Validating Reliability and Security Requirements in Public Sector Infrastructure Built by Small Companies

Roar E. Georgsen* and Geir M. Køien

University of South-Eastern Norway, Norway
E-mail: roar@aiwell.no
**Corresponding Author*

Received 22 April 2025; Accepted 22 April 2025

Abstract

Municipal infrastructure in Norway is built primarily by small specialist companies acting as subcontractors, mostly with minimal experience working with information and communication technology (ICT). This combination of inexperience and lack of resources presents a unique challenge. This paper applies model-based systems engineering (MBSE) using the systems modelling language (SysML) to combine validation of reliability and security requirements within a mission-aware interdisciplinary context. The use case is a 6LoWPAN/CoAP-based system for urban spill water management.

Keywords: Model-based systems engineering, security by design, ICT dependability.

Journal of ICT Standardization, Vol. 13_1, 1–24.
doi: 10.13052/jicts2245-800X.1311

1 Introduction

1.1 Motivation

According to the Intergovernmental Panel on Climate Change (IPCC), the near future will include intensifying weather events, resulting in heavy rainfall and flooding. Nine out of ten global disasters are water-related, and the UN Environment Programme (UNEP) estimates costs related to water infrastructure will amount to EUR 23 trillion by 2030 [1]. The World Bank reports that the cost of repairing water-related infrastructure in Europe will reach EUR 24 billion per year by 2050 [2]. At the same time, UN studies indicate that 6.3 billion people will be migrating into urban settings worldwide by 2050, amounting to 200,000 people per day [3]. This densification of cities reduces the paths available for surface water runoff, and urban areas are rapidly losing runoff capacity to clear their streets of rainwater after heavy precipitation. Using parks, playgrounds, and other green areas as temporary retention basins for surface water runoff is increasingly becoming a health hazard due to pollution from overspill between under-dimensioned waterways. More than 76,000 Norwegians have E.coli contaminated drinking water [4, 5], and a 2019 incident in Askøy municipality resulted in 2000 citizens falling ill and two deaths [6].

1.2 The Norwegian Case

Water infrastructure in Norway is owned and operated by local governments and is financed primarily through water fees. This limited funding combined with a severe shortage of engineers in the public sector has led to a maintenance shortfall estimated by SINTEF at NOK 320 billion [7, 8]. Therefore, municipalities are looking for new and cost-effective approaches to designing and managing urban water systems, introducing new technology such as distributed sensor networks and real-time data analysis. However, this has introduced new security risks since local authorities also have a severe ICT competence deficit [9–12]. Therefore, in addition to a climate change and water infrastructure maintenance shortfall, The Norwegian Directory of Health (Helsedirektoratet) has identified poor ICT security as one of the main threats to public health [13]. As pointed out in response to a 2021 Florida water system hack, under-regulation of water infrastructure ICT security has resulted in under reporting of poor security practices and related incidents [14]. In Norway, the Food and Safety Authority (Mattilsynet) regulates ICT security in water infrastructure. Whereas the regulations covering

ICT security practices in the power grid totals 9826 words, the equivalent section regulating water infrastructure is only 96 words [15]. The specifics are left entirely to the municipalities, with enforcement primarily based on self-reporting.

Due to insufficient internal competence, local authorities must rely on their existing supply network [16]. ICT security has sometimes been left out entirely in some water management systems, with control panels openly accessible via the Internet [17]. The reason for this is that the suppliers themselves lack competency in ICT security.

The building, infrastructure, and facilities sector is the largest on the Norwegian mainland, accounting for a third of all enterprises. However, of the 67,908 entities involved in mainland construction projects, only 36 companies employed more than 250 people in 2019, and most companies involved in projects had less than ten employees [18], a pattern repeated across all OECD countries [19]. These companies are highly specialized and struggle to meet the growing demand for documented safety and security in design. We propose that model-based systems engineering (MBSE) is a promising approach to coordinating this network of smaller suppliers in public sector infrastructure projects. A single model approach can support expert knowledge capture to inform on-ground engineering efforts and support project-wide validation of stakeholder requirements.

1.3 Layout of the Paper

Section 2 of this paper provides background on engineering in very small companies and a brief introduction to MBSE methodologies and tools in the context of system security engineering and security by design. Section 3 describes a specific use case, a wireless 6LoWPAN/CoAP-based system for urban spill water management, demonstrating safety and security by design using MBSE. Section 4 analyses the model and resulting changes to the original design arising from the modelling process. Finally, we note points for further research not covered in this paper.

2 Background

2.1 Engineering in Very Small Companies

Engineers working with product development in small companies typically take on a wide range of tasks outside their area of expertise. This is especially true when it comes to developing devices for the Internet of Things (IoT). The

demand for industrial devices with low power consumption drives a need for optimised custom hardware, and usability requirements drive connectedness and increase the scope of data collection. The same engineer will likely work on the entire stack from bare silicon to mobile applications, often as an architect, project manager and developer concurrently. Companies mitigate these competency gaps by working as part of supply networks [20], in practice by informal communication developer to developer. Attempts to enforce more systematic coordination between suppliers through required standards or service level agreements (SLAs) have largely failed. An examination of the supply chain to the Australian defence industry found that such legal structures did not impact actual engineering practices [21].

Small companies do not see standards outside their primary domain as relevant [22], and documents are produced primarily as legal and contractual requirements. Malicious actors increasingly target businesses too small to have their own IT departments because these are likely to have poor security. 60% of companies allow the use of personal devices, and 32% of small companies have no policies or procedures requiring employees to seek permission from or inform the company when doing so. 52% of security violations in small companies are discovered entirely by chance [23].

2.2 Model-based Systems Engineering

Model-based systems engineering (MBSE) is the *formalized application of modelling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases* [24].

The model is the sole source of truth and reflects the state of system development, and integrates multiple complementary perspectives in a compatible manner. This focus on an integrated model differentiates MBSE from simply engineering with models, where multiple models may have inconsistent assumptions and semantics. As system complexity grows or there are multiple stakeholders, document-centric approaches suffer from the increasing risk of overlooking critical information, as documentation tends to become incomplete and inconsistent. An integrated model enables connections between model elements, effective and efficient information retrieval, automatic propagation of changes, and automated model validation. A model-based approach supports an incremental generation of architecture from simple reusable component models, gradually constructing more complex subsystems [25]. The modelling environment can then automatically

generate any required documents from the model, with comprehensive traceability maintained for all system information, generated data, and design decisions. Using declarative models with explicit interfaces facilitates communication between engineering disciplines by establishing a shared context for discussion. MBSE is now starting to deliver rigour and effectiveness in complex systems development [26, 27]. Some view MBSE as a potentially revolutionary tool for optimising cost and quality [28]. Whereas a traditional functional decomposition approach captured 50% of problem understanding as measured by time to stakeholder consensus, modelling use cases and scenarios increased problem understanding to more than 90% on the first iteration [29].

