Review

A Comprehensive Technological Survey on Dependable Self-Managing CPS: The Decade of Researches on Correctness and Dependability

Peng Zhou[1,2,*], Decheng Zuo [1], Kunmean HOU [2], Zhan Zhang [1], Jian Dong [1], Jianjin Li [2] and Haiyin Zhou [1]

[1] School of computer science and technology, Harbin Institute of Technology, Harbin, Heilongjiang, China; zhoupeng@ftcl.hit.edu.cn, zuodc@hit.edu.cn, zz@ftcl.hit.edu.cn, dan@hit.edu.cn, haiyingzhou@hit.edu.cn

[2] LIMOS, UMR 6158 CNRS, University Clermont Auvergne, Aubière CEDEX, France; kunmean.hou@gmail.com, jjli@isima.fr

* Correspondence: zhoupeng@ftcl.hit.edu.cn, zz@ftcl.hit.edu.cn, Tel.: +86-0451-86403316

Abstract: Though Cyber Physical Systems (CPS) become very popular in last the decade, dependability of CPS is still a critical issue and related survey is rare. We try to spell out the jigsaw of technologies and figure out the technical trends of dependable self-managing CPS. This survey first recalls the motivation and the similar concepts. By analyzing four generic architectures, we summarize the common characteristics and related assurance technologies, and propose a more generic environment-in-loop processing flow of CPS and a formal interaction flow between physical space and cyber space. Further, the similarity between correctness and dependability is formally analyzed and the new five research questions of dependable self-managing CPS are presented. Then we review the critical technologies and related correctness verification & validation (V&V) methods, the architectures for dependable self-managing CPS. Further, the detail dependability management and V&V technologies are surveyed, which covers the areas of running-time fault management methods and whole life cycle V&V technologies, maintenance and available tool sets. For holistic CPS development, Modeling techniques and MDE (model driven engineering) based V&V methods are analyzed in detail. Then we complete the jigsaw of technologies and figure out the missing part. Further, we propose the technical challenges and the further direction. To our best knowledge, this is the first comprehensive survey on dependable self-managing CPS development and evaluation.

Keywords: Cyber Physical Systems; Industry 4.0; MDE; Hardware and Software Co-design; Lifetime Verification & Validation; Dependability; Correctness; Flexibility; Self-management; Self-adapting; Self-healing;

1. Introduction

The primary motivation of developing an artificial assistant system is to increase the productivity and to easy life. Various technologies have been created and developed, such as computer, communication technology, control theory and mechanism, etc. With multidisciplinary technologies being integrated, the complexity of the system increases exponentially. Moreover, these fragmented technologies can hardly cooperate to form an organic system without a holistic theory.

In the context of Information Communications Technology (ICT), Internet of Things (IoT) simplifies real world data sensing and collecting. Cloud system mines the connections within the sensory data with big data analytics and the decision support system helps us explore the alternative solutions. These technologies improve our understanding of the real world and increase our productivity. However nowadays, all these activities occur in cyberspace independently, the system can't influence the real world directly. Human beings have to make the last decision and take the last action to transform the physical world.

For control theory, a typical control system monitors the signals from the real world, and then processes these signals with partial or ordinary differential equations (PDE or ODE). Feedback

control is the common method widely applied to adjust system's behavior in next period; and feed-forward control may be applied to correct deviation and prevent errors. Compared to IoT, control system reacts in real-time, but is in smaller scale. The weaker process unit strangles the analysis of large quantities of signals/data, and perverts to apply of big data analytics. As a result, control system seems "naiver" than IoT.

Due to their own shortages, none of IoT and control system can realize the primary vision. A new system combining the advantages of two kind systems should be invented. It can sense and analyze the data like IoT, and interact with the real physical world in real-time as a control system. This motivation calls **Cyber Physical System** (CPS). It is a close-loop system, which can learn from and interact with the physical world automatically, and finally adapt its behavior to new situations. Take self-driving vehicles as an example, a car "watches" the road with radars, cameras and laser sensors, etc. The embedded computer draws a picture of the current road based on these sensor data and recognizes other objects such as other cars, passengers and obstacles. Then actions will be taken according to the analysis results to avoid collision. Furthermore, a self-driving car can cooperate with other cars through vehicle network, and connect the transportation data center for better decision supporting to avoid accidents and road traffic congestions. Considering the bright prospect drawn by CPS, some researchers regard it as a next revolution of technology which can rival the contribution of the Internet [1].

CPS was first proposed by the American National Science Foundation (NFS) in 2006. It was regarded as a practice of networking and information technology (NIT). One year later, the President's Council of Advisors on Science and Technology (PCAST) raised the CPS to ensure the continued leadership of the USA and recommend putting CPS as a top research agenda. On the other hand, from the view of industrial manufacturing, Germany government put forward **Industry 4.0** to promote the traditional industry to in 2012. Intelligent manufacture can fit the different requirements of customized orders, i.e. various small orders for nonstandard production. EU (Horizon 2020, in 2013), Japan (CPS Task Force, in 2015), China (Made in China 2025, in 2015) published their own plan on CPS/Industry 4.0. Though from different motives, CPS and Industry 4.0 try to achieve the same goal [2]. From the view of smart manufacturing, Hyosung S. K et al. surveyed the approaches in Germany, U.S., and Korea, which includes *Industry 4.0*, *Smart Manufacturing*, *Smart Factory*, *CPS*, *IoT*, *Smart Sensor*, *Big Data*, *Additive Manufacturing*, and *Hologram* [3]. Based on the papers [4, 5], we list most similar technical concepts. They are Wireless Sensor Actor/Actuator Networks (**WSAN**) [6], the **Fog** (an extended concept of IoT that likes the cloud system, but built with embedded systems) [7], Machine to Machine/Man (**M2M**) [8], System of System (**SoS**) [9], etc. In this paper, we use the term "CPS" to denote such a kind of system.

1.1 Motivation and goal of this survey

According the statistic result of "web of science", CPS has been widely applied in 271 fields, which range from "*computer science*" to "*sport science*". The top 10 research areas are illustrated in Figure 1 and the publications in the decade years are shown in Figure 2. We analyze the papers on "*web of science*", "*IEEE Xplore Digital Library*", and "*ACM Digital Library* ", and the statistic results are shown in Table 1. Though many papers are published, few researchers comprehensively studied the dependability of CPS. I.e. "web of science" includes 5187 papers with the keyword "*cyber physical systems*" due 2017-6-22, but only 70 papers are returned with keywords "*dependability*" and "*cyber physical systems*". To further review, only 36 papers explicitly include "*dependability*" in the keyword section, and 48 papers of them discuss dependability in one or more section. Only one survey paper is found, which is about the challenges of dependable CPS infrastructures [10]. Others just mention the keyword "*dependability*" in text without further discussion. A similar conclusion can be made according the survey on component-based CPS architecting since 2015 [11], only 9 of 1103 papers relate to reliability, and 6 papers focus on maintainability.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

View Records / Exclude Records	Field: Research Areas	Record Count	% of 5187	Bar Chart	Save Analysis Data to File
	COMPUTER SCIENCE	3913	75.439 %		Data rows displayed in table
	ENGINEERING	2983	57.509 %		All data rows (up to 200,000)
	TELECOMMUNICATIONS	1147	22.113 %		
	AUTOMATION CONTROL SYSTEMS	1139	21.959 %		
	MATHEMATICS	737	14.209 %		
	ENERGY FUELS	581	11.201 %		
	INSTRUMENTS INSTRUMENTATION	354	6.825 %		
	ROBOTICS	335	6.458 %		
	COMMUNICATION	331	6.381 %		
	BUSINESS ECONOMICS	237	4.569 %		
View Records / Exclude Records	Field: Research Areas	Record Count	% of 5187	Bar Chart	Save Analysis Data to File

(76 Research Areas value(s) outside display options.)
(271 records(5.225%) do not contain data in the field being analyzed.)

Figure 1. Top 10 research areas of CPS (from web of science, due 2017-6-22)

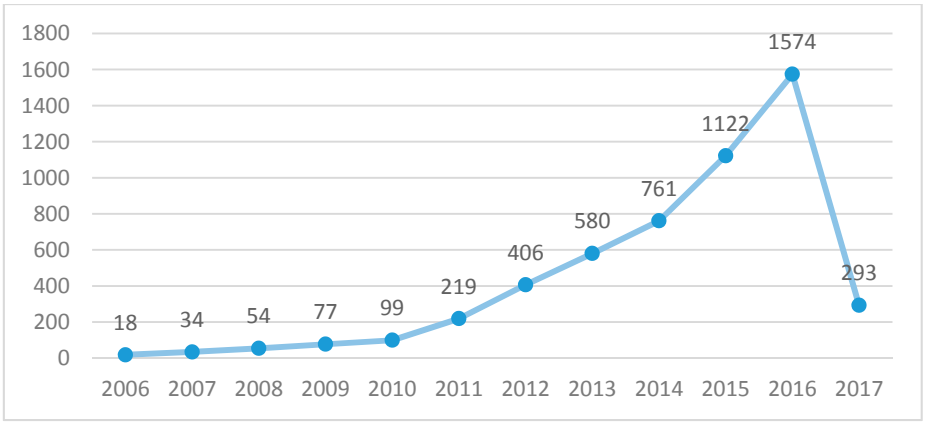

Figure 2 Statistic of published papers on web of science by year (due 2017-6-22)

Table 1. Paper statistic on CPS and dependability (due 2017-6-22)

Keywords	Web of Science	IEEE Xplore Digital Library	ACM Digital Library
Cyber physical systems	**5,187** (Topic)	**5,317** (meta data only) **1,371** (keywords[1])	**6,033** (full text) **415** (keywords)
Dependability & Cyber physical systems	**70** (Topic)	**72** (meta data only) **22** (keywords)	**4,146** (full text) **19** (keywords)
Reliability & Cyber physical systems	**464** (Topic)	**575** (meta data only) **69** (keywords)	**2,552** (full text) **33** (keywords)
Safety & Cyber physical systems	**539**(Topic)	**554** (meta data only) **91** (keywords)	**1,691** (full text) **63** (keywords)

[1] keywords is "Author Keywords". The search filed on "web of science" is topic, which includes title, abstract, author keywords, keywords plus; "web of science" doesn't support search within the field of "Author Keywords". The "meta data only" field on "IEEE Xplore Digital Library" includes the abstract, index terms, and bibliographic citation data. The "full-text" field on "ACM Digital Library" includes the abstract, index terms, text. The "Author Keywords" field means the keyword that Author listed in paper.

CPS can modify the real world, but cannot reverse the effects on physical space. It never emphasizes too much the importance of dependability to a practical CPS, especially reliability and safety. Otherwise CPS would not simply our life, but makes it tougher and more dangerous. A comprehensive survey of dependability will guide the further CPS research.

This work tries to provide a detailed, comprehensive survey on dependable self-managing CPS development and life cycle evaluation. The challenges and preconditions are also analyzed for

building a practical CPS. Some promising solutions are introduced for future research. CPS development involves many requirements, in this survey, we mainly focus on correctness, dependability and related characteristics that are unique in CPS.

In this paper, the three words "self-adapting", "self-healing" and "self-management" will repeat in many times. In addition, we use "self-adapting" to represent the ability of adapting the normal functions to new context which mainly highlights the correctness. "Self-healing" mainly speaks for the dependability, it is the ability of adapting the dependability management to new situations. "Self-management" includes the both concepts of "self-adapting" and "self-healing".

1.2 Literature Search rule

To cover as more papers as possible, we searched *Web of Science*, *ACM Digital Library*, *IEEE XPlore*, *Springer Digital Library*, *Elsevier Science Direct* and *Wiley Online Library*. Most cited papers are ESI papers. Both dependability management and CPS development are systematic engineering, which needs a long-form introduction and analysis. Therefore, we prefer to cite the *full paper* rather than the short paper. In addition, we prefer to cite the research on theory and practice rather than system introduction and systemic solutions rather than the concepts if they share the same topic/domain. ESI/SCI and highly cited papers are cited first. As there are several similar concepts to the item "CPS", the priority of items is shown as follow: "*CPS*"="*Industry4.0*" > "*IoT/Fog computing*" > "*M2M*" > "*SoS*" > "*WSN/WSAN*" > "*networked control system*" > "*embedded system*".

1.3 Structure of this survey

The rest of the paper is organized as follows: In section 2, we summarize four generic CPS architectures, the common requirements/characteristics and the common methods for these requirements. Furthermore, a more generic processing flow and a formal inter-processing flow of CPS are proposed. Section 3 briefly introduces the dependability and figures out the similarity between correctness and dependability with formal analysis. Then 5 new challenges of dependability are proposed in this section. Section 4 reviews the critical technologies and related correctness V&V methods, which includes the areas of self-adapting, real-time scheduling, observability and traceability. Section 5 focuses on architectures of the dependable self-managing CPS. The comprehensive methods for dependability management are introduced in Section 6, which include both design period and run-time methods, and also the related management tool sets are introduced. In Section 7, we make a special review on MDE based correctness and dependability development solutions for self-managing CPS. Then the shortages of current modeling and MDE technologies are analyzed in detail. In Section 8, we spell out the jigsaw of technologies and figure out the technical trends of dependable self-managing CPS. A concept of all-in-one solution is proposed for future CPS development. In section 9, some interesting features of CPS and issues beyond the technical area are discussed; the solutions for 5 challenges which proposed in section 3 are concluded.

2. Generic CPS and the common features

Lots of definitions are given from different perceptive for CPS [4]. Here, we just select two representative ones. One is proposed by CMU CPS group: "Cyber-Physical Systems (CPS) are integrations of computation, networking, and physical processes" [12]. Another is defined by NIST: "Cyber-Physical Systems or 'smart' systems are co-engineered interacting networks of physical and computational components" [13]. All of these definitions inevitably involve networking, computing, controlling, and interacting between cyber world and physical world.

CPS involves several basis technologies. Most of them have been investigated long time ago, like real-time system, control theory, embedded system, sensor network, etc. But CPS still can't work well if we just assemble them together, because the problem space of CPS are far beyond the domain of these technologies. To solve the problem, in the last decade, researchers have proposed lots of approaches on architectures/frameworks for CPS. For example, from the perspective of architecture, there are SOA (Service Oriented Architecture) based architecture [14-16], MAS (Multi-Agent System)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

[17-19], and other aspect oriented architectures like 5C architecture (5C stands for Connection, Conversion, Cyber, Cognition and Configure) [20], etc. From perspective of the key technical features, it contains self-management system, self-adaptive system [21], assembly-oriented architecture [22], collaborative model [23], etc. From the logical structure of CPS, it involves centralized architectures and decentralized architectures. Moreover there are also lots of explored technologies for building CPS, e.g. precision timed infrastructure [24], temporal isolation [25], precision time protocol [26], hierarchical scheduling, and various of modeling methods. Therefore, CPS has been applied in different areas: e.g. smart manufacture [17,27-29], smart transportation [30], smart city [31], precision agriculture [32], energy and power system [33], entertainment [34], etc. The further surveys on application areas refer to papers [4,5,35-36]. The architectures/frameworks for self-managing CPS will be discussed in detail in section 5.2.1.

2.1 Brief views of generic CPS architectures

One promising design of these architectures is the universal reference architecture, which is shown in Figure 3. It is proposed by the CPS public working group of the National Institute of Standards and Technology (NIST) in 2014 [13]. Considering the supporting from NIST, the reference architecture will be a strong competitor to the standard of architecture for the commercial CPS. This reference architecture focuses on decoupling CPS. From the vertical view, it introduces the CPS processing stack, which includes the domains of IoT, control system, modeling and simulation, business planning process (BPP) optimization, human machine interface (HMI), etc. From the horizontal view, it lists the technical requirements that involve timing guaranteeing, standardization and X-ability, etc. According to the reference architecture and the definition mentioned earlier, the CPS involves physical process, sensors (networks), data analysis system (decision support system, DSS) and actuators (networks). Meanwhile, even human beings can join the close-loop of processing and be a subsystem of the CPS.

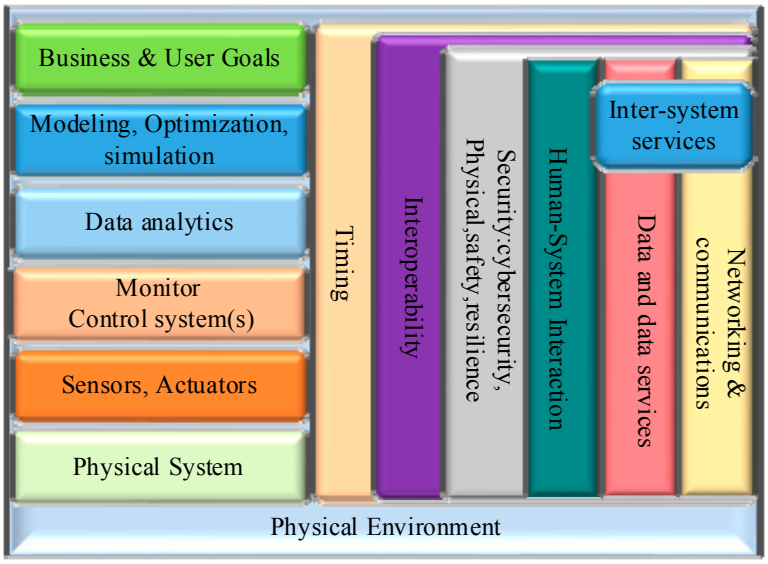

Figure 3. NIST reference CPS architecture

There are lots of dimensions to classify CPS, i.e. one famous classification is the concept map proposed by Lee's group [12]. As the architecture strongly affects the dependability, here, we classify CPS's architectures into 4 categories based on the structure. 1) **Centralized architecture**, which contains a global and unique DSS that analyzes the most of the events/information and generates all the decisions. 2) **Hierarchical architecture**, which comprises of several DSS subsystems with different power based on their position. 3) **Decentralized system,** contrary to centralized architecture, which has no obvious core and most of the subsystems are equal in function. 4) **Hybrid system,** in some sense, is an elaborate composition with three above architectures.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

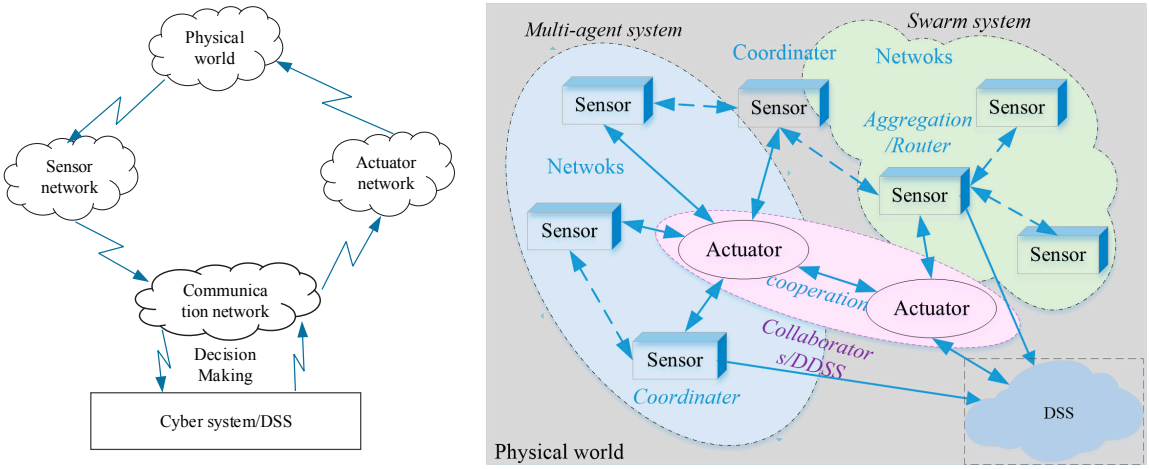

Figure 4. Generic CPS architecture: (**a**) Centralized/hierarchical CPS; (**b**) Decentralized/Hybrid CPS

The generic centralized architecture of CPS can be abstracted as Figure 4 (a). The function of sensors, DSS, and actuators are separated. Take the centralized CPS proposed in [37] as an example, data are collected and transmitted to DSS with IoT, and then analyzed with big data analysis on the cloud system (DSS servers). With the priori knowledge or the help of supervises, DSS makes decisions and sends them to actuator network, then the target actuators selectivity respond to the commands and take actions to transform the physical world. From the view of the maturity of technologies, the centralized CPS is the most controllable and dependable system compared to other architectures. However, the scalability of centralized CPS is limited because of the unavoidable communication delay. Apart from the shortage in flexibility, the centralized CPS also suffers single point failure, which decreases the availability, yet most CPS should provide 24/7 service.

Hierarchical CPS is one common solution to improve the scalability of centralized CPS. Just like the political structure, the local DSS can analyze partial events and make the simple decision. Meanwhile, it will send the fusion data to upper level DSS system, i.e. the graphical hierarchical CPS architecture [38,39]. Similarly, if the information is not sufficient to make decisions at current level, requests cascade up through the successive levels until to the top layer system. Compared with centralized CPS, hierarchical CPS is a tradeoff between scalability and real-time processing. As a result, local DSS should be smart enough to handle as many events as possible, and these processes should perform automatically because it is difficult for supervisors to monitor and interfere low-level subsystems. Meanwhile, the hierarchical architecture suffers more cascading failures. In addition, local DSS is short-sighted. Relying on local decision excessively would be rash and unsafe. It needs great wisdom and rich experience to make reasonable compromise on the functions and dependability between different hierarchies.

Decentralized architecture is a widely researched scalable solution which is illustrated in Figure 4 (b) (without DSS), i.e. the multi-agent solution in [18]. The agents can negotiate with each other based on predefined rules, and form a "team" to solve problems together, e.g. querying the data with the highest degree of confidence, collaborating and feeding back the weight to optimize the rule. In short, decentralized CPS is highly flexible but unpredictable and untraceable in structural variation. Autonomic computing system (ACS) is the common concept to build a dependable decentralized CPS, it integrates self-management/self-healing into the system. Yet self-healing introduces a self-referential paradox, which will be discussed in section 3.3. Besides, the local cluster is also myopic and unresponsive, because information diffusion is inefficient in decentralized systems. Furthermore, the resource utilization in a decentralized CPS is lower than the centralized architectures, because repetitive information should be recorded in each subsystem.

To overcome the drawbacks, hybrid CPS is orchestrated with these architectures at different levels and in different granularities, which is illustrated in Figure 4 (b) (with DSS). The subsystem may be a centralized/hierarchical system, i.e. connected vehicles; or maybe a decentralized system if

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

it is widely deployed [40]. Consequently, both centralized and decentralized structures exist in a hybrid CPS. Hybrid CPS tries to balances the flexibility, scalability and performance against the controllability, integrity and intelligence. In addition, it is also a good solution to integrate and extend the legacy system. However, hybrid CPS also introduces great challenges on dependability, the physical topology and logical topology are so complex and changeable that all current dependability management technologies fail to analyze or manage their dynamic behaviors. Therefore, a hybrid CPS with bad design suffers all the failures that may occur in other architectures.

2.2 Common characteristics of CPS

With the development of ICT and integrated manufacturing technology, to improve productivity, flexibility and sustainability, CPS is adopted in wider domains and more complex environment [4,36,41]. It introduces various requirements into the system. Lots of researches discussed the requirements/indexes of CPS [4,41-44]. We summarize and classify these requirements and the summary is shown in Figure 5.

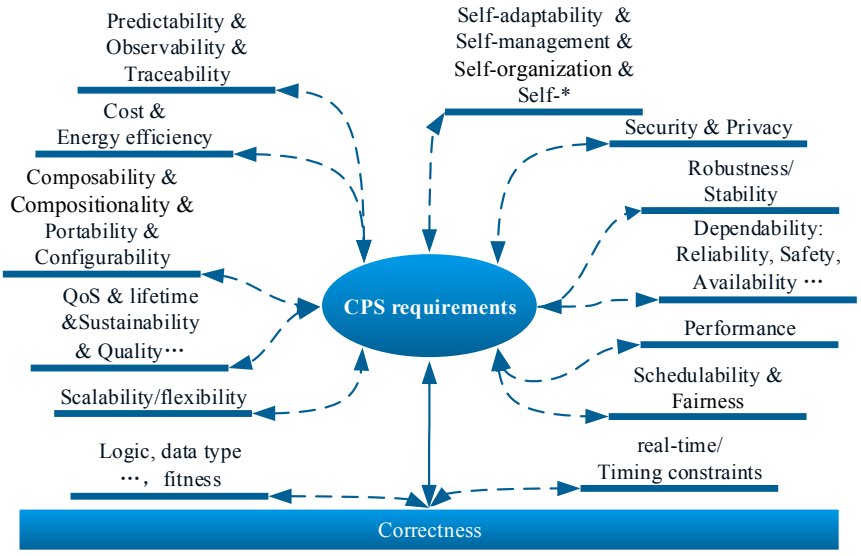

Figure 5. The requirements of CPS

As shown in Figure 5, CPS involves various requirements, such as **performance**, **schedulability** and **fairness**, **real-time** and **correctness**, etc. The correctness requirements include **functional** correctness and **temporal** correctness and some correctness enhancing requirements which are **composability, compositionality, self-* abilities**. The maintainability-related requirements consist of **predictability**, **observability, traceability, maintainability,** etc. The comprehensive requirements are **dependability**, **robustness**, **stability**, **security**, **sustainability, quality of service (QoS)** and **lifetime,** etc. In addition, the cost and scale related requirements contain **cost/performance ratio**, **energy efficiency**, **scalability** and **flexibility,** etc. Moreover, there are still abundant detail requirements for each level and each capacity of CPS. Due to space limitations, we don't list one by one.

