A Perspective on Ensuring Predictability in Time-critical and Secure Cooperative Cyber Physical Systems

Saad Mubeen*, Elena Lisova*, Aneta Vulgarakis Feljan†
*Mälardalen University, Sweden
†Ericsson Research, Sweden
saad.mubeen@mdh.se, elena.lisova@mdh.se, aneta.vulgarakis@ericsson.com

***Abstract*—Recent advancement in the development of embedded systems and in the integration of operational and industrial technologies has accelerated the progress of cyber-physical systems (CPSs) development. Cooperation of such systems allows to achieve new functionalities. However, often these systems are time-critical; hence, the developers of these systems are required to provide guarantees of the systems' properties, such as predictability and security. In this paper, we start by glancing through the research devoted to developing time predictable and secure embedded systems. Thereon, we extend the discussion to time-critical and secure CPSs and point out the challenges related to ensuring predictability during their development. In this context, we emphasize the importance of security as a prerequisite for time predictability. Moreover, we identify the gaps in the state of the art and describe our view on ensuring predictability in time-critical and secure CPSs.**

I. Introduction

Cyber-physical systems (CPSs) have been defined and interpreted in a number of ways by the research community. For instance, the International Conference on CPSs (ICPPS)[1] defines CPSs as the "physical and engineered systems whose operations are monitored, coordinated, controlled, and integrated by computing and communication." Lee and Seshia [1], [2] describe CPSs as the systems that emphasize the link between computation and physical processes, and as such the link among time, space and energy. Embedded computers monitor and control the physical processes, and vice versa the physical processes affect the computations. The radical transformation from an *embedded system* (ES) to a CPS comes from the emphasis on integration of physical processes and a more broad networking aspect. ESs have also been defined in several ways and there is no comprehensive agreed-upon definition of ESs as stated by Li and Yao [3]. According to one of the most widely used definitions of ESs [4], [5], an ES is designed to perform a dedicated functionality by means of a computer hardware, software, and perhaps additional mechanical components, sensors and other parts.

As this paper aims to investigate the support for predictability and security in CPSs that include one or more ESs, it is important to first clearly identify the boundary of the ES within the CPS, which is often ambiguous. In this context, we define an ES as the system consisting of hardware, software and interfaces (ports) to receive/send sensor/actuator signals and network messages. The inputs (sensor signals) arriving at the input interface and the computed outputs (actuation signals) delivered to the output interface of the system are considered parts of the ES as shown in Fig. 1. The actual sensors, actuators and physical processes that are sensed and controlled respectively are not considered as parts of the ES. Whereas, these entities, together with the ES and possibly communication with the cloud constitute a CPS as depicted in Fig. 1. For example, consider the airbag system in a car. If the car crashes and its deceleration is fast enough, the crash sensors are triggered, which send crash signals to the computing unit that produces the actuation signals to inflate the airbag. The sensor inputs, the computation unit (both hardware and software) and the actuation outputs are part of the ES. Whereas, the ES together with the crash sensors and the physical process (environment) in which they are deployed, the airbag actuator and the airbag itself constitute the CPS. Note that if two or more ESs are connected via an on-board network, the system is still regarded as an ES, more precisely as a distributed ES, as shown by the on-board interconnection of several ESs in Fig. 1.

The CPS considered in the above example is a time-critical system as it is required to provide a logically correct response (output) within a given amount of time (timing requirement); failing to do so can result in the system failure. Assume that the car in which the CPS is deployed is an autonomous vehicle. The car cooperates with other vehicles and/or road-side units (RSUs) to perform a certain cooperative functionality, e.g., avoid accidents. The above-mentioned CPS in the car together with the corresponding CPSs inside the other vehicles or RSUs provide an example of a time-critical cooperative CPS.