2.3 The Systems Modelling Language (SysML)

The systems modelling language (SysML) [30] is a general-purpose graphical notation for modelling complex cyber-physical and socio-technological systems. SysML provides graphical representations for modelling requirements, behaviour, structure, and parametrics. The Object Management Group (OMG) developed the notation jointly with the International Council on Systems Engineering (INCOSE) as a subset of UML 2 with extensions. However, version 2 has a new metamodel not constrained by UML that includes textual syntax alongside the graphical notation but remains visually similar to UML [31, 32]. Figure 1 provides an overview of the nine diagrams in SysML grouped by type and how they relate to UML 2.0. Some modifications, such as replacing the Class concept with an equivalent Block element, are primarily intended as a linguistic device to make the notation less software-centric. Other modifications introduce non-software concepts such as continuous, probabilistic and physical item flows into existing UML diagrams.

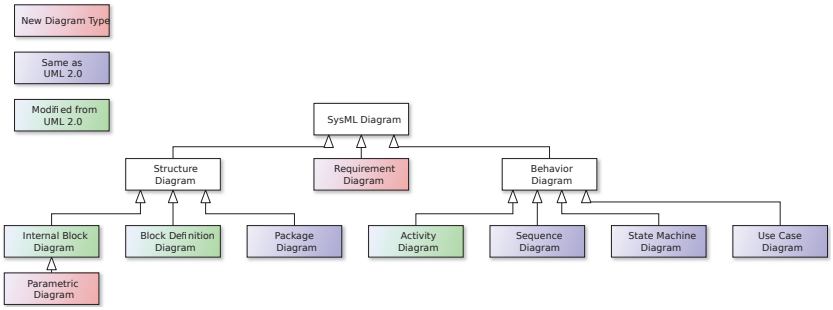

Figure 1 SysML diagram taxonomy [33].

2.3.1 SysML requirements modelling

The first of the two large innovations in SysML syntax is the notation for modelling requirements. Figure 2 shows a non-normative extension of the

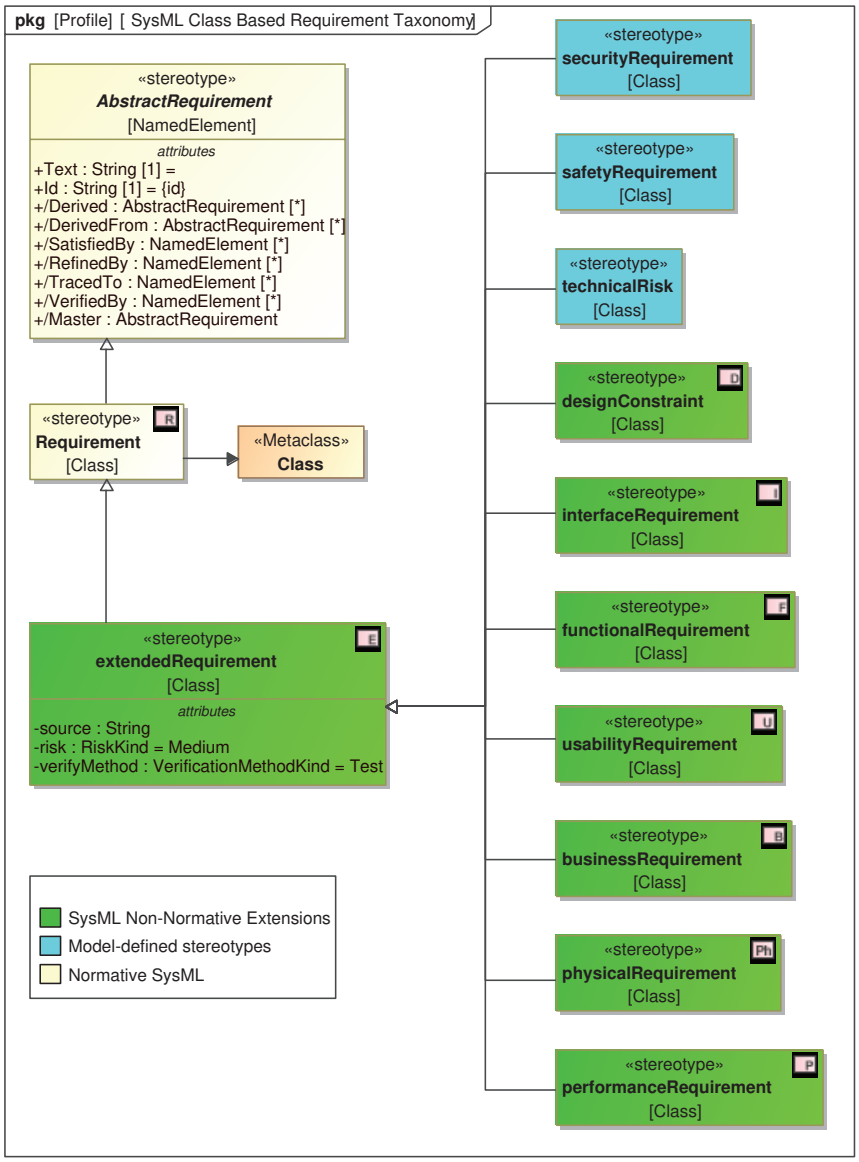

Figure 2 SysML class-based requirements taxonomy.

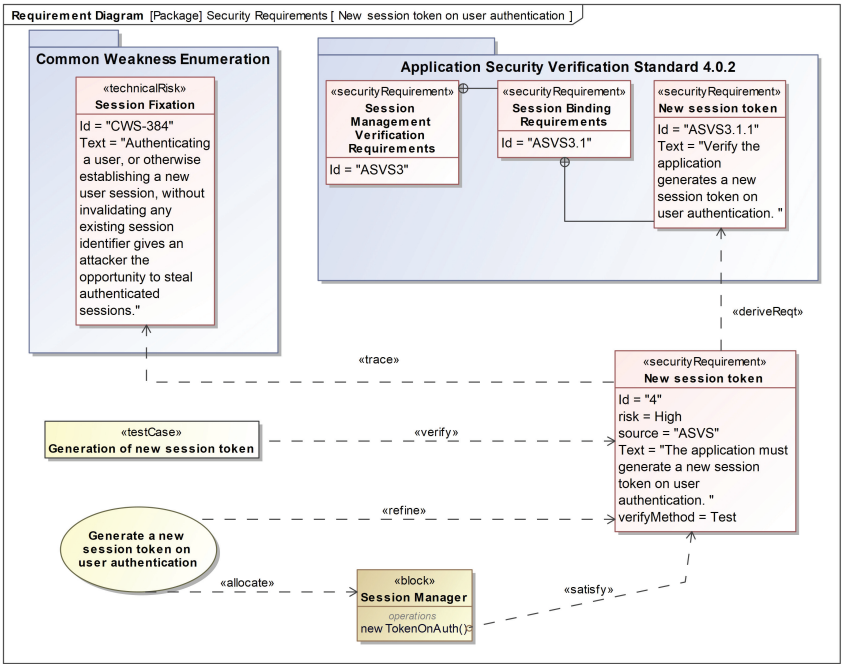

Figure 3 Using relationships to link model elements.