These requirements put forward a huge challenge on CPS engineering and requirements V&V. The design of CPS is an art of compromise. Tradeoff among these requirements is a multi-objective optimization problem, which is an NP-hard. To make things even worse, these requirements have nonlinear relationships and form a complex effect network with each other. One tiny change on this network can lead to big effects on the whole system, which is famously known as the butterfly effect. Therefore, it is almost impossible to find the best solution that fits for all requirements. Helplessly, we have to search for a suboptimal solution with few defects on each requirement. What's more, we should design a system that can solve the multi-objective optimization problem automatically by

taking into account the run-time context at the same time. To simplify, we will mainly focus on dependability and its sub-properties, the correctness and its' related requirements in this survey.

It's no doubt that **correctness** is the precondition for all other requirements, in return other requirements have complex positive and negative effects on correctness. E.g. the efforts for high **flexibility** introduce complexity which is harmful to correctness, and the flexible functions also make the correctness validation extremely complex. **Dependability**, especially the reliability, improves the correctness mostly. Because reliability mechanisms can keep the system perform normally even faults occur. However, fault-tolerate communication protocols also introduce time-varying delay which is harmful to time critical applications. Correctness V&V is not only about checking the consistency of static constraints, but also the dynamic behaviors in random cases, especially in the worst case. The correctness includes many sub-properties, such as the correctness of functional logic and data types and interfaces, the semantic fitness of context, real-time, timing, etc. And the temporal issues become especially critical in CPS [45], reproducing the timing behavior sequence is a necessary condition for causal inference for building smart CPS. The correctness V&V of CPS is still an open issue and it will be further discussed in section 4.

Due to the increasing complexity, the methodology of divide and conquer becomes less and less efficient. As system contains massive subsystems, the divisions between holistic behavior and individual behavior become larger. E.g. the *parasitic effects* in large scale integrated circuit design, the *electromagnetic interference* between electric components, etc. This phenomenon is a famous problem discussed in holism and reductionism. To the best of our knowledge, there is no efficient method and tool to well understand and handle the complexity at the synthesis level.

On current popular solution to solve the challenge of complexity are improving the **self-adaptability/self-management** of the system. The goal of self-adapting/self-management is to manage the system dynamically and automatically [46], and the concept of **self-learning** [47] goes even further. The survey [48] detailed the challenges of self-adaptation in dynamic environments from the view of dynamic optimization problems. To enable these self-abilities will inevitably introduce additional complexity, which sounds like using one complexity to deal with another complexity. For further understanding, we refer the reader to the surveys [17,46, 48].

Another solution to handle the complexity is improving the **composability** and c**ompositionality (C&C)** of components [4, 49-50]. Composability comes from the philosophy of reductionism, and on the contrary, compositionality is from of holism. As a kind of SoS, improving C&C is one best solution for CPS design. C&C has been mainly researched in the area of model driven engineering/development (MDE, MDD), which is very popular in MDE based CPS development and maintenance. Composability and compositionality are two positive premises for the self-organization system. A system with a good capability of C&C can be reorganized and reconstructed more easily and the new system contains higher correctness. Sztipanovits et al. aimed at the challenges of cross-domain *heterogeneous interactions* among physical and computational/networking domains, and presented a theory of composition to improve stability [51]. Nuzzo P. et al. adopt *contracts based components specification* and *abstraction* and provided a formal supporting for CPS design [52]. Paul A. et al. proposed a general formal framework for *architecture composability* based on an associative, commutative and idempotent architecture composition operator [53]. Sanjit A.S presented a formal methodology and a theoretical verification and synthesis framework that integrates *inductive learning* with *deductive reasoning* [54]. Stavros T. detailed the key principles of compositionality focusing on *interface design* for MDD [55]. A compositional *specification* theory for components *reasoning* is proposed in [56], this specification theory uses synchronistic input and output (I/O) actions to abstract the temporal ordering of behavior.

Dependability and correctness are highly interrelated and mutually reinforcing. We will detail analysis the challenges, causes and related methods of dependability in section 3 and 6. Here we focus on the common structural characteristics related to dependability.

Hybrid structures: As mentioned earlier, hybrid CPS is highly scalable and flexible, and it has also a good ability of (legacy) system integration. For this reason, on one hand, CPS modeling, design should take into account the hybrid structures. On the other hand, physical world is normally

modeled as a continuous system; meanwhile the cyber space, especially the digital (processing and computing) systems are discrete. The combination and cooperation between the continuous model and discrete model should be deeply investigated. Moreover, we should rethink the multidisciplinary features of CPS. The complex structures and theories of hybrid CPS lead dependability analysis and guarantee to a big challenge.

Heterogeneous components: No single technology can solve all problems. To build a large scale SoS for different situations, various types of components with different technical standards should be integrated into CPS, i.e. IEEE 802.11 (Wi-Fi), IEEE 802.15.1 (Bluetooth) and IEEE 802.15.4 (ZigBee, 6LoWPAN) are integrated in IoT [57-58]. In addition, even one function may be implemented in heterogeneous components for various reasons, e.g. saving energy with heterogeneous CPUs and improving the reliability with redundant heterogeneous components. In some sense, heterogeneous components and protocols can improve the dependability of the system. To MDE, heterogeneous tools will be integrated into the simulation platform [59] and the platform should also be well implemented and carefully verified to guarantee the correctness and dependability of the platform itself.

Distributed networked: Distribution and parallelism are two features of the physical world. To recreate the physical events and to interact with the physical world, distributed networked control system is the first choice. Information transmission takes time in distributed systems, and clock synchronization is the key methods to achieve consistent timestamp in distributed systems [45]. To guarantee the timing of events, a new hardware and software must be invented for CPS to support the global reference time [60] which is also called the Newtonian time. Guaranteeing the dependability of a large distributed networked system (CPS) is still an open issue.

Large-scale & big data: The scale of CPS and data are growing rapidly. A large number of sensors, computation and coordination nodes, cloud servers and actuators are connected in CPS. These subsystems create vast amounts of data continuously, which include the physical events, the status information about CPS itself, the control commands and the failure messages. Large-scale and real-time big data processing challenge the performance of CPS and the privacy of data [61], but on the other hand, the multi-domain big data is a good evidence for fault diagnosis to improve the dependability of CPS.

Dynamic (inter-)operation: As mentioned earlier, the physical world is a continuous system where the environment will affect the CPS continuously. All subsystems of CPS should be synchronized and coordinated with each other, then interact with the physical world dynamically. As a kind of subsystem, similarly, the dependability manager should monitor and heal the failed subsystems continuously. Yet the changeable structures introduce a big challenge to dependability management.

Human-in-the-loop: As CPS is assimilated into our daily lives more and more deeply, human-in-the-loop CPS is a certain trend. Human becomes as an uncontrollable subsystem whose behavior requires special consideration, e.g. lack of concentration, emotional behavior, unstable skills, etc. To design such a system should take the human machine interface (HMI) and the model of human behavior into account. Zoltán R. et al. summarized the current theories and practices on *interaction between human and system*, then proposed the research challenges of HMI in [62]. And a concept of *human service capability description model* is presented in [63]. Azfar K. et al. discussed the different *safety* approaches for heavy payload robots, highlighted the technology limits and real-time issues in human robot collaboration CPS. A generalized guideline is proposed with a case of human and robot interaction in industry [64]. Research on the dependability of human-in-the-loop CPS is difficult but an emergent issue.

2.3 Generic processing flow of CPS

According to the general definitions [3,4], the participators of the CPS are sensors, actuators, computing system (data analysis and decision support system), and commutation network among them, and some definitions may include the human beings. Most of researches ignore the physical system, which is the protagonist of physical space, or just mention it as the physical space/world

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

without further discussion. The main purpose of the CPS is to interact with the physical world, without the physical processing, the system is incomplete, especially for models. To build a dependable CPS, we should not only consider the functions from cyber perspective, but also the physical effect on cyber devices, and even the effect on the human being. Imagine that a man drives a car in terrible cold weather: for the driver, the low temperature will reduce his reaction reflex. For the car, the low temperature will reduce the friction of tire, weak the strength of materials, and even frozen the lubricant. It is no doubt that the car behaves abnormally. Moreover, in the case that the car moves downhill, the brake pads will be heated because of friction and the braking effect will become weaker. As a result, the braking distance will increase significantly.

From this point of view, we raise the position of physical space and propose a generic processing flow of environment in-the-loop CPS as shown in Figure 6. The cyber system core, which is the common concept of CPS, consists of sensors (network), actuators (network), networks and DSS. The physical system includes the entities of sensor and actuator devices, the human beings and the physical environment around them. We don't refer the DSS to physical space because DSS is deployed in the data center. It is in an ideal artificial environment, which is barely affected by the physical/natural world. In this flow, the blue arrows represent the common data processing flows in general CPS. We add yellow arrows to show the *effects of the physical environment* on the human and the devices. And the green arrows represent the *status information of infrastructures*, e.g. sensor, actuator and network, which is very important to dependability management. There are two roles of human being, one is the CPS manager (yellow) and another is the participator (blue) who joins the CPS direct as a subsystem. These two kinds of human can communicate with each other through extra channels like telephone, Internet, etc.

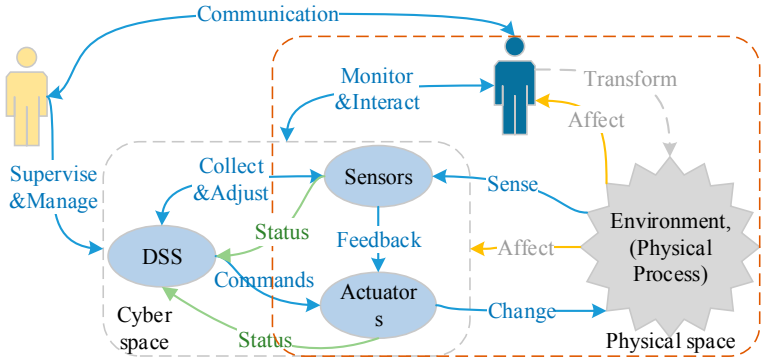

Figure 6. The generic processing flow of environment-in-loop CPS

The formal interaction flow between cyber space and physical space is illustrated in Figure 7. $s_c(t)$ and $s_p(t)$ are the statuses of cyber space and physical space at time t respectively. $s_c^m(t)$ and $s_p^m(t)$ are the observed statuses of cyber space and physical space respectively, where $s_c^m(t) \subset s_c(t)$ and $s_p^m(t) \subset s_p(t)$. The inputs for DSS are $s_c^m(t-j,\cdots,t); s_p^m(t-i,\cdots,t)$. $\{D_c(c,t)\}$ and $\{D_p(e,t,p)\}$ are the decision for cyber space and physical space. $\oplus\Delta s_c$ and $\oplus\Delta s_p$ represent the adjusted statuses of cyber space and physical space respectively. $\oplus(\int_t^{t+1} f_p(t)dt + c)$ is the inertia effect of the physical phenomenon. The physical status $s_p(t+1)$ at time $t+1$ is affected by cyber space, the inertia effect and the other entities like neighbor nodes, human being and natural phenomenon, etc.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

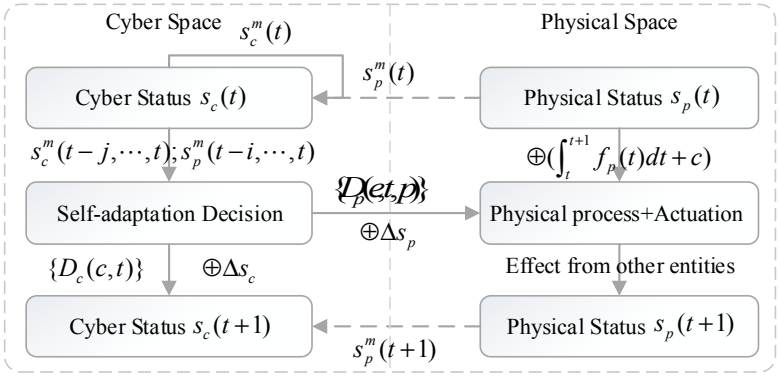

Figure 7. The formal interaction between cyber space and physical space

Notice that all the flow paths and processing policies changes dynamically. Following the self-adaptation rules, the structure of CPS changes continually. Hence, to be dependable, CPS should not only monitor the physical environment, but also collect the statuses programs and of devices. In some sense, the information of device statuses is also one kind of physical information. Furthermore, the DSS should take the effects of physical processes on devices into account during decision making. And for human-in-the-loop CPS, DSS even should consider the intentions of human beings, interact with human and reserve enough response time for human beings.

3. Dependability of self-managing CPS

Self-management (self-adapting) system has been widely researched [46, 65] for complex applications. As a complex system, we believe the self-managing CPS will be the mainstream in the future. With more and more self-adapting strategies are added into CPS, CPS become highly flexible in function and structure. It improves the flexibility of CPS, but also makes the dependability analysis and maintenance more difficult. Numerous researchers highlight the significant role the dependability plays in a practical CPS [4-5, 10, 41-42, 64]. Correctness is the premise to CPS, and dependability is the cornerstone to practical CPS. Researching the dependability of self-managing CPS can improve the practicability, decrease the cost of development and maintenance, and accelerate the industrialization of CPS.

Following autonomic computing, Harald et al. reviewed the self-healing approaches for large scale information systems, discussed the relationship between self-adapting and self-healing, and analyzed the properties of self-healing researches [66]. This survey can provide a significant reference value for the cloud based DSS design. Yet it doesn't take the temporal constraints into account which is very important to CPS. This is also one reason why we make this survey.

In this section, we first recall the definition of dependability and discuss the issues of the self-managing CPS, and formally analyze the relationship between dependability and correctness; then we present five new challenges for the dependable self-managing CPS.

3.1 Quick review of dependability

Dependability is an integrating property and has been researched for a long time. It encompasses three concepts: the threats consist of errors, faults and failures; the attributes include reliability–R(t), availability–A(t), safety–S(t), confidentiality, integrity, and maintainability–M(t); the means to achieve the dependability contain fault prevention, fault tolerance, fault remove, fault forecasting. And more detailed introduction on dependability and its threats and attributes are referred to [67]. The famous dependability tree is shown as Figure 8 and the failure processing flow is shown as Figure 9. Some researchers also regard the security as a sub-attribute of dependability. Due to the limited space, we just discuss the dependability without security.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

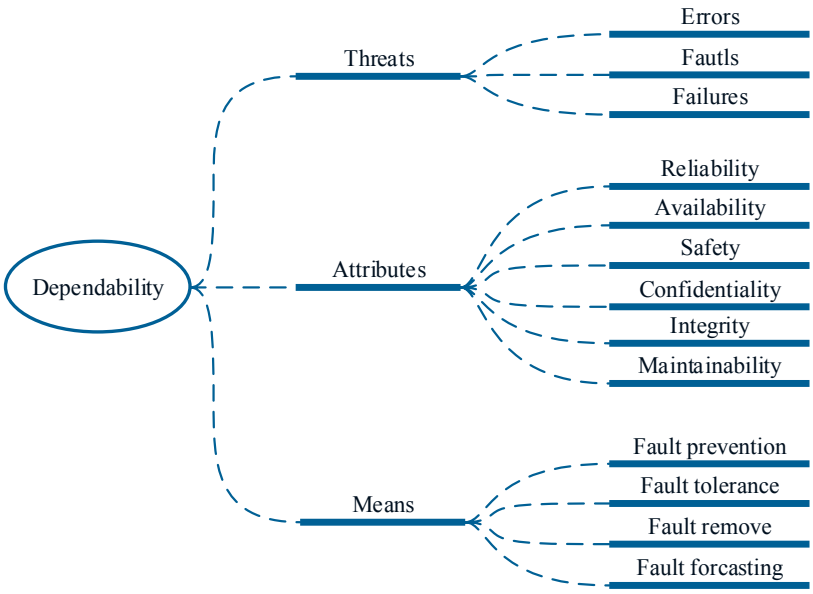

Figure 8. The dependability tree

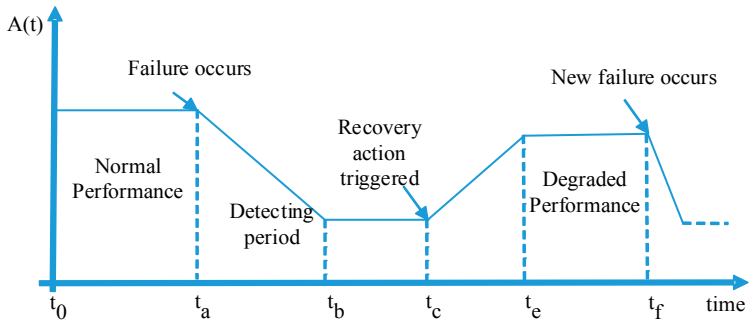

Figure 9. The failure processing flow

Reliability is continuity of correct service with the quantitative index of mean time to failure ($MTTF = R(t) = t_a$), it is a directly temporal measure of the correctness of a service. Availability is the readiness for correct service with the quantitative index of mean time between failure ($MTBF = t_e$), it is a percentage of the available time of the service. The maintainability is also a temporal index to define the ability to undergo, modifications, and repairs, it is generally defined by the mean time to repair ($MTTR = M(t) = t_e - t_a$). The relationship of reliability, availability and maintainability can be formalized as $A(T) = \int_0^T A(t)dt / T = R(T) / (R(T) + M(T)) = t_a / t_e$. Safety is the ability to keep the system away from catastrophic consequences, yet we general quantify the safety with the expected loss caused by failures. Confidentiality is defined as the absence of unauthorized disclosure of information. Integrity is described as the absence of improper system alterations.

Generally, to traditional systems, we just need to focus mainly on some partial attributes of dependability, e.g. reliability to rocket and availability to web service. Yet, it is more serious to CPS. As a large scale system, CPS needs good maintainability to locate the failed subsystems and recover service quickly. For 24/7 service supporting, CPS needs high reliability, availability and maintainability. Considering the critical application, CPS should be safety enough. To adapt to the uncertain situations smartly, CPS should provide integral and consistent information for DSS. Considering the large potential value of data, guaranteeing the confidentiality of information in the CPS is necessary. To develop a dependable self-managing CPS, we should take all these six attributes, but not limited, into account.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

3.2 The relationship between correctness and dependability of self-managing CPS

Roughly speaking, the most policies of self-healing mainly work in a feedback-adjust pattern, which is famous for detection-diagnose-recovery. This pattern can be abstracted as Figure 10. From the perspective of structure, it can be classified into two classes. (a) The self-healing measures are integrated deeply with normal function, the component will adjust its behavior for dependability while executing, i.e. some dependable control mechanisms. (b) The self-healing measures and the normal function are divided into two parts, in other words, dependability management and normal function are processing independently in time and space, i.e. most dependability measures in computer science area.

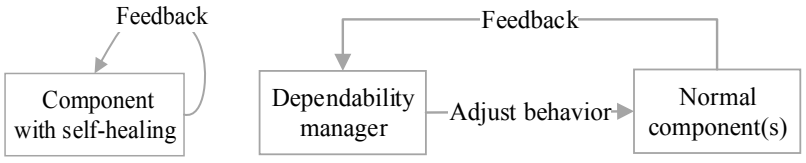

Figure 10. The patterns of dependability enhancing based on feedback: (**a**) Internal self-healing in a signal component; (**b**) External self-healing with the pattern of detection-diagnose-recovery

To self-adapt to a new context, the system should adjust its states or thresholds based on the current feedback statuses. As a typical feedback pattern, the correctness at time $t+\Delta t$ can be written as $C(t+\Delta t)=f_c(S_c(t)\oplus\Delta S_c^d(\Delta t),S_p^*(t+\Delta t))+f_\beta(S_c(t),S_p^*(t+\Delta t))$, it is a predicted value based on the statuses $S_c(t)$ at time t. $\Delta S_c(\Delta t)$ is the statuses corrected by self-adaptation and $S_p^*(t+\Delta t)$ is the real statuses of the physical world at $t+\Delta t$ which is generally approximated with a predicted value in practice. f_c is a fitness function of the correctness (matching degree) between the predicted statuses $S_c(t+\Delta t)=S_c(t)\oplus\Delta S_c^d(\Delta t)$ and the desired statuses $S_p^*(t+\Delta t)$. For an ideal adaptation, $S_c(t+\Delta t)=S_p^*(t+\Delta t)$. f_β is the inertia effect of current statuses in new situation which may have a negative effect on adaptation.

The dependability of the self-healing pattern (a) can be written as $R_n(t+\Delta t)=f_\alpha(S_c^{cyb}(t)\oplus\Delta S_c^{cyb}(\Delta t),S_{cyb}^*(t+\Delta t)*R_n(t))+f_\gamma*R_n(t)$ and the pattern (b) can be formalized as $R_n(t+\Delta t)=f_\alpha(S_c^{cyb}(t)\oplus\Delta S_c^{cyb}(\Delta t),S_{cyb}^*(t+\Delta t))*R_m(t)+f_\gamma*R_n(t)$. $R_n(t)$ and $R_m(t)$ are the dependability of the normal component and the dependability of self-healing component at time t respectively. f_α is a fitness function that represents the dependability measures. f_γ is a reliability decay function which depends on the environment. And $S_{cyb}^*(t)$ represents the actual status of devices/infrastructures (which are shown by the green arrows in Figure 6) and $S_{cyb}^*\subset S_p^*$, $S_c^{cyb}\subset S_c$. Considering that S_c^{cyb}, S_c, S_{cyb}^* and S_p^* are spatial/entity and temporal dependent, self-healing decision making is an extremely complex and heavy stuff.

Self-adaptation tries to improve the correctness of the system and the self-healing focuses on enhancing the dependability of the system. Both of them adjust the next-step actions of system based on current observed statuses. Considering the similarity between the first subexpressions of $C(t+\Delta t)$ and $R_n(t+\Delta t)$, and as well as the domain $S_c^{cyb}\subset S_c$, self-healing can be regarded as a common self-adapting service but dedicated to the system itself. In this context, dependability suffers all the challenges of correctness and is even more critical, i.e. fault-tolerate real-time scheduling [68]. As Gabriel et al. claimed in [69]: "Self-adaptive systems tend to be reactive and myopic, adapting in response to changes without anticipating what the subsequent adaptation needs will be". This drawback is extremely dangerous, especially for safety critical CPS, because self-healing would not be triggered until a fault has been detected. Unfortunately, the healing actions will be too late to avoid

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

failures in most cases. Considering the similarity between correctness and dependability to self-managing CPS, we use **C&D** as the abbreviation in later chapters.

Theoretically, healing actions are impossible to keep pace with the changeable and unpredictable real physical world. Fault prevention is an alternative solution, but it is difficult to forecast the correct $S^*_{cyb}(t+\Delta t)$ and $S^*_p(t+\Delta t)$, because both the CPS and the real world are nonlinear, thinking about the famous butterfly effect. What is worse, we cannot hold the assumption $\Delta t \to 0$ because of the non-negligible delay of network transmission. As a result, $S^*_p(t)$ may be quite different with $S^*_p(t+\Delta t)$ indeed. Meanwhile, the correctness/fitness of an action depends on the situation that the system stands. The S^*_{cyb} of system decays continually and the environment S^*_p is volatile, the dependability strategy $\Delta S^{cyb}_c(\Delta t)$ based on $S^{cyb}_c(t)$ may not fit for $S^*_{cyb}(t+\Delta t)$, i.e. some nodes may have crashed in the time Δt but $\Delta S^{cyb}_c(\Delta t)$ takes effect only when all nodes work together.

Hence, the self-healing strategy should take all factors into account and consider several steps ahead to reserve time for taking action. Yet such active self-healing introduces too much overhead that it is impractical for embedded systems. This puts the self-healing in the difficult position. The naive ideas to defuse the situation are 1) **improve the reliability $R_n(t)$ of subsystems**, 2) **improve infrastructure to keep the assumption** $\Delta t \to 0$ (i.e. high speed real-time network). Another direction is to find the **invariance properties (symptoms)** of components behavior that can tolerate the large Δt. The **prophetic strategies** are alternative solutions, which are detailed in section 4.1 and in section 5.3.2.

3.3 The challenges of the dependable self-managing CPS

Though lots of researches on dependability have been done, developing a dependable complex system is still an open research issue. Because 1) the cost-effectiveness of developing a highly reliable system increases exponentially [70]. 2) The coverage of testing is an NP-hard problem [71] which implies that no system can be completely correct. 3) The inconsistent conclusions caused by Byzantine failures during communication. 4) The increasing complexity makes completely V&V impossible. Despite the difficulties in engineering technologies, dependability is an unsolvable problem to the Turing machine system because of halting problem [72], which is a famous paradox of Gödel's incompleteness theorems in computability theory. In short, the halting problem shows that a program can't determinately finger out whether the program will halt (be failed) or not (correct) by itself over Turing machines. Fault/status detection is such a decision problem, which implies that fault detection is undecidable to close-loop Turing machine systems. Unfortunately, current computing systems are Turing machine based and fault detection is the premise of other means of dependability.

