Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Review

Vulnerabilities and safety assurance methods in Cyber-Physical Systems: A comprehensive review

Victor Bolbot[*], Gerasimos Theotokatos, Luminita Manuela Bujorianu, Evangelos Boulougouris, Dracos Vassalos

Maritime Safety Research Centre, Department of Naval Architecture, Ocean and Marine Engineering, University of Strathclyde, Henry Dyer Building, 100 Montrose Street, Glasgow, Scotland G4 0LZ, UK

ARTICLE INFO

Keywords:
Safety
Cyber-Physical Systems
Complexity sources
Safety assurance methods

ABSTRACT

As Cyber-Physical Systems (CPSs) are a class of systems advancing in a number of safety-critical application areas, it is crucial to ensure that they operate without causing any harm to people, environment and assets. The complexity of CPSs though, render them vulnerable and accident-prone. In this study, the sources of complexity are meticulously examined and the state-of-the-art and novel methods that are used for the safety assurance of CPSs are reviewed. Furthermore, the identified safety assurance methods are assessed for their compatibility with the technical processes during the system design phase and the methods effectiveness on addressing the different CPSs sources of complexity is investigated. Advantages and disadvantages of the different safety assurance methods are also presented. Based on the results of this review, directions for the safety enhancement of CPSs and topics for future research in the area of CPSs safety are provided.

1. Introduction

The continuous research and initiative projects developments have resulted in new types of systems implementing functionalities unforeseen in the past, thus requiring the development of new methods to ensure their safety [2]. One category of such systems is the Cyber-Physical Systems (CPSs) [2, 3].

The term *Cyber-Physical Systems* was first introduced through a series of discussions between the members of academic staff at Berkeley University in 2006 [4]. CPSs can be defined as systems which collect the information from the physical environment using sensors and communication channels, analyse it using controllers and affect the physical environment and relevant processes through actuators to achieve specific goal during their operation [4]. Compared with the mechatronic systems, CPSs consist of integrated components/subsystems and can be also interconnected with other CPSs [5].

CPSs have been used and advanced in a number of application areas including automotive systems, avionics systems, defence systems, manufacturing systems, process control systems, traffic control systems, robots, smart medical devices, smart home applications and maritime systems [4–7]. The CPSs could be classified into three large categories, although there may be an overlapping between these categories:

- Autonomous CPSs (ACPSs), which include the industrial and advanced robots and autonomous navigation systems [8].
- Networked CPSs or Cyber-Physical Systems of Systems (CPSoSs), which are large, distributed systems, for example, the smart grids and the railway systems [7].
- Industrial Automation and Control Systems (IACSs), which are used to control the physical processes in the oil and gas industry, nuclear industry, etc. [9].

Safety is a key requirement for the CPSs, where safety can be comprehended as 'the freedom from those conditions that can cause death, injury, occupational illness and damage or loss of equipment or property' [10] or as 'the freedom from unacceptable risk of physical injury or damage to the health of people, either directly, or indirectly as a result of damage to the property or to the environment' [11].

In this respect, it can be a special challenge to design safe CPSs as they belong to the category of *complex* systems [3, 4], where *complexity* expresses the increased unpredictability of the system's behaviour, which may jeopardize their safe and reliable operation [3, 12]. The unpredictability in CPSs coexists with tight interactions between their components, especially between the cyber and the physical parts, allowing little slack in their performance [13–15]. The combination of tight interactions with complexity is the perfect recipe for accidents, as

[*] Corresponding author.
E-mail address: victor.bolbot@strath.ac.uk (V. Bolbot).

https://doi.org/10.1016/j.ress.2018.09.004
Received 7 March 2018; Received in revised form 30 August 2018; Accepted 6 September 2018
Available online 07 September 2018

small deviations, due to a component degradation or unpredicted environmental disturbances can lead to a new emergent behaviour [16]. In this respect, complexity negatively affects CPSs, rendering them vulnerable, where vulnerability is a ‘weakness of a product or a system that may lead to the destruction if exposed to a threat’ [17].

Accidents in modern systems may have more severe consequences in terms of financial loss, human losses and environmental damage compared to systems used in past [2]. Furthermore, the public's tolerance to accidents today is much stricter than it was in the past, thus increasing the pressure on public authorities as well as the CPSs designers and operators [2]. This necessitates the proper identification of the complexity sources in CPSs and the application of appropriate safety assurance techniques during the CPSs design for their control, a proper understanding of limitations and advantages of each method along with the derivation of research directions to enhance the CPSs safety assurance methods.

A considerable number of studies focused on the state-of-the-art and novel methods and standards for safety assurance. Aizpurua and Muxika [18] reviewed the state-of-the-art safety assessment techniques, used for the design of systems in general. Kriaa et al. [9] provided an overview of the methods for a combined assessment of the safety and security of IACSs. De la Vara et al. [19] offered an overview of the available safety standards. Martin and Gorschek [20] reviewed the methods available for deriving requirements in safety-critical systems, without providing a detailed analysis on the verification activities and without a specific focus on CPSs. Naier et al. [21] carried out a systematic literature review on the evidence required for the safety certification, but they did not focus on particular methods, despite the fact that these methods are necessary for evidence generation in the system safety case.

Quite a few studies referred to the safety challenges and the available methods for CPSs. Engell et al. [7] made a reference to the specific CPSoSs challenges without explaining how these challenges are related to the safety and the methods that can be used. Zio [17] conducted a review of characteristics of complexity and some of the available methods for the risk assessment with respect to the requirements for the critical infrastructure, without providing a specific connection to CPSs. Dependability issues have been also reviewed and addressed in connection with CPSs in Guiochet et al. [8]. This study though was limited to ACPSs, without providing details and neglecting a number of methods applicable to other CPSs. Wolf and Serpanos [22] have also addressed the interconnection between safety and security in CPSs and reported a number of available methods. They omitted, however, a number of methods used by the safety engineers and did not provide an overview of the sources of complexity in CPSs.

The above discussion reveals the lack of a comprehensive and systematic review of the CPSs sources of complexity and the available methods for the CPSs safety assurance during their design, which is covered herein. The present review unique contribution is based on combining the concepts stemming from safety engineering, complexity theory and CPSs. Another distinct contribution is the discussion on the applicability of these methods to the different CPSs categories by addressing the sources of complexity in CPSs during their design. This review has a wider coverage than the previous studies and in this way, it can be an effective starting point for researchers in the CPSs safety area and give useful insights and directions for the CPSs discipline professionals.

In the next section, the main sources of complexity leading to accidents and hazardous situations in CPSs are classified and analysed. Subsequently, the available safety assurance standards and methods are classified and shortly described accompanied with recent and distinct examples of their applications to CPSs and mechatronic systems. Next, the compatibility of the methods with different system engineering technical activities, the effectiveness of these methods in addressing the sources of complexity in CPSs and the challenges related to the application of the safety assurance of CPSs are discussed and analysed. Based on the results of the discussion, directions for research and better practices in CPSs are proposed. Finally, the main findings of this study are summarised.

2. The sources of complexities in CPSs

2.1. Complexity types and their time variation

The *internal* systems complexity can be classified as structural, dynamic and organisational according to pertinent literature in the context of the system design and development [3, 17]. *Structural* complexity is a characteristic of the systems that consist of a large components number and have unpredictable interactions among their components [3, 17]. *Dynamic* complexity exists when the comprehension of the system behaviour and system dynamics in time is impeded [3]. *Organisational* complexity refers to the organisation of the group responsible for the design and the operation of the complex system [3]. An overview of sources of the structural and dynamic complexities leading to a hazardous situation or an accident is shown in Table 1.

The three types of complexity are not fully independent, and they are interrelated. Dynamic complexity is dependent on the structural complexity and can be caused by changes in the system environmental and operational conditions [17], or changes in the system and the associated management system with time [23]. Organisational complexity is primarily the result of the structural and secondarily the dynamic complexity [3].

All three complexity types are manifested during the actual operation of the system. Although the structural complexity is part of the system from the design phase, its impact on the system safety becomes obvious during the system operation. Dynamic complexity is introduced with time, increasing the complexity of the system, when the system is modified or altered under the environmental influence or faces unpredictable environmental disturbances. Organisational complexity is also manifested and varies with time, as the organisation of design and management team is flexible.

The systems complexity is relative and reduces with time as more knowledge is acquired for the system, its components and the components interactions either during the design or operation phases [12, 23]. In this respect, epistemic uncertainty, which is related to the lack of knowledge about the system is also reduced with time. The safety of a CPS can be ensured if its behaviour is handled and well understood during its design phase development. As knowledge is accumulated about system interactions allowing to effectively control them, it is easier to intervene during the design phase in order to develop a safer system. In other words, with respect to safety, it is important to drive the available information about the system from the area of “unknown unknowns” to the area of “known knowns” [1] as also shown in Fig. 1. This is implemented with the safety assurance methods which are discussed in the section 3.

Table 1
Sources of complexity in different CPSs categories.

Complexity types	Source	ACPSs	CPSoSs	IACSs
Structural	Heterogeneity		v	v
	Interoperability		v	v
	Connectivity	v	v	v
	Software-intensiveness	v	v	v
	Humans in the loop	v	v	v
Dynamic	Evolution in time	v	v	v
	Dynamic reconfiguration	v	v	v
	Autonomous decision making	v		
Organisational	The complexity of the design and operation team	v	v	v

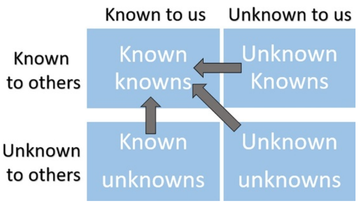

Fig. 1. The problem of complexity handling. Adapted from [1].

2.2. The sources of structural complexity in CPSs

The complexity in CPSs can be attributed to the fact, that many CPSs are *heterogeneous* systems consisting of a considerable number of different components, such as mechanical, electrical, control and networking, which cooperate for achieving the desired system behaviour [24–26]. The heterogeneity introduces problems as it impedes the understanding of the system interactions, especially between the cyber and physical parts, leading to increased unpredictability [27, 28]. The heterogeneity requires the involvement of engineers from different disciplines, which contributes to the increase of the organisational complexity.

The *interoperability* can be described as the ability of the systems/components to work together for the achievement of the common goal in a System of Systems (SoS)/System [4, 6]. The interoperability in CPSs can be viewed at a component level, as a result of interconnecting mechatronic subsystems and at a higher level as a result of interconnecting a number of CPSs [5]. Although this adds new functionalities to the system, it enhances the complexity of CPSs due to the fact that the networking elements of CPSs allow obtaining data from other CPSs or mechatronic systems, thus increasing the number and types of interactions [12, 29, 30]. Delays in delivering information and loss of information from one computing element to the other can lead to the realisation of significant hazards [22, 30]. Hidden defects of a component may expose the interconnected systems to cascading failures [17]. In this respect, it is important to ensure that the CPSs and their components are safely integrated [31].

The interoperability of the systems very often comes along with their interconnection, but this leads to problems related to *cybersecurity* [22, 32, 33]. The relationship between the safety and security in CPSs can be characterised as a relationship of conditional dependence, as cyberattacks can exploit deficiencies in the defence systems, protocols or human recklessness and directly affect the integrity or availability of the data and control systems [9, 22]. An example of compromising the system safety as a result of a cybersecurity breach was the case of the Stuxnet worm, which successfully destroyed the centrifuges pumps at Iranian nuclear facilities, impeding the process of uranium enrichment [9].

