Model-Based Information Security Management Systems Architecture

Volodymyr Mokhor[1], Oleksandr Bakalynskyi[1], Iaroslav Dorohyi[2,3], and Vasyl Tsurkan[1,3]

[1] G. E. Pukhov Institute for Modeling in Energy Engineering of National Academy of Sciences of Ukraine, 15, General Naumov Str., Kyiv, 03164, Ukraine
[2] Donetsk National Technical University, 56, Potebni Str., Lutsk, 43003, Ukraine
[3] National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Prospect Beresteiskyi, Kyiv, 03056, Ukraine

Abstract

The representation of the architecture of an information security management system has been investigated. The peculiarities of how the organization, as an environment, influences its key concepts and properties have been demonstrated. This influence is reflected in the elements of connections, principles, and the evolution of the information security management system. Describing its architecture within a specific organization—considering the type, size, and nature of the organization—has been reduced to constructing a model from the perspective of stakeholders. To achieve this, it is proposed to use systems engineering based on a model, preceded by an analysis of its features and advantages. Secure system development patterns and extensions of the Unified Modeling Language, including UMLsec and SysML, have been discussed. Among these, SysML has been highlighted, and its practical application has been analyzed. Additionally, relational probabilistic models have been researched as alternatives. Further stereotypes of the SysML language have been identified.

Moreover, a method for presenting and justifying the design of information security management systems, referred to as REISMSAD, has been proposed. Its use is oriented toward developing and documenting architectural decisions, as well as justifying the project's development within the organization. This approach has enabled the formulation of tasks for verifying the architectural decisions of information security management systems. As a result, it has become possible to apply formal methods based on labeled transition systems to solve these tasks. Consequently, achieving a unified representation of the architecture model of the information security management system has become feasible, primarily from the perspectives of various stakeholders. The language for modeling systems serves as the basis for such formalization. According to its graphical notation, an architecture element is represented as a block, along with its properties and the relationships between them.

Keywords

Risk assessment, risk treatment, information security management system architecture, model-based systems engineering, model-based architecture, interested party.

1. Introduction

The embodiment of the main concepts and properties of a system within its environment is reflected in its elements, connections, principles, and evolution [1]. Such reflection is conventionally represented by its architecture. The description of its creation within a specific environment and/or community of stakeholders is determined by a model constructed from a particular perspective [2].

A characteristic feature of creating information security management systems is the focus on ensuring the integrity of fundamental information properties within an organization. This is achieved through risk assessment, which guarantees stakeholders proper handling of these risks [3]. In this context, the organization is interpreted as the surrounding environment that defines the parameters and conditions influencing the information security management system. Meanwhile, its main concepts and properties are embodied in the elements and connections of the architecture [1, 4, 5].

ITS-2023: Information Technologies and Security, November 30, 2023, Kyiv, Ukraine
v.mokhor@gmail.com (V. Mokhor); baov@meta.ua (O. Bakalynskyi); argusyk@gmail.com (Ia. Dorohyi); v.v.tsurkan@gmail.com (V. Tsurkan)
0000-0001-5419-9332 (V. Mokhor); 0000-0001-9712-2036 (O. Bakalynskyi); 0000-0003-3848-9852 (Ia. Dorohyi); 0000-0003-1352-042X (V. Tsurkan)

With this interpretation, it becomes possible to establish probable influences on the information security management system from the organization's perspective, taking into account its type, size, and nature [4]. Additionally, the model considers the existence and interaction with other management systems, such as cybersecurity and private information security. This is crucial for addressing the needs, expectations, and limitations of stakeholders within defined boundaries (organization and organizational units). Therefore, a model-based representation of the architecture of information security management systems is highly relevant.

2. Related Work

Engineering systems with security in mind has several advantages. Systems with added security are evidently less prone to failures and breaches caused by malicious actors. The international standard ISO/IEC 21827 [5] defines additional advantages for engineering organizations that prioritize security, such as "savings from the absence of rework due to repeatability, predictable processes, and practices; recognition of the true capability to perform tasks, especially in selecting sources; and an emphasis on measured organizational competence (maturity) and improvement."

Considering the advantages in reliability and cost, there is a significant body of work in the field of security-focused systems engineering and software engineering. Several international standards relate to secure software/system engineering. As mentioned, ISO/IEC 21827 pertains to this field, providing motivation for security requirements in systems and defining ways to assess an organization's procedures for developing such systems. However, its primary focus lies in determining an organization's maturity regarding established processes, such as ensuring information security and cybersecurity. This standard is complemented by guidelines [4, 6, 7] that offer clear instructions on forming continuous and cohesive information security policies throughout the organization. Key components are formally defined, namely: confidentiality, integrity, availability, non-repudiation, and authenticity.

The Carnegie Mellon Software Engineering Institute published several "Secure Design Patterns" through its CERT program [8]. Designed for maximum reusability in various design scenarios, these high-level patterns provide strategies for use in system architecture, modular design, and implementation, all with a security orientation. Each pattern is inspired by a specific vulnerability, such as privilege escalation (a common method for executing malicious code or accessing resources without proper permission) or input validation flaws (a common method for injecting SQL commands through an interface to a database). Each pattern consists of predefined elements that define the circumstances under which the design pattern can be used, including entities and code snippets for application.

By analyzing information flow and system usage, it is possible to identify functionalities and interactions that correspond to anticipated design patterns and create solutions that help avoid common issues. With a plethora of standards and guidelines focused on secure architectures and design patterns that promote secure development, it is evident that security is a multifaceted, systemic issue. UMLsec is an extension of UML aimed solely at software development. This solution attempts to incorporate security into UML during the development stage [9]. In relevant works, Jürjens provides a comprehensive analysis of several standard UML diagrams that can supplement the security analysis of software systems [10]. Regarding the physical elements of a system, he relies on the deployment diagram to view the physical connections between devices.

While UMLsec is a useful step toward building secure system architectures, it is more beneficial to consider a systems view, supported by SysML. AVATAR is an environment based on SysML, developed within the European project EVITA, a consortium of security, academic, and automotive industry partners researching secure networks on automotive platforms [11]. AVATAR extends SysML through several stereotypes and security-based functions, including the use of temporary functions in standard state diagrams, allowing for the modeling of delays between states and minimal

processing time [12]. Other extensions are designed for using traditional cryptographic algorithms as part of the model. A formal proof language is then employed to verify that the security requirements of the system are met. While this represents an important advancement, the techniques proposed by the authors primarily focus on the cryptographic aspects of information security, with less detailed exploration of system aspects such as architectural vulnerabilities and access control. The central emphasis on encryption makes it unrealistic to detect vulnerabilities beyond confidentiality breaches using AVATAR in its current form.