Dependability is tightly intertwined with correctness in self-managing CPS. In order to adapt to the new situation perfectly, CPS will change its structure, which challenges the dependability engineering and V&V. Because the structures of traditional systems are predesigned and are fixed or rarely changed throughout the whole lifetime. Systems just adjust their behavior slightly, e.g. selecting a new logical branch, i.e. the if-else pattern; updating the states of next operation, i.e. the feedback control $x(t+1)=Ax(t+1)+Bu(t)$; or renewing the threshold/reference value, i.e. Kalman filtering based adjusting $x(t+1)=\sum_{i=0}^{n} w_i x(t-i)$. Systems barely change the topology of the (physical/logical) components. Even the TMR (Triple Modular Redundancy) system just partially achieves the goal. It decreases the connection degree between the majority gate and the modular from three to two and then to one. But no one new modular, which is not predesigned, is appended to the original system. No new (physical/logical) link (that is not predesigned) is created in the degraded system.

Yet, to a self-managing CPS, the structure and even the architecture may change to adapt to the new context [73], which means the link, the ports/interfaces for the new cooperator, and the matrix

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

of guard states may be absolutely different. Despite the traditional issues, the self-managing CPS still faces five new big challenges:

RQ1: ***How to quantify continuously the dependability of each subsystem under different context, especially in the unusual situation with sparse evidences***? I.e. the reliability of subsystem s under the context c_1 at time t_1 is $p_r(s \mid c_1(t_1))$, and c_2 at time t_2 is $p_r(s \mid c_2(t_2))$, how to model the probability function during $t_1 \to t_2$ as we can't build a continuous model to describe $c_1 \to c_2$?

RQ2: ***How to adapt the CPS's behavior when a subsystem joins or quits and guarantee the semantic consistency of correctness and dependability (C&D)***? I.e. imagine the scenario that adding an intern to a complex project midway, the ability of the intern, the arrangement of tasks and the communication. What's more, the project manager can even be replaced.

RQ3: ***How to tradeoff between dependability and other requirements dynamically?*** I.e. guaranteeing real-time performance in a degraded and energy critical CPS.

RQ4: ***How to make correct decisions for dependability management with incomplete, inconsistent, inaccurate data and disordered events?*** E.g. data lost, message delay, event disorders due to various reasons, such as the unpredictable network behavior caused by imprecise synchronization.

RQ5: ***How to remove cumulative errors and continuously assure the dependability of the program/subsystem that manages the dependability of the system*** (it is a self-reference problem similar to the barber paradox, because the dependability manager is also a subsystem of CPS)?

The challenges of RQ1, RQ2 and RQ3 are about the knowledge expression and application of dependability. The RQ4 and RQ5 challenges are a problem about the correctness and reliability of dependability management.

4. Approaches on the correctness V&V of self-adapting CPS

As analyzed in section 3.2, to self-managing CPS, dependability and correctness share numerous similarities. Compared with dependability, more researches have been done on correctness V&V of self-adapting CPS. Studying the approaches on the correctness V&V can improve the research on dependability V&V for dependable self-managing CPS.

A CPS with high correctness should behave properly in all situations. The proper behavior means that the CPS should take the right actions at the right time in the right place. In this context, taking right actions is performing the operations which fit the context best, the right time means at the precise time, the right place means choosing the correct objects at an exact physical location. The correctness issue in CPS is mainly introduced by the complex design to meet the complex requirements and the volatile application environments. And the detail discussion between complexity and correctness refer to the book [74] and paper [75].

Correctness is a compound requirement, which includes many sub-properties, such as interface constraints, real-time, the static functional logic, the timing behavior, fitness of context, etc. To guarantee the correctness of a CPS, we should validate every automatic constraint of each unit component, the all compounded constraints of every composition constructed with those unit components and the tradeoff among constraints in all cases.

The assurance of correctness is a traditional but still unsolved problem. Lots of testing and formal methods have been proposed which involve from consistency verification [76] to dynamic logic validation [77], and the targets include circuits, chips, network, software, systems and requirements, etc. [76-79]. Sara et al. analyzed the need of multiple levels testing on CPS and made the state-of-the-art survey on testing methods and conclude the challenges in CPS testing [80]. To avoid duplication of efforts with other surveys, in this section, we focus on the correctness related approaches on **V&V** of **self-adaptability, real-time scheduling and timing**. We discuss these characteristics in three parts: 1) the state of art of related technologies, 2) the V&V on them, 3) a short summary. Finally, the measures of **traceability** and **observability** are investigated for building the trustable human centered CPS.

4.1 Self-adaptability and validation

With the rapid increasing of complexity of applications, self-adaptive and context-aware become very popular in recent years. Surveys [46, 65] are recommended to help readers can catch a quick glimpse of relative self-adaptation methods in engineering. Self-adaptation strategies are also widely explored for CPS. Pekka A. and Jouni M. introduced a *middleware-based real-time SOA* for CPS to improve the reliability of CPS [14]. Zhang Y.F et al. proposed *agent based framework* and improved the reconfigurability and responsiveness of the shopfloor with gray relational analysis and the hierarchy conflict applied resolution method [19]. Compared with the biological adaptation of human brain, Preden J. S. et al. highlighted situation *self-awareness and attention* for fog and mist computing. The authors argued that CPS must be aware of its situation to perform the "correct" behavior that fits the current situation best [81].

SOA based system [14-17] and multi-agent system (MAS) [18-19, 82] are the two mainstreaming frameworks for self-adaptive systems. SOA framework is generally applied in centralized or hierarchical CPS. The capabilities of every service are described with WSDL following the Resource Description Framework Schema Specification (RDFSS) and represented the service with a UUID. Other services or operators can query the target service based on the content of RDFSS and UUID. To build large scale distributed CPS/IoT system, services may be packaged as Web resources [82]. Meanwhile MAS is popular for decentralized CPS, agents cooperate with each other and make decisions based on elections or voting, or some similar statistical information. Paulo L. et al. analyzed the state of art agent technologies in industrial CPS, proposed the design principles and analyzed applicability of MAS across 11 emerging CPS aspects [17].

Currently, most effects on self-adaptation are based on *rules* [37], *swarm intelligence* [18] or *knowledge* [83] or *ontology* [84]. With the improvement of big data analytics technologies, more and more machine learning based self-adapting solutions are researched, these approaches are smarter, and can adapt to various environments automatically. Most machine learning based approaches in CPS aim at data analysis. E.g. Wang H.T. et al. predicted the influence changes of facilities over dynamic vehicles with *Bayes method* and *trajectory-based Markov chain model* [85]. Chen Y.Y et al. introduced and objects searching system for IoT with *context-aware hidden Markov model* and ontology method [84]. Martin M. et al. compared the effect of deep learning and four multi-class classifiers which include *multiclass neural network*, *multiclass decision jungle*, *multiclass logistic regression* and *multiclass decision forest* on data processing in Industry 4.0 [86]. Some other researches focus on the behavior monitor and control. E.g. Kush R.V. et al. redefined safety based on the terms of risk, *epistemic uncertainty and harm for statistical machine learning* [87]. Alexander Maier presented an appropriate passive online learning algorithm for cyber-physical production systems *timed automaton learning* algorithm [88].

Yet these approaches need lots of training examples, and the overheads are too heavy to apply them on tiny sensor and actuator nodes. What's more, the connections between phenomenon and effects are based on statistical results of the whole entities. Such connections are lack of personalization, which may be unfit in some cases because CPS applications are location-aware and object-aware. It still needs to learn on time series of historical behaviors for every subsystem. Meanwhile, most self-adaptation actions are taken after the target phenomenon has been observed, which implies that there is almost no time left, especially for critical system. Gabriel A.M. et al. presented a *proactive latency-aware, self-adaptation* with the offline formal specifications of the adaptation tactics, a stochastic behavior model of the environment and a runtime probabilistic model checker [69]. Proactive adaptation should predict the context of system and environment in the future then make the best decision for adapting actions. Unfortunately, forecasting the future is a harder problem. As S.A Seshia et al. argued "the mainstream machine learning techniques of today do not perform exact learning"[89]. To achieve dependable self-adaptation, causal analysis and reasoning methods are needed urgently.

As the physical effect is hardly eliminated, the potential inaccuracies of machine learning [89] are intolerable in critical situations. The correctness V&V of adaption policies is indispensable, but compared with self-adaptation strategies, few researches of the V&V on self-adapting were published and most of them just posed the emergent challenges of self-adaptation V&V [89-91]. Following the

directions in [90-91], S.A Seshia et al. proposed a concept of trustworthy machine learning [89]. Ivan T. et al. presented a blueprint process for V&V of decentralized, self-organizing cyber-physical production systems which includes planning, engineering, commissioning, pre-operation, operation, maintenance and optimization phases [92]. In the context of theory, Zoltán T. et al. discussed the problem of verification based on factored Markov decision processes in a factored event-learning framework [93]. Hence, we can make a conclusion based on those approaches that the contract based or model based formal V&V is a promising method to guarantee the correctness of machine learning based on self-adapting.

To non-machine-learning based adaption, Marcello M.B et al. *formalized the semantics of the adaptation logic* of the component called OLIVE with constraint LTL over clocks (CLTLoc) and verified the properties with SMT-solver. The results show that the upper bound of the temporal cost of the correctness check. Further, the authors point out the need of ad hoc approaches to perform on-line verification and discourage the use of general formalisms [94]. Marwa H et al. first built a high level abstract model with event-B method and verified the correctness of self-adaptive systems with the transformation rules of pattern entity, tagged value and connections [95]. Paolo A et al. provided a formal rule based validation method for service-oriented applications. The authors validated the behavior by scenario-based simulation [96]. Youngil K et al. introduced a vehicle–driver closed-loop simulation method to verify the self-adaptation algorithm for integrated speed and steering control with the CarSim and MATLAB/Simulink [97]. Therefore, simulation based validation is also an alternative method.

Overall, self-adaptation is the technological trend of CPS, various measures of self-adaptivity are widely researched in different domains, and even the environmental factors have been considered. However, the V&V of self-adaptation is just at the beginning. Despite the complexity of self-adaptation, modeling the effects of environment and the behaviors of human beings are the two obstacles that must be overcome because they are the two most challenging parts.

4.2 V&V on real-time schedulability

Real-time is a traditional property of the critical application. The real-time scheduling methods have been widely researched for a long time [98]. The scheduling in CPS is also the common distributed real-time scheduling issues. Current researches focus on the scheduling with multi-constraints. E.g. *energy* and *temporal* constraints [99], *fault tolerance* and *temporal* constraints [68], *security* and *temporal* constraints [100], system and environment *states* and *priority* of tasks [101], etc. Furthermore, a recommended method of multithreading multi-agent scheduling was proposed to search the efficient and fast solution of complex problems in real-time featuring rapid dynamic changes and uncertainty [98]. A period and deadline selection method was presented to improve control performance [102]. A hybrid discrete particle swarm optimization-genetic algorithm was proposed for multi-task scheduling problem in service oriented manufacturing systems [103].

Similarly, the problem of V&V on real-time scheduling has also been widely studied. E.g. worst-case execution time (WCET) analysis [104], perceptions scheduling V&V [79]. A general model of mixed-criticality recurrent real-time tasks was proposed, and the analysis model took WCET, relative deadline, and period into account at two criticality levels [105]. A flexible model-based schedulability analysis approach is introduced for hierarchical CPS, which can analysis the real-time attributes with stochastic information through an extensive framework [106]. Qian Z. et al. proposed a time-constrained aspect-oriented Petri Net for CPS modeling. Then a general scheduling analysis method is presented by decomposing the model into four modules. Last, the schedulability of four modules is analyzed separately [107]. Divisible schedulability method is very useful to large scale system. Furthermore, a customized formal model is proposed to analysis the schedulability for event based concurrent reactive systems. The authors first formally defined an event-based scheduling policy and proposed a decentralized scheduling method, this method can automatically decompose the scheduling policies into atomic scheduling policies and analysis those policies independently [108].

One key constraint to scheduling arithmetic is that the arithmetic should have low temporal and space complexity. Whereas fast arithmetic generally is simple, but not smart enough to deal with the

problem having massive constraints. Unfortunately, as mentioned in section 4.1, the self-adaptive CPS is applied in the uncertain environment and suffers various constrains. Centralized scheduling itself will be the bottleneck of performance and accuracy/correctness. *Hierarchical scheduling* and *decentralized scheduling* will be the trend. To V&V, as CPS always consists of large amounts of subsystems, *decomposable methods* can highly improve the performance of schedulability analysis, especially for hierarchical scheduling and decentralized scheduling. Moreover, analyzing the schedulability with abundant constraints will be meaningful for arithmetic optimization and validation.

4.3 V&V on timing behavior

In order to be smart, we have to integrate the learning and reasoning methods into CPS. The premise for learning and reasoning is reproducing the relationship of physical events in cyber space. Guaranteeing the timing behavior is a necessary condition for building an intelligent CPS. Some papers have raised the challenge of timing assurance [20, 45, 109]. Yet considering the deviation between crystal oscillators in distributed nodes, the stochastic jitter of signals, the unpredictable delay of communication, the value of inputs and the branch selected at run-time. It will be very difficult to achieve the global reference time [60] in distributed CPS. In addition seamless integration and interaction between cyber and physical space demand to deliver information of sensing, processing, and control in correct order and in real-time. As timing has never been treated as a basic premise ago [109], it needs to reconsider the role of timing to CPS. Researches already try to redesign the whole system from hardware platforms [110] to operating system [111], from programming models [112] to architectures [25], from time synchronization protocols [26] to communication standards [113-115].

Timing is a sensitive runtime constraint. To verify and validate the timing behavior, we need to take all factors at different levels into account and build a comprehensive model, which includes the sub-model of hardware behaviors, software behaviors, network behaviors and environment actuations. To our best of knowledge, there are no such comprehensive models and modeling tools now. All approaches on V&V just take partial factors into account. E.g. Cardoso J. et al. examined the requirements of critical industrial application, proposed the temporal semantics for network communication, evaluated the effects of network limitations on time-critical systems with model based design and simulation [116]. A programming model called PTIDES is presented for distributed real-time embedded systems design. It extends the programming abstractions with temporal semantics [117]. In the context of model of computation (**MoC**), another two further works on PTIDES is discussed in [118-119]. The key idea of PTIDES is to cover the oscillations of delay, it sets a strict trigger condition and timestamp for event delivering and processing. In some sense, to communication, timing assurance is a stability issue of network transmission delay. M.C.F. Donkers et al. presented a modelling framework for discrete-time linear switched and parameter-varying NCSs (network connected systems), and the factors of time-varying transmission intervals, time-varying transmission delays, packet dropouts and communication constraints are considered in this framework. The authors analyzed the stability of such NCSs with a stochastic computational technique [120]. A methodology is presented to optimize timing behavior with a multi-objective evolutionary optimization algorithm called NSGA II. The authors modeled the timing interfaces of components with a timed-automata and verified the model with the UPPAAL based on computation tree logic [121].

Timing is an important functional requirement, which should be treated seriously in CPS. Considering the current process on timing research, the design of timing assurance infrastructures should be accelerated to offer better supports for the large scale CPS. The phenomenon and the effect of time-varying should be researched at all levels of CPS. As the jitter of timing is random, the statistical probabilities of jitter must be explored for simulation based V&V. Considering the significant amount of work, researchers and organizations should cooperate together to develop the V&V toolset as soon as possible.

4.4 Observability and Traceability

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

As CPS is too complex to be tested completely, improving the customers' confidence of system becomes a big challenge. It is meaningful to helping users notify the signs of abnormal behaviors and take preventive actions to reduce losses. It is valuable to save the evidences and the scenarios of failures for the maintainer. We need additional assistance measures to guarantee and to check the correctness of the system in run-time. This issue calls for high observability solution for CPS. Moreover, enhancing the traceability of CPS can improve efficiency and productivity, especially the tracing of failures and changings.

Modern systems are packaged so well that like a blackbox. It becomes more and more difficult to understand the behavior of the system and backtrack the history statuses based on limited detectors. Observability has been systemically investigated in the control system area for a long time. To CPS, observability researches mainly focus on the optimization and formal analysis. David Lawrence et al. first introduced observability into CPS as a part of the verification process to ensure reliability. To select the observation points efficiently, the authors built feedback model, recovery model, compensation model and degradation model with extended finite state automata [122]. Dimitri L. presented a formal method with the partially observed petri nets for discrete event systems and defined formal observation sequence. By computing the degree of confidence of past and future states and events, the authors estimated diagnosability, detectability, and predictability [123].

Though big data implicates lots of information, mining the potential value of trivial data is difficult, especially mining the comprehensive information involving multidisciplinary and multidimensional concept. To overcome this drawback, we need to improve the traceability of CPS to trace the changes of products, documents, requirements, specifications, bugs, system updating and subcomponents [124] etc. Most researches on traceability are belong to the MDE (Model Driven Engineering) domain. Kamal S. et al. introduced the definitions and techniques of traceability [124]. Stefan W. et al. investigated the traceability in requirements engineering and model-driven development [125]. To determine if a given source code change impacts the design of the system, Maen H. et al. presented an automatic approach that allows code-to-design traceability to be consistently maintained the evolvement of source code [126]. A framework called iTrace was proposed for the management and analysis of traceability information of different models in MDE projects [127]. A general theoretical framework called TF4SM was proposed for developing traceability solutions in small manufacturing companies [128].

Observability and traceability are a couple human-centered characteristics, which are useful to human-in-loop CPS. Observability improves the correctness in fine grain statuses, while traceability simplifies the management at a systemic level. Yet the increasing complexity decreases the observability of the system. It becomes more and more difficult to understand the current behavior of a system with limited detector. Higher level detectors are needed to show the comprehensive statuses of subsystems. To achieve this goal, on one hand, we need to export the invariable systemic properties; on the other hand, the detectors should be smart enough to make comprehensive conclusions but show the result visually. CPS are multidisciplinary and cross-domain applications, to easy the communication between the staffs with different backgrounds and to simply the management, the whole life-cycle MDE based traceability solution is urgently needed.

4.5 Brief summary

Correctness is the first citizen requirement for all systems during all periods. In this section, we focus on the rising characteristics in CPS and investigate the current solutions and V&V approaches on self-adaptability, real-time scheduling, timing behavior assurance, observability and traceability. Considering the changeable environment, it is impossible to validate the correctness of system in all the cases. As multi-objective decision is an NP problem, CPS has to apply the decisions with low. To guarantee the correctness of a CPS, traditional testing and validation measures are still useful, especially for subsystem design and software development and testing. As Lee stated that "a CPS theory is all about models" [129], MDE approaches on ***formal V&V*** and ***simulation based V&V*** will be the mainstream to guarantee the correctness. In addition, as a kind of SoS, improving the ***C&C abilities*** of subsystem can decrease the complexity of CPS remarkably, simply the V&V evidently

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

and enhance the self-adaptation fundamentally. In addition, ***timing behavior assurance*** is a new fundamental challenge to CPS architecture design.

5. Architecting dependable self-managing CPS

Considering the complexity of CPS, starting with a good design and developing with an efficient tool is a wise choice. Good design can not only decrease the cost and risk of system development, but also can improve the flexibility and ease the management (configuration, operation and maintenance) of the system. In this section, we mainly focus on the current approaches on architectures/frameworks of dependable self-managing CPS and the toolsets for CPS development. Finally, we make a short conclusion of the qualities of good architectures/frameworks. Here, we use the term "architecture" to represent "architecture or framework" in short.

Theoretically, the architecture defines the upper bound of dependability. To design a good architecture, we should firstly analyze the requirements of dependable self-managing CPS. Despite the requirements concluded in Figure 5, we should refine the requirements of dependability and self-management. E.U. WARRIACH et al. [130] defined the "*self-** properties" for self-management system as a set of abilities which are *self-healing*, *self-protection*, *self-awareness*, *self-organizing*, *self-synchronization* and *self-configuration*, and detailed both functional and nonfunctional requirements of self-healing in smart environment applications. The authors also indicated the abilities of dependable framework, which include monitoring, analysis, fault-detection, fault-prevention, fault-tolerance, fault-healing, and fault-notification.

As discussed in section 2.1, the popular architectures are SOA based architecture, MAS, and other aspect oriented architectures. Here, we analyze 97 approaches with self-* ability among more than 300 papers. We select 20 (in 26 papers) real CPS architectures (not concept) and list the key methods on C&D in those approaches in Table 2.

Table 2. Reaches on CPS architectures

Ref	Arch	Key Methods for correctness & dependability[1]
[14]	SOA based	Decouple, composite service; *Heartbeat, real-time FDIR, Fault tolerate based on middleware;*
[15]	SOA based	Unified Abstraction, Domain-Specific Description Schemas, **Formal** Semantics;
[16]	SOA based	**Knowledge**-Driven Service Orchestration, Ontology based service description;
[131]	SOA based	**Formal** Contract for physical property, dynamic physical behavior, hybrid system behavior;
[132]	SOA based (ebbits)	Proxy, **Virtualization**, **Middleware**, Ontology, **Semantic** **Knowledge**, **Rule** base context recognition; *Predictive maintenance*
[19]	MAS based	Self-Organizing & Self-Adaptive Models, **Rules** & **Knowledge** based Reasoning, proof-of-concept; *Exception Identification Model;*
[37]	MSA + cloud	Data-driven self-organization, intelligent negotiation based on contract net protocol, Deadlock prevention;
[133]	MSA + holons	Soft real-time MSA, Hard real-time function blocks (holons); ***Redundancy;***
[134]	Cloud based	**Virtualization**, multilevel smart **scheduling** algorithms; ***Redundancy***, *checkpoints;*
[135]	Cloud based	Distribution **middleware**, **Virtualized** interrupt model, Spatial & temporal isolation based on partitioning; *Fault isolation;*
[136]	Cloud based	**Virtualization**, Task migration, Evolutionary algorithm for placement, WCET response time guarantee;

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

[58]	Software defined	Network-centered (**SDN**), Technology **Standardization**;
[20,137]	5C Arch	Decouple,5C, **knowledge** based; *Prognostics and health management, Fault isolation & identification;*
[18,138,139]	Rainbow	Architecture-based self-adaptation (**ABSA**), (Re)**scheduling**, Strategy based, Mutation rules robustness tests;
[21,27,140,141]	DEECo	DSL, Decouple, IRM, Knowledge, Deterministic **semantics**, **Formal** analysis; *Proactive reasoning, Reliable communication*;
[83]	Na[2]	**Standardization**, Open-Knowledge-Driven, ontology;
[142]	Na	*8 steps comprehensive FDIR, Knowledge, Reasoning;*
[143]	Federation Arch	Component-based, Plug-in software, Plug-in runtime environment based on VM, Federation life-cycle management;
[144]	Na	*Fault mode, Reconfiguration, Rule based diagnosis, Reasoning;*
[145]	EVM[3]	**EVM**, Virtual Component, EVM DSL[4], **Formal** design, multi-level & multi-object **scheduling**

[1]The dependability methods are shown in italic; [2]Na: no explicit introduction is provided; [3]EVM: Embedded Virtual Machine. [4]DSL: Domain Specific Language.

Decoupling and abstraction are two common methods to tame the complexity in CPS. **Scheduling/migrating** tasks among the decoupled components/services/agents are the main technology to achieve the flexibility. The methods for self-adaptation that applied in these architectures are smart scheduling algorithms based on *rules*, *knowledge*, *strategies* and *evolvable design*. **Standardization** plays an important role in CPS services/subsystems orchestration, especially the orchestration of heterogeneous subsystems. Most of these architectures focus on the functional requirements of application and flexibility (self-adaptation). **Formal model** and **formal/mathematical analysis** are the main approaches to guarantee the correctness formally.

Traditional methods are still the mainstream to achieve the dependability. Thanks to the abound subsystems in CPS (especially the cloud based CPS and MAS based CPS), **subsystem** (agent, component) **level redundancy** is a popular solution to improve the availability of the system. To cloud based CPS, *VM based* resources isolation can simply the **fault isolation** approaches a lot. *Rule* or *knowledge* based **fault diagnosis** and **reasoning** are one solution that may mostly fit the flexibility requirement of CPS. As CPS is a holistic and organic system, **comprehensive FDIR** methods should be applied at all levels and all subsystems of CPS to avoid the bottleneck of dependability. **Middleware** also is an alternative solution to apply dependability measures on heterogeneous subsystems and to provide consistent behavior.