basic syntax similar across most modelling tools. The diagram also adds three new requirement types to the model. Typically, extending the Class metaclass is used to create subtypes of requirements that correspond to the traditional sections of a requirements document. This legacy view of requirements is practical when it is necessary to synchronise the model with externally stored text-based requirements. Figure 3 shows an example of importing predefined textual requirements into the model. However, the primary benefit of SysML requirements comes from the relationships defined on the AbstractRequirement type. Figure 3 models a requirement as being derived from a corresponding requirement in the Application Security Verification Standard (ASVS) and traced to a corresponding weakness with a CWS number. A use case that may include multiple model components refines the definition of the requirement, and the "verify" relationship links the requirement to a test case. The "satisfy" relationship creates a link in the model between the requirement and the property or function that satisfies it.

The "allocate" relationship can link use cases and functions to a component, but it can also link model elements directly to requirements. Suppose a

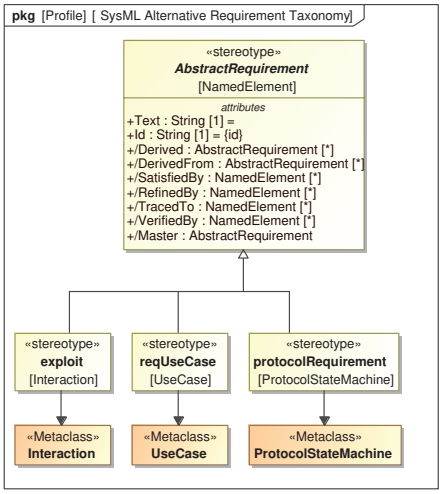

Figure 4 Non-class SysML requirements.

known vulnerability is modelled as a requirement and allocated to a component. In that case, that vulnerability will show up as an implicit requirement whenever a design uses an instance of that component. Domain experts can model requirements within their respective domains, and this will be enforced and constrain the design space for other users of the model.

Whereas the taxonomy in Figure 4 is useful when textual requirements already exist, a requirement can be an extension of any named element metaclass. Figure 4 shows the definition of three types of requirements based on different behavioural model elements. This feature enables SysML requirements to be more specific and less ambiguous than is possible with textual requirements.

2.3.2 SysML parametric modelling

The second innovation in SysML is the addition of notation for modelling constraints and mathematical relationships. Figure 10 shows an example of a parametric diagram. Constraint properties connect to value properties on the owning block, as well as to other constraint properties. The SysML primary specification does not stipulate a specific constraint language.

2.3.3 Executable SysML

In order to obtain the most benefit from MBSE, it is necessary to utilise SysML in a machine-readable manner. SysML relies on the OMG XML

Metadata Interchange (XMI) to exchange modelling data, and most modelling environments integrate well with external tools such as MS Excel, MATLAB, OpenModelica, Maple, or Mathematica. They can leverage these as supplemental solvers and simulation engines and combine tools based on their respective strengths. Another standard method of integration is the Functional Mockup Interface (FMI) [34]. A supplemental specification of SysML defines a modelling standard and syntax subset that allows for optimal integration and data exchange for executable models [35]. Several studies document the use of executable SysML used for in-situ real-time requirements monitoring of live systems [36, 37]. One study used a machine-to-machine message broker to connect a globally distributed luggage handling system to a SysML model performing real-time analysis [38].

2.3.4 The role of MBSE in security by design

A core idea of security by design is that, if one invests time early on, it is possible to avoid many common security problems. Fixing errors and mitigating vulnerabilities post-deployment can cost 100 times more than catching these errors in development [39]. Threat modelling can capture and disseminate expert security knowledge to the broader engineering team and provide security analysts with a more accurate understanding of the system. However, despite the well-known benefits, threat modelling is often performed late in the engineering lifecycle and often not at all unless mandated by customers or regulatory authorities [40]. In the context of MBSE, the role of the security expert is not primarily to audit the final design, but to help build a digital threat model that can be applied as a constraint on the global design. This ensures that the expensive resource that security experts represent is utilised with a high return on investment. A machine-readable threat model can automatically validate security requirements as the design process progresses. Also, being continuously alerted to violations of security constraints throughout development can help elevate security competency and awareness within the engineering team.

2.3.5 SysML applied to security and reliability

As systems become more complex and interconnected, it is increasingly challenging to construct models that properly reflect reality in terms of attacks. One approach is to recognise that a networked system is under constant threat from an infinite number of attackers. Consequently, the possible attacks, known and unknown, may be assumed to have a Poissonian probabilistic distribution in the same way as a random failure [41]. Malicious actions

could ideally be modelled as a probability distribution over the possible attack actions. This approach assumes that while it is helpful to be aware of known vulnerabilities and common attack vectors, it is potentially more fruitful to focus on possible fault states. Given that a significant proportion of vulnerabilities will always be unknown, these could be modelled as any other bug that may lead to system failure [42].

Systems theoretic process analysis (STPA) models unsafe states using control systems theory based on control actions and feedback loops, based on methods from industrial process control. STPA has been extended to include information security [43], and the methodology has been implemented as concurrent safety, reliability and security analysis using a single SysML model [44–46]. Because SysML syntax is not necessarily familiar to domain experts, much research has focused on transforming models so that specialists tools in safety and security modelling can be used in conjunction with the shared system model [47–49]. Using a single integrated model with custom views allows information captured using informal methods to feed more detailed analysis for high priority issues. The case study in Section 3 uses SysML to play the game Elevation of Privilege (EoP), a casual card game based on Microsoft's STRIDE threat model [50, 51]. Each card in EoP represents a specific threat from STRIDE and, in the case study, it is modelled as a SysML requirement. The model is used to visualise the system, and as each player plays a card, they must explain to the group what the threat on the card entails in the current context. This continues until all cards in the deck have been played. The game is intended to facilitate group discussions, and cards deemed relevant are linked in the model to specific components, interfaces or functions. Potential mitigations, notes, and requirements are added to the model as they come up during the game.

