Model-based Threat and Risk Assessment for Systems Design

Avi Shaked[1,2][a] and Yoram Reich[2][b]
[1]Cyber Division, Israel Aerospace Industries, Ashdod, Israel
[2]Systems Engineering Research Initiative, Faculty of Engineering, Tel Aviv University, Tel Aviv, Israel

Keywords: Threat and Risk Assessment, Model based Engineering, Cybersecurity, Security by Design, Systems Design.

Abstract: Integrating cybersecurity considerations in the design of modern systems is a significant challenge. As systems increasingly rely on connectivity and software to perform, cybersecurity issues of confidentiality, integrity and availability emerge. Addressing these issues during the design of a system – a security by-design approach – is desirable, and considered preferable to patching an existing design with extraneous components and mechanisms. In this paper, we present a model-based methodology for cybersecurity related systems design. This field-proven methodology takes into consideration cybersecurity threats alongside the system's composition and existing mechanisms, in order to communicate, assess and drive the incorporation of security controls into the system design. We discuss aspects of the methodology's design and how it relates to its real-life applications and usage context.

1 INTRODUCTION

Contemporary systems rely on software components and on connectivity to perform. Software components exist in diverse forms, such as Personal Computer based software, embedded software (e.g., software used in microcontrollers) or programmable logic (e.g., configuration of Field Programmable Gate Arrays components). Connectivity also manifests in various forms: a system can include internal connectivity between its components and/or can interact with external entities and systems (e.g., a connection to the Internet). Connectivity may be fixed and continuous, as in internal communication busses used throughout the system's operation; or temporal, as in a connection between a system component and a dedicated test equipment during acceptance tests or software version upgrades.

While software and connectivity are used to promote system capabilities (e.g., rapid deployment of new features and operation in swarms, respectively), they also open up the system to cyber-threats. These threats may impact the confidentiality, integrity and/or availability of the system, either in its entirety or with respect to its components. It is therefore of utmost importance to identify such potential threats and manage them as risks. A popular term used to denote such activities is Threat and Risk Assessment (TRA).

When designing a system, TRA may be used not only to assess the cybersecurity risk posture of the system under development, but also to affect its design to take into account the cybersecurity perspective, and consequently mitigate potential threats. Such an approach is commonly referred to as "security by-design."

In this paper we present a methodology for incorporating cybersecurity considerations in the design of systems based on TRA. In Section 2, we provide the relevant background related to security by-design methodology. In Section 3, we discuss our design research method. In Section 4, we present TRADES: a Threat and Risk Assessment for the Design of Engineered Systems methodology. In Section 5, we share information concerning the multiple case study validation of our methodology, based on its applications in *Israel Aerospace Industries (IAI)*. Section 6 discusses related approaches, and Section 7 summarizes key points and discusses the potential of further research efforts.

[a] https://orcid.org/0000-0001-7976-1942
[b] https://orcid.org/0000-0002-0922-8381

Shaked, A. and Reich, Y.
Model-based Threat and Risk Assessment for Systems Design.
DOI: 10.5220/0010187203310338
In *Proceedings of the 7th International Conference on Information Systems Security and Privacy (ICISSP 2021)*, pages 331-338
ISBN: 978-989-758-491-6

2 BACKGROUND

2.1 Contextual System Development Characteristics

We provide relevant context with respect to systems development in IAI, which affected the design of our methodology.

In IAI – as in many other high-tech industries and companies that develop complex, engineered systems – systems development is typically done within the scope of a development project. The development scheme favours a top-down approach, according to which system-level requirements and design are established to meet stakeholder requirements (and specifically, the purchaser of the system). These then drive the development of lower-level components.

Another related characteristic of systems development in IAI is its hierarchical approach, which views the system as a hierarchical organization of components. This is aligned with the view offered by Vee-model, a prominent system development model (Forsberg and Mooz, 1991). For example, the highest hierarchy of a system comprises several sub-systems; each of these sub-systems includes several lower-level components that may be further decomposed into even lower-level components, and so forth. The hierarchical organization provides engineering and managerial context and responsibilities (e.g., reflecting in a Work-Breakdown Structure associated with the development project).

