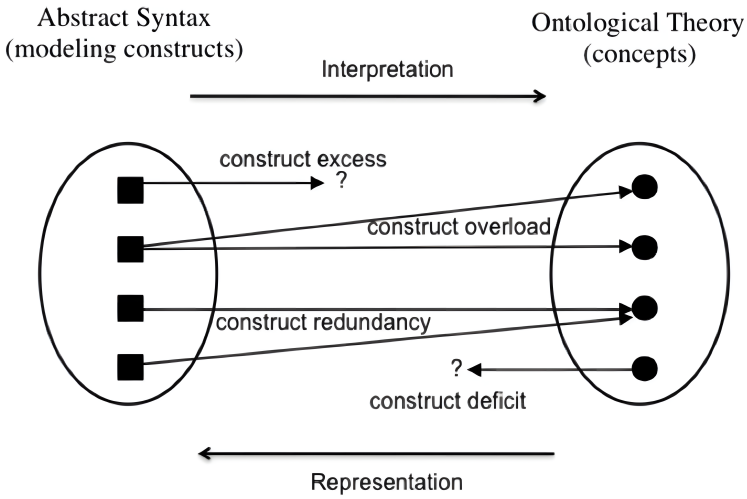

FIGURE 1.3: The illustration of the relation between modeling constructs in a language's syntax and ontological concepts (Azevedo, C. et al., 2015)

In a nutshell, we address the problem of understanding and modeling the security domain by designing adequate artifacts based on ontological foundations after systematically analyzing the literature.

1.5 Thesis structure

All chapters with contributions are based on peer-reviewed publications in major conferences or journals in conceptual modeling, formal ontology, and enterprise modeling. The original papers have been reviewed and improved. They tell a coherent story and form a monograph, which is divided into four parts:

1. **Baseline**: Chapter 2, Chapter 3;

2. **Ontological Foundations for Security**: Chapter 4, Chapter 5, Chapter 6;
3. **Practical Applications**: Chapter 7, Chapter 8;
4. **Conclusion**: Chapter 9, including **Appendix A** and **Appendix B**.

The thesis structure regarding each chapter, research question, and objective is the following:

- Chapter 2 presents a systematic literature review of security ontologies. It satisfies objective 1 by examining the state of the art of security ontologies. It is based on Oliveira et al., 2021.
- Chapter 3 concerns ontological foundations, so it does not contain novel contributions.
- Chapter 4 presents an ontological theory of prevention based on UFO. In particular, it extends the theory of perdurants of UFO, satisfying objective 2. It is based on Baratella et al., 2022.
- Chapter 5 presents a *Reference Ontology for Security Engineering* (ROSE). It addresses satisfies objective 3. It is based on Oliveira et al., 2022a.
- Chapter 6 presents a *Phishing Attack Ontology* (PHATO). It addresses satisfies objective 4. It is based on Oliveira, Calhau, and Guizzardi, 2023.
- Chapter 7 reports an ontological analysis of D3FEND cybersecurity model and recommendations to improve it. It addresses satisfies objective 5. It is based on Oliveira et al., 2023.
- Chapter 8 reports an ontological analysis of ArchiMate security-related elements. It addresses satisfies objective 6. It is based on Oliveira et al., 2022b and Oliveira et al., 2024.
- Chapter 9 presents final considerations and future works.
- Appendix A lists repositories and resources related to each contribution.
- Appendix B is a glossary of the novel notions introduced in this research work.

1.6 Publications

In this section, we list all the works we published during the period of the doctoral research. In the Core Publications section, we list those that directly contribute to this thesis, whilst in the Additional Publications section, those that do not.

1.6.1 Core publications

In peer-reviewed journal

- Oliveira, Í., Sales, T.P., Almeida, J.P.A., Baratella, R., Fumagalli, M., Guizzardi, G. (2024). Ontology-based Security Modeling in ArchiMate. Software and Systems Modeling. https://doi.org/10.1007/s10270-024-01149-1

In peer-reviewed international conference proceedings

- Oliveira, Í., Fumagalli, M., Prince Sales, T., Guizzardi, G. (2021). How FAIR are Security Core Ontologies? A Systematic Mapping Study. In: Cherfi, S., Perini, A., Nurcan, S. (eds) Research Challenges in Information Science. RCIS 2021. Lecture Notes in Business Information Processing, vol 415. Springer, Cham. https://doi.org/10.1007/978-3-030-75018-3_7
- Baratella, R., Fumagalli, M., Oliveira, Í., Guizzardi, G. (2022). Understanding and Modeling Prevention. In: Guizzardi, R., Ralyté, J., Franch, X. (eds) Research Challenges in Information Science. RCIS 2022. Lecture Notes in Business Information Processing, vol 446. Springer, Cham. https://doi.org/10.1007/978-3-031-05760-1_23
- Oliveira, Í., Sales, T.P., Baratella, R., Fumagalli, M., Guizzardi, G. (2022). An Ontology of Security from a Risk Treatment Perspective. In: Ralyté, J., Chakravarthy, S., Mohania, M., Jeusfeld, M.A., Karlapalem, K. (eds) Conceptual Modeling. ER 2022. Lecture Notes in Computer Science, vol 13607. Springer, Cham. https://doi.org/10.1007/978-3-031-17995-2_26
- Oliveira, Í., Sales, T.P., Almeida, J.P.A., Baratella, R., Fumagalli, M., Guizzardi, G. (2022). Ontological Analysis and Redesign of Security Modeling in ArchiMate. In: Barn, B.S., Sandkuhl, K. (eds) The Practice of Enterprise Modeling. PoEM 2022. Lecture Notes in Business Information Processing, vol 456. Springer, Cham. https://doi.org/10.1007/978-3-031-21488-2_6
- Oliveira, Ítalo., Engelberg, G., Barcelos, P.P.F., Sales, T.P., Fumagalli, M., Baratella, R., Klein, D., Guizzardi, G., (2023) Boosting D3FEND: Ontological analysis and recommendations. In: Formal Ontology in Information Systems: Proceedings of the Thirteenth International Conference (FOIS 2023). Vol. 377. Frontiers in Artificial Intelligence and Applications. IOS Press. https://ebooks.iospress.nl/doi/10.3233/FAIA231138

In peer-reviewed international workshops

- Oliveira, Ítalo, Calhau, R. F., Guizzardi, G. (2023). Toward a phishing attack ontology. In: ER2023: Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium, November 06-09, 2023, Lisbon, Portugal. https://ceur-ws.org/Vol-3618/forum_paper_25.pdf

1.6.2 Additional publications

In peer-reviewed journal

- Mário de Oliveira Rodrigues, C., Bezerra, C., Freitas, F. and Oliveira, I., 2020. Handling Crimes of Omission by reconciling a criminal core ontology with UFO. Applied Ontology, 15(1), pp.7-39. https://doi.org/10.3233/AO-200223

In peer-reviewed international conference proceedings

- Calhau, R.F., Prince Sales, T., Oliveira, Í., Kokkula, S., Ferreira Pires, L., Cameron, D., Guizzardi, G. and Almeida, J.P.A. (2024). A System Core Ontology for Capability Emergence Modeling. In: Proper, H.A., Pufahl, L., Karastoyanova, D., van Sinderen, M., Moreira, J. (eds) Enterprise Design, Operations, and Computing. EDOC 2023. Lecture Notes in Computer Science, vol 14367. Springer, Cham. https://doi.org/10.1007/978-3-031-46587-1_1

- Fumagalli, M., Engelberg, G., Sales, T.P., Oliveira, Í., Klein, D., Soffer, P., Baratella, R. and Guizzardi, G. (2023). On the Semantics of Risk Propagation. In: Nurcan, S., Opdahl, A.L., Mouratidis, H., Tsohou, A. (eds) Research Challenges in Information Science: Information Science and the Connected World. RCIS 2023. Lecture Notes in Business Information Processing, vol 476. Springer, Cham. https://doi.org/10.1007/978-3-031-33080-3_5

Part I

BASELINE

Chapter 2

A systematic mapping study of security ontologies

Recently, ontology-based approaches to security have been recognized as a relevant challenge and an area of research interest of its own. As the number of ontologies about security grows for supporting different applications, semantic interoperability issues emerge. Relatively little attention has been paid to the ontological analysis of the concept of security understood as a broad application-independent security ontology. *Core* (or *reference*) ontologies of security cover this issue to some extent, enabling multiple applications crossing domains of security (information systems, economics, public health, crime *etc.*). In this chapter, we investigate the current state-of-the-art on *Security Core Ontologies*. We select, analyze, and categorize studies on this topic, supporting a future ontological analysis of security, which could ground a well-founded security core ontology. Notably, we show that (a) most existing ontologies are not publicly findable or accessible; (b) foundational ontologies are under-explored in this field of research; and (c) there seems to be no common ontology of security. From these findings, we make the case for the need for a FAIR Core Security Ontology.

This chapter is based on the following paper:

- Oliveira, Í., Fumagalli, M., Prince Sales, T., Guizzardi, G. (2021). How FAIR are Security Core Ontologies? A Systematic Mapping Study. In: Cherfi, S., Perini, A., Nurcan, S. (eds) Research Challenges in Information Science. RCIS 2021. Lecture Notes in Business Information Processing, vol 415. Springer, Cham. https://doi.org/10.1007/978-3-030-75018-3_7.

2.1 Introduction

Security concerns are pervasive in society across different contexts, such as economics, public health, criminology, aviation, information systems, and cybersecurity, as well as international affairs. In recent years, multiple ontologies about security have been developed with the main goal of supporting different kinds of applications, such as the simulation of threats and risk management. Covering multiple application areas, *security ontologies* deal with many kinds of core and cross-domain concepts such as risk, asset, threat, and vulnerability (Kovalenko et al, 2018). An example of the current worries about security and, in particular, information security is the open letter addressed to the United Nations by the World Wide Web Foundation[1]. As the interest in security and related applications grows, the need for a rigorous analysis of the already existing resources and related concepts increases, with the main goal of enabling ontologies for information structure design and reuse. However, because of

[1] See https://webfoundation.org/2020/09/un-trust-and-security-letter/.

the different applications, the multiplicity of existing security ontologies dealing with different aspects of this domain brings back the issues of *semantic interoperability*, *domain understanding* and *data and model reusability*, suggesting the need for a common view, i.e., an explicit agreement about the semantics of the concepts therein. Core ontologies are intended to provide a solution to these problems, addressing to some extent the question of the general ontology of a given domain.

To better understand and organize the core ontologies of security state-of-the-art, we carry out a systematic mapping study by following the guidelines of Petersen et al, 2008. Our contribution is a mapping of the literature about this type of ontology, selecting and categorizing the papers, then identifying research gaps. In particular, we are interested in investigating how much the existing Security Core Ontologies abide by the FAIR principles (Jacobsen et al., 2020), i.e., how *Findable*, *Accessible*, *Interoperable* and *Reusable* are they?

This output is expected to be the basis of future research towards an ontological analysis of security, the development of a common ontology of security, and the development of an ontology-based security modeling language. The enterprise of building a general security ontology is a well-known open challenge in the field (Blanco et al, 2011). Indeed, the need for security ontology (rather than just taxonomy of security terms) was already recognized more than two decades ago (Donner, 2003).

This chapter is structured as follows. Section 2.2 establishes some definitions according to the literature; section 2.3 presents related work. Section 2.4 describes the process we followed in our mapping study; Section 2.5 presents the outcomes of our analysis; Section 2.6 briefly discusses some results. Section 2.7 marks some threats to the validity of this study. Section 2.8 finishes the chapter by discussing the main conclusions and prospects for future work.

2.2 Terminological remarks on ontology

The term "ontology" is semantically overloaded. In philosophy, ontology is concerned with "what there is", i.e., with the nature and characteristics of the categories of entities that are assumed to exist by some theory (Quine, 1948).

In Computer Science, "ontology" has several different meanings (Roussey et al, 2011), but one often cited definition is that "an ontology is a formal, explicit specification of a shared conceptualisation" (Studer et al, 1998). Obviously, each term in the *definiens* requires further elaboration; for that, we refer the reader to Guizzardi, 2007b. The notion of conceptualization is useful here because it implies a broader view of ontology: the things forming a conceptualization of a given domain are used to articulate abstractions of a certain state of affairs (Guizzardi, 2007b); a conceptualization is a sort of abstract model of some phenomenon in the world, identifying the relevant concepts and relations of that phenomenon (Studer et al, 1998). So, a definition that is not far from the original philosophical one. Adopting this view, we then are going to consider ontology as whatever expresses such a conceptualization for security at a general level, regardless of the language in which this conceptualization is expressed, i.e., this might be (a) a *conceptual model*, made in a conceptual modeling language (e.g., UML) or just stated in natural language, describing the entities and relations in the domain; (b) a *formal specification* of this conceptual model (for example, in a form of a set of description logic axioms); (c) the *executable information artifact* of this specification (a *Web Ontology Language* file, for example). These three meanings are interrelated and of interest here because we are aiming at surveying works that present core security ontologies in any of these senses.

Ontologies have different scopes or domain granularities. The broader their scope, the more generic their concepts. A *foundational ontology*, aka "upper ontology" or "top-level ontology", intends to establish a view of the most general aspect of reality, such as events, processes, identity, part-whole relation, individuation, change, dependence, causality *etc.* Examples include the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE), the Basic Formal Ontology (BFO), and the Unified Foundational Ontology (UFO). Foundational Ontologies offer key support in the development of high-quality core and domain ontologies, improving their consistency and interoperability (Guizzardi, 2006). *Core reference ontologies* are built to grasp the central concepts and relations of a given domain, possibly integrating several domain ontologies and being applicable in multiple scenarios (Roussey et al, 2011). The terms "core ontology", "reference ontology" and "core reference ontology" or even "common ontology" often denote the same type of artifact (Zemmouchi-Ghomari et al, 2009). In our context, this kind of ontology, implicitly or explicitly, deals with security-related concepts and relations across numerous domains of applications. Both foundational ontologies and core ontologies are application-independent, but the former are domain-independent as well.

2.3 Related work

The first systematic literature review on security ontologies was published by Blanco et al, 2008. The authors highlight that building ontologies in the information security domain is an important research challenge. They identified that most works were focused on specific application domains, and were still at the early stages of development, lacking the available source files of the security ontologies. They concluded that the security community at that time needed a complete security ontology able to provide reusability, communication, and knowledge sharing. More than a decade has passed since the publication of that study, so we can verify whether some of its conclusions still hold.

A review made by Sicilia et al, 2015, focused on information security ontologies that were published between 2014 and June 2015, which is a rather narrow period of analysis. review and categorize information security ontologies in the same way as Blanco et al, 2011. The former covers the period between 2004 and 2014, and it does not follow a systematic methodology. The latter is a systematic literature review, more aligned with our investigation, though it was made ten years ago; Blanco et al, 2011 noticed at that time that the majority of security ontologies were focused on formalizing concrete domains to solve a specific problem.

Sikos, 2019 collects and describes OWL ontologies in cybersecurity, including what he calls "Upper Ontologies for Cybersecurity", which is analogous to what we call core reference security ontologies. Implementations of security ontologies in other languages were not part of that analysis.

Meriah et al, 2020 propose a new classification of information security ontologies: (a) ontology-based security standards and (b) ontology-based security risk assessment. The goal of their analysis is specifically to support security stakeholders' choice of the appropriate ontology in the context of security compliance and risk assessment in an enterprise.

Ellerm et al, 2020 did a mapping study on security modeling in the context of Enterprise Architecture; they concluded there exists a necessity for reference models, security standards, and regulations in the context of micromobility to enable accurate and effective representation through modeling languages. Here, among other things,

we make a case for a similar conclusion about security in a broader context (i.e., beyond micromobility).

As we see, these useful reviews have some limitations, some of which we intend to address in this work. More importantly, *there is hitherto no mapping study exclusively on security core ontologies.* A reference ontology of security (in the sense discussed in Section 2.2 but also in the the same sense as Griffo, 2015 for legal relations, Sales et al, 2018 for Value and Risk, and Nardi et al, 2015 for Service), that is, applicable to several security sub-domains has yet to be proposed.

2.4 Methodology

Our procedure is linear and follows the guidelines of Petersen et al, 2008 for systematic mapping studies in software engineering, depicted in Figure 2.1:

1. **Research Questions:** We define and justify a set of input research questions, which give us the review scope, including inclusion-exclusion criteria;
2. **Search Procedures:** We carry out the searches, defining the total amount of papers;
3. **Screening of the Studies:** We screen them to define solely the relevant papers;
4. **Classification Scheme:** We analyze certain parts of the relevant papers (keywords, abstract, introduction *etc.*) aiming to formulate categories for classifying the papers;
5. **Results:** We finally gather the data, then produce a landscape of reference ontologies of security - described in the results section 2.5.

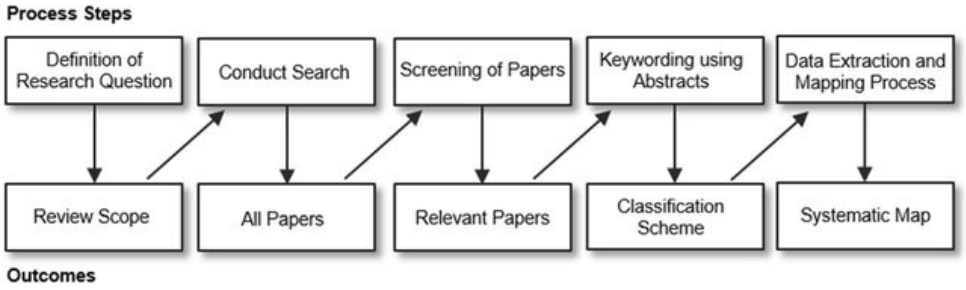

FIGURE 2.1: Systematic mapping process proposed by Petersen et al, 2008

In this chapter, when we talk about the object of our investigation, we use the terms “work”, “paper”, “study” and “research” interchangeably.

2.4.1 Research questions

Our study is driven by the following Research Questions (RQ) that define its scope:

1. *Which security core ontologies exist in the literature?*
2. *Which languages have been used to represent the core ontologies of security?*

3. *Are the specifications of the security core ontologies publicly available? If so, in which source (URL)?*

4. *Which foundational ontologies have been used in the design of security core ontologies?*

5. *Which terms appear most often in the core ontologies of security?*

RQ2 and **RQ4** directly speaks to the topic of *interoperability* (Guizzardi, 2020); **RQ3** to *findability*. *Accessibility* is indirectly assessed through findability, as the absence of the latter blocks the possibility of the former. Analogously, *reusability* is indirectly assessed through *interoperoperality* and, hence, **RQ4** (concerning the need for having rich meta-data about domain-related terms (Jacobsen et al., 2020)) but it is also related to **RQ2**, as the use of standard languages can foster the reusability of models. **RQ1** defines the space of models of our analysis and **RQ5** the space of concepts. Through **RQ5** we also take the first steps toward a common conceptualization of the domain of security.

Given the listed RQs, we define explicit inclusion and exclusion criteria. The final collection of papers is defined by the studies that, *simultaneously*, suffice every inclusion criterion, *and* that do *not* satisfy any exclusion criteria.

Inclusion Criteria

1. Studies whose goals include introducing an ontology in at least one of the three senses we defined in section 2.2: conceptual models expressed in any form, formal specifications, and executable information artifacts - each describing a general conceptualization of the security domain.

2. Studies presenting a security ontology that can be seen, at least partially, as a core reference ontology, that is, an application-independent ontology describing the general concepts and relations of the domain (Roussey et al, 2011), and thus, could be reused for different types of application.

3. Studies published in the last twenty years, that is, between 2000 and 2020 (included).[2]

Exclusion Criteria

1. Studies presenting application-based or microdomain ontologies of security; for example, an ontology method to solve the heterogeneity issues in a layered cloud platform (Tao et al, 2018).

2. Studies that are available solely in abstracts or slide presentations.

3. Publications that are not available in English.

4. Works about "ontological security", defined in *international relations studies* as "the need to experience oneself as a whole, continuous person in time - as being rather than constantly changing - in order to realize a sense of agency" (Mitzen, 2006).

5. Studies on security ontology as a philosophical issue. Though they should be useful for future ontological analysis of security-related notions, our work here is focused on the core ontology of security as information artifacts.

[2]Indeed, our searches suggest there is almost no ontology-based study about security before 2000.

2.4.2 Search procedures

Considering the RQs, in November 2020, we made several queries to the following databases, according to the search strings shown below in the exact described form: Web of Science, DBLP, ACM Digital Library, Science Direct, IEEE Xplore, Google Scholar, and Scopus. Here, the comma denotes different queries.

To formulate the search strings we assume a sort of "gold standard" based on our previous knowledge about studies that must be retrieved, such as Agrawal, 2016; Cherdantseva et al, 2013; de Franco Rosa et al, 2018; Fenz et al., 2009, plus the reference of the related works, such as Blanco et al, 2011. The goal of these search strings is to capture as many studies as possible that present some security ontology at the general level required by our scope (see especially inclusion criterion 2). At the same time, the search strings should not retrieve an overwhelming amount of papers; that is one of the reasons why they are different according to the database.

Though some papers appear in multiple databases, large databases end up hiding some relevant papers because of the number of results. This is why we use different search strings in different databases: in general, we make broader searches on smaller databases, like DBLP, and we make narrower searches on bigger databases, like Google Scholar. Moreover, we have experimented with and crosschecked several search string options in multiple databases before finally deciding the ones that follow.

For each database, all queries were made using *the most general field of search*, except when otherwise specified. The number of results retrieved from each database and query is written in parentheses below. We used the *Harzing's Publish or Perish software*[3] to make the queries to Scopus since this software allows a convenient visualization of results. The other queries were made directly to the respective databases. In DBLP we use the term "ontolog" to capture variations like "ontological", "ontologies", and "ontology", according to the search algorithm of this database.

DBLP (263) = security ontolog (258), core reference ontology (2), common security ontology (2), security core ontology (1)

Science Direct (113) = "core security ontology" OR "security ontology" OR "core reference ontology"

IEEE Xplore, ACM Digital Library (55, 67) = "core reference ontology" OR "common security ontology" OR "security core ontology" OR "core security ontology" OR "security conceptual model" OR "security modeling language" OR "conceptual model of security" OR "core ontology of security" OR "common ontology of security" OR "general security ontology" (15, 30), "security knowledge" AND "ontology" (40, 37)

Google Scholar (591) = "common security ontology" OR "security core ontology" OR "core security ontology" OR "security conceptual model" OR "security modeling language" OR "conceptual model of security" OR "core ontology of security" OR "common ontology of security" OR "general security ontology"

Through *Harzing's Publish or Perish software*, we used the "Keywords" search (the most general search) for all queries over Scopus, except for the last two, whose searches were made over "Title words" - constrained to the periods 2010-2015 and

[3] https://harzing.com/resources/publish-or-perish.

2016-2020.

Scopus, Web of Science (322, 294) = "core reference ontology" OR "common security ontology" OR "security core ontology" OR "core security ontology" OR "security conceptual model" OR "security modeling language" OR "conceptual model of security" OR "core ontology of security" OR "common ontology of security" (53, 160) OR "general security ontology" (4, 1) OR ("security knowledge" AND "ontology") (63, 36) OR security ontology (202, 97)[a]

[a]We have added double quotation marks for exact phrase search in this last query on Web of Science. Otherwise, more than 1400 papers are returned.

The author of this thesis was the main responsible for executing this phase, though discussion and revision were made with collaborators[4].

2.4.3 Screening of the studies

The previous phase of our mapping study found thousands of papers, as seen in the last subsection. To select those relevant for us, according to the aforementioned inclusion-exclusion criteria, we proceeded to read key parts of the text as much as necessary to decide whether (or not) each study satisfies each criterion. These parts include, in the following order, the title, keywords, and abstract, and if those were not sufficient, the introduction and conclusion, and, finally, if needed, the other sections. Moreover, we compared the results of our queries to works classified as security ontology with general purpose by other reviews (Blanco et al, 2011; Sicilia et al, 2015) both to select relevant works and to validate our queries. During this process, we realized that some selected studies just mention ontologies of other primary studies to achieve their purposes - hence, except when the former presents progress in the ontology itself, we keep only the primary study, whose main purpose was the introduction of the ontology.

This whole process was made by the author of this thesis, then the outcome was checked by collaborators, and then the author proceeded a double checking to guarantee the relevance of the selected papers according to the inclusion-exclusion criteria. After the conclusion of this phase, the number of relevant studies was reduced to 57. They were added to "My Library" on the Google Scholar profile of the first author for storage and metadata extraction.

2.4.4 Classification scheme

After this procedure, we propose the classification schemes listed below, which are related to the RQs. The classification procedure was executed by the first author, then the outcome was checked and discussed by the co-authors, and then the first authors proceeded a double checking.

- **Implementation language (RQ1, RQ2):** The language used to express the ontology, in particular for execution. In case no executable implementation (like OWL) exists, we mention only the conceptual modeling language, such as *Unified Modeling Language* (UML), the logic language (say, description logic), or natural language. We also use the term "UML-like" to refer to a non-specified diagrammatic language that looks like UML class diagrams.
- **Artifact availability (RQ3):** In case the security model had been implemented, is it publicly available? If so, in which source can it be found? We

[4]Mattia Fumagalli, Tiago Prince Sales, and Giancarlo Guizzardi.

TABLE 2.1: The 57 selected studies presenting core security ontologies grouped by their language of implementation (See RQ1, RQ2)

Language	Study
OWL	de Franco Rosa et al, 2018; Arogundade et al, 2012; Dos Santos Moreira et al, 2008 Pereira et al, 2012; Agrawal, 2016; Dritsas et al, 2005 Parkin et al., 2009; Fani et al, 2015; Ekelhart et al, 2007 Gyrard et al., 2013; Herzog et al, 2007; Fenz et al., 2009; Vorobiev and Bekmamedova, 2007 Oltramari et al., 2015; Oltramari et al., 2014; Ekelhart et al, 2006a Ekelhart et al, 2006b; Kim et al., 2005; Vale et al, 2019; Korger and Baumeister, 2018; Karyda et al, 2006; Kim et al, 2016 Casola et al, 2019; Beji et al, 2009; Souag et al, 2015; Tsoumas et al, 2006b; Boualem et al, 2017; Chen et al, 2018; Kim et al, 2020; Ramanauskaitė et al, 2013; Zheng-qiu et al, 2009; Guan et al, 2016; Tsoumas et al, 2006a; Mozzaquatro et al., 2015
UML	Chowdhury, 2014; Mayer, 2009; Elahi et al, 2009; Lund et al, 2003; Milicevic et al, 2010; Mayer et al, 2019; Fernandez et al, 2014; Pereira et al, 2019
Natural Language	Avizienis et al, 2004; Jonsson, 2006; Schumacher, 2003; Cherdantseva et al, 2013; Mouratidis et al., 2003; Massacci et al, 2011
UML-like	Vorobiev et al, 2010; El-Attar et al, 2015; Kang et al, 2013
RDF	Takahashi et al, 2015; Blanco et al., 2012
Description Logic	Amaral et al., 2006; Li et al, 2020
AS^3 Logic	Yau et al, 2014
XML	An Wang et al, 2010

have searched for the implemented model both inside the paper and on the internet in general, aiming at finding the latest version of the source and the file.

- **Foundational ontology (RQ4):** Whether or not the security ontology is based on some upper ontology, such as BFO, DOLCE, and UFO.
- **Concept words (RQ5):** We consider the term denoting security core concepts appearing in the selected studies, to describe their relative frequency. The goal is to support the identification of the most important concepts for a security common ontology.

2.5 Results

RQ1: Which security core ontologies exist in the literature? Our final data set of studies reporting core reference security ontologies, published between 2000 and 2020, ended up with 57 items. Their distribution in time is shown by Fig. 2.2. We notice there is no study published between the beginning of 2000 and the end of 2002. The list of the selected studies is attached at the end of the thesis, but the table. 2.1 already shows the collected studies presenting some security core ontology while classifying them by their representation language.

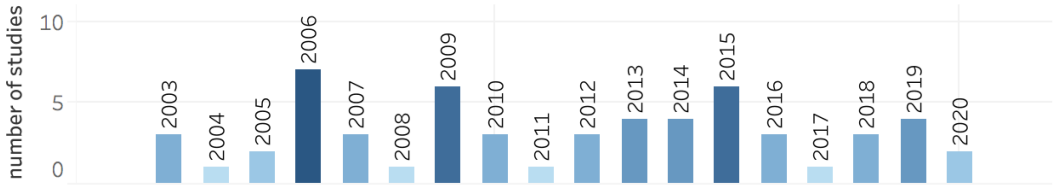

FIGURE 2.2: 57 studies presenting core reference security ontologies grouped by year

RQ2: Which languages have been used to represent the core ontologies of security? Using the data from Table 2.1, we plotted the pie chart shown in Fig. 2.3, which clearly shows the preference for OWL as the representation language of core security ontologies. This is not a surprise, considering that OWL 2 has been a standard recommended by W3C since October 2009.

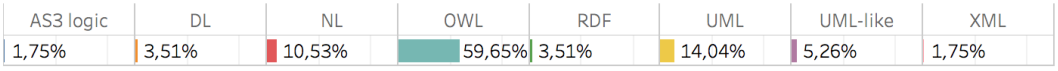

FIGURE 2.3: Proportions of representation languages in studies shown on Table 2.1

RQ3: Are the specifications of the security core ontologies publicly available? If so, in which source (URL)? After searching for the file containing the ontology both within the papers and on the internet, we were only able to find 6 of them, namely de Franco Rosa et al, 2018[5], Fani et al, 2015[6], Gyrard et al., 2013[7], Herzog et al, 2007[8], Blanco et al., 2012[9], Mozzaquatro et al., 2015[10]. We found some links, even when they were not included in their respective papers, in a dedicated catalog for security ontologies[11].

RQ4: Which foundational ontologies have been used in the design of security core ontologies? Among the 57 selected studies, only four have made use of some foundational ontology, which represents 7% of the total: Casola et al, 2019 uses BFO, and Massacci et al, 2011; Oltramari et al., 2015; Oltramari et al., 2014 use DOLCE. We briefly present them below.

Massacci et al, 2011 present an extended ontology for security requirements based on DOLCE that unifies concepts from the Problem Frames and Secure i* methodologies, and security concepts such as asset and threat.

Oltramari et al., 2014 propose an OWL-based ontological framework that is constituted by a domain ontology of cyber operations (OSCO), which is based on DOLCE and extended with a security-related middle-level ontology (SECCO). The authors later extend this framework with the Human Factors Ontology (HUFO) Oltramari et al., 2015 to support predictive cybersecurity risk assessment. Considering human factors, HUFO includes individual characteristics, situational characteristics, and relationships that influence the trust given to an individual.

Lastly, Casola et al, 2019 present a "first step towards an ISO-based Information Security Domain Ontology" to support information security management systems.

[5]Source: https://github.com/ferruciof/Files
[6]Source: http://semionet.rnet.ryerson.ca/ontologies/sio.owl
[7]Source: http://securitytoolbox.appspot.com/stac
[8]Source: https://www.ida.liu.se/divisions/adit/security/projects/secont/
[9]Source: https://sourceforge.net/projects/vulneranet/files/Wiki/
[10]Source: https://github.com/brunomozza/IoTSecurityOntology
[11]Catalog: http://lov4iot.appspot.com/?p=lov4iot-security

They show a high-level ontology for modeling complex relations among domains, and a low-level, domain-specific ontology, for modeling the ISO 27000 family of standards. To assure higher interoperability, they have made use of the principles behind BFO.

RQ5: Which terms appear most often in the core ontologies of security? Grasping the most important concepts of security is essential to devising a common ontology of security. This is the reason behind RQ5. A frequency table would be helpful to approach the issue. However, the results for RQ3 show few available files, which could be used for precise counting. To deal with that we count the frequency of the most general terms when *explicitly stated inside the ontology described in the very paper*. We also normalize some terms, for example avoiding plural, to reflect the frequency of the concept rather than the frequency of the word itself. The result is shown by Table 2.2, which shows the relative frequency of terms in the sense that it reliably presents the most common terms, though the exact counting can harmlessly vary.

Table 2.2: Relative frequency of most common concept terms

Concept Term	$\geqslant$ #	**Concept Term**	$\geqslant$ #
vulnerability	24	risk	9
asset	23	attacker	7
threat	21	control	7
countermeasure	12	stakeholder	6
attack	9	consequence	6

Among the 57 selected studies, each work presents a security ontology and each term appears only once in each ontology if it appears at all. Then we can conclude there exists no concept shared by all selected ontologies. This suggests a general lack of agreement between those security ontologies. At this point, we may wonder whether some of the selected ontologies have been more adopted than others. Since the number of citations (in Google Scholar) offers an approach to this question, we notice Avizienis et al, 2004 with more than 6500 citations stands out from any other work. Studies with the number of citations between 100 up to 300 citations are Mayer, 2009; Fenz et al., 2009; Ekelhart et al, 2007; Kim et al., 2005; Herzog et al, 2007; Tsoumas et al, 2006b; Cherdantseva et al, 2013.

2.6 Discussion: the need for a FAIR security ontology

The interest in security ontologies has been growing in the last fifteen years. Most likely because of the rapid growth of Web apps and the popularization of the internet, which remarkably increased information security concerns. However, these ontologies are not easily *findable* since only circa 10% of them are publicly available. Indeed, the lack of availability of security ontologies was noted by Blanco et al, 2008, so this scenario has not changed significantly so far.

Moreover, the use of foundational ontologies for grounding core security ontologies is still very incipient. The lack of a foundational ontology supporting the construction of a domain ontology is not a problem *per se*. However, studies have shown that foundational ontologies significantly contribute to preventing and detecting bad ontology design (Schulz, 2018), improving the quality and interoperability of domain and core ontologies (Keet, 2011). Indeed, modeling domain and core ontologies without making explicit the underlying ontological commitments of the conceptualization gives rise to semantic interoperability problems. There is a strong connection between the

ability to articulate domain-specific notions in terms of formal ontological categories in conceptual models, and the *interoperability* of these artifacts (Guizzardi, 2020).

Semantic interoperability is also hindered by the sole use of languages such as OWL, which merely address logical issues neglecting truly ontological ones (Guizzardi, 2020; Guizzardi, 2006). Once meaning negotiation and semantic interoperability issues have been established by the usage of an ontologically well-founded modeling language, knowledge representation languages such as OWL can be employed for ontology implementation if necessary (Guizzardi, 2006).

Still regarding interoperability, in our set of selected papers, only four ontologies grounded on a foundational ontology were identified, three of which are based on DOLCE. As demonstrated by Sales et al, 2018, risk (and, hence, risk management, including risk control measures) is an inherently relational phenomenon. This makes DOLCE an odd choice for grounding a reference ontology in this area, given that it does not support relational aspects (relational qualities and bundles thereof) (Guizzardi et al, 2015). In contrast, UFO comprises a rich theory of relations that has successfully been used to address related phenomena such as risk, value (Sales et al, 2018), and trust (Amaral et al., 2019).