CPSs are *software-intensive* systems, as the software is a primary entity of the CPSs cyber part [4]. However, errors in the employed software are of a great concern for the designer [31]. As the functionalities of a CPS increase the number of interactions and errors in software also increases and it is more likely a software error to lead to undesired behaviour [22]. An accident can be also caused by a fully functional software, due to improper handling of the system requirements [34, 35]. A noticeable example of an accident caused by the software flaws is an accident with the radiation therapy machine Therac-25, which led six patients to death due to the radiation overexposure [36].

Many of the CPSs involve humans in charge of the decision-making at a high level of control [2, 7]. Inadequate communication between people and machines, due to the lack of proper situational awareness can lead to an accident [2, 37]. This problem can be also caused by overreliance on the technology, as it was revealed by the crash of the Turkish Airlines flight in 2009 [38]. The human factors can be seen as an additional component to CPSs adding interactions and non-linearity.

2.3. The sources of dynamic complexity in CPSs

As several classes of CPSs are planned to operate for a very long period and during their operation, it may be required to add new functionalities or to improve their components performance. Some CPSs also follow an evolutionary design process with new updated versions being constantly launched on the market. In addition, a deterioration of the system components performance can result in unpredictable interactions in the system. Practically it means that the system components are *evolving in time* [7, 19, 39]. Yet, it may lead to hazardous situations, as an improper software or hardware update or component deterioration, through tight interactions may lead to an inappropriate behaviour of these systems. A typical example of such an accident was the Ariane 5 crash, where the inertial reference software that was installed on Ariane 4 was also installed on Ariane 5, but, as the trajectory of Ariane 5 was changed it led to new inappropriate interactions and subsequently accident [2].

Dynamic reconfiguration and *adaptability* can be defined as 'the capability of a system to change its state by adjusting its own configuration in response to different circumstances in the environment' [4]. The dynamic reconfiguration and adaptability can be also expressed in terms of fault tolerance [7, 40]. This ability of CPSs is to a great extent supported by intelligent prognosis and diagnosis techniques and allows CPSs to fail safely, avoiding accidents [7]. As a consequence, it is required to ensure that these reconfiguration functions, diagnostics and prognostics work properly taking into account the complexities of the physical processes and the system evolution in time [31, 41].

Autonomous decision-making is a property of robotic systems or autonomous CPSs and can be defined as an 'ability to undertake decisions and to perform simple and complex tasks by considering the changes in the environment without human intervention' [42]. It is required that the decision made by CPSs do not lead to an accident, as in a case referred by [8]. In addition, as the autonomous decision-making CPSs rely on data mining and machine learning algorithms, this behaviour can be quite unpredictable. This is owed to the challenges with verification of these algorithms [43–45].

3. Safety assurance methods for CPSs

3.1. Safety assurance concepts

Safety assurance can be defined as the process that includes 'all planned and systematic actions necessary to provide adequate confidence that a product, a service, an organisation or a functional system achieves acceptable or tolerable safety' [46], and is the path that has to be followed in order to develop a safety case [47]. The *safety case* can be described as "the argument that the safety requirements for an item are complete and satisfied by evidence compiled from work products of the safety activities during development" [48]. The main purpose of the safety case is to demonstrate that the systems are safe for the intended use and can be also used to present that appropriate mitigation of risks is implemented to an acceptable level [47].

The activities necessary for the development and management of the safety case through the different stages of the system lifecycle can be split into the following categories of activities based on the information provided in standards including IEC 61508 [11] and MIL-STD-882E [10]:

- Management of the design process to achieve adequate safety.
- Hazard identification and analysis.
- Risk analysis, risk evaluation and safety requirements elicitation.
- Risk control measures identification, cost-benefit analysis and application in the system design.
- Verification and validation activities.
- Presenting the results of the analysis through the developed safety case.

Table 2
Activities and methods used during safety assurance of CPSs.

Process management	Hazard identification and analysis	Risk analysis	Risk control measures	Verification and validation	Development of safety case
SAE ARP 4761	Traditional	Probability theory	Designing out the hazards	Fault injection	Claim Argument
ISO 14971	Systemic		Safety devices	Model checking	Evidence
IEC 61508 based	Human Reliability Analysis	Fuzzy theory	Fault tolerance techniques	Automated theorem proving	Goal Structuring
IEC 62508		State-of-the-art		Testing	Notation etc.
MIL-STD-882E	Failure logic synthesis and analysis	Modified methods			
TRL framework			Operational procedures and training etc.	Runtime verification	
ISO 31000					
Barrier management					

- Monitoring and updating the safety case throughout the system lifecycle.

These categories of activities constitute the necessary framework for the classification of the state-of-the-art and novel developments in the area of CPSs safety. The implementation of risk control measures is outside the scope of this study, as it is specific for each application area. The research on available safety case methods is also outside the aims of this study as their purpose is effective communication of results derived through safety assurance process [21]. An overview of the methods and activities employed for the CPSs safety assurance is presented in Table 2.

3.2. Safety assurance management standards

In total 47 safety standards have been referred to be used by safety engineers [19]. Herein only the most important of these standards are broadly discussed. Such a discussion is an inherent part of every review paper and for reasons of consistency is repeated below.

The standards related to the safety assessment can be classified into two major categories: the standards specifying the procedures for the risk management during operation and the standards supporting the safety guided design as illustrated in Table 3. This distinction is made on the basis that operational standards relate to the socio-technical systems and their operation.

Comparing ISO 31000, ISO 14971 and MIL-STD-882E, it can be easily observed that these standards follow the same basic steps, in specific the hazard and risk identification, the risk analysis, the risk treatment and the risk evaluation. The differences can be found in the documentation procedures and in the proposed methods for each step of the risk management. IEC 61508 and ARP 4761 have a totally different approach as they concentrate more on the design procedures than the other standards. ARP 4761 is more aligned with system engineering activities and follows a top-down approach for the safety assessment, whilst IEC 61508 is used for the safety requirements generation and verification. IEC 62508 focuses on the human-machinery interface and has a different structure and approach compared with the other standards. The Technology Readiness Level (TRL) framework is harmonised to a large extent with the waterfall system engineering life cycle approach, as it used to support the innovative systems design [52].

Table 3
Safety standards.

Standard	Industry	Application
SAE ARP 4761 [49]	Aviation	Design
ISO 14971 [50]	Biomedical	Design
IEC 61508 [11]	General, with specific standards in each industry.	Design
IEC 62508 [51]	General, with focus on the human-machinery interface	Design
MIL-STD-882E [10]	Military	Design
TRL framework [52]	General, design of new systems	Design
ISO 31000 [53]	General, management of systems	Operation
Barrier management [54]	Oil and gas	Operation

3.3. Hazard identification methods

Hazard identification and *analysis* is the process of defining all possible scenarios or sequences of events, which can lead to a realisation of a hazard. The hazard identification methods are described in the following subsections and an overview is also provided in Table 4.

3.3.1. Traditional hazard identification methods and their modifications

The techniques including Fault Tree Analysis (FTA), What-If, Failure Modes and Effects Analysis (FMEA), HAZard and OPerability studies (HAZOP), Event Tree Analysis (ETA), Layer protection analysis (LOPA), Process Hazard Review, Preliminary Hazard Analysis (PHA) are well established methods for hazard identification and analysis [55]. The majority of these methods are described in ISO 31000 relevant standards [53]. A short description of the application of their modified versions or combinations is given below.

Gamble et al. [40] used the HAZOP and Software Hazard Analysis and Resolution in Design (SHARD) to identify the faults in a simple robot and ChessWay transporter. Subsequently, the most critical faults were simulated in the virtual environment and their impact on the system was assessed. In the work of Soubiran et al. [56], the safety guided design of a train system was realised by employing PHA and FMEA methods. Peeters et al. [57] combined FTA and FMEA in a recursive manner and applied to an additive manufacturing system.

Security implications on the safety of CPSs have been addressed in the work of Sabaliuskalte and Mathur [58], where a design process was implemented which combined safety and security life cycles retrieved from the safety and security standards based on Attack Trees along with

Table 4
The hazard identification methods.

Traditional hazard identification methods	Systemic methods	Failure Logic Synthesis and Analysis (FLSA) methods	Human Reliability Analysis (HRA) methods
Traditional methods	STPA	AADL	First generation
Joint safety and cybersecurity approaches		HiP-HOPS	Second generation
		SysML based CFT	Third generation
Hazard identification methods for robots			
		SEFT	
		DSM	

Fault Trees. Sabaliauskaite and Adepu [59] improved the six-step model with information flow diagrams to consider the exchange of information in networks. Schmittner et al. [33] applied a Failure Mode, Vulnerabilities and Effects Analysis (FMVEA) and Combined Harm Assessment of Safety and Security for Information Systems in the automotive domain to an over the air update system of a car.

A number of traditional approaches have been enriched to consider the complexity of human robots interactions. An Environmental Survey Hazard Analysis, which is a variant of PHA was proposed for autonomous mobile robots by Dogramadzhi et al. [60] to ensure the completeness of identified hazardous scenarios. Guiochet [61] proposed a new method for hazard identification of interactions between humans and robots based on Unified Modelling Language (UML) and the classical HAZOP.

3.3.2. Systemic approaches

System-Theoretic Process Analysis (STPA) is a hazard identification method in which the system is represented by a control structure and the hazardous conditions are generated by the lack, the presence or the improper timing of the control actions [2]. The process of analysis is deductive, followed by identification of causal factors for unsafe control actions.

Rokseth et al. [62] applied the STPA to a Dynamic Positioning System (DPS) of a generic vessel, which is one of the CPSs on ships. The comparison with an adjusted form of FMEA showed that STPA provides a much broader understanding of hazards in the system, although its use has been suggested as supplementary to FMEA analysis. Sulaman et al. [63] compared FMEA and STPA by applying them to the forward collision avoidance system of a vehicle. The results showed that there were commonalities in the hazards identified by FMEA and STPA, but also both FMEA and STPA outcome into unique hazards, not found by the other method. Friedberg et al. [64] proposed the use of a modified STPA method for hazard analysis taking into account both safety and security requirements. In another work [65] STPA was integrated with a six-step model for the safety assessment of an autonomous vehicle.

3.3.3. Failure logic synthesis and analysis

Failure Logic Synthesis and Analysis (FLSA) [66] or Failure Logic Modelling [67, 68] are model-based methods for safety assessment. As such, FLSA methods are based on system models extended with fault model for capturing the components malfunctions and interactions between faults [49, 67, 68]. FLSA methods can be used to derive automatically Fault Trees and FMEA tables using appropriate algorithms [69].

The major FLSA methods include Failure Propagation and Transformation Notation (FPTN) techniques [70], Hierarchically Performed Hazard Origin & Propagation Studies (HiP-HOPS) [71], Architecture Analysis and Design Language (AADL) [72], Component Fault Trees (CFT) [73], State-Event Fault Trees (SEFT) [66]. Several other approaches use System Modelling Language (SysML) [74] and Design Structure Matrices (DSM) [75]. Some of the identified applications of these methods to mechatronic systems and CPSs are given below.

Dehlinger and Dugan [76] used the model developed in AADL to analyse the reliability of digital feed-water control system in a nuclear plant. Delange et al. [32] relied on AADL to model and assess safety-critical partitioned systems. For this purpose, they adopted semantics of AADL to model the partitioned architecture, security levels and the faults. In [74], FLSA was applied to the flight control system of an aircraft. In particular, the analysis was implemented as part of the general design framework. The requirements definition and analysis were supported by SysML block definition diagrams, whilst the system was represented by SysML activity diagrams.