Architecture-based self-adaptation (***ABSA*)** is an advance concept to achieve high flexibility. It can significantly expand the potential of adaptation, which is useful for overcoming the challenges from **RQ1** to **RQ5**. The ***multi-object scheduling*** algorithms that take the dependability into account is a key solution to overcome the challenge of **RQ3**, such kinds of algorithms should consider the multi-dimension and even multi-domain constraints and cooperate between cross-layers. Formal methods are a way to quantify the dependability and semantic methods are a good choice to check the fineness between architectures/decisions and contexts. Therefore, modeling with ***quantifiable formal semantic*** is a potential solution to conquer the challenge of **RQ1**. ***Standardization*** is the common solution to minimize the differences of behavior of heterogeneous subsystems, and it will contribute to solving the **RQ2** and **RQ4**. ***Middleware*** is a strong competitor to achieve ABSA and Standardization. ***Rules, knowledge, strategies based methods*** **(**scheduling or decision**)** may improve the accuracy of context adaptation, but the real performance depends on the achievement of **RQ4**. ***Virtualization*** is a great solution to prevent the propagation of errors and achieve fault isolation. However, VM based isolation is still to heave to sensors and actuators, and maybe container based isolation is a better choice. ***Redundancy*** can improve the upper bound of dependability, but it should be applied together with smart scheduling and decision making methods in self-managing CPS.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

Most of these architectures are networked, but only a few of them pay special attention on network organization and network management. *SDN* may be a good solution to improve the flexibility and dependability [146] of large scale CPS. **Real-time assurance** at architecture level is mentioned a little in most proposals, especially the instability delay in the network communication. The proposal in [136] introduced a kind WCET response time guaranteeing mechanism to manage the volatile delay from resource armament and placement. To our best knowledge, architecture level **timing** related method is still a concept. Precise time communication and time synchronization should be considered at a systemic level. The effects of fault recovery should be researched, especially the effects on real-time and timing behavior.

6. Dependable self-managing CPS engineering

Various dependability technologies are integrated for both CPS design and evaluation [147-149]. As the traditional methods for dependability has been widely surveyed, we just summarize these methods and discuss the shortages of them for dependable CPS engineering briefly, readers can learn further details from the reference surveys and books. In this section, we pay special attention on the approaches of dependable self-managing CPS in detail, which includes the technologies for dependable CPS development, the measures for dependability management in run-time, the dependability V&V methods and other related methodologies. A brief summary is given at the end of every topic. Considering the widely research on MDE, the MDE based dependability V&V methods will be analyzed in section 7.

6.1 The traditional dependability means and V&V

Generally speaking, correctness is the basic precondition for dependability management, and good dependability strategies can enhance the degree of correctness of systemic behavior and limit the loss. Dependability has been researched for a long time and is well surveyed. The core methodologies of these measures are temporal and spatial redundancy. Massimo T. et al. focused on the fault detection, isolation, and recovery (*FDIR*) and reviewed FDIR strategies in detail for the space systems engineering and summarized the experience and the lessons learned with a real case [150]. Z.W. Gao et al. concentrated on *fault diagnosis* and *fault-tolerate* and studied the real-time signal based techniques in control domain, which includes the time-domain, frequency-domain and time-frequency methods [151]. Focusing on hardware engineering, Sparsh M. et al. provided a comprehensive survey of *reliability* techniques for microarchitecture design (such as processor registers, cache and main memory, etc.) [152]. Thaha M. et al. surveyed the *fault detection algorithms* in WSN and performed a qualitative comparison of the latest fault detection researches including centralized, distributed and hybrid algorithms and analyzed the shortcomings, advantages of those algorithms. From the data-centric perspective, the authors took 7 types of faults into account which include offset fault, gain fault, stuck-at fault, out of bounds fault, spike faults, data loss faults and aggregation/fusion error [153]. Samira C. et al. reviewed the *fault tolerance methods* at different WSN levels and classified these methods into 4 categories: power management, flow management, data management, and coverage/connectivity [154]. More detail and systemic introduction to *dependability/reliability engineering* refer to the book [155-157]. It should be mentioned that all introduced dependability management belongs to the active fault handling measures. In some cases, "**let it crash**" is one best solution because fault management takes too much time and costs too much.

The traditional dependability V&V methods are surveyed in [155-162]. These methods include Fault Tree Analysis (*FTA*) e.g. Event tree analysis (*ETA*), Component Fault Tree (*CFT*), Software Fault Tree Analysis (*SFTA*), Fault Contribution Tree (*FCT*); Binary Decision Diagram (*BDD*), Availability block diagram (*ABD*), Reliability block diagrams (*RBD*), Reliability Graphs; Failure Modes and Effects Analysis (*FMEA*), Failure Modes Effects and Critical Analysis (*FMECA*), Software Common Cause Failure Analysis (*SCCFA*); Hazard and Operability Analysis (*HAZOP*), Functional Hazard Assessment (*FHA*), Risk Reduction Analysis (*RRA*), Cost-Benefit Analysis (*CBA*), Qualitative Criteria Analysis (*QCA*); *Markov chain* analysis**,** *Petri Net* analysis; and Boolean *SAT* and its generalization:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

Satisfiability Modulo Theories (*SMT*), Bounded Model Checking (*BMC*) other Higher-order-Logic based proving like *HOL4*, Isabelle/*HOL*; etc.

Another useful method to improve the dependability is the fault injection. As a measure for testing, it can be applied to improve the correctness. According to the ways that faults injected, fault injection can be classified as hardware-based (*HWFI*) [163], software-based (*SWFI*) [163,164], and simulation-based (*SMFI*) [163, 165]. It needs to develop the real target system first to apply the HWFI and SWFI. Generally, it is too late and costs too much to consolidate the dependability for complex system development. SMFI is integrated with MDE toolsets, but needs priori knowledge to model the behavior of the target system (both hardware and software), especially the model of faults and their effects. More detail introduction of fault injection is referred to [166].

As C&D are two comprehensive requirements, to reduce the cost and the risk, we should consider them through the whole period of development, verify and validate them as early as possible. In addition, this is also the key motivation of the famous V-model. The application of traditional methods for C&D in different periods of V-model can be roughly summarized in Figure 11. The '+' means that the methods in previous period can also be applied in the current period. I.e. the fault injection can also be applied at the stage of integration.

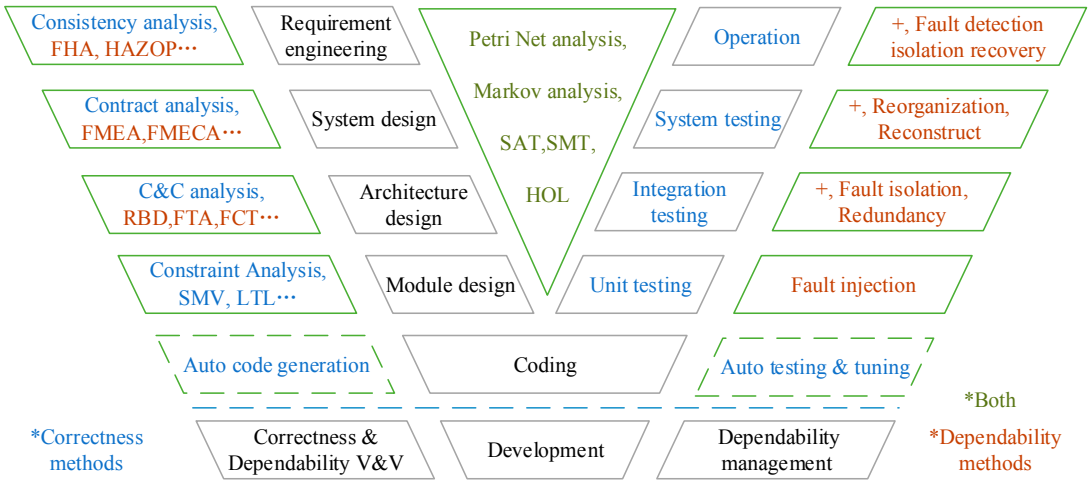

Figure 11. Correctness and Dependability engineering and V&V

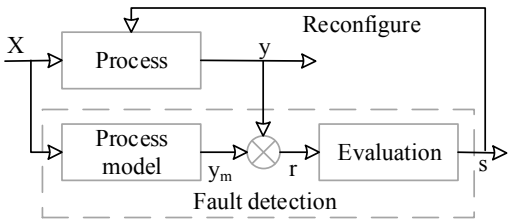

Figure 12. The traditional FDIR schema

The traditional methods are mainly appropriate for the general-purpose computers, which are well protected, or for the special applications that can tolerate high cost (like space application). However, these assumptions could no longer be held strictly in CPS. CPS is a large scale system and widely applied in various areas. It is cost-sensitive and exposed to an unpredictable environment. These methods are still useful for the basic subsystem designing, but become less and less efficient in protecting the CPS holistically. They ***just fit for the RQ1 challenge without context adaptation requirements*** because mechanical measures can hardly adapt to the changeable requirements smartly. Taking the FDIR as an example, the traditional schema is illustrated in Figure 12. Predefined process model is the precondition to apply the traditional FDIR. Notice that predefined model also implies the function and structure of the system are fixed. I.e. the dimension m and n of the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

watching matrix X_{m*n} of a control system generally is fixed in run-time. Concisely, it's impossible to build a sophisticated model of an arbitrary changed structure. Considering the self-driving car as another example, as the scenarios of transportation are infinite, the error source and the combination of faults are also infinite, testing or fault injection for all these cases is a mission impossible.

Meanwhile, the traditional V&V methods are also not powerful enough to solve the ***RQ2*** *challenge*. These methods are founded on the assumption of static structure and lack of dynamic semantics [167] to evaluate the varied structures and behaviors. I.e. the topology (the connection between the elements) of a fault tree or a block diagram is static. FTA and RBD can model the stable structure before or after the joining with two independent instances. However, FTA and RBD cannot analyze the process when a new subsystem is joining the CPS. The complexity is also another challenge to the V&V technologies, i.e. the state explosive problem to Markov, Petri Net based analysis. The unpredictable environment is another insurmountable obstacle to both traditional and the new V&V technologies. The unpredictable environment always means that we can't model the detail behaviors and test all possible cases.

Hence, smarter dependability management should be researched to adapt to the varied context and more dynamic V&V methods should be invented for guaranteeing the dependability of the dynamic structures and behaviors in CPS.

6.2 The run-time dependability management for self-healing CPS

There is no doubt about the important role that dependability plays in CPS application. Despite the traditional methods, more and more researches try to improve the flexibility of the dependability management methods in run-time. Run-time dependability management aims at the challenges of **RQ1** and **RQ5**. The survey on reliability aspect in section IV.B of the paper [35] can show us a preliminary view of the current researches on dependability of CPS. Most of these approaches focus on the adaptability of self-healing solutions or self-reconfiguration strategies.

In this section, we mainly focus on the modern methods of self-healing (self-adaptation has been analyzed in section 3). Following the classification of the means to achieve dependability, our investigation is organized into 3 parts: 1) *fault tolerance*, 2) *fault prediction (forecasting) & prevention*, 3) *maintenance*. Because fault remove approaches belong to the field of V&V or testing, we will analyze them in section 5.4.

6.2.1 Modern methods for fault tolerance

Fault tolerance is a key technology to improve the run-time dependability of systems. As there are numerous similar subsystems (i.e. sensor nodes) in CPS, spatial redundancy mechanism can be applied naturally at the subsystems level. Following the general process flow of fault tolerance, we organize the investigation into 3 parts: fault detection/diagnosis, fault isolation, fault recovery/ tolerance.

1. Modern fault detection/diagnosis

   Yang et al. [168] focused on the fault of context inconsistency and surveyed the improvement techniques and the challenges of deploying dependable pervasive computing systems from the fields of context management, fault detection, and uncertainty handling. The authors classified the detection methods into 1) statistical analysis-based *detection*, 2) *pattern matching-based detection*, 3) *coverage criteria-based detection*, 4) *simulation based detection*. Yet most surveyed papers just detect the inconsistency between the function context and the environment, but few of them detect the inconsistency between the dependability management context and the status of system and environmental, which are closer to the concept of correctness.

   From the academic view, knowledge-based fault diagnosis and fault-tolerate techniques are surveyed, which including *knowledge-based*, and *hybrid/active diagnosis*. Furthermore, the quantitative knowledge-based fault diagnosis can be further divided into 1) *statistical-analysis*

 doi:10.20944/preprints201707.0044.v1

data-driven fault diagnosis, 2) *nonstatistical-analysis data-driven fault diagnosis*, 3) *joint data-driven fault diagnosis*. [169] According to this survey, we can figure out that feedback (backpropagation to AI based algorithms) plays an important role in model training for fault diagnosis, and data is the key to achieve the highest quality diagnosis result (training).

Knowledge-based fault diagnosis generally needs huge resources (CPU and memory), which can be applied in the cloud system (DSS) but is not very suitable for sensors and actuators. What's more, data driven diagnose is good at recognizing the abnormal (unexpected) behaviors, but not the faults, which increases false alarms (False Positive rate).

To sensors and actuators, traditional fault detection/diagnosis methods are still the mainstreaming solutions, E.g. heartbeat methods, WCET based detection, and threshold/event/state based methods. Modern methods include: 1) *Model based fault detection* [170-171], yet the flexibility is limited. 2) *Probabilistic fault detection and isolation* [172], which may increase the rate of False Negative. 3) *Cooperative fault detection* [173], through exchanging (checking) information between neighbors. 4) *Distributed fault detection* [174], based on the statistical result of the neighbors' data. 5) *Machine learning based fault detection* [175-177], these methods are generally applied in the data center and based on the collected data.

In some sense, similar to WSN, the faults in CPS are local in the network. By taking advantages of the abound subsystems, WSN can tolerate partial failure of subsystems and loss of data [153]. Whereas CPS is location-aware and individual-sensitive, the data in each location is unique and may be irreplaceable to decision making, using the approximate value from neighbors will decrease the fitness of decision and increase the risk of failure. *Statistical based*, *time series based* and *multidimensional* data validation are needed to build a comprehensive view of the reliability (confidence) of data.

2. Modern fault isolation

As complementary to the traditional fault isolation methods, virtualization becomes a popular technology to isolate faults at the subsystems level, because it can isolate the resource and runtime environment for applications, which can prevent the fault propagation naturally. As we all known that redundancy based on Virtual Machines (VM) is the main way to achieve high availability in cloud system (DSS), and the sensors and actuators are naturally isolated by separated hardware. A detail survey on middleware based isolation for sensor and network is reviewed [178]. To CPS, virtualization can also be applied to improve the isolation of networks. A virtual platform called Xebra is proposed to provide isolated communication for CPS [179]. Both node-level and network-level virtualization are surveyed for WSN, the authors classify the network virtualization into 1) *Virtual Network/Overlay-based Solutions*, 2) *Cluster-based solutions and* 3) *Hybrid Solution based Middleware or VM.* [180]

3. Fault recovery/tolerance

As there are numerous subsystems, redundancy at node level is a natural solution to achieve fault tolerance in systematic CPS. Thanks to the virtualization technology, an advanced 2f+1 replication solution is designed to achieve the Byzantine fault-tolerate [181]. The effects (robustness) of network topology (redundancy and structure) on connection (recovery) are studied through the simulation of six kinds of topology [182]. Manuel R. et al. explored the role that diversity plays in defending the faults and attacks through qualitative analysis, the result shows that diversity can tolerate the common-mode failures but will introduce undetected

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

failures [183]. To tradeoff between the cost and reliability, a formal specification of spatial and temporal redundancy based reliability is proposed; a framework is designed to determine the maximum reliability with a recursive formulation [184].

To design a fault tolerance CPS, spatial or temporal redundancy is the main solution. As all the redundant components suffer the same environment, redundancy with isomorphic components cannot tolerate such failures. Diversity design with heterogeneous components may be an alternative solution because it can offer cross-validation. Notice that **flexible, accurate and high efficient fault detection/diagnosis methods will be the key to achieve dependable CPS**. In addition another important factor which should be taken into account is the **time cost** of fault diagnosis and recovery, especially to the time critical CPS.

6.2.2 Modern methods for fault prediction and prevention

Fault prediction & prevention predict the faults based on symptoms or abnormal behaviors or thresholds, then prevent the failure with proactive measures (i.e. replace the risky component) or replace with safe actions or reject (freeze) the dangerous operation (i.e. reject to start an unsafe car). A fault prevention framework is proposed, which consists of monitoring, fault-prediction, and adaptation mechanisms [185]. To prevent selecting the unhealthy node as cluster heads, a fault-prevention clustering protocol is designed, which includes failure prediction, cost evaluation, and clustering optimization [186].

As fault prediction is the precondition to prevent failure, most researches focus on this field. To evaluate the approaches of the battery power depletion prediction with received signal strength indicator (RSSI), six general models are tested, and the result shows that the AR model has the best performance [187]. The performance comparison of two fault prediction algorithms (multiple linear regression, and artificial neural networks) on the remaining lifetime of battery-powered WSN node is shown in [188], the results show positive improvement in power control and reliability and availability. Based on **AZOP** and **FMEA** analysis, a **simulation based** multi-agent fault prediction and diagnostic system is described in [189]. Further, the authors built a process-specific **ontology** to describe the concepts of process, diagnostic **ontology** to express the **knowledge** from human expertise and related operations. A prototype prediction based intelligent diagnostic system is introduced in [190]. The prototype system integrates qualitative and quantitative process models, operational experience with a real-time G2 **expert system**. Furthermore, the system extracts the "cause-effect'' rules and suggestions from the standard **HAZOP** analysis. The detail classification of online failure prediction methods is shown as Figure 13 [191].

The result of fault prediction is generally expressed with a probability value whose domain is $[0,1]$, while the execution of prevention and recovery operation is binary $\{0,1\}$ (do or do not). Mapping from $[0,1]$ to $\{0,1\}$ is incautious and risky. Kush R.V. et al. introduced a set of measures to guarantee the safety of machine learning, including inherently safe design, safety reserves, safe fail, and procedural safeguards [87]. Proactive prevention may take unnecessary, even unsuitable, operations which is not only wastes resources and increases the delay, but also may introduce new errors to system, *model based* or *knowledge based failure prediction & prevention* may relax this issue by causal inference. Furthermore, considering the real-time constraint and the complex environment and human activity effects, current failure prediction methods are still not powerful enough. *Long-term* and *short-term failure prediction* should be selectively implemented at different levels of CPS according to the accessible data set size. *Prophetic failure prediction* should be researched urgently for real-time, safety-critical CPS.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

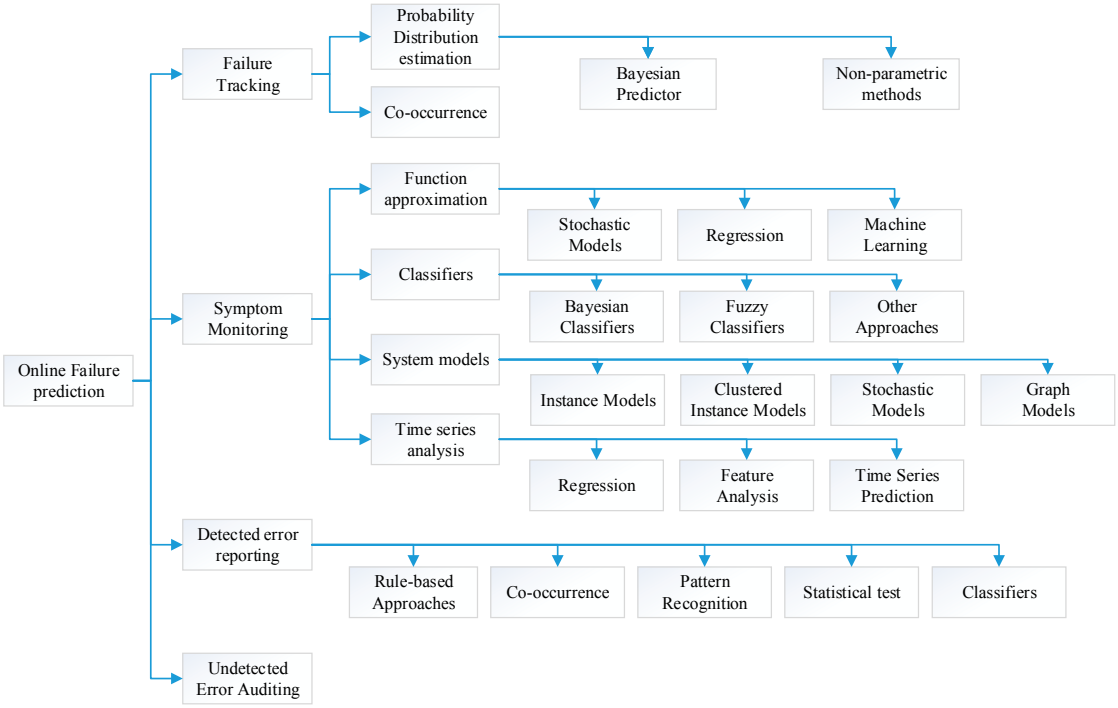

Figure 13. A taxonomy for online failure prediction approaches

6.2.3 Modern methods for maintenance

With the increasing complexity of CPS, locating the failed nodes among the millions of subsystems is a big challenge. It should be careful to recover or replace failed nodes to prevent to introduce new errors. Consequently, maintenance is a key technology to prevent the decay of CPS. Santiago R.Z et al. first reviewed the existing maintenance principles for linear complex systems (LCS), then figured out the non-applicable, exportable and adaptation required principles for the CPS by analyzing the similarities and differences between LCS and CPS [192]. According to the concept of maintenance 4.0 in [193], modern maintenance are main about self-healing, especially failure prediction. As most of these methods have been reviewed above, to avoid unnecessary duplication, the maintenance here represents recovering the system from the failures that belong the ability of self-healing, i.e. physically replacing a broken engine.

As an inference of the halting problem, it is impossible to develop a self-healing system, which can diagnose all faults and recovery from all failures automatically, so the dependable self-managing CPS still need some external assistances (maintenance). Two main directions to easy the maintenance are summarized. One is *simplifying failure identification and location*. Improving the observability and traceability of CPS is one way to achieve this goal. Another way is developing assistance program. A *condition* and *time* based maintenance model and framework is proposed and *domain knowledge* based decision making for maintenance is highlighted [194]. The Bayesian networks based assistance programs for dependability, risk analysis and maintenance is reviewed [195]. Another is *simplifying repairing/replacing*. The basic solution is designing the subsystem with high C&C so we can just replace the old failure subsystem without any additional adjustment. The advance solution is developing applications for the decision making for repairing. A linear programming-based model is developed to optimize the repair priorities and a stochastic model is introduced to examine these reparation policies [196].

6.2.4 Brief summary

To improve the run-time dependability of CPS, comprehensive fault management methods should be integrated at different levels carefully and the side effects introduced by these methods should be taken into account during the evaluation. According to the previous investigations above, some common trends on runtime methods for dependable self-managing CPS are: **1)** straightening

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

the reliability of infrastructures (hardware, software and network) using the traditional methods (for **RQ1**); **2)** improving the availability of sensor (network) and (network) with subsystem redundancy and reconstruction (for **RQ1**); **3)** integrating the fault detection/diagnose methods (i.e. FDIR) on sensor nodes and actuator nodes to block the propagation of errors and to limit the effect of failures (for **RQ1**); **4)** enhancing the fault isolation at subsystem/application level (i.e. virtualization technology) (for **RQ1**); **5)** using group intelligence (i.e. DFD, cooperative detection) to check each other automatically (for **RQ1 & RQ2**); **6)** applying data-driven methods (i.e. machine learning) on DSS to predict faults (for **RQ1 & RQ5**); **7)** building knowledge based methods (i.e. ontology) or exporter system on all subsystems (both DSS and end-devices) to evaluate the prediction and then taking reasonable actions based on the evaluation, and also knowledge based methods or exporter system can be applied to predict the faults if no big data available (for **RQ1 & RQ5**).

Fault tolerance, fault prediction & prevention methods can improve the dependability of the system, but also introduce new errors because they will increase the complexity. The risk of adding complexity (self-healing) to handle complexity (normal function) is formally discussed in [197]. To improve the performance of fault management at low cost, some advices are provided: 1) **keeping the fault management measures simple and verifiable**; 2) **taking the side effect of fault management into account** (e.g. time delay, resource budget etc.); 3) if recovery costs too much then system can just **notify the stakeholders** (other subsystems and managers) and **let it crash safely**. The issuers should be reminded: 1) the failure caused by *wrong timing behavior* and the prevention methods should be taken seriously. 2) *Environment effects* are not considered in current researches. As CPS is tightly integrated with physical processes, environment can affect the system directly which may totally change the distribution and rate of faults. 3) *Human effects* are also not evaluated. With the widely applying of CPS in daily life, human-in-loop CPS will be the trend and the unpredictable and controllable human behavior will be the new source of errors.

6.3 The formal V&V for dependable self-managing CPS

Considering the side effect (complexity) introduced by self-management, compromise is necessary. To tackle the challenge of **RQ3**, we need to evaluate the design of dependable self-managing CPS. In this section, we just focus on the available formal V&V methods for dependability, and MDE based methods will be detailed in section 6. To overcome the drawback of traditional V&V methods, one way is enhancing the traditional methods with dynamic and temporal semantics. Another is developing dedicated models with mathematical methods or with (new) high level dynamic formal language (HLFL), and the other solution is simulation based V&V.