In assessing *reusability*, we focus here on two aspects, namely, whether the ontologies *meet domain-relevant community standards* and whether they *provide rich metadata* Jacobsen et al., 2020. Regarding the former, one positive aspect is the fact that most of the ontologies found in our study are represented using international standard languages (e.g., OWL - which is a W3C standard - and UML - which is an OMG *de facto* standard - together account for 73,69% of all the models as per figure 2.2). This at least affords syntactic reusability as well as some predictability in terms of automated inference (in the case of OWL models). However, from a semantic point of view, reusability requires a safe interpretation of the elements being reused in terms of the correct domain categories. In this sense, rich meta-data grounded in well-understood ontological categories is as important for safe reusability as it is for safe interoperability. Here, the same limitations identified for the latter (e.g., the use of ontologically-poor languages such as OWL and UML (Guizzardi, 2006), and the lack of use of foundational ontologies) can also be identified as a hindrance to the former.

In summary, our study highlights the need for advancing on the proposal of Core Security Ontologies that are *Findable, Accessible, Interoperable and Reusable, i.e., FAIR* (Jacobsen et al., 2020).

2.7 Threat to validity

This kind of study has its intrinsic limitations. For example, it heavily depends on search strings and search engines to find relevant academic works. Although we sought to be comprehensive, different search strings and search engines could yield other results. Moreover, our findings are snapshots of the research state when we executed this study in 2021, therefore, they do not say anything about tendencies or future research. They reflect a portion of the past and are not generalizable, although they were produced in a reproducible way, as shown in Section 2.4, Figure 2.1. Despite limitations, our findings offer essential elements to advance our proposal of a security ontology in Chapter 5.

2.8 Conclusion

In this chapter, we presented a systematic mapping study about the literature on core reference security ontologies, considering the last twenty years of research. We started an analysis to understand this research scenario, the implementation languages that have been used, the availability of the ontology files, the domains, and the role of foundational ontologies in security ontologies. Our mapping study has made clear an important research gap in the security ontology field: there seems to be no domain-independent core security ontology in the same general sense as Griffo, 2015 for Legal Relations, Sales et al, 2018 for Value and Risk, and Nardi et al, 2015 for Service. Moreover, foundational ontologies are very underutilized in the field (interoperability). Another gap is the lack of public availability of the actual security core ontologies as artifacts (findability), which makes their analysis and (re)use difficult.

In Chapter 5, the results of this systematic review will support the development of a well-founded security ontology grounded in the Unified Foundation Ontology (Guizzardi, 2005) and as an extension of the Common Ontology of Value and Risk (Sales et al, 2018), following *FAIR* principles (Jacobsen et al., 2020).

Chapter 3

Ontological foundations

This chapter presents the Unified Foundational Ontology (UFO) to the extent that is needed for grounding the theory of prevention in Chapter 4 and the ontology of security in Chapter 5.

The *Unified Foundational Ontology* (UFO) is a domain-independent axiomatic theory developed to contribute to the foundations of Conceptual Modeling. (Guizzardi et al, 2015; Guizzardi et al., 2022; Guizzardi et al., 2021b) It is one of the most used foundational ontologies in conceptual modeling (Verdonck and Gailly, 2016), and for more than two decades now it has been successfully employed in many projects in different countries, by academic, government, and industrial institutions in the development of core and domain ontologies in different domains (e.g., Trust, legal relations and Constitutional Law, Risk and Value, Service, Software Requirements and Anomalies, Discrete Event Simulation, etc.). (Guizzardi et al, 2015)

Many ontological distinctions of UFO are embedded into OntoUML general-purpose modeling language, so its syntax can be defined as a pattern grammar. (Guizzardi, 2014; Zambon and Guizzardi, 2017) This means that OntoUML models are created by the iterative instantiation of ontology design patterns, each of which represents a UFO micro-theory. These patterns are useful from an engineering viewpoint because they help us understand and represent the domain of interest by specializing them into more specific concepts and relations. By doing so, it is possible to build larger ontologies by systematically reusing and combining these micro-theoretical fragments. (Ruy et al., 2017) This allows us to make numerous ontological commitments explicit, at the same time that it enforces a high degree of consistency between our OntoUML model of security and related OntoUML models, such as COVER.

UFO comprises a set of micro-theories, including (a) a theory of types and taxonomic structures with a theory of object identifiers; (b) a theory of part-whole relations; (c) a theory of particularized intrinsic properties, attributes, and attribute value spaces, which includes a view on datatypes as semantic reference structures; (d) a theory of particularized relational properties and relations, including a proposal for Weak Truthmaking connecting particularized properties to propositions; (e) a theory of roles; (f) a theory of events, including aspects such as event mereology, temporal ordering of events, object participation in events, causation, change, and the connection between events and endurants via dispositions; (g) a theory for multi-level modeling.

Figure 3.1 depicts the taxonomy of UFO, which can be divided into three large parts: (1) UFO-A, an ontology of endurants; (2) UFO-B, an ontology of perdurants; (3) UFO-C, an ontology of social and intentional entities. In what follows, we will describe the elements of each of them that we need for our ontological foundations of security.

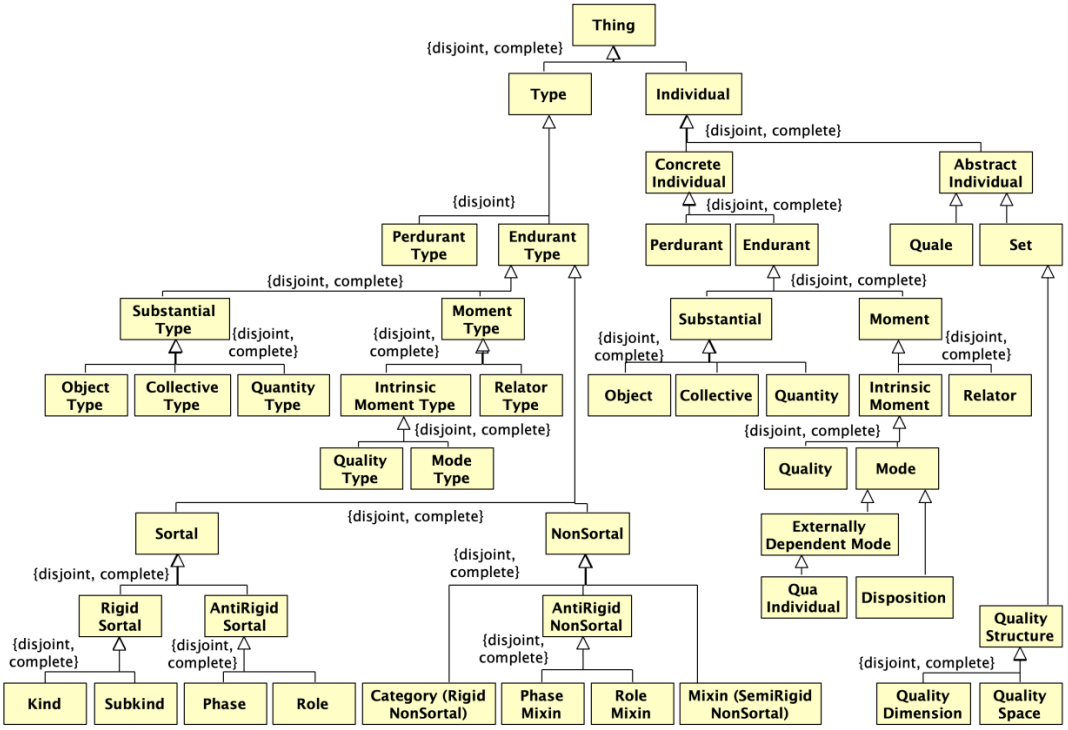

FIGURE 3.1: The Taxonomy of UFO (Guizzardi et al., 2022)

3.1 UFO-A: endurants

UFO makes a distinction between TYPE (universal) and INDIVIDUAL (token). One thing is the specific (individual) John's cup of tea; another is a cup of tea as a type of object, which may have subtypes, such as an ornamented cup of tea, a metal cup of tea, etc. Types and individuals are disjoint, and whatever exists is either a type or an individual. Individuals are instances of at least one type, so there is a mirroring of the taxonomy of individuals over the type level (for example, events are instances of types of events).

Types can be more or less saturated, i.e., they can include in their intension *individual concepts* (Guizzardi, 2005), provided that the pattern of features associated with the types is still possibly repeatable across multiple individuals. A standard (completely unsaturated) type is characterized only by general properties (e.g., the type "Physical Object" is characterized by general properties such as spatial extension, weight, and color). On the other extreme, there are individual concepts, that are completely saturated, i.e., they are instantiated by exactly one individual (single reference) and always the same individual (rigid designation). A semi-saturated type is characterized by both general properties as well as individual concepts. For example, the type "President of Italy" includes the general type "President" and the individual concept "Italy", but it is still a type instantiated by many individuals (e.g., Matarella, Ciampi). In particular, semi-saturated situation types will be relevant for our model of prevention (Chapter 4), e.g., the situation in which I am located in a South Tyrolean City includes a reference to me as an individual[1] but it can be instantiated by situations in which I am in Bolzano, Brunico, etc.

Individuals are classified as either ABSTRACT INDIVIDUAL (specific numbers, sets, propositions) or CONCRETE INDIVIDUAL (the ones that exist in space-time). The

[1] It also includes a reference to another semi-saturated type *South-Tyrolean City*.

latter branch is divided into PERDURANTS (entities that occur) and ENDURANTS (object-like entities). (Guizzardi, 2005; Guizzardi, G. et al., 2013) Classically, an endurant is an entity that is wholly present at any moment at which it exists, being able to undergo qualitative changes while keeping its identity. A person, a car, but also qualities, such as the temperature of an object, are examples of endurants. For instance, Mary, a person, can weigh 60 Kg in some circumstances, and weigh 65 Kg in another; her weight, an individual quality, can vary in different circumstances. In both cases, Mary and her weight are the same individuals before and after these changes.

Concerning the endurants, UFO subscribes to the so-called Aristotelian Square, thus, accounting for both SUBSTANTIALS (substantial individuals), i.e., independent entities (e.g., a person, a car, the Free University of Bozen-Bolzano), as well as particularized properties, i.e., existentially dependent entities or MOMENTS, these last ones termed *aspects*, abstract particulars, or variable tropes. Moments can only exist by inhering in other entities (their bearers). Inherence is a functional relation of existential dependence between a trope and its bearer. This notion of aspect includes both QUALITIES (e.g., the temperature of Mary, the color of a car) and DISPOSITIONS (e.g., a capacity to play volleyball, the electrical conductivity of a material). If an object changes (e.g., Mary's loss of a limb), it might acquire, lose, or have replaced some of its tropes.

For our purposes, the notion of disposition is particularly important. Several terms are used to refer to dispositions, both in philosophy and in ordinary language: 'power', 'ability', 'function', 'potency', 'capability', 'tendency', 'potentiality', and 'capacity'. There is a vast literature in philosophy on the topic of dispositions and how they are related to events and standard (categorical) properties (e.g., height, weight, color). Authors range from those that take all properties to be dispositions (e.g., color is the disposition to refract certain light wave ranges) to those that take all properties to be categorical (e.g., water solubility is just a proxy name for a particular crystalline structure). From a cognitive point of view, notions such as ability, function, liability, capacity, and capability are part of our commonsensical apparatus and our conceptual modeling toolbox. For these reasons, dispositions are taken as a primitive notion in UFO, but also in other foundational ontologies such as the Basic Formal Ontology (BFO) (Arp, R. et al., 2015). In the former, it is an instrumental notion in providing semantics to the notion of capability in enterprise architecture (Azevedo, C. et al., 2015) and defense frameworks (Miranda, G. et al., 2019).

3.2 Endurant types

As seen in Figure 3.1, UFO includes a taxonomy of types of endurants, which are the primary objects in structural conceptual modeling languages, such as UML class diagrams. This taxonomy is embedded into OntoUML through the UML profile mechanism.

An essential sort of endurant type is a KIND (sometimes, called *Substance Sortal*), which provides *uniform principles of individuation*, *identity*, and *persistence* to its instances. For example, person, dog, computer, car, headache, organization, and marriage are types that are typically considered to be kinds. Kinds apply to instantiating individuals in all possible situations in which these individuals exist, and hence kinds are RIGID types, applying necessarily to their instances. A SORTAL is either a kind or a specialization of a kind, and every sortal that is not a kind specializes exactly one kind. Since each individual in the universe of discourse must obey exactly

one principle of identity, which, in turn, is provided by a Kind, each sortal hierarchy has a unique Kind at the top, also referred to as *ultimate sortal*. (Guizzardi et al., 2021b; Guizzardi et al., 2022)

The specializations of a kind can be either rigid, in which case they are SUBKINDS (e.g., hatchback car as a subkind of car; financial organization as a subkind of organization) or they can be ANTI-RIGID, i.e., classifying only contingently their instances. Among the latter, we have sortals whose contingent classification conditions are intrinsic, called PHASES (e.g., teenager as a phase of person, hemorrhagic dengue fever as a phase of dengue fever, and tenured employment as a phase of employment), and we have sortals whose contingent classification conditions are relational, called ROLES (e.g., employee as a role of a person in the scope of an employment relator, and husband as a role of a person in the scope of a marriage relator). (Guizzardi et al., 2021b; Guizzardi et al., 2022)

Unlike sortals, NON-SORTALS (also called dispersive types) are types that represent common properties of individuals of multiple Kinds. Nonsortals can be (a) CATEGORIES: rigid types that define essential properties for their instances, e.g., the category 'physical object' describing the properties of having a mass and a spatial extension, common to things of the kinds car, person, bridge, cow, etc.; (b) PHASE MIXINS: anti-rigid types that define intrinsic contingent properties for their instances. For example, the phase mixin 'living animal' may apply to instances of the kinds person, dog, and horse; the phase mixin 'functional device' may characterize instances of the kinds computer, watch, and espresso machine; (c) ROLE MIXINS: anti-rigid types that define relational contingent properties for their instances. Examples include 'customer' for the kinds person and organization, but also 'insured legal relator' for the kinds employment and enrollment; (d) MIXINS: semi-rigid types that define properties that are essential to some of their instances but accidental to some other instances (e.g., being a 'music artist' is essential to bands but accidental to people). (Guizzardi et al., 2021b; Guizzardi et al., 2022)

Figure 3.2 depicts an example of an OntoUML model with most of the aforementioned ontological distinctions.

3.3 UFO-B: perdurants

In UFO, the categories of EVENT and ENDURANT are disjoint (say, a party should not be confused with the people that participate in it). But dispositions connect endurants and events since the latter are manifestations of objects' dispositions. Events represent changes from one SITUATION (individual state of affairs) to another due to the manifestation of objects' dispositions (e.g., capabilities, liabilities, vulnerabilities, etc.) (Guizzardi, G. et al., 2013; Benevides, A.B. et al., 2019). This creates the following pattern: certain situations activate certain dispositions, which are manifested by events wherein objects (the bearers of those dispositions) participate, leading to new situations. In this case, it is said that a situation *triggers* an event, which *brings about* another situation. For example, antivirus software has capabilities to search, detect, and remove viruses. Under the right settings, after detecting a virus, this software removes it from the device. This removal is an event of manifestation of the software's capabilities, and it brings about a new situation where the virus is not present in the device anymore.

If an event E_1 brings about a situation S that activates the dispositions that are manifested as event E_2, then we say that S *triggers* E_2, and that E_1 *directly_causes* E_2; if E_1 *directly_causes* E_2, and E_2 *directly_causes* E_3, then E_1 *causes* E_3, where *causes*

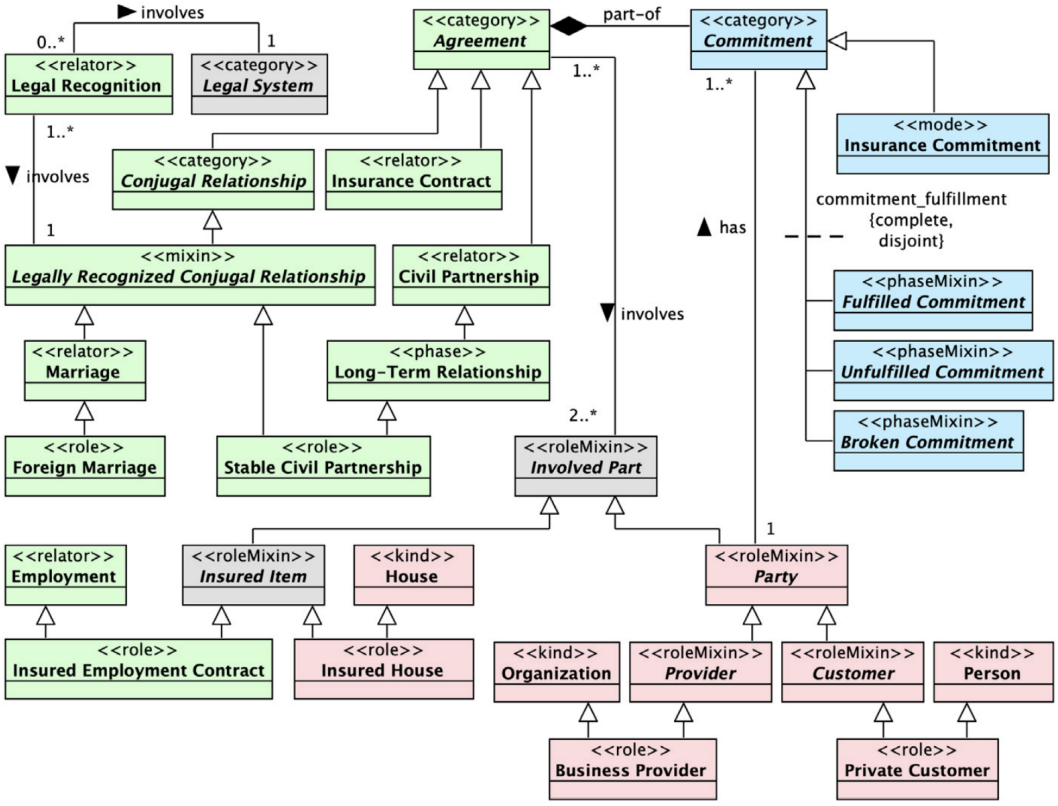

FIGURE 3.2: An example of an OntoUML model (Guizzardi et al., 2021b)

is a strict partial order relation. So *causes* is the transitive closure of *directly_causes*. (Guizzardi, G. et al., 2013)

Events can be composed of other events, forming a mereological structure: an event is *atomic* if, and only if, it has no (proper) parts; otherwise, it is a *complex event* (Guizzardi, G. et al., 2013; Benevides, A.B. et al., 2019). A Ph.D. defense presentation, a party, a security incident, and a natural disaster are examples of events. In contrast with endurants, which are wholly present whenever they are present, events have their parts scattered at different time points. When they are present, only some of their proper parts are present. Situations are composed of other individuals (Almeida, J.P. et al., 2018). For example, the situation in which I have a fever has both me and my temperature (in a particular state) as parts; the situation in which John is married to Mary has both of them and a particular relational trope bundle (their marriage) as parts.

UFO (Guizzardi, G. et al., 2013), in pace with Mumford, 2003 and Mumford and Anjum, 2011, assumes dispositions are properties that are only manifested in specific situations via the occurrence of events. For example, the disposition to attract metal of a magnet is only manifested via an event of metal attraction and movement, which is only manifested when a particular type of situation presents itself (e.g., there is a piece of metal, of a proper weight range, at a proper distance, in a surface that has the right friction).

A situation that triggers an event starts when this event starts, while a situation that is brought about by an event starts when this event ends. There is a unique situation that triggers a particular event occurrence, and there is a unique (maximal) situation that is brought about by an event, corresponding to the effects of the event

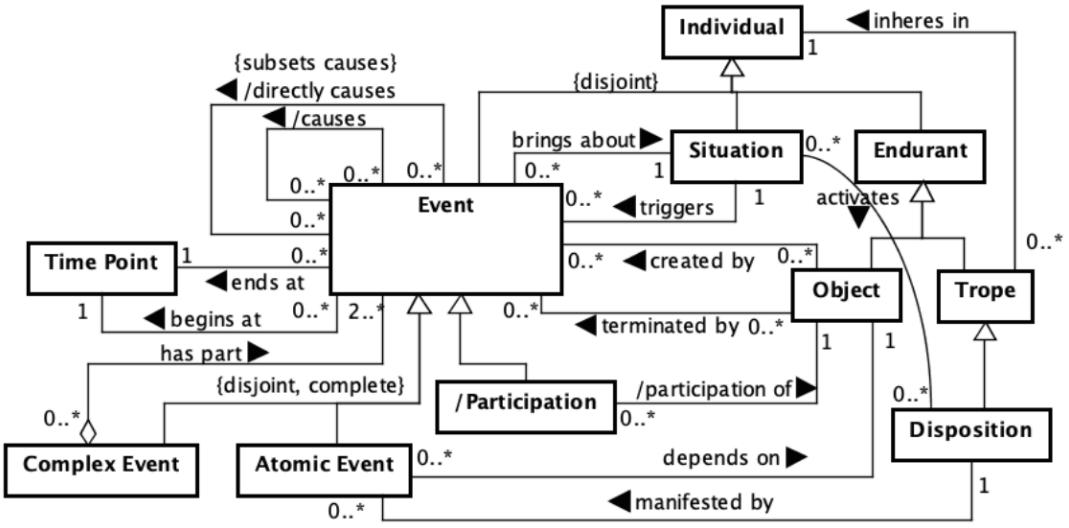

FIGURE 3.3: Individuals in UFO (Almeida, J.P. et al., 2019)

at the moment it ends. Moreover, since it is assumed that a disposition that is being manifested must exist throughout its manifestation, this disposition is not only *present in* the situation that activated it, but it is also present in its manifestation event and in the situation brought about by this event. If the disposition is present in a situation or an event, then the object in which the disposition inheres is also present there (due to existential dependence) (Guizzardi, G. et al., 2013; Benevides, A.B. et al., 2019).

Fig. 3.3 summarizes the individuals in UFO and their relevant relations to our discussion. Notice that no account for interactions between dispositions is provided, even though it is assumed, for example, that an event (manifesting a disposition) can somehow remove another disposition. (Benevides, A.B. et al., 2019)

3.4 UFO-C: intentional entities

UFO-C is an ontology of social and intentional entities built on the foundations provided by UFO-A and UFO-B. UFO-C is not as developed as the other two but contains certain important concepts for our discussion. AGENTS are substantials that can bear special kinds of moments called INTENTIONAL MOMENTS. This *intentionality* means the capacity of some properties of certain individuals to refer to possible situations of reality. Every intentional moment has a type (e.g., BELIEF, DESIRE, INTENTION) and a *propositional content*, which is an abstract representation of a class of situations that are referred to by the intentional moment. The propositional content of an Intention is a Goal. The relation between an intentional moment and a situation is the following: a situation, in reality, can *satisfy*, logically speaking, the propositional content of an intentional moment. (Guizzardi et al., 2008)

Part II

ONTOLOGICAL FOUNDATIONS FOR SECURITY

Chapter 4

Understanding and modeling prevention

Prevention is a pervasive phenomenon. It may occur as a natural phenomenon or as a result of intentional human intervention. It is about blocking an effect before it happens or stopping it as it unfolds: vaccines prevent (the unfolding of) diseases; seat belts prevent events causing serious injuries; circuit breaks prevent the manifestation of overcurrents. Many disciplines in the information sciences deal with modeling and reasoning about prevention. Examples include risk and security management as well as medical and legal informatics. Having an adequate conceptualization of this phenomenon is crucial for devising proper modeling mechanisms and tools to support these disciplines. Forming such a conceptualization is a matter of Formal Ontology. In this chapter, with the support of Unified Foundational Ontology (UFO), we conduct an ontological analysis of this and other related notions, namely, the notions of countermeasures and antidotes. As a result of this conceptual clarification process, we propose an ontology-based reusable module extending UFO and capturing the relations between these elements. Finally, we employ this module to illustrate a few cases in risk management.

This chapter is based on the following work:

- Baratella, R., Fumagalli, M., Oliveira, Í., Guizzardi, G. (2022). Understanding and Modeling Prevention. In: Guizzardi, R., Ralyté, J., Franch, X. (eds) Research Challenges in Information Science. RCIS 2022. Lecture Notes in Business Information Processing, vol 446. Springer, Cham. https://doi.org/10.1007/978-3-031-05760-1_23.

4.1 Introduction

In conceptual modeling, we need to represent both structural and dynamic aspects of reality. Within the latter, we find the recurrent phenomenon of prevention. In a nutshell, it is about blocking an event before it happens or interrupting it as it unfolds, as when vaccines prevent the unfolding of diseases, when seat belts prevent happenings that would cause grave injuries, or when the activation of circuit-break blocks an overcurrent from damaging the circuit. So prevention may occur as a natural phenomenon or as a result of intentional human intervention, a key aspect of the security domain. Prevention may occur with a certain degree of likelihood and with a certain level of effectiveness. Many disciplines in the information sciences deal with modeling and reasoning about prevention. For example, risk and security management is ultimately about prevention: once we identify the assets-at-risk, we want to prevent or mitigate the possible undesired consequences through countermeasures

such as barriers and antidotes; medical informatics frequently deals with the modeling of life-threatening events and their deterrents; normative systems and contracts in legal informatics deal with the design of rules to constrain unwanted behaviors. Given the importance of the topic, having a proper conceptualization of this phenomenon is crucial for devising proper modeling mechanisms and tools to support these disciplines.

Ontology-driven conceptual modeling (Verdonck, M. et al., 2015; Verdonck and Gailly, 2016) is an approach that employs Formal Ontology to analyze and (re)design conceptual models and modeling methodologies, languages, and tools. This approach relies on foundational ontologies, i.e., domain-independent philosophically-sound axiomatic theories. These ontologies provide an inventory of the most general aspects of reality, including classification and taxonomic structures, part-whole relationships, events, causality, dependence, etc. There is evidence showing that the support of foundational ontologies for building conceptual models helps to avoid bad design (Schulz, 2018), and to improve the quality of and interoperability between these models (Keet, 2011; Guizzardi, 2020) as well as between the systems built with their support.

One of the most used foundational ontologies in conceptual modeling is the Unified Foundational Ontology (UFO) (Guizzardi, 2005). In UFO, the unfolding of events is intimately connected to the notion of *dispositions* (e.g., capacities, capabilities, powers, abilities, liabilities, vulnerabilities, intentions). (Choi and Fara, 2021) In a nutshell, events are manifestations of dispositions of objects under certain situations. Moreover, causality, a relation between events, is defined in terms of the activation of certain dispositions in certain situations. (Guizzardi, G. et al., 2013)

In this chapter, we conduct an ontological analysis of prevention and related notions, namely, countermeasures and antidotes. The underlying research question is: *How can we understand and model the phenomenon of prevention?* We show how those notions can be grounded in relations between dispositions, and between dispositions and their manifestations. As a result of this conceptual clarification process, we propose an ontology-based reusable module extending UFO and capturing the relations between these elements. This module can then be reused in the design and integration of domain ontologies, conceptual domain models, and domain-specific modeling languages (e.g., for the domain of security modeling). Finally, we employ this module to illustrate a few cases in the risk management domain.

The remainder of this chapter is structured as follows: Section 4.2 presents several important elements of the concept of prevention. Section 4.3 presents our ontological analysis and culminates with a proposal of a reusable model of prevention capturing the results of this analysis. Section 4.4 shows an application of this model to illustrate some cases in risk management. Section 4.5 discusses related work and, in particular, how the ontological foundations of prevention are articulated in a competing foundational ontology, namely, the Basic Formal Ontology (BFO). Finally, Section 4.6 presents final considerations, including a discussion about the limitations of our approach, and future works.

4.2 Conceptual elements of prevention

According to the dictionary of Merriam-Webster[1], 'prevent' is a transitive verb possibly meaning (1) *to keep from happening or existing*; (2) *to hold or keep back* (hinder,

[1]The term 'prevent' in the Merriam-Webster: https://www.merriam-webster.com/dictionary/prevent.

stop); (3) *to deprive of power or hope of acting or succeeding.* Collins Dictionary[2] brings two related definitions: (1) *To prevent something means to ensure that it does not happen*; (2) *to prevent someone from doing something means to make it impossible for them to do it.* Related verbs include *stop*, *avoid*, *frustrate*, *restrain*, whereas related nouns of the uncountable noun 'prevention' include *elimination*, *safeguard*, *precaution*, *anticipation.* Ontologically unpacking the notion of prevention requires making sense of these linguistic meanings in a single picture.

In philosophy, the discussion about prevention is connected to a puzzle involving the analysis of disposition ascriptions. Traditionally, the term *canonical dispositions* is used to refer to disposition ascriptions that make explicit reference to their stimulus conditions and manifestations (e.g., the disposition to dissolve in water). (Choi and Fara, 2021) In general, dispositions are assumed to be reconceptualizable in terms of their canonical descriptions by explicitly identifying their stimulus conditions and their manifestations. This is the first step, originally suggested by Lewis, 1997, to clarify what means for an object to bear a disposition. The second step would be a conceptual analysis of the resulting canonical disposition. An intuitive analysis is the so-called *Simple Conditional Analysis*:

- An object o, at time t, is disposed to manifestations M when in circumstances C iff o would manifest M if it were the case that o is in C at time t.

A known problem with this analysis is that, in a modal sense, it is not *necessarily* true that if o were to undergo to C, it would give response M. Sometimes something happens, such that, though o is in C at t, it does not respond with M as it would normally do. In the alleged time gap between the activation of a disposition and its manifestation, something could happen to prevent manifestation M. Some philosophers argue some dispositions are *finkish* in the sense that the conditions for an object's acquiring or losing disposition d might be the same as d's stimulus conditions. (Lewis, 1997) This means it is metaphysically possible that dispositions are lost when their conditions of realization occur.

Bird, 1998 proposes another type of counter-example for the Simple Conditional Analysis of disposition descriptions: many dispositions have *antidotes.* An object o is disposed to display manifestations M under stimulus C. At time t_1 it receives stimulus C and so in the normal course of things, at some later time t_2, o gives response M. The time gap between t_1 and t_2 is what allows, in finkish cases, for the loss of a disposition. An antidote, on the other hand, is something which, when applied before t_2, has the effect of *breaking the causal chain* leading to M, so that M does not in fact occur. For example, a person can ingest a lethal dose of poison (say, arsenic poisoning), yet not die if a suitable antidote (say, dimercaprol against arsenic poisoning) is administered soon enough.

More generally, causes can be thwarted. They never guarantee their effects, even when they succeed in producing them because it *could* have been otherwise. (Mumford and Anjum, 2011) In this context, Mumford, 2003 makes a useful distinction:

- *α-conditions*: being *conditions that prevent the manifestation of a disposition though the disposition itself remains.* For instance, the lack of oxygen prevents a struck match from lighting though it remains flammable; the lack of a mate prevents a man from breeding though he remains fertile; placing a vase in a sturdy glass prevents it from being broken though it remains fragile.

[2]The term 'prevent' in the Collins Dictionary: https://www.collinsdictionary.com/dictionary/english/prevent.

- *β-conditions*: being *conditions that prevent something from having a disposition.* For example, a match being wet stops it from being flammable; a zero or low sperm count stops a male from being fertile; a strengthening process stops a vase from being fragile.

Mumford and Anjum, 2011 state that when an effect is prevented, it does not occur:

- Either because of some factor not being present;
- Or because of some extra factor P, a "preventer", disposing away from the effect.

In the sequel, we will explore two modes of prevention in terms of the ontological categories and relations of UFO, presented in Chapter 3: if the original disposition is removed from the object, or by interfering with the causal chain that is foreseen to bring about certain events. On the one hand, by exploring the relation between events and dispositions (events are always manifestations of dispositions), and between dispositions and situations (dispositions are only activated in certain situations); on the other hand, by exploring the notion of indirect causation, that is, between the activation of a disposition and the manifestation of a distal event, there could be a chain of intermediate dispositions activation, event manifestations and situations constituting a causal chain. (Benevides, A.B. et al., 2019)

4.3 Unpacking the notion of prevention

To advance our proposal for analyzing and modeling prevention, we need to extend UFO in three directions. Firstly, we have to generalize the relations between dispositions, their manifestations (events), and situations to the level of types. Secondly, we need to characterize situation types activating dispositions in terms of general requirements that make particular references to dispositions of complementary types. Thirdly, we have to define a notion of incompatibility between situations of certain types.

4.3.1 Lifting the discussion to the level of types

Let us first deal with types. One thing is the flammability of a piece of wood, another is flammability as a type of disposition. Flammability as a type can be associated with a type of event: instances of flammability are manifested via *Catching on Fire* events. Conversely, instances of the latter are always manifestations of dispositions of the flammability type. We therefore define a type-level relation of *manifestation* between event types E_T and disposition types D_T such that: *manifestation*(E_T,D_T) implies that every instance of E_T is a manifestation of an instance of D_T.

Analogously, flammability as a type can be associated with a type of situation via a type-level *activation* relation: instances of flammability are activated by situations of a given type, i.e., they are activated by situations in which there is enough presence of oxygen, the presence of an ignition heat source, etc. This description clearly defines a type as it can be instantiated by a multitude of individual situations, each with different ignition heat sources, different portions of oxygen with different volumes, etc. The same situation type may be connected via activation to dispositions of different types: for example, a situation in which the objects are exposed to high temperature may activate the flammability of woody material, but also several dispositions in the

human body. In contrast, a disposition type is activated (on the type level) by a unique situation type. We define the activation type-level relation between situation types S_T and disposition types D_T symbolized as *activation*(S_T,D_T).

We can also define a type-level counterpart of the UFO relation *brings about*. We call this relation bringing about of (symbolized as *bringingAboutOf*(E_T,S_T)). This relation, holding between event type E_T and situation type S_T, implies that instances of the former bring about instances of the latter.

A situation type associated via *activation* to a given disposition type must include the presence of other dispositions of a suitable kind. In this example, both oxygen and the ignition heat source have specific dispositions, which together with flammability produce a *Catch on Fire* event. In other words, particular events of catching on fire are not manifestations of flammability only but complex events composed of the interacting manifestation of all these dispositions together[3]. Here are some additional examples of this dependence between dispositions and their manifestation: a given disposition of a particular key to open a certain lock only manifests itself when it is in such a situation that the disposition to be opened is also present (inhering in that lock). The opening event is then a combined manifestation of these dispositions. Though this example involves explicitly the reciprocity of dispositions, all dispositions present this partnership aspect: the disposition of a person to swim under the water can only be manifested with the presence (and manifestation) of the dispositions that inhere in the water; the disposition of an object to roll on certain surfaces can only manifest itself with the presence (and manifestation) of the dispositions of the very surfaces (e.g., the friction of a certain kind). When one disposition is manifested, the others are also manifested, each one producing its particular manifestation, which combines to produce an effect.

To capture this dependence relation between a disposition d of a certain type D_T and other types of dispositions, we introduce here the *Mutual Activation Partner (MAP)* relation. MAP(D_T,D'_T) implies that, in order for instances d of D_T to be activated, it needs the presence in the activating situation of an instance of D'_T so that the manifestations of d are always part of complex events that are also composed of a manifestation of instances of D'_T. As a consequence, we have that any situation type S_T bearing an activation relation to instances of D_T must have in its instances (particular situations) instances of all D'_T associated via MAP to D_T. MAP is a relation of generic dependence and, hence, asymmetric and transitive, i.e., a strict partial order relation. This proposal takes elements from the notion of *mutual manifestation partnerships* put forth by many authors in the literature (Mumford, 2003; Mumford and Anjum, 2011; Mumford and Anjum, 2018; Baltimore, 2019).

