Cybersecurity Toolbox - User Manual

Utilizing the Model-based tool to enhance the Security Analysis Processes in Cross-Domain Systems Engineering

Author

Simon Eschlberger

Center for Dependable Systems Engineering

Salzburg University of Applied Sciences

August 2024

Contents

1 Introduction

Welcome to the user manual of the *Cybersecurity Toolbox*!

A few questions to yourself first:

- You need to have to find a way of how to investigate and improve the security of a complex application involving **distributed virtual and physical components**?
- You are struggling in this task to find a common way of **aggregating the knowledge regarding the systems security** of you, involved experts and results from security analysis tools you use?
- While doing this you are facing the challenge of finding a **baseline of a vocabulary regarding security terms** that every stakeholder can easily learn and understand and even is compatible with most of your software tools?

If you can answer at least one of the above questions to you with a more or less silent *yes*, then surely the *Cybersecurity Toolbox* presented in this document is worth a closer look for you. The Cybersecurity Toolbox encompasses a compact visual domain-specific language (DSL), which focuses on capturing security aspects of IT-intensive cyber-physical systems (CPSs).

Analysis regarding a system's cybersecurity aspects is challenging...

Spoken of today, risks of cyber-incidents regularly leading to financial, material and immaterial damage are not to be considered a niche topic, but inevitably became a major challenge impacting everybody's' daily life. To minimize the overall negative impact for users of IT-products the systems security should be payed high attention during its full lifecycle from the very first start on. This ranges from its design and development phase to its operating, maintenance and lastly its retirement and possible renewal. This statement is true in all the cases if the managed product is composed of software to be sold, cyber-physical system or consumed service.

Whether you encompass roles of a system architect, project owner, operator, developer, electromechanical engineer or a tester: You can and should contribute your expertise to the collaborative project asset to design and operate it in a safe way! Software tools are here to help you on running analytical tasks rating and improving security aspects of your system, but often they lack guidance or compatibility to integrate their inputs and analysis results in a holistic design process of a whole system to be designed.

Additionally, even to interpret the meaning of a finding revealed by such a tool you need to have a quite high level of expertise in the specific field. As an example, a software architect responsible for modifying a part of an application may not be able to understand the meaning of an attack path discovered by pentesting that leads to a command injection vulnerability. The architect likely does not have to know how to technically conduct the attack, but rather where the vulnerability is located and which exploited weaknesses to be mitigated to prevent successful execution of the attack.

So what the *Cybersecurity Toolbox* can do for you?

As you may guess now, the Cybersecurity Toolbox's purpose is not to replace the specific analysis and simulation tools you are using, but to provide a common language and a common way of how to aggregate the gathered precious knowledge from your tools. Also you are able to better plan security analysis and measures to be done as they can be consistently documented at high level abstraction in a model repository capable to serve as the single source of truth.

To summarize shortly, the Cybersecurity Toolbox tries to offer you a widely applicable approach to...

- ... enhance **collaboration between stakeholders** having interest to build a secure system
- ... improve the **interoperability of specific tools and strategies** used in the design process
- ... consistently **aggregate the knowledge** regarding security aspects of the system

1.1 About this Document

This manual serves as your gateway to understanding the basic principles of the *Cybersecurity DSL* and introduces you to effectively utilizing the *Cybersecurity Toolbox* to orchestrate analysis related to cybersecurity employing methods of model-based systems engineering. Its primary aim is to provide you with a comprehensive introduction to practical usage of the *Cybersecurity Toolbox* for system development. If you are more interested in the scientific background of the DSL and relations to standards it relies on, be sure to have a look on our website [1] or the scientific publications of the Center for Dependable Systems Engineering (CDSE) [2].

The majority of the resources in this document are available at our website for free for viewing, downloading and using. This includes the metamodel of the Cybersecurity DSL, the installation files for the Cybersecurity Toolbox and example models used to demonstrate the usage of the Cybersecurity Toolbox. However, to follow the examples in this manual, you need to download and license the software "Enterprise Architect"(EA) from Sparx Systems. EA is a feature-rich modeling IDE widely used for model-based systems engineering (MBSE) in the industry in these days. If you are interested in a specific aspect of the DSL or the toolbox, want to contribute some feedback or need information not available in this document, feel free to contact us via the contact form on the *dsse.at-website* or via eMail.

1.2 Targetted Usage of the Cybersecurity Toolbox

As outlined in the introduction, the Cybersecurity Toolbox is a compact visual modeling language focusing on aggregation of high-level security aspects of CPSs. Structural aspects such as results of pentesting, threat modeling, and failure mode and effects analysis (FMEA) can

[1] https://dsse.at

[2] https://www.en-trust.at/publications/desos/

be consistently consolidated and managed in one shared repository. To accomlish this task the DSL provides a small set of predefined elements and relationships. To improve understanding and stakeholder communication, elements are visually represented and are reduced to make it an easy-to-learn modeling language. Furthermore, names of relations and elements are based on common terminology used in cybersecurity standards. Currently, the commercially availableMBSE-tool *Enterprise Architect* as a modeling environment. This is currently the only existing implementation of the DSL.

2 Language Definition and Implementation

The Cybersecurity Toolbox was developed following a generic structured approach for designing DSLs. Figure 1 visualizes this concept. Each layer of this stack encapsulates a predefined aspect part of the DSL's overall conceptualization and implementation. The created concept called the *TILO-Stack* consists of the layers *Language Ontology*, *Language Specification*, *Language Implementation*, and created *Toolbox Features*.