A system is considered to be predictable if its state/behavior can be forecasted at any point in time, given a known execution environment or a set of assumptions. The prediction can be made either qualitatively or quantitatively, and during the different system lifecycle stages. Predictability of a system is related to proving, demonstrating or verifying the fulfillment of the requirements that are specified on the system, whether they are functional or extra-functional. This paper focuses on the system development stage, and on two extra-functional requirements that affect the system's predictability: timing and security. The term predictability has also been used lately in the artificial intelligence community, where a system is capable

[1] http://iccps.acm.org/2019/?q=CFP

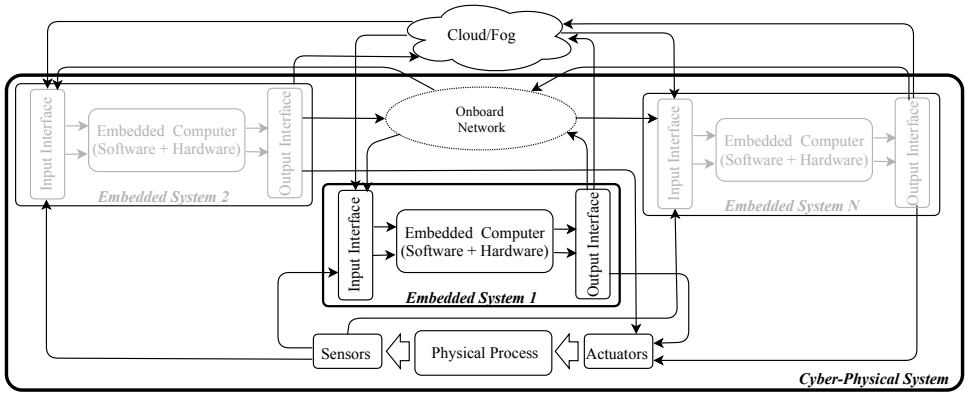

Fig. 1: Defining the boundaries of ES and CPS.

of predicting the future state changes and executing appropriate actions beforehand (e.g., for predictive maintenance), however this is outside of the scope of this paper.

Extensive research has been devoted in defining and studying predictability with respect to time [6], [7], [8], [9] and security [10], [11], [12] in the ESs community. Security is a broad term, that can be defined as a system property that allows it "to perform its mission or critical functions despite risks posed by threat" [13], where a threat can be defined as "the potential source of an adverse event" [13]. A threat is realized by an attack that exploits a vulnerability, i.e., a flow in the system, and targets one of the system assets. A concrete threat realization is an attack. One of the main security objectives in ESs is to consider data integrity and authenticity, as it is crucial to have enough confidence that the data received from the sensors represents the physical process correctly. That is, the data is not modified by an adversary or injected by an adversary masking the real data. In order to ensure that the input data is correct and not modified maliciously, we can as well use prediction algorithms (extrapolation) and also security mechanisms for integrity and authentication checks.

The main objective of this paper is to conduct an investigation of key issues involved in ensuring predictability in time-critical and secure cooperative CPSs. We start by exploring the state of the art for ensuring these properties in the ESs community, and identify the level of existing support during the design time in a broader context, i.e., for CPSs. In this regard, we seek to answer the following questions.

1) Are the existing frameworks for the development of CPSs expressive enough to specify timing and security requirements on the various components of CPSs?
2) Are there any existing methods and techniques that can formally verify the specified timing and security requirements of CPSs at the design time?
3) Are there any existing techniques to support time predictable and secure runtime environment for CPSs?

If the answer to any of the above questions is "no" or "may be", we further investigate the following two questions.

- What is missing from the existing solutions?
- How can the existing solutions be extended to support the development of time-critical and secure CPSs?

Note that while answering the posed questions, we identify potential gaps in the state of the art and propose extensions to the existing techniques and frameworks.

II. Running Example: Autonomous Quarry

In this work we illustrate our ideas with the help of an example of autonomous quarry, which is depicted in Fig. 2. A crusher machine crushes big stones into smaller ones. The crushed stones are transported by battery-powered autonomous haulers. The haulers cooperate with each other for efficient transportation of the crushed material. It is crucial for the production efficiency that the haulers arrive at the battery charging stations in time to avoid stopping in the middle of the quarry with drained batteries. It is undesirable for the haulers to approach the stations too early with still enough charge left in their batteries. The other important aspect is safe and efficient transportation. The haulers should have an updated map of the quarry with updated available routes and location of the static objects to prevent any accidents, e.g., crashing into obstacles or falling into pits. Each hauler should also be aware of the location of the rest of the haulers to avoid collisions with each other. We assume that the haulers receive this information (e.g., a map of dynamic objects) from a communication center, which has an established communication link with each hauler. Moreover, the haulers receive the control information regarding speed, direction and required actions from the communication center. In case of an immediately detected hazard or a communication failure, the haulers are capable of overriding a command from the communication center and relying fully on the information from the on-board sensors and local processing of the surrounding environment. Note that these systems operate in harsh environments, e.g., due to extreme amount of dust. Furthermore, these systems also share the environment with humans, thus safety is a crucial property to assure. In these systems, support for time predictability is crucial in assuring safety. Interestingly, wireless communication channels and increased connectivity among the vehicles impose security threats that can affect predictability, thereby jeopardizing the system safety.