Additional attempts to explicitly model security systems within SysML and UML are proposed, as seen in [13, 14]. Most of these approaches modify existing diagrams to allow for an explicit description of security requirements; however, they offer little in terms of automatically extracting relevant information from an existing model to identify security vulnerabilities.

An alternative to building on UML or SysML models is the use of Probabilistic Relational Models (PRMs). PRMs are similar to class diagrams in UML but utilize probabilistic attributes (both quantitative and qualitative) to represent the probabilistic characteristics of classes and the relationships between attributes. In [15], a PRM-based language for modeling architectures at the organizational level is proposed (CySeMoL). The probabilistic inference mechanism generates attack probabilities (i.e., the probability that an attack will be executed and successfully completed) based on data from previous attacks. The advantage of this technique is that system architects can reuse components and inherit previously stored attack vector data. Additionally, all known types of attacks, including denial of service, are captured, providing a comprehensive overview of the attack landscape.

3. SysML extension for modeling security systems

SysML is a powerful modeling language that can provide significant benefits for developers concerned with security. First and foremost, SysML offers a standardized way for system engineers and security experts to view systems of interest, helping to bridge the gap between these disciplines.

Models are typically created as part of the system lifecycle or, in the case of deployed systems, constructed retroactively for documentation purposes. Therefore, using models as part of the security assurance process aligns well with the typical use of SysML, resulting in minimal additional modeling overhead.

Since models are often regarded as “living documents” that are maintained to reflect design architecture choices and modifications, they serve as valuable sources of information for information security professionals who require an accurate, up-to-date representation of the system for its proper protection.

As previously discussed, SysML, in its original form, includes many built-in provisions for stakeholders involved in security assurance. However, the idea of adding new features to extensible modeling languages to better address security concerns is not without precedent, and SysML supports such capabilities. For systems engineers, security is a primary objective when reviewing model data. By applying the paradigm of separating concerns, systems engineers can specifically address security issues as a key part of the system lifecycle, including information security management systems.

3.1. Developing a Threat Model

To determine what information is valuable from a security standpoint, it is essential to represent concepts that are significant within the context of SysML. Figure 1 illustrates the relationship between threat and risk [5].

This model provides a high-level description of risks related to information security. It defines how threats exploit vulnerabilities in information assets. Countermeasures specific to these vulnerabilities help minimize the risk to information assets. These relationships aid in identifying the

connections between threats, vulnerabilities, countermeasures, risk, and information assets. Furthermore, SysML's holistic view of the system allows for the expansion of these relationships to encompass a broader range of concepts. Figure 2 presents a more comprehensive threat model, intended to serve as the foundation for the security view within SysML.

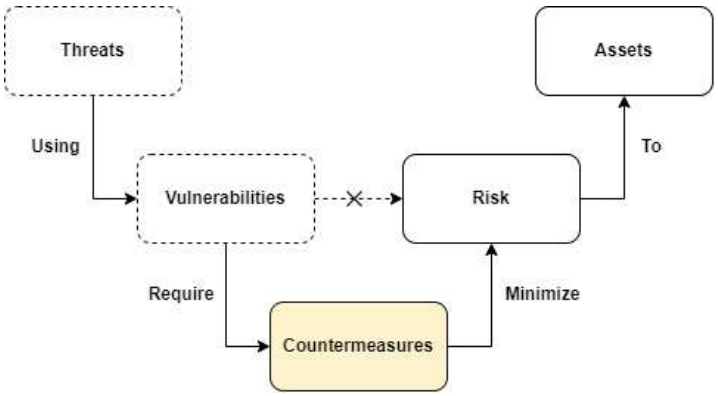

Figure 1: Relationship between "Threat-Risk" [5]