3.3.4. Human reliability analysis

The objective of Human Reliability Analysis (HRA) methods is to identify and assess the hazards and the risk to the system coming from the human actions with negative consequences [77]. HRA methods can be classified into three major categories: the first, second and third generation methods [78]. As the CPSs seem to be vulnerable in digital human-machine interface link, after a short reference to state-of-the-art methods, the emphasis has been given to the novel methods that can address this problem.

The first generation methods consider the human error dependent primarily on the task implemented and the impact of the environment can be considered as having secondarily contribution. State-of-the-art first-generation methods include the Technique for Human Error Prediction, the Accident Sequence Evaluation Program and the Human Cognitive Reliability [17, 77, 78]. Recently, Petrillo et al. [79] proposed a novel model, called Human Error in Industrial Emergency Conditions for assessment of human performance in terms of safety under emergency condition by considering more carefully external factors through a combination of methods with application to a control room of a petrochemical plant. Li et al. [37] estimated the situation awareness of a human operator in a digital nuclear power plant by combining fuzzy logic and Analytical Hierarchical Process.

According to the second generation methods, the environmental context, the cognitive operations of humans and commission errors have greater importance in the occurrence and the determination of the accident path. The major difference between the first and second generation methods can be found in the treatment of the contextual factors [78]. Examples of second generation methods include the Cognitive Reliability and Error Analysis Method and A Technique for Human Error Analysis (ATHEANA) [78]. Olivares et al. [77] proposed a new method with the reduced dependency of analysis on expert judgement, by incorporating cognitive mechanisms and considering the system information available to the operator.

The third generation methods depend on simulation in a virtual environment and therefore consider more effectively the involved dynamic phenomena [78]. State-of-the-art third generation methods include the Accident Dynamics Simulator-Information Decision and Action in Crew in nuclear industry [80] and the Man-Machine Integration Design and Analysis System in the aviation and aerospace industry [81]. In a more recent work, Petrillo et al. [82] combined other two methods, the Simulator for Human Error Probability Analysis [83] and Performance Shaping Factors dependency model developed by Boring [84] to an emergency scenario in the control room of petrochemical plant. This merging was required to assess the effect of fatigue on control room operator. Fan et al. [85] developed a simulation platform to assess the impact of human errors on the human-machinery interface on the accidental scenarios in a nuclear power plant.

3.4. Risk analysis

Risk Analysis (RA) is ‘a comprehensive, structured and logical analysis method aimed at identifying and assessing risks in complex technological systems for the purpose of cost-effectively improving their safety and performance’ [86]. A number of methods can be employed to estimate the risk and risk metrics including Bayesian Belief Networks (BBN), Fault Trees, Event Trees, Agent-Based Models, Reliability Block Diagrams (RBD), Markov Models (MM), Monte Carlo simulations, Petri nets, Boolean logic Driven Markov Processes [18]. FLSA methods can be also used for the purposes of RA, by incorporating failure rates in the analysis. Examples of recent applications and the novel or distinct methods are given in the next paragraph.

Gonçalves et al. [87] presented a safety assessment of Unmanned Aerial Vehicle based on Petri Nets. The operation of the system was represented by Petri nets and simulated to verify that the undesired events are avoided. Aldemir et al. [88] used a Markov/cell-to-cell mapping technique and a dynamic flow graph method, which are two system logic modelling methodologies, to implement an RA to the digital feed water control system of a nuclear power plant. Abdo et al. [89] considering the increased probability of cyberattacks, proposed to analyse the effect of cybersecurity on the overall risk level of an industrial unit by combining the attack trees with the classical Bow-Tie approach. Qiu et al. [90] implemented an RA of a car navigation system by using Valuation-Based System description. Wang et al. [91] used the Monte Carlo simulations of the system model to assess the impact of the cyberattacks on the safety of a nuclear plant. Wu and Zheng [92] applied timed coloured Petri Nets for the safety assessment of the railway level crossing control system.

3.5. Verification & validation

3.5.1. Model checking

Model checking is ‘an automated technique that, given a finite-state model of a system and a formal property, systematically checks whether this property is held for a given state in that model’ through reachability analysis [36]. Model checking can be employed for the purposes of the fault injection as well [69]. Some examples of the application of model checking to CPSs and mechatronic systems are listed below.

In Lee et al. [31], a medical CPS were modelled in a virtual environment and the system behaviour was checked to satisfy all the safety properties using a model checker. Zhang et al. [93] applied a hybrid interface automata to model two cars. The novelty of that approach was that the genetic algorithm was adopted for the verification purposes by employing unconstrained dynamic programming. Bresolin et al. [94] implemented a formal verification of an autonomous medical robotic system with a focus on puncturing function by abstracting the system with hybrid automata. Paoletti et al. [95] employed statistical model checking tool for verification of pacemaker performance under presence of faults and different physiological conditions. Cavalcanti et al. [96] have extended an existing formalism for modelling heterogenous robots swarms and verified their abilities using an established model checker. Bohlender and Kowalewski [35] proposed a new modelling framework to identify inconsistencies in process variables saved by CPSs during their restarting procedures, which may lead to safety implications.

3.5.2. Fault injection

Fault injection is a dependability technique, in which the system behaviour is observed under the presence of faults to guarantee that the system behaviour will be sound despite the presence of malfunctions in the system [40, 67, 97].

A number of fault injection methods was based on the system abstraction. Bozzano et al. [98] implemented the fault injection technique for assessing a satellite platform by expressing the system using an AADL graphical notation [72], and the fault injection was implemented in the COMPASS toolset [99] using model checking. Li et al. [100] used the AltaRica language [101] to model the fault propagation in an aircraft power plant. They followed a process quite similar to the steps of the ARP4754 standard. Dal Lago et al. [102] applied the fault injection to the thermostat model. The system was modelled using the xSAP safety platform [103] and system responses to thermostat timing faults were investigated. Leupolz [104] implemented a safety assessment of a tunnel height control system, by describing its normal and faulty behaviour in S# modelling and analysis framework [104].

Other approaches employed more detailed models. Clarke and Zuliani [105] used a stochastic model checking, the importance sampling and the cross-entropy methods for the verification of the properties of a gasoline station fuel control system. In [14], the Functional Failure Identification and Propagation model was used to identify the impact of common cause failures such as flooding or fire in a nuclear reactor coolant supply system.

Lastly, fault injection can be applied to the built system, although there is a risk of incurring a system damage [97]. Alemzadeh et al. [106] applied the fault injection to the robotic surgical platform. The faults were identified using STPA, injected in control part of the system and their impact assessed under realistic conditions.

3.5.3. Automated theorem proving

Whilst the model checking is an inductive method, the *automated theorem proving* is a deductive method used for the formal verification of CPSs through the application of axiom and reasoning rules [15]. The automated theorem proving has been applied to air and train traffic control systems, automotive systems, mobile robot navigation and surgical robot systems [107]. Some of the applications of theorem proving methods to CPSs are described below.

Petnga and Austin [25] applied the automated theorem proving to a smart car control system, approaching a yellow light using three different models to assess its temporal and real-time properties. Sanwal and Hasan [108] applied a higher-order-logic theorem prover for analysis of a harmonic oscillator, represented using a second-order homogeneous linear differential equations. Hulette et al. [109] used a thermostat hybrid model to verify the temperature bounds and highlighted the delay impact of the analogue-digital converter on the system performance. Ivanov et al. [110] proposed a framework for formalising the semantics for CPSs safety verification through a theorem-prover environment based on predicates on system executions.

3.5.4. Testing

Testing is an activity with the primary purpose to verify whether the set of system requirements are satisfied by conducting a number of *test cases*, where a test case can be described as a set of preconditions, inputs and expected results [111]. State-of-the-art methods are described in the next paragraph, while in the following paragraphs reference is given to the recent applications and novel approaches.

The software part of CPSs can be tested using either the structure-based (white-box) or the specification-based (black-box) techniques [111]. A special type of testing for CPSs is the Model-Based Testing (MBT), in which models are used to generate the requirements, the test inputs and expected results [111, 112]. Conformance testing is applied in the MBT context to ensure that the properties of a detailed model are also held in a simpler model which was used for the analysis [112]. In Model-in-the-Loop (MIL), Software-in-the-Loop (SIL), Processor-in-the-Loop (PIL) and Hardware-in-the-Loop (HIL) testing developed software is tested using detailed simulations [113].

Some of the approaches focused on testing the CPSs hardware controlling physical processes. Martins-Filho et al. [113] applied PIL and HIL testing of a satellite attitude control system allowing in this way more realistic assessment of the implementation of the controller. Hansen et al. [114] demonstrated an application of MIL and SIL testing to a car window control system based on block diagrams developed in a simulation environment. Claessen et al. [115] proposed a new logic for the hardware testing to improve the quality of countermeasures identified by tests by measuring the severity of failures. Matinnejad et al. [116] proposed a new method for MBT of CPSs in a simulation environment with the use of a test prioritisation algorithm based on coverage and diversity criteria.

Other approaches instead focused on testing the CPSs decision-making functions. Mullins et al. [117] have implemented a simulation supported black-box testing of the unmanned underwater vehicle enhanced by adaptive search algorithms and unsupervised clustering techniques to reduce the number of test cases. Tian et al. [44] proposed an automated tool for the verification of deep neural networks enabled autonomous driving functions for improving the test coverage using an estimation of neuron coverage and synthetic images.

3.5.5. Runtime verification

As some of the safety properties of the CPSs may not be thoroughly verified and tested during the design and building phases, it may be beneficial to conduct verification during operation [118]. *Runtime verification* can be characterized as a ‘lightweight formal method that attempts to verify that execution traces (whether at runtime or offline from logs) conform to a given specification’ [119].

Kane et al. [119] applied runtime verification to an adaptive cruise control of a vehicle based on HIL testing using partial oracle tests. Online verification of CPSs has been also implemented by Mitsch et al. [118] to ensure the safety of a water tank system. This approach was achieved by comparing the real-time system behaviour with that of the system model used for verification. Habermaier et al. [120] employed a suitable framework for runtime verification of a production cell with separation of faults into different classes for reducing the computational cost.

3.6. *Integrated approaches*

Each safety-related activity focuses on ensuring safety at a specific system design stage and or by considering specific hazards in the system. Hazard identification deals with accident scenarios development, risk assessment aims at predicting the accident scenarios occurrence, while verification activities are applied to reduce the frequency of hazardous scenarios. In this respect, there is no surprise that integrated approaches for the safety assurance of CPSs attracted interest in previous studies.

A number of approaches focused on the integration of hazard identification or similar methods and verification activities. Abdulkhaleq and Wagner [121] have combined the STPA and the model checking for ensuring safe software properties for an adaptive cruise speed control system. Rokseth et al. [122] used the STPA for deriving test objectives for the ship power management system. Blackburn et al. [123] used FTA and FMEA to derive the requirements for the verification of a robot collision avoidance system through testing and theorem proving. Zhang et al. [124] proposed to incorporate belief, uncertainty and measurement profiles to enhance MBT with application to a smart home and the logistics of a manufacturing system. Finally, Sharvia and Papadopoulos [125] combined FLSA with model checking for the safety assessment of the brake by wire system.