Let us now make explicit an additional requirement for situations of type S_T activating dispositions of the type D_T. A situation type S_T activating a particular disposition d is semi-saturated in the following way: its instances must be situations in which d and, hence, its bearer are present. This follows directly from UFO's constraints that the manifesting disposition must be present in the situations preceding and succeeding its manifestation and from the existential dependence between the disposition and its bearer. Similarly, we can define semi-saturated event types. For example, the event of *Nina Simone's singing* is still a type that can be instantiated by multiple occurrences, all of which have Nina Simone as a participant.

Now, let's define a notion of *incompatibility* between situation types. Guizzardi, R. et al., 2013 define the notion of *conflict* between situations: situation s conflicts

[3]For this reason, Molnar and Mumford, 2006; Mumford and Anjum, 2011 call events *polygenic* manifestations.

with situation *s'* if they cannot obtain concurrently. We here lift this notion to the type-level by defining an incompatibility relation between situation types S_T and S'_T: *incompatible*(S_T,S'_T) implies that there are no two instances of these two types that obtain in overlapping time intervals. Notice that these situation types are semi-saturated, that is, they must share some relevant dispositions or objects. For instance, a situation with a damp match x is compatible with a situation that contains a different dry match *x'* even if the two situations temporally overlap.

Finally, lifting the discussion to the level of types is necessary for introducing the chances that certain types of events might happen. Likelihood only inheres in types, not in individuals, i.e., a particular event either occurs or does not and it cannot be repeated at different time points. In contrast, events of certain types can have instances occurring with more or less frequency and likelihood. Sales et al, 2018 define two notions of likelihood that can be adopted here: the *Triggering Likelihood* inheres in a Situation Type, and it refers to how likely a Situation Type will trigger an Event Type once a situation of this type becomes a fact; the *Causal Likelihood* inheres in an Event Type, and it states that, given the occurrence of an event e and a certain Event Type E_T, how likely e will - directly or indirectly - cause another event of type E_T to occur.

4.3.2 A model for prevention and related notions

Given that events are complex bundles of dispositions' manifestations, a clear way to block the occurrence of events is somehow interfering with the interacting dispositions (i.e., the mutual activation partners). The notion of canonical dispositions, which specify the stimulus conditions and manifestations, obfuscates this interaction because it focuses on a single disposition under certain conditions. As a consequence, prevention is restricted to cases where we have the removal of dispositions from their bearers.

We argue that prevention must involve some kind of removal of dispositions from the scene (i.e., from the activating situation), but removing it from its bearer is just one way of doing so (let us call it *case a*). An obvious alternative is the removal of the bearer of that disposition from that situation (*case b*). In yet another manner, we can have the removal of a required partner disposition (*case c*) that would otherwise produce the event at hand. A special case of c (*case d*) is when a disposition *d'* of type D'_T (incompatible with a suitable mutual activation partner) is present in that situation (e.g., high humidity inhering in otherwise flammable objects, which are incompatible with a required *dryness* for those object). For example, the catching on fire of combustible object x can be prevented by: removing x's flammability, for instance, by altering the molecular structure of that object (case a), but also by removing x's from a situation where there are the right conditions (e.g., enough oxygen and ignition temperature), or by removing those conditions from that situation (e.g., producing a vacuum eliminating the presence of oxygen and its properties, humidifying the object).

All these cases can be generalized in the following rule: prevention of events of type E_T that are manifestations of dispositions of type D_T occurs when an event of type E'_T brings about a situation of a type S'_T that is incompatible with the situations required to activate instances of D_T, i.e., *manifestation*(E_T,D_T), *activation*(S_T,D_T), *bringingAboutOf*(E'_T,S'_T) *and incompatible*(S_T,S'_T). In other words, an event e prevents the occurrence of an event *e'* iff e brings about a situation that is incompatible with any situation that could activate the disposition of which *e'* is a manifestation. Lifting this to the type level: *prevention*(E_T,E'_T) implies that the occurrence of events of type

E_T brings about situations that are incompatible with the conditions required for the occurrence of events of type E'_T. Bear in mind that these event and situation types are semi-saturated properly, i.e., guaranteeing the presence (co-reference) of the same disposition and bearer. For example, it is the event of *Humidifying object x* that prevents an event of *Catching on Fire of object x*. Obviously, humidifying flammable objects, in general, does not prevent other flammable objects from catching on fire. Figure 4.1 summarizes this idea on the type level, that is, in terms of regularities.

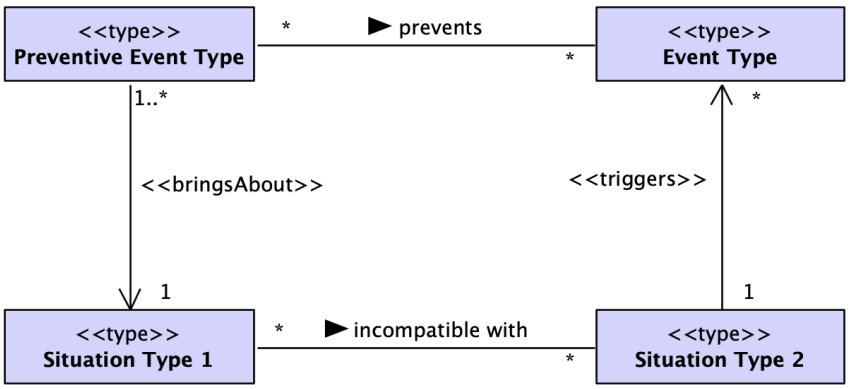

FIGURE 4.1: Prevention schema: certain types of events bring about situations of a given type, such that other types of situations are impossible, resulting in the prevention of the types of events that are triggered by these situations

This characterization of prevention follows from the MAP relation between disposition types and the consequent constraint that it imposes on the situation types S_T needed to activate a disposition of a given type D_T. As previously explained, a situation s of type S_T activating disposition d of type D_T must have present therein d, its bearer, and instances of dispositions of all types connected to D_T via MAP.

Several *patterns* emerge as logical consequences from that notion of incompatibility between situations because it 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, 2005 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 (Blomqvist and Sandkuhl, 2005)) and, then, arranged according to domain entities, forming "architecture patterns" (Blomqvist and Sandkuhl, 2005) 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

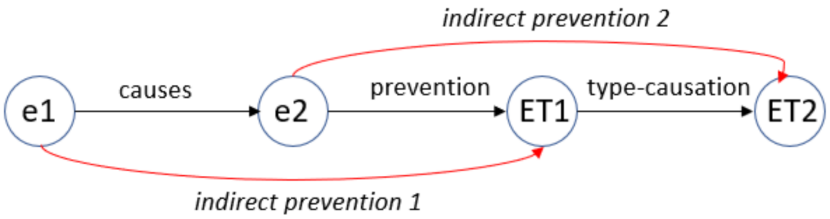

FIGURE 4.2: Two types of indirect prevention

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).

4.3.3 Breaking a causal chain of events

Intuitively, if an event *e* causes an event *e'*, and *e'* prevents events of type E_T, we would be inclined to accept that *e* prevents instances of E_T. This suggests a distinction between *direct* and *indirect prevention*, analogous to direct and indirect causation. Thinking about this chain of causation also allows us to deal with the case of *antidotes* (Bird, 1998). As previously discussed: what makes the world tick according to UFO is a sequence of events bringing about situations that activate dispositions that bring about other situations that activate other dispositions, and so on. So, if we have that an event *e* indirectly causes an event of type E_T, we can interfere in the causal chain connecting *e* to events of type E_T. Using the mechanism we described above, we can execute an event of type E'_T that prevents one of the events of type E^i_T that would otherwise occur in that causal chain. For instance, the event in which John drinks poison would cause his death. Looking into the causal chain, we can be more precise: John's drinking of that poison causes a series of biochemical reactions in his body that eventually cause his death. This can be avoided if John takes an *antidote* in time, i.e., if John ingests a substance that has the disposition when manifested, of preventing an event in that causal chain.

One way of producing indirect prevention is if an event *e* causes an event *e'*, and *e'* prevents events of type E_T, so we say *e* indirectly prevents E_T. Another way of producing indirect prevention is if an event *e* prevents events of type E_T, which is causally connected to E'_T, so we say *e* indirectly prevents E'_T. For example, an event *my car engine failure* causes the event *my car stops in the traffic*, which prevents the events of (semi-saturated) type *me attending the job interview*; if I had attended the job interview, *I would have gotten the job* - a type of event that is historically dependent on the events of type *me attending the job interview*. Indirect prevention plays an important role in security engineering, because Security Mechanisms (a) may produce a chain of events that eventually prevents directly the desired type of event or (b) may block a causal chain of undesired types of events. Figure 4.2 depicts these two types of indirect prevention.

Antidotes are a particular case of *Countermeasures*. In general, given a disposition *d* whose manifestations are of type E_T, countermeasures are designed interventions

that endow a setting containing d with other dispositions $\{d_1, ..., d_n\}$, whose manifestations prevent any instance of E_T. More specifically, *Countermeasure Mechanisms* are designed such that: they contain dispositions of type D_T, and given the situations of type S_T that would trigger events that would (directly or indirectly) cause instances of E_T, the instances of S_T instead activate the instances D_T whose associated event type prevent E_T. For example, a circuit break contains a disposition to close the circuit in a situation where there is a current above a certain threshold. The manifestation of that disposition of the circuit breaker thus prevents the event of an overcurrent.

Our analysis makes explicit several ways in which countermeasures can be designed:

1. We can remove the disposition d whose manifestation we want to avoid (this can be done by removing the object with that disposition from the setting at hand);

2. We can remove from the scene required activation partners (e.g., produce a vacuum to prevent fires);

3. We can include in that setting a disposition that is incompatible with a mutual activation partner (e.g., humidifying a flammable object, removing dryness as a required property);

4. We can design countermeasure mechanisms surrounding the bearer of d, which can prevent the manifestation of d.

Although our examples so far focus on the disposition of physical objects, the proposed model can also be applied to social examples. For instance, if we want to prevent a theft from happening in a building, we can: remove the vulnerabilities of the access points to the building. Alternatively, we can suitably remove mutual activation partners, namely, the capacity and/or intention inhering in potential burglars. For example, by employing locking materials that are very resistant and complex to circumvent, plus legal mechanisms that create severe liabilities for offenders, we can at the same time eliminate vulnerabilities of access points, and eliminate matching capacities as well as intentions.

Fig. 4.3 summarizes the discussion of this session. The fragment in blue is our extension to the original UFO fragment proposed by Guizzardi, G. et al., 2013; Benevides, A.B. et al., 2019 (in black).

4.4 Prevention applied to risk management

In this section, we apply our proposed model of prevention to reason about cases in risk management. Before doing that, however, let us harmonize our discussion with the vocabulary and concepts of that domain. We do that with the help of the Common Ontology of Value and Risk (COVER) (Sales et al, 2018). In a nutshell, for COVER: *value* is a relational property emerging from the relations between certain *capacities* (dispositions) of certain objects (the *value object*) and the goals of a particular agent. In other words, the value of that object to that agent is the degree to which those capacities can be enacted (as manifestations) to bring about situations that satisfy the goals of the agent. *Risk*, in a sense, is anti-value, or "value with the reverse polarity" having an analogous formulation: risk is a relational property emerging from the relations between the *vulnerabilities* (dispositions) of an *object at risk*, as well as

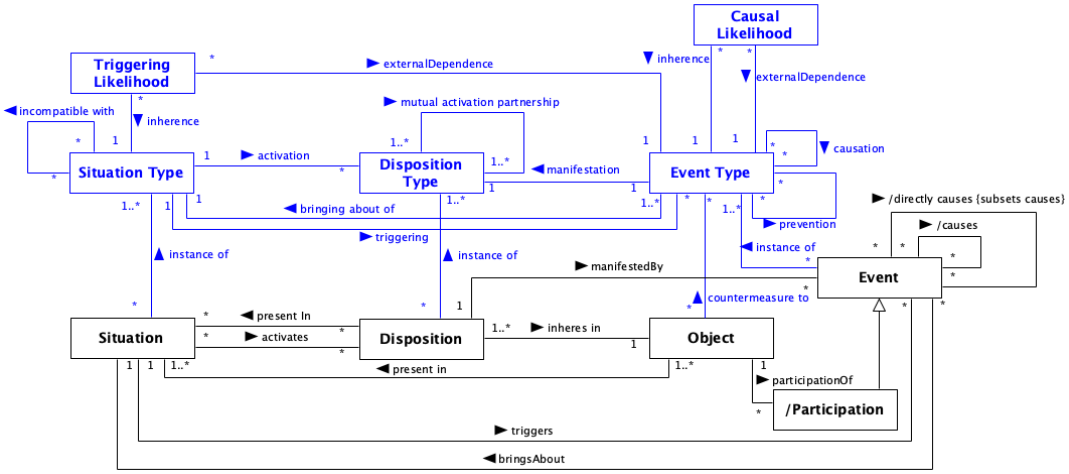

FIGURE 4.3: Extending UFO with type-level relations to model prevention

the *threatening capacities* (dispositions), and sometimes intentions (dispositions) of a *threatening entity*. The combined manifestation of these dispositions (vulnerabilities, capacities, intentions) are threat events[4] that can (directly or indirectly) cause loss events, i.e., events that hurt or break the goals of a particular agent. This last point highlights another manner in which value and risk are connected: the object at risk must be a value object to a given agent. In other words, things that have no value cannot be said to be at risk. In a sense, risk is always the risk of destruction of value.

In the sequel section, we employ our model of prevention, as well as the vocabulary put forth by COVER to address exemplars of the four *cases a-d* previously identified:

Application 1 - Therac-25 accidents (Leveson and Turner, 1993): In a known tragic event, the Therac-25, a medical equipment for radiation therapy, malfunctioned and exposed several patients to an excessive and in some cases lethal dose of radiation. This was caused by an anomaly in the software controlling the medical equipment. This case appears as an illustration of a UFO-based core ontology of software risks and anomalies proposed by Duarte, B. et al., 2021. Following their analysis, a preventive event of upgrading the software would have eliminated said software anomaly (a propensity to cause Race Conditions, i.e., a disposition) from the program copy installed in that machine. In other words, a proper event of *Software Upgrading* would have prevented threat events of the type *Megavolt X-Ray Activation*, consequent threat events of *High Dosage Radiation Exposure* and, ultimately, loss events of *Patient Death*. This exemplifies a case of prevention by removal of an undesired disposition (the vulnerability of the software copy), i.e., an exemplar of *case a*.

Application 2 - Caged tiger in a zoo: This example is frequently used by the community of the Bowtie methodology[5], which is a prominent professional risk assessment methodology (Ruijter and Guldenmund, 2016). Caging a tiger is an event that brings about a lasting situation in which the tiger, although maintaining its threatening capabilities, is separated from the public (bearers of vulnerabilities). In this case, neither the capacities of the tiger nor the vulnerabilities of members of the public w.r.t. to these capacities are present in the same situation (*case b*). Now,

[4] According to our model, vulnerabilities, capacities, and intentions of a certain type are mutual activation partners!

[5] E.g., https://www.bowtiepro.com/examples/htmlexport/hazardref.htm.

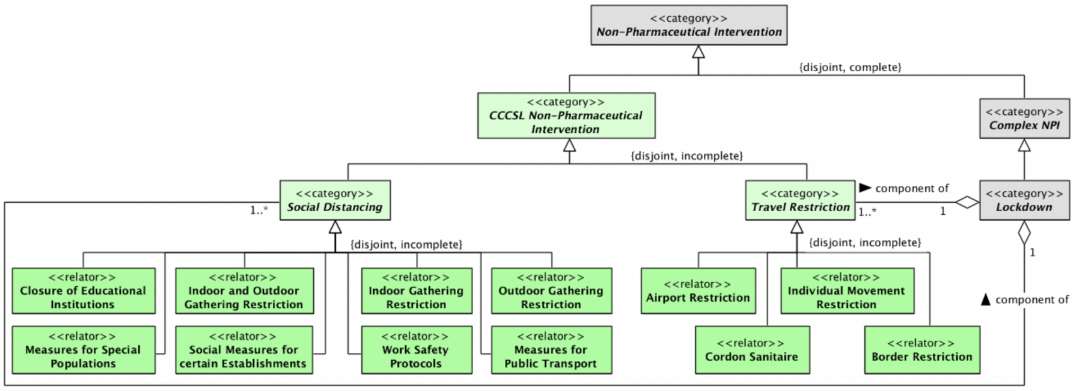

FIGURE 4.4: A Model of Lockdown (Fabio, I. et al., 2021)

suppose the tiger escapes from its cage. A common countermeasure would be employing a dart projectile to sedate the animal. The correct use of this measure is an event that brings about a situation in which several mutual activation partners for the tiger's threatening capacities are removed, namely, the tiger's consciousness and, consequently, its intention to attack (*case c*). In both cases, there is the prevention of threatening events of the type *Attacking the Public*, which in turn prevent loss events of the type *Having members of the Public Dangerously Injured*.

Application 3 - Non-Pharmaceutical Interventions: Fabio, I. et al., 2021 conduct an ontological analysis of non-pharmaceutical interventions and, in particular, of the semantically overloaded term "Lockdown". As illustrated in Figure 4.4, a lockdown is a complex form of non-pharmaceutical intervention that potentially aggregates different forms of *Social Distance* Measures as well as different forms of *Travel Restrictions* measures. Lockdowns are, therefore, countermeasure mechanisms (*case d*) intended to dissuade people from (i.e., ideally remove their *intention* of) placing themselves in situations where a vulnerability (to virus contamination) could be manifested - a contagion event is a manifestation of the infectability of the virus collective[6] (capacity to enter human cells, reproduce, elevate the immunological system, etc) together with the vulnerability of human subjects. In other words, lockdowns are meant (by design) to systematically produce situations in which these dispositions (capacities, on one side, and vulnerabilities, on the other) cannot be both present (*case d*). In contrast, pharmaceutical interventions typically act by endowing human subjects with antidotes (e.g., capacity to destroy the virus before it can enter the cell - in the case of vaccines, or capacity to interrupt the causal chain between infection and the emergence of symptoms - anti-viral medicine).

Notice that for all these cases we could assume a probabilistic perspective captured by the *Triggering and Causal Likelihood* properties (see Fig. 4.3) that can be ascribed to types of preventive events.

4.5 Related work

The concept of disposition as a property of objects appears in several foundational ontologies. The ISO/IEC 21838-2:2021 standard BFO is the one among these that presents the most extensive treatment of dispositions and their interactions (including different types of prevention), and this is why we focus the discussion of this section on it.

[6] A single virus cannot infect anyone; infectibility is a disposition of a virus collective.

Like in UFO, in BFO, a disposition is a dependent entity that inheres in an object and that is realized by events in certain circumstances. However, unlike UFO, BFO also requires that every disposition is grounded in some physical quality of its bearer (e.g., in the way *water solubility* is grounded on some complex crystalline structure of sugar[7]). As a consequence, in BFO, for an object to lose some disposition, it must lose some quality. (Goldfain, A. et al., 2010; Arp, R. et al., 2015)

Goldfain, A. et al., 2010 employ BFO in the construction of the Infectious Disease Ontology (IDO). There, three kinds of interactions between dispositions in BFO are presented: blocking dispositions, complementary dispositions, and collective dispositions. If D_1 is a disposition and D_2 is a blocking disposition for D_1 (called blocked disposition), then it must be the case that the manifestation of D_2 prevents the manifestation of D_1. This prevention is understood in two ways: (a) *incompatible occurrences*, when the manifestation of D_1 and the manifestation of D_2 are somehow incompatible occurrences, that is, they cannot exist simultaneously or one negatively regulates the other; by definition, a process P_1 *negatively_regulates* a process P_2 when the unfolding of P_1 decreases the frequency, rate, or extent of P_2; (b) *incompatible qualities*, when the manifestation of D_2 makes an object gain a quality that is incompatible with some quality that the same object would have acquired through the manifestation of D_1.

Now, the case of incompatible occurrences in BFO involves one key observation. On the one hand, since occurrences (events) are manifestations of dispositions and, in BFO, dispositions are grounded on physical qualities, the case of incompatible occurrences boils down to a case of *incompatible qualities*. So, (a) above is reducible to case (b). On the other hand, it might be the case that dispositions D_1 and D_2 are compatible, but their manifestations are incompatible due to the manifestation of further disposition D_3 present in the situation at stake (our case d). Consider the following example: Tom has the legal right (a disposition) to vote on candidate A; he also has the right to vote on candidate B. Now, the manifestation of the former prevents the manifestation of the latter (and vice-versa). What makes the manifestations incompatible in this case is not the incompatibility of the respective initial dispositions, but some countermeasure introduced in the situation - viz., rules of preventing double voting.

In our model, all these cases are generalized by having incompatible situations. The incompatibility of occurrences can be reduced to the incompatibility of their triggering conditions (situation types). Moreover, we advocate that in making dispositional analysis, we are seldom interested in these qualities but in the dispositions that they ground and that are removed from scenes. Likewise, when analyzing tropes that must co-occur with a disposition to enable its activation, we are not interested in qualities *per se* but in the proper grounded dispositions (mutual activation partners). For example, in the flammability example, we are not interested in the volume of the oxygen mass presented by the dispositions that can only inhere on oxygen masses of a certain volume. Further, BFO's treatment of blocking dispositions seems to omit the case where the manifestation of a disposition is prevented by the absence or removal of its bearer from the circumstances that would allow the realization of the disposition.

[7] UFO does not make such a commitment. The reasons are related to the fact that dispositions and qualities interact at different levels. For instance, the crystalline structure of sugar is itself grounded in the disposition of the molecules constituting sugar to bind in a specific way. Moreover, as discussed before, the distinction between qualitative/categorical and disposition properties is not settled in the literature.

Complementary dispositions are somehow manifested together in the same process: for example, the functions of hammers and nails, locks and keys. To address this case, the authors proposed that different dispositions form a whole with a collective disposition that is manifested in a single event. (Goldfain, A. et al., 2010) A collective disposition is defined as a "disposition inhering in an object aggregate *OA* in virtue of the individual dispositions of the constituents of *OA* and that does not itself inhere in any part of *OA* or in any larger aggregate in which *OA* is part". For example, the collective capability of two people to lift together a heavy object; and the collective capability of the crowd to do a wave due to the capability of each person to stand up at the appropriate time.

In our approach, the phenomena of blocking, complementary, and collective dispositions are generalized in one single framework: complementary and collective dispositions are simply reducible to MAP relation, whereas the blocking disposition can be generalized in terms of incompatible situations.

4.6 Final considerations

In this paper, we have presented an ontological analysis of the notion of prevention by relying on and extending the Unified Foundational Ontology (UFO). In particular, we relied on this ontology's notions of events, dispositions, situations, and their ties. By lifting the UFO modeling of these categories to the level of types, we managed to propose a general model of prevention between events of certain types. This model is then used to analyze the cases of: (i) antidotes, which interfere in a causal chain whose outcome we intend to prevent; (ii) other countermeasures that can (directly or indirectly) prevent events of a given type from manifesting; (iii) countermeasure mechanisms, which are designed interventions that endow a given setting containing a disposition of interest with antidotes or other countermeasures. Finally, we used this framework to interpret elements from the COVER value and risk core ontology and applied it to analyze three cases in the literature. These cases constitute a preliminary analysis, which will be extended in future work.

COVER has been applied to analyze and extend the enterprise architecture standard Archimate in terms of its value (Sales, T. et al., 2019) and risk (Sales, T. et al., 2018) modeling capabilities. In Chapter 5, we extend COVER with the model proposed here to build a core ontology of security and safety. In the same spirit of the aforementioned works, in Chapter 8, this ontology will then serve as a basis for analyzing and extending security elements of Archimate.

In the future, we intend to extend our model of prevention to generally deal with aspects of *dispositional (gradable) interference* (Mumford and Anjum, 2011). If an effect is interfered with, it still occurs but differently: maybe not as strongly as it could have occurred, or not quite in the same way, or perhaps it becomes delayed. Interference typically happens because of some extra factor being present that disposes away from the type of event identified as the effect. (Mumford and Anjum, 2011) Instead of being limited to blocking cases, we intend to deal with cases in which dispositions can either decrease or increase the degree of manifestation of other dispositions. For example, protection against shocks does not eliminate the fragility of a glass but it reduces the manifestation of a shattering event; a catalyst is a disposition that can accelerate the manifestation of other dispositions. Currently, our model can deal with interference regarding the frequency of events (via the different likelihood properties). We believe that prevention can be formulated as a limited case

of decreasing interference. Finally, we intend to provide a full formalization of our model with formal validation and consistency checking[8].

[8]Project repository: https://purl.org/prevention-ontology.

Chapter 5

An ontology of security from a risk treatment perspective

In Risk Management, security issues arise from complex relations among objects and agents, their capabilities and vulnerabilities, the events they are involved in, and the value and risk they pose to the stakeholders at hand. Further, there are patterns involving these relations that crosscut many domains, ranging from information security to public safety. Understanding and forming a shared conceptualization and vocabulary about these notions and their relations is fundamental for modeling the corresponding scenarios so that proper security countermeasures can be devised. Ontologies are instruments developed to address these conceptual clarifications and terminological systematization issues. Over the years, several ontologies have been proposed in Risk Management and Security Engineering. However, as shown in recent literature, they fall short in many respects, including generality and expressivity - the latter impacting their interoperability with related models. We propose a *Reference Ontology for Security Engineering (ROSE)* from a Risk Treatment perspective. Our proposal leverages two existing Reference Ontologies: the *Common Ontology of Value and Risk* and a *Reference Ontology of Prevention* (presented in Chapter 4), both of which are grounded in the *Unified Foundational Ontology* (UFO). ROSE is employed for modeling and analyzing some cases, in particular providing clarification to the semantically overloaded notion of Security Mechanism.

This chapter is based on the following work:

- Oliveira, Í., Sales, T.P., Baratella, R., Fumagalli, M., Guizzardi, G. (2022). An Ontology of Security from a Risk Treatment Perspective. In: Ralyté, J., Chakravarthy, S., Mohania, M., Jeusfeld, M.A., Karlapalem, K. (eds) Conceptual Modeling. ER 2022. Lecture Notes in Computer Science, vol 13607. Springer, Cham. https://doi.org/10.1007/978-3-031-17995-2_26.

5.1 Introduction

In Risk Management, security issues arise from complex relations among objects and agents, their capabilities and vulnerabilities, the events they participate in, and the value and risk they ensue to the stakeholders at hand. Moreover, there are patterns involving these relations that crosscut many domains, including aviation, information systems, chemical industry, public safety, and national defense (ISO, 2018). According to ISO 31000, the purpose of Risk Management is the *creation and protection of value* by a "systematic application of policies, procedures, and practices to the activities of communicating and consulting, establishing the context and assessing, treating, monitoring, reviewing, recording and reporting risk" (ISO, 2018).

Understanding the details and having a shared conceptualization and vocabulary about those notions and their relations is fundamental for modeling and analyzing the corresponding scenarios so that proper security countermeasures can be devised. Once this conceptualization task is done properly, it is possible to model and reason about actual and possible scenarios to assess and counter the risks through security mechanisms.

Models representing risk and security scenarios play an important role in the understanding, analysis, communication, and training in Risk Management. They guide what questions should be asked, and the type of data that should be collected; they establish relations between pieces of information and help give meaning to data; they define the ways risks can be treated; they provide a shared conceptual framework among stakeholders which support communication and training. (Kjellén, 2000)

Ontologies are instruments developed in many domains to address the tasks related to conceptual clarification and terminological systematization. Indeed, the need for a general security ontology was already noticed by Donner, 2003 as a way of rigorously organizing the knowledge about the security of information systems, helping to report incidents more effectively, and sharing data and information across organizations. Then, several ontologies have been proposed in Risk Management and Security Engineering to offer support for many conceptual problems and applications. For example, risk and security assessment (Oltramari et al., 2015); data integration and interoperability (Chen et al, 2018); simulation of threats to corporate assets (Ekelhart et al, 2006b). In parallel, domain-specific modeling languages, such as CORAS (Lund, Solhaug, and Stølen, 2010), Bowtie Diagrams (Saud, Israni, and Goddard, 2014), and the risk and security overlay of the ArchiMate language (Band et al., 2015), implicitly assume an ontology of risk and security in their modeling constructs. An adequate reference ontology of this domain would be able to analyse, (re)design, and integrate languages like these, improving their modeling capabilities, in a way analogous to how the *Common Ontology of Value and Risk* (COVER) (Sales et al, 2018) has been used to redesign Archimate w.r.t. risk and value modeling (e.g., (Sales, T. et al., 2019; Sales, T. et al., 2018)).

Existing proposals for conceptualizing risk and security - counting current security core ontologies and the metamodels of domain-specific modeling languages - fall short in many respects, including generality (e.g., they tend to suffer from premature domain optimization) and expressivity (e.g., they tend to be represented through ontologically neutral modeling languages, missing ontological distinctions) - the latter impacting on their interoperability with related models. Often, these problems come from the fact that these models are designed as lightweight ontologies (i.e., focused on computational aspects) as opposed to Reference ontologies (i.e., focused on ontological precision and conceptual adequacy). For example, although CORAS language, Bowtie diagrams and ArchiMate language show an appropriate degree of generality for representing different scenarios, they are informal languages which, in ontological terms, conflate the *object*, its *capability*, and the associated *event* and *situation* concerning security mechanisms. On the other hand, security core ontologies presented in computational logic languages, such as OWL, are often narrow by having specific applications in mind, missing at least the desirable generality. These core ontologies of security even fall short w.r.t. the *FAIR principles*, i.e., basic management standards for scientific artifacts, as shown in Chapter 2.

To address these limitations, we employ an Ontology-Driven Conceptual Modeling approach (Verdonck, M. et al., 2015) to propose a *Reference Ontology for Security Engineering (ROSE)* from a Risk Management perspective. The primary purpose of ROSE is to support activities related to what the ISO 31000 calls *Risk Treatment*

process. (ISO, 2018) Alternatively, one could refer to this as *security engineering of cybersocial systems*, because of the nature and pervasiveness of the problem of devising mechanisms for controlling and preventing the risks in cyber-physical and social systems (e.g., woody gates, circuit breakers, antivirus software, lockdown norms).

Our proposal leverages two existing Reference Ontologies, namely, the *Common Ontology of Value and Risk* and a *Reference Ontology of Prevention* (presented in Chapter 4), both of which are grounded in the *Unified Foundational Ontology* (UFO) (Chapter 3). This chapter addresses the main research question of this thesis regarding understanding and modeling the security domain. Finally, ROSE is employed for modeling and analyzing some cases, in particular providing clarification to the semantically overloaded notion of Security Mechanism.

In what follows, Section 5.2 presents the requirements we expect ROSE to fulfill. Section 5.3 presents our main contribution, the reference ontology of security termed ROSE, reusing an extended and reinterpreted version of COVER. Section 5.4 shows how ROSE satisfies the proposed requirements. Section 5.5 discusses the main related works. Section 5.6 marks our final considerations.

5.2 Requirements for a reference ontology of security

ISO 31000 defines that the process of Risk Management involves communication and consultation about the risks, risk assessment, risk treatment, recording and reporting, and monitoring and review. (ISO, 2018) In this view, the purpose of risk treatment is to select and implement options for addressing risk. Risk treatment involves an iterative process of: "(a) formulating and selecting risk treatment options; (b) planning and implementing risk treatment; (c) assessing the effectiveness of that treatment; (d) deciding whether the remaining risk is acceptable; (e) if not acceptable, taking further treatment" (ISO, 2018). ISO 31000 states that selecting the most appropriate risk treatment option(s) involves balancing the potential benefits derived in relation to the achievement of the objectives against costs, effort, or disadvantages of implementation. Risk treatment options include: "(a) avoiding the risk by deciding not to start or continue with the activity that gives rise to the risk; (b) taking or increasing the risk to pursue an opportunity; (c) removing the risk source; (d) changing the likelihood; (e) changing the consequences; (f) sharing the risk with another party or parties (including contracts and risk financing); and (g) retaining the risk by informed decision" (ISO, 2018). ROSE can be seen as a way of *ontologically unpacking* (Guizzardi et al., 2021a) this notion of risk treatment, according to the risk treatment options, through an ontological analysis based on the Unified Foundational Ontology (UFO) and its general-purpose conceptual modeling language OntoUML. (Guizzardi, 2005)

Taking these tasks into account, we distinguish three types of requirements that ROSE shall satisfy:

Analysis Requirements (AR): domain-specific capabilities associated to the tasks the ontology should help to realize:

1. Since security engineering requires risk assessment as a previous step in the risk management process, ROSE shall support the identification and assessment of risks;

2. ROSE shall support activities associated with security engineering in multiple domains.

Ontological Requirements (OR): domain-specific concepts and relations the ontology should have to realistically represent its domain of interest and thus support what it is intended to support:

1. ROSE shall support the task of representing the risk treatment options (a)-(g), which are directly connected to the AR2;
2. ROSE shall include both risk and security concepts, explaining explicitly how they interact with one another, including the ones mapped by Oliveira et al., 2021 as the most common in security core ontologies: *Vulnerability*, *Risk*, *Asset*, *Attacker*, *Threat*, *Control*, *Countermeasure*, *Stakeholder*, *Attack*, *Consequence*;
3. ROSE shall be able to distinguish intentional and non-intentional threats because this distinction impacts the risk treatment options.

Quality Requirements (QR): domain-independent characteristics the ontology is expected to possess, so it becomes a better artifact:

1. *Domain appropriateness* (Guizzardi, 2005) - ROSE shall capture the relevant entities and relations of the domain through an ontological analysis;
2. *Generality* - Since security crosses multiple different areas, a security reference ontology should represent the most general concepts of the domain;
3. *FAIR principles* - ROSE should be Findable, Acessible, Interoperable and Reusable. (Jacobsen et al., 2020)

5.3 A Reference ontology for security engineering

We are interested in UFO conceptualization of prevention (presented in Chapter 4), which involves multiple ways of stopping or forestalling certain types of events because this sort of dynamics plays a fundamental role in the domain of security. We build ROSE as an extension and reinterpretation of the Common Ontology of Value and Risk (COVER), applying the theory of prevention. COVER has been chosen, because it includes several concepts and relations about value and risk that are crucial for a reference ontology of security. Indeed, COVER has been used to evaluate and redesign ArchiMate language regarding value and risk. (Sales, T. et al., 2019; Sales, T. et al., 2018) Additionally, COVER has been successfully applied for modeling different domains, such as trust (Amaral et al., 2019), software anomalies (Duarte et al., 2021), among others. However, COVER assumes specific future events are entities of its domain of discourse (Sales et al, 2018), an assumption that is inconsistent with the UFO theory of events, which claims that particular events are immutable entities in the past (Guizzardi, G. et al., 2013). We adopt the UFO assumption with the support of higher-order types to represent future events as types of events and review some cardinality constraints in COVER.

Furthermore, likelihood or probability can be ascribed to events of certain types, as described by Sales et al, 2018 and incorporated in UFO theory of prevention (Chapter 4) in the following way: Triggering Likelihood inheres in a Situation Type, and it refers to how likely a Situation Type will trigger an Event Type once a situation of this type is brought about by an event; the Causal Likelihood inheres in an Event Type, and it means the chances of an event causing, directly or indirectly, another one of a certain type.