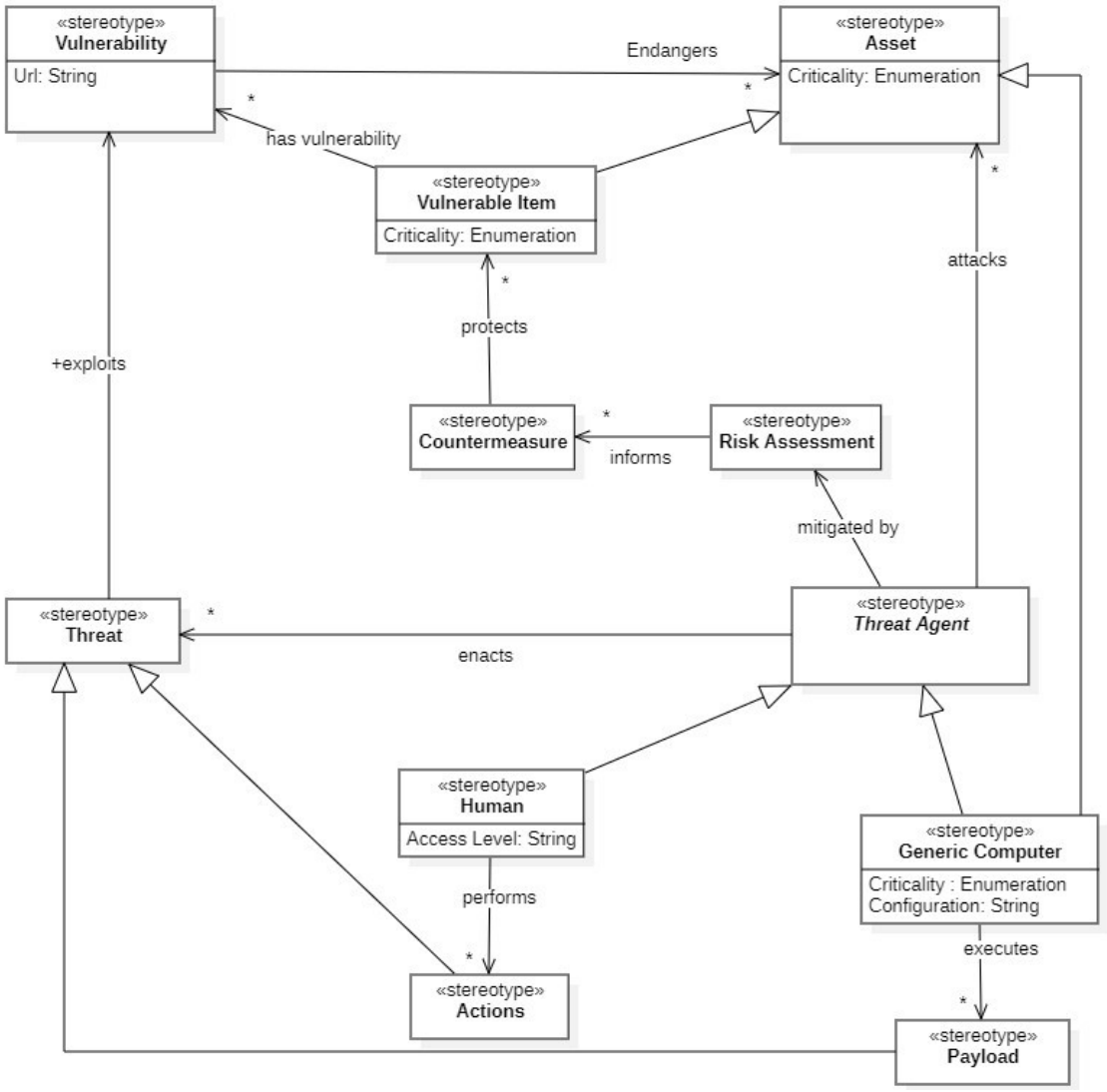

Figure 2: Threat Agent Profile Model

Figure 2 presents a series of interconnected stereotypes that can be applied to blocks when modeling information security management systems. In addition to the concepts presented in Figure 1, the SysML Threat Agent Profile describes how threats result from threat agents and reveals the information assets that are vulnerable due to the exploitation of vulnerabilities by known threats.

Threat agents can be either human or computational devices (such as tablets or laptops) infected with malicious software and capable of transmitting additional malicious payloads. Human threats refer to actions (such as inserting a USB drive into a computer or deliberately disabling security systems), while computational threats are modeled as actions that spread malicious software packages or initiate more general threats (such as sending malicious commands). Computational devices serve as both sources of threats and information assets that need to be protected due to the potential for cascading attacks with multiple malicious payloads.

The model also illustrates how countermeasures are defined based on risk assessments of specific threat agents, thereby closing the loop from threat agent to countermeasure.

3.2. Developing a Data Model Profile

Although the threat agent profile provides information on how threats interact with information assets and vulnerable elements, it lacks a model of the interaction between assets and data. Without considering data relationships from an information security perspective, the model remains incomplete. Figure 3 presents a graphical representation of the data model, illustrating the relationships between information assets, data storage devices, and communication protocols.

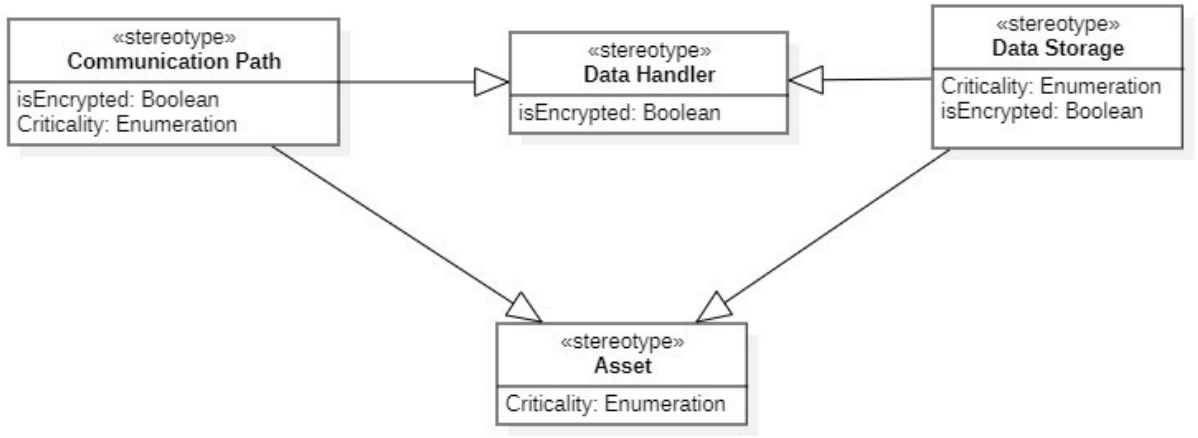

Figure 3: Data Model Profile

In its current state, the information in the data model regarding communication protocols is limited, as it does not capture the layered nature of these protocols. In this work, the "Communication Path" stereotype should be considered a model of the physical connections between devices, or, in the case of wireless networks, an acknowledgment that data can circulate between connected assets.

3.3. Extension of the modeling tool

With the corresponding profiles contained in the system, the following stereotypes can be applied to elements with existing models:

1. *Asset*: defines any resource that can be the target of an attack. The "criticality" representation of the asset is maintained so that different information assets can be prioritized by importance within the information security management system.
2. *Communication Path*: this stereotype can be applied to any relationship between model elements and any block representing a communication protocol, such as TCP/IP. It provides

the ability to represent all communication paths, whether encrypted or not, and assess their criticality to the information security management system.
3. *Data Storage*: data storage devices are information assets and data processors, so they can indicate whether data is stored, encrypted, and what its criticality is.
4. *Vulnerability*: applied to classes. These classes capture the URLs of known vulnerability reports and allow them to be associated with specific vulnerable elements (information assets).
5. *Vulnerable Item*: the vulnerable item is stored in the threat model as a type of information asset. It should have a countermeasure that protects it and is associated with the vulnerability class.

3.4. View on Information Security

Adding threat and data models to SysML allows for the inclusion of new information in the model and creates a security view—specifically, a collection of visual representations of the system that displays this new information. The security view should address the key security issues as defined in [4, 6, 7].

3.4.1. Confidentiality

Confidentiality is defined as "the property that information is not made available or disclosed to unauthorized individuals, entities, or processes" [6, 7]. To consider confidentiality, it is essential to take into account how information is communicated. The "Communication Path" stereotype requires that all communication paths be marked as encrypted or not. It is important to note that by using a simple logical flag, a systems engineer can declare channels as encrypted, leaving the details about encryption techniques to be developed in a more detailed specification later. Currently, there is no formal representation of encrypted data in the model. Defining data as another type of information asset will allow it to be included in the threat model. However, this approach will not apply the "isEncrypted" flag, as it is a concept that does not apply to all types of assets [4, 6, 7].