Several other approaches focused on the enhancement of HRA. Kaber and Zahabi [126] enhanced System Hazard Analysis (SHA) to account also for the human failures in the system controlled by the operator. For this purpose, the HRA methods and the reliability methods were combined to identify the overall reliability of the system in the SHA. Subsequently, the generated software code was checked for its equivalence with the developed model. Kim et al. [127] instead focused on the operator erroneous actions under the presence of malicious cyberattacks, impacting the availability of the safety systems on a nuclear plant. In this case, the system was analysed using Fault Trees and then the ATHEANA method was used to refine the human operator errors, which were assumed to be driven by cyberattacks. Jiang et al. [128] investigated the reliability of the human-machine interface of a nuclear plant by employing Bayesian network theory, experiments and cognitive process model. Ham and Park [129] used classification and regression trees to analyse the data coming through operation to estimate more carefully the probabilities of human errors in a nuclear plant.

Other researchers focused on how to combine and improve the verification techniques. Seceleanu et al. [130] combined an MBT and model checking for the verification of a wind turbine safety properties. This was implemented to ensure greater coverage of the potential state spaces. Ge et al. [131] used an integrated approach based on S3 software [131], for the safety assurance of a robot protection system. Based on the set of the safety requirements, a model of the system was used to support model checking and test scenarios generation techniques. Nguyen et al. [39] used inference tools to identify mismatches in the specifications for the physical and control parts of direct current power converters and an automotive fuel control system, which can be used as inputs to model checking tools. Araujo et al. [132] proposed the use of model checking to specify the soundness criteria for CPSs model-based conformance testing.

4. Analysis of the methods used for safety assurance in CPSs

4.1. *Applicability to different processes of system engineering during design*

A number of approaches used for the safety assurance in CPSs have been described in the previous sections. However, not all of them are applicable at the same sCPSs design stage. Each method requires a different amount of information. This affects the effectiveness of each method implementation, as the methods are dependent on the available data. The purpose of this section discussion is to investigate the applicability of the methods in the various system engineering processes and to identify the methods that can be effectively applied in each process.

The system engineering approach can be split into six life cycle stages and fourteen technical processes [133]. However, the primary technical processes applied during the design phase are the requirements definition, the architecture definition, the design definition, the system analysis as well as verification and validation processes. The system requirements definition, the system architecture definition and the system design definition process are applied in sequence, whereas the other activities are applied in parallel to the other technical processes. In this respect, these processes were used as a basis for the methods classification. The hazard identification methods and RA, in general, are used to derive the safety requirements for the system [20], however, they require a different level of information, which may include the concept, the architecture or the detailed system description. Similarly, the verification activities require a different set of information. This forms a basic distinction for the classification of the methods, which is depicted in Fig. 2 and explained in detail below.

The SysML/UML based approaches are used to formally state the requirements of the systems and to develop the background for its verification [9]. FTA, FMEA, ETA, RBD, BBN, MM, Petri Nets, Fault/Attack Trees and Monte Carlo simulation can be applied using a different amount of information about the system either on functional level or on a component level [9, 53]. LOPA is based on the information derived through PHA/HAZOP [53] but requires much less information on hazards than the FTA [53], whilst HAZOP is usually applied to a detailed system description and is more exhaustive and expensive method [53]. PHA requires limited information about the system, thus can be applied at a much earlier stage [53, 56]. STPA rely on the system functional representation, with STPA also being applicable during the conceptual design and the requirements definition phase [62]. FLSA, as a model-based approach, usually requires a more detailed description of the system including its architecture and information on its components, with their failure modes [69], thus it can be applied at a later stage of the system design. HRA methods require a different spectrum of information and can be applied from the beginning of the system development by constantly incorporating the new information generated during the design, leading in this way to a safer system design [51].

Fault injection can be applied to different models of the system [40, 69] or to an actual system [106]. Model checking is also applicable to the detailed system design or developed software structure [31, 94, 121]. Automated theorem proving works with an abstraction of a model, but this abstraction can be developed only based on a detailed model [107]. Testing is implemented after the detailed design of the system or its components [111]. On the other hand, runtime verification requires an already built system and is used during the systems operation phase [118].

4.2. *Dealing with sources of complexity*

An overview of the investigated methods applicability is provided in the form of a heatmap in Table 5, whilst a more detailed discussion is given in the next sections.

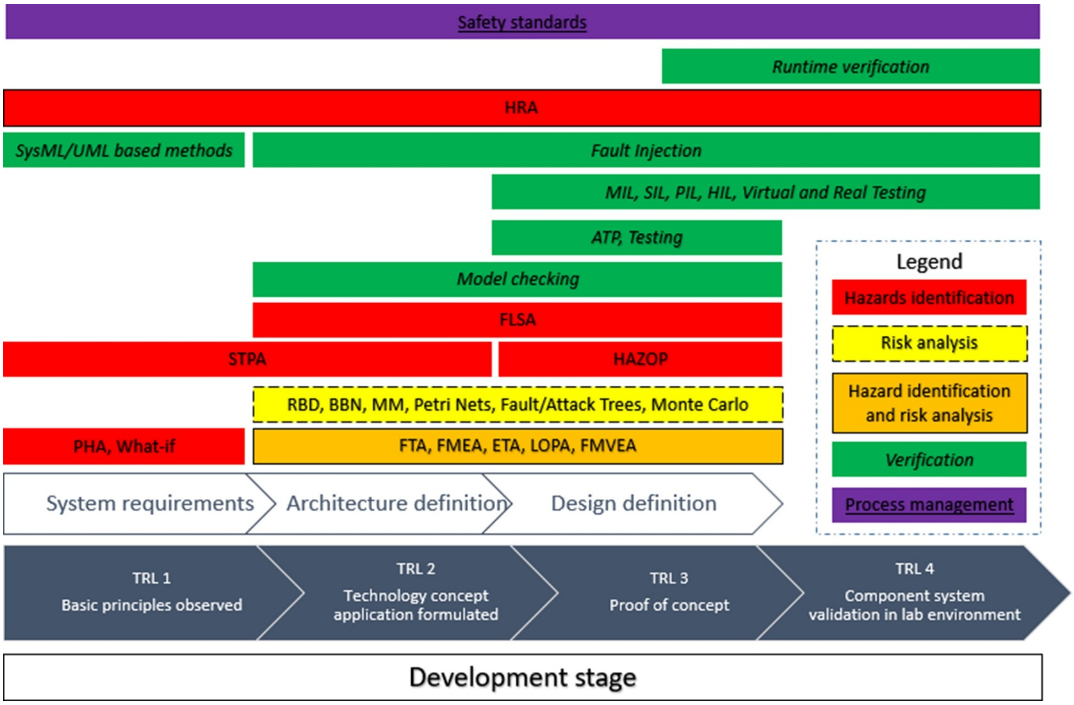

Fig. 2. Methods and their applicability to system engineering processes.

Table 5
Heatmap on methods for CPSs.

	CPSs type	THIM	STPA	FLSA	HRA	FI	MC	ATP	T	RT
Heterogeneity	IACSs	++	++	+++	NA	+++	++	+	++	+++
	CPSoSs	++	+	+	NA	++	+	+	+	+
Interoperability	IACSs	++	++	+++	NA	++	++	+	++	+++
	CPSoSs	+	+	+	NA	++	+	+	+	+
Connectivity	ACPSs	+	+	+	NA	+	+++	+++	+++	+++
	IACSs	+	++	+	NA	++	+++	+++	+++	+++
	CPSoSs	+	+	+	NA	++	+++	+++	+++	+++
Software-intensive	ACPSs	++	++	++	NA	++	+++	+++	+++	+++
	IACSs	++	+++	++	NA	++	+++	+++	+++	+++
	CPSoSs	+	+	++	NA	++	+++	+++	+++	+++
Humans in the loop	ACPSs	+	+	+	+	+	+	++	++	+
	IACSs	+	+	+	++	+	++	++	++	++
	CPSoSs	+	+	+	+	+	+	++	++	+
Evolution in time	ACPSs	-	-	+	NA	++	++	++	++	+++
	IACSs	-	-	++	NA	++	++	++	++	+++
	CPSoSs	-	-	++	NA	++	++	++	++	+++
Dynamic Reconfiguration	ACPSs	++	++	++	NA	+++	++	+	++	+++
	IACSs	++	++	+++	NA	+++	++	+	++	+++
	CPSoSs	++	++	+	NA	+++	++	+	++	+
Autonomous decision-making	ACPSs	+	+	-	NA	++	++	++	+++	+++
Advantageous		+++								
Applicable		++								
Applicable with changes		+								
Not advantageous		-								
Not applicable		NA								

THIM: Traditional Hazard Identification Methods
STPA: System-Theoretic Process Analysis
FLSA: Failure Logic Synthesis & Analysis
HRA: Human Reliability Analysis
RA: Risk Assessment

FI: Fault Injection
MC: Model Checking
ATP: Automated Theorem Proving
T: Testing
RT: Runtime Verification

4.2.1. Dealing with sources of structural complexity

Heterogeneity refers to the problem of unpredictable interactions due to the presence of different types of components. Heterogeneity of CPSs can be encountered by using the appropriate semantics in their design [24, 26]. As such, the traditional hazard identification methods, the FLSA methods, the model checking and the theorem proving techniques can, up to a certain extent, deal with the problem of heterogeneity, as their semantics are strong enough to represent the heterogeneous components of a CPS. However, the richness of the expressiveness of these approaches is limited compared with detailed models for capturing the components interactions and thus specific physical phenomenas can be ignored [24]. Consequently, fault injection to detailed models and systems, as well as testing applied to the system models or real systems is more suitable for addressing the system heterogeneity and the unpredictability of the components interactions [22]. Still, fault injection depends on the employed hazard identification methods [67], whereas testing provides a limited scenarios coverage. In addition, it may be challenging to implement actual testing in medical CPSs [95]. Systemic approaches like STPA, instead, can deal with the inherent problems at a high level and they can capture better the interactions between the system components. In specific, STPA focuses on the interactions between the cyber part and the physical part of a CPS [2, 13]. The automated theorem proving methods can generalise the dynamic hybrid systems used in model checking and provide the dynamic features that the hybrid systems cannot capture [7], therefore it is suitable for capturing the complex interactions between the control systems and the environment as well as for their verification.

Interoperability instead focuses on the safety implications related to the cooperation of the systems and their subsystems. As such, it requires the proper definition and examination of the functional dependencies between the system components and subsystems. In the traditional hazard identification techniques, such a dependency is identified manually, which impedes the whole process and renders it error-prone [100]. STPA can deal with the conflicting control actions that can arise in an integrated system [29]. The FLSA and fault injection techniques can provide a useful insight into the functional dependencies between the components faults, especially the fault injection to a detailed model [97], however, it may be not possible to identify unintended interactions between seemingly unrelated components [66]. Model checking and theorem proving can identify design inconsistencies related to interoperability in the system model [36]. However, model checking of SoS can be challenging, due to a big number of states that can be required for fully implementing the analysis [66, 96]. The applicability of theorem proving to the problem of interoperability is limited due to its semi-automatic nature. Integration and system testing can be used to identify inconsistencies in the system or the SoS [2, 10, 11].

Connectivity refers to the hazards induced by cybersecurity threats. These are not under the scope of the hazard identification techniques. However, the methods from both the safety and security disciplines can be suitably adjusted to identify and to incorporate cybersecurity induced hazards in the whole process of safety assurance [9]. Traditional hazard identification techniques like FMEA and FTA [33, 58] and the systemic methods [64] have been adjusted for the analysis of the system safety and security. By updating the semantics of the FLSA methods it is also possible to implement the safety and security analysis as reported in [32]. The model checking and the theorem proving techniques have been already applied for the verification of the cybersecurity properties [9, 22]. Testing can be also used to verify the performance of a system under the presence of attacks as well as to assess the performance of communication protocols [22].