A significant manifestation of the hierarchical approach is that – throughout systems development – the systems-engineering team responsible for the development of a component in a specific hierarchy defines requirements for the lower hierarchy's components, according to which the lower-level components are designed and eventually verified. These requirements are expected to relate to multiple aspects of functionality, performance and quality, with cybersecurity aspects included.

In general, the cost of performing changes late in the system development life-cycle or after a system is fielded is increasingly high (compared to performing these changes earlier or avoiding them by proper design in advance), and changes that originate from a cybersecurity perspective are no exception. It is, therefore, important to communicate the need for cybersecurity mechanisms as early as possible in the development effort (Mead and Stehney, 2005; Shevchenko et al., 2018).

Also, it is noteworthy that IAI development projects typically last for several years, and that the resulting systems remain operational for a long period of time (a 10-20 year operational period is not uncommon). During such extended periods of development and operations, the organization (and often, its customers) considers it important to have an up-to-date documentation of the system's design. Also, throughout such lengthy life-cycles, the cybersecurity threat landscape is subject to change (due to newly emerging attack technologies and techniques, for example), and this introduces further challenges to the threat and risk assessment of the systems.

2.2 Threat and Risk Assessment

Typically, the threat and risk assessment of systems is expressed as natural language text, tables and free-form diagrams (that typically provide a specific view of a given design). A threat model related publication (Bodeau and McCollum, 2018), for example, depicts a representative example as a story with free-form diagrams. Another representative example is a radar system security research report (Cohen et al., 2019), in which threats are expressed in the form of a table. The aforementioned documentation is typically prepared ad-hoc and is not necessarily aligned with the actual system design. This does not support rigorous engineering nor the establishing of the cybersecurity posture throughout the system life-cycle. The free-form approach to TRA indicates a gap in sound, practical methodology.

Some conceptual frameworks relating to TRA and to security by-design exist, providing important concepts and guidance. Two prominent examples follow. First, MITRE – a not-for-profit organization which specializes in systems engineering and cybersecurity – offers a threat modelling framework (Bodeau and McCollum, 2018), which identifies the system composition and data flows as critical in evaluating the potential impact of a cyber-attack. Second, the United States National Institute of Standards and Technology (NIST) offers an authoritative source – NIST SP 800-160 – which discusses systems security-by-design, and provides exhaustive natural language guidelines for introducing security considerations in systems engineering activities (Ross et al., 2016). Both aforementioned publications lack a concrete approach for applying TRA for systems design. They do, however, stress the importance of introducing requirements for the system and its constituents (alternately, introducing capabilities/functions to the system and its constituents) from a security perspective. Introducing quality related system requirements has also been acknowledged as an effective mechanism to incorporate security concepts in early stages of development (Mead and Stehney, 2005). However, we are unaware of any rigorous, field-proven model-based approach to do so. We discuss some related approaches in Section 6.

3 METHOD

TRADES is a domain-specific methodology for security by-design. TRADES was developed based on concepts and insights from another research project in which we developed another domain-specific methodology (Shaked and Reich, 2019).

First, TRADES was designated as a model-based engineering methodology. We consider this as TRADES' significant value proposition. Our main goal was to promote a rigorous, lasting depiction of a system's cybersecurity design, and addressing this based on a well-defined, unambiguous data model (Ramos et al., 2011) seemed essential; especially when considering the aforementioned lengthy development and operation life-cycle and the changing threat landscape. A model-based approach facilitates the ability to use the model as a single source of reference, establishing communication and coordination between cybersecurity experts, system engineers, managers and regulators. Specifically, modelling languages were suggested as a prospective approach to promote the definition and communication of security related abstractions (Mailloux et al., 2018). Digital models can also be maintained throughout the development, reflecting and communicating any changes to both the system and its cybersecurity assumptions (e.g., applicable threats, included security mechanisms and risk management decisions).