III. Time Predictability in CPSs

This section provides a comprehensive discussion on time predictability in time-critical ESs and CPSs.

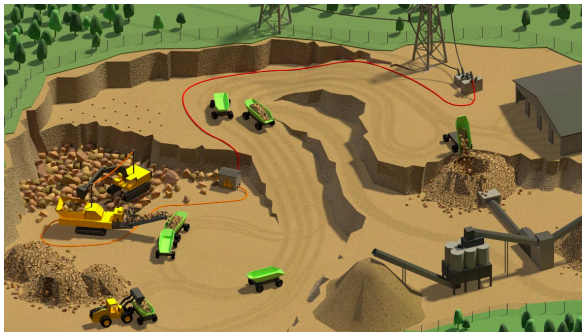

Fig. 2: Example of an autonomous quarry [14].

A. Predictability in Time-critical ESs

In this subsection, the scope of the discussion on time predictability is within the on-board ESs in each hauler in the example (discussed in Section II). Due to the time-critical nature of many ESs, it is required that all actions by these systems are performed in a timely manner such that the specified timing requirements are satisfied. Hence, at the design time, the developers of these systems need to verify and ensure that the systems are time predictable. Time predictability is a well-defined term in the real-time embedded systems theory [6], [7], [8], [9]. For a given system model and a set of assumptions, the system is said to be time predictable if it is possible to prove or demonstrate that all specified timing requirements will be satisfied when the system is executed.

The timing requirements can be specified on individual tasks, set of tasks and task chains in a node (single- or multi-core processor), individual messages in a network, and chains of tasks and messages in a distributed ES. Traditionally, the timing requirements mostly referred to the deadlines of tasks, messages and task chains [15]. That is, the response time of a task, message or a task chain, counted from the arrival of the input value at the input ports or network interfaces until the delivery of the computed output value to the output ports or network interfaces of the system, must not exceed the specified deadline. In the past few years, the research community extended the use of timing requirements beyond the traditional deadline requirement by considering several other timing requirements such as the reaction and age constraints, among others. These requirements have been incorporated in several domain-specific modeling languages, e.g., Timing Augmented Description Language (TADL2) [16], EAST-ADL [17] and the Rubus Component Model (RCM) [18]. These requirements have also become part of the domain-specific standards such as the automotive standard AUTOSAR [19].

The predictability of a time-critical ES is ensured by verifying its timing behavior during the design time and supporting its predictable execution at runtime. The timing behavior can be verified by using the schedulability analyses, whereas the runtime environment can be provided by means of a real-time operating system (RTOS). There is a plethora of schedulability analyses developed by the research community [15], [20], [21]. Similarly, there are many RTOSs that provide predictable runtime environment, e.g., VxWorks, Rubus, FreeRTOS, just to name a few. In crux, the development of predictable time-critical ESs has gained considerable maturity.

B. Predictability in Time-critical CPSs

The time-critical ESs are only one part of time-critical CPSs. Hence, the scope of time predictability in the CPS must extend the boundaries of the ESs to include the timing impact of sensors and actuators that are deployed in physical processes. The sensor values in the running example might be received from other haulers, RSUs or the control center. Two key parameters have been identified in this regard [22].

1) *Ready Time of Inputs:* The ready time of inputs in the CPS is referred to as the interval of time between the instant when the value of a sensor that is deployed in the physical process changes and the instant when the changed value appears at the input interface of the ES. The ready time of all sensor inputs coming from the physical processes should be time predictable.
2) *Ready Order of Inputs:* In the case of more than one sensor value arriving at the input interface of an ES, the computed output that controls the physical processes depends heavily on the arrival order of the inputs. The desired function of the CPS requires a specific arrival order of the inputs from the sensors that are deployed in the physical processes, which must be time predictable. This is referred to as the order predictability.