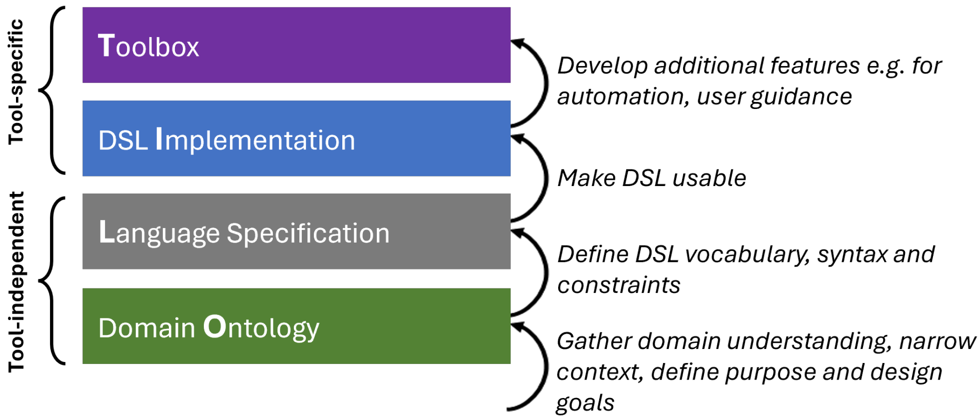


Figure 1: The TILO-Stack for DSL development

This guide mainly explains the layers *Language Specification* and *Language Implementation* in detail. The *Language Ontology* is a theoretical concept that defines the foundation of the language's structure, intentional usage, and semantics. The *Toolbox Features* layer encapsulates things like user-guidance, automation features for modeling, reusable templates, and program-based validation rules.

2.1 Implemented Language Components

A simplified metamodel of the developed Cybersecurity DSL is shown in Figure 2. This is also the way how the visual representation of the DSL looks in the modeling IDE Enterprise Architect. The elements of the *Cybersecurity DSL* are divided into 5 different viewpoints each addressing a different concern emerging somewhere in the overall cybersecurity analysis process. A color-coding system helps identify which elements belong to which diagram type, making navigation easier. The metamodel includes the following viewpoints:

- **TOE Classification (yellow):** Focuses on the system's assets that need analysis and abstracts the system's cybersecurity properties to be protected.

- **Threat Analysis (blue):** Looks at threats exploiting system vulnerabilities and evaluates risks; considers how threats behave and how mitigations reduce these risks.
- **Damage and Impact Analysis (red):** Assesses damage and potential system failures from risks to rate the severity of those risks.
- **Risk Treatment (green):** Makes high-level decisions on how to handle identified and rated risks.
- **Security Requirements (purple):** Defines what needs to be done to ensure system security and measures the effectiveness of countermeasures.

Visual symbols also help to categorize the elements. These outer shapes of elements are used:

- **Circle:** Component or aspect within system boundaries.
- **Triangle:** An external element affecting the system.
- **Rectangle:** An achievable condition.
- **Hexagon:** A decision influencing the system's design.
- **Round-edged square:** Container element holding external artifacts for analysis.

The border around an element shows if it is meant to be changed or not:

- **Solid border line:** Not intended to be changed.
- **Dashed border line:** Intended to be changed or influenced.

The connectors between elements are stereotyped with expressive names to be self-explanatory to some extent. This makes it also easier to talk and discuss model parts between involved stakeholders. Connectors are all of a unidirectional type to prevent confusion about the order of reading. Color-coding furthermore indicates to which element of a equally color-coded viewpoint element the connector is intended to be drawn.

A simplified metamodel of the developed DSL, including all entities and relations, is shown in Figure 2. Each viewpoint will be explained in the following sections. While these viewpoints follow the order of steps in a cybersecurity analysis, it is not mandatory to follow this order. The DSL is flexible and can be used in different ways depending on the user's needs.

2.1.1 Target of Evaluation (TOE) Classification

The first aspect of the DSL clarifies which *Target of Evaluations (TOEs)* it refers to. This narrows down the system model to the parts being analyzed for cybersecurity. It ensures traceability to the system model by associating TOEs with system components. This is mostly meaningful for system models resembling deployment diagrams because cybersecurity aspects depend on technical implementations. TOEs are often mapped 1:1 to system components but can also include collections of components or communication flows. Evaluating *Cybersecurity*