3.4.2. Integrity

Integrity is defined as "the property of ensuring the accuracy and completeness of information assets" [6, 7]. Using SysML adds additional meaning to "integrity," as a security view will not benefit from continuous security monitoring if the model's integrity is compromised. Implementing checksums for software and data allows for periodic automated checks to ensure that the current configuration has maintained its integrity. Recording checksums for software and data in the toolchain is feasible since updates are typically less frequent than those in standard IT systems and should always be executed under a controlled deployment strategy. Such a strategy should include updates to the system models to reflect changes [4, 6, 7].

3.4.3. Availability

Availability is defined as "the property of being accessible and usable upon demand by an authorized entity" [5–7]. In the least harmful case, attacks compromising the availability of information assets may deprive users of access to remote services or processes, incurring significant financial costs. In the worst-case scenario, a service whose access is denied could involve an internal control mechanism that, if denied for a sufficiently prolonged period, might lose control over a critical system or service. Therefore, it is important not only to check interfaces for potential denial-of-service sources affecting users but also to work backward from critical information assets to identify connections with threats that could "flood" the controller with messages, ensuring the implementation of sensitive message processing procedures. Most of the information necessary for evaluation is included in the original

SysML diagram. However, additional stereotyped delineation of ports on vulnerable elements allows querying the tool to identify all ports not associated with mitigation actions against threats. In [16], the availability of management systems used in the energy sector is investigated through interviews with enterprise employees. The systems considered have SCADA components, disconnection management, and distribution management. The study showed that many disconnections were related to software issues on both primary and backup systems, as well as the underutilization of hardware redundancy specifically for such cases. This has implications for security; if failover recovery techniques are not applied (where the backup system has a completely different design from the primary), vulnerabilities found in the software of one system will also exist in the other.

3.4.4. Non-repudiation

Non-repudiation is defined as "the ability to prove that an action or event occurred, so that this event or action cannot be repudiated later" [5]. Indeed, actors in the information security management system can trigger events by sending explicit commands or by reporting on variables associated with predicted triggers. To ensure non-repudiation, an important factor is accurate data logging, which indicates who executed a command and when. This underscores the significance of data integrity, as malicious actors might attempt to deliberately edit logs to compromise the system's non-repudiation. Additionally, data recorded in logs is of little value if the source of the command is unidentified.

3.4.5. Authenticity

Authenticity is defined as "the property that ensures the identity of the information asset matches the claimed identity. Authenticity relates to entities such as users, processes, systems, and information" [5]. Authenticity is characterized by two types of interactions: direct (through device authentication) and indirect (through the role of authentication in the authorization process). Authentication is an extremely complex issue because, for security-critical systems, the need for authentication can introduce an unacceptable delay between the detection of an event and the response to it. Moreover, actions related to emergency management (e.g., emergency shutdowns) often involve the same actions that a malicious user might want to exploit. Establishing a policy that relaxes security during emergencies is dangerous, as it creates an additional incentive for attackers to induce failures and may encourage legitimate users to perform potentially hazardous actions to bypass security measures. Additionally, encryption and authentication increase the bandwidth requirements of the communication flow, which is unacceptable for real-time management.

After successful user authentication, one can proceed to a separate authorization process. People, as threat objects, should be associated with the access level they pose to the system, making risk assessment more realistic. However, it is challenging to identify other elements within the threat model that should be associated with security levels, as specific implementations of the authentication process go beyond the scope of the high-level models considered in this work. The "Asset" stereotype is not directly linked to security levels, as a single device may have multiple modes of functionality, each with its own access level. For example, parameterization typically has a higher security level than simple on/off commands.

4. Model-based information security management systems architecture

The precondition for defining the architecture of an information security management system is the analysis of the organization's activities as the surrounding environment. Based on established internal and external circumstances, requirements are formulated by stakeholders. Implementing these requirements allows for the consideration and, ultimately, the satisfaction of defined needs,

expectations, and limitations. Consequently, transforming stakeholders' perspectives into a vision for a technical solution is essential. This transformation is crucial for presenting the characteristics, attributes, and both functional and non-functional requirements of the information security management system within the organization [1, 2, 4].

Considering the interests of stakeholders and the established functional and non-functional requirements is achieved by creating alternative variants of information security management system architectures. Each architecture defines elements and relationships between them, as well as interfaces for interacting with other systems within the organization and with other organizations (or structural parts of the same organization). Additionally, systems for managing cyber and information security should consider their dependency on quality management systems, as these generally form an integrated organizational management system [2, 4]. Interfaces and interactions characterize the boundaries of the system's creation and implementation. These aspects are taken into account through the development of a model of the architecture of the information security management system. To address this task, the use of system engineering based on a model is proposed [17].

The proposed methodology allows for a shift away from a document-based approach to building a model of the architecture of the information security management system. This approach achieves uniformity in its representation, primarily from the perspectives of different stakeholders. Such uniformity is important for both a clear understanding and the effective meeting of their needs, expectations, and limitations. The foundation of this formalization is the SysML modeling language, extended with additional security stereotypes, as previously mentioned, and an additional set of stereotypes that describe the architecture of the security system according to [1], which will be presented below. In graphical notation, an element of the architecture of the information security management system is represented as a block, along with its properties and relationships to other elements. Furthermore, the structure of each block can be further detailed through its ports and interfaces. This capability is useful for depicting interactions both within the elements of the architecture of the information security management system and with the surrounding environment.