Software becomes hazardous due to improper handling of the requirements sets [2, 29, 34]. Another problem related to software is its structural complexity and unpredictability of its behaviour under different inputs [22]. Some of the traditional methods like a Control HAZOP (a slightly modified form of HAZOP), FTA, FMEA or SHARD can be used for software hazard identification and analysis [34, 40, 53]. However, these methods would require additional methods like the fault injection to complement the process of the hazardous scenarios identification, and obtain the understanding of which deviations and failures are important [34]. STPA can support the identification of hazards from software by addressing the whole operational context [2, 34]. FLSA methods can provide information on how the control actions of software are related to the physical failures or faults, and how they will propagate in the system. Yet, the FLSA methods focus mostly on failures on a high level and not in the software structure [71, 74, 125]. Furthermore, they omit the context in which a software can be particularly hazardous. Fault injection can be applied to ensure the fault tolerance of the system to the software failures as described in [40] inheriting all the deficiencies and advantages of hazard identification methods. Formal methods have been applied for the software verification in embedded systems [36] as they can assist in the process of identifying bugs and inconsistencies [36]. The implementation of formal methods, though, requires the appropriate safety-oriented requirements, as a specific condition will not be considered as safe or unsafe unless defined by the user [121]. Testing is the state-of-the-art approach for verifying the properties of the cyber part of the system [111]. However, testing is vulnerable to the set of requirements and it may be impossible to test rigorously all the software functionalities [111].

The presence of humans results in a number of new interactions for the CPSs. The traditional hazard identification methods consider the human factors only implicitly [34]. STPA can be used for hazard analysis of human-machine interactions when suitably adjusted [2]. This method also assesses the human performance in the organisational context [2]. HRA, as it is expected, is suitable for addressing the hazardous interactions related to a human-machine interface [51]. The second generation HRA methods are more suitable than the first generation, as they consider the impact of cognitive tasks which are very important in human-machine interaction [78, 85]. Comparing with the first and second generation methods, simulation-based approaches can be used to develop an extended knowledge database and to identify more effectively the dynamic hazardous situations [78]. Furthermore, they can be used for operators training and system testing [78]. Testing is a part of a typical design process for a human-machine interface [51, 134]. Formal methods have been also applied for studying the human machinery interface [134]. Real-time verification has the potential to identify and confront those conditions related to improper human actions too.

4.2.2. Dealing with sources of dynamic complexity

Evolution in time refers to the modification of the system during its operation. This modification requires a proper management of changes [19, 39]. As such the traditional hazard identification techniques and STPA will be quite tedious in checking all the potential change influences. Instead, the model-based approaches are much more effective [7]. Runtime verification will be advantageous as it can prevent hazards from improper software updates. The runtime verification is applicable in tackling the problems related to heterogeneity, interoperability, software intensiveness and connectivity as it implements a continuous checking of the system properties to ensure their correctness, provided that there is adequate redundancy to switch to a healthy subsystem [22, 119]. The evolution in time requires the automatisation of the safety assurance process pointing towards the integrated approaches [19].

Safe dynamic reconfiguration depends on the proper system hazards identification, the effective functioning of the diagnosis and prognosis techniques as well as system control functions. The traditional safety assessment methods including FMEA can be useful for the identification of the hazards in the physical part and can be also used for the development of the diagnosis and prognostics algorithms [41]. The comparison of the STPA with FMEA showed that STPA provides a much broader understanding of the system hazards, although it is proposed to be used as a supplementary to the FMEA analysis [62, 63]. The FLSA methods can be better integrated into the design process, reducing the effort and errors during hazard identification. This makes them a useful tool for systems analysis and the development of diagnosis and prognosis techniques. However, FLSA may be weak in capturing the sequences of events in the system [66]. Fault injection techniques by using model checkers have been applied for the verification of the dynamic reconfiguration control functions and the diagnosability of a spacecraft faults [98]. In this way, it is possible to ensure that diagnostics are as effective as, the model and hazard identification techniques employed for the model checking purposes. Implementation of the theorem proving seems to be limited to these applications. Fault injection assisted by simulation models for the faults have been also proposed for the verification of prognostic techniques [41], accompanied by testing of the used algorithms [41]. However, the practical verification and validation of prognostics require a big amount of data, which are difficult to be obtained [41]. Real-time verification of prognostics seems to be tackling the problem of the data availability and is also advantageous in terms of financial cost as it reduces the need for testing of prognostic algorithms [135].

The hazard identification becomes more complicated when it comes to the autonomous decision-making system due to the complexity of the environment and the number of the anticipated hazardous scenarios [60]. Another problem arises due to uncertainty in the performance of the employed artificial intelligence algorithms. In this respect, modified traditional hazard identification techniques like HAZOP or PHA have been applied [8, 61]. The complexity of the autonomous decision-making in the environment seems to be a problem that cannot be addressed in detail by STPA. The FLSA methods are not applicable for the hazard identification in decision-making, as they mostly deal with interactions at component level. Fault injection can provide useful insights into the control functions of robots under the presence of faults in sensors and actuators [40, 97]. Formal methods have been also applied to autonomous systems [8]. Virtual testing is an essential tool for the verification of autonomous decision-making systems [117]. Physical experiments and simulation-based testing are more effective than the formal methods in capturing the interactions between robots and their environment [136]. However, it is necessary to ensure the complete coverage of the scenarios during testing [117]. Formal verification of the used machine learning algorithms seems to be an area of intensive ongoing research [43, 44].

4.3. Challenges with the application of methods for CPSs safety assurance

The usage of the CPSs safety assurance methods described above is not straightforward as specific difficulties may arise during the different stages of their implementation. Some of the main challenges along with the advantages of the methods for CPSs safety assurance are listed in Table 6.

5. Directions for future research and designing safer CPSs

Based on the analysis in the previous section it is possible to identify the general practices and directions for further research, which would support the CPSs safe design and operation as discussed below.

- Improvement of the available hazard identification methods and usage of advanced hazard identification techniques to ensure the completeness of the identified accident scenarios [2, 47]. The processes for implementation of hazard analysis must be enhanced through additional automation or combination of methods to reduce the dependency on the expert's skills [55], or supported by simulation techniques and modelling [14]. Detailed model simulations could potentially substitute the traditional hazard identification techniques in CPSs and be used to derive the STPA, HAZOP results and Fault Trees. Systemic methods like STPA could be potentially automatised in a similar manner with FLSA. The traditional and systemic hazard identification techniques could be extended for hazard identification in CPSoSs. FLSA methods can be also extended to consider the cyberattacks induced failures and their application can be investigated on more complex systems like CPSoSs, with an adapted formalism and analysis algorithms. Applicability of machine learning algorithms for hazard identification in the above cases can be also explored.
- Due to the criticality of human-machine interactions in the next generation CPSs, it would be beneficial to thoroughly investigate the human-machine interactions through advanced methods. However, it would be beneficial also to consider carefully the impact of latency and overreliance on the technology and the information and the format in which it is presented to the user. The impact of the human reliability must be also investigated in view of multiple operators in CPSoSs and cyberattacks.
- Studies to enhance CPSs RA should be further pursued, as it constitutes a useful method for decision making during systems safety assurance. This includes the procedures for accurately and efficiently predicting the failure rates to assist in arguing about an achieved safety level. The implementation of RA in CPSs can be also improved by the use of big data analytics for hardware, software, physical and human failure rates as in [129] for human operator errors.
- Facilitating the implementation of the model checking and the automated theorem proving techniques by overcoming the problems related to formalism, state-explosion and further automatisation of the theorem-proving is deemed requisite in future studies. It would be also interesting to investigate the applicability of theorem proving methods for fault injection studies and the verification of the dynamic reconfiguration functions in CPSs.
- Testing is conducted in the final stages of the design phase but has limited scenarios coverage. The combination of formal methods and model-based approaches is expected to automate the implementation of testing and render it more focused and efficient [112].

Table 6
Advantages and disadvantages of the methods.

	Pros	Cons
Traditional hazard identification methods	Well established	Manual process [100]
	Simple and adjustable semantics	Depends on the expertise of safety engineer [48, 55] Enhancement is required to consider the cyberattacks [9] and interactions with robots [60, 61] Do not consider errors in requirements [34, 40] Implicit incorporation of human factors [34]

(continued on next page)

Table 6 (*continued*)

	Pros	Cons
STPA	A wider perspective on the hazards [62, 63] Capturing the control structure and covering conflicting actions in CPSs [2, 13, 29] Does not need an update to consider cybersecurity induced hazards [64]	Manual process [48] Depends on the expertise of safety engineer [48] Specific guidelines are required for identification of context variables Quantitative analysis is not recommended [2]
FLSA	Better consistency with models [49, 67, 68] Repeatability and automatisation of analysis [67, 68] System behaviour can be modelled based on local failure behaviour [68, 69, 71, 72]	Applied to relevantly simple systems Does not consider dynamic reconfiguration [66] Still dependent on the localised version of FMEA Semantics should be updated to consider cyberattacks [32]
HRA	Well established Second and third generation methods consider the cognitive processes and dynamic phenomena [78, 85]	The scarcity of data [37, 78, 129] Missing connection with the ergonomic design of the human-machinery interface Need to incorporate the impact of latency and situational awareness properly [37]
Risk assessment	Cost-effective design [137] Facilitates decision makings [137] Necessary to demonstrate ALARP principle in safety case	Subject to biases [137] Unavailability of accurate data [137] Dependent on assumptions [2]
Fault injection	Impact of different faults can be assessed on more accurate models or actual system [40] Dynamic reconfiguration also considered Functional dependencies are captured Suitable for verification of prognostics [41]	Applied at a later stage of design [66] Generated Fault Tree consists of disjoint minimal cut sets impeding understanding of their physical meaning [66] Depends on hazard identification methods
Model checking	Effective in the identification of design, software and integration errors [36], cybersecurity vulnerabilities [9], human-machinery interface [134] Add rigour to the analysis [31, 94] Model-based approach [36]	Dependent on requirements [121] Challenges with a selection of appropriate abstraction [36] Inaccurate approximation of continuous aspects [108] State explosion problem [66, 96] Challenges with machine learning algorithms [43, 44]
Automated Theorem Proving	Stronger expression capabilities [7, 134] Identification of cybersecurity [9] and human-machine interface inconsistencies [51, 134] No state explosion problem [134] Model-based approach [36] Less consuming [15]	Dependent on requirements [121] Requires trained experts [134] Not fully automated [134] Difficult to handle [15] Challenges with machine learning algorithms [43, 44]
Testing	Applied on the actual or virtual system and subsequently better coverage of interactions [22, 117, 122] A primary verification method for software and software updates [111], prognostics [41]	Applied at a later stage of the design phase Dependent on the set of requirements, test coverage criteria and test cases [112, 116] Challenges with application to SoS
Runtime verification	Complexity sources tackled at the runtime [22, 119, 135]	Can be resource intensive [120]

Application of the testing process with the assistance of learning and prioritisation algorithms to ACPSs and CPSoSs can lead to a significant reduction of the required test cases.