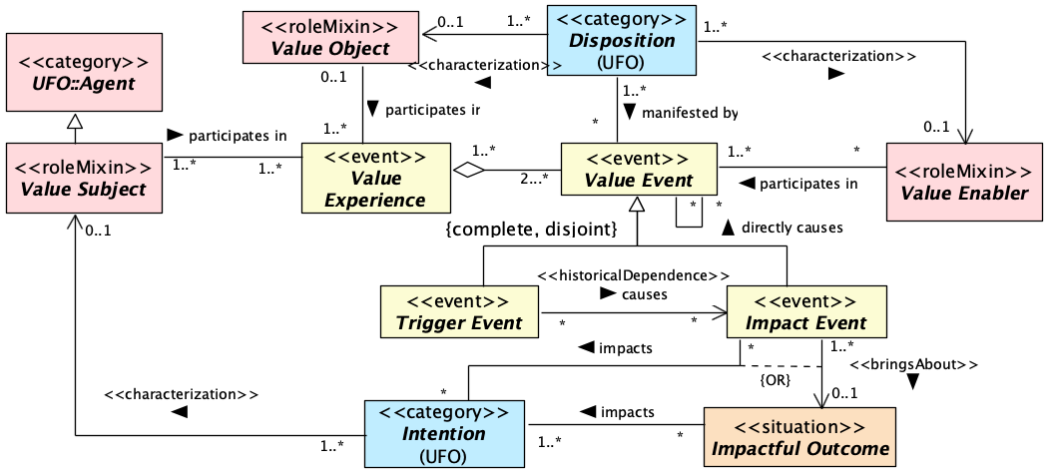

FIGURE 5.1: Value Experience in COVER (Sales et al, 2018)

Our approach is to understand the domain of security as the *intersection* between the domain of value and risk, understood under the terms of COVER (Sales et al, 2018), and the theory of prevention. In this sense, a SECURITY MECHANISM creates value by, systematically, protecting certain goals from risk events. In COVER, value is a relational property that emerges from the relations between the capacities 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 - the goal is simply the propositional content of an intention (Guizzardi et al., 2008). This understanding of value is depicted in Figure 5.1. Risk is the anti-value: risk events are the manifestations of capacities, vulnerabilities, and, sometimes, intentions that inhere in an agent; these events bring about a situation that hurts the goal of a given agent. Like value, security is a relational property that emerges from the relations between the capabilities of objects and the goals of an agent; the manifestations of these capabilities bring about a situation that impacts the goal of an agent in a very specific way: preventing risk events. In what follows we develop this conceptualization, firstly by extending COVER, then by presenting an ontology of security.

5.3.1 Extending the Common Ontology of Value and Risk

In COVER, RISK EVENT is the result of the manifestations of THREAT CAPABILITY of THREAT OBJECT and VULNERABILITY of OBJECT AT RISK or of RISK ENABLER. A THREAT EVENT is one with the potential of causing a LOSS EVENT, which brings about a LOSS SITUATION that hurts an INTENTION of an AGENT called RISK SUBJECT. (Sales et al, 2018)

The assumption that a THREAT EVENT can be intentional is implicit in COVER, so we make it explicit specializing it through the class ATTACK, an ACTION caused by an INTENTION of an AGENT called ATTACKER, which specializes THREAT OBJECT. Traditionally, the presence or not of intention in a THREAT EVENT is raised to set the difference between security and safety, respectively (Berg, Hutten, and Prins, 2021), though in both cases the goal is the prevention of the LOSS EVENT.

An important addition to COVER is the understanding that THREAT CAPABILITY, VULNERABILITY and, sometimes, INTENTION are dispositions associated with types whose instances maintain a mutual activation partnership with each other: a THREAT OBJECT can only manifest its THREAT CAPABILITY if a VULNERABILITY

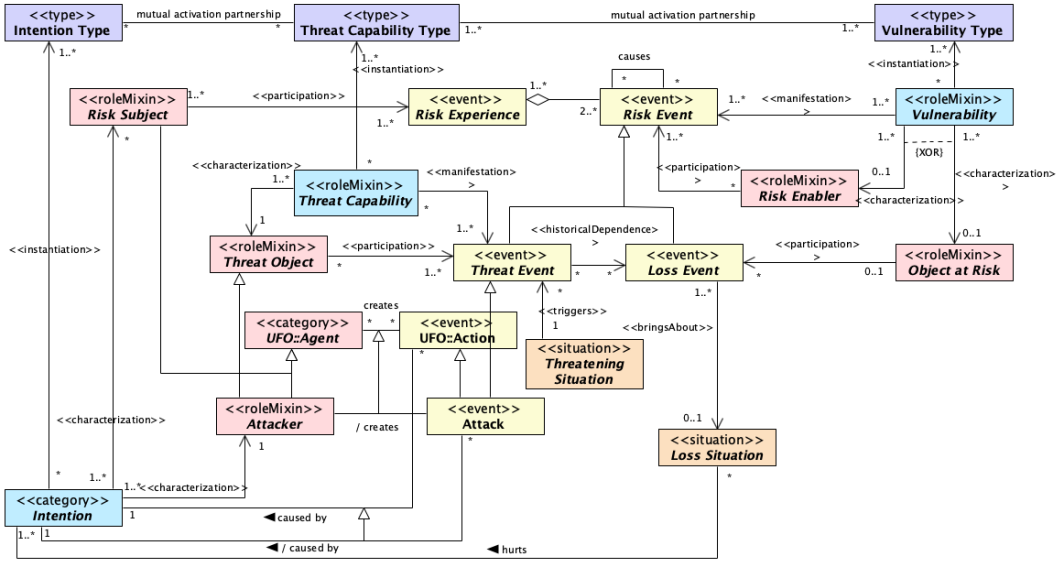

FIGURE 5.2: COVER extension concerning risk concepts and relations

can be exploited; if the THREAT OBJECT creates an ATTACK, then the INTENTION is also required. Analogously, a VULNERABILITY is only manifested in the presence of a THREAT CAPABILITY. This *generic dependence* relation among these entities determines some 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.

We need to mention briefly the notions of VALUE ASCRIPTION and RISK ASSESSMENT in COVER because they can represent the quantification of value and risk. (Sales et al, 2018) In a nutshell, an AGENT, called VALUE ASSESSOR or RISK ASSESSOR, evaluates her value and risk experience, considering the satisfaction or dissatisfaction of her goals by the manifestation of dispositions of certain objects. This judgment is then a reified relational entity to which a *quality* can be assigned - for example, like in the severity scale of risk matrix with discrete or continuous values (e.g., $\langle$*Low, Medium, High*$\rangle$).

Our extension of COVER is represented in OntoUML language in Figure 5.2. 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 Guizzardi et al., 2022. 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.

5.3.2 Unpacking the notion of security mechanism

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). The AGENT creating a SECURITY MECHANISM is not necessarily the one who is protected by its proper functioning, i.e., the PROTECTED SUBJECT. Both agents, nonetheless, 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 it designed 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, like when a person builds a wall for their own house.

An Intention can be generic or specific, according to how specific the situation that satisfies it is. For example, in the aerospace domain some goals related to the costs of the mission are generic because they can be satisfied by more funding or an assurance; even goals related to replaceable engineering parts can be satisfied by other parts of the same type. However, the completion of the mission is a specific goal that can only be satisfied by a specific situation. This distinction is important because certain security mechanisms only work for generic goals. For instance, a space company that transfers some of its risks to an insurance company can be protected from financial loss, but not from the losses caused by the explosion of a space shuttle. Ultimately, Generic Intention can only be impacted by a setting with generic Value Objects (money, for example). Still, the Specific Intention may be satisfied by a specific setting with generic Value Objects (say, the need for money under a deadline of bankruptcy).

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 dispositions called Control Capability. The manifestation of this kind of disposition is a Protection Event, specialized in Control Chain Event and Control Event, where the former can cause the latter. 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 situations of the type that triggers risk events of certain types. Since risk events are specialized in Threat Event and Loss Event, the Controlled Situation Type is incompatible with the Threatening Situation Type or with the Loss Triggering Situation Type. Figure 5.3 shows this ontological unpacking of the notion of Security Mechanism.

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.

Consider an antivirus software (Security Mechanism) in Anna's computer (Anna is a Protected Subject, but also a Risk Subject). It was designed by a software company (Security Designer) that has its own interest in seeing the antivirus capability (Control Capability), under the right settings (Protection Trigger), working properly (manifesting the Protection Event). Under the right settings (Protection Trigger), the antivirus searches for malware (the very search can be considered a Control Chain Event of the causal chain, while the malware as a software is a Threat Object). Suppose Anna's computer is infected by malware, which was a Threat Event in the process of causing a Loss Event (say, erasing Anna's files in her computer, where her files are the Object at Risk). This event of infection (an Attack) was only possible due to the conjunction of malicious Intention of someone (an Attacker), the Threat Capability of this person, and

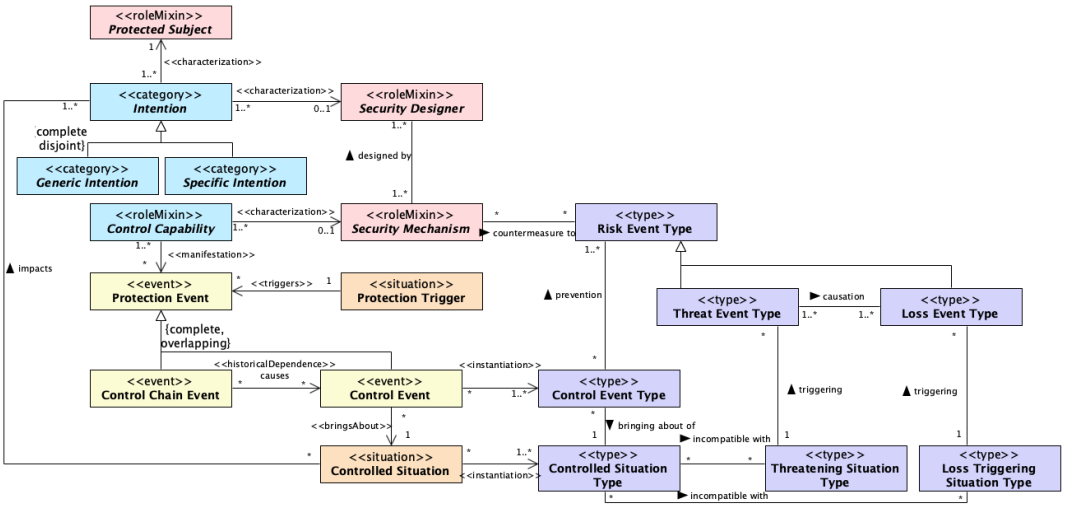

FIGURE 5.3: Unfolding Security Mechanism.

the VULNERABILITY of Anna or her computer (RISK ENABLER). However, before the manifestation of the LOSS EVENT, as the antivirus software is running, an event in the control chain causes a CONTROL EVENT of a type that is incompatible with the LOSS TRIGGERING SITUATION TYPE (say, the situations that activate the execution of malware's code to delete Anna's files), therefore preventing the LOSS EVENT of certain type (the loss of Anna's files) that would have hurt Anna's goals. Instead, a CONTROLLED SITUATION (say, Anna's computer free from the referred malware) became a fact brought about by the CONTROL EVENT (say, interrupting the malware running process and deleting it), impacting positively Anna's goals.

Notice that both kinds of indirect prevention play an important role in security: (1) when a CONTROL CHAIN EVENT indirectly prevents a RISK EVENT TYPE through the causation of a CONTROL EVENT, which prevents directly a RISK EVENT TYPE; and (2) when the CONTROL EVENT prevents indirectly the LOSS EVENT TYPE that is causally connected to the directly prevented THREAT EVENT TYPE. In the next section, we show how ROSE, which includes the extended ontology of value and risk from COVER as well as the ontology of security represented in Figure 5.3, satisfies the requirements proposed in Section 5.2.

5.4 Evaluation

ROSE incorporates a modified extended version of COVER, capturing risk and value concepts necessary to support risk identification and assessment (AR1, OR2, QR1). Indeed, by the notions of TRIGGERING LIKELIHOOD and CAUSAL LIKELIHOOD ascribed to types of situations and events, ROSE can support probabilistic assessment of value, risk, and security happenings (AR1). By the VALUE ASCRIPTION and RISK ASSESSMENT inherited from COVER, ROSE can support the comparison of choices involving values, risks, and security, so supporting the decision-making process, since in Risk Management the chances as well as the impact of risky and valuable options should be considered (AR1).

This machinery allows for the representation of risk treatment options (a) and (b) of ISO 31000 (AR2): the first case regards the case that, from the point of view of the experience of the same AGENT that is VALUE ASSESSOR and RISK ASSESSOR, the risk is assessed as higher than the expected ascribed value; in this scenario, considering the

chances of the respective events, the Agent may choose not to start or to continue the activity; the second case is the opposite, when all things considered, the possible success of the endeavor is assessed by the Agent as more valuable than its associated risks; in this scenario, the Agent may choose to pursue an opportunity, despite of the risks.

Still regarding AR2, ROSE shows the ambiguity of the risk treatment option (c) "removing the risk source" of ISO 31000 since there are multiple interacting entities that can be considered the "risk source": instances of Threat Object, Object at Risk, Risk Enabler, Threat Capability, Intention, and Vulnerability. It is possible to remove the Threat Capability without removing the Threat Object, though removing the latter implies the removal of the former due to the existential dependence of dispositions on their bearers. For example, a caged lion in a zoo is in such a situation, brought about by a caging event, such that its Threat Capability and the Vulnerability of the visitors cannot be present in the same situation, though both dispositions remain untouched. However, if the lion escapes and is sedated by a dart gunshot, the lion, while unconscious, loses its Threat Capability.

ROSE also shows the risk treatment options (c), (d), and (e) of ISO 31000 are interconnected (AR2). Since the Protection Event is an instance of Event Type that has its associated Triggering Likelihood and Causal Likelihood, the effect of prevention happens with a given likelihood. This means the chances of risk events of a certain type happening are different before and after the introduction of the Security Mechanism. The "consequences" of risk treatment option (e) are simply the loss events of certain types, but a Loss Event can be the Threat Event that causes another Loss Event - for example, a fire in the university office is a Loss Event for the university, but it is a Threat Event that can potentially harm the lives of employees.

Risk treatment option (f) of ISO 31000, "sharing the risk with another party or parties (including contracts and risk financing)", was already described as a Security Mechanism that is only applicable to protect Generic Intention by the dispositions of interchangeable Value Objects (AR2). In this case, the money or the replaceable object may be lost, but, once the equivalent reposition takes place, the events of the type associated with the initial loss are prevented. So the initial loss is both a Loss Event (the loss of money) and a Threat Event for future losses (the consequences of that).

The last risk treatment option of ISO 31000 concerns retaining the risk by informed decision, that is, the decision to be taken about *residual risks* (Katsikas, 2013), the risks left after the treatments. This option is a combination of the previous ones (AR2): it says that, once options (c), (d), (e), and (f) are implemented, we return to options (a) and (b) in an iterative decision process, as described in the standard (ISO, 2018). Again, ROSE can inform such a decision-making process by representing scenarios involving value, risk, and security. Residual risks are known to be difficult to assess (Katsikas, 2013), but ROSE offers a precise picture of the scenario before and after the security mechanism implementation (AR2).

ROSE includes all concepts mapped in Chapter 2 as the most common in security core ontologies, as requested by OR2. Indeed, ROSE ontologically unpacks them and explains their interactions in detail, also distinguishing between intentional and non-intentional threats, and how this matters for security (OR3). It is worth noting that ROSE allows for the representation of *redundancy* and multiple layers of protection in security engineering, as different security mechanisms can be designed to be countermeasures of the same type of Risk Event. This fact suggests ROSE can help to

build an ontology of resilience, as this concept can be seen as the prevention of Loss Events after the occurrence of Threat Events thanks to the effects produced by Security Mechanisms.

Concerning the quality requirements, ROSE has shown a rich set of ontological distinctions, thanks to the support of UFO and COVER, maintaining generality, which is noted, for example, by the fact the Security Mechanism can be an object of different kinds, including physical and social objects (QR2). ROSE reuses COVER with some modifications, showing a level of interoperability with a close domain (value and risk), which can be further exploited through connection to other UFO-based ontologies ((QR2, QR3). As UFO and OntoUML are formally defined in First-Order Logic, having support for an OWL implementation[1], ROSE benefits from that, in terms of formality and capacities, since it is expressed in OntoUML, making the use of ROSE for supporting formal reasoning easier. Finally, to make ROSE findable and accessible (Jacobsen et al., 2020), we provide it publicly in a repository with related information[2].

5.5 Related work

The closest related works to ours are proposals of reference ontologies of security based on some foundational ontology. In a recent literature review Oliveira et al., 2021 (Chapter 2), only four with this approach were found, while nearly all the 57 selected security core ontologies miss every FAIR principle. Casola et al, 2019 propose an ISO-based information security domain ontology, represented in OWL and designed under the principles of the Basic Formal Ontology (BFO), to facilitate the management of standards-related documents and compliance in an Information Security Management System. Massacci et al, 2011 use the upper ontology DOLCE to combine two other ontologies to support the activity of modeling security requirements, representing the proposed ontology in the Extended Backus-Naur Format. Oltramari et al., 2015, based on DOLCE, continue a previous work presenting human factors in this ontology for cyber security operations. To the best of our knowledge, none of them is publicly available besides what can be found inside each corresponding paper. Moreover, they present a limited scope concerning security, given their respective specific aims.

There exist UFO-based ontologies addressing security or related concepts. Zhou et al., 2017 propose an ontology to support hazard identification using some UFO categories, though presenting the ontology in UML, instead of OntoUML. Specific security aspects are not addressed therein. The proposal of Adach et al., 2022 is more related to ours: based on UFO, but represented in UML, a "Combined Security Ontology" (CSO) that could be aligned with other ontologies. In CSO, a countermeasure is an Action and an asset is a Kind. In ROSE, we take a different ontological interpretation of these notions. Regarding the former, an Action may be the manifestation of a Control Capability of a Security Mechanism, countermeasures are Objects, not necessarily agents, e.g., a software firewall. Regarding the latter, the type Asset cannot be a Kind, because being an asset depends on the relations the object has with other entities: firstly, nothing is necessarily an asset, but only to the extent the thing's dispositions, when manifested, satisfy someone's goals; moreover, entities of different kinds can be assets. Thus, this notion would be better modeled as role mixin in OntoUML/UFO. So CSO does not seem to commit to UFO to its full extent. The Dysfunctional Analysis Ontology (DAO) (Debbech et al., 2020) continues

[1] See: https://purl.org/nemo/gufo

[2] ROSE repository: https://purl.org/security-ontology. (QR3)

the Goal-Oriented Safety Management Ontology (GOSMO) and aims at providing a systematization of the goal-oriented dysfunctional analysis through a terminological clarification to prevent hazards. They are represented in UML and OWL, making the same (in our view, mistaken) choice of interpreting Safety Measures as an Actions.

5.6 Final considerations

We have presented an ontological analysis of security mechanisms, making explicit the relations among objects and agents, their capabilities and vulnerabilities, the events they participate in and that affect them, and the value and risk they ensue to the stakeholders at hand. The result of this analysis was a concrete artifact called *Reference Ontology for Security Engineering* (ROSE), filling a gap left by the Common Ontology of Value and Risk (COVER) that lacked security-related concepts. With the support of the theory of prevention from the Unified Foundational Ontology, our ontology shows the different general ways by which a security mechanism works. In the future, we intend to combine ROSE with other UFO-based ontologies, particularly to address legal aspects. In Chapter 8, we employ ROSE for evaluating and (re)designing the security-related elements in the ArchiMate language.

Chapter 6

Toward a phishing attack ontology

Phishing attacks are the most common form of social engineering where attackers intend to deceive targeted people into revealing sensitive information or installing malware. To understand the dynamics of phishing attacks and design suitable countermeasures, particularly the promotion of phishing awareness, cybersecurity researchers have proposed several domain conceptual models and lightweight ontologies. Despite the growing literature in ontology engineering highlighting the advantages of employing upper and reference ontologies for domain modeling, current phishing attack models lack ontological foundations. As a result, they suffer from a number of shortcomings, such as false agreements, informality, and limited interoperability. To address this gap, we propose a *Phishing Attack Ontology* (PHATO) grounded in the *Reference Ontology for Security Engineering* (ROSE) (Chapter 5) and the *Common Ontology of Value and Risk* (COVER), which are both founded in the *Unified Foundational Ontology* (UFO) (Chapter 3). Our proposal is represented through the OntoUML ontology-driven conceptual modeling language, benefiting from its ecosystem of tools and domain ontologies. We also discuss some implications of PHATO for the design of anti-phishing countermeasures.

This chapter is based on the following publication:

- Oliveira, Ítalo, Calhau, R. F., Guizzardi, G. (2023). Toward a phishing attack ontology. In: ER2023: Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium, November 06-09, 2023, Lisbon, Portugal. https://ceur-ws.org/Vol-3618/forum_paper_25.pdf.

6.1 Introduction

In cybersecurity, social engineering is a type of attack in which the attacker exploits human vulnerabilities to breach security goals (confidentiality, integrity, availability, etc.) (Wang, Sun, and Zhu, 2020). Phishing attacks are the most common form of social engineering where attackers intend to deceive targeted people into revealing sensitive information or installing malware (Internet Crime Complaint Center, 2022). In 2022, the Internet Crime Complaint Center of FBI (Federal Bureau of Investigation) reported more incidents of phishing than any other type of computer crime in the U.S. (Internet Crime Complaint Center, 2022). The same report defines phishing as “The use of unsolicited email, text messages, and telephone calls purportedly from a legitimate company requesting personal, financial, and/or login credentials”(Internet Crime Complaint Center, 2022). It is clear that phishing attacks involve different ways of combining technical and social elements.

Because of that, to understand the dynamics of phishing attacks and design appropriate countermeasures, such as phishing awareness training, cybersecurity researchers have proposed several domain conceptual models, lightweight ontologies, and informal conceptualizations of phishing (Tseng et al., 2013; Alshanfari et al., 2020; Wang et al., 2021; Tchakounté, Molengar, and Ngossaha, 2020; Mouton et al., 2014; Li, Wang, and Ni, 2022; Alkhalil et al., 2021; Lastdrager, 2014). Despite the growing literature in ontology engineering highlighting the advantages of employing upper and reference ontologies for domain modeling, all current phishing attack models lack explicit ontological foundations.

A foundational (top-level or upper) ontology is a specific consistent set of ontological theories, capable of providing support to the tasks of domain analysis, conceptual clarification, and meaning negotiation — that are critical when one has to build an ontology as a computational artifact (Guizzardi, 2020). There is evidence that top-level ontologies help with the development of high-quality core and domain ontologies, improving their consistency and interoperability (Guizzardi, 2006). A (well-founded) core ontology specifies, under a foundational ontology, the central concepts and relations of a given domain (e.g., Risk, Value, Trust, Security, etc.). Upper ontologies effectively contribute to detecting and preventing ontology design mistakes (Schulz, 2018), enhancing the quality and interoperability of domain and core ontologies (Keet, 2011). The following analogy helps to clarify this point: foundational ontologies and reference domain ontologies work as *ontology engineering frameworks*, by accelerating and improving the practice of ontology engineering, just like web development frameworks (e.g., React, Angular, Django, etc.) accelerate and improve the practice of web development.

As a result of that lack of explicit foundations, existing phishing ontologies may suffer from a number of known shortcomings, such as false agreements, informality, limited interoperability, and unintended instances. At minimum, the guidance provided by a foundational ontology can improve domain ontologies in these respects as shown, for instance, by Oliveira et al., 2023 in relation to a popular cybersecurity ontology written in OWL. To address this gap, we propose a *Phishing Attack Ontology* (PHATO)[1], a well-founded phishing model, grounded in the *Reference Ontology for Security Engineering* (ROSE) (Oliveira et al., 2022a), presented in Chapter 5, and the *Common Ontology of Value and Risk* (COVER) (Sales et al, 2018), which are both founded in the *Unified Foundational Ontology* (UFO) (Guizzardi et al., 2022), exposed in Chapter 3. Our proposal is represented through the OntoUML ontology-driven conceptual modeling language, benefiting from its ecosystem of tools and domain ontologies. We also discuss some implications of PHATO for the design of anti-phishing countermeasures.

The remainder of this chapter is structured as follows: Section 6.2 presents several common elements about phishing attacks that will be helpful to support our proposal. Section 6.3 presents the main contribution: a *Phishing Attack Ontology* (PHATO). The same Section 6.3 briefly discusses some implications for the design of anti-phishing countermeasures. Section 6.4 debates related work. Section 6.5 finishes with limitations and future work.

[1]The acronym plays with two related ideas: in Portuguese, “phato” - when the ‘ph’ is pronounced like an ‘f’ - sounds like “fato” (*fact*, in English); when it is pronounced or like a ‘p’, it sounds like “pato” (*duck*), which is a brazilian slang for gullible. Both senses come together in the idea that phishing involves lying about facts to deceive a target.

6.2 Elements of phishing attacks in cybersecurity

Phishing is a form of social engineering attack, along with baiting, pretexting, tailgating, ransomware, impersonation on the help desk, diversion theft, dumpster diving, shoulder surfing, *Quid Pro Quo*, pop-up windows, robocalls, reserve social engineering, online social engineering, phone social engineering, stealing important documents, fake software, pharming, SMSishing, whitelisting flow, and potentially others. (Salahdine and Kaabouch, 2019)

The word "phishing" is a variation of the term "fishing" where the act of phishing resembles that of fishing in the following sense: the attacker lures a victim by using a sort of bait, then fishes for personal or confidential information from the victim (Chiew, Yong, and Tan, 2018). Jakobsson, 2005 describes it as the "marriage of technology and social engineering", remarking that successful attacks use both of these components in a strategic manner. Because of that, to prevent phishing attempts and their consequences, one should understand both elements.

There are many definitions of phishing in the literature (for a list with 113 distinct definitions, see Lastdrager, 2014). According to this study, phishing can be defined as a "scalable act of deception whereby impersonation is used to obtain information from a target" (Lastdrager, 2014). The attacker can utilize various channels (emails, instant messages, voice calls, etc.) to either deceive the victim directly by a scam or to deliver payload through an indirect manner to obtain personal or confidential information (login, passwords, bank account number, etc.) from the victim (Chiew, Yong, and Tan, 2018). Sometimes, phishing involves tricking people into making them install malware, such as ransomware, which, then, will enable the stealing of confidential information or other asset. The damage caused by successful phishing attacks includes not only financial loss but also loss of reputation, fines from regulations, reduced productivity, intellectual property theft, and national security risk, affecting individuals, companies, and states.

Because phishing attacks cleverly exploit *human vulnerabilities*, they can circumvent the vast majority of an organization's or individual's security measures. As put by Hong, 2012, it doesn't matter how many firewalls, encryption software, certificates, or two-factor authentication mechanisms an organization has if the person behind the keyboard falls for a phish. Wang, Zhu, and Sun, 2021; Wang et al., 2021 enumerate (non-exhaustively) many vulnerabilities according to the following classification:

- **Cognition and Knowledge**: Ignorance, inexperience, thinking set and stereotyping, prejudice or bias, conformity, intuitive judgment, low level of need for cognition, heuristics, and mental shortcuts.
- **Behavior and Habit**: Laziness, carelessness and thoughtlessness, fixed-action patterns, habitual behaviors.
- **Emotions and Feelings**: Fear, curiosity, anger, excitement, tension, happiness, sadness, disgust, surprise, guilt, impulsion, fluke mind.
- **Human nature**: Self-love, sympathy, helpfulness, greed, gluttony, lust.
- **Personality traits**: Conscientiousness, extraversion, agreeableness, openness, neuroticism.
- **Individual characters**: Credulity, friendliness, kindness and charity, courtesy, humility, diffidence, apathy, hubris, envy.

According to Jakobsson and Myers, 2006, the most common form of phishing attacks includes three key components: the *lure*, the *hook*, and the *catch*. The lure consists of a phisher spamming a large number of users with an email message that appears to be from some legitimate institution that has a presence on the Internet. The message often uses a convincing story to encourage the user to follow a URL hyperlink embedded in the email to a website controlled by the phisher and to provide it with certain requested information. The social engineering aspect of phishing attacks typically makes itself known in the lure, as the spam offers some plausible reason for the user to provide confidential information to the website that is hyperlinked by the spam. The hook commonly consists of a website that imitates the appearance of a reputable agent (say, a famous company's website). The goal of the hook is for victims to be directed to it via the lure portion of the attack and for the victims to disclose confidential information to the site. The catch involves the phisher making use of the collected information for some illegal purpose such as fraud or identity theft.

Phishing attacks can be classified in different ways (see, for instance, Alkhalil et al., 2021). Frequently, the literature mentions "spear phishing" when the decoy is personalized to trick a specific individual or organization; "whaling phishing" when the attacker targets specifically senior executives or high-profile individuals; "vishing phishing" occurs if the attacks are performed via voice over the internet protocol (VoIP); "interactive voice response phishing" is performed by using an interactive voice response system to make the target enter the private information as if it is from a legitimate business or bank. "Business Email Compromise Phishing" mimics the whaling by targeting big "fishes" in corporate businesses to get access to their business emails, calendars, payments, accounting, or other private information. (Salahdine and Kaabouch, 2019)

6.3 A phishing attack ontology (PHATO)

Given the elements of phishing attacks described in Section 6.2 and our ontological foundations described in Chapter 3 and Chapter 5, we propose a *Phishing Attack Ontology* (PHATO)[2] by specializing the concepts of ROSE. Following this principle, we say a Scammer specializes an Attacker (a specialization of Threat Object) and *impersonates* an Impersonated Reputable Agent (a person, a company, an organization, etc.). A Scammer has an Intention to Phish and the capability to do so, an Impersonation Capability to Deceive Target, which specializes Threat Capability. It is clear that both intrinsic aspects are necessary for the manifestation of an event called Impersonation of Reputable Agent to Deceive Target wherein a Lure participates (a given email or SMS message, for example). Moreover, a third element must be in a Situation that can trigger a Phishing Contribution: an Exposure of an ancillary entity called Phishing Enabler (for instance, the target's phone number, email address, or computer network). In other words, there is a *mutual activation partnership* relation among Intention to Phish, Impersonation Capability to Deceive Target, and Exposure. They are ultimately manifested by a complex event: a Phishing Attack, a specialization of a Threat Event.

At this point, human vulnerabilities usually do not play a major role yet. However, they are essential for the manifestation of an Asset Catch, an event wherein a Hook and, naturally, an Asset participate. A number of Target's Fragilities may be present in a Vulnerability Condition that triggers Asset Catches, i.e., anyone

[2] All related files of PHATO can be found at: https://purl.org/phishing-ontology.

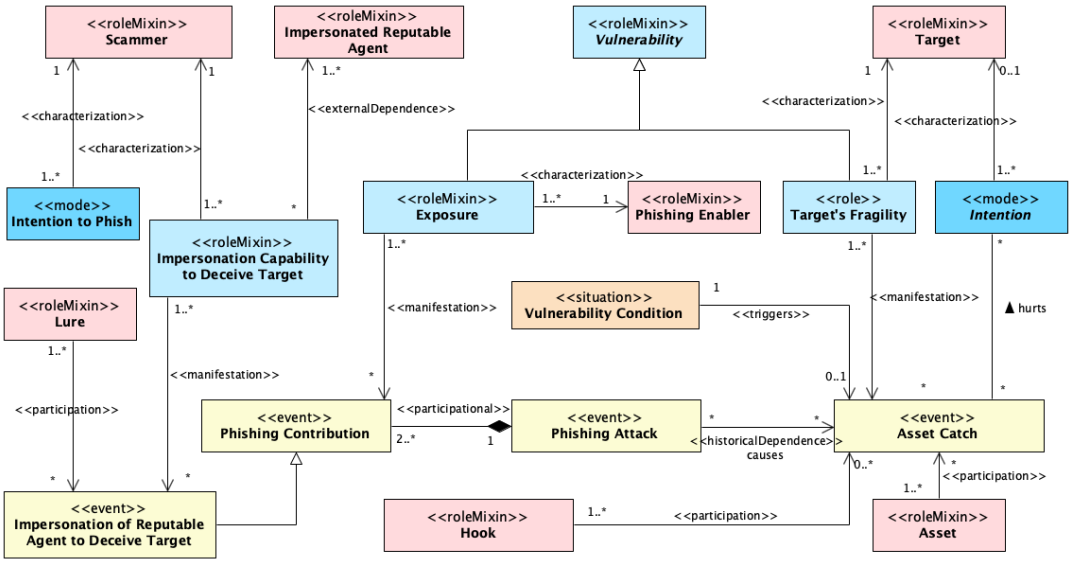

FIGURE 6.1: A Phishing Attack Ontology (PHATO).

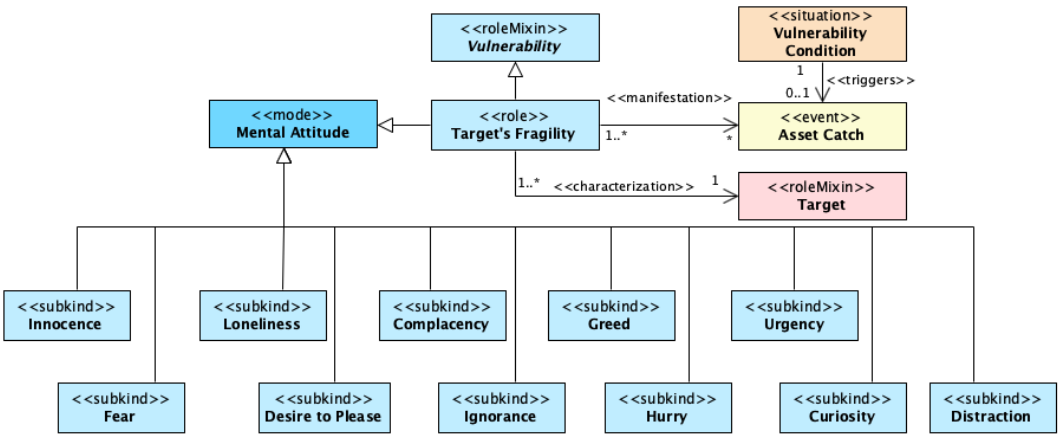

FIGURE 6.2: A non-exhaustive list of Target's Fragilities.

can fall for a phish under the right conditions. ASSET CATCHES are LOSS EVENTS that *hurt* TARGET'S INTENTIONS to preserve her ASSETS (VALUE OBJECTS, OBJECT AT RISK). A TARGET is clearly a RISK SUBJECT.

As described in Section 6.2, many human mental attitudes can play the role of a TARGET'S FRAGILITIES, such as INNOCENCE, FEAR, COMPLACENCY, DESIRE TO PLEASE, GREED, IGNORANCE, CURIOSITY, URGENCY, DISTRACTION, LONELINESS, just to cite a few. Figure 6.2 displays a non-exhaustive list of them, whereas Figure 6.1 presents the core elements of PHATO. Our ontology is rich enough to allow the specialization of several important concepts to achieve a better classification of the entities within the domain. For example, types of LURE can correspond to different strategies or messages employed by a SCAMMER. PHISHING ATTACK can be classified into SPEAR PHISHING ATTACK, WHALING PHISHING ATTACK, etc. The role of ASSET can be played by PASSWORD, LOGIN, etc. Instances of HOOK can be phishing websites. The rich scheme of PHATO may support the design of datasets or their integration for, e.g., machine learning tasks in this area.