Second, as in our previous research and in accordance with others (for example, Pullonen et al. (2019)), our domain-specific language – which is part of the model-based methodology – was designed to be as minimal as possible, yet expressive, to support its cognitive usability. We initially identified the key ontological entities (threat, security control, and component; as described in the next section) and their relationships, and designed our model and representations accordingly. The design of our model and representations included cognitive considerations that were established as effective in our previous research and were reapplied here with domain-related adaptations. Furthermore, our domain-specific methodology was designed to fit the contextual aspects of systems development (as mentioned in Section 2.1).

We implemented our domain-specific methodology using software modelling technologies. While the details regarding the implementation of TRADES on top of a modelling infrastructure is beyond the scope of this paper, we provide an open-source tool to support the dissemination and use of the methodology[3].

The methodology was applied to several real-life cases, and this further establishes its validity, in accordance with the multiple case study research methodology. A brief overview of these applications is provided in Section 5, with lessons learned discussed both in Section 4 (where we introduce the design of the methodology) and in the concluding discussion.

4 TRADES METHODOLOGY

In this section, we present the TRADES methodology. Prominent design considerations (of the methodology) are explained throughout.

4.1 System Design: Ontology and Its Representation

In TRADES, systems and their constituents are represented by a "***component***" typed element. A component is an abstract concept, which may represent logical and/or physical/structural entities. Considering the context in which we wished to apply TRADES – specifically the hierarchical approach to systems development – system composition is portrayed in TRADES using the aggregation of components in hierarchies. In addition to being aligned with the overall systems development methodology of IAI, the use of hierarchies contributes to designing security in various level of abstraction, while maintaining traceability between low-level and high-level concepts (Mead and Stehney, 2005). It is noteworthy that hierarchies are components themselves. A "box"-like notation is used to represent a component. A two-dimensional shape has the ability to be used as a container to reflect a design hierarchy, and in TRADES, a component can indeed serve as a container to include other components.

[3] The open source TRADES modelling tool is available at https://github.com/iai-cyber/TRADES.

Another aspect of system design that TRADES emphasizes is the exchange of data between components. This exchange is denoted using a directional arrow between components (representing an “affect relation” model element), which conveys a “***data***” typed element. The use of a directional shape is designed to force the cybersecurity analyst to “think” in terms of the data, its source and its target, as opposed to thinking in terms of links between entities.

TRADES includes another system design element: “***security control***.” While TRA related aspects of the security control ontological entity (and its directly corresponding model element) are explained in the subsequent section, we note that this element represents a security control (also commonly referred to as “mitigation mechanism”) that is found/designed in the system or in its constituents. This designation of the security control as a system design element is reflected in the design of the TRADES notation (i.e., its representation): a security control is placed within the boundaries of a component, denoting that the implementation of the security control is the responsibility of the component owner. A security control element is currently represented as a box with a dashed outline, differentiating it from a component element (whose outline is solid).

Figure 1 provides an example of the system design notation of TRADES. A system – SystemA – comprises two subsystems: SubsysA1 and SubsysA2. The model includes lower level hierarchy details for SubsysA1, with two components (Component1 and Component2) exchanging two data items (Data1 from Component1 to Component2 and Data2 in the opposite direction). Two security controls are identified in the model: Control1 – is associated with the top hierarchy (it is therefore located directly under the SystemA component); and Control2 is associated with a component in the second level of hierarchy (SubsysA1).

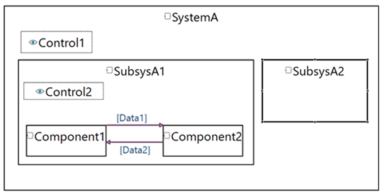


Figure 1: TRADES notation for capturing system design.

4.2 TRA: Ontology and Its Representation

In TRADES, the TRA revolves around the threats that are identified as applicable to the system (and to its constituents). Corresponding with the threat ontological entity, we use “***threat***” model elements to denote threats. Since threats are external to the system design, they appear – in the TRADES analysis diagram – outside of the system boundaries (that are denoted by the outline of the box-like shape of the component element). In order to further facilitate cognitive differentiation between design elements and threat elements, the notation used for threat elements takes the form of an ellipse.