As a well-established and well-understood technique, FTA has various extensions for analyzing dynamic behaviors such as Dynamic Fault Tree (DFT) [159-160] for capturing the sequence dependent dynamic behavior with functional dependency gate and SPARE gate; Temporal Fault Trees (TFT) [159-160] for modeling temporal requirements with temporal logic and Pandora TFT [160, 198]; Component fault trees (CFT) [159-160] for managing the complexity of modern systems by hierarchical decomposition of systems, and Hybrid Fault Tree Automaton [199]; Repairable Fault Trees [159] for analyzing the effect of repair policies, State event fault trees [160], etc. Petri net is another popular formal method for CPS V&V. The Definition and an object-oriented Petri nets formal modeling approach is proposed for High-Confidence CPS [200]. A GSPN model is built to analyze the security and dependability of digital control systems in nuclear power plants [201]. A multi-level directed acyclic graphs, interdependent coupled networks.

To quantify the impact of unavailability of SCADA system, a mathematical method which combined with mean failure cost (MFC) metric and the classic availability formulation is proposed [202]. A new set of solutions based on undirected graph is proposed to verify the topology verification and locate the failure in Software Defined Networks (SDN) [203]. A dedicated symbolic model based on a combination of existing notions of robustness is introduced designing robust CPS [204]. To overcome the incomparable of the dual conjunctive scheme, a method is developed for verifying N-inference diagnosability [205]. To evaluate the ability of withstanding and recovering from irresistible catastrophic, Alexander A. G introduced the concept of resilience and present two classes of formal

models, which are multi-level directed acyclic graphs and interdependent coupled networks [206]. To formalize and assess the reconfigurable CPS, Linas L et al. formally defined the dynamic system architecture in Event-B, and verified the correctness based on derived model. To guarantees the resilience of data processing, the authors transformed the Event-B model to the statistical UPPAAL and evaluated the likelihood of resilience under different system parameters with a statistical model checking [207]. To analyze the unpredictable and random components at times at different levels of abstraction, higher-order logic (HOL) is used to model cyber-physical transportation systems formally. Then the authors obtained a randomized model by adding appropriate random variables and the probability or expectation of the system based on formally reasoning [208]. Taking TTA, FlexRay, and TTCAN as examples, I. Saha et al. analyzed the three startup algorithms of time-triggered architectures and presented specified formal models of faults for these architectures, then proposed different verification approaches for the three startup algorithms [209].

Generally, simulation based analysis is integrated with these extended FTA to overcome the complexity of the model. The analysis methods include Algebraic, Markov Chain, Petri Nets, Bayesian network, Monte Carlo simulation [159]. To analyze the impacts of the cyber layer of composite power systems; a multi-state Markov chain model is built; non-sequential Monte-Carlo simulation is applied to evaluate the reliability and availability [210]. More simulation based V&V will be introduced in section 6.

6.4 The related toolsets for CPS design and V&V

There are numerous existing toolsets for embedded system or control system development, yet these toolsets are fragmented. Most of them are just professional at developing the subsystems of CPS. Some of them are aspect specified, e.g. LabView for embedded instrument engineering, SCADE for embedded control software development, Simulink for multi-domain dynamic systems design, etc. Some are good at network design and simulation, e.g. Opennet, OMNET++, NS2, and NS3 etc. Some are domain specified, e.g. Ardupilot for airplane engineering, AutoSar for vehicle engineering etc. Though we have many choices for developing every subsystem of CPS, it is still unable to form a holistic toolset for CPS development. The first reason is that developing CPS are beyond the original ability of these tools. The second reason is the incompatible standardization (i.e. file type) and the inconsistent description (i.e. data and message type), which barricaded the cooperation between these tools. The third is the closed implementation, which limits the integration with other tools.

To build a holistic toolset for CPS, abundant researches have been done. From the perspective of engineering, Robert H et al. [211] reviewed the existing tools and standards, component- and model-based design flow, component programming and deployment practices, SOA engineering methods and CPS, interoperability and information models, simulation and modeling for CPS. In addition, the authors explained the supporting of a toolset to the integration of the virtual and physical engineering with an industrial case study. The example case uses an engineering software environment called vueOne, which is a toolset for Ford's virtual engineering in Powertrain assembly. Based on the concept of model driven architecture, a platform-based design environment called METROII is proposed [212]. It integrates FSM and KPN, provides mapping between SystemC and METROII, uses execution semantics in three phases to realize the model of computation (MoC). Another tool based on SystemC is introduced as a virtual platform for MDD based dependable CPS software design; the tool is prototypically integrated with Simulink as a part of AUTOSAR toolset [213]. To integrate WSN to CPS, a simulation toolset for cyber-physical automation systems is proposed. This toolset contains GILOO (the appended software module), COntiki OS JAva Simulator (COOJA), a cross-level wireless sensor network simulator, and the LabVIEW system design software [214]. To minimize or avoid future consequences and disruptions on the cyber physical production systems (CPPS), a prototype knowledge-based tool is designed for digital factory. Furthermore, the tool has been evaluated with both industrial and scientific stakeholders, the result shows it is useful to help the stakeholders aware of their decision consequences in CPPS design activities [215]. Another toolset called COMPASS is developed for safety-critical systems. It is a component-based modelling and system-software co-engineering tool which is integrated with the core language called SLIM, the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

new symbolic model verifier (NuSMV), a probabilistic model checker called MRMC, the requirements analysis tool (RAT) [216]. Compared with other modeling languages, SLIM, an extension of the Architecture Analysis and Design Language (AADL), has a formal grammar, which makes it possible to model and analysis the normal and error behavior formally with COMPASS. A formal toolset is designed for the formal design and safety analysis of wheel brake system. It includes a contract-based design tool for architecture modeling, a model-based safety analysis platform for modeling the behavior at the leaves of architecture, and a model checker for verifying the model [217]. Lots of researches work on this topic, due to the limited space, we will not introduce them one by one.

According the analysis of current researches on CPS toolset development, two trends can be found: 1) ***MDD/MDE based toolsets*** become the mainstream, and more detail survey on MDE will be analyzed in section 6. 2) ***Integrating with several mature tools*** in different domains (at different levels) are the main choice. Moreover, this choice also can be classified into two major types. One is developing *new coordinator/modular* to extend or combine current tools. But it is difficult to keep consistent semantics and behaviors between different tools, and the efficiency is low. Another is designing a *new modeling language* and then transforming the unified model as an input of other mature tools. The unified reference model always contains the richest information. However, some detail has to be abandoned during the transforming because current tools can just describe part of the unified reference model. Another challenge is the timing behavior, which is mentioned in section 3.3. Yet, few of current tools have taken the timing into account. Maybe a completely new tool should be developed or a set of current tools should be redesigned to ensure the timing behavior.

6.5 Brief summary

As self-managing CPS is integrations of various technologies, many measures should be applied during the whole life cycle of engineering to holistically guarantee the dependability. Diverse measures are recommended to provide cross-validation, e.g. cross validate with model based diagnosis and data-driven diagnosis, or even knowledge based diagnosis. Further, these measures should be integrated organically to avoid the inconsistent in fault prediction and diagnose at different levels, and to minimize the unexpected side effects, which needs great wisdom to achieve the balance. MDE shows us a promising way to evaluate the dependability management measures in the early development period.

7. MDE based V&V for dependable self-managing CPS

Though there are still some problems in project management, MDE has shown remarkable advantages in developing complex software for industrial applications [218]. From perspective of IoTPS (Internet of Things People and Services), the role of MDD plays in IoT development is discussed and the quantified benefit is evaluated [132, 219]. To tame the complexity and to easy the communication between teams from different background, we believe that MDE will be the mainstream to develop the CPS. The generic life cycle of MDE based CPS development is illustrated in Figure 14. To avoid going into infinite detail of molding and tool design, we just focus on the technologies of MDE based V&V on this survey. MDE tries to solve Q1 to Q5 in design period. One similar concept called model-based dependability assessment (MBDA) is introduced in [160].

Hundreds of modeling and simulation tools have been designed in both control area and embedded system area. These tools or languages can be classified into five classes: 1) *hardware modeling*, 2) *architecture modeling/system-level modeling/software modeling*, 3) n*etwork modeling and simulation*, 4) *Logical theorem based provers*, 5) *generic modeling & simulation*. Based on the surveys [52,220-223] and our knowledge, part of popular tools/languages are exhibited in Table 3.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

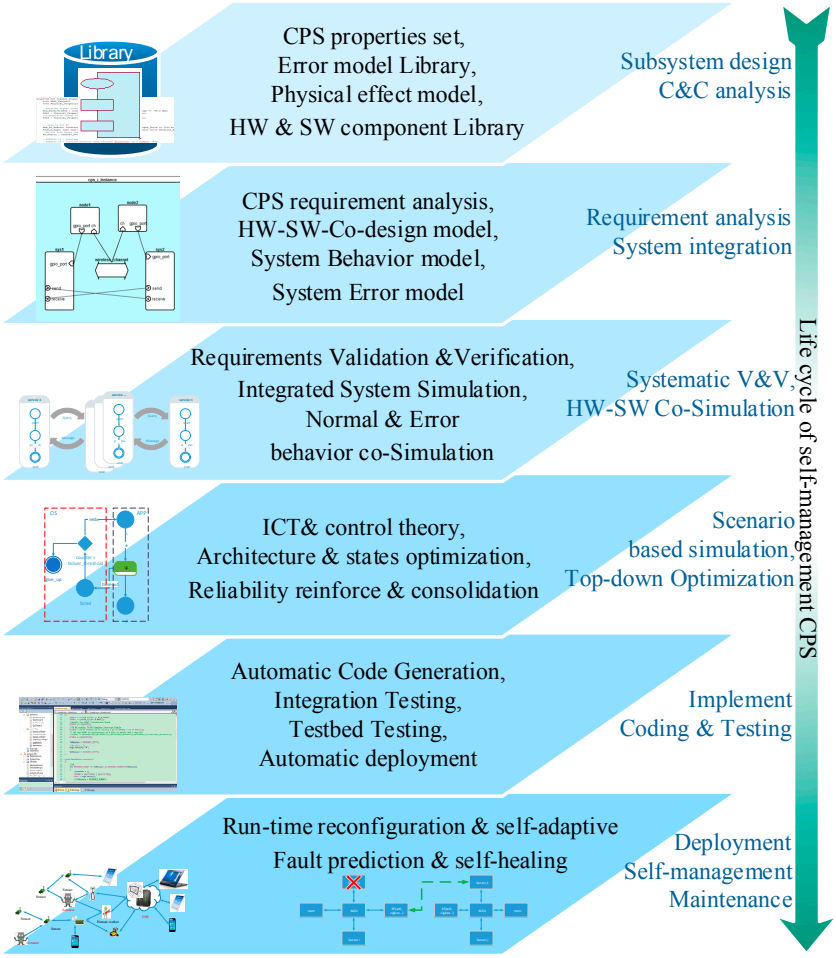

Figure 14. MDE based CPS development (an AADL based view)

Table 3. Current modeling and simulation solutions for CPS

Tools/languages	Modeling and simulation Tools[1]
Hardware modeling	Verilog, VHDL, SystemC/TLM and Analog-Mixed-Signal (AMS), Property Specification Languages, Modelica, etc.
Architecture modeling	SysML, MARTE, AADL, Behavior Interaction Priority (BIP), Abrial's B, Event-B, Functional Mockup Interface (FMI), Statecharts, etc.
Network modeling & simulation	NS2, NS-3, OPNET, OMNeT++, QualNet, SWANS++, JiST/SWAN, ANSim, COOJA, GloMoSim, ATEMU, etc.
Logical theorem	SMT, Temporal logic (LTL, CTL, TCTL, HRELTL, HTL, MTL), Hybrid Automata, Timed Automata, State Machines, Esterel, etc.
Generic modeling & simulation	Matlab/Simulink, Ptolemy II, ArduPilt, AutoSar, SCADE, LibView, AtoM, Modelyze, OpenMETA, etc.

[1] Due to the limitation of page space, some references of tools/languages are not included in this paper.

7.1 MDE based CPS approaches

MDE is regarded as a promising solution to tame the complexity and to simplify the process of engineering. Following the trend, more and more researchers attempt to use MDE to facilitate CPS development and V&V. In this section, we pay special attention on the modeling and analysis technologies applied in MDE based CPS.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

According to the literature statistic on component-based CPS architecting [11], models are used in 45 over 49 selected studies (detail analysis the paper); the common choice for modeling is applying architecture description languages or component languages, such as, AADL, AUTOSAR, Simplex, PRISMA, DEECo and Modelica, etc. Furthermore, the detail statistical analysis of the selected 49 papers is expanded on five subjects: contributions, lifecycle support, construction, extra-functional properties support, domains and components lifecycle support.

Another similar survey on MDE based life cycle management of CPS is proposed [224]. Based on the analysis of CPS's evolution, the authors pointed out the trend of self-organizing and context based CPS, the uncertainty issues in design and the challenges for CPS integration. Then the authors introduced the recent researches on functional and architectural design, methods and tools for the modeling and simulation, product lifecycle management, architecture and behavior paradigm. Lastly, a generic MDE based workflow (design, modeling, simulation and integration) for self-configuration agent-based Cyber Physical Production Systems (CPPSs) is discussed. The concepts of condition monitoring, model prediction, failure prediction, and maintenance are presented with a wind turbine case.

We select 23 relatively systemic researches from 221 related papers on MDE based CPS V&V practices. The analysis of these papers is shown in Table 4. Multidisciplinary technologies are integrated to support systemic analysis of different requirements. MDE is applied for various reasons. Here, we analyze the MDE approaches from 4 dimensions: 1) *the type of models*, 2) *the type of modeling means*, 3) *the universality of model/research*, 4) *the engineering period that the approach belongs to*. The main topics and the key solutions of selected papers are also introduced. Moreover, the statistical analysis of these literatures is shown in Figure 15. According to the Figure 15.d, currently, most researches focus on the design V&V at the early period of engineering.

Table 4. Current approaches on MDE based CPS

Ref	Cha[1]	Topic	Key solutions
[95]	D,*D*,G,*V*	Correctness V&V	Formal specification V&V on Event-B based formal model.
[96]	D,*D*,D,*V*	Correctness V&V	Proof-of-concepts with formal framework SCA-ASM.
[97]	C&D,*E*, D,*V*	Simulation based Correctness V&V	Dedicated driver model, closed-loop simulation based V&V
[208]	P,*F*,D,*R*	Unpredictable assumptions & requirements V&V	Formal modeling with HOL, randomized models for formally reasoning.
[225]	C,*F*,G,*V*	Simulation & statistical model checking	Build adequate high-level abstraction with SALMA, stochastic delays and errors are considered.
[226]	C,*E*,M,*R*	Physical side Non-functional requirements modeling.	Refined model with precise control and more detailed simulation parameters.
[227]	C&D,*F*, M,*W*	MDE process workflow	Introduce the MDE methodology with integrated models (Simulink+AADL).
[228]	C&D,*D*, G,*T*	Debugging, run-time assertion	Runtime assertion with debugging, framework through predefined constraints with new dedicated language.
[229]	C&D,*H*, G,*V*	Model-to-Model (M2M) transformations	Model transform between AADL (formally validation) and Simulink (simulation with state flow model).

[230]	D,*F*, M&D,*V*	M2M transformations	Multitude of models in a variety of formalisms, guarantee structural and semantic consistency.
[231]	D&P,*H*, G,*V*	Model (switch guard) enhancing	A dedicated language for mode-switch guards to enhance reasoning the uncertainty.
[232]	D,*D*,G,*V*	Consistency assurance for behavioral self-adaptation	Invariants based adaptation with container managed embedded UML state machine.
[233]	D,*H*,G,*V*	Co-design of scheduling and control	Mathematical formulation and analysis with 6-tuple model for IWCN.
[234]	N,*D*,G,*V*	SDN development & verification	DSML based SDN constraints verification, Model-Driven toolset with code generation.
[235]	D,*F*,D,*V*	Complexity management, consistency checking.	Consistency analysis of conjunctive & disjunctive, multi-model heterogeneous verification.
[236]	P,*E*,M,*V*	Statistical Model Checking	Cross-Entropy method based Sampling, Monte-Carlo method based rare events estimation.
[237]	D,*D*,G,*V*	Complexity management, context-aware systems verification	Model context-aware system's adaptive behavior with formal semantics, verify based on invariants.
[238]	C&D,*H*, G,*V*	Tool for Non-functional property checking	SysML/MARTE, Modelica & Statistical Model Checking integration with Functional Mock-up Interface.
[239]	D,*F*,G,*T*	Distributed model-based testing	Modeling with UTA, using serialization service and NTP Protocol to achieve Δ-testability (< 1 ms).
[240]	C&D&P, *E&D*, M&D,*W*	modeling, co-simulation & optimization (correctness, reliability)	Co-simulation with a set of tools; integrating continuous, time-discrete & event-discrete models; real-time synchronized distributed simulation.
[241]	C&D, *F&E&H*, G&D[2],*V*	Integrated formal modeling, joint-analysis	Integrated model with 5 formal specifications (global model, Interface, CS, Atomic Model, Coupled Model)
[242]	D,*D*,D, *V&T*	Simulation-based testing and evaluation, tool Integration	Three-in-One simulators (Traffic, Network Driving), integrated with interface coordinator.
[243]	C&D&P, *F&E&D*, G,*V*	Distributed simulation (MoC)	Integration of Ptolemy II and HLA, use discrete events, synchronous-reactive, synchronous dataflow, process networks, continuous-time

[1] Characteristics: <model type, *mean*, universality, *engineering period*>; model type= <continuous(C), discrete (D), possibility (P)>; mean = < equations (E), formal language (F), dedicated language/generic programming language (D), hybrid (H) >; universality = <generic model/method (G), methodology (M), domain dedicated model/method (D)>; engineering period= <system analysis/design/V&V (V), requirement analysis(R), testing/debugging (T),development(D), whole life cycle (W)>; and all 'N' means the item cannot be inferred from the paper or its related references. [2]generic method but introduced with a dedicated case (model)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

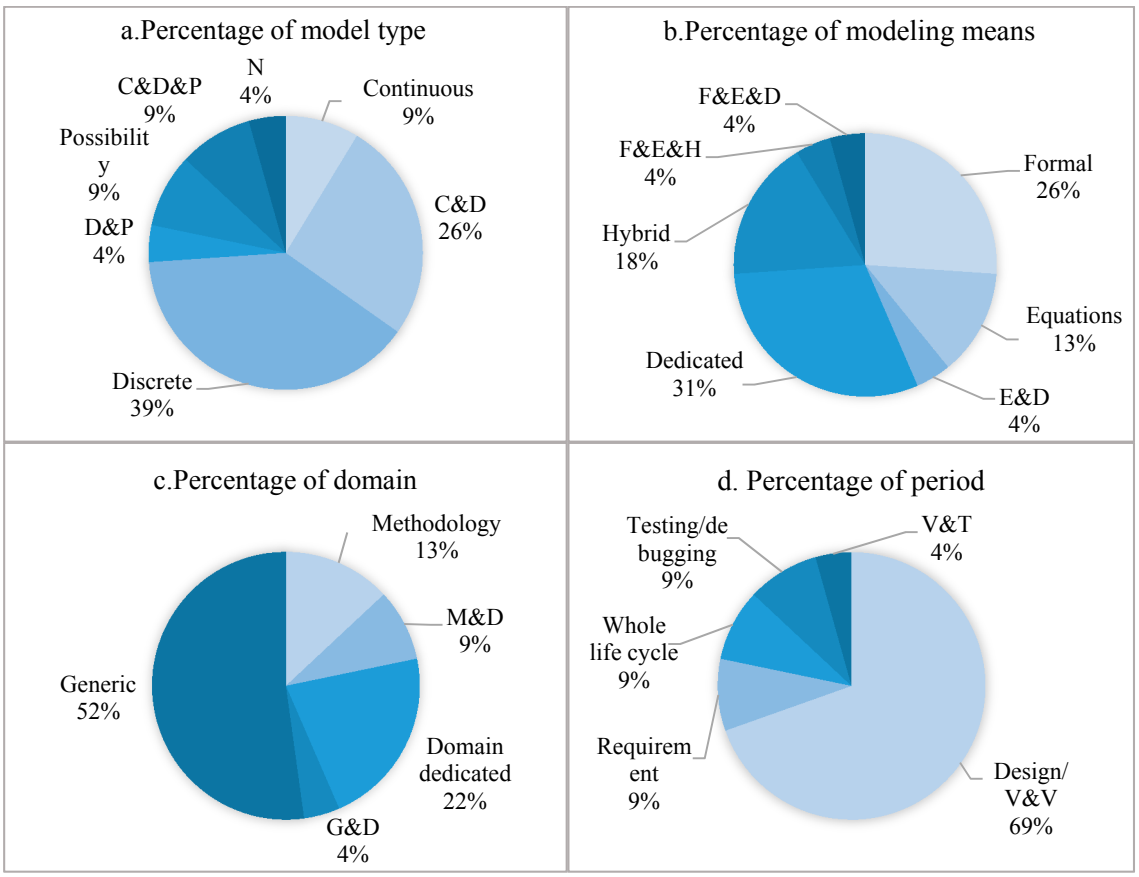

Figure 15. The literature statistic on MDE based CPS (the amount of selected paper is 23), a. percentage of model type; b. percentage of modeling means; c. percentage of MDE research domain; d. percentage of engineering period that the approach belongs to

7.2 MDE on dependable (self-management) CPS V&V

In short, fault diagnosis depends on the analysis on the *causal relationship* between symptom and error source. The core idea of fault recovery is breaking or disabling the *cause-effect connection* by removing the error source or preventing fault trigger events. So ***discovering the causality is the core for dependability management***. As an all-in-one solution, MDE can ***naturally provide a holistic view of the causal relationship*** of the error in the system because ***the models in MDE are formed with the internal logic and objects*** (*possible error sources*)***, the interfaces*** (*available fault propagation paths*) ***and the interaction*** (*the direction/causality of fault propagation*) ***between components***.

Yet, as mentioned at the beginning, few researches have published on the dependability of CPS. In the section, we ease the restrictions of eligible papers to *MDE based* and *model based* dependability V&V. A comprehensive survey on MDE of fault-tolerate (software) systems is presented [244], and 10 MDE approaches are analyzed from 13 characteristics. Yet this survey is dedicated to high reliable/available component-based information and communication system, no MDE approach for physical side is analyzed. Another detail survey on models based the root-cause analysis (RCA) is proposed [245]; the authors introduced the models, the inference techniques and the theoretical complexity of inference. Combined with the two surveys [244-245] and the analysis in section 6.1, we can draw the workflow for MDE based dependable CPS V&V which is illustrated in Figure 16.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

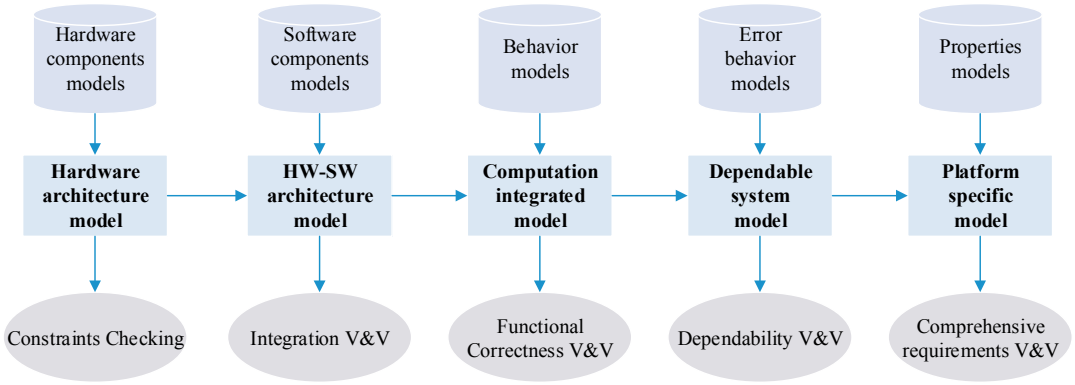

Figure 16. Model integration workflow of MDE (Top-down design view in AADL)

For MDE based CPS dependability V&V, we need to generate the RCA models from the reference models of MDE. In order to generate the RCA model automatically, the additional effects should be taken. 1) *Specializing the error behavior model*: developers should sign all original sources of errors and define the trigger conditions and the occurrence probability of each faults; and for the safety-critical system evaluation, the consequence of failures should be given. 2) *Embedding the specialized error behavior model in computation integrated model*: developers should bind the states and variables between normal model and error model, define the connection of normal behavior and error behavior. 3) *Transforming the dependable system model to RCA models or simulation model*, i.e. transforming the hybrid AADL behavior model to petri net model. Then we can do dependability V&V with the current RCA tools. As mentioned above, another alternative solution of step 3 is *generating executable model*, and *verifying dependability with simulation*. The workflow of MDE based V&V and model based V&V are illustrated in Figure 17. We reviewed both MDE based and model based dependability V&V researches on CPS, which is shown in Table 5.