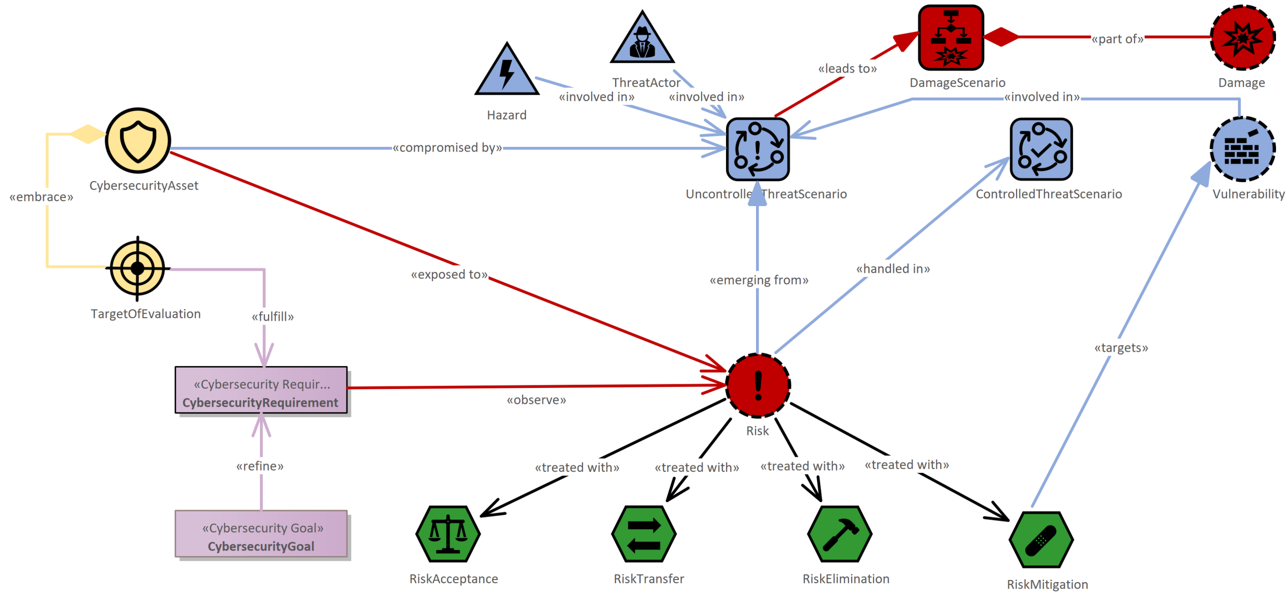


Figure 2: Simplified Metamodel of the *Cybersecurity DSL*

Assets (which need protection for their confidentiality, integrity, or availability) is also part of this phase. If *Security Requirements* are already defined for technical components, they are mapped to the corresponding TOEs.

2.1.2 Threat Analysis: Uncontrolled Threat Scenario

Next is evaluating possible bad actions on *Cybersecurity Assets*, leading to *Risks* that could affect the system. The *Cybersecurity DSL* distinguishes between *Threat Actors* (malicious entities) and *Hazards* (non-malicious risks like natural disasters or technical failures). The core element, the *Uncontrolled Threat Scenario*, describes how these threats interact with the system, resulting in negative impacts. It includes sub-diagrams or artifacts created by external tools, such as attack trees or penetration testing reports. This scenario identifies *Vulnerabilities* and documents *Risks*, which are mapped back to the *Cybersecurity Assets* exposed to them.

2.1.3 Damage and Impact Analysis

This optional but valuable phase, known as *Damage and Impact Analysis*, is conducted after the *Threat Analysis*. It evaluates the relevance of identified *Vulnerabilities* and estimates the severity of *Risks* based on likelihood and impact. Cascading failures can cause significant damage to the system's confidentiality, integrity, and availability. Using structured approaches like FMEA helps assess the potential impact of various *Threat Scenarios*, enabling informed decision-making for *Risk Treatment*.

2.1.4 Risk Treatment

After identifying *Risks*, the *Risk Treatment* view decides how to handle them. Each *Risk* must be assigned to at least one *Risk Treatment Decision* to ensure complete coverage. *Risk Treatment Decisions* balance risk severity and likelihood. Not all countermeasures should be implemented, and some *Risks* may be accepted if their impact and likelihood are low compared to the effort required to reduce them. There are four possible risk treatment decisions:

- **Risk Avoidance:** Eliminate the root cause, possibly by removing a non-essential component.
- **Risk Mitigation:** Reduce the likelihood or impact of a risk.
- **Risk Transfer:** Move the responsibility to another component.
- **Risk Acceptance:** Accept the risk if reducing it is not worth the effort.

The DSL documents these decisions and creates traceability to system components, helping system architects or developers understand their tasks. This consolidation of information fosters transparency and coherence, guiding informed decision-making and facilitating collaboration among stakeholders.

2.1.5 Threat Analysis: Controlled Threat Scenario

While many tools stop after identifying vulnerabilities, threats, risks, and mitigations, the *Cybersecurity DSL* goes further. It ensures countermeasures are effective by constructing a *Controlled Threat Scenario* within the *Threat Analysis* viewpoint. This scenario includes artifacts proving the success of countermeasures. Unlike the *Uncontrolled Threat Scenario*, this one should show less negative impact, indicating that certain attack steps are now unreachable or impossible.

2.1.6 Security Requirements

After identifying and rating *Risks*, and defining *Risk Treatment Decisions*, the final phase defines *Security Requirements*. These specify the actions needed to modify the system design or implementation and monitor *Risks*. *Security Requirements* are precise, allowing compliance to be measured. They are refined from high-level *Security Goals* and mapped back to corresponding TOEs, ensuring a comprehensive understanding of the necessary security measures. This iterative process enables continuous improvement of the system's security in response to evolving threats and vulnerabilities.