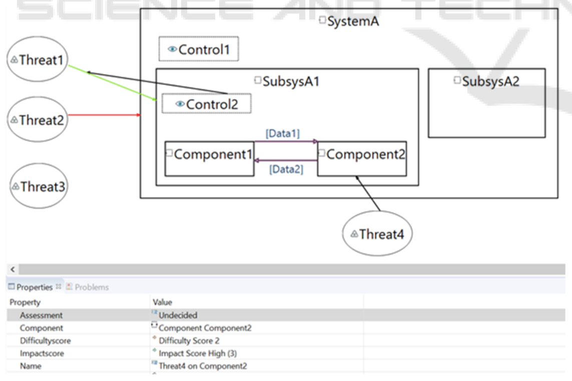
Threats may appear regardless of their applicability to a specific design. This promotes the ability to import a repository of threats whose applicability to the system under assessment needs to be determined[4].

Once a threat element appears in the model, it may be allocated to one or more components, marking that it may be applicable to the specific components. This allocation is graphically denoted in the TRADES diagram using an arrow from the threat to the component. A non-trivial aspect of this allocation is that it is recognized to be a key model element for TRA: “***threat allocation.***” This was identified only after the two first applications of TRADES in IAI. After a threat has been allocated to a component, the allocation is the entity according to which the risk assessment of the system is performed. Specifically, potential impact and difficulty (with high difficulty denoting low feasibility) are assessed with respect to the specific threat allocation. The TRADES model includes elements relating to both impact and difficulty, and these elements may be associated with a threat allocation, characterizing its assessment with respect to the two risk management traits (“***impact***” and “***difficulty***” elements exist in the TRADES model, corresponding with the respective TRA scoring system ontological attributes, but are not represented in the TRADES diagram design).

Another task in assessing the cybersecurity risks relates to the security controls that are included in the system design. For this, security controls are linked with the threat allocation which they are believed to mitigate, using a directional arrow from control to the threat allocation arrow, designating a “***threat mitigation***” element.

[4] For example, we successfully imported all of MITRE’s CAPEC attack patterns (https://capec.mitre.org) as threats to our models, resulting in over 500 threats available for a cybersecurity analyst to consider when performing TRA.

Figure 2 demonstrates the full design notation of TRADES. The system (of Figure 1) is shown alongside several threat elements. Threat1 was allocated to SubsysA1, Threat2 was allocated to SystemA, Threat3 remains unallocated, Threat4 is allocated to Component2. Control2 is associated as a mitigation for the Threat1 to SubsysA1 allocation. Control1 was not associated with any threat allocation (which may signify, for example, that it is either redundant or irrelevant to the cybersecurity perspective[5]). The figure exemplifies an additional representational aspect of the TRADES methodology: a colouring notation is used to reflect attributes of some elements. Specifically, the Threat1 to SubsysA1 threat allocation is marked in green, indicating it has been estimated as an acceptable risk (possibly based on the mitigation by Control2); while the Threat2 to SystemA threat allocation is marked in red, indicating it has been estimated as a gap (an unacceptable risk). The Threat4 to Component2 threat allocation appears in black, as remaining undecided (this is based on the "Assessment" property value, shown in the "Properties" section below the diagram). Some other properties of the threat allocation are also shown. The "Difficulty score" and "Impact score" properties are of particular interest, as they are associated with the respective elements (associating the threat allocation with the "2" difficulty score model element and the "high (3)" impact score model element).

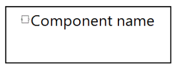
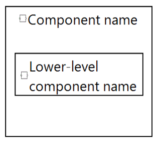
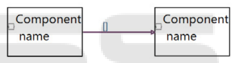
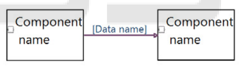
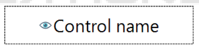
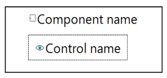
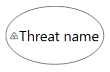
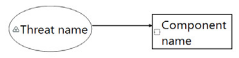
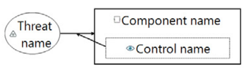


Property	Value
Assessment	Undecided
Component	Component Component2
Difficultyscore	Difficulty Score 2
Impactscore	Impact Score High (3)
Name	Threat4 on Component2

Figure 2: TRADES full notation.