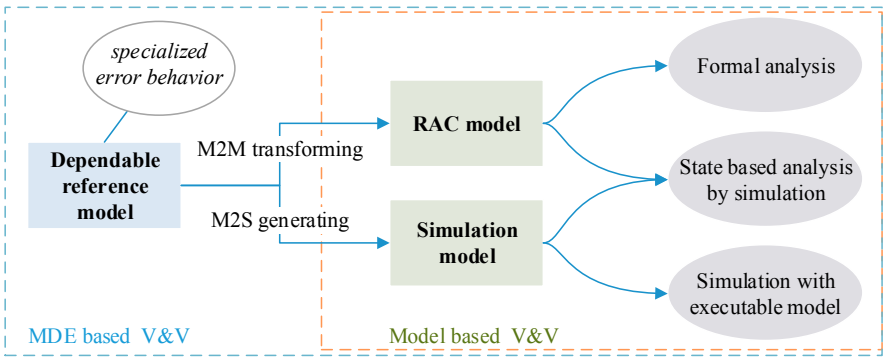

Figure 17. The role of model based V&V in MDE process

Table 5. Recent researches on MDE/model based dependable (self-management) CPS V&V

Ref	Type[1]	V&V[2]	Key analysis technologies
[140]	Sim	SF, Rb	Simulation and statistical analysis
[210]	RCA	R	RBD, Markov Chain (MC), Monte-Carlo simulation
[216, 246]	M2M	C,D,P	NuSMV, FTA, FMEA, HSIA, MC
[247]	RCA	R	MC
[248]	RCA	R, A, ST	Stochastic Petri Net
[249]	M2M	D	Dependability domain ontology, FMEA
[250]	RCA	D	MC, Stochastic Activity Network
[251]	M2M	C,A,Rb	BDD, BMC, FTA, FMEA
[252]	M2M	C,SF,	probabilistic temporal logic language, MC

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

[253]	M2S	C,Rb,R, SC	Simulation & calculation
[254]	M2M	R,A,M,SF	Bayesian Belief Network
[255]	Automata & LTL	C,R	model-based diagnosis, contract checking
[256]	M2S	SF, Rs	Simulation and statistical analysis

[1] RCA means analyzing the dependability with original dependable model (without transforming), M2M means transforming reference model to RCA model, Sim means analyzing with original simulation model, M2S means generating the simulation model from reference model; [2] Available items: Correctness (C), Dependability (D),Reliability(R), Availability(A), Safety(SF), Sustainability(ST), Robustness(Rb), Maintainability(M), Performance(P), Security(SC), Reusability (Rs)

7.3 Conclusion on CPS modeling

Model is the core of MDE methods, it limits the power of MDE based V&V. The control model generally is abstracted at a low level with PDE/ODE equations, which is dedicated to a special case. The lack of high level abstraction raises the complexity of V&V, increases the cost of analysis and simulation, and also decreases the flexibility of the model. By contrast, high level abstraction (i.e. UML model) is more generic but may miss details, and the discrete model is inappropriate for describing the dynamic behavior. Furthermore, in some cases, only statistical probability is available during the design cycle, no other accurate data/behavior can be referred to (i.e. the time when a fault occurs). As a result, the probability (non-determined) model may be the only choice, and probability is the nature method to describe the stochastic events in CPS.

The *mathematical* model can provide the quantitative analysis, but generally is one model for one target system, and it can only analyze one property/index of the target system with one model, which normally is applied in the domain specified model as highlighted in [226]. For CPS, building a set of pure mathematical models is a heavy work. It is a skillful practice and requires deep understanding of the internal mechanisms of system and physical phenomenon. *Formal* models can also provide rigorous analysis and it is generally applied for correctness V&V. Similar to the mathematical model, it is impossible to provide abstraction of CPS at all levels in different details with one formal model. A set of formal models are required to describe the systemic CPS, i.e. [241]. Guaranteeing the consistency of these models is a big challenge. *Dedicated modeling language* can fit the target case very well, but is poor in universality. By contrast, *generic programming language* has great universality but is clumsy. Both of the two languages convert the task of modeling to programming, and the correctness of the model itself also should be verified carefully. *Hybrid* method is to take the advantages of different methods and mix them together to build a CPS model. To overcome the difficulties of interaction between methods (i.e. sharing the states and values), some interfaces and operations should be defined.

7.4 Conclusion on MDE based CPS V&V

There are two types of V&V, one is static analysis and another is dynamic simulation. Static analysis means the model is static during analysis, it checks the model based on the syntax, constraints, rules or contracts, what should be mentioned that the analysis means itself may be dynamic (i.e. statistical model checking). Most MDE based static analysis approaches transform the meta-model (CPS model) to checking model or apply the checking methods on the meta-model. MDE based dynamic simulation fallows the concept Models of Computation (MoC). It takes the model as a program. By abstracting the objects in the real world, the researchers build a dynamic and executable model or generate an executable model according the meta-model, then execute the model and verify the detailed behavior of these objects.

Due to the complexity of CPS, it is almost impossible to analyze systematically the CPS's behavior in a proper detail with formal methods. Simulation based V&V is the alternative solution. Roughly speaking, there are four main types of modeling & simulation solutions: 1) *Mathematical (numerical) simulation*, i.e. control system simulation with Simulink; 2) *Symbolic based simulation* (automated theorem proving), e.g. temporal logic, symbolic logic or automata; 3) *Monte Carlo*

simulation (probability simulation), i.e. DFT analysis with Monte Carlo method; 4) *Event-driven simulation*, i.e. network protocol simulation with NS-2. These four solutions have been well investigated and widely applied in different disciplines at different levels, ***but they have not been integrated organically***.

For physical level simulation, i.e. simulating physical objects with physical laws: it is a cost efficient way for hardware design, which may accelerate the development significantly. **Building more faithful model** and **developing the simulator (emulator) supporting more precise physical parameters** are the main research area for MDE on CPS, especially the effects of random events e.g. [62, 226].

For strategies level simulation: it is a complimentary technique for formal analysis, and tries to solve the problem which is difficult for quantitative calculation, e.g. the performance of random events triggered scheduling, stochastic process, etc. **guaranteeing/estimating the coverage of the simulation** is the main challenge.

For components' interaction simulation: it analyzes the interaction behavior of components, e.g. synchronization, communication, resource competition, etc. generally the structure of these components is static. Behavior Interaction Priority (BIP) simulation and hardware-in-the-loop simulation are two representative researches. As CPS has a dynamical architecture, **modeling** and **simulating the dynamical interaction among components** is a challenging mission.

*For multi-agents (*nodes*) simulation*: current researches focus on the wired/wireless network communication and cooperation amongst abundant nodes (agents), e.g. NS2 (network), cellular automata (cooperation). Yet, normally, the nodes in the simulation are homogeneous. In addition, for simplicity, other factors are ignored, i.e. the ideal physical environment (transmission radius of radio is an ideal circle). As argued earlier, the physical environment has a strong influence on cyber space; such assumptions will induct the simulation to generate a result with a little reference value for dependable CPS V&V. Like physical level simulation, **to build more refined communication channel model** and **to develop a high fidelity simulation need to take into account more precise physical parameters.** Moreover **developing a high performance distributed simulator with strict time synchronization** will improve the quality of simulation significantly**.**

For CPS detail behavior and effects simulation: this is the ultimate goal for simulation based CPS V&V. As all factors at all levels can affect the behavior of CPS, comprehensive simulation is necessary for *adaptation strategies* (correctness) *decision evaluation (*correctness*), prophetic dependability management (*dependability*), environment-in-loop evaluation (*C&D*),* and *human-in-loop evaluation (*C&D*)* etc.

7.5 Discussion on CPS modeling & simulation

There is no doubt that computer modeling & simulation has emerged as an indispensable method for CPS research, but 1) what kind of model can faithfully reproduce the behaviors of CPS and environment, discrete model or continuous model? 2) Can the selected model naturally cooperates with other runtime technologies (i.e. data-driven machine learning)? The answer of the first question may depend on philosophical view of the physical world and cyber world. Is the physical world discrete or continuous? Do we live in a probabilistic or deterministic world? Here, we do not want to go deep into the philosophical discussion. In fact, we just show some possible views, which may improve the understanding of CPS and design better models and simulation tools.

Generally, in Newtonian space-time, physical process is believed to be continuous, and it is modeled with algebraic equations or differential equations (e.g. ODEs, PDEs and DAEs etc.). Yet, model electronic computers can't process the continuous signal directly (generally, it should be converted into digital signal with ADC first). What's more, from the quantum space-time view, could it be possible to analyze the physical process with discrete model without losing precision and real-time? How to achieve a good balance between performance and fidelity? Is it possible to build a grand unification model (hybrid model) to represent the processes of computing, communication, actuations and physical effects with on-demand abstractions? Considering the limited perception of the physical world and the cyber space, are formal methods flexible enough to demonstrate the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

complex stochastic behavior of CPS? If not, is the probability method a better solution? An ideal solution should be able to model the physical process, the computation and the communication and the interaction among them.

In order to be analyzable, formal/mathematical methods ignore so many factors that it is almost impossible to analyze the detail behavior. Simulation is the only systematical solution. There are 4 kinds of simulation for V&V: 1) *mathematical/numerical simulation*; 2) *automated theorem proving*; 3) *Monte Carlo simulation*; 4) *event-driven simulation*. However, none of them can model a complex CPS independently as it involves multi-disciplines. E.g. **1)** the Zeno effect troubles the control system simulation [129] and there are few attempts at building numerical model for software. **2)** Most satisfiability (SAT) problems are the NP problem (actually, k-SAT is an NP problem when k>2) which implies that validation has poor performance, and formalisms generally not well suited to reflect the stochastic nature of communication in a cyber-physical setting [224]. 3) It may very difficult to get the exact probability distribution in real world, and the cases with small probability may never be tested. **4)** Event-driven model is dedicated to discrete system, it can't constantly represent the physical process.

It seems that no solution is perfect enough for modeling CPS alone. 1) Maybe we should organically integrate both discrete model and continuous model, non-deterministic (probability) model and deterministic (formal) model together (i.e. co-simulation toolset [215]). In addition, the key challenges will be how to describe the fine grain interaction between models and improve the performance of V&V. Or 2) we can build a unified reference model and then transform it into a proper model based on the different purposes for V&V (i.e. AADL based toolset [216]). Improving current model to support domain-specific properties and keeping the consistency semantic during model transforming are the two main challenges.

8. The Jigsaw and Future Directions of CPS Technology

CPS is a new multidisciplinary cross-domain and rapid progressing research area. It requires great skills to trade off among these complex requirements. The challenging problem is run-time context adaptation, CPS should work out a compromise automatically to adapt to various scenarios dynamically. What makes things even worse is that the data/events for decision making may be incomplete, insufficient and even incorrect. With the increasing scale of CPS, self-management will be an important ability for CPS. Developing a dependable self-managing CPS demands not only self-healing capability, but also high maintainability for the maintenance from both inside and outside of the system.

8.1 Available and missing measures

To develop the dependable self-managing CPS and to maintain the correctness and dependability through the whole life cycle, comprehensive methods should be integrated at different levels of the system. We conclude the popular (may be potential) methods and frameworks from thousand papers for dependable CPS engineering. Correctness V&V methods, self-adaptation solutions and V&V methods, dependability and V&V methods, MDE solutions (toolsets) are investigated at four main levels: 1) hardware level (nodes and network devices), 2) basic software level (operating system and network protocols), 3) middleware and framework, 4) domain/aspect oriented solutions. Considering the readability, we select part of the results, which are illustrated in Figure 18.

Based on the jigsaw of technologies, we can draw seven conclusions. **1)** The researches of precision timing behavior are just at the beginning and most of them focus on the hardware & basic software level, few systemic tools or theories on this field are proposed. **2)** The dependability methods can be applied at any level, while the weight of flexibility increases with the rising of level. **3)** Simulation based methods and dynamic logic based models (whose process of analysis is also a simulation based) dominate the V&V of correctness and dependability at the high level. **4)** C&C is a promising solution to improve the flexibility and to mitigate the increasing of the complexity. **5)** Highly flexible architecture (i.e. ABSA, SDA, and SDN) is the base to achieve the self-management

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

and a key solution to improve the maintainability. **6)** MDE is a holistic solution to tame the complexity and the decrease the risk of CPS engineering. It should be mentioned that **7)** specifications [257-258] is also the useful measure to guarantee the quality of CPS.

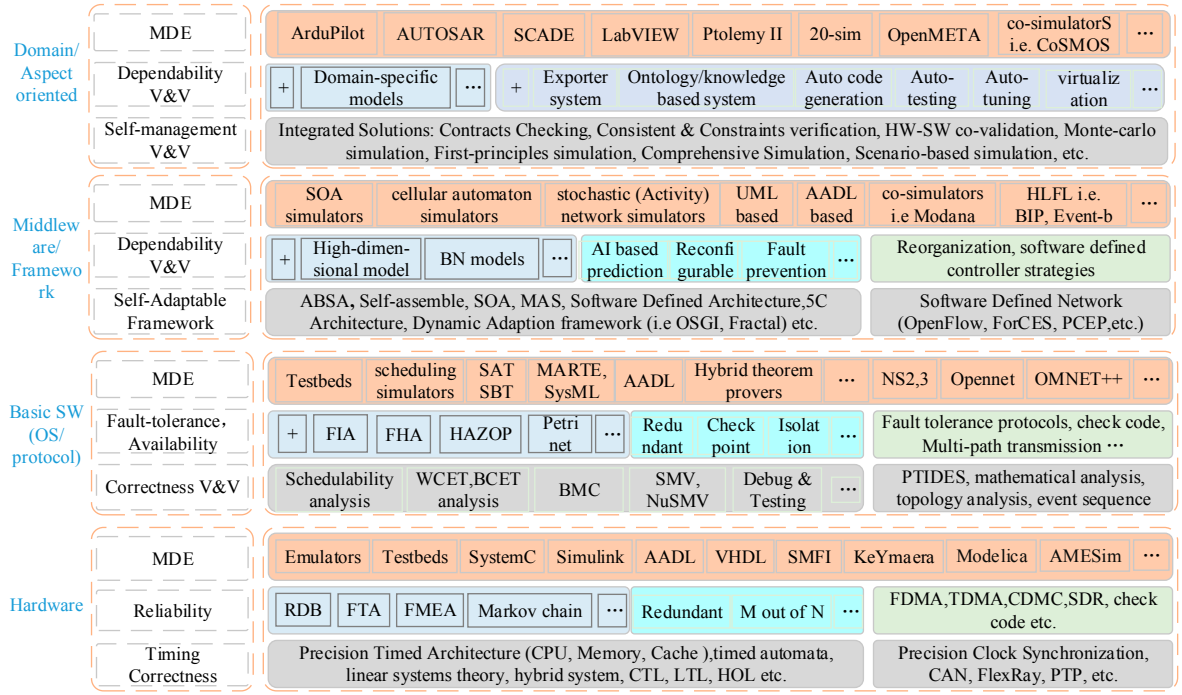

Figure 18. Available measures for dependable self-managing CPS at different levels[1]

Seven missing pieces of current jigsaw are concluded: **1)** theory, infrastructure and tools supporting for timing behavior at high level are urgently needed. **2)** Precision timed wireless network is emerging. **3)** Network integrated V&V and large scale supported toolset are needed. **4)** The coverage and performance of testing and simulation should be improved. **5)** The side effect of dependability management should be continually studied. **6)** Dynamic strategies for tradeoff among various constraints, which aware the environment and the statues of the system are required. **7)** The research on the cooperation of these dynamic strategies should be put on the agenda.

8.2 Technical drections of dependable self-managing CPS development

CPS contains various technologies. However, the constraint spaces of the requirements of these technologies are just a subset of CPS's. To be applicable for CPS, first, these technologies should be improved to satisfy requirements of CPS. Second, the involving technologies should share the consistent constraints and the same meaning of communication data. CPS is an organic integration. In fact a lots of technologies are available, but they are still not enough for CPS development. In this section, we briefly summarize the challenges of dependable self-managing CPS development discussed above, which are shown in Table 6. In addition, the urgency of technology depends on 1)

1. We prefer to list the theory or the type of technologies rather than the detail tool in the figure. Yet, MDE is an integration of technologies, we have to name the classic toolsets instead. The priority of selection is theory > category of technologies > language > tool. The frequency is the main selection rule.

Notice: it's impossible to list all the technologies in one picture, and just some popular methods are mentioned. Due to our limited knowledge, some other methods may be missed. All suggestions are welcomed to refine the jigsaw.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

the maturity of the technology [5], 2) the social expectation and acceptance of technology [42], 3) the degree that other technologies depends on target technology (based on Figure 18).

Table 6. New technical challenges of dependable self-managing CPS

Technical Area	New Technical Technologies[1]	Urgency	Target
Hardware & software infrastructure development	Precision timed, real-time HW & SW	High	Timing
	Standardization of subsystem (interfaces)	Medium	C&C
	Low power devices	Medium	Energy
Network communication & management	Precision timed network transmission	High	Timing
	Real-time (wired & wireless) communication	High	Real-time
	Heterogeneous network management	Medium	Maintainability
Architecture design	Atomic service & subsystem design	Low	C&C
	C&C contract, interoperable subsystems	Medium	Self-*
	Discrete-continuous subsystem integration	Medium	Correctness
	Invariant behavior of integration	High	Correctness
	Theory for dynamic architecture	High	Flexibility
	Methodology for holistic design	Medium	Design
Middleware	FDIR middleware & Node level self-healing	Medium	Dependability
	Light-weight virtualization & migration	Medium	Self-*
	Domain ontology, Knowledge database	Medium	Self-*
	Service discovery & combination	High	Self-*
Consistent spatio-temporal & context cognition	Global reference time for large scale CPS	High	Timing
	Low cost clock synchronization	Medium	Correctness
	Global location reference for mobile CPS	Low	Correctness
	Consistent data and context assurance	High	Correctness
Lifecycle management (self-management)	Manage dynamic & changeable architecture	High	C&D
	Multi-objective (prophetic) adaptation	High	C&D
	Knowledge-driven decision making	High	C&D
	Decision/adaptation safety/evaluation	Medium	Safety
	Situation aware self-healing & notification	High	Dependability
	Causality analysis	High	C&D
	HMI for Human-in-loop CPS	High	Usability
Modeling & validation	Newtonian time model for temporal behavior	High	Fidelity
	Dynamic architecture modeling	High	Fidelity
	Multidisciplinary modeling	High	Modeling
	Consistent of model transforming	High	Correctness
	Evaluation the correctness of models	High	Correctness
	Holistic modeling theory or methodology	Medium	Modeling
	Situation based model V&V	Medium	V&V
Simulation	Discrete-continuous-probability co-simulation	Medium	V&V
	Holistic multidisciplinary simulation	High	V&V
	Environment-in-loop simulation	Medium	V&V
	Human-in-loop simulation	High	V&V
	Fidelity evaluation	High	Correctness
MDE tools	MDE toolchains (design, V&V, coding, testing)	Medium	Efficient

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 July 2017 doi:10.20944/preprints201707.0044.v1

	Whole life cycle V&V supporting	Medium	V&V

[1] The challenges of security and traditional technologies are not discussed in this table.

8.3 Future direction: an concept of all-in-one solution

As argued in section 6.5, simulation is the most promising solution for V&V of complex system and strategies. However, it is still impossible to provide complete V&V for the self-adaptation strategies and the self-healing strategies during design period, because the possible scenarios/contexts are infinite. Runtime V&V can improve the efficiency significantly, and models@run.time is the solution for runtime V&V. Notice that simulations for the self-adaptation and the self-healing strategies demand the additional architecture information. Therefore, we can go a step further and build all-in-one simulation for dependable self-managing CPS. The Figure 19 illustrates the conceptual solution.

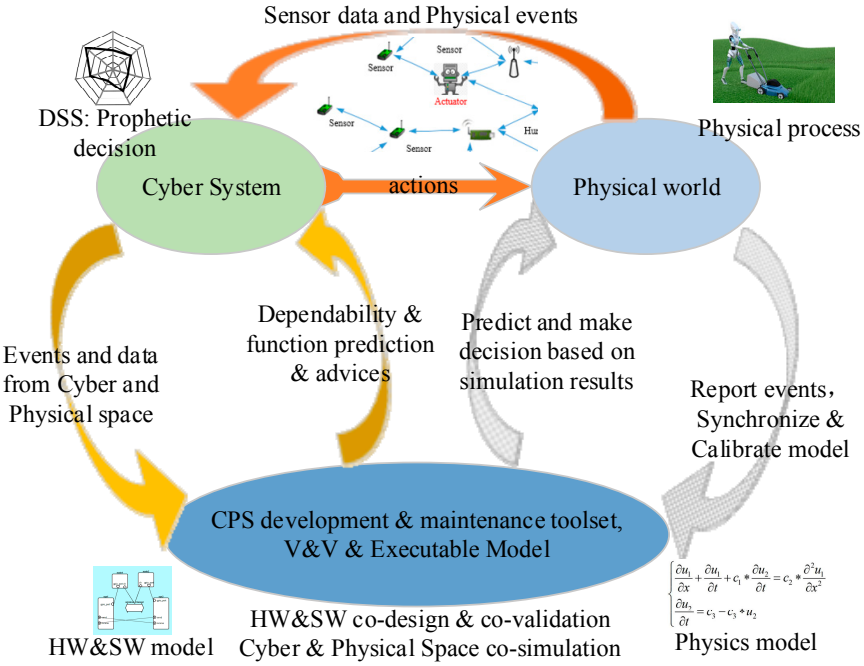

Figure 19. Co-validation and co-simulation through the whole life cycle of CPS

To achieve this goal, we should build the models of both the cyber system and the physical world. With the improvement of MDE tools, all these models will be available with little additional cost. The cyber system just needs to send a copy of data and events to the simulator then the simulator can evaluate the current situation and search the potential self-management strategies. Further, the simulator can forecast the future with highest occurrence possibility and provide the best strategy or a set of possible strategies (advices). This forms a close-loop runtime V&V for dependable self-managing CPS. With MDE based solution for development period V&V, we can build a whole life cycle evaluation for dependable self-managing CPS.

9. Discussion and conclusion

CPS will deeply involve in daily life. The human being becomes a subsystem of CPS, which is an uncontrollable and unpredictable factor to CPS. The researches in [62-64, 259-260] introduce the recent exploration from molding human being's behavior to management for this problem. What's more, CPS is tightly embedded in the physical world and will be influenced by the environment easily. Some researchers have already started to take the environmental factors into account [261-262].

Dependable self-managing CPS developing and maintaining is a fine art, which demands holistic measures, including modeling technologies, development technologies, project management technologies and standardization of these technologies. The system will not perform well if any part is imperfect. Though numerous researches have been done in the decade, the jigsaw implies that both the modeling technologies and the infrastructures and self-management technologies must be continuous improved for practical CPS.

Model is the key to CPS theory, and MDE is the key to CPS development. Unfortunately, both of them need to be improved for CPS engineering. However, models and MDE tools can't be developed by a single team. It calls for cooperation of stakeholders coming from different engineering disciplines. As dependability and correctness depends on the context, domain-specific models and environment-specific/scenario-specific models are necessary for detail analysis. Domain-specific modeling languages will be a better solution than generic modeling language to build a high-fidelity CPS model. Therefore, we can embed the platform, environment and scenario dependent properties in a generic reference model.

Formal V&V and simulation based V&V are two useful solutions to find the defects in design period. However, current methods can't systematically analyze the dependability and correctness of CPS alone. Integrated methods are the common solutions, yet guaranteeing the consistency of these methods is still an open issue. What's more, integrated methods are still not enough for systematically analysis. Cross-level and multidiscipline simulation are urgently needed for CPS V&V. To avoid unnecessary V&V, models@run.time and runtime V&V are recommended. In addition runtime dependability management should be integrated into CPS.

Self-managing CPS can't be achieved without dynamic architecture. Architecture-based self-adaptation (ABSA) is a great concept for CPS development, but we need more technical detail. Software defined architecture and software defined network show an alternative solution. The core solution is separating the control logic of structure and topology from physical elements, and decoupling the architecture management flow and functional process flow. Meanwhile, virtualization can isolate the applications by providing separated runtime environments. Consequently, we can control the VM or the container instead of embedding the control logic in functional logic of applications. Such decoupling decreases the complexity of logic and improves the dependability and flexibility of CPS. Software defined CPS will be a promising solution for developing dependable self-managing CPS.

Compared with model driven adaptation, data-driven is more flexible and higher adaptive. Machine learning (ML) can learn to adapt to the new environment, but the learning process is time and cost consuming. Current online learning methods may be inapplicable for safety-critical CPS. Besides, error data (error behavior) is relatively scarce because failure occurs at low frequency and error data is difficult to be collected, i.e. a failed system can't send back data. Lack of training data hinders the application of ML based fault diagnosis for CPS. Model based fault injection can alleviate this issue, but it can't solve totally the problem because the scenarios are infinite. We can use ML to classify the scenarios (building the association between symptom and scenarios) and design a FDIR model (building the causality between symptom/failure and error) for each type of scenario, then create some methods to smoothly transfer from one model to another (for **RQ1**). Moreover, we can check the results from ML models with knowledge base methods or expert system (for **RQ1** & **RQ4** & **RQ5**).

Building specification and improving the C&C of subsystem can fundamentally decrease the complexity of architecture and service reconstruction (for **RQ2**), but it still can handle the complexity caused by increasing scale. As CPS is SoS, it is built with abundant similar subsystems, we can take the advantage of the self-similarity of CPS's architecture and behavior to slow the increasing complexity. E.g. we can use the transfer learning method to learn the model/behavior from similar subsystem (for **RQ3** & **RQ4**). The subsystem can monitor and check with each other (for **RQ4** & **RQ5**).