2.2 Toolbox

Since the current language implementation is available for Enterprise Architect, the toolbox features are integrated into the modeling environment by using the IDE's programmatic extension mechanism. The toolbox features are designed to guide the user through the modeling process and to ensure the correct usage of the language. These toolbox features exist (black font) or are planned (grey font) to be implemented:

- **Model Templates:** The toolbox provides templates for common modeling tasks. It allows users to create new models quickly by selecting a template and customizing it to their needs
  → For example a selection of model template for a new cybersecurity analysis to be filled out or extended by the user.
- **Model Validation:** The toolbox checks selected parts of the model for completeness and consistency. It ensures that all required elements are present and that the relationships

between elements are correct.
→ For example if every risk has a risk treatment decision assigned

- **Model Guidance:** The toolbox provides guidance on how to use the language. It explains the meaning of each element and relationship and provides examples of how to use them.
→ For example inbuilt language reference window for quick lookup

- **Model Automation:** The toolbox automates repetitive tasks in the modeling process. It generates elements and relationships based on user input and updates the model when changes are made.
→ For example a dialog for batch creation of risk elements dependent on predefined rules

3 Language Components of the Cybersecurity DSL

All language components proposed in the previous chapter are based on a OMG MOF-conformant metamodel, which is available for download at the dsse.at-website [3].

The metamodel incorporates a formal tool-independent *Language Specification*. It incorporates all entities and relations of the Cybersecurity DSL. To employ a standardized model-based approach the OMG Meta Object Facility (MOF) was used to define each general part of a DSL's specification:

- **Abstract Syntax Model (ASM):** Vocabulary of the language, including all elements (entities and relationships), their properties, and their grammar.
- **Concrete Syntax Model (CSM):** Defines how the language is represented, in the case of a visual modeling language this includes element-assigned symbols, colors, and shapes.
- **Semantic Model (SM):** Specifies the meaning languages' elements

All language components including their visual representation and semantics are added to this document in the appendix 6.

3.1 TOE Classification

Figure 3.1 shows the components and their relations in the viewpoint *TOE Classification*.

[3]https://www.dsse.at/

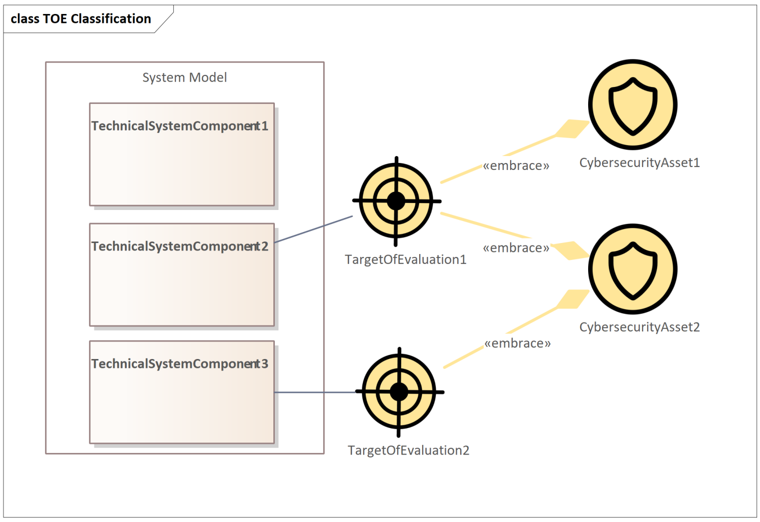


Figure 3: TOE Classification

3.2 Security Requirements

Figure 3.2 shows the components and their relations in the viewpoint *Security Requirements*.

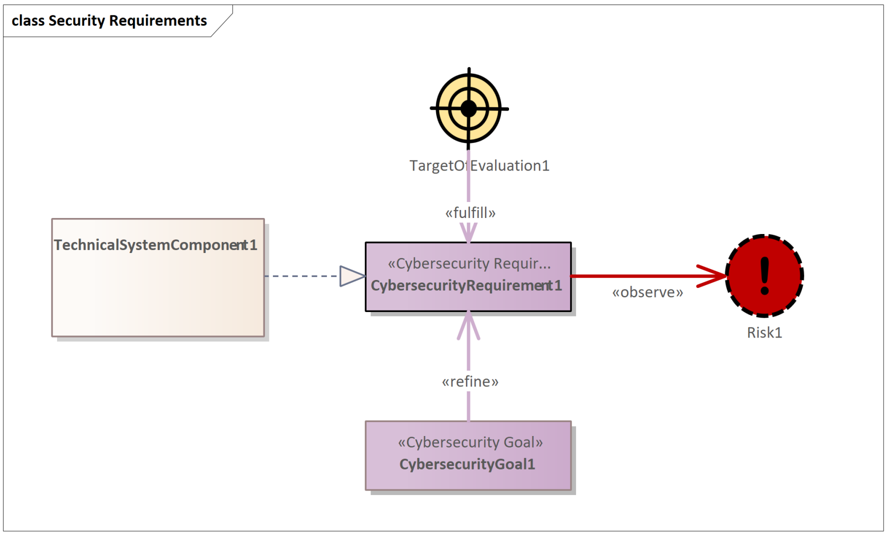


Figure 4: Security Requirements

3.3 Threat Analysis

Figure 3.3 shows the components and their relations in the viewpoint *Threat Analysis*.