Table 1 summarizes the identified ontological concepts for the cybersecurity related systems design domain and how they are addressed in the TRADES model and in the TRADES diagram. This table exemplifies the one-to-one correspondence between ontology and model elements as well as between model elements and their representation, based on the ontological clarity (Wand and Weber, 1993) and the semiotic clarity (Moody and van Hillegersberg, 2008) design principles. The table also identifies the leading design perspective – Systems Engineering (SE) or Cybersecurity (Cy) – that is associated with each concept, stressing that cybersecurity domain experts (or – according to NIST's terminology – Systems Security Engineering) need to collaborate with systems engineers in order to design a system with cybersecurity considerations.

Table 1: TRADES methodology with respect to domain ontology.

Ontological concept	**Leading design perspective**	**Model element**	**Representation notation in TRADES diagram**
System components (logical/ physical)	SE	Component	
Hierarchical composition	SE	Component as container	
Data flow	SE	affect relation	
Data	SE	Data	 (data conveyed by data flow)
Security control	Cy	Control	
Security control's allocation to component	Cy	composition relation (component comprises control)	
Threat	Cy	Threat	
Association of threat with component	Cy	Threat allocation relation	
Security control's threat mitigation	Cy	Threat mitigation relation	
Impact	Cy & SE	Impact score	Out of diagram scope (represented in other model views / as attributes)
Feasibility	Cy	Difficulty score	Out of diagram scope (represented in other model views / as attributes)

[5] Controls may be used for purposes other than for mitigating threats. For example, a system may include controls based on its availability objectives or its safety objectives.

In terms of systems development yields, the primary output of applying TRADES to a system design is the identification of security controls that need to be included in each component, in the respective level of hierarchy. Once a security control is allocated to a component, it is established as a function of the component. The security related function is subsequently defined in terms of requirements; and from this point on, they enter the development cycle just like any other requirement. This approach is well-aligned with the approach encouraged by NIST (Ross et al., 2016).

5 APPLICATION REPORT

Since its development, TRADES has been used to analyse and support the design of more than 10 different systems, including remotely piloted aerial vehicle systems, radar systems, airborne systems and missile systems.

TRADES was applied to systems in various stages of the system development life-cycle. Some applications were designated to provide an existing system's security posture in order to support the planning of future development blocks. Other applications were made during the course of development, for supporting the system design as it was devised (by system engineers, hardware engineers and software engineers). Also, several applications were made during the project proposal phase, in order to communicate cybersecurity aspects with stakeholders, specifically with the acquisition authority (customer) and regulators.

The resulting system models ranged from a model of 10 system components in three hierarchies, 5 data items, 5 threats and 4 controls to a model of about 900 system components in four hierarchies, 150 data items, 30 threats and no controls[6].

In all of the applications, TRADES was evaluated as promoting communication and coordination between the parties involved.

One difficulty we experienced in some of the applications relates to the lack of a dedicated element for the communication link itself, as opposed to the data it conveys. We found that system-engineers (in practice) often prefer to present the system architecture with physical links and protocols as opposed to data exchanges; and that some cybersecurity experts like to identify risks primarily based on these communication links.

6 RELATED WORK

TRADES' leading motivation is incorporating security related perspective into systems design. As such, TRADES shares a common underlying approach with NIST SP800-160 (Ross et al., 2016) and the SQUARE (Security Quality Requirements Engineering) methodology (Mead, 2007). The NIST publication provides considerations, and does not provide a concrete methodology as TRADES. SQUARE does provide high level process definitions, but – unlike TRADES – it is not a model-based approach, and it does not provide any model which corresponds with the domain ontology. As an example, one of the steps in SQUARE is "Categorize requirements," with initial requirements and architecture identified as inputs to the association of the requirements with the relevant hierarchy and component (Mead and Stehney, 2005). TRADES offers a more concrete take on this, as it specifies and captures the aspects of the design as well as of the TRA that are needed in order to assign requirements to the appropriate hierarchy and component. With respect to this, we also note that SQUARE's identification of inputs for this step (of its prescribed process) misses an important input: the identification of a threat allocation, and particularly the identification of the hierarchy/component to which a threat is allocated. This allocation has implications to the association of requirements for security controls with a specific hierarchy/component.