With the support of ROSE and COVER, it is possible to assign a given likelihood that instances of a type of VULNERABILITY CONDITION trigger ASSET CATCH events

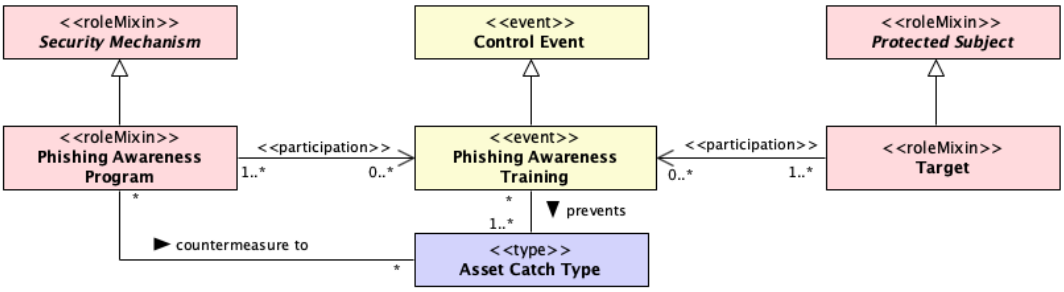

FIGURE 6.3: An example of anti-phishing countermeasure, Phishing Awareness Program.

of a particular type. We can say, for instance, that the more fragilities a person bears, the higher the chances of their falling for a phishing attack. Phishing awareness training, from this perspective, is a sort of CONTROL EVENT that eliminates or attenuates certain MENTAL ATTITUDES or builds new CONTROL CAPABILITIES so that these MENTAL ATTITUDES no longer play the role a fragility (e.g., when one is able to control their curiosity, greed or fear). This, in turn, helps preventing ASSET CATCHES to some degree. For example, the TARGET acquires cybersecurity knowledge through the training, which can eliminate TARGET's IGNORANCE in relation to some common LURE. Consequently, the associated ASSET CATCH events are prevented because the type of SITUATION that could trigger them has been ruled out. Similarly, it is possible to assign a given likelihood for instances of a type of PHISHING ATTACK cause ASSET CATCH events of a specific type. In other words, we can analyze which kinds of PHISHING ATTACKS are the most successful at capturing ASSETS.

By representing a number of interconnected entities that are relevant for PHISHING ATTACKS and ASSET CATCH events, PHATO can support the design of suitable countermeasures. For example, phishing awareness training is one of the most efficient ways of preventing people from falling for a phish (Jansson and Solms, 2013). Figure 6.3 displays such a case where a PHISHING AWARENESS PROGRAM is designed to be a *countermeasure to* ASSET CATCH events of a certain type by the manifestation of PHISHING AWARENESS TRAINING wherein the TARGETS participate. In this case, a PHISHING AWARENESS PROGRAM is a social entity whose capabilities are manifested by PHISHING AWARENESS TRAINING, which may remove some of TARGET'S FRAGILITIES, therefore preventing certain types of ASSET CATCH event.

6.4 Related work

Although there are numerous ontology-based works in security and cybersecurity, recent literature reviews (Oliveira et al., 2021; Martins et al., 2020; Martins et al., 2022) have shown they are mostly focused on specific applications and lack ontological foundations. As a consequence, an in-depth ontological analysis of phishing attacks is missing. For example, Tseng et al., 2013 are interested in automated phishing detection with the aid of a proposed taxonomy. Similarly, in Tchakounté, Molengar, and Ngossaha, 2020 a description logic and OWL ontologies are proposed to represent scenarios of e-mail phishing attacks. The latter also presents a list of ontology-based works, which are lightweight (DL, RDF, OWL, frames) and application-focused. A few of these works resemble an ontological account but they focus on social engineering in general, not phishing attacks. Even these offer no more than an OWL (Alshanfari et al., 2020; Wang et al., 2021) or UML-like (Mouton et al., 2014; Li, Wang, and

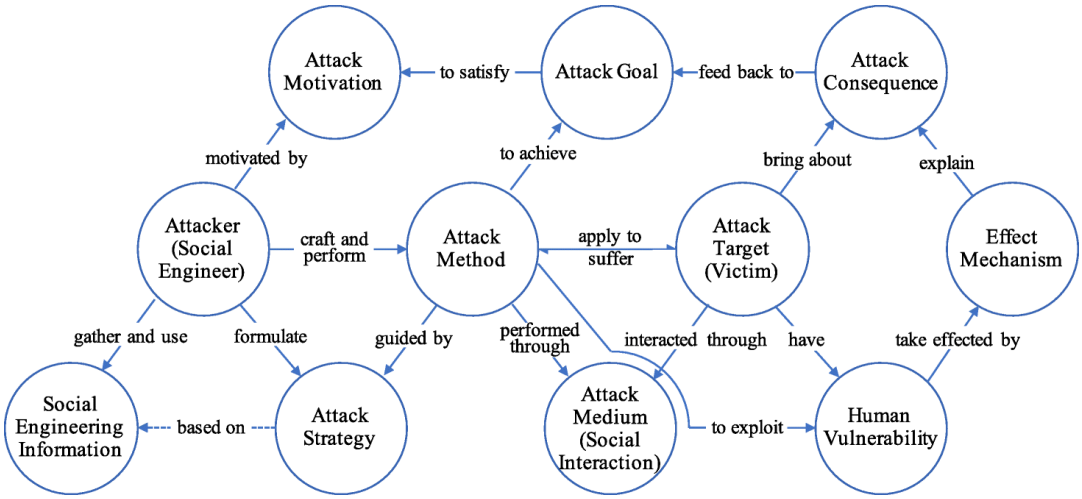

FIGURE 6.4: The domain ontology of social engineering in cybersecurity proposed by Wang *et al* Wang et al., 2021.

Ni, 2022) ontology. Just like noticed by Oliveira et al., 2021 for security ontologies, we could not find publicly any artifacts reported by this literature, which makes any evaluation more difficult (for instance, to check logical consistency and unintended instances). Therefore, to the best of our knowledge, the present work is the first analysis and conceptualization of phishing attacks employing a foundational ontology. For comparison, Figure 6.4 depicts an interesting proposal of a domain ontology of social engineering by Wang et al., 2021, where we can clearly notice the lack of ontological distinctions among the classes (objects, events, modes, situations, etc.). This may yield too many unintended instances, as shown by Oliveira et al., 2023 by analyzing a well-known cybersecurity ontology, as we will see in Chapter 7.

6.5 Final considerations

Dealing appropriately with phishing attacks is currently one of the main challenges in the cybersecurity field. They pose a special threat to people's and organization's assets by combining smartly social and technical elements. Understanding and modeling phishing attacks are part of the solution. However, current models present a number of limitations due to their lack of ontological foundations, such as informality and unintended instances. With the aid of the *Reference Ontology for Security Engineering* (ROSE), which is based on the *Common Ontology of Value and Risk* (COVER) and the *Unified Foundational Ontology* (UFO), we propose the first well-founded *Phishing Attack Ontology* (PHATO). We have shown that PHATO represents the key elements of phishing attacks found in the research literature. We also discussed some implications of PHATO for the design of anti-phishing countermeasures.

However, PHATO, naturally, needs further validation, such as web semantic applications, expert assessment, formal validation, integration of datasets, and others. Furthermore, integrating PHATO with a *competence ontology* (Calhau, Azevedo, and Almeida, 2021; Calhau and Almeida, 2022; Calhau et al., 2023b) can improve its capability of modeling human factors involved in phishing attacks and countermeasures. For example, competencies as types of MENTAL ATTITUDES that counteract TARGET'S FRAGILITIES: critical thinking, cybersecurity knowledge and skills, etc.

Because these competencies often emerge from the interaction of other MENTAL ATTITUDES, a *system core ontology* (Calhau et al., 2023a) can come in handy. We intend to continue this work in this direction in the future.

Part III

PRACTICAL APPLICATIONS

Chapter 7

An ontological analysis of D3FEND cybersecurity model

Formal Ontology is a discipline whose business is to develop formal theories about general aspects of reality such as identity, dependence, parthood, truthmaking, causality, etc. A foundational ontology is a specific consistent set of these ontological theories that support activities such as domain analysis, conceptual clarification, and meaning negotiation. A (well-founded) core ontology specifies, under a foundational ontology, the central concepts and relations of a given domain. Foundational and core ontologies can be seen as *ontology engineering frameworks* to systematically address the laborious task of building large (more specific) domain ontologies. However, in research and industry, it is common that ontologies as computational artifacts are built without the aid of any framework of this kind, often yielding modeling mistakes and representation gaps. In this chapter, we analyze a case in the domain of cybersecurity, namely, the D3FEND artifact - an OWL knowledge graph of cybersecurity countermeasure techniques proposed by the MITRE Corporation. Based on the *Reference Ontology for Security Engineering* (ROSE), presented in Chapter 5, our investigation reveals semantic issues and opportunities for improvement in D3FEND, including missing concepts, semantic overload of terms, and lack of constraints that cause an under-specification of the model. As a result of our ontological analysis, we propose several suggestions for the appropriate redesign of D3FEND to overcome those issues.

This chapter is based on the following publication:

- Oliveira, Ítalo., Engelberg, G., Barcelos, P.P.F., Sales, T.P., Fumagalli, M., Baratella, R., Klein, D., Guizzardi, G., (2023) Boosting D3FEND: Ontological analysis and recommendations. In: Formal Ontology in Information Systems: Proceedings of the Thirteenth International Conference (FOIS 2023). Vol. 377. Frontiers in Artificial Intelligence and Applications. IOS Press. https://ebooks.iospress.nl/doi/10.3233/FAIA231138.

7.1 Introduction

In any field, as the complexity of the domain grows, there is a need to standardize the interpretation of domain notions for both human communication and machine inferencing. Taxonomies offer a first step in this direction listing classes of things that are subsumed under certain categories. Ontologies as computational artifacts represent a network of concepts, relations, and constraints about the domain at hand.

In contrast, in philosophy, Formal Ontology is a discipline that aims to develop formal theories about general aspects of reality, including the definition of identity, properties, dependence, part-whole relation, causality, events, etc. A foundational (top-level or upper) ontology is a specific consistent set of these ontological theories,

capable of providing support to the tasks of domain analysis, conceptual clarification, and meaning negotiation — that are crucial when one has to build an ontology as a computational artifact (Guizzardi, 2020). Indeed, top-level ontologies help with the development of high-quality core and domain ontologies, improving their consistency and interoperability (Guizzardi, 2006). A (well-founded) core ontology specifies, under a foundational ontology, the central concepts and relations of a given domain (e.g., Risk, Value, Trust, Security, etc.). Upper ontologies effectively contribute to detecting and preventing ontology design mistakes (Schulz, 2018), enhancing the quality and interoperability of domain and core ontologies (Keet, 2011). To make an analogy, foundational ontologies, and reference domain ontologies work as *ontology engineering frameworks*, by accelerating and improving the practice of ontology engineering, just like web development frameworks (e.g., React, Angular, Django, etc.) accelerate and improve the practice of web development. Nevertheless, surprisingly, both in research and industry, ontologies as computational artifacts are often built without the aid of any framework of this kind (Oliveira et al., 2021; Martins et al., 2022), favoring recurrent modeling mistakes and gaps.

In this chapter, we dive into the domain of security as a particular case study. In this domain, the need for ontology development was already acknowledged two decades ago by Donner, 2003, while a recent systematic mapping study of the literature (presented in Chapter 2) has revealed the limitations of the current security ontologies (Oliveira et al., 2021). In particular, this latter study shows that foundational ontologies are seldom used in the practice of engineering these artifacts. More specifically, in cybersecurity, the situation is not different, as shown by Martins et al., 2020; Martins et al., 2022. In this domain, an artifact that stands out by its increasing popularity among practitioners and scholars is D3FEND. It is a novel knowledge graph of cybersecurity countermeasure techniques proposed by the MITRE Corporation (Kaloroumakis and Smith, 2021), which aggregates a catalog of defensive cybersecurity techniques and their relationships to offensive techniques. D3FEND's primary goal is to help standardize the vocabulary used to describe defensive cybersecurity technology functionality. Recent cybersecurity studies make use of it for the process of identification and assessment of cyber threats, and response against them (Kaiser et al., 2022; Sadlek, Čeleda, and Tovarňák, 2022; Shin et al., 2022; Akbar et al., 2022; Aghamohammadpour, Mahdipour, and Attarzadeh, 2022), among other applications, including the design of a game to support security education and risk assessment (Luh et al., 2022). D3FEND is also an example of an ontology developed without an explicit tie to an upper ontology, and to the best of our knowledge, there is no systemic ontological analysis of this artifact, whose validity seems to be taken for granted. The combination of these factors makes D3FEND an interesting target for our analysis: *What possible issues does D3FEND present due to its lack of foundations? How could we address them?*

Based on the *Reference Ontology for Security Engineering* (ROSE) (Oliveira et al., 2022a), presented in Chapter 5, which is a core ontology of the security domain founded in the *Unified Foundational Ontology* (UFO) (Guizzardi et al, 2015; Guizzardi et al., 2022), we proceed with an ontological analysis of the conceptual model behind D3FEND, revealing several semantic issues and opportunities for improvement that could be addressed by relying on a foundational ontology as support. In particular, our analysis identifies cases of *semantic overload*, *missing concepts*, and a *systematic lack of constraints*. Under the assumptions of UFO and ROSE, we also suggest how the issues identified can be solved, thus contributing to improving D3FEND accordingly. The implications of our analysis make a case in favor of employing foundational and reference ontologies in ontology engineering practice, as captured

by Varzi's dictum "No ontology without Ontology" (Varzi, 2019). Through this work, we expect to contribute to the development of the ontology engineering practice in cybersecurity, in general, and the D3FEND project, in particular.

The remainder of this chapter is structured as follows: section 7.2 briefly presents the object of our analysis, namely, D3FEND. Section 7.3 employs ROSE and UFO to proceed with an ontological analysis of D3FEND. In doing that, we identify both general semantic issues of its conceptual model as exposed by UFO, as well as domain-specific issues regarding its conceptualization of security, according to ROSE. In Section 7.4, we indicate several concrete opportunities for improving D3FEND according to our frameworks and analysis; Section 7.5 concludes this chapter by presenting some final considerations.

7.2 The D3FEND knowledge graph of cybersecurity countermeasures

Given the necessity of specifying cybersecurity countermeasures and capabilities, a team at the MITRE Corporation has built D3FEND (Kaloroumakis and Smith, 2021) (which stands for "Detection, Denial, and Disruption Framework Empowering Network Defense")[1]. The motivation is that practitioners should know not only what threats a capability claims to address, but also how exactly these threats are addressed from a security architecture and engineering viewpoint, and under what conditions a solution would work. This is particularly important, for example, to inform acquisitions and investigations in cybersecurity.

D3FEND is an OWL specification representing types and relations that aim to define both the central concepts in the cybersecurity countermeasure domain and the relations necessary to connect those concepts to each other. The process of construction of D3FEND has followed a *bottom-up approach*, by surveying patents from the U.S. Patent Office, existing knowledge bases (MITRE Cyber Analytic Repository, ATT&CK knowledge base), and other data sources (academic papers, technical specifications, and publicly available product technical documentation) (Kaloroumakis and Smith, 2021). The creators of D3FEND have made a deliberate choice to defer alignment to a foundational or reference ontology in a *top-down approach*[2].

A cybersecurity countermeasure is understood as "any process or technology developed to negate or offset offensive cyber activities" (Kaloroumakis and Smith, 2021). D3FEND is intended to provide not only an understanding of what a countermeasure does but also how it does what it does. D3FEND does not prescribe specific countermeasures, nor does it evaluate their effectiveness and priority. However, by standardizing the vocabulary of cybersecurity countermeasures, D3FEND may support these activities. The primary audience of D3FEND is security systems architecture experts and technical executives who make acquisition or investment decisions.

The fundamental idea of the D3FEND model involves relating Offensive Techniques, taken from a portion of MITRE's ATT&CK framework[3], and Defensive Techniques through Digital Artifacts. By using OWL and SPARQL reasoning services, D3FEND is able to show which Defensive Techniques somehow counter

[1]The D3FEND official website is https://d3fend.mitre.org/. Here is MITRE's announcement of D3FEND:
https://www.mitre.org/news-insights/impact-story/mitres-d3fend-connects-cyber-community-counter-threats.

[2]Personal communication with Peter Kaloroumakis and others directly involved in the creation of D3FEND (in the D3FEND Slack channel).

[3]MITRE ATT&CK is a knowledge base of adversary tactics and techniques based on real-world observations. Here is its official website: https://attack.mitre.org/.

which Offensive Techniques due to the mediation of Digital Artifacts with which they are both *associated*. This also defines the scope of D3FEND since it does not include administrative and supportive countermeasure functionalities, but only those that directly counter adversary behavior. Moreover, any measure that is not directly related to Digital Artifacts is not under D3FEND's scope. For instance, a strong password policy is in its scope because it directly affects an organization's technology configuration baseline, thus it involves digital artifacts. In contrast, investments in employee cybersecurity awareness through training programs do not directly interact with Digital Artifacts, so this kind of measure is outside D3FEND's scope. (Kaloroumakis and Smith, 2021)

Whereas the ATT&CK framework (Strom et al., 2020) deals with adversary behavior via Offensive Techniques organized by the tactical objectives they support (Offensive Tactic), D3FEND deals with Defensive Techniques organized by the tactical objectives they support (Defensive Tactic). Digital Artifacts are in-between, being affected by both offensive and defensive techniques. Tactics represent "the what" of an action, a defensive or offensive goal to be achieved by the means of Techniques ("the how"), which *enable* the Tactics. Offensive Tactics subsume Collection, Command and Control, Credential Access, Execution, among others. Defensive Tactics subsume Deceive, Evict, Detect, Harden, among others. None of these lists are intended to be exhaustive. Finally, events are introduced by the concept Digital Event, but it is still a work in progress with a minor role in D3FEND's main use cases.

7.3 Ontological analysis of the D3FEND knowledge graph

We have already explained the method of ontological analysis in Chapter 1.4, summarized in Figure 1.3. But let us recap it here for convenience. In Rosemann, Green, and Indulska, 2004, an ontological analysis framework is described. The general idea is to compare two ontologies (as domain descriptions), assuming one is the reference to assess the other. Notice, however, that as shown by Guizzardi, 2007a, this sort of analysis is more than a matter of direct comparison between the actual structures of these models; it is a matter of reconstructing the underlying intended conceptualizations of these models, i.e., about making their ontological assumptions explicit. Here, UFO and ROSE are our references to analyze D3FEND. Specifically, three recurrent semantic deficiencies of D3FEND will be demonstrated in the sequel, namely: (a) *ontological incompleteness*, when there is an element in the reference ontology that finds no representation in the evaluated ontology; (b) *construct overload*, when two disjoint notions in the reference ontology are represented by the very same element in the evaluated ontology; (c) *under-specification*, when missing domain constraints allow for unintended models of the ontology.

Before we delve into ontological issues, we should highlight that D3FEND is a work in progress. When we made the analysis reported in this chapter, the latest version of D3FEND at the time (named '0.11.0-BETA-1') was *logically inconsistent*, which can be shown by the reasoners (Pellet or FaCT++ 1.6.5, for example) available on the Protégé ontology editor[4]. For this reason, in our analysis, we work with a previous (beta) version of D3FEND (named '0.10.1-BETA-1'), released in June

[4]Protégé is available at `https://protege.stanford.edu/`.

2022[5]. The goal of this analysis is not to cover all semantic issues within D3FEND, but primarily to highlight a few of them that could have been prevented (and which can be fixed) by relying on the support of a foundational ontology. These problems if not properly addressed can impact the reusability, interoperability, and *domain appropriateness* of that artefact (Guizzardi, 2007a). An ontology should not only capture intended instances (scenarios that satisfy the ontology specification) but also exclude unintended ones (Guarino, Oberle, and Staab, 2009). Our analysis intends to show that the analyzed version of D3FEND fails in capturing intended instances and excluding unintended ones. For transparency and reproducibility of this research, all the related files are publicly available at `https://purl.org/d3fend-analysis`.

7.3.1 General semantic issues within D3FEND

The analyzed version of D3FEND is clearly under-specified thus missing many important constraints. In particular, it systematically lacks many constraints that should establish *disjointness* between classes. In other words, several classes that, even *intuitively*, are expected to be disjoint are not disjoint, including Digital Artifact and Physical Artifact (for example, Hardware Device), Physical Object and Digital Object, Digital Artifact and Digital Event, Physical Location and Physical Object, among others. Actually, similar issues have been found in other large ontologies, such as Schema.org[6], where, for example, LocalBusiness is both a Place and an Organization[7]. In this case specifically, under UFO assumptions, Organization and Place can be seen as different Objects with different unique principles of identity, so they cannot be a subtype of one another (Guarino and Welty, 2004; Guizzardi, 2006). The case of D3FEND is often simpler than that because it involves confusion between, for instance, an Event, an Aspect, and an Object, represented by Figure 3.1, as we will see — a case of construct overload. *We conjecture that, without the aid of the systematic taxonomy of a foundational ontology, ontology engineers usually drop constraints to avoid inconsistencies as the ontology gets bigger, consequently admitting unintended instances.*

In Figure 7.1, we represent a fragment of D3FEND, as an UML class diagram, in order to show several semantic issues revealed as unintended subsumption relations. Figure 7.1 displays the general idea that (offensive or defensive) Techniques are *associated with* Digital Artifacts and can *enable* Offensive or Defensive Tactics.

- **Physical Objects and Locations and Digital Artifacts:** By inference, Physical Object $\sqsubseteq$ Digital Artifact. Consider that an Artifact, in D3FEND, according to Wordnet[8], is "a man-made object taken as a whole". Clearly, not all Physical Objects are artifacts, let alone Digital Artifacts. Moreover, Physical Objects (i.e., things that exist in the world having spatial extension) are necessarily not digital objects and, hence, not Digital Artifacts. Furthermore, we have that Physical Location $\sqsubseteq$ Digital Artifact, which can be criticized on the same grounds.

[5] While the study of this chapter was being reviewed, another version of D3FEND was released in January 2023, named '0.12.0-BETA-2', which fixed some of the issues that we identified in our analysis, including the logical inconsistency found by the reasoners. Nonetheless, our main argument about the importance of ontological foundations in the ontology engineering practice still holds.

[6] See: `https://schema.org/`.

[7] LocalBusiness in Schema.org: `https://schema.org/LocalBusiness`.

[8] See: `http://wordnet-rdf.princeton.edu/id/00022119-n`.

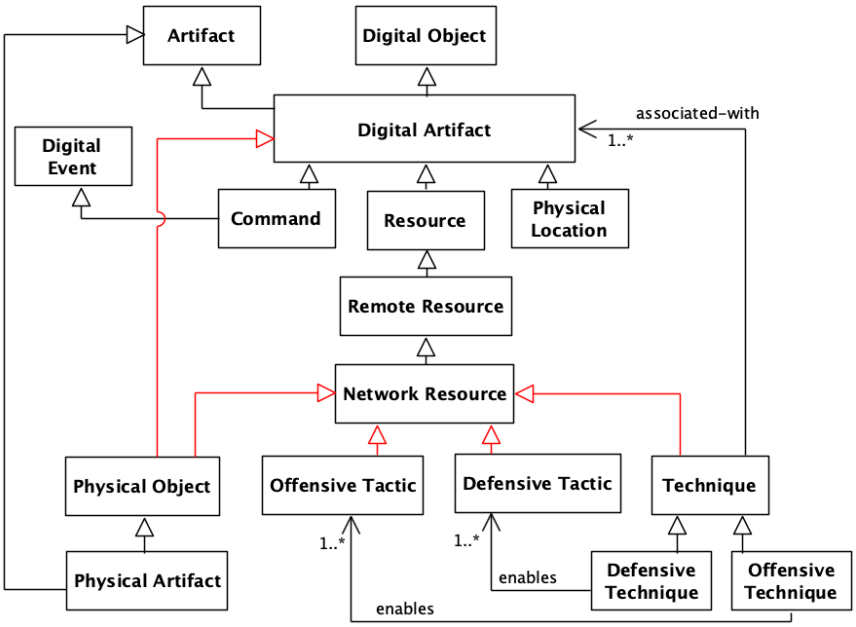

FIGURE 7.1: A fragment of D3FEND 0.10.1-BETA-1 expressed as a UML class diagram. Black elements are asserted in the ontology. Red elements are inferred.

- **Tactic and Technique:** TECHNIQUE are specialized into offensive and defensive, although *incompletely* and with *type overlapping*. This means that an instance of DEFENSIVE TECHNIQUE can also be an instance of OFFENSIVE TECHNIQUE, and that individuals that are neither of these can be an instance of TECHNIQUE. Likewise, there are OFFENSIVE TACTIC and DEFENSIVE TACTIC, although there is no explicit generalization set called TACTIC within D3FEND. The meanings of these categories are unclear. We believe that (offensive or defensive) TACTIC can be interpreted as an INTENTION (to perform certain actions), which in UFO is an ASPECT (more specifically, an INTRINSIC MODE). TECHNIQUE seems to either refer to a NORMATIVE DESCRIPTION (Guizzardi et al., 2008) describing an EVENT TYPE or that EVENT TYPE itself. MODES, NORMATIVE DESCRIPTIONS or, more generally, OBJECT (of which NORMATIVE DESCRIPTION is a subtype) and EVENT TYPES are mutually disjoint categories. By inference, TECHNIQUE is a DIGITAL OBJECT, but it is commonly annotated with definitions that use action verbs (which suggests its strong connection to EVENT TYPES). Moreover, D3FEND also contains the following constraints: TECHNIQUE $\sqsubseteq$ REMOTE RESOURCE and OFFENSIVE TACTIC $\sqcup$ DEFENSIVE TACTIC $\sqsubseteq$ NETWORK RESOURCE. These are clearly unintended constraints, particularly when we notice that NETWORK RESOURCE $\sqsubseteq$ DIGITAL ARTIFACT, which means that a TACTIC can be a DIGITAL ARTIFACT, mixing up INTENTIONS with OBJECTS, and, allowing that a TACTIC could be also a TECHNIQUE. It seems that the notion of TECHNIQUE collapses ontologically different entities (construct overload) while at the same time suffering from the systematic lack of proper constraints.

- **Types and Instances:** in UFO, individuals are instances of at least one type. D3FEND, actually, makes use of types explicitly: REFERENCE TYPE and REFERENCE. However, they bear no relation to each other. The REFERENCE TYPES include, e.g., the individuals PATENT and INTERNET ARTICLE, which seem to

be better categorized as types rather than individuals. At the same time, there are classes called Patent and Internet Article, which, however, do not include those individuals as their instances.

- **Digital Artifacts and Digital Events:** Numerous classes are *explicitly* asserted (i.e., not inferred) to be simultaneously a Digital Artifact and a Digital Event, such as Command, DNS Lookup, User Action, System Call. Once again, an unintended fusion between an Event and an Object, which should have been defined as disjoint ontological categories (Figure 3.1).

As a final example of missing constraints, we made an experiment by creating an individual that is, *concomitantly*, a Defensive Tactic, a Defensive Technique, a Digital Event, a Digital Object, an Offensive Tactic, an Offensive Technique, an Agent, a Proposition, a Sensor, an Assessment, a Physical Location, a Physical Object, a Reference, and a Reference Type. In contrast with what one would expect from an ontology capturing the real-world semantics of these notions, no inconsistency results from that.

7.3.2 Domain-specific ontological issues within D3FEND

ROSE is not an ontology of cybersecurity *per se*. However, it is an ontology of security engineering that can be specialized to capture subdomains of security, including cybersecurity. We here show that D3FEND does not address a number of questions and patterns of security, which negatively impacts its expressivity/domain appropriateness. Particularly, D3FEND core conceptual model contains three interconnected main ontologies: digital artifact, attack, and defense ontologies. From a ROSE perspective, they can be seen as, respectively, the ontologies of value, risk, and security. With these notions in mind, we can directly identify domain appropriateness issues in D3FEND.

The first issue refers to the lack of subjects within D3FEND. Although it includes the concept of Agent, which subsumes Organization and Person[9], it does not seem to play a role in D3FEND's core conceptual model. As ROSE and COVER show, the phenomena of value, risk, and security depend on the subjects' Intentions that are affected by the manifestations of dispositions. As an Aspect, an Intention is existentially dependent on their bearers, the subjects. In other words, D3FEND currently lacks the subjects that would bear Offensive or Defensive Tactics. The practical implication of this deficit is that it is not possible to recognize which Persons and Organizations are being affected by the Techniques that are associated with Tactics. In summary, this is a case of ontological incompleteness regarding Value Subject, Risk Subject, Protected Subject, and Security Designer.

Analogously, the concepts of Threat Object and Attacker are currently missing in D3FEND, so it is not possible to identify the Objects that are sources of a Risk Event or a Attack. Furthermore, the conditions that favor the appearance of a Risk Event (Threatening Situation) or the conditions that favor the occurrence of a Control Event (Protection Trigger) are absent. As a result, we cannot properly describe and assess the situations associated with risk or security.

Then, considering that a cybersecurity countermeasure is defined as "any process or technology developed to negate or offset offensive cyber activities" (Kaloroumakis and Smith, 2021), there is a blending of different entities that compose the UFO-B

[9]Notice that, in D3FEND, Person and Organization are not disjoint classes, and the former is oddly a subclass of ∃*has-member*.Person.

pattern shown by Figure 3.3, which appears within the ontologies of value (Figure 5.1), risk (Figure 5.2), and security (Figure 5.3): the notion of Technique obfuscates the distinction between an Object (say, a Security Mechanism, its capability (say, a Control Capability), the event or process that is the manifestation of this capability (Control Event), and the resulting state of the world (Controlled Situation) that impacts (positively or negatively) an Intention of a subject. This is a clear case of construct overload. D3FEND, actually, includes a notion of Capability, but which, ontologically speaking, must be interpreted either as an Intention or a Proposition, given that it is subsumed by Capability Feature Claim. In any case, neither intentions nor propositions are capabilities, in fact, all these types are (again) mutually disjoint. Curiously, D3FEND does not include the notion of Vulnerability, which is one of the most common concepts among security Oliveira et al., 2021 and cybersecurity Martins et al., 2020 ontologies.

7.4 Concrete proposals for improving D3FEND

Proposing a complete analysis and improvement of D3FEND is out of the scope of this work. However, we can indicate benefits that can be incorporated into D3FEND according to the ROSE/UFO ontological framework and the analysis conducted here. Based on ROSE, the strategy we suggest is the following: (1) specializing the value ontology with cybersecurity domain-specific entities to capture D3FEND's digital artifact ontology; (2) specializing the risk ontology into the cybersecurity domain to capture the ATT&CK framework — D3FEND's attack ontology (the entities under the class named 'ATTACK Thing' defined in the OWL file); (3) specializing the security ontology into the cybersecurity domain to capture the defensive dimension of D3FEND.

In general, the taxonomic parts of D3FEND can be, systematically, added to its improved version, including the lists of Offensive Tactic, Offensive Technique, Defensive Tactic, Defensive Technique, and Digital Artifact, but now introducing the right constraints inherited from UFO and ROSE, thus, removing inconsistencies. Moreover, interestingly, MITRE's ATT&CK framework maintains a list of threat groups cataloged by security community[10], but D3FEND does not make use of it. This catalog would be suitable to populate/instantiate the Threat Object category.

Information security practitioners and scholars often refer to the notions of *Confidentiality*, *Integrity*, and *Availability* as the "CIA triad", the fundamental elements of security controls in information systems (Samonas and Coss, 2014). They can be used not only to specialize the subjects' Intentions (security goals (Sumra, Hasbullah, and AbManan, 2014)) but also to specialize Loss Event and Loss Situation. The latter would result in an incomplete generalization set with the derived types Loss of Confidentiality, Loss of Integrity, and Loss of Integrity. Currently, D3FEND lacks these domain-specific distinctions.

We advocate that following the suggestions and underlying principles discussed in this chapter would produce an ontologically improved version of D3FEND. However, the resulting artifact would also have the additional benefit of facilitated interoperability with other UFO-based ontologies in related domains including, naturally, those in the domain of risk (Sales et al, 2018) and risk propagation (Fumagalli et al., 2023) but also Trust (Amaral et al., 2019) and Law.

[10]List of threat groups, registered by MITRE: https://attack.mitre.org/groups/.

7.5 Final considerations

Ontology engineering is not an easy task. One of its main challenges involves the appropriate conceptualization of the domain of interest because the ontology is supposed to not only correctly represent the domain but also excluded unintended interpretations. To address this problem there are foundational and reference ontologies, exemplified by the Unified Foundational Ontology's (UFO) ecosystem. Still, ontologies, as computational artifacts, are often built without the assistance of this sort of ontology engineering framework.

D3FEND is a novel OWL knowledge graph of cybersecurity countermeasures that is gaining popularity among practitioners and academics alike. Exactly because it is practically relevant as well as a work in progress, we believe it can substantially benefit from processes of detailed ontological analyses and systematic improvement recommendations in line with what we put forth in this chapter.

In this chapter, with the support of UFO and the Reference Ontology for Security Engineering (ROSE), we systematically identify several semantic issues and opportunities for improvement within D3FEND. These issues include cases of semantic overload, ontological incompleteness (both at a general and domain level), and recurrent lack of constraints (underspecification). They dent D3FEND's reusability, interoperability, and correctness with regard to the cybersecurity domain. More to the point, they could have been avoided (and can be addressed) with the support of a foundational ontology. So, this case adds to the growing evidence supporting the thesis that ontological foundations really matter in the ontology engineering practice.

As previously mentioned, Oliveira et al., 2021 and Martins et al., 2020, respectively, systematically analyzed a multitude of ontologies in the security and, more specifically, cybersecurity domains. As shown there, very few of these ontologies have been developed with the support of foundational ontologies - despite the criticality of the domain. However, to the best of our knowledge, there is no similar work providing an ontological analysis of, and improvement recommendation proposal for, the D3FEND knowledge graph - again, despite the wide practical impact and diffusion of that ontology among practitioners. So, we believe this chapter brings contributions to the development of the ontology engineering practice in cybersecurity, in general, and the D3FEND project, in particular.

The natural next step of our research is to: (1) propose a complete ontological analysis of D3FEND with the support of ROSE/UFO, aiming at exhaustively identifying semantic issues and opportunities for improvement; (2) generate the OWL version of the improved D3FEND knowledge graph with the support of gUFO[11]; (3) demonstrate the implications of these improvement interventions in real-world use cases, such as cybersecurity risk assessment. In addition to that, as a complementary work, we shall conduct an analogous ontological analysis of the complementary ATT&CK framework.

[11] gUFO - standing for gentle UFO - is a lightweight OWL implementation of the UFO ontology (Almeida et al., 2020).

Chapter 8

Ontology-based security modeling in ArchiMate

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 is able to describe risk treatment options, according to ISO 31000. To illustrate our proposal, besides presenting multiple application examples, we proceed with a real-world elucidative study case from the cybersecurity domain.