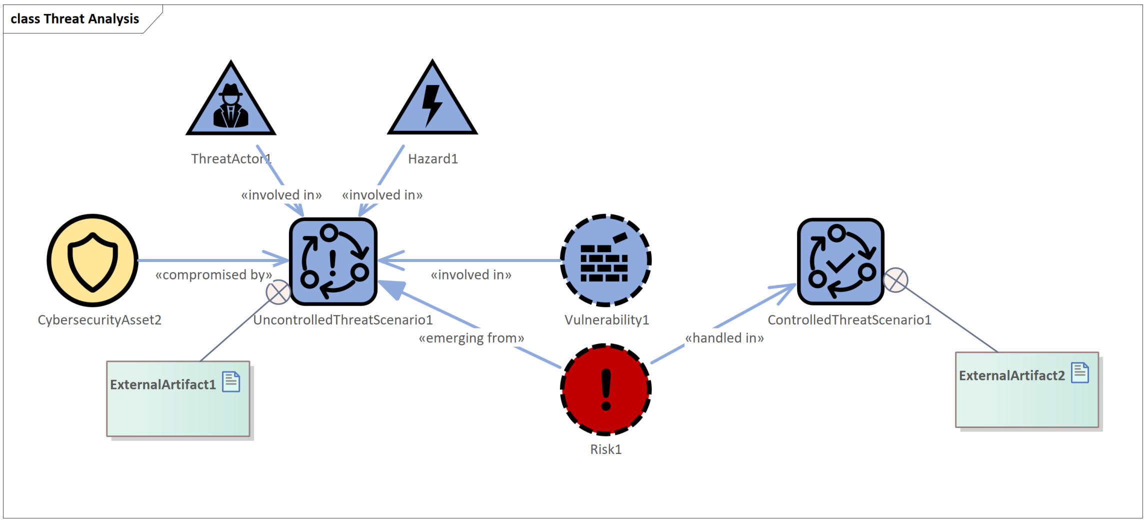


Figure 5: Threat Analysis

3.4 Impact Analysis

Figure 3.4 shows the components and their relations in the viewpoint *Impact Analysis*.

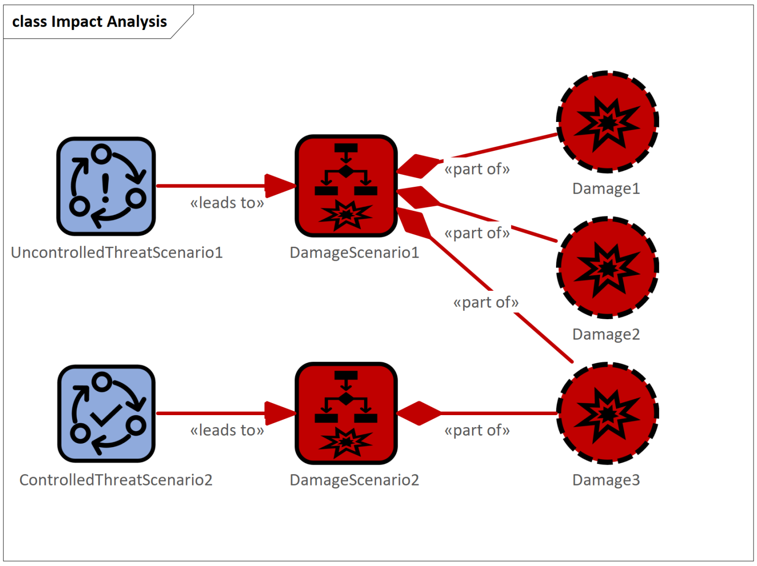


Figure 6: Impact Analysis

3.5 Risk Treatment

Figure 3.5 shows the components and their relations in the viewpoint *Risk Treatment*.

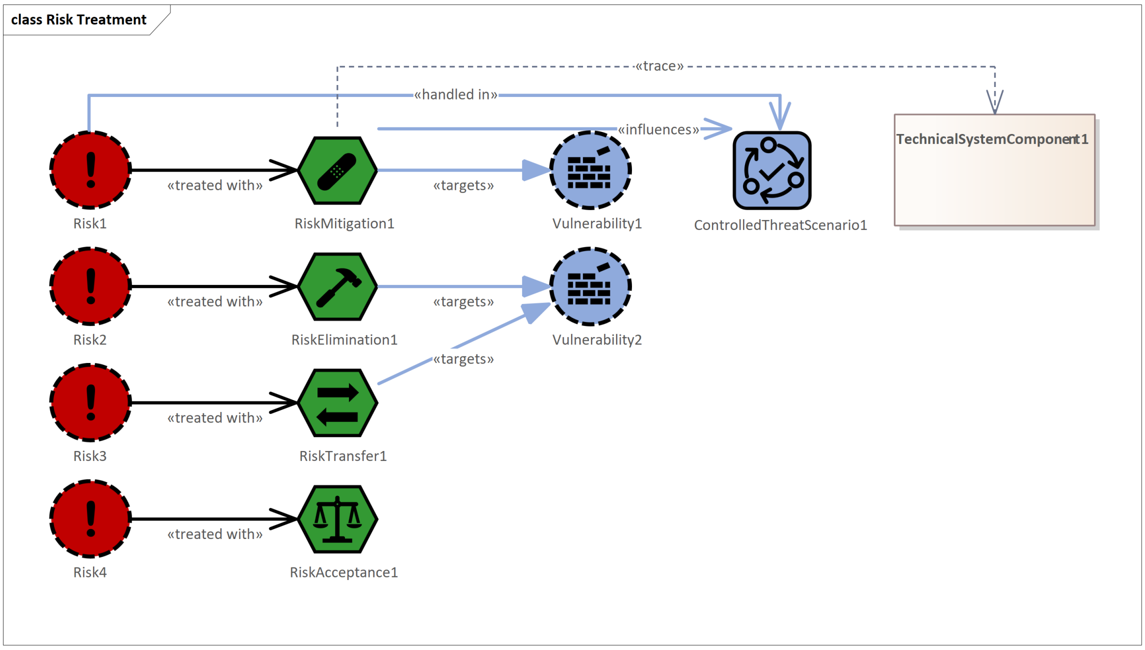


Figure 7: Risk Treatment

4 Examples of Using the Cybersecurity Toolbox

This section contains examples of how to use the Cybersecurity Toolbox. The examples are intended to provide a starting point for users to quickly get a hands-on experience of how to utilize the toolbox to conduct cybersecurity analysis. The examples are not exhaustive and are intended to be used as a reference for users to understand how to use the Cybersecurity Toolbox and the Cybersecurity DSL.