Threat modelling approaches that rely on diagrammatic representation exist[7], with data flow diagrams (DFDs) being a popular diagrammatic form, as discussed in a review of available threat modelling methods (Shevchenko et al., 2018). The review specifically identifies that using DFDs is insufficient for threat modelling; further emphasizing that the common, DFD-based approaches fail to derive

[6] Specifically, in the largest model yet, the identification of controls was deemed out of scope, and was left to a potential future work. The maximum number of control elements in a specific TRADES application so far is about 50.

[7] These should not be confused with model-based engineering methods. Model-based engineering requires the establishment of an explicit underlying model – commonly known as "meta-model" – and its realization in the form of a database of modelling elements; with diagrams providing a viewpoint into the database elements and/or supporting updates to the database (e.g., adding or updating elements). Purely diagrammatic methods – such as those presented shortly – often lack the formal ontological foundations and the establishment of a rigorous data model.

pertinent domain ontology, which is required in order to perform a systematic TRA. Specifically, upon examination of a known DFD-based tool (Microsoft's Threat Modelling Tool), we found that DFD has not been adapted – as a formal representation – to include threat modelling elements; and that its system design perspective in threat modelling remains information-flow oriented. Correspondingly, the DFD representation misses the representation of security controls as well as the hierarchical organization of systems, and it was deemed inappropriate for system level TRA scenarios in IAI (and companies employing similar development paradigms).

MITRE's conceptual threat modelling framework (Bodeau and McCollum, 2018) illustrates cyber threat modelling key constructs as a meta-model. Its focus is on threat-related elements, without providing any system-related elements. MITRE's Common Attack Pattern Enumeration and Classification (CAPEC) scheme exhibits similar focus on the threat element, and lacks a systematic approach (Yuan et al., 2014). In comparison, TRADES sponsors a more holistic view of conducting TRA, taking into account contextual aspects, and specifically those related to the system design (that is under assessment) as well as to the management of threat allocations with respect to the relevant design hierarchy.

TRADES supports top-down systems development approach that facilitates security by-design, introducing a dedicated cybersecurity perspective that can be used to design security features throughout systems development. In comparison, TLDR (Mahler et al., 2020) and PE-BPMN (Pullonen et al., 2019) were deemed inappropriate as security by-design approaches for such context. TLDR (Threat identification, ontology-based Likelihood, severity Decomposition, and Risk integration) is a threat and risk assessment methodology which focuses on the analysis of an existing design, and does not offer a security design perspective. PE-BPMN (Privacy Enhanced Business Process Model and Notation) is business process oriented, and is primarily aimed for privacy-related analysis of a given process design; as opposed to TRADES which is systems oriented, and aims to feed a wider scope of cybersecurity aspects into the system requirements and design. PE-BPMN primarily focuses on confidentiality aspects (and, to some extent, also includes data integrity aspects), whereas TRADES is more inclusive, and can be used to introduce aspects of confidentiality, of integrity – with respect to both performance and data – and of availability.

7 DISCUSSION

Designing systems with cybersecurity considerations is essential, as systems increasingly depend on software and connectivity to perform. We presented an applicable methodology – TRADES – for supporting the incorporation of a cybersecurity perspective to systems design.

Specifically, TRADES supports the elicitation of security related requirements. This is based on identifying desirable security controls in a contextualized form (of pertinent threat allocations and risks); and associating these controls as functional capabilities in different hierarchical levels of the system design.

Based on multiple real-life applications of TRADES, its contribution to the communication and coordination of the cybersecurity perspective and of pertinent system design between those involved in development efforts was established. The TRADES-based models are maintainable throughout the development life-cycle.