- The improvements of the runtime verification of CPSs will allow the standardisation of this approach and will enable preventing the deficiencies not found during safety assurance process to be realised during operation in accidents.
- Further research towards the integration of the hazard analysis, design processes and verification methods through a Model-Based System Engineering and Model-Based Safety Assessment approaches must be pursued. The integration of tools will allow the easier and faster development of the safety case. This will also support the more effective and faster change impact analysis. Integration between the enhanced hazard identification methods for autonomous CPSs and the testing methods will support the better allocation of resources during the testing phase of the autonomous CPSs and will guide better design of simulation environment for testing, test oracles and prioritisation of the test cases.
- Since the autonomous decision-making systems are dependent on machine learning algorithms, it is necessary to ensure that the performance of algorithms is satisfactory using appropriate verification techniques such as the testing and model checking [44].
- Standardisation of the verification and validation processes of diagnostics and prognostics algorithms must be pursued since the prognostics and diagnostics constitute the basis for implementing the reconfiguration functions in CPSs. This includes ways to overcome the issues with the experimental data low availability [41]. This can be leveraged by combining simulation-based fault injections techniques for hazard identification with acquired actual data during the system testing and deployment phases.

6. Conclusions

In this study, the sources of complexity in CPSs and the available standards and methods for safety assurance have been reviewed. The primary sources of complexity have been identified as structural, which is related to the interactions between components in CPSs and dynamic or related to changes with time in the system, both of them rendering the CPSs prone to accidents. As the knowledge accumulates for the system interactions, the complexity decreases. During the CPSs design phase, this is achieved by applying appropriate safety assurance methods. The methods used to reduce the uncertainty in CPSs during design can be classified into those necessary for hazard identification and risk analysis in the system and those required for the system properties verification during different stages of design.

The findings of this review have shown that the traditional hazard identification methods have an applicability to CPSs, although they may be not adequate to capture properly all the interactions. It is required to update them to account for the cybersecurity induced hazards and new hazards emerging from the interactions between robots and systems. Systemic methods can be used to get a wider coverage of scenarios, but their manual character impedes their application. FLSAs are methods with a high level of automatisation, but they still require an adjustment to be applied to more complex systems as they do not capture the sequences of events in CPSs. HRAs have been applied for ensuring safer interactions between humans and machines, but they do not seem adequate to properly capture the issues related to latency and overreliance on technology. In addition, they must be enhanced to consider more accurately the information provided to the operator on the human-machinery interface.

While fault injection is strong in capturing the interactions in the CPSs, it is also dependent on hazard identification techniques and the model used for the analysis. Model checking can add rigour to the verification process, but it is limited by its expressiveness. In addition, its applicability to CPSoS may be challenging. Automated theorem proving despite the richness and its applicability to CPSs, is of the semi-automatic nature, thus with limited use. Runtime verification system seems to be a promising solution to the verification process but it faces a number of technical difficulties due to its high computational demand.

Future research must focus on the enhancement of the methods used for hazard identification. In addition, research for automatisation and integration of CPSs safety assurance methods should be pursued. Last, but the not least, appropriate integration of data generated during the CPSs design and operation for safety assurance purposes should be investigated.

It would be impossible to provide an overwhelming picture of the state-of-the-art and novel methods in a review paper, as there are at least 800 techniques, methods, databases or models for system safety assessment [138]. Still, though this study can be an effective starting point and give useful insights and directions for researchers in the area of CPSs safety.

Acknowledgement

The authors are thankful to Dr Romanas Puisa from Maritime Safety Research Centre and to Dr George Psarros, Dr Ole Christian Astrup, Dr Knut Erik Knutsen and Dr Pierre C Sames from DNV GL for their valuable comments and support. The authors are also thankful to anonymous reviewers and editor comments whose contribution significantly improved the quality of present work.

Appendix A. List of abbreviations

AADL	Architecture Analysis and Design Language
ACPSs	Autonomous Cyber-Physicals Systems
ATHEANA	A Technique for Human Error Analysis
BBN	Bayesian Belief Networks
CFT	Component Fault Trees
CPSs	Cyber-Physical Systems
CPSoSs	Cyber-Physical Systems of Systems
ETA	Event Tree Analysis
FLSA	Failure Logic Synthesis and Analysis
FMEA	Failure Modes and Effects Analysis
FMVEA	Failure Modes, Vulnerabilities and Effects Analysis
FTA	Fault Tree Analysis
HAZOP	HAZard and OPerability studies
HIL	Hardware In the Loop
HiP-HOPS	Hierarchically Performed Hazard Origin & Propagation Studies
HIL	Hardware-in-the-Loop
HRA	Human Reliability Analysis
IACSs	Industrial Automation and Control Systems
ISO	International Organization for Standardisation
LOPA	Layer Protection Analysis
MBT	Model-Based Testing
MIL	Model-in-the-Loop
MM	Markov Models
PHA	Preliminary Hazard Analysis
PIL	Processor-in-the-Loop
RA	Risk Analysis or Risk Assessment
RBD	Reliability Block Diagrams
SEFT	State-Event Fault Trees
SHARD	Software Hazard Analysis and Resolution in Design
SIL testing	Software-in-the-Loop testing
SoS	System of Systems
SysML	System Modelling Language
STPA	System-Theoretic Process Analysis
TRL	Technology Readiness Levels
UML	Unified Modelling Language

Appendix B. Research articles used to identify the safety assurance methods

An overview of publications used for the review is provided in Tables B1 and B2 based on their classification per industry sector, method and CPSs type.

Table B1
Publications per industry sector.

Aerospace	Automotive	Aviation	Medical	Maritime	Nuclear	Railway	Robotics	CI	Other
[81, 98], [113],	[25, 33], [39, 44], [63, 90], [93, 114], [115, 116], [119, 121], [131, 125] [132]	[74], [78], [81], [87], [100]	[31], [94, 95], [106]	[62], [117], [122]	[14], [37], [76], [80], [128], [129], [85], [88], [127]	[56], [91], [92]	[40], [60], [61], [96], [123], [126],	[59], [64], [105], [130],	[32], [35], [39], [57], [58], [79], [82], [77], [89], [102], [104], [108], [109], [110], [115], [118], [120], [124]

Table B2
Publications per method and CPSs type.

		ACPSs	CPSoSs	IACSs
Traditional hazard identification methods	Combination	[40]	[56]	[57]
	Cybersecurity incorporated	[65]	[59]	[33, 58]
	Interactions with robots	[60, 61]		
STPA			[64]	[62, 63]
FLSA				[32, 74, 76]
HRA	First generation			[17, 37, 77-79]
	Second generation			[77, 78]
	Third generation			[78, 80-82, 85]
RA		[87]	[92]	[88–91]
Fault injection	Abstraction	[98]		[100, 102, 104]
	Simulation			[14, 105]
	Real system	[40, 106]		
Model checking		[94, 96]	[93, 96]	[31, 35, 95]
Automated theorem prover		[25]		[108-110]
Testing	Hardware testing	[113]		[114-116]
	Decision-making algorithms testing	[44, 117]		
Runtime verification			[120]	[118, 119]
Integrated	Hazard identification and verification	[123]	[124]	[121, 122, 132]
	Hazard identification and HRA	[126]		[127–129]
	Verification activities	[131]		[39, 130]

References

[1] Luft J, Ingham H. The johari window. Hum Relations Training News 1961;5:6–7.
[2] Leveson NG. Engineering a safer world: systems thinking applied to safety. London, England: The MIT press; 2011.
[3] Sinha K. Structural complexity and its implications for design of Cyber-Physical Systems. Cambridge, MA, USA: Massachusetts Institute of Technology; 2014.
[4] Gunes V, Peter S, Givargis T, Vahid F. A survey on concepts, applications, and challenges in Cyber-Physical Systems. KSII Trans Internet Inf Syst 2014;8:4242–68.
[5] Hehenberger P, Vogel-Heuser B, Bradley D, Eynard B, Tomiyama T, Achiche S. Design, modelling, simulation and integration of Cyber Physical Systems: methods and applications. Comput. Ind 2016;82:273–89.
[6] Möller DP. Guide to computing fundamentals in Cyber-Physical Systems. Switzerland: Springer International Publishing; 2016.
[7] Engell S, Paulen R, Reniers MA, Sonntag C, Thompson H. Core research and innovation areas in Cyber-Physical Systems of systems. Cyber physical systems design, modelling, and evaluation. Lecture notes in computer science, 9361. Netherlands, Amsterdam: Springer; 2015. p. 40–55.
[8] Guiochet J, Machin M, Waeselynck H. Safety-critical advanced robots: a survey. Rob Auton Syst 2017;94:43–52.
[9] Kriaa S, Pietre-Cambacedes L, Bouissou M, Halgand Y. A survey of approaches combining safety and security for industrial control systems. Reliab Eng Syst Saf 2015;139:156–78.
[10] US Department of Defense. Department of defense standard practice: System safety. U.S. Department of Defense; 2012. MIL-STD-882E.
[11] Standardization IOf. Functional safety of electrical/electronic/programmable electronic safety-related systems - IEC 61508. Part 1: general requirements. United Kingdom, London: British Standard Institution; 2010.
[12] Johansen IL, Rausand M. Defining complexity for risk assessment of sociotechnical systems: a conceptual framework. Proc Inst Mech Eng Part O 2014;228:272–90.
[13] Qureshi ZH. A review of accident modelling approaches for complex socio-technical systems. Workshop on safety critical systems and software and safety-related programmable systems. Australia, Adelaide: Australian Computer Society, Inc.; 2007. p. 47–59.
[14] Sierla S, O'Halloran BM, Karhela T, Papakonstantinou N, Tumer IY. Common cause failure analysis of cyber–physical systems situated in constructed environments. Res Eng Des Theory Appl Concurrent Eng 2013;24:375–94.
[15] Bagade P, Banerjee A, Gupta SKS. Validation, verification, and formal methods for Cyber-Physical Systems. In: Rawat DB, Jeschke S, Brecher C, editors. Cyber-Physical Systems. Boston: Academic Press; 2017. p. 175–91.
[16] Perrow C. Normal accidents: living with high risk technologies. Princeton, New Jersey: Princeton University Press; 1999.
[17] Zio E. Challenges in the vulnerability and risk analysis of critical infrastructures. Reliab Eng Syst Saf 2016;152:137–50.
[18] Aizpurua JI, Muxika E. Design of dependable systems: an overview of analysis and

verification approaches. In: Naqvi S, Dini P, editors. The 5th international conference on dependability. IARIA; 2012. p. 4–12.

[19] Vara JLdl, Borg M, Wnuk K, Moonen L. An industrial survey of safety evidence change impact analysis practice. IEEE Trans Softw Eng 2016;42:1095–117.

[20] Martins LEG, Gorschek T. Requirements engineering for safety-critical systems: a systematic literature review. Inf Softw Technol 2016;75:71–89.

[21] Nair S, de la Vara JL, Sabetzadeh M, Briand L. An extended systematic literature review on provision of evidence for safety certification. Inf Softw Technol 2014;56:689–717.

[22] Wolf M, Serpanos D. Safety and security in Cyber-Physical Systems and internet-of-things systems. Proc IEEE 2018;106:9–20.

[23] Leveson NG. Complexity and Safety. In: Hammami O, Krob D, Voirin J-L, editors. Complex systems design & management 2011. Germany, Berlin: Springer; 2012. p. 27–39.

[24] Rajhans A, Bhave A, Ruchkin I, Krogh BH, Garlan D, Platzer A, et al. Supporting heterogeneity in Cyber-Physical Systems architectures. IEEE Trans Autom Control 2014;59:3178–93.

[25] Petnga L, Austin M. Ontologies of time and time-based reasoning for MBSE of Cyber-Physical Systems. Procedia Comput Sci 2013;16:403–12.

[26] Zheng X, Julien C, Kim M, Khurshid S. Perceptions on the state of the art in verification and validation in Cyber-Physical Systems. IEEE Syst J 2015;11:2614–27.