3 Case Study

3.1 6LoWPAN/CoAP-based Urban Flood Prevention

The system of interest is a siphonic drainage system intended to mitigate the insufficient capacity of urban waterways in removing spill water during heavy rainfall (Figure 5). Such flooding is a critical public health concern because contaminated spill water and sewage may overflow into drinking water or pollute beaches and green areas. The system increases throughput by removing air from drainage pipes, thus creating a suction effect that siphons away water at several times the rate of a non-pressurised system.

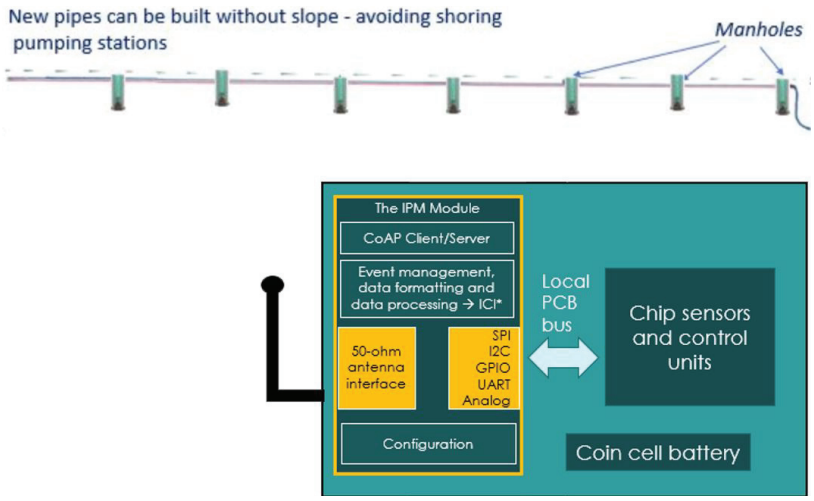

Figure 5 Siphonic drain system with wireless communication (Aiwell Water/Radiocrafts).

Because the system works by increasing the capacity of the existing pipes, there is no need for wholesale replacement of the water transport system, a minimal need for digging, and significantly less disruption to commercial activities, making the solution very cost-effective. Siphonic drainage is a mature technology standard in roof wells, but needing to keep the pipes under pressure has made it impractical for street drainage. However, using low-frequency long-range wireless networks enables real-time monitoring that could make maintaining such systems more feasible. The proposed solution uses a 6LoWPAN 868MHz IPv6 mesh network with CoAP as the application-level protocol, purchased as a module-on-chip with an integrated network stack mounted on the developing company's custom-designed PCB.

3.2 Playing Elevation of Privilege

The company developing the drainage system designs valves and end-user applications in-house but outsources PCB design and firmware development. Makers of the module-on-chip act as consultants, and other experts and stakeholder representatives are brought in as required. This ad-hoc network communicates informally to arrive at a common understanding of the system requirements.

The game Elevation of Privilege starts with constructing a visual model of the system, and Figure 6 shows the beginnings of a high-level model using

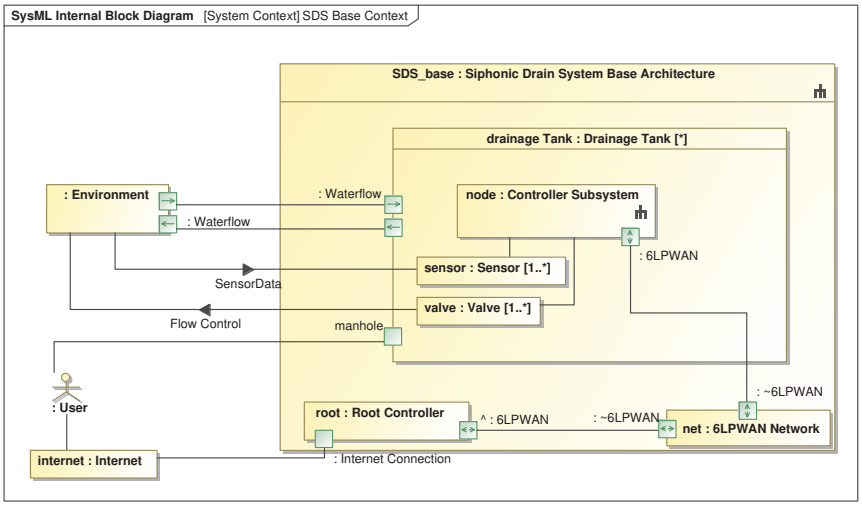

Figure 6 A basic context model serves as the starting point for playing Elevation of Privilege.

an internal block diagram. The primary purpose of this activity is to identify system flows, interfaces, and interaction points. Players may use predefined parts or define new ones during the game. As the game progresses, users may navigate the model to reveal different levels of abstraction. Figure 7 shows the same diagram as in Figure 6, but with a greater level of detail. The model highlights the CoAP interface due to known security vulnerabilities previously identified during the modelling exercise. Navigating to these elements will give details on the vulnerability, including links to further details elsewhere. Because an attacker needs access to the CoAP network to exploit the vulnerability, and the 6LoWPAN network is encrypted on the MAC level, a requirement is added in Figure 8 to maintain the CoAP traffic within that network and not expose it on the Internet. Notes are added directly to the diagram as prompts for potential solutions. The actual solution will depend on trade-off studies based on cost and reliability metrics in the parametric model.

3.3 Parametric Model of Reliability and Availability Metrics

3.3.1 Mathematical model

We assume a single repairable unit representing a single node in the network for our example reliability and availability model. For simplicity of

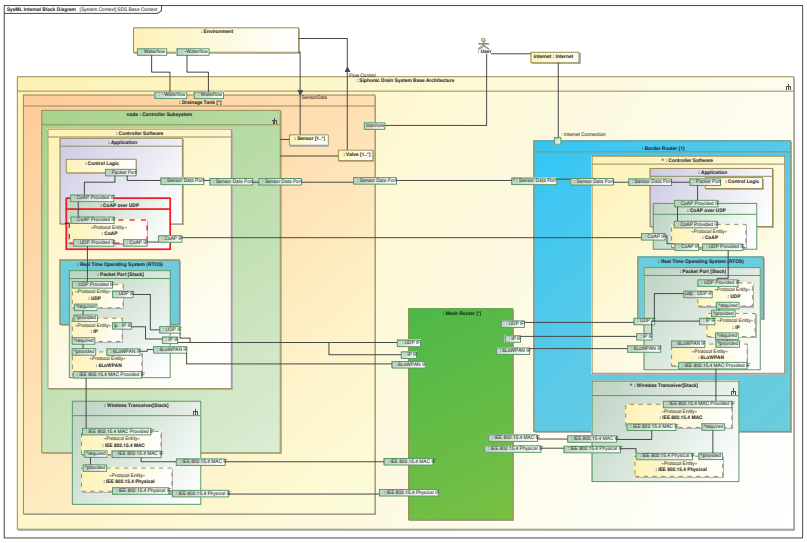

Figure 7 Zooming in on the context model to reveal details. Interfaces are modelled as layers [52]. The CoAP interface is highlighted due to a previously captured vulnerability.