We attribute the aforementioned qualities of TRADES to its underlying domain-specific, model-based approach. Specifically, the TRADES model and notation were derived from profound ontological understanding of the domain; and the resulting models are consistent and unambiguous (as opposed to commonly used free-form diagrams and text, for example).

While the current TRADES model is aligned with our domain's ontology, we are still considering how to address the difficulty associated with some system engineers and cybersecurity experts' desire to include a dedicated element to denote a link and its characteristics. We are also evaluating the effectiveness of additional colouring notations. While we use colouring notations rigorously, based on our TRADES data model, we have yet to crystalize a definitive colouring notation (i.e., one that is to be used as a standard for all applications). We hope that our further research will be fruitful with respect to these.

Future research may integrate TRADES with other modelling domains (e.g., systems engineering model-based engineering methods or frameworks), as well as with cybersecurity related threat repositories[8].

[8] See, for example, footnote #4 for our preliminary take on this.

Integration potential also exists in the integration between TRADES and PE-BPMN, synthesizing the two approaches to facilitate a security by-design approach to system development which also takes system-related processes (such as operational processes or maintenance processes) into account.

REFERENCES

Bodeau, D. J., & McCollum, C. D., 2018. System-of-systems threat model. The Homeland Security Systems Engineering and Development Institute (HSSEDI) MITRE: Bedford, MA, USA.

Cohen, S., Gluck, T., Elovici, Y., & Shabtai, A., 2019. Security Analysis of Radar Systems. In Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy (pp. 3-14).

Forsberg, K. and Mooz, H., 1991. The relationship of system engineering to the project cycle. In INCOSE International Symposium (Vol. 1, No. 1, pp. 57-65).

Mahler, T., Elovici, Y., & Shahar, Y., 2020. A New Methodology for Information Security Risk Assessment for Medical Devices and Its Evaluation. arXiv preprint arXiv:2002.06938.

Mailloux, L.O., Beach, P.M. and Span, M.T., 2018. Examination of security design principles from NIST SP 800-160. In *2018 Annual IEEE International Systems Conference (SysCon)* (pp. 1-8). IEEE.

Mead, N. R., & Stehney, T., 2005. Security quality requirements engineering (SQUARE) methodology. ACM SIGSOFT Software Engineering Notes, 30(4), 1-7.

Mead, N.R., 2007. How to compare the Security Quality Requirements Engineering (SQUARE) method with other methods (No. CMU/SEI-2007-TN-021). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

Moody, D. and van Hillegersberg, J., 2008. Evaluating the visual syntax of UML: An analysis of the cognitive effectiveness of the UML family of diagrams. In International Conference on Software Language Engineering (pp. 16-34). Springer, Berlin, Heidelberg.

Pullonen, P., Tom, J., Matulevičius, R. and Toots, A., 2019. Privacy-enhanced BPMN: Enabling data privacy analysis in business processes models. *Software and Systems Modeling*, *18*(6), pp.3235-3264.

Ramos, A.L., Ferreira, J.V. and Barceló, J., 2011. Model-based systems engineering: An emerging approach for modern systems. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(1), pp.101-111.

Ross, R., McEvilley, M. and Oren, J., 2016. NIST Special Publication 800-160: Systems Security Engineering. *National Institute of Standards and Technology*.

Shaked, A., and Reich, Y., 2019. Designing development processes related to system of systems using a modeling framework. *Systems Engineering*, *22*(6), 561-575.

Shevchenko, N., Chick, T.A., O'Riordan, P., Scanlon, T.P. and Woody, C., 2018. Threat modeling: a summary of available methods. Carnegie Mellon University Software Engineering Institute Pittsburgh United States.

Wand, Y. and Weber, R., 1993. On the ontological expressiveness of information systems analysis and design grammars. *Information Systems Journal*, *3*(4), pp.217-237.

Yuan, X., Nuakoh, E.B., Beal, J.S. and Yu, H., 2014. Retrieving relevant CAPEC attack patterns for secure software development. In *Proceedings of the 9th Annual Cyber and Information Security Research Conference* (pp. 33-36).