4.1 Only the Cybersecurity DSL: Smart Light Switch

Tutorial coming soon...

4.2 Interfacing with a System Modeling Language: Smart Grid Ransomware Attack

Tutorial coming soon...

5 Installation of the Cybersecurity Toolbox

Following sections tell you how the Cybersecurity Toolbox is installed in modeling environments.

5.1 Installation in Enterprise Architect

The Cybersecurity DSL is currently available as an MDG Technology for Enterprise Architect. To install it, take the following steps:

1. Download the Cybersecurity DSL MDG Technology file from the official website (*Cybersecurity-*

Toolbox.xml) and save it with a persistent path on a location in your local file system.

2. Open Enterprise Architect and create a new empty project.
3. Go to the *Specialize* menu tab and click on *Publish Technology* → *Import MDG Technology*.
4. Select and open the downloaded file containing the MDG Technology. You can choose the radio button down left to only import it to the current project or make it available for all project the currently logged in user creates.
5. Click on the *OK* button.
6. Select the downloaded MDG Technology file.
7. The Cybersecurity DSL should now be available in the *Specialize* → *Manage Technology* list. You possibly need to restart Enterprise Architect to apply all changes. In this menu you can enable or disable it.
8. When you create a new diagram, you can now select a type of diagram from the Cybersecurity DSL giving you access to use the Cybersecurity DSL elements.

To make it easy for you to create a new model structure, you can use the provided template (*BasicCybersecurityAnalysisStructure.xmi*). The template contains a predefined structure for the Cybersecurity DSL. To use the template, follow these steps:

1. Download the template file from the dsse.at-website (*BasicCybersecurityAnalysisStructure.xmi*) and save it somewhere on your local file system.
2. Select a location in your project browser where you want to import the template structure.
3. Got to the menu tab *Publish* → *Import Package*→ *Import Package from Native/XMI File*.
4. Select the downloaded template file, check the box *Strip GUIDs* and click on the *Import* button.
5. After a short time, the template structure should be imported into your project and visible in the project browser for you to work on.

This process will be made more convenient in the future by providing a msi-based installer file. Previously mentioned features like additional user-guidance, model validation, and automation features will be contained there.

6 Language Reference

Following tables contain all elements and relationships part of the Cybersecurity DSL. It is an extract of the metamodel (Figure 8) of the DSL and is intended to be used as a reference for the user viewing or modifying a cybersecurity analysis model.

Table 1: Language reference table for the *Cybersecurity DSL*

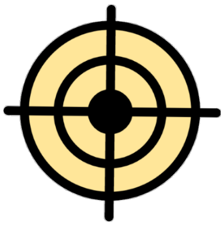
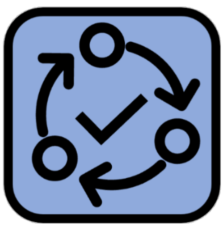
Language Element	Semantics Description
 Target Of Evaluation	The "Target of Evaluation" (TOE) serves as the entry point for conducting Cybersecurity Analysis. It is strongly related to a single technical component (physical or software) of a system architecture model. The TOE itself has to fulfill prescribed Cybersecurity Requirements. Furthermore, progressing further in Cybersecurity Analysis, a TOE embraces one or more Cybersecurity Assets that have their working integrity protected to enable normal functioning of the TOE.
 Cybersecurity Asset	A "Cybersecurity Asset" represents an abstracted feature or functionality that is inherently part of a TOE, which needs to be protected by security measures to keep the TOE working as intended. This element strongly relates to the "cybersecurity property" in ISO 21434, which is generally defined as an attribute that can be worth protecting. Its impact on systems' health can be rated via the CIA triad (confidentiality, integrity, availability).
 Uncontrolled Threat Scenario	An "Uncontrolled Threat Scenario" is based on a Threat Scenario. It is the initial Threat Scenario for conducting the Threat Analysis, used to identify vulnerabilities and depict emerging risks.
 Controlled Threat Scenario	A "Controlled Threat Scenario" is based on a Threat Scenario. It is a modified version of the initially existing Uncontrolled Threat Scenario, used to identify vulnerabilities and depict emerging risks. The Controlled Threat Scenario is therefore heavily influenced by applied Countermeasures and depicts how the treatment of vulnerabilities can help to reduce one or more Risks.