This chapter is based on the following papers:

- Oliveira, Í., Sales, T.P., Almeida, J.P.A., Baratella, R., Fumagalli, M., Guizzardi, G. (2022). Ontological Analysis and Redesign of Security Modeling in ArchiMate. In: Barn, B.S., Sandkuhl, K. (eds) The Practice of Enterprise Modeling. PoEM 2022. Lecture Notes in Business Information Processing, vol 456. Springer, Cham. https://doi.org/10.1007/978-3-031-21488-2_6.
- Oliveira, Í., Sales, T.P., Almeida, J.P.A., Baratella, R., Fumagalli, M., Guizzardi, G. (2024). Ontology-based Security Modeling in ArchiMate. Software and Systems Modeling. https://doi.org/10.1007/s10270-024-01149-1

8.1 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. (Lankhorst, 2017) 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. (ISO, 2018) The TOGAF Series Guide to "Integrating Risk and Security within a TOGAF Enterprise Architecture" (The Open Group, 2019) 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 (The Open Group, 2019). 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 (Band et al., 2019), an attempt to introduce risk and security concepts into the ArchiMate language—the Open Group's conceptual modeling language for Enterprise Architecture. (The Open Group, 2023)

Though the RSO is based on risk and security frameworks (COSO, ISO, TOGAF, and SABES) (Band et al., 2019), it has already been shown to have some limitations concerning its conceptualization of risk concepts (Sales, T. et al., 2018), 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) (Guizzardi, 2005) and the *Common Ontology of Value and Risk* (COVER) (Sales et al, 2018), 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 (Sales, T. et al., 2018). 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 (Sales, T. et al., 2018).

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, to produce an improved artifact?

Here, by employing a similar methodology of ontological analysis (tracing back to Rosemann, Green, and Indulska, 2004; Guizzardi, 2005), 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 (Oliveira et al., 2022b). Our analysis is grounded in the *Reference Ontology for Security Engineering* (ROSE) (Oliveira et al., 2022a), 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 (Sales, T. et al., 2018).

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 can describe risk treatment options, according to ISO 31000 (ISO, 2018). 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 chapter is structured as follows:

- In Section 8.2, we provide brief methodological considerations;
- In Section 8.3, 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 8.4, we proceed with an ontological analysis of the security elements of RSO by showing its semantic shortcomings, according to the methodology described in Chapter 3;
- In Section 8.5, 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 8.6, 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 8.7, we show that our proposal can represent risk treatment options of ISO 31000;
- In Section 8.8, we illustrate our proposal through an application involving a real-world security incident and the enterprise's reaction to it;
- We conclude with an extended discussion on related work in Section 8.9 and final remarks in Section 8.10.

8.2 Methodological considerations

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 we proposed an ontology of prevention in Chapter 4 as a general foundation for our ontology of security in Chapter 5. This theory of prevention implies the existence of multiple patterns explaining why certain types of events are prevented. We will explore the implications of these patterns for security modeling in ArchiMate.

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. (Guizzardi, 2005) 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" (Rosemann, Green, and Indulska, 2004), which is, in our case, ROSE (Oliveira et al., 2022a) concerning the security domain. Ideally, according to Rosemann *et al.* (Rosemann, Green, and Indulska, 2004), 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 (Rosemann, Green, and Indulska, 2004). With the support of this framework, summarized in Figure 1.3, we identify shortcomings concerning the security modeling capability of the RSO in Section 8.4.

Our general approach is aligned with "Design Science Research" (Hevner et al., 2004; Dresch, Lacerda, and Antunes, 2015) in that there is a focus on an artifact and its cycles of (re)design and evaluation. The artifact is justified through its *relevance* to a certain context of the 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 (ISO, 2018).

8.3 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 (Band et al., 2019)–the target of our analysis. A well-founded redesign regarding risk elements of RSO has been proposed (Sales, T. et al., 2018) by our previous work. As security modeling requires risk modeling, this is our natural starting point.

8.3.1 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 (Band et al., 2019), 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 (Band et al., 2019).

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.

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" (Band et al., 2019). 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 (Band et al., 2019).

The notion of Security Principle is less developed in the RSO white paper (Band et al., 2019). 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 (The Open Group, 2023). 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 (The Open Group, 2023). 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 (Band et al., 2019).

Figure 8.1 summarizes how RSO proposes to represent risk and security elements

in ArchiMate (Band et al., 2019). 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.

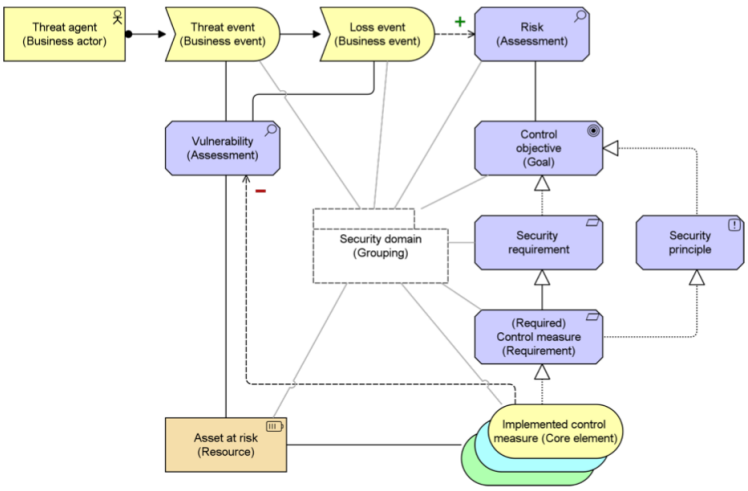

Figure 8.1: Mapping of Risk and Security Elements to the ArchiMate language Band et al., 2019

To exemplify how the RSO can be used, we present two examples extracted from Band et al., 2019, 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 8.2 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.

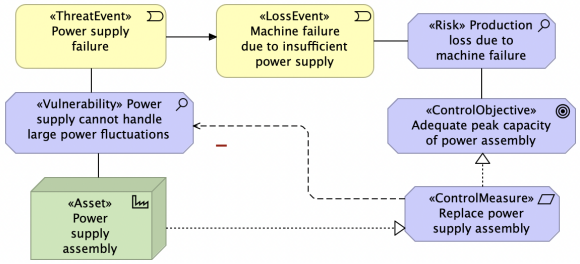

Figure 8.2: Example from the case of the Coldhard Steel company Band et al., 2019

The second example (Figure 8.3) 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.

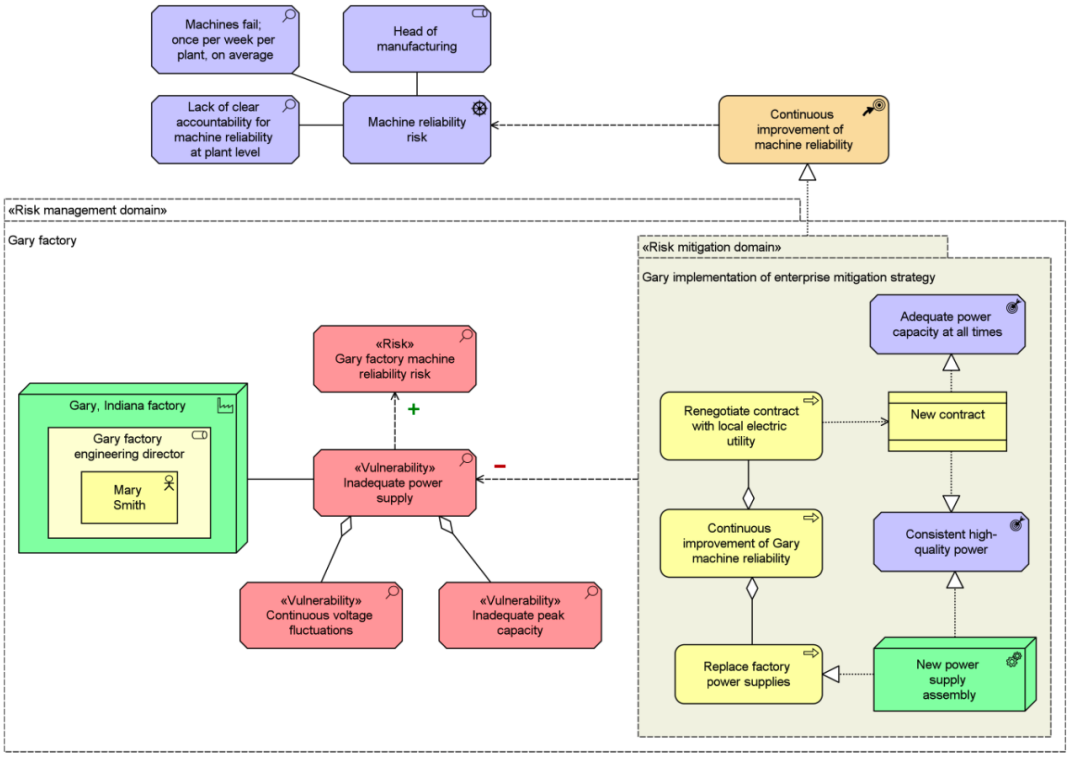

Figure 8.3: Mitigation of Machine Failure Risk at Coldhard Steel Gary Factory Band et al., 2019

Table 8.1 lists the risk and security elements according to ArchiMate elements they specialize, including their definitions from Band et al., 2019.

8.3.2 Ontology-based risk modeling in ArchiMate

Sales, T. et al., 2018 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 8.4 shows the proposal of Sales, T. et al., 2018 for evolving the RSO, while Table 8.2 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 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).

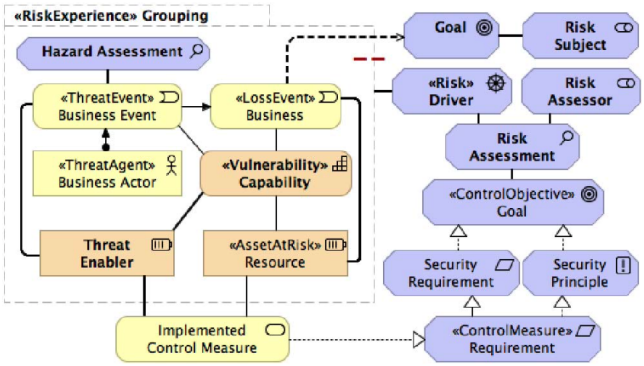

Figure 8.4: Proposal of Sales, T. et al., 2018 for evolving the Risk and Security Overlay

8.4 Ontological analysis

Taking ROSE as a reference, our ontological analysis relies on concepts described in Chapter 1.4, namely: (a) ontological incompleteness (or construct deficit); (b) construct overload; (c) construct redundancy; (d) construct excess. Then, in Section 8.5, this analysis supports the redesign of the RSO of ArchiMate by the introduction or elimination of elements.

8.4.1 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 be placed, since in Figure 8.1 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 by Band et al., 2019 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. 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 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.*

8.4.2 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 8.3. 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 8.2), suggesting confusion between the Object at Risk and elements of the pattern of Security Mechanism. We term this issue *Limitation L2.*

8.4.3 Lack of distinction between baseline and target architectures

The implementation and migration concepts of ArchiMate are used to describe how an architecture will be realized over time through changes (Lankhorst, 2017), 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, 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 8.4) connects Implemented Control Measure to Threat Enabler and Asset at Risk, 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.*

8.4.4 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 8.4. 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

[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.

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*.

8.4.5 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 (Sales, T. et al., 2018). 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*.

8.4.6 Interdependence relation among risk capabilities

As shown by ROSE, in its risk aspects (Figure 5.2), 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 8.3 summarizes the ontological limitations.

8.5 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.

8.5.1 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 (Sales et al, 2018). 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 by Sales, T. et al., 2019, we can say that a prevention event (type) decreases the likelihood of the occurrence of events of a certain type (Figure 8.5). 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 8.6. 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.

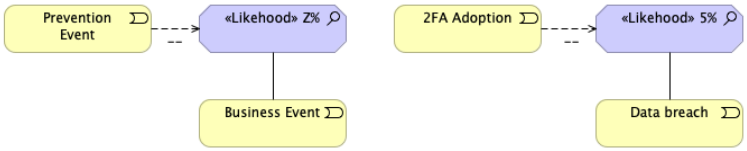

FIGURE 8.5: A representation of a prevention event that decreases the likelihood of the occurrence of events of a certain type

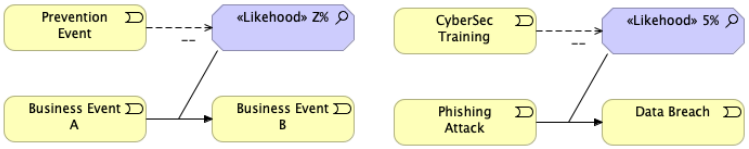

FIGURE 8.6: A representation of a prevention event that decreases the likelihood of an event causing events of a certain type

Representing prevention the way expressed by Figure 8.5 and Figure 8.6 – 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 Sales, T. et al., 2019 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 8.7. 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 Chapter 4. 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 8.8. 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 – 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 8.9. 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.

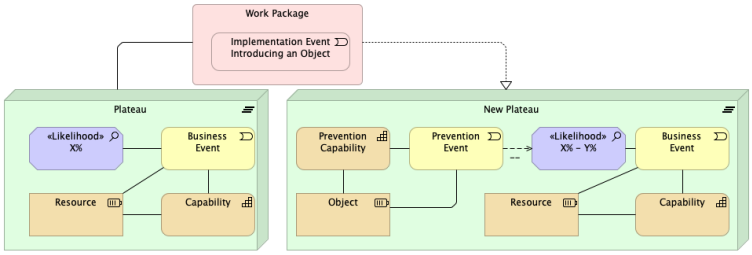

FIGURE 8.7: A representation of prevention in ArchiMate that includes objects, their capabilities, and temporal changes

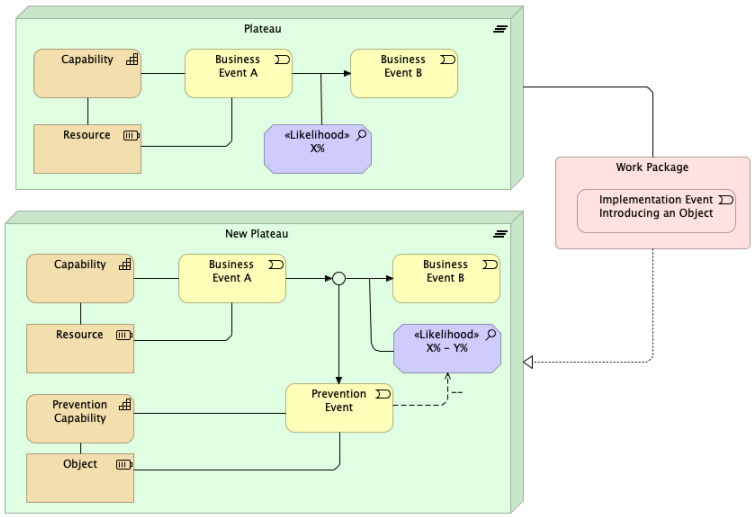

FIGURE 8.8: A representation of prevention in ArchiMate that includes objects, their capabilities, and temporal changes, throughout a causal chain

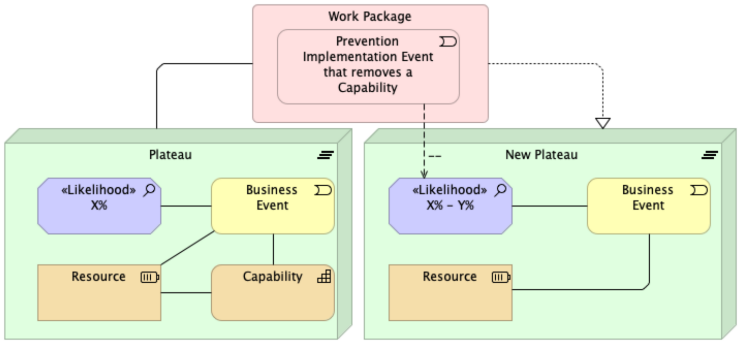

FIGURE 8.9: A representation of prevention in ArchiMate where an implementation event removes a capability, therefore decreasing the likelihood of the associated events in the new plateau

8.5.2 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 by Sales, T. et al., 2018, 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 (Sales, T. et al., 2019), 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 events like these. This is exactly what a Security Mechanism does systematically, following the ROSE and the theory of prevention (Oliveira et al., 2022a; Baratella et al., 2022). 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 show a different configuration of the interdependent risk entities, 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 Sales, T. et al., 2019 since security is a matter of specific creation of value through the prevention of risks. Figure 8.10 shows our proposal to evolve the security aspects of the RSO, highlighting in bold the constructs and relations we propose. Table 8.4 shows our proposal of the representation of security concepts in ArchiMate based on ROSE.

Figure 8.11 exemplifies our proposal using the same example from the RSO involving a Loss Event of production loss caused by a Threat Event of power

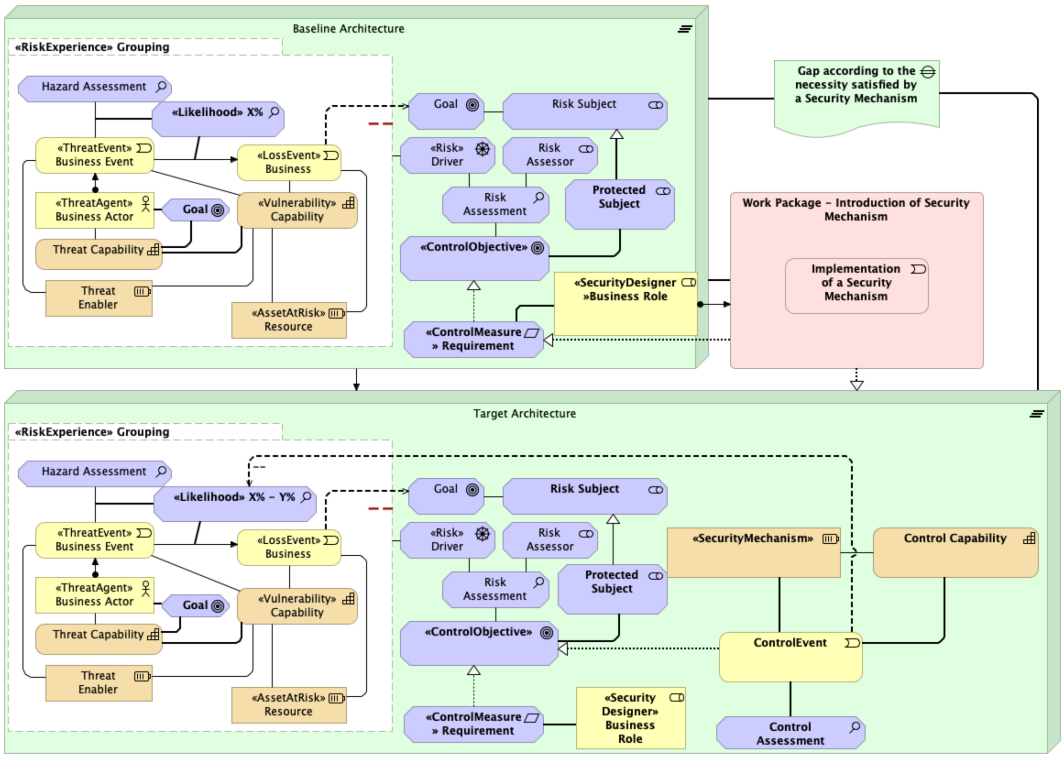

FIGURE 8.10: Proposal for evolving the security aspects of the Risk and Security Overlay of ArchiMate

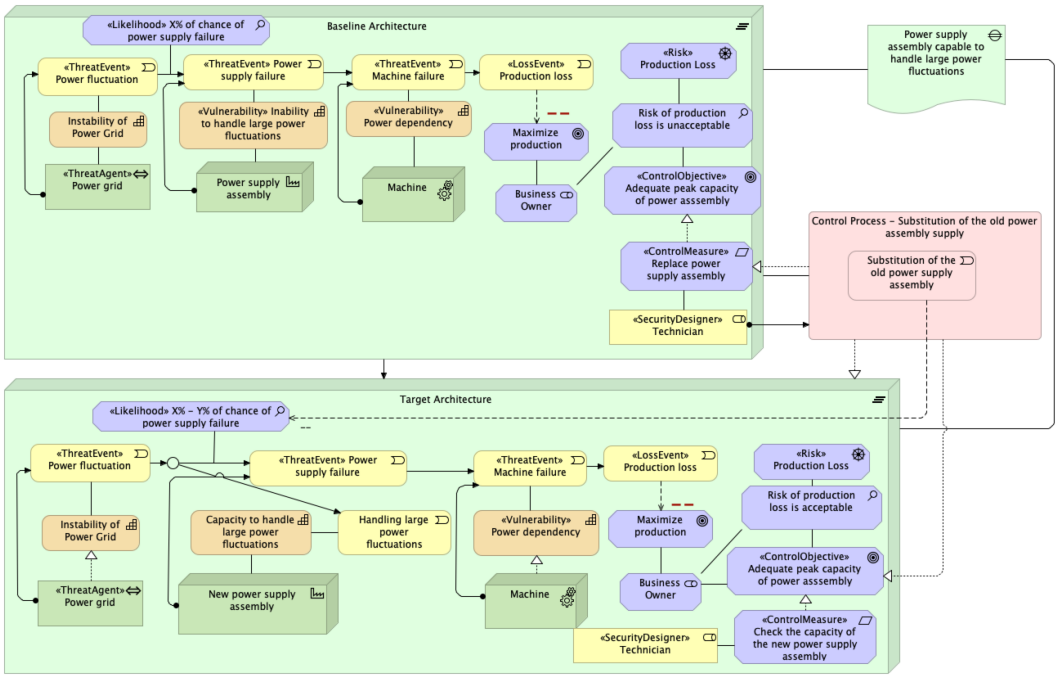

FIGURE 8.11: Example of modeling the introduction of a security mechanism

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 is able to handle large power fluctuations, so the likelihood of power supply failure is lower; the original power supply assembly was totally removed from the scene, which means its vulnerability was also removed from the scene. This is one of the ways of prevention (Baratella et al., 2022). 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 (Reason, 1990).

We provide the resulting files with related information in a public repository[1]. Our proposal is well-documented on a dedicated website.[2]

8.6 Ontology-based security modeling patterns

In Chapter 4, we described general patterns of prevention according to an ontological theory grounded in UFO Baratella et al., 2022. In Section 8.5.1, we proposed ways of introducing this ontology of prevention in ArchiMate, particularly displayed in Figure 8.7, Figure 8.8, and Figure 8.9. In Section 8.5.2, we proposed a redesign of ArchiMate according to ROSE (Figure 8.10), which assumes the ontology of prevention. Now, we will develop those patterns of prevention implied by this redesigned artifact.

Gangemi and Presutti, 2009 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 the Threat Agent is a person, a potential criminal, not a purposeless object.

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

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 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 (Sales, T. et al., 2018) simply aggregates partner dispositions concerning 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 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.

8.6.1 Removing a threat capability

Figure 8.12 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 8.9. 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.

8.6.2 Removing a threat agent

Figure 8.13 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 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, and 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%.

8.6.3 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

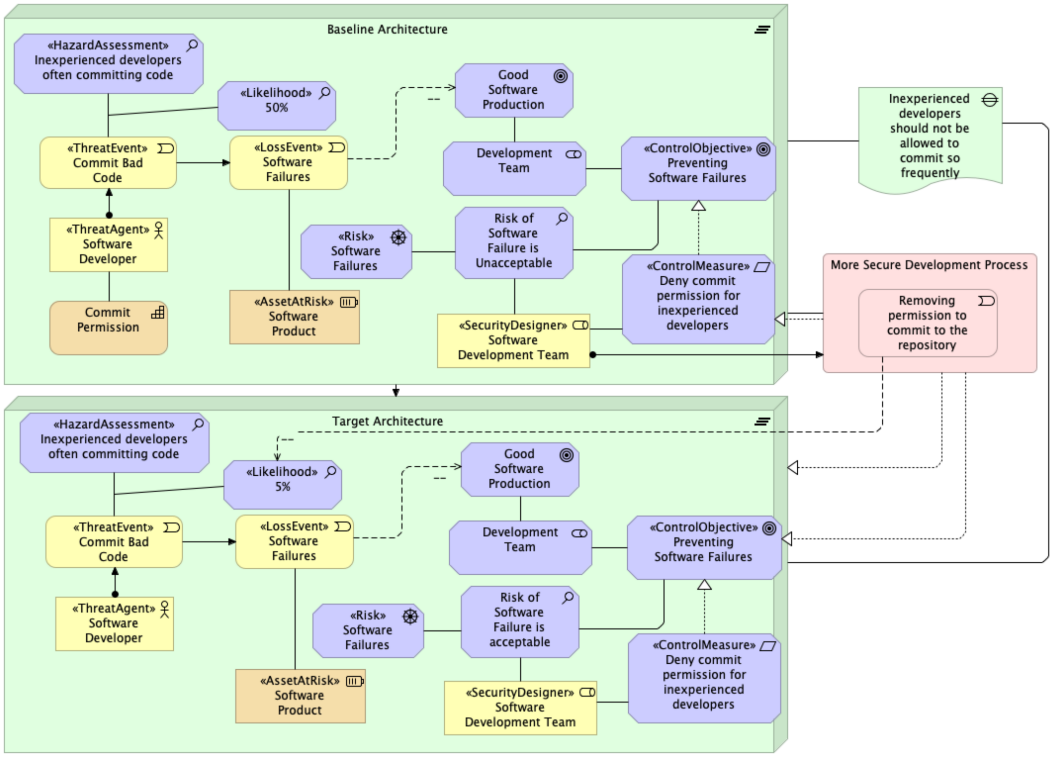

FIGURE 8.12: Example of removal of the THREAT CAPABILITY from the target architecture by a CONTROL EVENT that is a prevention implementation event. Therefore, the associated THREAT EVENT cannot occur

kind, among others. Figure 8.14 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.

8.6.4 Removing a vulnerability

Figure 8.15 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 8.15 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 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.

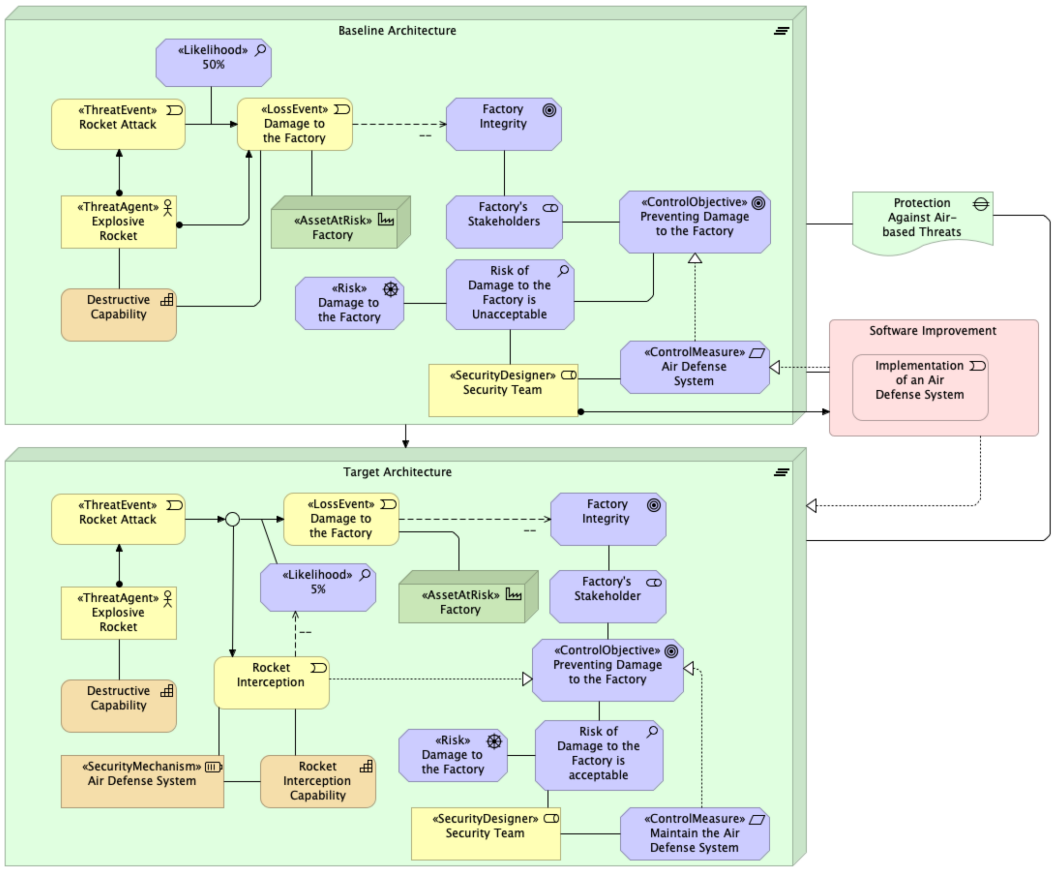

FIGURE 8.13: Example of removal of the THREAT AGENT from the scene by the introduction of a SECURITY MECHANISM that is capable of destroying the agent and, therefore, its capabilities, so preventing the associated events

8.6.5 Removing an asset at risk

Figure 8.16 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 very SECURITY MECHANISM may hold its own VULNERABILITIES.

8.7 Evaluation: representing risk treatment options of ISO 31000

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).

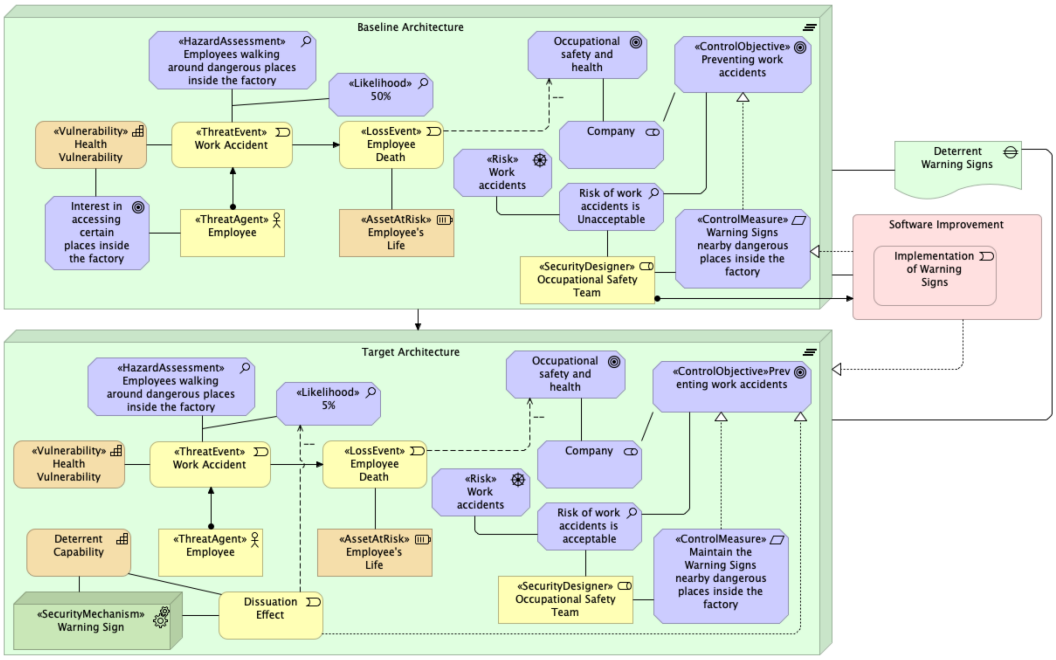

FIGURE 8.14: Example of removal of the THREAT AGENT's GOAL from the target architecture thanks to the introduction of a SECURITY MECHANISM that has a deterrent capability

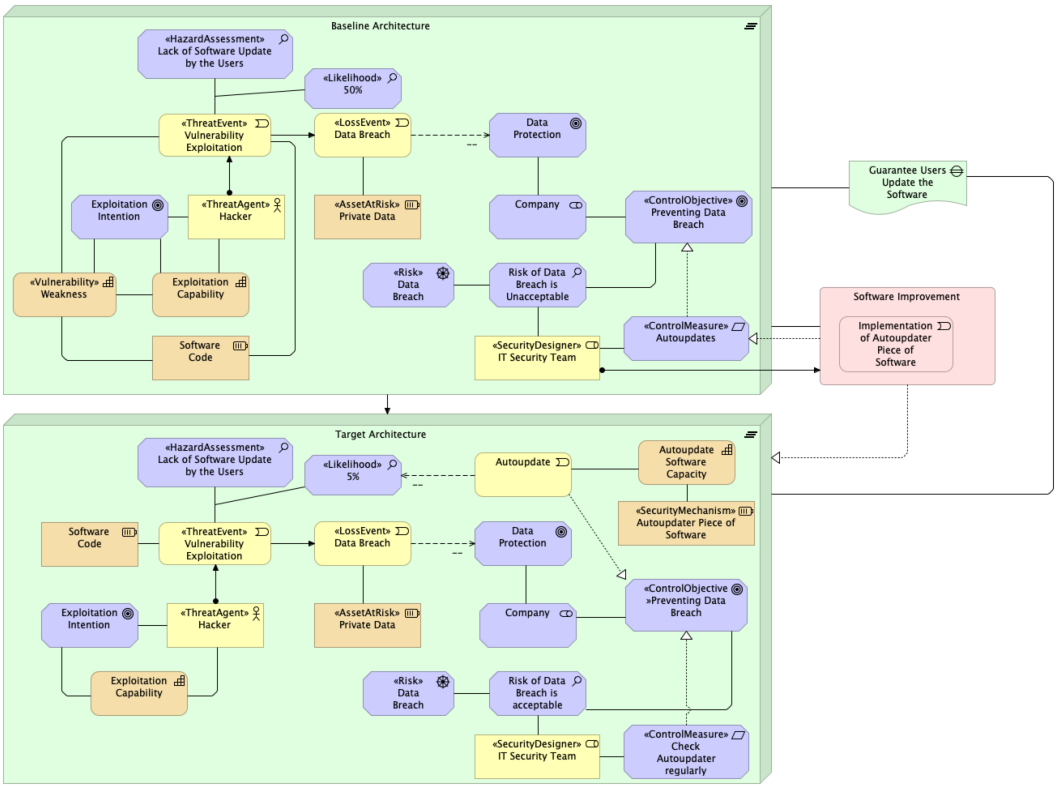

FIGURE 8.15: Hardening, removing vulnerability. Example of removal of the THREAT AGENT from the scene by the introduction of a SECURITY MECHANISM that is capable of destroying the agent and, therefore, its capabilities, so preventing the associated events

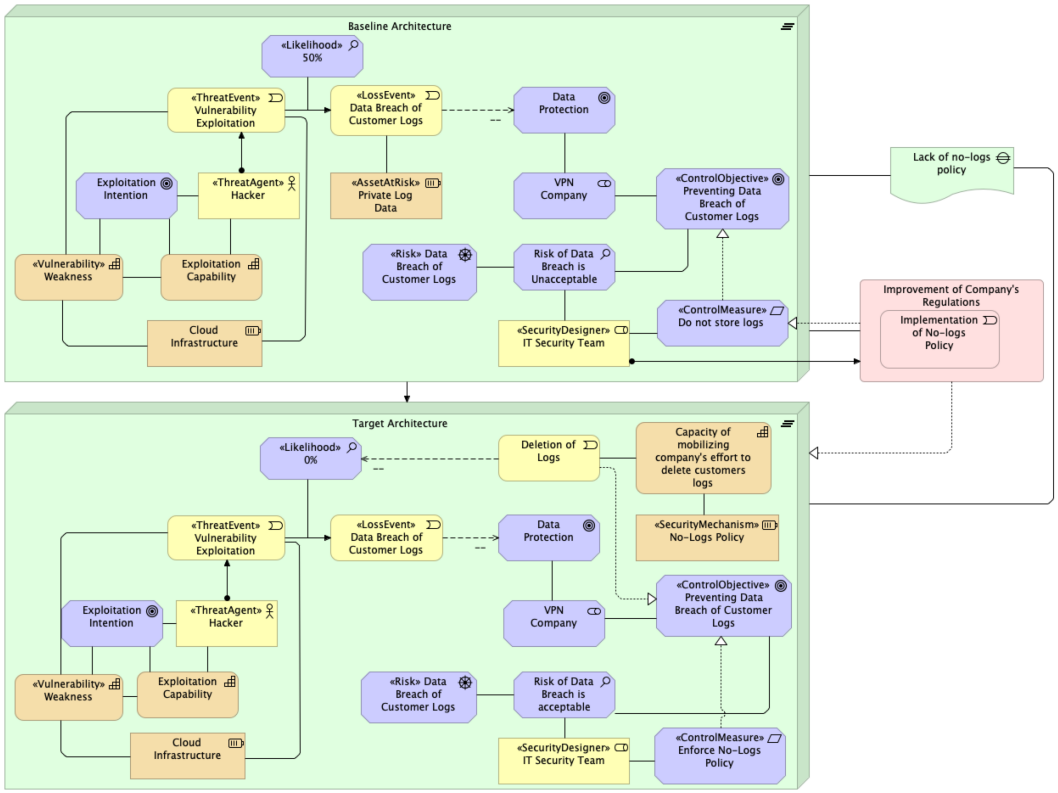

FIGURE 8.16: Example of removal of the ASSET AT RISK from the scene by the introduction of a new regulation as a SECURITY MECHANISM. Therefore, the associated LOSS EVENT cannot occur

As stated in Chapter 5, the *Reference Ontology for Security Engineering* (ROSE) takes into account the risk treatment options defined by ISO 31000 (Oliveira et al., 2022a). Our overlay, based on ROSE, inherits similar features, being able to represent those options in ArchiMate. The list below shows each option (ISO, 2018) 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; this case may correspond to the pattern of removing an ASSET AT RISK explained in Section 8.6.5;

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 Chapter 5, 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 8.6.1), removing THREAT AGENTS (Section 8.6.2), removing VULNERABILITIES (Section 8.6.4), or even removing ancillary entities with partner dispositions;

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 8.5.1; 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.