These two parameters are not considered in the definition of time predictability for ESs because they exist outside their interfaces. In order for a CPS to be time predictable, it should be possible to prove or demonstrate at the design time that all the timing requirements specified on the system are satisfied. These requirements include the timing requirements on the computations and communications as well as the requirements that constrain the ready time and ready order of the inputs. If a CPS offloads computations to the cloud as shown in Fig. 1, the definition of time predictability should also consider the timing impact of offloaded computations to the cloud controllers [23]; however, it is out of the scope of this paper.

It should be noted that even if a time-critical CPS is proven to be predictable at the design time, the predictability of the system can be jeopardized at runtime due to security threats to the time-critical data entering the system, e.g., from sensors, networks or other CPSs. Hence, the security of the data is integral to the time predictability of the system. Such security aspects will be discussed in the next sections.

IV. Security in Time-critical CPSs

This section discusses the design-time security challenges in ESs and relates them to the security of time-critical CPSs. Further, it explores the feasibility of corresponding security solutions in ESs to the security challenges in the CPSs.

A. Security Challenges and Solutions in ESs and CPSs

The security challenges in ESs have been extensively addressed by the research community [10]. The core security challenges for ESs at the design time are as follows.

1) *Resource Gap:* Many ESs struggle with keeping up with the requirements on computation and energy consumption to support security solutions.
2) *Flexibility:* Security is dynamic in nature and it can be challenging for ESs (which are often static) to provide flexible platform to support constant security updates.

3) *Tamper Resistance:* It is a challenge for ESs to counteract attacks coming from malware, which are capable of executing the downloaded applications.
4) *Security Assurance:* Assurance of security for ESs that tend to have increased complexity is a challenge.
5) *Cost:* Security solutions can be costly, thus it is a challenge to find the right balance between acceptable security level and the system design investment in low-cost devices.

To tackle the first challenge of limited resources the research community has developed many lightweight security solutions [24], [25]. Generally, this challenge does not concern CPSs as they can have relatively more resources, e.g., the haulers in the running example may contain large batteries, powerful processors (e.g., multi-cores) and high-bandwidth onboard networks (e.g., Ethernet). Thus, from the security perceptive CPSs do not have strict resource limitations.

The challenge of developing flexible platforms to provide security in CPSs is inherited from ESs. Although, this challenge is already addressed in ESs [26], it is yet to be tackled in CPSs, which are more open and connected compared to ESs and thus have higher risks and stricter security requirements. The hauler in the running example is more vulnerable to security attacks as compared to an ES (e.g., brake-by-wire system), due to its exposure to more attack surfaces, e.g., the hauler receives time-sensitive information via wireless links from the other haulers and the control center. Hence, an investigation is required to check feasibility of the solutions from the ESs domain for CPSs.

The challenge of tamper resistance can be tackled for ESs via lightweight security solutions and generally by incorporating security considerations into the system design. For CPSs which are more connected and especially for cooperating CPSs, the challenge requires a dedicated effort starting from the concept phase of the system design.

Security assurance in ESs is challenging because the system security is not composable, i.e., composing the system from individual components with specified security levels does not ensure the security of the composed system. Thus, considering a secure ES as a part of the CPS is not straightforward to assure an overall system-level security. The assurance of ESs is addressed for real-time properties [27] and safety [28], e.g., the ISO 26262 [29] functional-safety standard for road vehicles provides guidelines for assuring that any unreasonable risks due to malfunctions of electrical and electronic systems are avoided. However, the challenge of security assurance for ESs is not solved yet. The notion of the security assurance case exists [30], however there are not that many works in the area. The main showstopper is the dynamic nature of security, as new threats and vulnerabilities are constantly being discovered [31]. In the case of time-critical cooperative CPSs, this challenge gets even more complex because the system's decisions rely on time-sensitive information coming from other systems via wireless communication links.

Considering how costly security solutions can be for ESs, there is a number of risk assessment techniques allowing to find a balance between the possible possessed risk and the solution to mitigate or prevent it. Similar to the As Low As Reasonable Practicable (ALARP) [32] from the safety domain, an appropriate level of system security is defined when the resources required to be invested in breaching the security are compared to the value of the system assets. Many time-critical and cooperative CPSs are safety-critical meaning that the expenses threshold for the design phase of the system is high due to the criticality level.