Table 1: Language reference table for the *Cybersecurity DSL*

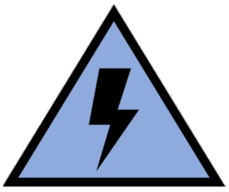
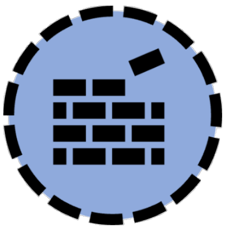
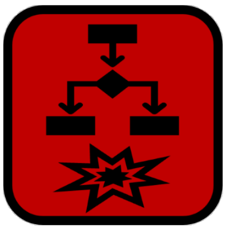
Language Element	Semantics Description
 Threat Actor	Threat Actors can be internal or external to the organization, and they may have a variety of motivations for attacking, such as financial gain, data theft, or simply causing damage.
 Hazard	A "Hazard" is a type of Threat Agent, which is not itself considered as an individual or a group of persons. Environmental impact factors (weather, car accident, corrosion) are typical hazards. Other examples may be faulty software or insufficiently approved physical components. In contrast to a Threat Actor, a Hazard does not have the intrinsic intention to sabotage the system or gain personal advantages by system abuse.
 Vulnerability	"Vulnerability" in a system refers to a weakness or flaw that could be exploited by attackers to compromise the security of the system. It's like a gap or a soft spot in the system's defenses that, if not addressed, could allow unauthorized access, data breaches, or other malicious activities. Identifying vulnerabilities is crucial for cybersecurity because it helps in understanding where the system might be at risk.
 Risk	A (cybersecurity) Risk is defined in ISO 21434 as the potential for a cybersecurity incident to occur, which may lead to failures and damages on the system and its environment. Its importance to implement countermeasures is assessed by multiplying the likelihood and the damage potential of that incident.
 Damage Scenario	A "Damage Scenario" contains the chain of action of negative impacts of one or more Threat Scenarios, that lead to some damage to the system or its components. Analytic procedures like FMEA (Failure Mode and Effects Analysis) can be applied to this part of the Impact Analysis to methodologically analyze possible system failures.

Table 1: Language reference table for the *Cybersecurity DSL*

Language Element	Semantics Description
 Damage	"Damage" refers to the harm that can happen either to the system itself or to the people who own or use the system. It's like the negative impact or bad things that can happen when there's a cybersecurity issue. This harm could include things like the system not working properly, data getting lost, manipulated, or stolen, or even harm to the people using the system.
 Risk Mitigation	"Risk Mitigation" focuses on reducing the likelihood or impact of a risk through proactive measures and controls. Organizations implement strategies to address vulnerabilities and minimize the overall risk exposure.
 Risk Elimination	Risk Elimination is the strategic choice to circumvent exposure to a particular risk altogether. Organizations steer clear of activities or situations that could lead to the identified risk, aiming to prevent potential harm or adverse outcomes.
 Risk Acceptance	"Risk Acceptance" is not a Countermeasure but is a valid Risk Treatment Decision. It involves acknowledging a specific risk without actively attempting to alter its likelihood or impact. Organizations consciously decide to tolerate the potential consequences, often when the cost of implementing Countermeasures outweighs the expected loss.
 Risk Transfer	"Risk Transfer" involves shifting the financial burden of a risk to another party (at the business level) or other system components. This approach aims to protect systems assets by passing on the responsibility for potential losses to an external entity like an insurance or other system components which produce thereby a lower risk.

Table 1: Language reference table for the *Cybersecurity DSL*

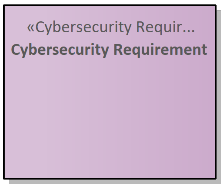
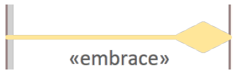
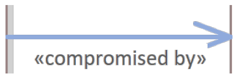
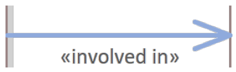
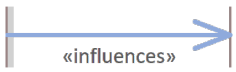
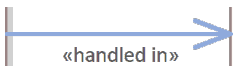
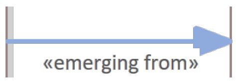
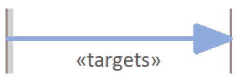
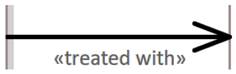
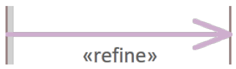
Language Element	Semantics Description
	A "Cybersecurity Requirement" represents a specific, actionable statement that translates one or more high-level cybersecurity goals into practical directives. It serves as a concrete guideline for the implementation of countermeasures to handle related Risks. Cybersecurity goals, as defined by ISO/SAE 21434, represent high-level statements that outline the desired security outcomes for an automotive system or component. They are formulated during the early stages of product development, typically as a result of the Threat and Risk Assessment (TARA) process. Cybersecurity goals serve as guiding principles for the implementation of security controls and countermeasures throughout the development lifecycle. Therefore, a cybersecurity requirement is a measurable need, or capability that must be satisfied by a system, or system element to ensure its overall intact security. In contrast to a cybersecurity goal, it must follow the 4C-rule of requirements (Complete, Correct, Concise, Confirmable).
	The relation "embrace" indicates, that the element at the target of this relation is a part of the element at the source in an abstracted way. So the source element needs for example the target element as a property to ensure the desired way of functioning.
	The relation "compromised by" indicates, that the element at the target of this relation has the possibility to harm some features of the element positioned at the source of this connector.
	The relation "involved in" indicates, that the element at the source of this relation performs some action or passively participates in or at the target element.
	The relation "influences" indicates, that the element at the source of this relation is designed to modify the behavior of the target element.
	The relation "handled in" indicates, that the negative influence of the element at the source of this relation is to be considered at special focus in or at the target element.
	The relation "emerging from" indicates, that the undesirable existence of the target element of this relation has come up, because of some structural or behavioral properties of the source element.
	The relation "targets" indicates, that the element at the source of this relation aims to compensate some negative aspect of the target element, which badly influences the system or a part of it.
	The relation "exposed to" indicates, that the element at the source of this relation is potentially negatively influenced by the target element. This is because the target element has one or more direct ways of correspondence to the source element.