CPS is integrations of multidisciplinary technologies. Holistic design theory and systematic development tools are the emergency technical needs for CPS engineering. Notice that project management methodology and standardization of CPS can improve the quality of design and

simplify the maintenance. However, none of them can be achieved by a single team, so international multidisciplinary cooperation is urgently needed to investigate the dependable self-managing CPS development and evaluation.

Acknowledgments: This work was supported by a grant from National High Technology Research and Development Program of China (863 Program) (No. 2013AA01A205), and National Natural Science Foundation of China (No. 61370085).

Author Contributions: The survey is a product of the intellectual collaboration of all authors. Z.P. contributed to investigation and writing; Z.D.C. and K.M.H supervised the work; Z.D.C. and K.M.H and Z.Z. contributed to manuscript organization; D.J. and LJ.J and Z.H.Y. provided insightful comments and suggestions.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M., Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems* **2013,** *29* (7), 1645-1660.

2. Mosterman, P. J.; Zander, J., Industry 4.0 as a cyber-physical system study. *Software & Systems Modeling* **2016,** *15* (1), 17-29.

3. Kang, H. S.; Lee, J. Y.; Choi, S.; Kim, H.; Park, J. H.; Son, J. Y.; Kim, B. H.; Do Noh, S., Smart manufacturing: Past research, present findings, and future directions. *International Journal of Precision Engineering and Manufacturing-Green Technology* **2016,** *3* (1), 111-128.

4. Gunes, V.; Peter, S.; Givargis, T.; Vahid, F., A Survey on Concepts, Applications, and Challenges in Cyber-Physical Systems. *KSII Transactions on Internet and Information Systems (TIIS)* **2014,** *8* (12), 4242-4268.

5. Leitão, P.; Colombo, A. W.; Karnouskos, S., Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges. *Computers in Industry* **2016,** *81*, 11-25.

6. Lu, C.; Saifullah, A.; Li, B.; Sha, M.; Gonzalez, H.; Gunatilaka, D.; Wu, C.; Nie, L.; Chen, Y., Real-time wireless sensor-actuator networks for industrial cyber-physical systems. *Proceedings of the IEEE* **2016,** *104* (5), 1013-1024.

7. de Brito, M. S.; Hoque, S.; Steinke, R.; Willner, A., Towards Programmable Fog Nodes in Smart Factories. *2016 Ieee 1st International Workshops on Foundations and Applications of Self* Systems (Fas*W)* **2016**, 236-241.

8. Wan, J.; Chen, M.; Xia, F.; Di, L.; Zhou, K., From machine-to-machine communications towards cyber-physical systems. *Computer Science and Information Systems* **2013,** *10* (3), 1105-1128.

9. Ceccarelli, A.; Bondavalli, A.; Froemel, B.; Hoeftberger, O.; Kopetz, H., Basic Concepts on Systems of Systems. In *Cyber-Physical Systems of Systems: Foundations – A Conceptual Model and Some Derivations: The AMADEOS Legacy*, Bondavalli, A.; Bouchenak, S.; Kopetz, H., Eds. Springer International Publishing: Cham, 2016; pp 1-39.

10. Völp, M.; Asmussen, N.; Härtig, H.; Nöthen, B.; Fettweis, G. In *Towards dependable CPS infrastructures: Architectural and operating-system challenges*, Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on, IEEE: 2015; pp 1-8.

11. Crnkovic, I.; Malavolta, I.; Muccini, H.; Sharaf, M.; Ieee, On the Use of Component-Based Principles and Practices for Architecting Cyber-Physical Systems. *Proceedings 2016 19th International Acm Sigsoft Symposium on Component-Based Software Engineering* **2016**, 23-32.

12. Cyber-Physical Systems. http://cyberphysicalsystems.org/ (accessed 2017-6-20).

13. Webinar, K. NIST Cyber-Physical Systems Public Working Group (CPS PWG). https://www.nist.gov/sites/default/files/documents/el/CPS-PWG-Kickoff-Webinar-Presentation-FINAL.PDF (accessed 2017-6-20).

14. Alho, P.; Mattila, J., Service-oriented approach to fault tolerance in CPSs. *Journal of Systems and Software* **2015,** *105*, 1-17.

15. Pfrommer, J.; Stogl, D.; Aleksandrov, K.; Navarro, S. E.; Hein, B.; Beyerer, J., Plug & produce by modelling skills and service-oriented orchestration of reconfigurable manufacturing systems. *At-Automatisierungstechnik* **2015,** *63* (10), 790-800.

16. Dai, W.; Huang, W.; Vyatkin, V.; Ieee, Knowledge-Driven Service Orchestration Engine for Flexible Information Acquisition in Industrial Cyber-Physical Systems. In *Proceedings 2016 Ieee 25th International Symposium on Industrial Electronics*, 2016; pp 1055-1060.

17. Leitao, P.; Karnouskos, S.; Ribeiro, L.; Lee, J.; Strasser, T.; Colombo, A. W., Smart Agents in Industrial Cyber–Physical Systems. *Proceedings of the IEEE* **2016,** *104* (5), 1086-1101.

18. Giordano, A.; Spezzano, G.; Vinci, A. In *Smart Agents and Fog Computing for Smart City Applications*, International Conference on Smart Cities, Springer: 2016; pp 137-146.

19. Zhang, Y.; Qian, C.; Lv, J.; Liu, Y., Agent and cyber-physical system based self-organizing and self-adaptive intelligent shopfloor. *IEEE Transactions on Industrial Informatics* **2016**.

20. Lee, J.; Ardakani, H. D.; Yang, S.; Bagheri, B., Industrial big data analytics and cyber-physical systems for future maintenance & service innovation. *Procedia CIRP* **2015,** *38*, 3-7.

21. Kit, M.; Gerostathopoulos, I.; Bures, T.; Hnetynka, P.; Plasil, F. In *An architecture framework for experimentations with self-adaptive cyber-physical systems*, 2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, IEEE: 2015; pp 93-96.

22. Dong, J.; Xiao, T.; Zhang, L., A Prototype Architecture for Assembly-Oriented Cyber-Physical Systems. In *AsiaSim 2012*, Springer: 2012; pp 199-204.

23. Zhong, H.; Nof, S. Y., The dynamic lines of collaboration model: Collaborative disruption response in cyber–physical systems. *Computers & Industrial Engineering* **2015,** *87*, 370-382.

24. Broman, D.; Zimmer, M.; Kim, Y.; Kim, H.; Cai, J.; Shrivastava, A.; Edwards, S. A.; Lee, E. A. In *Precision timed infrastructure: Design challenges*, Electronic System Level Synthesis Conference (ESLsyn), 2013, IEEE: 2013; pp 1-6.

25. Bui, D.; Lee, E.; Liu, I.; Patel, H.; Reineke, J. In *Temporal isolation on multiprocessing architectures*, Proceedings of the 48th Design Automation Conference, ACM: 2011; pp 274-279.

26. Dierikx, E. F.; Wallin, A. E.; Fordell, T.; Myyry, J.; Koponen, P.; Merimaa, M.; Pinkert, T. J.; Koelemeij, J. C.; Peek, H. Z.; Smets, R., White Rabbit Precision Time Protocol on Long-Distance Fiber Links. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control* **2016,** *63* (7), 945-952.

27. Bures, T.; Hnetynka, P.; Plasil, F. In *Strengthening architectures of smart CPS by modeling them as runtime product-lines*, Proceedings of the 17th international ACM Sigsoft symposium on Component-based software engineering, ACM: 2014; pp 91-96.

28. Jäckel, M.; Falk, T.; Landgrebe, D., Concept for Further Development of Self-pierce Riveting by Using Cyber Physical Systems. *Procedia CIRP* **2016,** *44*, 293-297.

29. Wang, L.; Törngren, M.; Onori, M., Current status and advancement of cyber-physical systems in manufacturing. *Journal of Manufacturing Systems* **2015,** *37* (Part 2), 517-527.

30. Liu, H.; Sun, D.; Liu, W., Lattice hydrodynamic model based traffic control: A transportation cyber-physical system approach. *Physica a-Statistical Mechanics and Its Applications* **2016,** *461*, 795-801.

31. Cassandras, C. G., Smart Cities as Cyber-Physical Social Systems. *Engineering* **2016,** *2* (2), 156-158.

32. Nasir, H. A.; Muhammad, A., Control of Very-Large Scale Irrigation Networks: A CPS Approach in a Developing-World Setting. *IFAC Proceedings Volumes* **2011,** *44* (1), 10739-10745.

33. Moness, M.; Moustafa, A. M., A survey of cyber-physical advances and challenges of wind energy conversion systems: prospects for internet of energy. *IEEE Internet of Things Journal* **2016,** *3* (2), 134-145.

34. Raffe, W. L.; Tamassia, M.; Zambetta, F.; Li, X.; Mueller, F. F. In *Enhancing theme park experiences through adaptive cyber-physical play*, Computational Intelligence and Games (CIG), 2015 IEEE Conference on, IEEE: 2015; pp 503-510.

35. Khaitan, S. K.; McCalley, J. D., Design techniques and applications of cyberphysical systems: A survey. *IEEE Systems Journal* **2015,** *9* (2), 350-365.

36. Rajkumar, R. R.; Lee, I.; Sha, L.; Stankovic, J. In *Cyber-physical systems: the next computing revolution*, Proceedings of the 47th Design Automation Conference, ACM: 2010; pp 731-736.

37. Wang, S.; Wan, J.; Zhang, D.; Li, D.; Zhang, C., Towards smart factory for Industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. *Computer Networks* **2016,** *101*, 158-168.

38. Wang, S.; Zhang, Y.; Yang, Z.; Chen, Y. In *A Graphical Hierarchical CPS Architecture*, System and Software Reliability (ISSSR), International Symposium on, IEEE: 2016; pp 97-105.

39. Brusaferri, A.; Ballarino, A.; Cavadini, F. A.; Manzocchi, D.; Mazzolini, M. In *CPS-based hierarchical and self-similar automation architecture for the control and verification of reconfigurable manufacturing systems*, Emerging Technology and Factory Automation (ETFA), 2014 IEEE, IEEE: 2014; pp 1-8.

40. Santos, T.; Ribeiro, L.; Rocha, A. D.; Barata, J. In *A system reconfiguration architecture for hybrid automation systems based in agents and programmable logic controllers*, Industrial Informatics (INDIN), 2016 IEEE 14th International Conference on, IEEE: 2016; pp 98-103.

41. Sanislav, T.; Miclea, L., Cyber-physical systems-concept, challenges and research areas. *Journal of Control Engineering and Applied Informatics* **2012,** *14* (2), 28-33.

42. Broy, M.; Cengarle, M. V.; Geisberger, E. In *Cyber-physical systems: imminent challenges*, Monterey workshop, Springer: 2012; pp 1-28.

43. Subhani, S.; Shi, H.; Cobben, J. In *A survey of technical challenges in wireless machine-to-machine communication for smart grids*, Power Engineering Conference (UPEC), 2015 50th International Universities, IEEE: 2015; pp 1-6.

44. Esterle, L.; Grosu, R., Cyber-physical systems: challenge of the 21st century. *e & i Elektrotechnik und Informationstechnik* **2016,** *133* (7), 299-303.

45. Broman, D.; Derler, P.; Eidson, J., Temporal issues in cyber-physical systems. *Journal of the Indian Institute of Science* **2013,** *93* (3), 389-402.

46. Krupitzer, C.; Roth, F. M.; VanSyckel, S.; Schiele, G.; Becker, C., A survey on engineering approaches for self-adaptive systems. *Pervasive and Mobile Computing* **2015,** *17*, 184-206.

47. Yu, X.; Xue, Y., Smart Grids: A Cyber-Physical Systems Perspective. *Proceedings of the Ieee* **2016,** *104* (5), 1058-1070.

48. Novoa-Hernández, P.; Corona, C. C.; Pelta, D. A., Self-adaptation in dynamic environments-a survey and open issues. *International Journal of Bio-Inspired Computation* **2016,** *8* (1), 1-13.

49. Nelaturi, S.; de Kleer, J.; Shapiro, V.; Ieee, Combinatorial Models for Heterogeneous System Composition and Analysis. *2016 11th Systems of System Engineering Conference (Sose), Ieee* **2016**.

50. Gössler, G.; Sifakis, J., Composition for component-based modeling. *Science of Computer Programming* **2005,** *55* (1-3), 161-183.

51. Sztipanovits, J.; Koutsoukos, X.; Karsai, G.; Kottenstette, N.; Antsaklis, P.; Gupta, V.; Goodwine, B.; Baras, J.; Wang, S., Toward a science of cyber–physical system integration. *Proceedings of the IEEE* **2011,** *100* (1), 29-44.

52. Nuzzo, P.; Sangiovanni-Vincentelli, A. L.; Bresolin, D.; Geretti, L.; Villa, T., A platform-based design methodology with contracts and related tools for the design of cyber-physical systems. *Proceedings of the IEEE* **2015,** *103* (11), 2104-2132.

53. Attie, P.; Baranov, E.; Bliudze, S.; Jaber, M.; Sifakis, J., A general framework for architecture composability. *Formal Aspects of Computing* **2016,** *28* (2), 207-231.

54. Seshia, S. A., Combining induction, deduction, and structure for verification and synthesis. *Proceedings of the IEEE* **2015,** *103* (11), 2036-2051.

55. Tripakis, S., Compositionality in the science of system design. *Proceedings of the IEEE* **2016,** *104* (5), 960-972.

56. Chen, T.; Chilton, C.; Jonsson, B.; Kwiatkowska, M. In *A compositional specification theory for component behaviours*, European Symposium on Programming, Springer: 2012; pp 148-168.

57. Lien, S.-Y., Resource-optimal heterogeneous machine-to-machine communications in software defined networking cyber-physical systems. *Wireless Personal Communications* **2015,** *84* (3), 2215-2239.

58. Wan, J.; Tang, S.; Shu, Z.; Li, D.; Wang, S.; Imran, M.; Vasilakos, A. V., Software-Defined Industrial Internet of Things in the Context of Industry 4.0. *Ieee Sensors Journal* **2016,** *16* (20), 7373-7380.

59. Lora, M.; Muradore, R.; Quaglia, D.; Fummi, F., Simulation alternatives for the verification of networked cyber-physical systems. *Microprocessors and Microsystems* **2015,** *39* (8), 843-853.

60. Tan, Y.; Goddard, S.; Perez, L. C., A prototype architecture for cyber-physical systems. *ACM Sigbed Review* **2008,** *5* (1), 26.

61. Jara, A. J.; Genoud, D.; Bocchi, Y. In *Big data for cyber physical systems: An analysis of challenges, solutions and opportunities*, Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2014 Eighth International Conference on, IEEE: 2014; pp 376-380.

62. Rusák, Z.; Horváth, I.; Hou, Y.; Ji, L. In *Towards affordance based human-system interaction based on cyber-physical systems*, GI-Jahrestagung, 2014; pp 2007-2012.

63. Sowe, S. K.; Simmon, E.; Zettsu, K.; de Vaulx, F.; Bojanova, I., Cyber-Physical-Human Systems: Putting People in the Loop. *IT Professional* **2016,** *18* (1), 10-13.

64. Khalid, A.; Kirisci, P.; Ghrairi, Z.; Pannek, J.; Thoben, K.-D., Safety Requirements in Collaborative Human–Robot Cyber-Physical System. In *Dynamics in Logistics*, Springer: 2017; pp 41-51.

65. Alegre, U.; Augusto, J. C.; Clark, T., Engineering context-aware systems and applications: A survey. *Journal of Systems and Software* **2016,** *117*, 55-83.

66. Psaier, H.; Dustdar, S., A survey on self-healing systems: approaches and systems. *Computing* **2011,** *91* (1), 43-73.

67. Avižienis, A.; Laprie, J.-C.; Randell, B., Dependability and its threats: a taxonomy. In *Building the Information Society*, Springer: 2004; pp 91-120.

68. Krishna, C., Fault-tolerant scheduling in homogeneous real-time systems. *ACM Computing Surveys (CSUR)* **2014,** *46* (4), 48.

69. Moreno, G. A.; Cámara, J.; Garlan, D.; Schmerl, B. In *Proactive self-adaptation under uncertainty: A probabilistic model checking approach*, Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, ACM: 2015; pp 1-12.

70. Sha, L., Using simplicity to control complexity. *IEEE Software* **2001,** *18* (4), 20-28.

71. Ohmann, P.; Brown, D. B.; Neelakandan, N.; Linderoth, J.; Liblit, B., Optimizing Customized Program Coverage. In *2016 31st Ieee/Acm International Conference on Automated Software Engineering*, Lo, D.; Apel, S.; Khurshid, S., Eds. 2016; pp 27-38.

72. Turing, A. M., On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London mathematical society* **1937,** *2* (1), 230-265.

73. Sanislav, T.; Mois, G.; Miclea, L. In *A new approach towards increasing cyber-physical systems dependability*, Carpathian Control Conference (ICCC), 2015 16th International, IEEE: 2015; pp 443-447.

74. Müller, S. M.; Paul, W. J., *Computer architecture: complexity and correctness*. Springer Science & Business Media: 2013.

75. Jóźwiak, L., Heterogeneous MPSoC technology for modern cyber-physical systems. *Electronic Components and Materials* **2014,** *44* (4), 264-279.

76. Bhave, A.; Krogh, B. H.; Garlan, D.; Schmerl, B.; Ieee, View Consistency in Architectures for Cyber-Physical Systems. In *2011 ACM/IEEE Second International Conference on Cyber-Physical Systems*, 2011; pp 151-160.

77. Zheng, X.; Julien, C.; Kim, M.; Khurshid, S., On the State of the Art in Verification and Validation in Cyber Physical Systems. **2014**.

78. Zheng, X.; Julien, C. In *Verification and validation in cyber physical systems: research challenges and a way forward*, Proceedings of the First International Workshop on Software Engineering for Smart Cyber-Physical Systems, IEEE Press: 2015; pp 15-18.

79. Zheng, X.; Julien, C.; Kim, M.; Khurshid, S., Perceptions on the state of the art in verification and validation in cyber-physical systems. *IEEE Systems Journal* **2015**.

80. Asadollah, S. A.; Inam, R.; Hansson, H. In *A Survey on Testing for Cyber Physical System*, IFIP International Conference on Testing Software and Systems, Springer: 2015; pp 194-207.

81. Preden, J. S.; Tammemäe, K.; Jantsch, A.; Leier, M.; Riid, A.; Calis, E., The benefits of self-awareness and attention in fog and mist computing. *Computer* **2015,** *48* (7), 37-45.

82. Castro, M.; Jara, A. J.; Skarmeta, A. F., Enabling end-to-end CoAP-based communications for the Web of Things. *Journal of Network and Computer Applications* **2016,** *59*, 230-236.

83. Iarovyi, S.; Mohammed, W. M.; Lobov, A.; Ferrer, B. R.; Lastra, J. L. M., Cyber-Physical Systems for Open-Knowledge-Driven Manufacturing Execution Systems. *Proceedings of the Ieee* **2016,** *104* (5), 1142-1154.

84. Chen, Y.; Zhou, J.; Guo, M., A context-aware search system for Internet of Things based on hierarchical context model. *Telecommunication Systems* **2016,** *62* (1), 77-91.

85. Wang, H.; Li, Q.; Yi, F.; Li, Z.; Sun, L., Influential spatial facility prediction over large scale cyber-physical vehicles in smart city. *EURASIP Journal on Wireless Communications and Networking* **2016,** *2016* (1), 103.

86. Mi, M.; Zolotov, I. In *Comparison between multi-class classifiers and deep learning with focus on industry 4.0*, 2016 Cybernetics & Informatics (K&I), IEEE: 2016; pp 1-5.

87. Varshney, K. R.; Alemzadeh, H., On the Safety of Machine Learning: Cyber-Physical Systems, Decision Sciences, and Data Products. *arXiv preprint arXiv:1610.01256* **2016**.

88. Maier, A. In *Online passive learning of timed automata for cyber-physical production systems*, Industrial Informatics (INDIN), 2014 12th IEEE International Conference on, IEEE: 2014; pp 60-66.

89. Seshia, S. A.; Hu, S.; Li, W.; Zhu, Q., Design Automation of Cyber-Physical Systems: Challenges, Advances, and Opportunities. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **2016**.

90. Amodei, D.; Olah, C.; Steinhardt, J.; Christiano, P.; Schulman, J.; Mané, D., Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565* **2016**.

91. Seshia, S. A.; Sadigh, D.; Sastry, S. S., Towards verified artificial intelligence. *arXiv preprint arXiv:1606.08514* **2016**.

92. Tritchkov, I.; Goetz, H. In *Verification and Validation of Decentralized, Self-Organizing Cyber-Physical Production Systems*, Foundations and Applications of Self* Systems, IEEE International Workshops on, IEEE: 2016; pp 112-117.

93. Tősér, Z.; Lőrincz, A. In *The cyber-physical system approach towards artificial general intelligence: the problem of verification*, International Conference on Artificial General Intelligence, Springer: 2015; pp 373-383.

94. Bersani, M. M.; García-Valls, M. In *The cost of formal verification in adaptive CPS. An example of a virtualized server node*, High Assurance Systems Engineering (HASE), 2016 IEEE 17th International Symposium on, IEEE: 2016; pp 39-46.

95. Hachicha, M.; Dammak, E.; Halima, R. B.; Kacem, A. H. In *A correct by construction approach for modeling and formalizing self-adaptive systems*, Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2016 17th IEEE/ACIS International Conference on, IEEE: 2016; pp 379-384.

96. Arcaini, P.; Riccobene, E.; Scandurra, P., Modeling and validating self-adaptive service-oriented applications. *ACM SIGAPP Applied Computing Review* **2015,** *15* (3), 35-48.

97. Koh, Y.; Her, H.; Yi, K.; Kim, K., Integrated Speed and Steering Control Driver Model for Vehicle–Driver Closed-Loop Simulation. *IEEE Transactions on Vehicular Technology* **2016,** *65* (6), 4401-4411.

98. Shepilov, Y.; Pavlova, D.; Kazanskaia, D., Multithreading MAS Platform for Real-Time Scheduling. *International Journal of Software Innovation* **2016,** *4* (1), 48-60.

99. Bambagini, M.; Marinoni, M.; Aydin, H.; Buttazzo, G., Energy-Aware Scheduling for Real-Time Systems: A Survey. *Acm Transactions on Embedded Computing Systems* **2016,** *15* (1).

100. Baek, H.; Lee, J.; Lee, Y.; Yoon, H., Preemptive Real-Time Scheduling Incorporating Security Constraint for Cyber Physical Systems. *Ieice Transactions on Information and Systems* **2016,** *E99D* (8), 2121-2130.

101. Zhang, K.; Zhang, D.; de La Fortelle, A.; Wu, X.; Grégoire, J., State-driven priority scheduling mechanisms for driverless vehicles approaching intersections. *IEEE Transactions on Intelligent Transportation Systems* **2015,** *16* (5), 2487-2500.

102. Tan, L.; Du, C.; Dong, Y., Control-Performance-Driven Period and Deadline Selection for Cyber-Physical Systems. *2015 10th Asian Control Conference (Ascc)* **2015**.

103. Wu, S.-y.; Zhang, P.; Li, F.; Gu, F.; Pan, Y., A hybrid discrete particle swarm optimization-genetic algorithm for multi-task scheduling problem in service oriented manufacturing systems. *Journal of Central South University* **2016,** *23*, 421-429.

104. Lv, M.; Guan, N.; Zhang, Y.; Deng, Q.; Yu, G.; Zhang, J.; Ieee Computer, S. O. C., *A Survey of WCET Analysis of Real-Time Operating Systems*. 2009; p 65-+.

105. Baruah, S.; Ieee, Schedulability analysis for a general model of mixed-criticality recurrent real-time tasks. *Proceedings of 2016 Ieee Real-Time Systems Symposium (Rtss)* **2016**, 25-34.

106. Chadli, M.; Kim, J. H.; Legay, A.; Traonouez, L.-M.; Naujokat, S.; Steffen, B.; Larsen, K. G., A Model-Based Framework for the Specification and Analysis of Hierarchical Scheduling Systems. In *Critical Systems: Formal Methods and Automated Verification*, TerBeek, M. H.; Gnesi, S.; Knapp, A., Eds. 2016; Vol. 9933, pp 133-141.

107. Qian, Z.; Yu, H. In *A TAOPN approach to modeling and scheduling cyber-physical systems*, Information Science and Applications (ICISA), 2013 International Conference on, IEEE: 2013; pp 1-7.

108. Jiang, J.-M.; Zhu, H.; Li, Q.; Zhao, Y.; Zhao, L.; Zhang, S.; Gong, P.; Hong, Z., Analyzing Event-Based Scheduling in Concurrent Reactive Systems. *ACM Transactions on Embedded Computing Systems (TECS)* **2015,** *14* (4), 86.

109. Lee, E. A., Computing needs time. *Communications of the ACM* **2009,** *52* (5), 70-79.

110. Patel, H. D.; Lickly, B.; Burgers, B.; Lee, E. A. *A timing requirements-aware scratchpad memory allocation scheme for a precision timed architecture*; DTIC Document: 2008.