To transform the model views of the architecture according to the ISO 42010 standard [1], additional stereotypes of the SysML language should be created. The following stereotypes will be utilized:

1. *Architecture Description*: Provides a description of the architecture.
2. *Architecture*: Represents the architecture as an entity.
3. *Stakeholder*: Describes the role, position, person, or organization that has an interest, right, share, or requirement regarding the "Entity of Interest."
4. *Stakeholder Perspective*: Describes the stakeholder's perspective on the "Entity of Interest" within its "Concern."
5. *Entity of Interest*: Describes the subject of the architecture description.
6. *Environment*: Describes the context, conditions, and impact on the "Entity of Interest."
7. *Concern*: Describes the stakeholder's concern that is important to them.
8. *Aspect*: Describes a specific part of the character or nature of the "Entity of Interest."
9. *Architecture View*: Describes a specific informational component that is part of the "Architecture Description."
10. *Architecture Viewpoint*: Describes a set of conventions, interpretations, and uses of the "Architecture View" to highlight one or more "Concerns."
11. *Model Kind*: Describes the model category, distinguished by its key characteristics and modeling conventions.
12. *View Component*: Describes a specific part of one or more "Architecture Views."
13. *Model-Based View Component*: Describes a specific part of one or more "Architecture Views" that is governed or formulated by a specific "Model Kind."

14. *Non-model Based View Component*: Describes a specific part of one or more "Architecture Views" that is governed or formulated without using models.

Taking these into account, the architecture of the information security management system can be represented by the following model in SysML graphical notation (Figure 4).

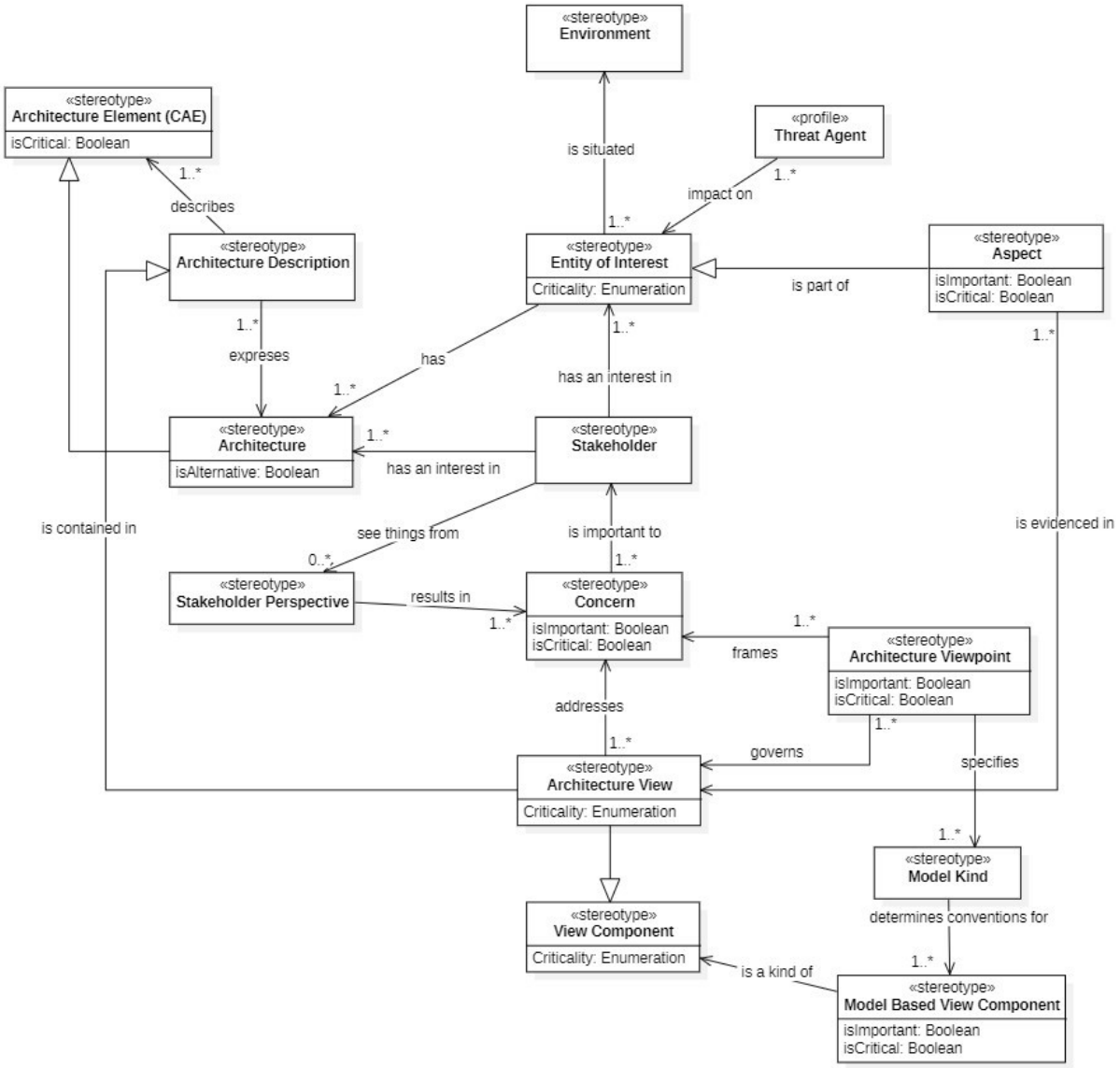

Figure 4: Model of the information security management system architecture

4.1. The method of representing the architecture of the information security management system

The method of representing the architecture of the Information Security Management System (ISMS) that we will discuss is called the Rationale and Evaluation of Information Security Management System Architecture Design (REISMSAD). REISMSAD aims to assist architects in creating and documenting architectural designs with a focus on architectural decisions and project justifications. It encompasses three types of architectural knowledge: design issues (represented by the "Stakeholder" element), design decisions (represented by the "Architecture Description" element), and project outcomes (represented by the "Architecture" element). These knowledge objects are represented by standard SysML objects.

The design issue refers to the set of materials that influence the designer's or project's decisions. This entity encapsulates concepts such as functional requirements (e.g., scenarios), non-functional requirements (e.g., all quality attributes), and project contexts. It also captures information about design decisions and project justifications. Project outcomes include the decisions made, such as classes, components, interfaces, and usage scenarios. Each individual type of architectural knowledge object is captured by applying a predefined tag template of the stereotype.

In essence, REISMSAD is represented as a directed acyclic graph that connects the architecture elements (CAE – "Architecture Element" elements) with the architecture rationale elements (AR – "Architecture Description" elements) using the directed relationship ARtr. AR encapsulates the "Architecture Description" result for the "Architecture." Since AR has a 1:1 relationship with the "Architecture," it serves as the decision point for justifying the architecture. The relationship between CAE and AR is represented by the directed association ARtr, which signifies a cause-and-effect relationship.