B. Security as a Prerequisite for Time Predictability

Time predictability is built upon assumptions about the system and its environment. One of the common assumptions in ESs is integrity of systems' inputs, i.e., the inputs are not modified, forged, injected or deleted by a malicious adversary. However, the possible physical access to the system by an adversary is not covered by the classical assumptions of time predictability in ESs. Hence, there is an implicit dependence of time predictability on the security of input data. Shifting from ESs to CPSs, the assumption regarding security becomes even a bigger concern, as the system becomes less isolated and more complex. In cooperative CPSs, securing the data integrity becomes more challenging due to increased connectivity in these systems, which brings new attack surfaces and vulnerabilities if security is not addressed at the design time. For example, the time predictability of the autonomous haulers in the running example can be analyzed and guaranteed under a set of assumptions, including the assumption that the integrity of the sensor data is intact. An autonomous hauler gets command information about its movement from the control center. We also assume that the hauler has some local intelligence allowing it to sustain any temporary loss of control information, e.g., due to a failure in the communication channel [33]. For making its own decisions regarding its current actions, a hauler needs to have an updated map of the quarry and approximated location of the other vehicles. If an adversary is able to forge the command information containing the speed, direction and acceleration, the time predictability of the system that is already verified at the design time will not hold any more. Another possible scenario can be seen if a failure occurs in the communication channel due to degradation of communication quality or jamming, the time predictability of the system can be jeopardized, e.g., due to forged map or sensor information. Thus, to support time predictability in time-critical cooperative CPSs, the systems' security must also be supported.

V. Position

A. Position Regarding Time Predictability

In order to support the development of time-critical CPSs, the questions posed in Section I are refined as follows.

1) Are the existing models, languages and frameworks for the development of CPSs expressive enough to specify timing requirements not only on the computation and communication times but also on the ready times and ready order of the inputs acquired from the physical processes?
2) Are there any existing methods and techniques that can formally verify the specified timing requirements at the design time to support pre-runtime verification of CPSs?
3) Are there any existing techniques to support predictable run-time environment for CPSs, which can provide bounded delays with regards to the computation, communication, ready times and ready order of the sensor inputs?

To answer the first question we explore the existing development models, languages and frameworks for CPSs. The existing works in the computation and communication parts support the specification of timing requirements, which are sufficient to support the corresponding part of time-critical CPSs. For example, consider the automotive domain, where the AUTOSAR standard includes a comprehensive timing model that is able to specify 21 different timing constraints. Similarly, the EAST-ADL modeling language and several other component models including RCM are expressive enough to model and specify the timing requirements. Another example can be seen in the avionics domain, where the existing frameworks support the specification of timing requirements [34], [35].

However, the support for specifying the timing requirements with respect to the input ready times and the input ready order is still missing. There are a few works that discuss these terms [22]; however, the formal semantics of the requirements and corresponding ready times and order are still missing. This, in turn, hampers the specification of holistic timing requirements in time-critical CPSs, i.e., the timing requirements that constrain the delivery time of the output of an actuator corresponding to the time when a new input is generated from a sensor that is deployed in the physical process.

We believe, the existing *Order Constraint* used in the computations, which is included in the AUTOSAR standard and several other modeling languages such as EAST-ADL, TADL2 and RCM can be extended to support the input ready order timing requirement in CPSs. The Order Constraint constrains an order among the occurrences of events [18], [19]. If this definition is adapted to constrain the arrival order of the sensor values from the physical process then this constraint can be applied to constrain the input ready order in CPSs. On the other hand, the semantics of the input ready times and corresponding requirement need to be defined and included in the existing frameworks for the development of CPSs.

The second question can be answered by exploring the existing schedulability techniques [15], [20], [36]. The support to verify time predictability at the design time in the computation and communication parts seem to be mature. If the Order Constraint is extended to support the specification of timing requirement on the inputs arrival order in CPSs, the corresponding timing analysis can be used to verify this requirement at the design time [18]. To the best of our knowledge, the timing analysis to verify the timing requirement on the input ready times is missing from the state of the art.