8.8 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[3]. Our 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 8.17 and Figure 8.18.

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 to steal those Asset 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

[3]The main sources describing what happened and LastPass's security reactions are the following blog posts:
(1) https://blog.lastpass.com/2023/03/security-incident-update-recommended-actions/,
(2) https://support.lastpass.com/help/incident-1-additional-details-of-the-attack.

home networks by removing that VULNERABILITY, which corresponds to our pattern of Section 8.6.4. 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 the threat actor. 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.

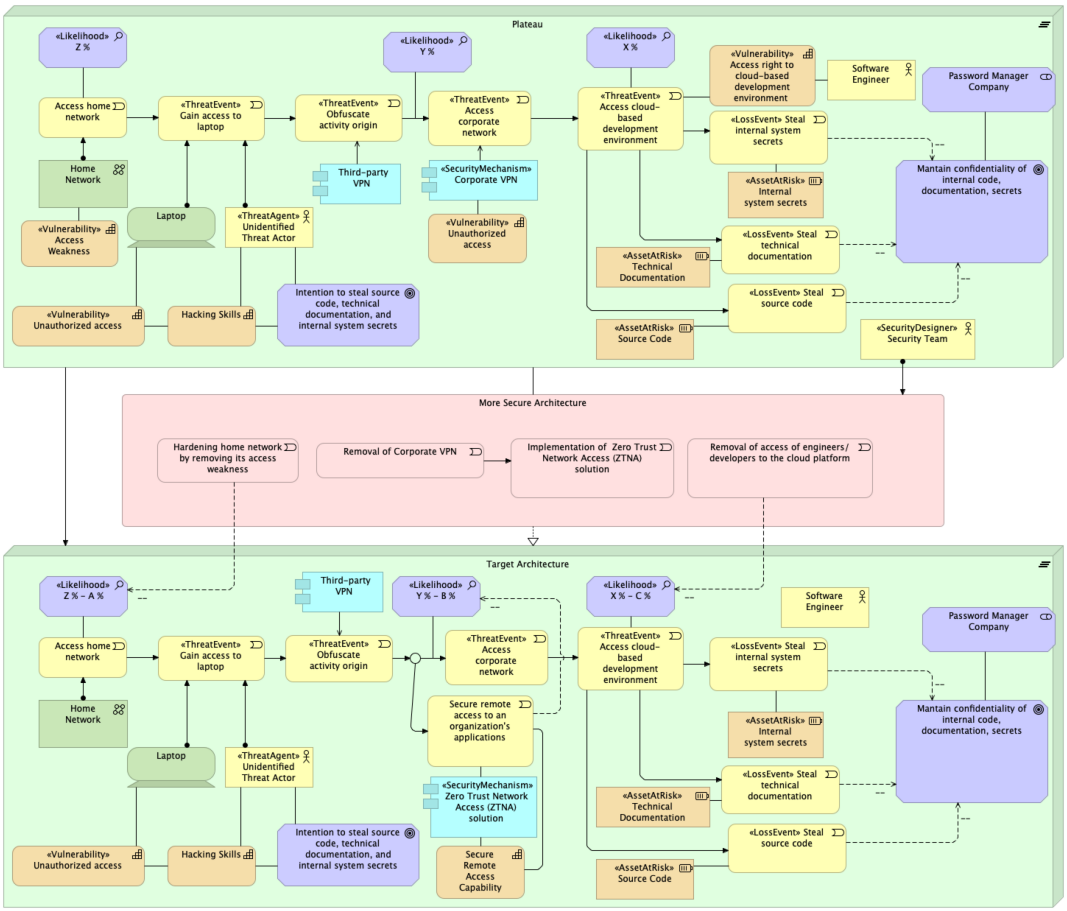

FIGURE 8.17: A representation of incidents and security reactions of the type of LastPass's first incident in August 2022

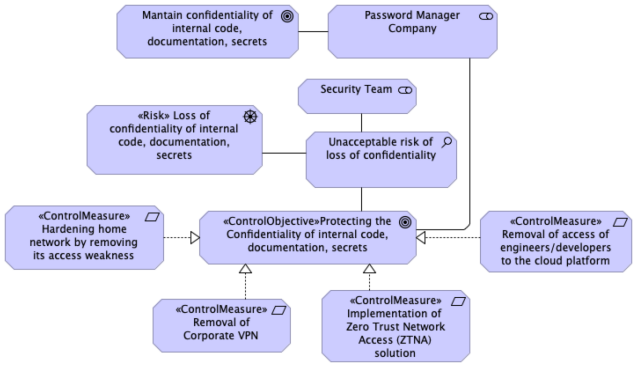

FIGURE 8.18: Motivation layer containing security elements regarding LastPass's first incident in August 2022

8.9 Related work

Fernandes, Ramalho, and Silva, 2022 analyze 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.

Ellerm and Morales-Trujillo, 2020 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 Ellerm and Morales-Trujillo, 2020.

Grov, Mancini, and Mestl, 2019 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 8.3.1, was presented in a research paper (Jonkers and Quartel, 2016), as shown in Figure 8.19. 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 research conducted by Mayer and his collaborators (Mayer

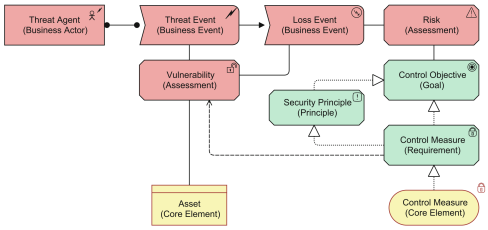

FIGURE 8.19: Risk and security concepts as specializations of ArchiMate concepts, extracted from Jonkers and Quartel, 2016

and Feltus, 2017) 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, Grandry, Feltus, and Dubois, 2013 make use of the works done by Mayer and Feltus, 2017 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 Bosch, 2014. 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.

Almeida et al., 2019 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 (Bradley, 2021) 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, Hoogenboom, 2019 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 implements cybersecurity vocabulary through ArchiMate standard elements.

Grov, Mancini, and Mestl, 2019 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 by Tovstukha, 2017 that proposes an alignment between Mal-activity diagrams and ArchiMate in the context of Information System Security Risk Management, and another one by Artem, 2018 that compares the Secure Socio-Technical Systems models and ArchiMate's RSO.

8.10 Final considerations

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) (Oliveira et al., 2022a), 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. (Sales, T. et al., 2018; Sales, T. et al., 2019) 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.

Table 8.1: 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 which 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.

Table 8.2: Representation of risk concepts in ArchiMate based on COVER Sales, T. et al., 2018

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

Table 8.3: 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.

TABLE 8.4: 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»

Part IV

CONCLUSION

Chapter 9

Final considerations

In each of the previous chapters, we have already presented final considerations including discussions of the benefits of the contributions of that chapter, as well as its main advantages when compared to related work in the literature, and limitations. For this reason, in this Chapter, we will review our contributions in light of our research objectives, argue for the relevance of this work for researchers and practitioners, systematize limitations, and consider future perspectives for our research work.

9.1 Research contributions

Each publication reports several scientific contributions. We have given special attention to the transparency, usability, and accessibility of our work, so we are maintaining corresponding public repositories on GitHub (see Appendix A).

Objective 1: *By identifying the state of the art in security ontologies, we intend to motivate and support our proposed security ontology.*

In Chapter 2, we produced a systematic mapping study of the literature concerning security core ontologies. By doing so, we satisfy our *Objective 1*. Among our findings, we discovered that (a) most of the existing ontologies of security are not publicly findable or accessible, (b) foundational ontologies are under-explored in the security domain, (c) there seems to be no common ontology of security, and (d) that the most common concepts in the studied security core ontologies are: *Vulnerability*, *Risk*, *Asset*, *Attacker*, *Threat*, *Control*, *Countermeasure*, *Stakeholder*, *Attack*, *Consequence*. This work was not only useful for motivating the need for a reference domain ontology for security but also for gathering elements for the construction of this artifact.

Objective 2: *By proposing a general ontology of prevention based on UFO, we intend to ground our proposed security ontology.*

In Chapter 4, we proposed a UFO-based model for the prevention phenomenon, which is a key notion in areas such as risk management, bioinformatics, health sciences, and legal informatics, among others. Prevention is understood as the impossibility of manifestation of dispositions in a given setting (the required situation type for the activation of certain dispositions). To account for the specification of this setting, we propose a novel notion of *Mutual Activation Partnership* among dispositions. Based on our concept of prevention, we explain the meaning of breaking a causal chain through antidotes, two different types of indirect prevention, a notion of countermeasure mechanism as a designed intervention to yield prevention, and multiple patterns through which prevention can occur.

Objective 3: *By proposing a reference domain ontology of security from a risk treatment perspective (termed ROSE), according to ISO 31000, we address the main research problem of understanding and modeling security.*

In Chapter 5, based on our ontological theory of prevention reported in Chapter 4, we proposed a well-founded *Reference Ontology for Security Engineering* (ROSE) accounting for risk treatment options of ISO 31000. ROSE is presented as an ontological analysis of the notion of Security Mechanism. As a UFO-based ontology, ROSE interoperates well with other UFO-based related ontologies, particularly with the Common Ontology of Value and Risk. ROSE allows us to uncover ambiguities in risk treatment options of ISO 31000 and represent each option.

Objective 4: *By proposing a reference domain ontology of phishing attacks specializing ROSE's concepts, we show how ROSE can be tuned for more specific domains.*

In Chapter 6, we presented a reference domain ontology for phishing attacks (PHATO). PHATO exemplifies how ROSE can be properly specialized in a more specific domain. PHATO accounts for numerous ways through which phishing attacks can happen and succeed, and it supports the analysis of countermeasures for phishing attacks.

Objective 5: *Based on ROSE, proceeding with an ontological analysis of the D3FEND cybersecurity model plus recommendations for improvements. This shows ROSE can support the evaluation of information artifacts within the security domain.*

As a practical application of ROSE, in Chapter 7 we reported an in-depth ontological analysis of the D3FEND cybersecurity model, a well-known industry knowledge graph. We showed several general and domain-specific issues within D3FEND, particularly a systematic lack of important ontological constraints. We conjectured that, without the aid of the systematic taxonomy of a foundational ontology, ontology engineers usually drop constraints to avoid inconsistencies as the ontology gets bigger, consequently admitting unintended instances. We gave several recommendations for improving the D3FEND model. This analysis adds other evidence in favor of well-founded models.

Objective 6: *Proceeding with an ontological analysis and redesign of security-related elements of the ArchiMate's Risk and Security Overlay. This shows how ROSE can support an ontologically sound redesign of domain-specific languages related to the security domain.*

Based on ROSE, another practical application was reported in Chapter 8: an in-depth ontological analysis of security elements within the Risk and Security Overlay (RSO) of ArchiMate. Our analysis identifies six limitations concerning the security modeling capabilities of the RSO. We presented a general representation of the ontological theory of prevention through ArchiMate language. We proposed a well-founded redesign of the security elements of ArchiMate. We showed ontology-based security modeling patterns within our redesign proposal. Similarly to ROSE, our redesign proposal can represent risk treatment options of ISO 31000. We made a dedicated documentation website available to support updates and usability of our proposal, including risk and security elements[1].

[1] https://unibz-core.github.io/security-archimate/.

9.2 Relevance for researchers and practitioners

For *enterprise modeling researchers*, our ontologies lay a well-founded conceptual foundation that can be leveraged in the (re)engineering of tools and modeling languages to assist organizations in representing and reasoning with security engineering. As demonstrated in the ArchiMate applications in Chapter 8, the ontologies can be directly reused to identify semantic limitations in existing approaches and improve them.

For *ontology engineers*, a similar point has been stressed in Chapter 7: building knowledge graphs without ontological foundations is highly error-prone. Well-founded domain ontologies such as COVER and ROSE support better-quality knowledge graph development.

For *enterprise architects*, we have developed a coherent ArchiMate extension for representing security elements way beyond the simple notion of decreasing a vulnerability from the original RSO. Our effort will continue to document our proposal to make it more easily usable by practitioners.

We believe that the clarification of security-related notions (for instance, Security Mechanism) can support the development of information systems for defense purposes, public health management, and national security, to name a few.

9.3 Limitations

There are two major limitations of this research work: (1) one regarding the theory of prevention, which does not account for the interference phenomenon; and (2) another regarding risk management conceptualization.

If an effect is interfered with, it still occurs but differently: maybe not as strongly as it could have occurred, or not quite in the same way, or perhaps it becomes delayed. Interference typically happens because of some extra factor being present that disposes away from the type of event identified as the effect. (Mumford and Anjum, 2011) Our ontological theory of prevention, as it is, does not support this sort of phenomenon. This is a limitation considering risk treatment options often involve mitigation measures, which is a form of interference. We intend to address this point in the future along with a full first-order logic formalization of the theory.

Furthermore, as shown in Figure 1.2, the risk management process involves more than risk assessment and risk treatment. For example, elements of communication, consultation, monitoring, review, recording, and reporting have been left untouched by our research work. Therefore, by no means, we complete an ontological analysis of risk management as a whole. In particular, the monitoring aspect of risk management is intertwined with the security one because some security mechanisms can only manifest their control capabilities if monitoring mechanisms are triggered earlier.

9.4 Future perspectives

We hope this research is only the beginning of much more that we want to do. We have already discussed future work in each chapter. Now, we would like to mention specifically what we are going to do next and their relevance.

A Formal Ontology of Prevention

A full first-order logic formalization of our ontological theory of prevention, organized as a consistent module of UFO, including an account for the interference phenomenon. This work goes hand in hand with a new ongoing formalization of UFO-A,

UFO-B-, UFO-C. Our team has been working on how to properly modularize UFO micro-theories and how to support automatic test-driven ontology engineering with theorem provers. We have been experimenting with Lean theorem prover and program language for this purpose Moura and Ullrich, 2021. The consequences of having an ontology module of prevention in this fashion are, among others, the deepening understanding of prevention and interference, rigorous validation, the possibility of computational simulations, and portability to knowledge representation languages, such as OWL, particularly as a gUFO module.

Object-Event Simulation for Risk and Security Modeling

Object-event simulation is a new general Discrete Event Simulation paradigm based on two important ontological categories: objects and events. (Wagner, 2018; Guizzardi and Wagner, 2010) We plan to leverage this approach to support computational simulations for risk and security modeling. In particular, we plan to empower our ArchiMate proposal for Enterprise Risk Management with this approach, so that ArchiMate models can be simulated. The success of this work will deliver a toolbox combining well-founded enterprise risk and security modeling with automated support for analyzing and reasoning about these models. These two aspects have been lacking in enterprise architecture, according to Grov, Mancini, and Mestl, 2019. In Chapter 8, we addressed the first gap by building well-founded models.

A Phishing Attack Ontology

We have been working on improving the phishing attack ontology (PHATO), proposed in 6. For example, besides reviewing and adding concepts, we have been validating PHATO by different means: (a) more extensive literature study on phishing attacks; (b) multiple validation sessions with phishing experts; (c) alignment with notions presented in phishing datasets. Furthermore, once complete, this improved PHATO will allow us to make statistical simulations of phishing attacks, testing scenarios with and without security mechanisms, thanks to the object-event simulation approach.

A Unified Ontology of Value, Risk, and Security

There are several themes to be explored through the combination of value risk, and security. A unified ontology for COVER and ROSE can provide new light for understanding notions such as incident and resilience. In particular, we plan to explore the insight that incidents represent the actual prevention of value, whereas risks represent the possible prevention of value. By this line of reasoning, security is phrased as a case of double prevention. Besides further validation of COVER and ROSE, this work can leverage the theory of prevention to the maximum by ontologically unpacking the notions of risk, incident, and security around the ontologies of value and prevention.

A Well-Founded Cybersecurity Model for Threat Intelligence

A well-founded version of D3FEND is a natural next step for the research work reported in Chapter 7. However, we plan to go further by combining D3FEND and ATT&CK. The resulting artifact will be used by a recent natural language processing framework (van Ede, 2023) that automatically extracts actionable threat intelligence from threat intelligence reports and classifies it into the artifact taxonomy. By doing so, we expect to advance the state of the art in the identification and explanation of (cyber)security events.

An Ontological Analysis of FMEA

Failure mode and effects analysis (FMEA) was introduced in 1949 by the US Armed Forces for performing a failure mode effect and criticality analysis. The goal was to classify failures "according to their impact on mission success and personnel/equipment safety." It was later adopted in the Apollo space program to mitigate risk due to small sample sizes. In the late 1970s, the Ford Motor Company introduced FMEA to the automotive industry for safety and regulatory consideration after the Pinto affair. In the 1980s, the automotive industry began implementing FMEA. Although developed by the military, the FMEA method is now extensively used in a variety of industries including semiconductor processing, food service, plastics, software, aeronautics, automotive, healthcare, and many others. (Carlson, 2014)

According to Spreafico, Russo, and Rizzi, 2017, although there are many attempts to create more rigorous definitions of FMEA, there is not yet a proper ontology that could transversally solve many different classes of problems. Because of that, an ontological analysis of FMEA based on ROSE and COVER will be not only scientifically important but also practically impactful for the industry.

All in all, it's been worth it. 😉

Appendix A

Project Repositories

Just like software, research changes and evolves toward refinement, revision, or novel applications. Hopefully, the outcomes of this thesis will develop into further branches of theoretical and practical applications. The online locations below will help to keep track of this evolution.

Ontology of Prevention

- PURL: https://purl.org/prevention-ontology.

Reference Ontology for Security Engineering (ROSE)

- PURL: https://purl.org/security-ontology.

Ontological Analysis of D3FEND Cybersecurity Model

- PURL: https://purl.org/d3fend-analysis.

Phishing Attack Ontology (PHATO)

- PURL: https://purl.org/phishing-ontology.

Ontology-based Security Modeling in ArchiMate

- DOI: https://doi.org/10.5281/zenodo.10005209.
- Website: https://unibz-core.github.io/security-archimate/.

Appendix B

Ontology Vocabulary

This Appendix contains a list of concepts that the research work of this Ph.D. thesis has introduced. Therefore, only novel notions are listed here, not, for example, concepts that are already present in the Unified Foundational Ontology (UFO) or the Common Ontology of Value and Risk (COVER). Second-order types (for instance, Intention Type) are also excluded here.

A

Asset: In the context of phishing attacks, an Asset (*roleMixin*, *object*) is a relational object that the Scammer intends to obtain through a Phishing Attack. From the viewpoint of a Value Subject, an Asset is Value Object. From the viewpoint of a Risk Subject, an Asset is an Object At Risk. Examples: Password, login, and bank account information.

Attack: In the security domain, an Attack (*event*) is a Threat Event that requires the Intention of an Agent (Attacker) for its manifestation. Examples: robbery, terrorist attacks, and aggression.

Attacker: In the security domain, an Attacker (*roleMixin*, *object*) is a Threat Object that is an Agent whose Intention is necessary for an Attack. Examples: Robber, scammers, and terrorists.

Asset Catch: In the context of phishing attacks, an Asset Catch (*event*) is an event wherein a Hook and an Asset participate. It is triggered by a Vulnerability Condition (*situation*). It occurs when a Phishing Attack succeeds. Example: When people give their email address and password to a fake website (Hook) owned a Scammer.

B

...

C

Control Capability: In the security domain, a Control Capability (*intrinsic mode*, *roleMixin*) is a disposition whose manifestation is a Protection Event (a Control Chain Event or a Control Event). A Security Mechanism aggregates Control Capabilities. However, other objects may also have one or more Control Capabilities. Examples: Antivirus software has Control Capabilities to scan and eliminate malware. Air defense systems have Control Capabilities to intercept certain air threats. Warning signs have Control Capabilities to dissuade people from doing certain things.

Control Chain Event: In the security domain, a Control Chain Event (*event*) is an event that can *cause* a Control Event, that is, a Control Chain Event indirectly *prevents* Threat Events or Loss Events of a certain type. Control Chain Events are manifestations of Control Capabilities. It is the security counterpart of a Trigger Event, in the value domain, and Threat Event, in the risk domain. Examples: The search procedure of anti-virus software. Any middle step event that causes a Control Event.

Control Event: In the security domain, a Control Event (*event*) is an Impact Event (Value Event) that directly *prevents* Threat Events or Loss Events of a certain type. Control Events are manifestations of Control Capabilities. It is the security counterpart of a Impact Event, in the value domain, and Loss Event, in the risk domain. Control Events *impact* positively Security Designer's and Protected Subject's Intentions. Examples: To realize the goal of preventing software failure, removing permission to commit to the repository can be an adequate Control Event. Rocket interception can be a Control Event as a manifestation of an air defense system's Control Capabilities.

Controlled Situation: In the security domain, a Controlled Situation (*situation*) is brought about by a Control Event. It is a situation whose type is *incompatible with* the situations of the type that triggers Risk Events of certain types. In other words, once a Control Event occurs, it brings about a "controlled" or "secure" state of affairs concerning certain types of Risk Events. Examples: Caged monitored animals in a zoo. Knowledgeable employees after phishing awareness training. Virus-free computer after malware removal by anti-virus software.

Countermeasure to: In the context of modeling dynamical aspects of reality, given a disposition d whose manifestations are of type E_T, countermeasures are designed interventions that endow a setting containing d with other dispositions $\{d_1, ..., d_n\}$, whose manifestations prevent any instance of E_T. More specifically, *Countermeasure Mechanisms* are designed such that: they contain dispositions of type D_T, and given the situations of type S_T that would trigger events that would (directly or indirectly) cause instances of E_T, the instances of S_T instead activate the instances D_T whose associated event type prevent E_T. Example: a circuit break contains a disposition to close the circuit in a situation where there is a current above a certain threshold. The manifestation of that disposition of the circuit breaker thus prevents the event of an overcurrent.

D

Designed by: In the security domain, a Security Mechanism is always *designed by* an Agent called the Security Designer to be a countermeasure to Risk Events of a certain type. The *design by* relation can be understood as a link between the creation or implementation of an Security Mechanism and the responsible Agent for that. This is useful to distinguish between the implementation of a security setting, the security setting itself, and its functioning (Protection Event).

E

Exposure: In the context of phishing attacks, exposure (*roleMixin*, *intrinsic mode*) is a disposition to be exposed, discovered by an Agent, particularly a Scammer. Its manifestation is necessary to compose a Phishing Attack.

F

...

G

Generic Intention: In the context of security, an Intention (*category*, *mode*) can be generic or specific, according to how specific the situation that satisfies it is. For example, in the aerospace domain, some goals related to the costs of the mission are generic because they can be satisfied by more funding or an assurance; even goals related to replaceable engineering parts can be satisfied by other parts of the same type. However, the completion of the mission is a specific goal that can only be satisfied by a specific situation. This distinction is important because certain Security Mechanisms only work for generic goals. For instance, a space company that transfers some of its risks to an insurance company can be protected from financial loss, but not from the losses caused by the explosion of a space shuttle. Ultimately, Generic Intention can only be impacted by a setting with generic Value Objects (money, for example), but the Specific Intention may be satisfied by a specific setting with generic Value Objects (say, the need for money under a deadline of bankruptcy).

H

Hook: In the context of phishing attacks, a Hook (*roleMixin*, *object*) commonly consists of a website or other means that imitates the appearance of a reputable agent (say, a famous company's website). The goal of the Hook is for victims to be directed to it via the Lure portion of the attack and for the victims to disclose confidential information to the site. A Hook participates in an Asset Catch event.

I

Impersonation Capability to Deceive Target: In the context of phishing attacks, an Impersonation Capability to Deceive Target (*roleMixin*, *intrinsic mode*) is a Threat Capability that is *externally dependent* on an Impersonated Reputable Agent. Its manifestation is necessary to compose a Phishing Attack.

Impersonation of Reputable Agent to Deceive Target: In the context of phishing attacks, an Impersonation of Reputable Agent to Deceive Target is an event that composes a Phishing Attack by being the manifestation of a Impersonation Capability to Deceive Target. Lures participate in this event.

Impersonated Reputable Agent: In the context of phishing attacks, an Impersonated Reputable Agent (*roleMixin*, *object*) is an Agent whom an Scammer impersonates to deceive a Target. Examples: IT famous companies. Bank brands.

Incompatible with: In the context of modeling dynamical aspects of reality, prevention of events of type E_T that are manifestations of dispositions of type D_T occurs when an event of type E'_T brings about a situation of a type S'_T that is *incompatible with* the situations required to activate instances of D_T. This incompatibility means these situations cannot be obtained concurrently. On the type-level, *incompatible*(S_T,S'_T) implies that there are no two instances of these two types that are obtained in overlapping time intervals. These situation types are *semi-saturated*,

that is, they must share some relevant dispositions or objects. For example, a situation with updated software is compatible with a situation that contains a different outdated software, even if the two situations temporally overlap, though they would be incompatible if the referred software was the same individual in both situations.

Intention to Phish: In the context of phishing attacks, an Intention to Phish (*mode*) is an Intention that is necessary for an Impersonation of Reputable Agent to Deceive Target event and, therefore, a Phishing Attack.

J

...

K

...

L

Lure: In the context of phishing attacks, a Lure (*roleMixin*, *object*) consists of a phisher spamming a large number of users with an email message or other means that appears to be from some legitimate institution that has a presence on the Internet. The message often uses a convincing story to encourage the user to follow a URL hyperlink embedded in the email to a website controlled by the phisher and to provide it with certain requested information. The social engineering aspect of phishing attacks typically makes itself known in the Lure, as the spam offers some plausible reason for the user to provide confidential information to the website that is hyperlinked by the spam.

M

Mutual Activation Partnership: In the context of modeling dynamical aspects of reality, a Situation Type associated via *activation* to a given Disposition Type must include the presence of other dispositions of a suitable kind. For example, oxygen and the ignition heat source have specific dispositions, which together with flammability produce a catch on fire event. In other words, particular events of catching on fire are not manifestations of flammability only but complex events composed of the interacting manifestations of multiple dispositions. All dispositions display this partnership aspect: the disposition of a person to swim under the water can only be manifested with the presence (and manifestation) of the dispositions that inhere in the water; the disposition of an object to roll on certain surfaces can only manifest itself with the presence (and manifestation) of the dispositions of the very surfaces (e.g., the friction of a certain kind). When one disposition is manifested, the others are also manifested, each one producing its particular manifestation, which combines to produce an effect (complex event).

To capture this dependence relation between a disposition d of a certain type D_T and other types of dispositions, we introduce here the *Mutual Activation Partner* (MAP) relation. MAP(D_T,D'_T) implies that, for instances d of D_T to be activated, they need the presence in the activating situation of an instance of D'_T so that the manifestations of d are always part of complex events that are also composed of a manifestation of instances of D'_T. As a consequence, we have that any Situation Type S_T bearing an activation relation to instances of D_T must have in its instances (particular situations) instances of all D'_T associated via MAP to D_T. MAP is a

relation of generic dependence and, hence, asymmetric and transitive, i.e., a strict partial order relation.

An additional requirement for situations of type S_T activate dispositions of the type D_T. A situation type S_T activating a particular disposition d is semi-saturated in the following way: its instances must be situations in which d and, hence, its bearer are present. This follows directly from UFO's constraints that the manifesting disposition must be present in the situations preceding and succeeding its manifestation and from the existential dependence between the disposition and its bearer. Similarly, we can define semi-saturated event types. For example, the event of *Nina Simone's singing* is still a type that can be instantiated by multiple occurrences, all of which have Nina Simone as a participant.

N

...

O

...

P

Phishing Attack: In the context of phishing attacks, a Phishing Attack (*event*) is a complex Threat Event composed by the manifestation of an Impersonation Capability to Deceive Target, Intention to Phish, and Exposure of a Phishing Enabler (for example, email address). A Lure, a Scammer, and a Phishing Enabler participate in a Phishing Attack. The goal of a Phishing Attack is to cause an Asset Catch event, which means a Scammer obtains Assets from Targets.

Phishing Contribution: In the context of phishing attacks, Phishing Contributions (*event*) are events that compose Phishing Attacks.

Phishing Enabler: In the context of phishing attacks, a Phishing Enabler (*roleMixin*, *object*) is an ancillary entity (a Target's phone number, email address, or computer network) that is not necessarily an Asset sought by a Scammer though a Phishing Attack.

Prevention: In the context of modeling dynamical aspects of reality, prevention is a relation between two types of events: *prevention*(E_T,E'_T) implies that the occurrence of events of type E_T brings about situations that are *incompatible with* the conditions required for the occurrence of events of type E'_T. These event and situation types are *semi-saturated* in the sense that there is the presence (co-reference) of the same disposition and bearer. For example, it is the event of *Humidifying object x* that prevents an event of *Catching on Fire of object x*. Obviously, humidifying flammable objects, in general, does not prevent other flammable objects from catching on fire. Besides, notice an event (or a type of event) cannot hold a prevention relation with a specific event, which is always an immutable existing entity, but an event can prevent a type of event, therefore precluding the occurrence of instances of this type.

Given this definition, there is a sense of distinguishing between two types of *indirect prevention*. One way of producing indirect prevention is if an event e causes an event *e'*, and *e'* prevents events of type E_T, so we say e indirectly prevents E_T. Another way of producing indirect prevention is if an event e prevents events of type

E_T, which is causally connected to E'_T, so we say e indirectly prevents E'_T. For example, an event *my car engine failure* causes the event *my car stops in the traffic*, which prevents the events of (semi-saturated) type *me attending the job interview*; if I had attended the job interview, *I would have gotten the job* - a type of event that is historically dependent on the events of type *me attending the job interview*. Indirect prevention plays an important role in security engineering because Security Mechanisms (a) may produce a chain of events (Control Chain Events) that eventually prevents directly the desired type of event or (b) may block a causal chain of undesired types of events (Threat Events).

Protected Subject: In the security domain, a Protected Subject (*roleMixin*, *object*) is a Risk Subject whose Intentions are positively impacted by Protection Events.

Protection Trigger: In the security domain, a Protection Trigger (*situation*) is a situation that triggers Protectiom Events. Example: an overcurrent detection triggers a circuit breaker action.

Q

...

R

...

S

Security Designer: In the security domain, an Agent who is responsible for the creation or implementation of a Security Mechanism is called Security Designer (*roleMixin*, *object*). Example: Cybersecurity experts at an IT company.

Security Mechanism: In the security domain, a Security Mechanism (*roleMixin*, *object*) is an object (physical or social) designed to be a *countermeasure to* certain types of Risk Events by aggregating Control Capabilities. Examples: walls, warning signs, air defense systems, security standards, and anti-COVID-19 rules.

Scammer: In the context of phishing attacks, a Scammer (*roleMixin*, *object*) is a Threat Object that possesses an Intention to Phish and Impersonation Capability to Deceive Target.

Specific Intention: In the context of security, an Intention (category, mode) can be generic or specific, according to how specific the situation that satisfies it is. For example, in the aerospace domain, some goals related to the costs of the mission are generic because they can be satisfied by more funding or an assurance; even goals related to replaceable engineering parts can be satisfied by other parts of the same type. However, the completion of the mission is a specific goal that can only be satisfied by a specific situation. This distinction is important because certain Security Mechanisms only work for generic goals. For instance, a space company that transfers some of its risks to an insurance company can be protected from financial loss, but not from the losses caused by the explosion of a space shuttle. Ultimately, Generic Intention can only be impacted by a setting with generic Value Objects (money, for example), but the Specific Intention may be satisfied by a specific setting with generic Value Objects (say, the need for money under a deadline of bankruptcy).

T

Target: In the context of phishing attacks, a victim is called Target (*roleMixin*, *object*), a subtype of Risk Subject.

Target's Fragility: Many factors influence someone to fall for a Phishing attack. These are called Target's Fragilities (*mode*, *role*). Examples: Innocence, Loneliness, Complacency, Greed, Urgency, Fear, Desire to Please, Ignorance, Hurry, Curiosity, Distraction.

U

...

V

Vulnerability Condition: In the context of phishing attacks, a Vulnerability Condition (situation) is a state of affairs in which a Target bears certain dispositions (Target's Fragilities), such that an Asset Catch event is triggered. Examples: Being in a hurry. Being tired. Being naive.

W

...

X

...

Y

...

Z

...

Bibliography

Adach, Malina et al. (2022). "A Combined Security Ontology based on the Unified Foundational Ontology". In: *Intl. Conf. on Semantic Computing*, pp. 187–194.

Aghamohammadpour, Ali, Ebrahim Mahdipour, and Iman Attarzadeh (2022). "Architecting threat hunting system based on the DODAF framework". In: *The Journal of Supercomputing*, pp. 1–28.

Akbar, Khandakar Ashrafi et al. (2022). "Knowledge Mining in Cybersecurity: From Attack to Defense". In: *Data and Applications Security and Privacy XXXVI. DBSec 2022*. Vol. 13383. Springer, pp. 110–122.

Alkhalil, Zainab et al. (2021). "Phishing attacks: A recent comprehensive study and a new anatomy". In: *Frontiers in Computer Science* 3, p. 563060.