Table 1: Language reference table for the *Cybersecurity DSL*

Language Element	Semantics Description
	The relation "leads to" indicates, that the element at the source of this relation is the reason why the element at the target of the relation comes into existence.
	The relation "observe" indicates, that the element at the source of this relation depends from one or more tasks in the context to measure parameters related to the target element.
	The relation "treated with" indicates, that the target element tries to either explicitly accept or compensate one or more disadvantages of the source element.
	The relation "refine" indicates, that the element at the target of this relation was created to specify, decompose and define in more detail the usually less strictly defined source element and its parameters. This procedure can involve the task to identify and determine measurable parameters used to prove or disprove that some goals are reached.
	The relation "fulfill" in the context of the Cybersecurity DSL expresses that a TOE is dependent from and must comply to prescribed Cybersecurity Requirements. To fulfill a Cybersecurity Requirement, one or many TOEs may need to implement security mechanisms to fulfill the requirement.

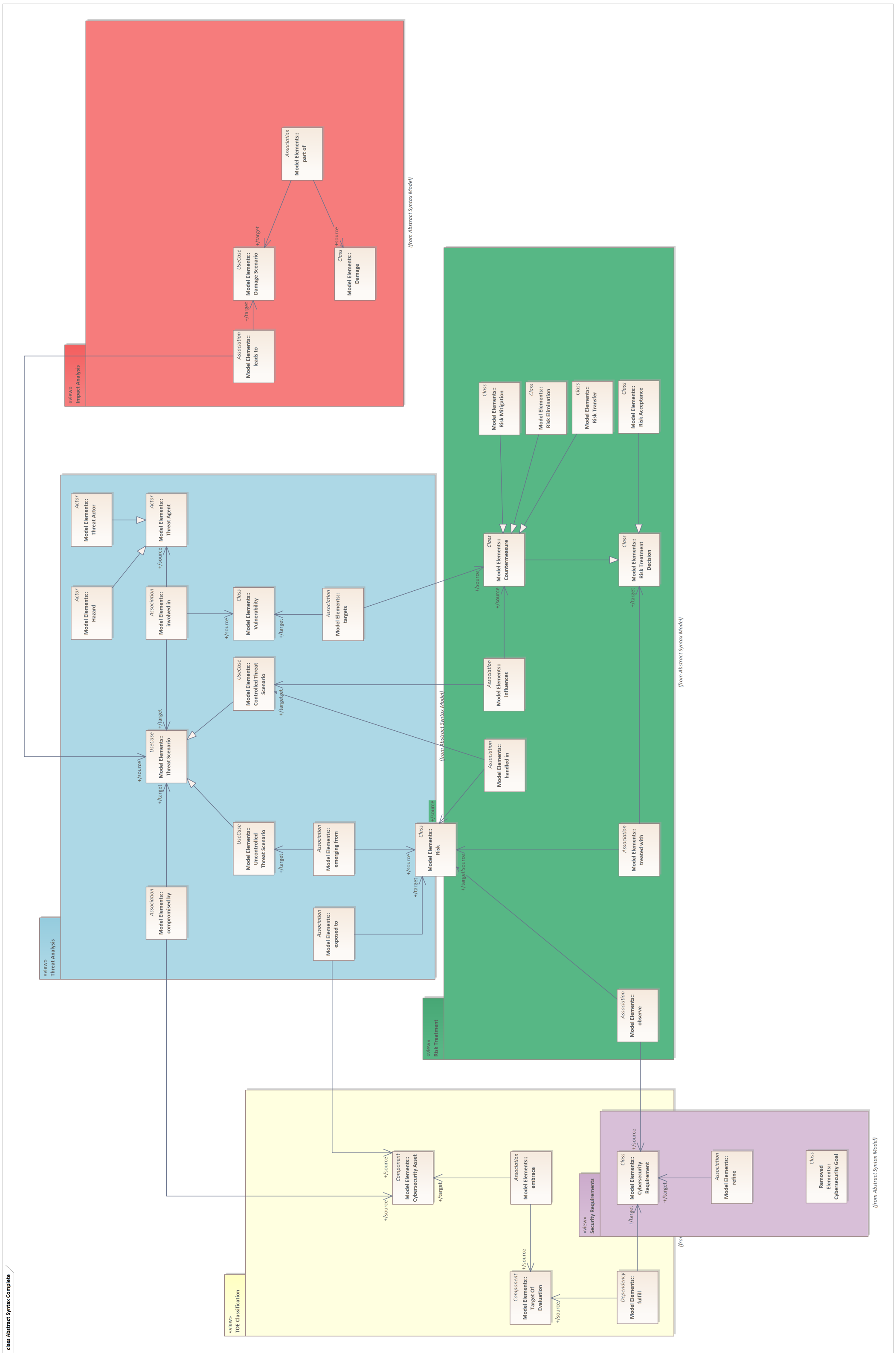


Figure 8: MOF-conformant Abstract Syntax Mode of the Cybersecurity DSL