The answer to the third question is similar to that of the second one. The existing execution frameworks and tools support predictable runtime environments in the computation and communication parts as discussed in Section III-A. However, these techniques need to be extended to provide upper bounds on the input ready times and to enforce the inputs ready order.

B. Position Regarding Security

The observed tendency is that there exist several techniques, analyses and frameworks for addressing security in time-critical CPSs. However, the systematic way of incorporating security in the system development at the design time and its run-time assessment during operational phase, which is especially of importance for cooperative CPSs, is not mature. The solutions considered in this paper for time predictability hold only if the data integrity, authentication and authorization are supported. Due to the strong binding between time predictability and security, this paper renders security as a prerequisite for time predictability in time-critical CPSs.

One of the core challenges in developing secure CPSs is that a particular solution (e.g., a hash-function to check integrity of messages) does not provide by itself any guarantees, as one has to pay attention to its implementation and security policies to be able to verify if this solution actually covers the required security objective. Thus, we advocate the top-down approach for addressing security at the system level and building a security assurance case to systematize the way security is provided and supported. An assurance case can be defined as "*an enabling mechanism to show that the system meets its prioritized requirements*" [37]. It aims at reasoning about the system's trustworthiness. The assurance case can be built for a system property such as safety [38], security [30] or ethics [39]. We envision the security assurance case as a way to collect and structure arguments over the system being acceptably secure. Security is dynamic by nature and requires run-time updates and refinements. It is not feasible to develop a security case from the scratch whenever there is an update. The challenge of handling the updates in an efficient way within a security case is an open challenge.

Now we answer the questions (posed in Section **??**) in relation to the current state of the art in security for time-critical CPSs. First, there are several existing methods for requirements elicitation of security [40], [41] as well as of joint safety and security [42], [43]. Second, there are many existing techniques and tools for the formal verification of these requirements [44], [45]. However, there is a lack of support for run-time frameworks that can provide secure run-time environment for the systems, which is crucial considering how often updates and patches can be required.

VI. Summary and future work

Cooperative CPSs require a tight combination of and coordination between computational and physical processes. These systems resulted in the recent years from the confluence of technologies in ESs, distributed systems, dependable systems, and often real-time systems with advances in networking, microcontrollers, sensors, actuators and even artificial intelligence. CPSs must operate safely, securely, efficiently and in real-time, and therefore predictability with regards to timing and security requirements is critical for their development. In order to identify the key issues involved in the development of time-critical and secure cooperative CPSs, in this paper, we first draw a parallel between ESs and CPSs. We then explore the research and look into a number of existing frameworks and techniques devoted to developing time predictable and secure ESs. We conclude that the state of the art work from the ESs community is inadequate for the more complex and open CPSs, as the boundaries of a CPS extend beyond the system network interfaces. Moreover, time predictability in ESs is built upon assumptions about the system and its environment. Shifting from ESs to CPSs, this assumption does not hold anymore, as even if a time-critical CPS is proven to be time predictable at the design time, the predictability of the system

may be jeopardized at run-time due to security threats on the data entering the system via its sensors, networks or even other CPSs. The existing methods and techniques for building time predictable CPSs hold only if the data integrity, authentication and authorization are supported. Therefore, we argue that there is a gap and future work should be devoted to developing frameworks that render security as a prerequisite for time-critical CPSs. We have illustrated our ideas on one cooperative time-critical CPS of autonomous quarry.

ACKNOWLEDGEMENT

The work in this paper is supported by the Swedish Knowledge Foundation (KKS), the Swedish Foundation for Strategic Research (SSF), and the Swedish Governmental Agency for Innovation Systems (VINNOVA) through the projects DPAC and HERO, Serendipity, and DESTINE respectively.

REFERENCES

[1] E. A. Lee and S. A. Seshia, *Introduction to embedded systems: A cyber-physical systems approach*. Mit Press, 2016.

[2] E. A. Lee, "Cyber physical systems: Design challenges," in *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, May 2008, pp. 363–369.

[3] Q. Li and C. Yao, *Real-Time Concepts for Embedded Systems*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 2003.

[4] M. Barr, "Embedded Systems Glossary." http://www.netrino.com/Embedded-Systems/Glossary.

[5] M. Barr and A. Massa, *Programming Embedded Systems*. O'Reilly Media, Inc., 2006.