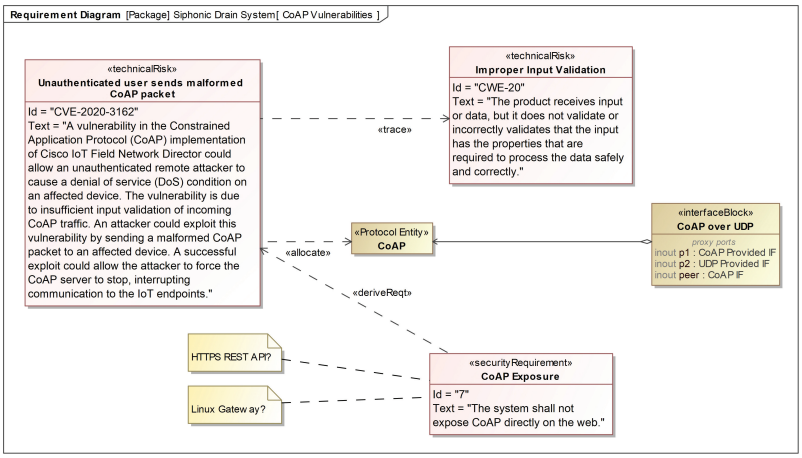

Figure 8 Previously modelled CoAP vulnerability [53].

illustration, we use a non-state-space model for the single nodes. We also assume a constant failure rate, which is an appropriate approximation for single unit models [54]. However, on the network level, the model would have to account for local state space, and transmission errors would depend on distance, medium, and environmental noise.

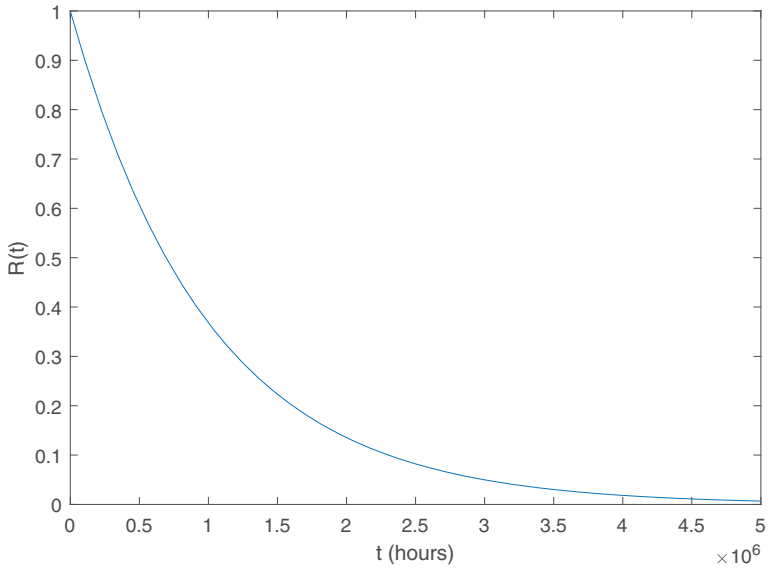

Figure 9 Survival function of unit with constant failure rate $\lambda = 10^{-6}$.

The failure rate $h(t)$ is the probability that given a system is up at time t, it will be down at $t+\Delta t$ with Δt being the shortest time step accounted for by the model. The unit's survival probability function is given in Equation (1).

$$R(t) = e^{-\int_0^t h(u)\mathrm{du}} \tag{1}$$

Because we assume a constant failure rate, we can simplify Equation (1) by replacing the function $h(t)$ with constant λ. Also, because we assume each unit is perfectly repairable or replaceable, we introduce a time a at which the survival function in Equation (2) returns to its initial value of 1. Figure 9 shows the survival function for a single repair cycle from Equation (2) plotted with a failure rate of one failure per 10^6 hours.

$$R(t) = e^{-\lambda(t-a)} \tag{2}$$

Given Equation (2), the expected mean time to failure (MTTF) would be:

$$MTTF = a + \int_0^\infty e^{-\lambda(t-a)} dt = a + \frac{1}{\lambda} \tag{3}$$

The mean time to repair (MTTR) is given by Equation (4). The exponential case is given in Equation (5). The repair function $G(t)$ gives the

probability that if a repair cycle is started at $t = 0$, the unit will be up by time t. In the exponential case, we assume a constant repair rate μ and start time a.

$$MTTR = \int_0^\infty (1 - G(t))\mathrm{dt} \tag{4}$$

$$MTTR = a + \int_0^\infty (1 - e^{-\mu(t-a)})\mathrm{dt} = a + \frac{1}{\mu} \tag{5}$$

Given both failure rate and repair rate, it is possible to calculate system reliability and availability. Reliability denotes the probability of continuous operation of the system, whereas availability denotes the system being up at a given time. Equation (6) gives the instantaneous availability, i.e. the probability that the system is up at time t. In contrast, Equation (7) gives the interval availability, i.e. the expected proportion of time the system will be up until time t. However, over time the two values converge on what is known as the steady-state availability of the system, formally defined in Equation (8).

$$A(t) = \frac{\mu}{\lambda + \mu} + \frac{\lambda}{\lambda + \mu} e^{-(\lambda+\mu)t} \tag{6}$$

$$A_I(t) = \frac{\mu}{\lambda + \mu} + \frac{\lambda}{t(\lambda + \mu)^2}(1 - e^{-(\lambda+\mu)t}) \tag{7}$$

$$\lim_{t\to\infty} A_I(t) = \lim_{t\to\infty} A(t) = A = \frac{\mu}{\lambda + \mu} \tag{8}$$

Equation (9) gives the probability that the system will operate continuously without failure from time t to τ. For the current system of interest, this would be relevant to ascertain that the system remains continuously operational for the duration of a flooding event.

$$IR(t, \tau) = e^{-\lambda\tau} A(t) \tag{9}$$

The mean time between failures (MTBF) is given by the sum of MTTF and MTTR.