Definition 1. The REISMSAD Model is a labeled transitive system, where:

- *CAE* is the set of nodes representing critical and non-critical elements of the architecture;
- CAE^0 is a subset of initial states of architecture elements;
- *AR* is the set of nodes representing architecture rationale;
- A is a finite set of action labels;
- Σ is the set of system variables;
- L is the labeling function $L: CAE \rightarrow 2^{\Sigma}$ for elements;
- R is the set $R \subseteq (CAE \times AR) \cup (AR \times CAE)$ of directed links between nodes, which satisfies the following conditions:
    1. All nodes in AR must be associated with at least one cause and one effect, meaning there exists $\forall r \in AR$ a cause $e \in CAE$ such that $(e, r) \in R$ and there is an effect $e' \in CAE$, where $(r, e') \in R$.
    2. There is no subset of R that forms a directed cycle.

The primary form of the model construct is a path of the form $\{CAE_1, CAE_2, \dots\} \rightarrow AR_1 \rightarrow \{CAE_a, CAE_b, \dots\}$, where CAE_1, CAE_2 are inputs or causes of the decision AR_1 and CAE_a, CAE_b are outputs or consequences of this decision. Figure 5 illustrates a diagram demonstrating this defined model construct. The multiplicity of relationships in the diagram shows that the motivational and resulting sets *CAE* are non-empty sets connected through a single element *AR*. The uniqueness constraint on the diagram specifies that each instance of *CAE* cannot be depicted more than once.

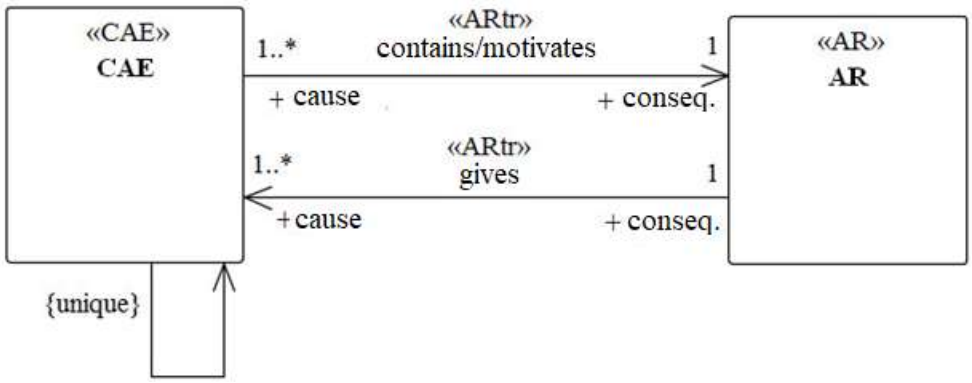

Figure 5: Cause-Effect Relationship between CAE and AR

Directed links ARtr represent cause-and-effect relationships. CAE leads to AR through the motivation or constraints of the "Architecture Description," which in turn generates CAE of type "Architecture" having "Architecture Description." CAE can function as both an input and an output when used for two decisions. As an input, it can represent artifacts of the following types:

requirement, precedent, class, and implementation. As an output, it can be a new or revised design element.

In REISMSAD, architectural elements (CAE) are artifacts that form parts of the architecture design. They include needs that must be satisfied, technical and organizational constraints imposed on the architecture project, assumptions to be verified, and design objects that result from architectural design.

Architectural elements can also be classified from different architectural perspectives. This classification focuses on various aspects of the project solution. The following perspectives on architecture are used for classification: logic, data level, application, technologies, and security.

It is assumed that the architecture view includes requirements and environmental factors. Five categories of architectural views have been created, describing different aspects of influence on architectural design. System requirements include both functional and non-functional requirements. This classification allows the architect to trace the process of justifying decisions back to specific classes of root causes during analysis.

"Architecture" elements are the results of the design process and can be classified according to the following architectural perspectives:

- Data-based – used by applications.
- Application-based – processing logic and software structure, critical and non-critical services.
- Technology-based – technologies and environments used for system implementation and deployment, critical and non-critical components.
- Security-based – security profiles, risk management, critical and important components.

The REISMSAD approach allows for three types of architectural justifications: quantitative, qualitative, and alternative architecture. Qualitative justification represents the justification process and arguments in textual form, essentially outlining the pros and cons of each project decision. Quantitative justification utilizes various criteria to evaluate project decisions. The third type involves documenting and storing discarded alternative project decisions, which can be reviewed to assess the sufficiency of existing evaluation parameters for current architectural projects and for future use in other projects.

It should be noted that architectural decisions can evolve over time due to changes in organizational processes and shifts in the environment itself. As these decisions evolve, the original architectural design and the description of the decision-making process for this project may be lost. Therefore, it is necessary to preserve the entire history of the project's evolution. To address this need, an extended REISMSAD model is proposed.

Definition 2. The Extended REISMSAD Model is a labeled transitive system

$$REISMSADe = (CAE, CAE^0, AR, A, R, \Sigma, L),$$

with a bijective mapping function SSf: $(CAE \to CAE_h) \cup (AR \to AR_h)$ for each architectural element or descriptions of architecture, where:

$$REISMSAD_{cur} = (CAE_{cur}, CAE^0_{cur}, AR_{cur}, A_{cur}, R_{cur}, \Sigma_{cur}, L_{cur})$$

current model of architecture, and the following conditions are satisfied:

$CAE_{cur} \subseteq CAE$;

$AR_{cur} \subseteq AR$;

$A_{cur} \subseteq A$;

$\Sigma_{cur} \subseteq \Sigma$;

$CAE^0_{cur} \subseteq CAE^0$;

$L_{cur}: CAE_{cur} \to 2^{\Sigma_{cur}}$;

$R_{cur} \subseteq R \cap \big((CAE_{cur} \times AR_{cur}) \cup (AR_{cur} \times CAE_{cur})\big)$,

there is no subset of R_{cur} that forms a directed cycle;

$REISMSAD_h = (CAE_h,\ CAE_h^o, AR_h, R_h, \Sigma_h, L_h)$, – historical architecture model, and the following conditions hold:

$CAE_h \subseteq CAE$;

$AR_h \subseteq AR$;

$A_h \subseteq A$;

$\Sigma_h \subseteq \Sigma$;

$CAE_h^0 \subseteq CAE^0$;

$L_h: CAE_h \to 2^{\Sigma_h}$;

$R_h \subseteq R \cap \big((CAE_h \times AR_h) \cup (AR_h \times CAE_h)\big)$,

there is no subset of the set R_h that forms a directed cycle;
and for which the following conditions hold:

1. $CAE_h = CAE \setminus CAE_{cur}$.
2. $AR_h = AR \setminus AR_{cur}$.
3. $A_h = AR \setminus A_{cur}$.
4. $\Sigma_h = \Sigma \setminus \Sigma_{cur}$.
5. $CAE_h^0 = CAE^0 \setminus CAE_{cur}^0$.
6. $L_h = L \setminus L_{cur}$.