[6] J. A. Stankovic and K. Ramamritham, "What is predictability for real-time systems?" *Real-Time Sys.*, vol. 2, no. 4, pp. 247–254, Nov 1990.

[7] L. Thiele and R. Wilhelm, "Design for timing predictability," *Real-Time Sys.*, vol. 28, no. 2, pp. 157–177, Nov 2004.

[8] R. Kirner and P. Puschner, "Time-predictable computing," in *Software Technologies for Embedded and Ubiquitous Systems*. Springer Berlin Heidelberg, 2010, pp. 23–34.

[9] D. Grund, J. Reineke, and R. Wilhelm, "A Template for Predictability Definitions with Supporting Evidence," in *Bringing Theory to Practice: Predictability and Performance in Embedded Systems*, ser. OpenAccess Series in Informatics, vol. 18, Dagstuhl, Germany, 2011, pp. 22–31.

[10] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady, "Security in embedded systems: Design challenges," *ACM Trans. Embed. Comput. Syst.*, vol. 3, no. 3, pp. 461–491, Aug. 2004.

[11] D. N. Serpanos and A. G. Voyiatzis, "Security challenges in embedded systems," *ACM Trans. Embed. Comput. Syst.*, vol. 12, no. 1s, pp. 66:1–66:10, Mar. 2013.

[12] J. Jurjens, "Developing secure embedded systems: Pitfalls and how to avoid them," in *Companion to the Proceedings of the 29th International Conference on Software Engineering*, ser. ICSE COMPANION '07, 2007, pp. 182–183.

[13] R. Kissel, *Glossary of key information security terms*. U.S. Dept. of Commerce, National Institute of Standards and Technology, 2006.

[14] M. G. Doyle, How Volvo CE is engineering a quarry run by electric loaders and haulers for big cuts to costs and emissions, 2016, https://www.equipmentworld.com, accessed on September 15, 2018.

[15] L. Sha, T. Abdelzaher, K.-E. A. rzén, A. Cervin, T. P. Baker, A. Burns, G. Buttazzo, M. Caccamo, J. P. Lehoczky, and A. K. Mok, "Real Time Scheduling Theory: A Historical Perspective," *Real-Time Systems*, vol. 28, no. 2/3, pp. 101–155, 2004.

[16] Timing Augmented Description Language (TADL2) syntax, semantics, metamodel Ver. 2, Deliverable 11, Aug. 2012.

[17] "EAST-ADL Domain Model Specification, V2.1.12,," http://www.east-adl.info/Specification/V2.1.12/EAST-ADL-Specification_V2.1.12.pdf.

[18] S. Mubeen, T. Nolte, M. Sjödin, J. Lundbäck, and K.-L. Lundbäck, "Supporting timing analysis of vehicular embedded systems through the refinement of timing constraints," *Software & Systems Modeling*, 2017.

[19] "AUTOSAR Techincal Overview, Release 4.1, Rev. 2, Ver. 1.1.0., The AUTOSAR Consortium, Oct., 2013," http://autosar.org.

[20] N. Feiertag, K. Richter, J. Nordlander, and J. Jonsson, "A Compositional Framework for End-to-End Path Delay Calculation of Automotive Systems under Different Path Semantics," in *CRTS Workshop*, dec. 2008.

[21] S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Support for end-to-end response-time and delay analysis in the industrial tool suite: Issues, experiences and a case study," *Computer Science and Information Systems*, vol. 10, no. 1, 2013.

[22] B. Sun, X. Li, B. Wan, C. Wang, X. Zhou, and X. Chen, "Definitions of predictability for cyber physical systems," *Journal of Systems Architecture*, vol. 63, pp. 48 – 60, 2016.

[23] S. Mubeen, P. Nikolaidis, A. Didic, H. Pei-Breivold, K. Sandstrm, and M. Behnam, "Delay mitigation in offloaded cloud controllers in industrial iot," *IEEE Access*, vol. 5, pp. 4418–4430, 2017.

[24] M. S. Rohmad, H. Hashim, and A. Saparon, "Lightweight cryptography on programmable system on chip: Standalone software implementation," in *2015 IEEE Symposium on Computer Applications Industrial Electronics (ISCAIE)*, April 2015, pp. 151–154.