$$MTBF = MTTF + MTTR \tag{10}$$

The final part of the mathematical model defines expressions for the sensitivity of system availability. Equation (11) defines the sensitivity of the system to changes in time to failure. Equation (12) defines the sensitivity of

the system to changes in time to repair.

$$S_F = \frac{\partial A}{\partial MTTF} = \frac{MTTR}{(MTBF)^2} \tag{11}$$

$$S_R = \frac{\partial A}{\partial MTTR} = \frac{MTTF}{(MTBF)^2} \tag{12}$$

S_F and S_R are essential when evaluating alternative measures for improving system availability. S_F is always positive, as system availability will always go up as the mean time to failure goes up. S_R is always negative, as increasing the mean time to repair will reduce system availability. If MTTF is much larger than MTTR, reducing time to repair will always have a much more significant impact on system availability than increasing time to failure. Reducing repair time could be more straightforward than mitigating the reason behind the failure. In such cases, it could be cost-effective to leave a vulnerability unmitigated if the expected frequency of a successful exploit is low and the cost of mitigation is high, compared to the ease and cost of restoring the system to a healthy state. S_F and S_R can provide stakeholders with a quantitative measure to evaluate this trade-off when comparing design alternatives and validating requirements.

3.3.2 Implementing the mathematical model using SysML

Figure 10 shows the mathematical model from Section 3.3.1 implemented using a SysML parametric diagram. Because the model is parametric, it is possible to vary the failure rate and repair rate and evaluate the system's reliability and availability. SysML is agnostic when it comes to the language used in defining constraints, and most tools allow for solvers to be set at the level of individual constraint blocks. This makes it possible to use different solvers for different constraints. The result is a parametric model that is composable and reusable. Figure 11 shows how a generic parametric reliability model is applied to a component through generalisation. It is then possible to automatically aggregate component level analysis results into a system level analysis. If this is combined with other parametric models, such as component cost models, it is possible to evaluate the total cost of a system with different designs, taking into account a combination of unit cost, cost of repair, cost of downtime, and cost of redundancy. This can provide a quantitative input to stakeholders evaluating alternative mitigations to any identified vulnerabilities in the system.

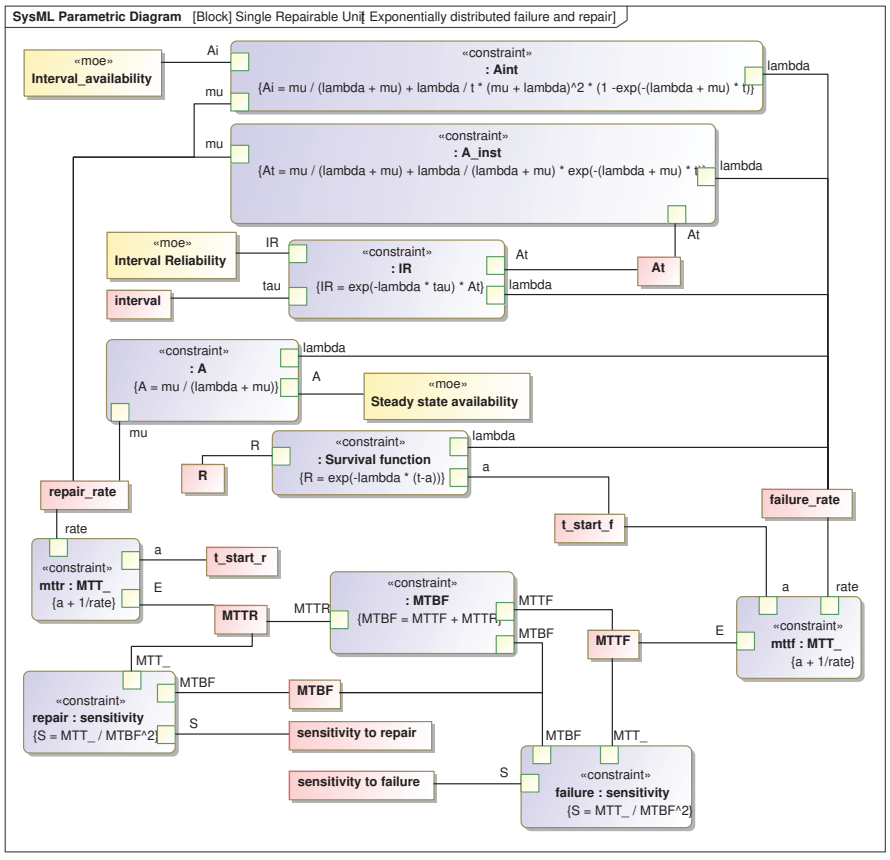

Figure 10 Parametric model of a single repairable unit.

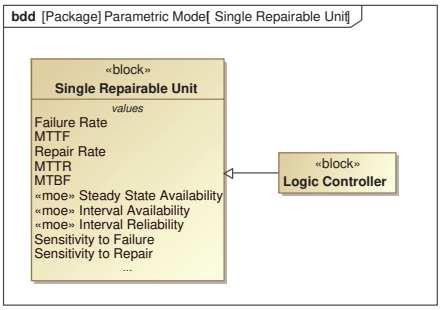

Figure 11 Applying the parametric model through generalisation.

4 Analysis

The vulnerability in the CoAP interface identified as part of the threat modelling exercise in Section 3.2 could be mitigated in different ways. One alternative proposed by players during Elevation of Privilege was to add a gateway unit to avoid exposing CoAP traffic to the Internet. However, a parametric analysis as illustrated in Section 3.3.2 showed this option to have a high cost both for hardware and the development of an API. Whitelisting nodes is another alternative, but this option also had a high overhead. Looking at the details of the vulnerability (CVE) and weakness (CWE) linked to in the model, as shown in Figure 8, it became apparent that these were primarily a concern for mobile networks in China, and a microcontroller as used in the system under analysis could simply reset to return to a safe state with no loss of sensitive data or resources. A more cost-effective solution in the analysed context would therefore be a simple hardware watchdog resetting the system in the case of traffic congestion.

5 Conclusions and Future Work

This paper examined how model-based systems engineering using the systems modelling language can support the culture of informal cooperation amongst the small specialist companies acting as subcontractors in public infrastructure projects and aid in efficient validation of requirements. A case study demonstrated a multi-disciplinary combined analysis of reliability and security. As part of an informal session with multiple stakeholders, a card game, Elevation of Privilege based on STRIDE, was used to prompt discussion and elicit information that could feed into the digital model for further analysis. Whereas the informal discussions resulted in multiple suggestions for mitigating vulnerabilities, a formal parametric model constructed by others in a separate session provided stakeholders with quantitative metrics to evaluate the alternatives.