The corresponding model diagram is presented in Figure 6.

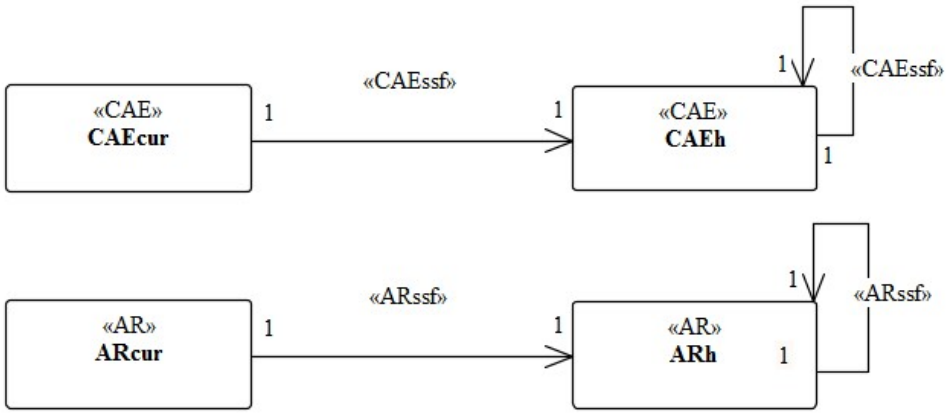

Figure 6: Diagram of the extended model REISMSADe

Definition 3. The model of the architectural solution for the Information Security Management System is defined as a parallel composition of all individual design decisions concerning the systems, components, and processes that constitute the architecture of the ISMS, as specified in the form of LTS REISMSAD, i.e.: $M = REISMSAD_1||_{\Lambda_1}..||_{\Lambda_{n-1}} REISMSAD_n$, де $REISMSAD_1, \dots, REISMSAD_n$ – the corresponding LTS, $\Lambda_1, \dots, \Lambda_{n-1}$ – are defined on the model synchronization pairs.

4.2. Verification of models based on LTS

Methods for model verification (MV) typically specify RS using temporal logic formulas [18]. Temporal logic formulas describe the temporal properties of computations (sequences of transitions) of the model. The most commonly used logics in MV methods are Computational Tree Logic (CTL), Linear Time Logic (LTL), and a combination of both, CTL⋆. An overview of model verification methods is provided in [19].

The task of verifying models of architectural decisions in information security management systems can be formulated as follows:

Task 1. Given a model of architectural decision $M = P_1|| \dots ||P_n$, where $P_1, \dots, P_n$ is the LTS of the REISMSAD system. A formula (specification) φ in temporal logic with respect to the variables of

model M is provided. It is necessary to check the validity of formula φ in model M (denoted as $M \vDash \varphi$).

4.3. Task of verification of parameterized models of architectural decisions in information security management systems.

This work considers a generalization of the model verification problem. Verification methods are applied to architectural models of ISMS represented as SysML diagrams, consisting of a finite set of architectural elements and architectural descriptions specified as Labeled Transition Systems (LTS). Since the set of initial configurations is infinite, the set of architectural decision models with varying numbers of processes is also infinite. Verifying several randomly selected models from this set does not guarantee that the specifications will be fulfilled for all models in the set. For such systems, the task can be generally formulated as follows:

Task 2. Given an infinite family of finite models of architectural decisions in ISMS denoted as $\mathcal{F} = \{M_n\}$, parameterized by a parameter $n \in N$. A formula (specification) φ of temporal logic is given. It is necessary to verify the validity of the formula φ on all models $\mathcal{F}$, that is $M_n \vDash \varphi$, for all n. This task is called parameterized model verification (PMV).

The formulation of Task 2 requires clarification since the general formulation does not explicitly specify how to define the family $\mathcal{F}$ and the specification φ. Therefore, we will consider the following variant of Task 2.

Task 3. Given an infinite family of finite models of architectural decisions in ISMS denoted as $\mathcal{F} = \{M_n\}$, parameterized by a parameter $n \in N$. Each model $M_n = Q||P_1|| \dots ||P_n$ consists of an LTS of a fixed model REISMSAD Q and n instances of LTS of alternative models P_i. $I \subseteq \mathbb{N}$ is a fixed finite set of indices of observed alternative models, instances of prototypes P. The specification φ of temporal logic is given relative to variables defined in the models P_i, $i \in I$ and variables of the model Q. It is necessary to verify the validity of the formula φ on all models in the family $\mathcal{F}$, that is $M_n \vDash \varphi$, for all n.

In fact, in the formulation of Task 3, only one fixed model Q and n instances of prototypes P are used. We can consider a variant of the task where there are several fixed models $Q_1, \dots, Q_m$ and several alternative models $P^a, P^b, \dots, P^z$, and a model

$$M_n = Q_1 \| \dots \| Q_m \left\| P_1^a \right\| \dots \| P_{n_a}^a \dots \left\| P_1^z \right\| \dots \| P_{n_z}^z,$$

where $n_a + \dots + n_z = n$ is the parameter. In some cases, this task can be reduced to the formulation of Task 3 by constructing the LTS of the model Q as a parallel composition of LTS of models $Q_1, \dots, Q_m$, and the LTS of the prototype P as a parallel composition of LTS of prototypes $P^a, P^b, \dots, P^z$.

The proposed formulation of the tasks in the form of (1)–(3) enables the application of a wide range of formalized methods for verifying parameterized models specified as Labeled Transition Systems (LTS). In this work, one of the methods for searching invariants [19] was employed for the verification of the constructed models during the implementation of a unified design system.

5. Conclusion

The main concepts and properties of information security management systems in their environment are reflected by elements, connections, principles, and evolution. The environment can refer to the organization as a whole or to a specific structural part. Depending on its type, size, and nature, its influence is assessed. This representation within a specific environment is characterized by an architectural model of the information security management system, built from the perspective of stakeholders.