[25] G. Bansod, N. Raval, and N. Pisharoty, "Implementation of a new lightweight encryption design for embedded security," *IEEE Trans. on Information Forensics and Security*, vol. 10, no. 1, pp. 142–151, 2015.

[26] I. Hiroaki, M. Edahiro, and J. Sakai, "Towards scalable and secure execution platform for embedded systems," in *2007 Asia and South Pacific Design Automation Conference*, Jan 2007, pp. 350–354.

[27] S. Konrad and B. H. C. Cheng, "Real-time specification patterns," in *Proceedings of the 27th International Conference on Software Engineering*, ser. ICSE '05. New York, NY, USA: ACM, 2005, pp. 372–381.

[28] A. Kornecki and J. Zalewski, "Safety assurance for safety-critical embedded systems: Qualification of tools for complex electronic hardware," in *1st International Conference on Information Technology*, 2008.

[29] International Organization for Standardization (ISO), *ISO 26262: Road Vehicles - Functional Safety*, ISO Std., 2011.

[30] C. B. Weinstock, H. F. Lipson, and J. Goodenough, "Arguing security creating security assurance cases," 2014.

[31] P. Johnson, D. Gorton, R. Lagerström, and M. Ekstedt, "Time between vulnerability disclosures: A measure of software product vulnerability," *Computers & Security*, 2016.

[32] R. Melchers, "On the ALARP Approach to Risk Management," *Reliability Engineering and Sys. Safety*, vol. 71, no. 2, pp. 201 – 208, 2001.

[33] S. Girs, I. Šljivo, and O. Jaradat, "Contract-based assurance for wireless cooperative functions of vehicular systems," in *43rd Annual Conference of the IEEE Industrial Electronics Society (IECON)*, Oct. 2017.

[34] M. Paulitsch, H. Ruess, and M. Sorea, "Non-functional avionics requirements," in *Leveraging Applications of Formal Methods, Verification and Validation*. Springer Berlin Heidelberg, 2008, pp. 369–384.

[35] A. Löfwenmark, Timing Predictability in Future Multi-Core Avionics Systems, Licentiate Thesis, Department of Computer Science and Information Systems, Linköping University, 2017, ISBN: 978-91-7685-564-5.

[36] T. Feld, A. Biondi, R. I. Davis, G. Buttazzo, and F. Slomka, "A survey of schedulability analysis techniques for rate-dependent tasks," *Journal of Systems and Software*, vol. 138, pp. 100 – 107, 2018.

[37] North Atlantic Treaty Organization, "Engineering for system assurance nato in programmes," 2010.

[38] R. Weaver, J. Fenn, and T. Kelly, "A pragmatic approach to reasoning about the assurance of safety arguments," in *8th Australian Workshop on Safety Critical Systems and Software*, 2003.

[39] I. Šljivo, E. Lisova, and S. Afshar, "Agent-centred approach for assuring ethics in dependable service systems," in *13th IEEE World Congress on Services*, Jun. 2017.

[40] C. Haley, R. Laney, J. Moffett, and B. Nuseibeh, "Security requirements engineering: A framework for representation and analysis," *IEEE Transactions on Software Engineering*, vol. 34, no. 1, pp. 133–153, Jan 2008.

[41] J. McDermott and C. Fox, "Using abuse case models for security requirements analysis," in *Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99)*, Dec 1999, pp. 55–64.

[42] C. Raspotnig, P. Karpati, and V. Katta, "A combined process for elicitation and analysis of safety and security requirements," in *Enterprise, Business-Process and Information Sys. Modeling*, 2012, pp. 347–361.

[43] T. Gu, M. Lu, and L. Li, "Extracting interdependent requirements and resolving conflicted requirements of safety and security for industrial control systems," in *1st International Conference on Reliability Systems Engineering (ICRSE)*, 2015, pp. 1–8.

[44] G. Howard, M. Butler, J. Colley, and V. Sassone, "Formal Analysis of Safety and Security Requirements of Critical Systems Supported by an Extended STPA Methodology," in *IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*, 2017, pp. 174–180.

[45] A. Iliasov, A. Romanovsky, L. Laibinis, E. Troubitsyna, and T. Latvala, "Augmenting event-b modelling with real-time verification," in *1st International Workshop on Formal Methods in Software Engineering: Rigorous and Agile Approaches (FormSERA)*, June 2012, pp. 51–57.