Future work will expand on Elevation of Privilege by creating similar decks of cards to cover reliability hazards, stakeholder modelling, and usability, and accessibility design. While physical playing cards were viewed positively by stakeholders, future work will also examine digital card games implemented as part of the modelling environment. Previous work has shown small companies to be resistant to moving away from informal means of collaboration. However, early results indicate that model-based automation can support standardisation and quality improvement even in an informal

setting. In future studies, we plan to document how this approach affects compliance and resistance to change in small companies in more detail.

Whereas the case study described in this article was conducted in Norway, the capacity problems and technical debt in water infrastructure are global [1–3], as is the prevalence of small companies in the globalised supply chain [19] and the skills-gap and reporting issues as relates to system security [14]. This indicates that the analysis of the case study could be extended to other countries and other sectors.

Declarations

This research was presented at the 23rd International Symposium on Wireless Personal Multimedia Communications (WPMC2020) in October 2020 and funded by The Research Council of Norway and Aiwell Water as part of the Research Council's Industrial PhD Programme, in collaboration with The University of South-Eastern Norway. The programme requires that the lead researcher is an employee of the participating company, and the topic must be of clear relevance to the company's activities. The first author is the CTO of Aiwell. Dassault Systèmes provided the licenses for Cameo Systems Modeller used in the research.

References

[1] United Nations Environment Programme. Sustainable infrastructure and finance – How to contribute to a sustainable future. Technical report, United Nations, 2016.

[2] Julie Rozenberg. *Beyond the Gap: How Countries Can Afford the Infrastructure They Need While Protecting the Planet*. 2019.

[3] United Nations, Department of Economic and Social Affairs, and Population Division. *World Urbanization Prospects: The 2018 Revision*. 2019.

[4] Statistisk sentralbyrå (SSB). Kommunal vannforsyning. https://www.ssb.no/natur-og-miljo/vann-og-avlop/statistikk/kommunal-vannforsyning. Accessed on 2020-07-01.

[5] Ole Petter Pedersen. 76.000 har E. coli i drikkevannet – sjekk din kommune her. https://www.tu.no/artikler/83-000-har-e-coli-i-drikkevannet-sjekk-din-kommune-her/467719, June 2019.

[6] Jannicke Nilsen. Varaordfører: Høydebassenget stenges for godt. https://www.tu.no/artikler/varaordforer-pa-askoy-hoydebasseng-stengt-for-godt/467738, June 2019.
[7] Frode Skår. Investeringsbehovet fortsetter å øke. https://www.norskvann.no/index.php/10-nyheter/2420-investeringsbehovet-fortsetter-%C3%A5-%C3%B8ke, October 2020.
[8] May Rostad. Finansieringsbehov i vannbransjen 2016–2040. Technical Report 223/2017, Norsk Vann, 2017.
[9] Hilde Ludt and Morten Engebretsen. Sikkerhetskrav i IKT-anskaffelser. Technical Report 01/2019, Oslo Kommune Kommunerevisjonen, 2019.
[10] Øystein Solaas. 1 av 6 ansatte i Bergen kommune ga fra seg passord: Et gjenstridig problem. https://www.digi.no/artikler/kommentar-1-av-6-ansatte-i-bergen-kommune-ga-fra-seg-passord-et-gjenstridig-problem/479522, November 2019.
[11] Heidi Sævold. Fersk rapport: Oslo kommune stilte for dårlige sikkerhetskrav til egne IT-systemer. https://www.digi.no/artikler/fersk-rapport-oslo-kommune-stilte-for-darlige-sikkerhetskrav-til-egne-it-systemer-br/457890, February 2019.
[12] Marius B. Jørgenrud. Nesten hundre norske kommuner har usikre nettsider. https://www.digi.no/artikler/nesten-hundre-norske-kommuner-har-usikre-nettsider/480840, December 2019.
[13] Overordnede risiko-og sårbarhetsvurderinger for nasjonal beredskap i helse- og omsorgssektoren 2019. Technical Report IS-2841, Helsedirektoratet, 2019.
[14] Brian Krebs. What's Most Interesting about the Florida Water System Hack? That We Heard about It at All. . https://krebsonsecurity.com/2021/02/whats-most-interesting-about-the-florida-water-system-hack-that-we-heard-about-it-at-all, February 2021.
[15] Per Helge Seglsten. Paragrafene som dekker datasikkerhet for kraftverk har 9.826 ord. De for vannverk har 96. https://www.digi.no/artikler/paragrafene-som-dekker-datasikkerhet-for-kraftverk-har-9-826-ord-de-for-vannverk-har-96/507440, March 2021.
[16] Harald Brombach. IT-sikkerheten i vannverk og fjøs får svært lite oppmerksomhet hos tilsynsmyndigheten. https://www.digi.no/artikler/it-sikkerheten-i-vannverk-og-fjos-far-svaert-lite-oppmerksomhet-hos-tilsynsmyndigheten/474093, October 2019.
[17] Harald Brombach. Kontrollpanelet til norsk vannverk lå åpent på internett. https://www.digi.no/artikler/kontrollpanelet-til-norsk-vannverk-la-apent-pa-internett/466822, June 2019.

[18] Statistisk sentralbyrå (SSB). Number of enterprises, by economic activity and size groups. https://www.ssb.no/en/virksomheter-foretak-og-regnskap/statistikker/foretak/aarleg-omsetning-og-sysselsetting/2020-07-09. Accessed on 2020-07-09.

[19] Organisation for Economic Co-operation and Development (OECD). Entrepreneurship at a Glance 2017. Technical report, Paris, France, September 2017.

[20] Leontin K. Grafmüller, Stephan Hankammer, Sarah Hönigsberg, and Hendrik Wache. Developing complex, mass-customized products in SME networks: Perspectives from co-creation, solution space development, and information system design. *International Journal of Industrial Engineering and Management*, 9(4):215–227, December 2018.

[21] Xuan-Linh Tran. *Systems Engineering Tool Selection Framework for Australian Defence Small and Medium Enterprises*. PhD thesis, University of South Australia, 2014.

[22] Angela D. Robinson. Very small entities (VSE); The final systems engineering (SE) frontier. In *2018 Annual IEEE International Systems Conference (SysCon)*, pages 1–4, Vancouver, BC, April 2018. IEEE.

[23] Trusler og Trender 2021. Technical report, NorSIS, 2021.

[24] Systems Engineering Vision 2020. Technical Report version 2.03, International Council on Systems Engineering (INCOSE).

[25] J Stephen Topper and Nathaniel C Horner. Model-Based Systems Engineering in Support of Complex Systems Development. *Johns Hopkins APL Technical Digest*, 32(1):14, 2013.