Almeida, JPA et al. (2020). "gUFO: a lightweight implementation of the Unified Foundational Ontology (UFO)". In: *URL http://purl. org/nemo/doc/gufo*.

Almeida, Rafael et al. (2019). "A conceptual model for enterprise risk management". In: *Journal of Enterprise Information Management* 32.5, pp. 843–868.

Almeida, J.P. et al. (2018). "Towards an ontology of scenes and situations". In: *Proc. IEEE CogSIMA'18*. IEEE, pp. 29–35.

— (2019). "Events as entities in ontology-driven conceptual modeling". In: *International Conference on Conceptual Modeling*. Springer, pp. 469–483.

Alshanfari, Issam et al. (2020). "Ontology-based formal specifications for social engineering". In: *International Journal of Technology Management and Information System* 2.1, pp. 35–46.

Amaral, Glenda et al. (2019). "Towards a reference ontology of trust". In: *Intl. Conf. on Cooperative Information Systems*. Vol. 11877, pp. 3–21.

An Wang et al (2010). "An ontological approach to computer system security". In: *Information Security Journal: A Global Perspective* 19.2, pp. 61–73.

Arp, R. et al. (2015). *Building ontologies with basic formal ontology*. Mit Press.

Artem, Zaitsev (2018). "Comparison of STS and ArchiMate Risk and Security Overlay". Master's Thesis. University of Tartu.

Asnar, Yudistira, Paolo Giorgini, and John Mylopoulos (2011). "Goal-driven risk assessment in requirements engineering". In: *Requirements Engineering* 16, pp. 101–116.

Azevedo, C. et al. (2015). "Modeling resources and capabilities in enterprise architecture: A well-founded ontology-based proposal for ArchiMate". In: *Information systems* 54, pp. 235–262.

Baltimore, Joseph A (2019). "Expanding the vector model for dispositionalist approaches to causation". In: *Synthese* 196.12, pp. 5083–5098.

Band, Iver et al. (2015). *Modeling enterprise risk management and security with the ArchiMate language*.

Band, Iver et al. (2019). *How to Model Enterprise Risk Management and Security with the ArchiMate Language*. Tech. rep. W172. The Open Group.

Baratella, Riccardo et al. (2022). "Understanding and Modeling Prevention". In: *Research Challenges in Information Science. RCIS 2022*. Vol. 389–405. Springer, pp. 389–405.

Benevides, A.B. et al. (2019). "Representing a reference foundational ontology of events in SROIQ". In: *Applied Ontology* 14.3, pp. 293–334.

Berg, Bibi van den, Pauline Hutten, and Ruth Prins (2021). "Security and safety: An integrative perspective". In: *Intl. Security Management*. Springer, pp. 13–27.

Bird, Alexander (1998). "Dispositions and Antidotes". In: *The Philosophical Quarterly* 48.191.

Blanco et al (2008). "A systematic review and comparison of security ontologies". In: *3rd Intl. Conf. Availability, Reliability and Security*. Ieee, pp. 813–820.

— (2011). "Basis for an integrated security ontology according to a systematic review of existing proposals". In: *Computer Standards & Interfaces* 33.4.

Blomqvist, Eva and Kurt Sandkuhl (2005). "Patterns in ontology engineering: Classification of ontology patterns". In: *International Conference on Enterprise Information Systems*. Vol. 4. SCITEPRESS, pp. 413–416.

Bosch, S.F. van den (2014). "Designing secure enterprise architectures. A comprehensive approach: framework, method, and modelling language". Master's Thesis. Enschede, The Netherlands: University of Twente.

Bradley, Steven (2021). *Modelling SABSA® with ArchiMate®*. Tech. rep. T100. The SABSA Institute.

Calhau, Rodrigo F. and Joao Paulo A. Almeida (2022). "Zooming in on Competences in Ontology-based Enterprise Architecture Modeling". In: *2022 IEEE 26st International Enterprise Distributed Object Computing Workshop (EDOCW)*.

Calhau, Rodrigo F., Carlos L. B. Azevedo, and João Paulo A. Almeida (2021). "Towards Ontology-based Competence Modeling in Enterprise Architecture". In: *25th IEEE Int. EDOC Conference (EDOC 2021)*. IEEE. DOI: 10.1109/edoc52215.2021.00018.

Calhau, Rodrigo F et al. (2023a). "A System Core Ontology for Capability Emergence Modeling". In: *27th IEEE Int. EDOC Conference (EDOC 2023)*. IEEE.

Calhau, Rodrigo F. et al. (June 2023b). "Modeling Competence Framework Elements with an Ontology-based Approach". In: *2023 IEEE 25th Conference on Business Informatics (CBI)*. IEEE. DOI: 10.1109/cbi58679.2023.10187498. URL: https://doi.org/10.1109/cbi58679.2023.10187498.

Carlson, Carl S (2014). "Understanding and applying the fundamentals of FMEAs". In: *Annual Reliability and Maintainability Symposium*. Vol. 10, pp. 1–35.

Chiew, Kang Leng, Kelvin Sheng Chek Yong, and Choon Lin Tan (2018). "A survey of phishing attacks: Their types, vectors and technical approaches". In: *Expert Systems with Applications* 106, pp. 1–20.

Choi, Sungho and Michael Fara (2021). "Dispositions". In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Spring 2021. Stanford University.

Debbech, Sana et al. (2020). "An Ontological Approach to Support Dysfunctional Analysis for Railway Systems Design." In: *J. Univers. Comput. Sci.* 26.5, pp. 549–582.

Donner, Marc (2003). "Toward a security ontology". In: *IEEE Security & Privacy* 3, pp. 6–7.

Dresch, Aline, Daniel Pacheco Lacerda, and José Antônio Valle Antunes (2015). "Design Science Research". In: *Design Science Research: A Method for Science and Technology Advancement*. Cham: Springer International Publishing, pp. 67–102. ISBN: 978-3-319-07374-3. DOI: 10.1007/978-3-319-07374-3_4. URL: https://doi.org/10.1007/978-3-319-07374-3_4.

Duarte, Bruno Borlini et al. (2021). "An ontological analysis of software system anomalies and their associated risks". In: *Data & Knowledge Engineering* 134, p. 101892.

Duarte, B. et al. (2021). "An ontological analysis of software system anomalies and their associated risks". In: *Data & Knowledge Engineering* 134, p. 101892.

Ellerm, Augustus and Miguel Ehécatl Morales-Trujillo (2020). "Modelling security aspects with archimate: a systematic mapping study". In: *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, pp. 577–584.

Ellerm et al (2020). "Modelling Security Aspects with ArchiMate: A Systematic Mapping Study". In: *Euromicro Conf. on Software Engineering and Advanced Applications*. IEEE, pp. 577–584.

Fabio, I. et al. (2021). ""What exactly is a lockdown?": Towards an Ontology-based modeling of lockdown interventions during the COVID-19 pandemic". In: *Brazilian Seminar on Ontology Research (ONTOBRAS 2021)*.

Fernandes, Andre D, Daniel Ramalho, and Miguel Mira da Silva (2022). "Enterprise Risk Management and Information Systems: a Systematic Literature Review". In: *International Conference on Information Resources Management (CON-FIRM)*. Association for Information Systems.

Fumagalli, Mattia et al. (2023). "On the Semantics of Risk Propagation". In: *International Conference on Research Challenges in Information Science*. Springer, pp. 69–86.

Gangemi, Aldo and Valentina Presutti (2009). "Ontology design patterns". In: *Handbook on ontologies*. Springer, pp. 221–243.

Goldfain, A. et al. (2010). "Dispositions and the infectious disease ontology". In: *Formal Ontology in Information Systems*. IOS Press, pp. 400–413.

Grandry, Eric, Christophe Feltus, and Eric Dubois (2013). "Conceptual integration of enterprise architecture management and security risk management". In: *2013 17th IEEE International Enterprise Distributed Object Computing Conference Workshops*. IEEE, pp. 114–123.

Griffo, Cristine (2015). "Ufo-l: A core ontology of legal concepts built from a legal relations perspective". In: *Doctoral Consortium Contributions, IC3K-KEOD*.

Grov, Gudmund, Federico Mancini, and Elsie Margrethe Staff Mestl (2019). "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, pp. 215–225.

Guarino, Nicola, Giancarlo Guizzardi, and John Mylopoulos (2020). "On the philosophical foundations of conceptual models". In: *Information Modelling and Knowledge Bases* 31.321, p. 1.

Guarino, Nicola, Daniel Oberle, and Steffen Staab (2009). "What is an ontology?" In: *Handbook on Ontologies*. Springer.

Guarino, Nicola and Christopher A Welty (2004). "An overview of OntoClean". In: *Handbook on ontologies*, pp. 151–171.

Guizzardi, Giancarlo (2005). *Ontological foundations for structural conceptual models*. CTIT, Centre for Telematics and Information Technology.

— (2006). "The role of foundational ontologies for conceptual modeling and domain ontology representation". In: *2006 7th International Baltic Conf. on databases and information systems*. IEEE, pp. 17–25.

— (2007a). "On ontology, ontologies, conceptualizations, modeling languages". In: *and (Meta) Models, Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV, IOS*.

— (2007b). "On ontology, ontologies, conceptualizations, modeling languages, and (meta) models". In: *Frontiers in artificial intelligence and applications* 155, p. 18.

Guizzardi, Giancarlo (2014). "Ontological patterns, anti-patterns and pattern languages for next-generation conceptual modeling". In: *Conceptual Modeling: 33rd International Conference, ER 2014, Atlanta, GA, USA, October 27-29, 2014. Proceedings 33*. Springer, pp. 13–27.

— (2020). "Ontology, ontologies and the "I" of FAIR". In: *Data Intelligence* 2.1-2, pp. 181–191.

Guizzardi, Giancarlo and Gerd Wagner (2010). "Towards an ontological foundation of discrete event simulation". In: *Proceedings of the 2010 Winter Simulation Conference*. IEEE, pp. 652–664.

Guizzardi, Giancarlo et al. (2008). "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*, pp. 127–140.

— (2021a). "Ontological Unpacking as Explanation: The Case of the Viral Conceptual Model". In: *Conceptual Modeling. ER 2021*, pp. 356–366.

Guizzardi, Giancarlo et al. (2021b). "Types and taxonomic structures in conceptual modeling: A novel ontological theory and engineering support". In: *Data & Knowledge Engineering* 134, p. 101891.

Guizzardi, Giancarlo et al. (2022). "UFO: Unified foundational ontology". In: *Applied ontology* 17.1, pp. 1–44.

Guizzardi et al (2015). "Towards ontological foundations for conceptual modeling: The unified foundational ontology (UFO) story". In: *Applied ontology* 10.3-4, pp. 259–271.

Guizzardi, G. et al. (2013). "Towards ontological foundations for the conceptual modeling of events". In: *Int. Conf. on Conceptual Modeling*. Springer, pp. 327–341.

Guizzardi, R. et al. (2013). "Ontological distinctions between means-end and contribution links in the i* framework". In: *Int. Conf. on Conceptual Modeling*. Springer.

Hevner, Alan R et al. (2004). "Design science in information systems research". In: *MIS quarterly*, pp. 75–105.

Hong, Jason (2012). "The State of Phishing Attacks". In: *Commun. ACM* 55.1, pp. 74–81. ISSN: 0001-0782. DOI: 10.1145/2063176.2063197. URL: https://doi.org/10.1145/2063176.2063197.

Hoogenboom, Cedric (2019). "An Enterprise Architecture Approach to Implementing the NIST Cyber Security Framework". Master's Thesis. Leiden, The Netherlands: Leiden University.

Internet Crime Complaint Center (2022). *Internet Crime Report*. Tech. rep. Federal Bureau of Investigation of The United States of America. URL: https://www.ic3.gov/Media/PDF/AnnualReport/2022_IC3Report.pdf.

ISO (2018). *ISO 31000:2018 - Risk management – Guidelines*.

ISO/IEC (2019). *ISO/IEC 31010:2019 - Risk management – Risk Assessment Techniques*.

Jacobsen, Annika et al. (2020). "FAIR Principles: interpretations and implementation considerations". In: *Data Intelligence* 2.1-2, pp. 10–29.

Jakobsson, Markus (2005). "Modeling and preventing phishing attacks". In: *Financial Cryptography*. Vol. 5. Citeseer.

Jakobsson, Markus and Steven Myers (2006). *Phishing and countermeasures: understanding the increasing problem of electronic identity theft*. John Wiley & Sons.

Jansson, K and Rossouw von Solms (2013). "Phishing for phishing awareness". In: *Behaviour & information technology* 32.6, pp. 584–593.

Jonkers, Henk and Dick A. C. Quartel (2016). "Enterprise Architecture-Based Risk and Security Modelling and Analysis". In: *Graphical Models for Security*. Ed.

by Barbara Kordy, Mathias Ekstedt, and Dong Seong Kim. Vol. 9987. Cham: Springer, pp. 94–101. ISBN: 978-3-319-46263-9.

Kaiser, Florian K et al. (2022). "Cyber threat intelligence enabled automated attack incident response". In: *2022 3rd International Conference on Next Generation Computing Applications (NextComp)*. IEEE, pp. 1–6.

Kaloroumakis, Peter E and Michael J Smith (2021). "Toward a knowledge graph of cybersecurity countermeasures". In: *The MITRE Corporation*. URL: https://d3fend.mitre.org/resources/D3FEND.pdf.

Katsikas, Sokratis K. (2013). "Risk Management". In: *Computer and Information Security Handbook*. Ed. by John R. Vacca. 3rd ed. Morgan Kaufmann, pp. 507–527. ISBN: 978-0-12-803843-7.

Keet, C Maria (2011). "The use of foundational ontologies in ontology development: an empirical assessment". In: *ESWC*. Springer, pp. 321–335.

Kjellén, Urban (2000). *Prevention of accidents through experience feedback*. CRC Press.

Kovalenko et al (2018). "Knowledge Model and Ontology for Security Services". In: *Intl. Conf. on System Analysis & Intelligent Computing*. IEEE, pp. 1–4.

Lankhorst, Marc (2017). *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer.

Lastdrager, Elmer EH (2014). "Achieving a consensual definition of phishing based on a systematic review of the literature". In: *Crime Science* 3.1, pp. 1–10.

Leveson, Nancy G and Clark S Turner (1993). "An investigation of the Therac-25 accidents". In: *Computer* 26.7, pp. 18–41.

Lewis, David (1997). "Finkish dispositions". In: *The Philosophical Quarterly* 47.187.

Li, Tong, Xiaowei Wang, and Yeming Ni (2022). "Aligning social concerns with information system security: A fundamental ontology for social engineering". In: *Information Systems* 104, p. 101699.

Luh, Robert et al. (2022). "PenQuest reloaded: A digital cyber defense game for technical education". In: *2022 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, pp. 906–914.

Lund, Mass Soldal, Bjørnar Solhaug, and Ketil Stølen (2010). *Model-driven risk analysis: the CORAS approach*. Springer Science & Business Media.

Martins, Beatriz Franco et al. (2020). "Conceptual characterization of cybersecurity ontologies". In: *IFIP Working Conference on The Practice of Enterprise Modeling*. Springer, pp. 323–338.

— (2022). "A framework for conceptual characterization of ontologies and its application in the cybersecurity domain". In: *Software and Systems Modeling* 21.4, pp. 1437–1464.

Mayer, Nicolas and Christophe Feltus (2017). "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, pp. 106–116.

Meriah et al (2020). "Analysing Information Security Risk Ontologies". In: *International Journal of Systems and Software Security and Protection* 11.1, pp. 1–16.

Miranda, G. et al. (2019). "Foundational Choices in Enterprise Architecture: The Case of Capability in Defense Frameworks". In: *Proc. IEEE EDOC'19*. IEEE, pp. 31–40.

Mitzen, Jennifer (2006). "Ontological security in world politics: State identity and the security dilemma". In: *European journal of international relations* 12.3, pp. 341–370.

Molnar, George and Stephen Mumford (Nov. 2006). *Powers*. Oxford University Press.

Moura, Leonardo de and Sebastian Ullrich (2021). "The lean 4 theorem prover and programming language". In: *Automated Deduction–CADE 28: 28th International Conference on Automated Deduction, Virtual Event, July 12–15, 2021, Proceedings 28*. Springer, pp. 625–635.

Mouton, Francois et al. (2014). "Towards an ontological model defining the social engineering domain". In: *ICT and Society: 11th IFIP TC 9 International Conference on Human Choice and Computers, HCC11 2014, Turku, Finland, July 30–August 1, 2014. Proceedings 11*. Springer, pp. 266–279.

Mumford, Stephen (2003). *Dispositions*. Clarendon Press.

Mumford, Stephen and Rani Lill Anjum (2011). *Getting causes from powers*. OUP.

— (2018). "Powers and Potentiality". In: *Handbook of Potentiality*. Springer Netherlands, pp. 261–278.

Mylopoulos, John (1992). "Conceptual modelling and Telos". In: *Conceptual modelling, databases, and CASE: An integrated view of information system development*, pp. 49–68.

Nardi et al (2015). "A commitment-based reference ontology for services". In: *Information systems* 54, pp. 263–288.

Oliveira, Ítalo, Rodrigo F Calhau, and Giancarlo Guizzardi (2023). "Toward a phishing attack ontology". In: *ER-Companion 2023: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium* (Lisbon, Portugal). CEUR Workshop Proceedings 3618. Aachen, pp. 10–21. URL: https://ceur-ws.org/Vol-3618/forum_paper_25.pdf.

Oliveira, Ítalo et al. (2021). "How FAIR are Security Core Ontologies? A Systematic Mapping Study". In: *Research Challenges in Information Science*. Pp. 107–123.

Oliveira, Ítalo et al. (2022a). "An Ontology of Security from a Risk Treatment Perspective". In: *Conceptual Modeling. ER 2022*. Vol. 13607. Cham: Springer, pp. 365–379. ISBN: 978-3-031-17995-2. DOI: 10.1007/978-3-031-17995-2_26.

Oliveira, Ítalo et al. (2022b). "Ontological Analysis and Redesign of Security Modeling in ArchiMate". In: *The Practice of Enterprise Modeling. PoEM 2022*. Vol. 456. Cham: Springer, pp. 82–98. ISBN: 978-3-031-21488-2. DOI: 10.1007/978-3-031-21488-2_6.

Oliveira, Ítalo et al. (2023). "Boosting D3FEND: Ontological analysis and recommendations". In: *Formal Ontology in Information Systems: Proceedings of the Thirteenth International Conference (FOIS 2023)*. Vol. 377. Frontiers in Artificial Intelligence and Applications. IOS Press. DOI: 10.3233/FAIA231138.

Oliveira, Ítalo et al. (2024). "Ontology-based Security Modeling in ArchiMate". In: *Software and Systems Modeling*. in press.

Petersen et al (2008). "Systematic mapping studies in software engineering". In: *12th Intl. Conf. Evaluation and Assessment in Soft. Engineering (EASE) 12*, pp. 1–10.

Poli, Roberto (1993). "Husserl's conception of formal ontology". In: *History and philosophy of logic* 14.1, pp. 1–14.

Quine, Willard V (1948). "On what there is". In: *The review of metaphysics*, pp. 21–38.

Reason, James (1990). "The contribution of latent human failures to the breakdown of complex systems". In: *Philosophical Transactions of the Royal Society of London. B, Biological Sciences* 327.1241, pp. 475–484.

Rosemann, Michael, Peter Green, and Marta Indulska (2004). "A reference methodology for conducting ontological analyses". In: *Conceptual Modeling. ER 2004*. Vol. 3288. Springer, pp. 110–121.

Roussey et al (2011). "An introduction to ontologies and ontology engineering". In: *Ontologies in Urban development projects*. Springer, pp. 9–38.

Ruijter, Alex de and Frank Guldenmund (2016). "The bowtie method: A review". In: *Safety science* 88, pp. 211–218.

Ruy, Fabiano et al. (2017). "From reference ontologies to ontology patterns and back". In: *Data & Knowledge Engineering* 109, pp. 41–69.

Sadlek, Lukáš, Pavel Čeleda, and Daniel Tovarňák (2022). "Identification of Attack Paths Using Kill Chain and Attack Graphs". In: *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*. IEEE, pp. 1–6.

Salahdine, Fatima and Naima Kaabouch (2019). "Social engineering attacks: A survey". In: *Future internet* 11.4.

Sales, Tiago Prince (2019). "Ontological Foundations for Strategic Business Modeling: The Case of Value, Risk and Competition". PhD thesis. University of Trento.

Sales et al (2018). "The common ontology of value and risk". In: *Intl. Conf. on Conceptual Modeling*. Springer, pp. 121–135.

Sales, T. et al. (2018). "Ontological analysis and redesign of risk modeling in archimate". In: *Proc. IEEE EDOC'18*. IEEE, pp. 154–163.

— (2019). "A pattern language for value modeling in ArchiMate". In: *Proc. CAISE'19*. Springer, pp. 230–245.

Samonas, Spyridon and David Coss (2014). "The CIA strikes back: Redefining confidentiality, integrity and availability in security." In: *Journal of Information System Security* 10.3.

Saud, Yaneira E, Kumar Israni, and Jeremy Goddard (2014). "Bow-tie diagrams in downstream hazard identification and risk assessment". In: *Process Safety Progress* 33.1, pp. 26–35.

Schulz, Stefan (2018). "The Role of Foundational Ontologies for Preventing Bad Ontology Design". In: *4th Joint Ontology Workshops (JOWO)*. Vol. 2205. CEUR-WS.

Shin, Youngsup et al. (2022). "Focusing on the Weakest Link: A Similarity Analysis on Phishing Campaigns Based on the ATT&CK Matrix". In: *Security and Communication Networks* 2022.

Sicilia et al (2015). "What are information security ontologies useful for?" In: *Research Conf. on Metadata and Semantics Research*. Springer, pp. 51–61.

Siena, Alberto, Mirko Morandini, and Angelo Susi (2014). "Modelling risks in open source software component selection". In: *Conceptual Modeling: 33rd International Conference, ER 2014, Atlanta, GA, USA, October 27-29, 2014. Proceedings 33*. Springer, pp. 335–348.

Sikos, Leslie F (2019). "OWL ontologies in cybersecurity: conceptual modeling of cyber-knowledge". In: *AI in Cybersecurity*. Springer, pp. 1–17.

Spreafico, Christian, Davide Russo, and Caterina Rizzi (2017). "A state-of-the-art review of FMEA/FMECA including patents". In: *computer science review* 25, pp. 19–28.

Strom, Blake E et al. (2020). "MITRE ATT&CK®: Design and Philosophy". In: *The MITRE Corporation*. URL: https://www.mitre.org/news-insights/publication/mitre-attck-design-and-philosophy.

Studer et al (1998). "Knowledge engineering: principles and methods". In: *Data & knowledge engineering* 25.1-2, pp. 161–197.

Sumra, Irshad Ahmed, Halabi Bin Hasbullah, and Jamalul-lail Bin AbManan (2014). "Attacks on security goals (confidentiality, integrity, availability) in VANET: a survey". In: *Vehicular Ad-hoc Networks for Smart Cities: First International Workshop, 2014*. Springer, pp. 51–61.

Tao et al (2018). "Multi-layer cloud architectural model and ontology-based security service framework for IoT-based smart homes". In: *Future Generation Computer Systems* 78, pp. 1040–1051.

Tchakounté, Franklin, Djeguedem Molengar, and Justin Moskolai Ngossaha (2020). "A description logic ontology for email phishing". In: *International Journal of Information Security Science* 9.1, pp. 44–63.

The Open Group (2019). "Integrating Risk and Security within a TOGAF® Enterprise Architecture". In: *The Open Group Guide white paper*. URL: www.opengroup.org/library/g152.

— (2023). *ArchiMate® 3.2 Specification*. URL: https://pubs.opengroup.org/architecture/archimate3-doc/.

Tovstukha, Iuliia (2017). "Management of Security Risks in the Enterprise Architecture using ArchiMate and Mal-activities". Master's Thesis. University of Tartu.

Tseng, Shian-Shyong et al. (2013). "Building a frame-based anti-phishing model based on phishing ontology". In: *International Conference on Advances in Information Technology*.

van Ede, Thijs Sebastiaan (Nov. 2023). "Comprehending Security Events: Context-Based Identification and Explanation". English. PhD Thesis - Research UT, graduation UT. Netherlands: University of Twente. ISBN: 978-90-365-5888-4. DOI: 10.3990/1.9789036558891.

Varzi, Achille (2019). "Carnapian Engineering." In: *Ontology Makes Sense*, pp. 3–23.

Verdonck, Michael and Frederik Gailly (2016). "Insights on the use and application of ontology and conceptual modeling languages in ontology-driven conceptual modeling". In: *International Conference on Conceptual Modeling*. Springer, pp. 83–97.

Verdonck, M. et al. (2015). "Ontology-driven conceptual modeling: A systematic literature mapping and review". In: *Applied Ontology* 10.3-4, pp. 197–227.

— (2019). "Comparing traditional conceptual modeling with ontology-driven conceptual modeling: An empirical study". In: *Information Systems* 81, pp. 92–103.

Wagner, Gerd (2018). "Information and Process Modeling for Simulation–Part I". In: *Journal of Simulation Engineering* 1, pp. 1–1.

Wang, Zuoguang, Limin Sun, and Hongsong Zhu (2020). "Defining social engineering in cybersecurity". In: *IEEE Access* 8, pp. 85094–85115.

Wang, Zuoguang, Hongsong Zhu, and Limin Sun (2021). "Social engineering in cybersecurity: Effect mechanisms, human vulnerabilities and attack methods". In: *IEEE Access* 9, pp. 11895–11910.

Wang, Zuoguang et al. (2021). "Social engineering in cybersecurity: a domain ontology and knowledge graph application examples". In: *Cybersecurity* 4, pp. 1–21.

Xiong, Wenjun and Robert Lagerström (2019). "Threat modeling–A systematic literature review". In: *Computers & security* 84, pp. 53–69.

Zambon, Eduardo and Giancarlo Guizzardi (2017). "Formal definition of a general ontology pattern language using a graph grammar". In: *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, pp. 1–10.

Zemmouchi-Ghomari et al (2009). "Reference ontology". In: *Int. Conf. on Signal Image Technology and Internet Based Systems*. IEEE, pp. 485–491.

Zhou, Jiale et al. (2017). "An ontological approach to identify the causes of hazards for safety-critical systems". In: *System Reliability and Safety*, pp. 405–413.

Selected Studies

Agrawal, Vivek (2016). "Towards the Ontology of ISO/IEC 27005: 2011 Risk Management Standard". In: *Intl. Symp. on Human Aspects of Information Security & Assurance*, pp. 101–111.

Amaral, Fernando Naufel do et al. (2006). "An ontology-based approach to the formalization of information security policies". In: *Intl. Enterprise Distributed Object Computing Conf. Ws.* IEEE.

Arogundade et al (2012). "Towards an ontological approach to information system security and safety requirement modeling and reuse". In: *Information Security Journal: A Global Perspective* 21.3, pp. 137–149.

Avizienis et al (2004). "Basic concepts and taxonomy of dependable and secure computing". In: *IEEE transactions on dependable and secure computing* 1.1, pp. 11–33.

Beji et al (2009). "Security ontology proposal for mobile applications". In: *10th Intl. Conf. Mobile Data Management: Systems, Services and Middleware*. IEEE.

Blanco, Francisco J. et al. (2012). "Vulnerapedia: Security Knowledge Management with an Ontology". In: *Intl. Conf. on Agents and Artificial Intelligence*, pp. 485–490.

Boualem et al (2017). "Maintenance & information security ontology". In: *Intl. Conf. on Control, Decision and Information Technologies*. IEEE, pp. 312–317.

Casola et al (2019). "A First Step Towards an ISO-Based Information Security Domain Ontology". In: *Intl. Conf. on Enabling Technologies: Infrastructure for Collaborative Enterprises*. IEEE, pp. 334–339.

Chen et al (2018). "Research on Ontology-based Network Security Knowledge Map". In: *Intl. Conf. on Cloud Computing, Big Data and Blockchain*. IEEE, pp. 1–7.

Cherdantseva et al (2013). "A reference model of information assurance & security". In: *Intl Conf on Availability, Reliability and Security*. IEEE, pp. 546–555.

Chowdhury, Mohammad Jabed Morshed (2014). "Security risk modelling using SecureUML". In: *16th Int'l Conf. Computer and Information Technology*. IEEE, pp. 420–425.

de Franco Rosa et al (2018). "Towards an ontology of security assessment: A core model proposal". In: *Information Technology-New Generations*. Springer, pp. 75–80.

Dos Santos Moreira et al (2008). "Ontologies for information security management and governance". In: *Information Management & Computer Security*.

Dritsas et al (2005). "Employing ontologies for the development of security critical applications". In: *Challenges of Expanding Internet: E-Commerce, E-Business, and E-Government*. Springer, pp. 187–201.

Ekelhart et al (2006a). "Ontology-based business knowledge for simulating threats to corporate assets". In: *Int. Conf. on Practical Aspects of Knowledge Management*. Springer, pp. 37–48.

— (2006b). "Security ontology: Simulating threats to corporate assets". In: *Intl. Conf. on Information Systems Security*. Springer.

Ekelhart et al (2007). "Security ontologies: Improving quantitative risk analysis". In: *Annual Hawaii Intl. Conf. on System Sciences*. IEEE, 156a–156a.

El-Attar et al (2015). "Extending the UML statecharts notation to model security aspects". In: *IEEE Transactions on Software Engineering* 41.7, pp. 661–690.

Elahi et al (2009). "A modeling ontology for integrating vulnerabilities into security requirements conceptual foundations". In: *Intl. Conf. on Conceptual Modeling*. Springer, pp. 99–114.

Fani et al (2015). "An Ontology for Describing Security Events." In: *SEKE*, pp. 455–460.

Fenz, Stefan et al. (2009). "Formalizing information security knowledge". In: *Intl. Symp. on Information, Computer, and Communications Security*, pp. 183–194.

Fernandez et al (2014). "A security reference architecture for cloud systems". In: *WICSA 2014 Companion Volume*, pp. 1–5.

Guan et al (2016). "An ontology-based approach to security pattern selection". In: *Intl. J. of Automation and Computing* 13.2, pp. 168–182.

Gyrard, Amelie et al. (2013). "The STAC (security toolbox: attacks & countermeasures) ontology". In: *Intl. Conf. on World Wide Web*, pp. 165–166.

Herzog et al (2007). "An ontology of information security". In: *Intl. J. of Information Security and Privacy* 1.4, pp. 1–23.

Jonsson, Erland (2006). "Towards an integrated conceptual model of security and dependability". In: *Intl. Conf. on Availability, Reliability and Security*. IEEE.

Kang et al (2013). "A security ontology with MDA for software development". In: *Intl. Conf. on Cyber-Enabled Distributed Computing and Knowledge Discovery*, pp. 67–74.

Karyda et al (2006). "An ontology for secure e-government applications". In: *Intl. Conf. on Availability, Reliability and Security*. IEEE, 5–pp.

Kim, Anya et al. (2005). "Security ontology for annotating resources". In: *Int. Conf. on Ontologies, Databases and Applications of Semantics*. Springer, pp. 1483–1499.

Kim et al (2016). "Analytical study of cognitive layered approach for understanding security requirements using problem domain ontology". In: *Asia-Pacific Software Engineering Conference*. IEEE, pp. 97–104.

— (2020). "Understanding and recommending security requirements from problem domain ontology: A cognitive three-layered approach". In: *Journal of Systems and Software* 169, p. 110695.

Korger, Andreas and Joachim Baumeister (2018). "The SECCO ontology for the retrieval and generation of security concepts". In: *Case-Based Reasoning Research and Development*.

Li et al (2020). "An ontology-based learning approach for automatically classifying security requirements". In: *Journal of Systems and Software*, p. 110566.

Lund et al (2003). "UML profile for security assessment". In: *Tech.Report STF A* 3066.

Massacci et al (2011). "An extended ontology for security requirements". In: *Intl. Conf. on Advanced Information Systems Engineering*. Springer, pp. 622–636.

Mayer, Nicolas (2009). "Model-based management of information system security risk". PhD thesis. University of Namur.

Mayer et al (2019). "An integrated conceptual model for information system security risk management supported by enterprise architecture management". In: *Software & Systems Modeling* 18.3, pp. 2285–2312.

Milicevic et al (2010). "Ontology-based evaluation of ISO 27001". In: *Conference on e-Business, e-Services and e-Society*. Springer, pp. 93–102.

Mouratidis, Haralambos et al. (2003). "An ontology for modelling security: The Tropos approach". In: *Intl. Conf. on Knowledge-Based and Intelligent Information and Engineering Systems*. Springer, pp. 1387–1394.

Mozzaquatro, Bruno Augusti et al. (2015). "Towards a reference ontology for security in the Internet of Things". In: *Intl Work. on Measurements & Networking*. IEEE, pp. 1–6.

Oltramari, Alessandro et al. (2014). "Building an Ontology of Cyber Security." In: *Conf. on Semantic Technology for Intelligence, Defense, and Security*. Vol. 1304, pp. 54–61.

— (2015). "Towards a Human Factors Ontology for Cyber Security." In: *Stids*, pp. 26–33.

Parkin, Simon E. et al. (2009). "An information security ontology incorporating human-behavioural implications". In: *Proceedings of SIN'09*, pp. 46–55.

Pereira et al (2012). "An Ontology Approach in Designing Security Information Systems to Support Organizational Security Risk Knowledge." In: *KEOD*, pp. 461–466.

— (2019). "A STAMP-based ontology approach to support safety and security analyses". In: *Journal of Information Security and Applications* 47, pp. 302–319.

Ramanauskaitė et al (2013). "Security ontology for adaptive mapping of security standards". In: *Intl. J. Computers, Communications & Control* 8.6, pp. 813–825.

Schumacher, Markus (2003). "Toward a security core ontology". In: *Security engineering with patterns*. Springer, pp. 87–96.

Souag et al (2015). "A security ontology for security requirements elicitation". In: *Intl. Symp. Engineering secure software and systems*. Springer, pp. 157–177.

Takahashi et al (2015). "Reference ontology for cybersecurity operational information". In: *The Computer Journal* 58.10, pp. 2297–2312.

Tsoumas et al (2006a). "Security-by-Ontology: A knowledge-centric approach". In: *IFIP International Information Security Conference*. Springer, pp. 99–110.

— (2006b). "Towards an ontology-based security management". In: *Intl. Conf. on Advanced Information Networking and Applications*. Vol. 1, pp. 985–992.

Vale et al (2019). "An Ontology for Security Patterns". In: *38th Int. Conf. of the Chilean Computer Science Society*. IEEE, pp. 1–8.

Vorobiev, Artem and Nargiza Bekmamedova (2007). "An ontological approach applied to information security and trust". In: *Australasian Conf. on Information Systems*, p. 114.

Vorobiev et al (2010). "An ontology-driven approach applied to information security". In: *Journal of Research and Practice in Information Technology* 42.1, p. 61.

Yau et al (2014). "An adaptable distributed trust management framework for large-scale secure service-based systems". In: *Computing* 96.10, pp. 925–949.

Zheng-qiu et al (2009). "Semantic security policy for web service". In: *Int. Symp. Parallel and Distributed Processing with Applications*. IEEE, pp. 258–262.