[27] Sampigethaya K, Poovendran R. Aviation Cyber–physical systems: foundations for future aircraft and air transport. Proc IEEE 2013;101:1834–55.

[28] Liu Y, Peng Y, Wang B, Yao S, Liu Z. Review on Cyber-Physical Systems. IEEE/CAA J Autom Sinica 2017;4:27–40.

[29] Placke S, Thomas J, Suo D. Integration of multiple active safety systems using STPA SAE; 2015. Technical Paper.

[30] Kim KD, Kumar PR. Cyber–physical systems: a perspective at the centennial. Proc IEEE 2012;100:1287–308.

[31] Lee I, Sokolsky O, Chen S, Hatcliff J, Jee E, Kim B, et al. Challenges and research directions in medical Cyber-Physical Systems. Proc IEEE 2012;100:75–90.

[32] Delange J, Pautet L, Feiler PH. Validating safety and security requirements for partitioned architectures. In: Kordon F, Kermarrec Y, editors. Reliable software technologies. Ada-Europe: lecture notes in computer science, 5570. Springer; 2009. p. 30–43.

[33] Schmittner C, Ma Z, Schoitsch E, Gruber T. A case study of FMVEA and CHASSIS as safety and security co-analysis method for automotive Cyber-Physical Systems. *1st ACM* workshop *on Cyber-Physical Sys*tem security. Singapore: Republic of Singapore: ACM; 2015. p. 69–80.

[34] Thomas J. Extending and automating a systems-theoretic hazard analysis for requirements generation and analysis. Massachusetts Institute of Technology; 2013.

[35] Bohlender D, Kowalewski S. Design and verification of restart-robust industrial control software. In: Furia CA, Winter K, editors. *Integrated* formal methods *2018*. Lecture Notes in Computer Science, 11023. Maynooth, Ireland: Springer; 2018. p. 47–68.

[36] Baier C, Katoen J-P. Principles of model checking. Cambridge, Massachusets: MIT press; 2008.

[37] Li P, Zhang L, Dai L, Zou Y, Li X. An assessment method of operator's situation awareness reliability based on fuzzy logic-AHP. Saf Sci 2018.

[38] Kevin Anthony H, Masooda B. Trust in automation: integrating empirical evidence on factors that influence trust. Hum Factors 2014;57:407–34.

[39] Nguyen LV, Hoque KA, Bak S, Drager S, Johnson TT. Cyber-Physical specification mismatches. ACM Trans Cyber-Phys Syst 2018;2(23):1–6.

[40] Gamble C, Pierce K, Fitzgerald J, Bos B. Co-modelling of faults and fault tolerance mechanisms. In: Fitzgerald J, Larsen PG, Verhoef M, editors. Collaborative design for embedded Systems: co-modelling and co-simulation. Berlin, Heidelberg: Springer; 2014. p. 185–97.

[41] Goebel K, Daigle M, Saxena A, Sankararaman S, Roychoudhury I, Celaya JR. Prognostics the science of prediction. USA: Create Space Independent Publishing Platform; 2017.

[42] Murashov V, Hearl F, Howard J. Working safely with robot workers: Recommendations for the new workplace. J Occup Environ. Hyg 2016;13:D61–71.

[43] Huang X, Kwiatkowska M, Wang S, Wu M. Safety verification of deep neural networks. In: Majumdar R, Kuncak V, editors. Computer aided verification 2017, 10426. Heidelberg, Germany: Springer; 2017. p. 3–29. Lecture Notes in Computer Science.

[44] Tian Y, Pei K, Jana S, Ray B. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. Proceedings of the 40th international conference on software engineering. ACM; 2018. p. 303–14.

[45] Phillips B, Blackburn M. Verification points for self-adaptive systems. Procedia Comput Sci 2014;36:118–23.

[46] European Commission. Commission implementing regulation (EU) No 1035/2011. Off J Eur Union 2011:19.

[47] Dezfuli H, Allan B, Smith C, Stamatelatos M, Youngblood R. NASA system safety handbook. Volume 1, system safety framework and concepts for implementation. USA, Washington D.C: National Aeronautics and Space Administration; 2011.

[48] International Organization for Standardization. ISO 26262: road vehicles — functional safety. Part 1: vocabulary. United Kingdom, London: British Standard Industries; 2011.

[49] Joshi A, Whalen M, Heimdal MPE. Model-based safety analysis final report. Nasa Langley Research Center; 2006.

[50] International Organization for Standardization. Medical devices-application of risk management to medical devices. ISO 14971. United Kingdom, London: British Standard Instituition; 2012. 2007.

[51] International Organization for Standardization. Guidance on human aspects of dependability - EN 62508. United Kingdom, London: British Standard Institution; 2010.

[52] EARTO. The TRL scale as a research & innovation policy tool. EARTO Recommendations; 2014.

[53] International Organization for Standardization. Risk management - guidelines - ISO 31000. United Kingdom, London: British Standards Institution; 2018.

[54] Petroleum Safety Authority. Principles for barrier management in the petroleum industry. Norway: Petroleum Safety Authority; 2013.

[55] Cameron I, Mannan S, Németh E, Park S, Pasman H, Rogers W, et al. Process hazard analysis, hazard identification and scenario definition: Are the conventional tools sufficient, or should and can we do much better? Process Saf Environ Prot 2017;110:53–70.

[56] Soubiran E, Guenab F, Cancila D, Koudri A, Wouters L. Ensuring dependability and performance for CPS design: application to a signaling system. In: Rawat DB, Jeschke S, Brecher C, editors. Cyber-Physical Systems. Boston: Academic Press; 2017. p. 363–75.

[57] Peeters JFW, Basten RJI, Tinga T. Improving failure analysis efficiency by combining FTA and FMEA in a recursive manner. Reliab Eng Syst Saf 2018;172:36–44.

[58] Sabaliauskaite G, Mathur AP. Aligning Cyber-Physical System safety and security. In: Cardin M-A, Krob D, Lui PC, Tan YH, Wood K, editors. Complex systems design & management asia. Singapore: Springer; 2015. p. 41–53.

[59] Sabaliauskaite G, Adepu S. Integrating Six-step model with information flow diagrams for comprehensive analysis of Cyber-Physical system safety and security. IEEE 18th International Symposium on High Assurance Systems Engineering (HASE). 2017. p. 41–8.

[60] Dogramadzi S, Giannaccini ME, Harper C, Sobhani M, Woodman R, Choung J. Environmental hazard analysis-a variant of preliminary hazard analysis for autonomous mobile robots. J Intell Rob Syst 2014;76:73–117.

[61] Guiochet J. Hazard analysis of human–robot interactions with HAZOP–UML. Saf Sci 2016;84:225–37.

[62] Rokseth B, Utne IB, Vinnem JE. A systems approach to risk analysis of maritime operations. Proc Inst Mech Eng Part O 2017;231:53–68.

[63] Sulaman SM, Beer A, Felderer M, Höst M. Comparison of the FMEA and STPA safety analysis methods–a case study. Softw Qual J 2017:1–39.

[64] Friedberg I, McLaughlin K, Smith P, Laverty D, Sezer S. STPA-SafeSec: safety and security analysis for Cyber-Physical Systems. J Inf Secur Appl 2016;34:183–96.

[65] Sabaliauskaite G, Liew LS, Cui J. Integrating autonomous vehicle safety and security analysis using STPA method and the six-step model. Int J Adv Secur 2018;11:160–9.

[66] Sharvia S, Kabir S, Walker M, Papadopoulos Y. Model-based dependability analysis: state-of-the-art, challenges, and future outlook. In: Soley R, Ali N, Grundy J, Tekinerdogan B, editors. Software quality assurance. Boston: Morgan Kaufmann; 2016. p. 251–78.

[67] Adler R, Domis D, Höfig K, Kemmann S, Kuhn T, Schwinn J-P, et al. Integration of component fault trees into the UML. International conference *on* model driven engineering languages *and* systems. Oslo, Norway: Springer; 2010. p. 312–27.

[68] Li S, Li X. Study on generation of fault trees from Altarica models. Procedia Eng 2014;80:140–52.

[69] Papadopoulos Y, Walker M, Parker D, Sharvia S, Bottaci L, Kabir S, et al. A synthesis of logic and bio-inspired techniques in the design of dependable systems. Ann Rev Control 2016;41:170–82.

[70] Fenelon P, McDermid JA. An integrated tool set for software safety analysis. J Syst Softw 1993;21:279–90.

[71] Papadopoulos Y, McDermid J. Hierarchically performed hazard origin and propagation studies. In: Felici M, Kanoun K, Pasquini A, editors. Computer safety, reliability and security. Lecture notes in computer science, 1698. Toulouse, France: Springer; 1999. p. 139–52.

[72] Feiler P, Rugina A. Dependability modeling with the architecture analysis & design language (AADL). Carnegie Melon University; 2007.

[73] Kaiser B, Liggesmeyer P, Mäckel O. A new component concept for fault trees. In: Lindsay P, Cant T, editors. 8th Australian workshop on safety critical systems and software. Australian Computer Society, Inc.; 2003. p. 37–46.

[74] Choley J-Y, Mhenni F, Nguyen N, Baklouti A. Topology-based safety analysis for safety critical CPSs. Procedia Comput Sci 2016;95:32–9.

[75] Roth M, Wolf M, Lindemann U. Integrated matrix-based Fault Tree generation and evaluation. Procedia Comput Sci 2015;44:599–608.

[76] Dehlinger J, Dugan JB. Analyzing dynamic fault trees derived from model-based system architectures. Nuclear Eng Technol 2008;40:365–74.

[77] Olivares RDC, Rivera SS, Mc Leod JEN. A novel qualitative prospective methodology to assess human error during accident sequences. Saf Sci 2018;103:137–52.

[78] Boring RL. Fifty years of THERP and human reliability analysis. Idaho National Laboratory (INL); 2012.

[79] Petrillo A, De Felice F, Falcone D, Silvestri A, Zomparelli F. A hybrid probabilistic model for evaluating and simulating Human Error in Industrial Emergency conditions (HEIE). J Fail Anal Prev 2017;17:462–76.

[80] Chang YHJ, Mosleh A. Cognitive modeling and dynamic probabilistic simulation of operating crew response to complex system accidents: part 1: overview of the IDAC model. Reliab Eng Syst Saf 2007;92:997–1013.

[81] Gore BF. Man–machine integration design and analysis system (MIDAS) v5: augmentations, motivations, and directions for aeronautics applications. In: Cacciabue PC, Hjälmdahl M, Luedtke A, Riccioli C, editors. Human modelling in assisted transportation. Milano, Italy: Springer; 2011. p. 43–54.

[82] Petrillo A, Falcone D, De Felice F, Zomparelli F. Development of a risk analysis model to evaluate human error in industrial plants and in critical infrastructures. Int J Disaster Risk Reduct 2017;23:15–24.

[83] Di Pasquale V, Miranda S, Iannone R, Riemma S. A simulator for human error probability analysis (SHERPA). Reliab Eng Syst Saf 2015;139:17–32.

[84] Boring RL. How many performance shaping factors are necessary for human reliability analysis? Idaho National Laboratory (INL); 2010.

[85] Fan C-F, Chan C-C, Yu H-Y, Yih S. A simulation platform for human-machine interaction safety analysis of Cyber-Physical Systems. Int J Ind Ergonomics 2018;68:89–100.

[86] Stamatelatos M, Dezfuli H, Apostolakis G, Everline C, Guarro S, Mathias D, et al. Probabilistic risk assessment procedures guide for NASA managers and practitioners. 2011.