[26] Azad M. Madni and Michael Sievers. Model-based systems engineering: Motivation, current status, and research opportunities. *Systems Engineering*, 21(3):172–190, 2018.

[27] Nadia A Tepper. Exploring the use of Model-based Systems Engineering (MBSE) to develop systems architectures in naval ship design. Technical report, Massachusetts Institute of Technology, Cambridge, MA, 2010.

[28] Chris Paredis. Model-based systems engineering: A roadmap for academic research. *Frontiers in Model-Based Systems Engineering, Atlanta, GA*, 2011.

[29] Raymond Jorgensen. Defining Operational Concepts using SysML: System Definition from the Human Perspective. *INCOSE International Symposium*, 21(1):3005–3138, 2011.

[30] Object Management Group (OMG). OMG Systems Modeling Language (OMG SysML), v1.6, 2018.
[31] Systems Modeling Language (SysML) v2 RFP ad/2017-12-02, 2017.
[32] Hans Peter de Koning. What to Expect from SysML v2. In *MBSE2020 Workshop*, page 19, 2020.
[33] MovGP0. SysML Diagram Taxonomy. https://commons.wikimedia.org/wiki/File:SysML_Diagram_Taxonomy.svg, 22 January 2013, 01:56:32.
[34] Modelica Association. Functional Mock-up Interface for Model Exchange and Co-Simulation – Version 2.0.1, 2019.
[35] Object Management Group (OMG). SysML Extension for Physical Interaction and Signal Flow Simulation – Version 1.0, 2018.
[36] Robert Karban, Nerijus Jankevičius, and Maged Elaasar. ESEM: Automated systems analysis using executable sysml modeling patterns. In *INCOSE International Symposium*, volume 26, pages 1–24. Wiley Online Library, 2016.
[37] Minjun Seo and Roman Lysecky. Non-intrusive in-situ requirements monitoring of embedded system. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 23(5):1–27, 2018.
[38] Sylvia Melzer, Jan Philip Speichert, Oliver C Eichmann, and Ralf God. Simulating cyber-physical systems using a broker-based SysML toolbox. In *Proc. 7th Int. Workshop Aircr. Syst. Technol.(AST)*, pages 411–420, 2019.
[39] Jim Johnson. CHAOS 2020 Beyond Infinity. Technical report, Standish Group International, 2020.
[40] Adam Shostack. Elevation of Privilege: Drawing Developers Into Threat Modeling. In *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*, 2014.
[41] Bjarne E. Helvik, Karin Sallhammar, and Svein J. Knapskog. Integrated Dependability and Security Evaluation Using Game Theory and Markov Models. In Yi Qian, James Joshi, David Tipper, and Prashant Krishnamurthy, editors, *Information Assurance*, The Morgan Kaufmann Series in Networking, pages 209–245. Morgan Kaufmann, Burlington, January 2008.
[42] Bjarne E. Helvik, Petra Vizarreta, Poul E. Heegaard, Kishor Trivedi, and Carmen Mas-Machuca. Modelling of Software Failures. In Jacek Rak and David Hutchison, editors, *Guide to Disaster-Resilient Communication Networks*, Computer Communications and Networks, pages 141–172. Springer International Publishing, 2020.

[43] Ivo Friedberg, Kieran McLaughlin, Paul Smith, David Laverty, and Sakir Sezer. STPA-SafeSec: Safety and Security Analysis for Cyber-Physical Systems. *Journal of Information Security and Applications*, 34:183–196, June 2017.

[44] Bryan T. Carter, Georgios Bakirtzis, Carl R. Elks, and Cody H. Fleming. Systems-Theoretic Security Requirements Modeling for Cyber-Physical Systems. *Systems Engineering*, 22(5):411–421, 2019.

[45] B. T. Carter, G. Bakirtzis, C. R. Elks, and C. H. Fleming. A Systems Approach for Eliciting Mission-Centric Security Requirements. In *2018 Annual IEEE International Systems Conference (SysCon)*, pages 1–8, April 2018.

[46] G. Bakirtzis, B. T. Carter, C. R. Elks, and C. H. Fleming. A Model-Based Approach to Security Analysis for Cyber-Physical Systems. In *2018 Annual IEEE International Systems Conference (SysCon)*, pages 1–8, April 2018.

[47] Edward Huang, Leon F McGinnis, and Steven W Mitchell. Verifying SysML activity diagrams using formal transformation to Petri nets. *Systems Engineering*, 23(1):118–135, 2020.

[48] Florian Lugou, Letitia W Li, Ludovic Apvrille, and Rabéa Ameur-Boulifa. Sysml models and model transformation for security. In *2016 4th International Conference on Model-Driven Engineering and Software Development (MODELSWARD)*, pages 331–338. IEEE, 2016.

[49] Rabéa Ameur-Boulifa, Florian Lugou, and Ludovic Apvrille. Sysml Model Transformation for Safety and Security Analysis. In *Security and Safety Interplay of Intelligent Software Systems*, pages 35–49. Springer, 2018.

[50] Adam Shostack. Elevation of Privilege: Drawing Developers into Threat Modeling. In *USENIX Summit on Gaming, Games, and Gamification in Security Education*, page 12, 2014.

[51] Adam Shostack. *Threat Modeling: Designing for Security*. Wiley, Indianapolis, IN, 2014.

[52] Peter M. Shames and Marc A. Sarrel. A modeling pattern for layered system interfaces. *INCOSE International Symposium*, 25(1):914–927, 2015.

[53] National Institute of Standards and Technology (NIST). National Vulnerability Database (NVD) – CVE-2020-3162. https://nvd.nist.gov/vuln/detail/CVE-2020-3162#range-4768798.

[54] Kishor Shridharbhai Trivedi and Andrea Bobbio. *Reliability and Availability Engineering: Modeling, Analysis, and Applications*. Cambridge University Press, New York, NY, USA, 2017.

Biographies

Roar E. Georgsen received his B.Eng. in Computer Engineering and an M.Sc. in Systems Engineering from the University of South-Eastern Norway (USN). Currently, he is the CTO of Aiwell and an Industrial PhD Research Fellow with USN in Horten, Norway. His research interests include model-based systems engineering, digital transformation in small engineering teams, and integrated safety, security and reliability design.

Geir M. Køien received his PhD from Aalborg University, on access security for mobile systems. He has also worked for many years in industry, including LM Ericsson Norway and Telenor R&D. During these years he worked extensively with mobile systems and with security and privacy. He has also worked with the Norwegian Defence Research Establishment and with Norwegian Communications Authority on various security and communications related projects. Currently, he is a professor with the University of South-Eastern Norway (USN).