To achieve this, the advantages of using systems engineering with security considerations in organizations have been analyzed. Secure design patterns aimed at preventing vulnerabilities from being exploited by threats to the security of information assets have been discussed. Special attention has been paid to extending typical modeling languages, such as UMLsec. Among them, SysML has been highlighted, and the AVATAR environment based on it has been studied as an example. Additionally, the application of relational probabilistic models has been analyzed as an alternative to using UML and SysML.

The appropriateness of using the SysML modeling language as an extension of UML has been justified. In this context, the visualization of the relationship between threat and risk has been considered, leading to an extension with a detailed representation of threat agents. The interaction between information assets and data has also been included, expanding the list of stereotypes (e.g., data repository, communication pathway). Furthermore, perspectives on information security assurance have been identified, with a focus on information properties, primarily complemented by non-repudiation and authenticity. This has resulted in the proposal of additional SysML stereotypes to transform the defined perspectives.

A method for presenting and justifying the design of information security management systems, called REISMSAD, has been proposed. Its application is oriented toward developing and documenting architectural decisions, as well as justifying the project. It encompasses three types of architectural knowledge: design issues (the "Interest" element), design decisions (the "Architecture Description" element), and project design results (the "Architecture" element). The proposed method facilitates justification through quantitative and qualitative aspects, as well as the use of alternative architectures. This has led to the formulation of a task for verifying architectural decisions within information security management systems, allowing for the application of formal methods based on LTS for this purpose.

Therefore, the use of model-based systems engineering enables a shift away from a document-based approach to constructing architectural models of information security management systems. This approach achieves uniformity in representations, primarily from the perspectives of various stakeholders. The SysML modeling language serves as the foundation for such formalization.

Acknowledgements

The authors would like to express their gratitude to colleagues who participated in discussions on the research materials presented at various scientific and technical seminars and conferences.

References

[1] ISO/IEC/IEEE 42010, Software, systems and enterprise. Architecture description, 2022. URL: https://www.iso.org/standard/74393.html.

[2] ISO/IEC/IEEE 15288, Systems and software engineering. System life cycle processes, 2023. URL: https://www.iso.org/standard/81702.html.

[3] V. Mokhor, O. Bakalynskyi, V. Tsurkan, Probabilistic criterion of information security management system development, in: Proceedings of the XIX International scientific and practical conference Information Technologies and Security, ITS 2023, CEUR Workshop, Aachen, Germany, 2023, volume 2577, pp. 159–168. URL: http://ceur-ws.org/Vol-2577/ paper13.pdf.

[4] ISO/IEC 27001, Information security, cybersecurity and privacy protection. Information security management systems. Requirements, 2022. URL: https://www.iso.org/standard/ 27001.

[5] ISO/IEC 21827, Information technology. Security techniques. Systems Security Engineering. Capability Maturity Model® (SSE-CMM®), 2008. URL: https://www.iso.org/standard/ 44716.html.

[6] ISO/IEC 27032, Cybersecurity. Guidelines for Internet security, 2023. URL: https://www.iso.org/standard/76070.html.
[7] ISO/IEC 27005, Information security, cybersecurity and privacy protection. Guidance on managing information security risks, 2022. URL: https://www.iso.org/standard/80585.html.
[8] Technical Report CMU/SEI-2009-TR-010, (2009), Secure Design Patterns, 2009. URL: https://insights.sei.cmu.edu/documents/813/2009_005_001_15110.pdf.
[9] J. Jürjens, UMLsec: Extending UML for Secure Systems Development, In: JM. Jézéquel, H. Hussmann, S. Cook (Eds.) "UML" 2002 – The Unified Modeling Language. UML 2002, volume 2460 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2002, pp. 412-425. doi: 10.1007/3-540-45800-X_32.
[10] J. Jürjens, Secure Systems Development with UML, Springer, Berlin, Heidelberg, 2004. doi: 10.1007/b137706.
[11] EVITA: E-safety vehicle intrusion protected applications, 2008. URL: www.evita-project.org.
[12] G. Pedroza, L. Apvrille, D. Knorreck, AVATAR: A SysML Environment for the Formal Verification of Safety and Security Properties. in: Proceedings of the 11th Annual International Conference on New Technologies of Distributed Systems, IEEE Xplore, Paris, France, 2011, pp. 1–10. doi: 10.1109/NOTERE.2011.5957992.
[13] M. M. Jamjoom, A. S. Alghamdi, I. Ahmad, Service Orientated Architecture Support in Various Architecture Frameworks: A Brief Review. in: Proceedings of the World Congress on Engineering and Computer Science, IMECS 2012, Newswood Limited, San Francisco, USA, 2012, volume II, pp. 1–6.
[14] Y. Dorogyy, Meta model of compensatory-decompensational approach for architecture of critical it infrastructure designing. in: Proceedings of the 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2018, IEEE Xplore, Lviv-Slavske, Ukraine, 2018, pp. 223–228. doi: 10.1109/ TCSET.2018.8336191.
[15] H. Holm, T. Sommestadm, M. Ekstedt, L. NordströM, CySeMoL: A tool for cyber security analysis of enterprises. in: Proceedings of the 22nd International Conference and Exhibition on Electricity Distribution, CIRED 2013, IEEE Xplore, Stockholm, Sweden, 2013, pp. 1–4. doi: 10.1049/cp.2013.1077.
[16] M. Jensen, C. Sel, U. Franke, H. Holm, L. Nordstrom, Availability of a SCADA/OMS/DMS System – A Case Study. in: Proceedings of the IEEE PES Innovative Smart Grid Technologies Conference Europe, ISGT Europe, 2010, IEEE Xplore, Gothenburg, Sweden, 2010, pp. 1–8. doi: 10.1109/ISGTEUROPE.2010.5638912.
[17] OMG Systems Modeling Language. URL: https://www.omg.org/spec/SysML/1.6.
[18] S. F. Telenyk et al., Methods for Investigating Properties of High-performance Infrastructures. The review, Control systems and computers, 1 (2015), 3–13.
[19] Y. Y. Dorogyy, V. V. Tsurkan, A survey of parametric model verification methods, Collection of scientific publications of NUS, 12 (1) (2020), 82–90. doi: 10.15589/znp2020.1(479).10.