[87] Gonçalves P, Sobral J, Ferreira LA. Unmanned aerial vehicle safety assessment modelling through Petri Nets. Reliab Eng Syst Saf 2017;167:383–93.

[88] Aldemir T, Guarro S, Mandelli D, Kirschenbaum J, Mangan LA, Bucci P, et al. Probabilistic risk assessment modeling of digital instrumentation and control systems using two dynamic methodologies. Reliab Eng Syst Saf 2010;95:1011–39.

[89] Abdo H, Kaouk M, Flaus JM, Masse F. A safety/security risk analysis approach of industrial control systems: a cyber bowtie – combining new version of attack tree with bowtie analysis. Comput Secur 2018;72:175–95.

[90] Qiu S, Rachedi N, Sallak M, Vanderhaegen F. A quantitative model for the risk evaluation of driver-ADAS systems under uncertainty. Reliab Eng Syst Saf 2017;167:184–91.

[91] Wang W, Cammi A, Di Maio F, Lorenzi S, Zio E. A Monte Carlo-based exploration framework for identifying components vulnerable to cyber threats in nuclear power plants. Reliab Eng Syst Saf 2018;175:24–37.

[92] Wu D, Zheng W. Formal model-based quantitative safety analysis using timed Coloured Petri Nets. Reliab Eng Syst Saf 2018;176:62–79.

[93] Zhang Y, Shi J, Zhang T, Liu X, Qian Z. Modeling and checking for Cyber–Physical System based on hybrid interface automata. Pervasive Mobile Comput 2015;24:179–93.

[94] Bresolin D, Geretti L, Muradore R, Fiorini P, Villa T. Formal verification of robotic surgery tasks by reachability analysis. Microprocess Microsyst 2015;39:836–42.

[95] Paoletti N, Patan A, Kwiatkowska M. Closed-loop quantitative verification of rate-adaptive pacemakers. ACM Trans Cyber-Phys Syst 2018;2:1–31.

[96] Cavalcanti A, Miyazawa A, Sampaio A, Li W, Ribeiro P, Timmis J. Modelling and verification for swarm robotics. In: Furia CA, Winter K, editors. *Integrated* formal methods *2018*. Lecture Notes in Computer Science, 11023. Maynooth, Ireland: Springer; 2018. p. 1–19.

[97] Elks CR. Development of a fault injection-based dependability assessment methodology for digital and I & C systems: United States nuclear regulatory commission. Office of Nuclear Regulatory Research; 2012.

[98] Bozzano M, Cimatti A, Katoen J-P, Katsaros P, Mokos K, Nguyen VY, et al. Spacecraft early design validation using formal methods. Reliab Eng Syst Saf 2014;132:20–35.

[99] Bozzano M, Cimatti A, Katoen J-P, Nguyen VY, Noll T, Roveri M. The COMPASS approach: correctness, modelling and performability of aerospace systems. In: Buth B, Rabe G, Seyfarth T, editors. International conference on computer safety, reliability, and security. Lecture Notes in Computer Science, 5775. Springer; 2009. p. 173–86.

[100] Li Y, Gong Q, Su D. Model-based System safety assessment of aircraft power plant. Procedia Eng 2014;80:85–92.

[101] Arnold A, Point G, Griffault A, Rauzy A. The AltaRica formalism for describing concurrent systems. Fundamenta Informaticae 1999;40:109–24.

[102] Dal Lago L, Ferrante O, Passerone R, Ferrari A. Dependability assessment of SOA-based CPS with contracts and Model-Based fault injection. IEEE Trans Ind Inf 2018;14:360–9.

[103] Bittner B, Bozzano M, Cavada R, Cimatti A, Gario M, Griggio A, et al. The xSAP safety analysis platform. In: Chechik M, Raskin J-F, editors. International conference on tools and algorithms for the construction and analysis of systems. Springer; 2016. p. 533–9.

[104] Leupolz J, Knapp A, Habermaier A, Reif W. Qualitative and quantitative analysis of safety-critical systems with. Int J Softw Tools Technol Transfer 2017.

[105] Clarke EM, Zuliani P. Statistical model checking for Cyber-Physical Systems. In: Bultan T, Hsiung P-A, editors. Automated technology for verification and analysis: 9th international symposium. Lecture notes in computer science, 6996. Springer; 2011. p. 1–12.

[106] Alemzadeh H, Chen D, Lewis A, Kalbarczyk Z, Raman J, Leveson N, et al. Systems-Theoretic safety assessment of robotic telesurgical systems. In: Koornneef F, van Gulijk C, editors. 34th international conference computer safety, reliability, and security. Lecture notes in computer science, 9337. Springer; 2015. p. 213–27.

[107] Platzer A. Logic & proofs for Cyber-Physical Systems. In: Olivetti N, Tiwari A, editors. International joint conference on automated reasoning. Lecture Notes in Computer Science, 9706. Springer; 2016. p. 15–21.

[108] Sanwal MU, Hasan O. Formal verification of Cyber-Physical Systems: coping with continuous elements. In: Minh Ho Chi, editor. Computational science and its applications. Minh Ho Chi, editor. Lecture notes iin computer science, 7971. Vietnam: Springer; 2013. p. 358–71.

[109] Hulette GC, Armstrong RC, Mayo JR, Ruthruff JR. Theorem-proving analysis of digital control logic interacting with continuous dynamics. Electron Notes Theor Comput Sci 2015;317:71–83.

[110] Ivanov I, Panchenko T, Nikitchenko M, Sunmade F. On formalization of semantics of real-time and Cyber-Physical Systems. In: Hu Z, Petoukhov S, Dychka I, He M, editors. Advances in computer science for engineering and education. Cham: Springer International Publishing; 2019. p. 213–23.

[111] International Organization for Standardization. Software and systems engineering — software testing ISO/IEC/IEEE 29119. United Kingdom, London: British Standards Institution; 2013.

[112] Aerts A, Reniers M, Mousavi MR. Model-based testing of Cyber-Physical Systems. In: Rawat DB, Jeschke S, Brecher C, editors. Cyber-Physical Systems. Boston: Academic Press; 2017. p. 287–304.

[113] Martins-Filho LS, Santana AC, Duarte RO, Junior GA. Processor-in-the-Loop simulations applied to the design and evaluation of a satellite attitude control. In: Awrejcewicz J, editor. Computational and numerical simulations. IntechOpen; 2014.

[114] Hansen N, Wiechowski N, Kugler A, Kowalewski S, Rambow T, Busch R. Model-in-the-Loop and software-in-the-loop testing of closed-loop automotive software with arttest. Germany, Bonn: Informatik; 2017. p. 1537–49. 2017.

[115] Claessen K, Smallbone N, Eddeland J, Ramezani Z, Åkesson K. Using valued booleans to find simpler counterexamples in random testing of Cyber-Physical Systems. IFAC-PapersOnLine 2018;51:408–15.

[116] Matinnejad R, Nejati S, Briand L, Bruckmann T. Test generation and test prioritization for simulink models with dynamic behavior. IEEE Trans Softw Eng 2018:1–25.

[117] Mullins GE, Stankiewicz PG, Hawthorne RC, Gupta SK. Adaptive generation of challenging scenarios for testing and evaluation of autonomous vehicles. J Syst Softw 2018;137:197–215.

[118] Mitsch S, Platzer A. ModelPlex: verified runtime validation of verified Cyber-Physical System models. Formal Methods Syst Des 2016;49:33–74.

[119] Kane A, Fuhrman T, Koopman P. Monitor based oracles for Cyber-Physical System testing: practical experience report. IEEE 44th dependable systems and networks. IEEE; 2014. p. 148–55.

[120] Habermaier A, Eberhardinger B, Seebach H, Leupolz J, Reif W. Runtime model-based safety analysis of self-organizing systems with S#. 2015 IEEE international conference on self-adaptive and self-organizing systems workshops. IEEE Computer Society; 2015. p. 128–33.

[121] Abdulkhaleq A, Wagner S. A software safety verification method based on system-theoretic process analysis. In: Bondavalli A, Ceccarelli A, Ortmeier F, editors. *Computer safety, reliability, and* security. Lecture notes in computer science, 8696. Florence, Italy: Springer; 2014. p. 401–12.

[122] Rokseth B, Utne IB, Vinnem JE. Deriving verification objectives and scenarios for maritime systems using the systems-theoretic process analysis. Reliab Eng Syst Saf 2018;169:18–31.

[123] Blackburn MR, Austin MA, Coelho M. Modeling and cross-domain dependability analysis of Cyber-Physical Systems. 2018 Annual IEEE international systems conference (SysCon). 2018. p. 1–8.

[124] Zhang M, Ali S, Yue T, Norgren R, Okariz O. Uncertainty-wise Cyber-Physical System test modeling. Softw Syst Model 2017:1–40.

[125] Sharvia S, Papadopoulos Y. Integrating model checking with HiP-HOPS in model-based safety analysis. Reliab Eng Syst Saf 2015;135:64–80.

[126] Kaber D, Zahabi M. Enhanced hazard analysis and risk assessment for human-in-the-loop systems. Hum Factors 2017;59:861–73.

[127] Kim HE, Son HS, Kim J, Kang HG. Systematic development of scenarios caused by cyber-attack-induced human errors in nuclear power plants. Reliab Eng Syst Saf 2017;167:290–301.

[128] Jiang J, Wang Y, Zhang L, Wu D, Li M, Xie T, et al. A cognitive reliability model research for complex digital human-computer interface of industrial system. Saf Science 2018;108:196–202.

[129] Ham D-H, Park J. Use of a big data analysis technique for extracting HRA data from event investigation reports based on the Safety-II concept. Reliab Eng Syst Saf 2018.

[130] Seceleanu C, Johansson M, Suryadevara J, Sapienza G, Seceleanu T, Ellevseth S-E, et al. Analyzing a wind turbine system: from simulation to formal verification. Sci Comput Program 2017;133:216–42.

[131] Ge N, Jenn E, Breton N, Fonteneau Y. Integrated formal verification of safety-critical software. Int J Softw Tools Technol Transfer 2018;20:423–40.

[132] Araujo H, Carvalho G, Mohaqeqi M, Mousavi MR, Sampaio A. Sound conformance testing for Cyber-Physical Systems: theory and implementation. Sci Comput Program 2018;162:35–54.

[133] INCOSE. Systems engineering handbook: a guide for system life cycle processes and activities. Fourth Edition ed. USA, New JerseyHoboken: John Wiley & Sons, Inc; 2015.

[134] Bolton ML, Bass EJ, Siminiceanu RI. Using formal verification to evaluate human-automation interaction: a review. IEEE Trans Syst Man Cybern Syst 2013;43:488–503.

[135] Gupta JS, Trinquier C, Medjaher K, Zerhouni N. Continuous validation of the PHM function in aircraft industry. IEEE 1st reliability systems engineering conference. IEEE; 2015. p. 1–7.

[136] Sotiropoulos T, Waeselynck H, Guiochet J, Ingrand F. Can robot navigation bugs be found in simulation? An exploratory study. IEEE international conference on software quality, reliability and security. IEEE; 2017. p. 150–9.

[137] Goerlandt F, Khakzad N, Reniers G. Validity and validation of safety-related quantitative risk analysis: a review. Saf Sci 2016;99:127–39.

[138] Everdij MHC, Blom HAP, et al. Safety methods database. In: Allocco M, Bush D, Çeliktin M, Kirwan B, Mana P, Mickel J, editors. Netherlands: Netherlands Aerospace Centre NLR; 2016.