111. Zou, J.; Matic, S.; Lee, E. A. In *PtidyOS: A lightweight microkernel for Ptides real-time systems*, Real-Time and Embedded Technology and Applications Symposium (RTAS), 2012 IEEE 18th, IEEE: 2012; pp 209-218.

112. Lickly, B.; Liu, I.; Kim, S.; Patel, H. D.; Edwards, S. A.; Lee, E. A. In *Predictable programming on a precision timed architecture*, Proceedings of the 2008 international conference on Compilers, architectures and synthesis for embedded systems, ACM: 2008; pp 137-146.

113. Elattar, M.; Dürkop, L.; Jasperneite, J. In *Utilizing LTE QoS features to provide a reliable access network for cyber-physical systems*, Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on, IEEE: 2015; pp 956-961.

114. Aristova, N. I., Ethernet in industrial automation: Overcoming obstacles. *Automation and Remote Control* **2016,** *77* (5), 881-894.

115. Watteyne, T.; Handziski, V.; Vilajosana, X.; Duquennoy, S.; Hahm, O.; Baccelli, E.; Wolisz, A., Industrial wireless ip-based cyber–physical systems. *Proceedings of the IEEE* **2016,** *104* (5), 1025-1038.

116. Cardoso, J.; Derler, P.; Eidson, J. C.; Lee, E. A. In *Network latency and packet delay variation in cyber-physical systems*, Network Science Workshop (NSW), 2011 IEEE, IEEE: 2011; pp 51-58.

117. Eidson, J.; Lee, E. A.; Matic, S.; Seshia, S. A.; Zou, J. In *A time-centric model for cyber-physical applications*, Workshop on Model Based Architecting and Construction of Embedded Systems (ACES-MB), 2010; pp 21-35.

118. Derler, P.; Eidson, J. C.; Goose, S.; Lee, E. A.; Matic, S.; Zimmer, M. In *Using ptides and synchronized clocks to design distributed systems with deterministic system wide timing*, 2013 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS) Proceedings, IEEE: 2013; pp 41-46.

119. Derler, P.; Lee, E. A.; Zimmer, M., Logically Synchronous Models of Distributed Systems with Explicit Timing Specifications.

120. Donkers, M.; Heemels, W.; Bernardini, D.; Bemporad, A.; Shneer, V., Stability analysis of stochastic networked control systems. *Automatica* **2012,** *48* (5), 917-925.

121. Balasubramaniyan, S.; Srinivasan, S.; Buonopane, F.; Subathra, B.; Vain, J.; Ramaswamy, S., Design and verification of Cyber-Physical Systems using TrueTime, evolutionary optimization and UPPAAL. *Microprocessors and Microsystems* **2016,** *42*, 37-48.

122. Lawrence, D.; Buchs, D.; Wellig, A. In *Using instrumentation for quality assessment of resilient software in embedded systems*, International Workshop on Software Engineering for Resilient Systems, Springer: 2014; pp 139-153.

123. Lefebvre, D., Fault diagnosis and prognosis with partially observed Petri nets. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2014,** *44* (10), 1413-1424.

124. Souali, K.; Rahmaoui, O.; Ouzzif, M. In *An overview of traceability: Definitions and techniques*, Information Science and Technology (CiSt), 2016 4th IEEE International Colloquium on, IEEE: 2016; pp 789-793.

125. Winkler, S.; Pilgrim, J., A survey of traceability in requirements engineering and model-driven development. *Software and Systems Modeling (SoSyM)* **2010,** *9* (4), 529-565.

126. Hammad, M.; Collard, M. L.; Maletic, J. I., Automatically identifying changes that impact code-to-design traceability during evolution. *Software Quality Journal* **2011,** *19* (1), 35-64.

127. Santiago, I.; Vara, J. M.; de Castro, M. V.; Marcos, E. In *Towards the effective use of traceability in model-driven engineering projects*, International Conference on Conceptual Modeling, Springer: 2013; pp 429-437.

128. Bordel Sánchez, B.; Alcarria, R.; Martín, D.; Robles, T., TF4SM: A Framework for Developing Traceability Solutions in Small Manufacturing Companies. *Sensors* **2015,** *15* (11), 29478-29510.

129. Lee, E. A., The Past, Present and Future of Cyber-Physical Systems: A Focus on Models. *Sensors* **2015,** *15* (3), 4837-4869.

130. Warriach, E. U.; Ozcelebi, T.; Lukkien, J. J. In *Self-* Properties in Smart Environments: Requirements and Performance Metrics*, Intelligent Environments (Workshops), 2014; pp 194-205.

131. Ye, L.; Qian, K.; Zhang, L., S-PDH: A CPS Service Contract Framework for Composition. In *Service-Oriented Computing - Icsoc 2015 Workshops*, Norta, A.; Gaaloul, W.; Gangadharan, G. R.; Dam, H. K., Eds. 2016; Vol. 9586, pp 79-90.

132. Brizzi, P.; Conzon, D.; Khaleel, H.; Tomasi, R.; Pastrone, C.; Spirito, A.; Knechtel, M.; Pramudianto, F.; Cultrona, P. In *Bringing the Internet of Things along the manufacturing line: A case study in controlling industrial robot and monitoring energy consumption remotely*, Emerging Technologies & Factory Automation (ETFA), 2013 IEEE 18th Conference on, IEEE: 2013; pp 1-8.

133. Wang, L.; Haghighi, A., Combined strength of holons, agents and function blocks in cyber-physical systems. *Journal of Manufacturing Systems* **2016,** *40*, 25-34.

134. Shu, Z.; Wan, J.; Zhang, D.; Li, D., Cloud-Integrated Cyber-Physical Systems for Complex Industrial Applications. *Mobile Networks & Applications* **2016,** *21* (5), 865-878.

135. Perez, H.; Javier Gutierrez, J.; Peiro, S.; Crespo, A., Distributed architecture for developing mixed-criticality systems in multi-core platforms. *Journal of Systems and Software* **2017,** *123*, 145-159.

136. Jablkowski, B.; Gabor, U. T.; Spinczyk, O., Evolutionary planning of virtualized cyber-physical compute and control clusters. *Journal of Systems Architecture* **2017,** *73*, 17-27.

137. Bruton, K.; Walsh, B. P.; Cusack, D. O.; O'Donovan, P.; O'Sullivan, D. T. J., Enabling Effective Operational Decision Making on a Combined Heat and Power System using the 5C Architecture. In *5th Cirp Global Web Conference - Research and Innovation for Future Production*, Alexopoulos, K., Ed. 2016; Vol. 55, pp 296-301.

138. Camara, J.; Correia, P.; de Lemos, R.; Garlan, D.; Gomes, P.; Schmerl, B.; Ventura, R., Incorporating architecture-based self-adaptation into an adaptive industrial software system. *Journal of Systems and Software* **2016,** *122*, 507-523.

139. Cámara, J.; de Lemos, R.; Laranjeiro, N.; Ventura, R.; Vieira, M. In *Robustness evaluation of the rainbow framework for self-adaptation*, Proceedings of the 29th Annual ACM Symposium on Applied Computing, ACM: 2014; pp 376-383.

140. Masrur, A.; Kit, M.; Matěna, V.; Bureš, T.; Hardt, W., Component-based design of cyber-physical applications with safety-critical requirements. *Microprocessors and Microsystems* **2016,** *42*, 70-86.

141. Gerostathopoulos, I.; Bures, T.; Hnetynka, P.; Keznikl, J.; Kit, M.; Plasil, F.; Plouzeau, N., Self-adaptation in software-intensive cyber-physical systems: From system goals to architecture configurations. *Journal of Systems and Software* **2016,** *122*, 378-397.

142. Park, J.; Lee, S.; Yoon, T.; Kim, J. M., An autonomic control system for high-reliable CPS. *Cluster Computing* **2015,** *18* (2), 587-598.

143. Axelsson, J.; Kobetski, A. In *Architectural Concepts for Federated Embedded Systems*, Proceedings of the 2014 European Conference on Software Architecture Workshops, ACM: 2014; p 25.

144. Gang, L.; GuPing, Z. In *Self-Reconfiguration Architecture for Distribution Automation System Based on Cyber-Physical*, Power and Energy Engineering Conference (APPEEC), 2012 Asia-Pacific, IEEE: 2012; pp 1-4.

145. Pajic, M.; Chernoguzov, A.; Mangharam, R., Robust Architectures for Embedded Wireless Network Control and Actuation. *Acm Transactions on Embedded Computing Systems* **2012,** *11* (4).

146. Wu, J.; Huang, Y.; Kong, J.; Tang, Q.; Huang, X. In *A study on the dependability of software defined networks*, 2015 International Conference on Materials Engineering and Information Technology Applications (MEITA 2015), 2015.

147. Lira, D. N.; Borsato, M., Dependability Modeling for the Failure Prognostics in Smart Manufacturing. In *Transdisciplinary Engineering: Crossing Boundaries*, Borsato, M.; Wognum, N.; Peruzzini, M.; Stjepandic, J.; Verhagen, W. J. C., Eds. 2016; Vol. 4, pp 885-894.

148. Cooray, D.; Kouroshfar, E.; Malek, S.; Roshandel, R., Proactive self-adaptation for improving the reliability of mission-critical, embedded, and mobile software. *IEEE Transactions on Software Engineering* **2013,** *39* (12), 1714-1735.

149. Kapitanova, K.; Hoque, E.; Stankovic, J. A.; Whitehouse, K.; Son, S. H. In *Being SMART about failures: assessing repairs in SMART homes*, Proceedings of the 2012 ACM Conference on Ubiquitous Computing, ACM: 2012; pp 51-60.

150. Tipaldi, M.; Bruenjes, B., Survey on Fault Detection, Isolation, and Recovery Strategies in the Space Domain. *Journal of Aerospace Information Systems* **2015,** *12* (2), 235-256.

151. Gao, Z.; Cecati, C.; Ding, S. X., A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches. *IEEE Transactions on Industrial Electronics* **2015,** *62* (6), 3757-3767.

152. Mittal, S.; Vetter, J. S., A survey of techniques for modeling and improving reliability of computing systems. *IEEE Transactions on Parallel and Distributed Systems* **2016,** *27* (4), 1226-1238.

153. Muhammed, T.; Shaikh, R. A., An analysis of fault detection strategies in wireless sensor networks. *Journal of Network and Computer Applications* **2017,** *78*, 267-287.

154. Chouikhi, S.; El Korbi, I.; Ghamri-Doudane, Y.; Saidane, L. A., A survey on fault tolerance in small and large scale wireless sensor networks. *Computer Communications* **2015,** *69*, 22-37.

155. Isermann, R., *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media: 2006.

156. Stapelberg, R. F., *Handbook of reliability, availability, maintainability and safety in engineering design*. Springer Science & Business Media: 2009.

157. Birolini, A., Reliability engineering: theory and practice. 7th(2014) ed.; Springer, New York, NY: 2002.

158. Hasan, O.; Ahmed, W.; Tahar, S.; Hamdi, M. S.; Simos, T. E.; Tsitouras, C. In *Reliability block diagrams based analysis: A survey*, AIP Conference Proceedings, AIP Publishing: 2015; p 850129.

159. Ruijters, E.; Stoelinga, M., Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools. *Computer Science Review* **2015,** *15-16*, 29-62.

160. Kabir, S., An overview of fault tree analysis and its application in model based dependability analysis. *Expert Systems with Applications* **2017,** *77*, 114-135.

161. Höfig, K.; Domis, D. In *Failure-dependent execution time analysis*, Proceedings of the joint ACM SIGSOFT conference--QoSA and ACM SIGSOFT symposium--ISARCS on Quality of software architectures--QoSA and architecting critical systems--ISARCS, ACM: 2011; pp 115-122.

162. Ahmed, W.; Hasan, O.; Tahar, S. In *Formal dependability modeling and analysis: a survey*, International Conference on Intelligent Computer Mathematics, Springer: 2016; pp 132-147.

163. Alexandrescu, D.; Sterpone, L.; Lopez-Ongil, C.; Ieee, Fault Injection and Fault Tolerance Methodologies for Assessing Device Robustness and Mitigating against Ionizing Radiation. In *2014 19th Ieee European Test Symposium*, 2014.

164. Natella, R.; Cotroneo, D.; Madeira, H. S., Assessing Dependability with Software Fault Injection: A Survey. *Acm Computing Surveys* **2015,** *48* (3).

165. Kooli, M.; Di Natale, G.; Ieee, A Survey on Simulation-Based Fault Injection Tools for Complex Systems. *2014 9th Ieee International Conference on Design & Technology of Integrated Systems in Nanoscale Era (Dtis 2014)* **2014**.

166. Benso, A.; Prinetto, P., *Fault injection techniques and tools for embedded systems reliability evaluation*. Springer Science & Business Media: 2003; Vol. 23.

167. Tsigkanos, C.; Kehrer, T.; Ghezzi, C., Architecting dynamic cyber-physical spaces. *Computing* **2016,** *98* (10), 1011-1040.

168. Yang, W.; Liu, Y.; Xu, C.; Cheung, S.-C., A survey on dependability improvement techniques for pervasive computing systems. *Science China Information Sciences* **2015,** *58* (5), 1-14.

169. Cecati, C., A survey of fault diagnosis and fault-tolerant techniques—part II: fault diagnosis with knowledge-based and hybrid/active approaches. *IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS* **2015**.

170. Li, F.; Shi, P.; Wang, X.; Agarwal, R., Fault detection for networked control systems with quantization and Markovian packet dropouts. *Signal Processing* **2015,** *111*, 106-112.

171. Xia, J.; Jiang, B.; Zhang, K.; Xu, J., Robust Fault Diagnosis Design for Linear Multiagent Systems with Incipient Faults. *Mathematical Problems in Engineering* **2015**.

172. Esfahani, P. M.; Lygeros, J., A tractable fault detection and isolation approach for nonlinear systems with probabilistic performance. *IEEE Transactions on Automatic Control* **2016,** *61* (3), 633-647.

173. Reppa, V.; Polycarpou, M. M.; Panayiotou, C. G., Distributed Sensor Fault Diagnosis for a Network of Interconnected Cyber-Physical Systems. *Ieee Transactions on Control of Network Systems* **2015,** *2* (1), 11-23.

174. Dong, H.; Wang, Z.; Ding, S. X.; Gao, H., A Survey on Distributed Filtering and Fault Detection for Sensor Networks. *Mathematical Problems in Engineering* **2014**.

175. Warriach, E. U.; Tei, K. In *Fault detection in wireless sensor networks: A machine learning approach*, Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference on, IEEE: 2013; pp 758-765.

176. Krishnamurthy, S.; Sarkar, S.; Tewari, A. In *Scalable anomaly detection and isolation in cyber-physical systems using bayesian networks*, ASME 2014 Dynamic Systems and Control Conference, American Society of Mechanical Engineers: 2014; pp V002T26A006-V002T26A006.

177. Sanislav, T.; Merza, K.; Mois, G.; Miclea, L. In *Cyber-physical system dependability enhancement through data mining*, Automation, Quality and Testing, Robotics (AQTR), 2016 IEEE International Conference on, IEEE: 2016; pp 1-5.

178. Khalid, Z.; Fisal, N.; Rozaini, M., A Survey of Middleware for Sensor and Network Virtualization. *Sensors* **2014,** *14* (12), 24046-24097.

179. Ahn, S.; Yoo, C.; Lee, S.; Lee, H.; Kim, S. J., Implementing virtual platform for global-scale cyber physical system networks. *International Journal of Communication Systems* **2015,** *28* (13), 1899-1920.

180. Khan, I.; Belqasmi, F.; Glitho, R.; Crespi, N.; Morrow, M.; Polakos, P., Wireless Sensor Network Virtualization: A Survey. *Ieee Communications Surveys and Tutorials* **2016,** *18* (1), 553-576.

181. Dettoni, F.; Lung, L. C.; Luiz, A. F., Using Virtualization Technology for Fault-Tolerant Replication in LAN. In *New Results in Dependability and Computer Systems*, Zamojski, W.; Mazurkiewicz, J.; Sugier, J.; Walkowiak, T.; Kacprzyk, J., Eds. 2013; Vol. 224, pp 131-140.

182. Quattrociocchi, W.; Caldarelli, G.; Scala, A., Self-Healing Networks: Redundancy and Structure. *Plos One* **2014,** *9* (2).

183. Rodriguez, M.; Kwiat, K. A.; Kamhoua, C. A. In *On the use of design diversity in fault tolerant and secure systems: A qualitative analysis*, Computational Intelligence for Security and Defense Applications (CISDA), 2015 IEEE Symposium on, IEEE: 2015; pp 1-8.

184. Hazra, A.; Dasgupta, P.; Chakrabarti, P. P., Formal assessment of reliability specifications in embedded cyber-physical systems. *Journal of Applied Logic* **2016,** *18*, 71-104.

185. Warriach, E. U.; Ozcelebi, T.; Lukkien, J. J. In *Fault-prevention in smart environments for dependable applications*, Self-Adaptive and Self-Organizing Systems (SASO), 2014 IEEE Eighth International Conference on, IEEE: 2014; pp 183-184.

186. Zheng, J.; Wang, P.; Li, C.; Mouftah, H. T. In *An Efficient Fault-Prevention Clustering Protocol for Robust Underwater Sensor Networks*, Communications, 2008. ICC'08. IEEE International Conference on, IEEE: 2008; pp 2802-2807.

187. Yano, I. H.; Oliveira, V. C.; de Mello Fagotto, E. A.; Mota, A. D. A.; Mota, L. T. M., Predicting battery charge depletion in wireless sensor networks using received signal strength indicator. *Journal of Computer Science* **2013,** *9* (7), 821-826.

188. Warriach, E. U.; Ozcelebi, T.; Lukkien, J. J. In *A Comparison of Predictive Algorithms for Failure Prevention in Smart Environment Applications*, Intelligent Environments (IE), 2015 International Conference on, IEEE: 2015; pp 33-40.

189. Lakner, R.; Németh, E.; Hangos, K. M.; Cameron, I. T. In *Multiagent realization of prediction-based diagnosis and loss prevention*, International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Springer: 2006; pp 70-80.

190. Németh, E.; Lakner, R.; Hangos, K. M.; Cameron, I. T., Prediction-based diagnosis and loss prevention using qualitative multi-scale models. *Information Sciences* **2007,** *177* (8), 1916-1930.

191. Salfner, F.; Lenk, M.; Malek, M., A survey of online failure prediction methods. *ACM Computing Surveys (CSUR)* **2010,** *42* (3), 10.

192. Ruiz-Arenas, S.; Horváth, I.; Mejía-Gutiérrez, R.; Opiyo, E., Towards the maintenance principles of cyber-physical systems. *Strojniški vestnik-Journal of Mechanical Engineering* **2014,** *60* (12), 815-831.

193. Kans, M.; Galar, D.; Thaduri, A., Maintenance 4.0 in Railway Transportation Industry. In *Proceedings of the 10th World Congress on Engineering Asset Management*, Koskinen, K. T.; Kortelainen, H.; Aaltonen, J.; Uusitalo, T.; Komonen, K.; Mathew, J.; Laitinen, J., Eds. 2016; pp 317-331.

194. Verma, A. K.; Srividya, A.; Ramesh, P., A systemic approach to integrated E-maintenance of large engineering plants. *International Journal of Automation and Computing* **2010,** *7* (2), 173-179.

195. Weber, P.; Medina-Oliva, G.; Simon, C.; Iung, B., Overview on Bayesian networks applications for dependability, risk analysis and maintenance areas. *Engineering Applications of Artificial Intelligence* **2012,** *25* (4), 671-682.

196. Sleptchenko, A.; Johnson, M. E., Maintaining Secure and Reliable Distributed Control Systems. *INFORMS Journal on Computing* **2014,** *27* (1), 103-117.

197. Heegaard, P. E.; Helvik, B. E.; Nencioni, G.; Wäfler, J., Managed dependability in interacting systems. In *Principles of Performance and Reliability Modeling and Evaluation*, Springer: 2016; pp 197-226.

198. Edifor, E.; Walker, M.; Gordon, N., Quantification of Simultaneous-AND gates in temporal fault trees. In *New Results in Dependability and Computer Systems*, Springer: 2013; pp 141-151.

199. Chiacchio, F.; D'Urso, D.; Compagno, L.; Pennisi, M.; Pappalardo, F.; Manno, G., SHyFTA, a Stochastic Hybrid Fault Tree Automaton for the modelling and simulation of dynamic reliability problems. *Expert Systems with Applications* **2016,** *47*, 42-57.

200. Ma, Z.; Fu, X.; Yu, Z. In *Object-oriented petri nets based formal modeling for high-confidence cyber-physical systems*, Wireless Communications, Networking and Mobile Computing (WiCOM), 2012 8th International Conference on, IEEE: 2012; pp 1-4.

201. Cho, C.-S.; Chung, W.-H.; Kuo, S.-Y., Cyberphysical security and dependability analysis of digital control systems in nuclear power plants. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2016,** *46* (3), 356-369.

202. Aissa, A. B.; Abercrombie, R. K.; Sheldon, F. T.; Mili, A. In *Quantifying the impact of unavailability in cyber-physical environments*, Computational Intelligence in Cyber Security (CICS), 2014 IEEE Symposium on, IEEE: 2014; pp 1-8.

203. Kozat, U. C.; Liang, G.; Kokten, K.; Tapolcai, J., On Optimal Topology Verification and Failure Localization for Software Defined Networks. *Ieee-Acm Transactions on Networking* **2016,** *24* (5), 2931-2944.

204. Rungger, M.; Tabuada, P. In *A symbolic approach to the design of robust cyber-physical systems*, Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on, IEEE: 2013; pp 3932-3937.

205. Takai, S.; Kumar, R. In *Verification of generalized inference diagnosability for decentralized diagnosis in discrete event systems*, Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on, IEEE: 2015; pp 1-8.

206. Ganin, A. A.; Massaro, E.; Gutfraind, A.; Steen, N.; Keisler, J. M.; Kott, A.; Mangoubi, R.; Linkov, I., Operational resilience: concepts, design and analysis. *Scientific reports* **2016,** *6*.

207. Laibinis, L.; Klionskiy, D.; Troubitsyna, E.; Dorokhov, A.; Lilius, J.; Kupriyanov, M., Modelling Resilience of Data Processing Capabilities of CPS. In *Software Engineering for Resilient Systems*, Majzik, I.; Vieira, M., Eds. 2014; Vol. 8785, pp 55-70.

208. Mashkoor, A.; Hasan, O. In *Formal probabilistic analysis of cyber-physical transportation systems*, International Conference on Computational Science and Its Applications, Springer: 2012; pp 419-434.

209. Saha, I.; Roy, S.; Ramesh, S., Formal Verification of Fault-Tolerant Startup Algorithms for Time-Triggered Architectures: A Survey. *Proceedings of the IEEE* **2016,** *104* (5), 904-922.

210. Han, Y.; Wen, Y.; Guo, C.; Huang, H., Incorporating Cyber Layer Failures in Composite Power System Reliability Evaluations. *Energies* **2015,** *8* (9), 9064-9086.

211. Harrison, R.; Vera, D.; Ahmad, B., Engineering methods and tools for cyber–physical automation systems. *Proceedings of the IEEE* **2016,** *104* (5), 973-985.

212. Davare, A.; Densmore, D.; Guo, L.; Passerone, R.; Sangiovanni-Vincentelli, A. L.; Simalatsar, A.; Zhu, Q., METROII: A Design Environment for Cyber-Physical Systems. *Acm Transactions on Embedded Computing Systems* **2013,** *12*.

213. Becker, M.; Kuznik, C.; Mueller, W. In *Virtual platforms for model-based design of dependable Cyber-Physical system software*, Digital System Design (DSD), 2014 17th Euromicro Conference on, IEEE: 2014; pp 246-253.

214. Ferracuti, F.; Freddi, A.; Monteriu, A.; Prist, M., An Integrated Simulation Module for Cyber-Physical Automation Systems. *Sensors* **2016,** *16* (5).

215. Francalanza, E.; Borg, J.; Constantinescu, C., A knowledge-based tool for designing cyber physical production systems. *Computers in Industry* **2017,** *84*, 39-58.

216. Bozzano, M.; Cimatti, A.; Katoen, J.-P.; Nguyen, V. Y.; Noll, T.; Roveri, M., Safety, dependability and performance analysis of extended AADL models. *The Computer Journal* **2011,** *54* (5), 754-775.

217. Bozzano, M.; Cimatti, A.; Pires, A. F.; Jones, D.; Kimberly, G.; Petri, T.; Robinson, R.; Tonetta, S., Formal Design and Safety Analysis of AIR6110 Wheel Brake System. In *Computer Aided Verification, Pt I*, Kroening, D.; Pasareanu, C. S., Eds. 2015; Vol. 9206, pp 518-535.

218. Whittle, J.; Hutchinson, J.; Rouncefield, M., The state of practice in model-driven engineering. *Software, IEEE* **2014,** *31* (3), 79-85.

219. Conzon, D.; Brizzi, P.; Kasinathan, P.; Pastrone, C.; Pramudianto, F.; Cultrona, P.; Ieee, Industrial application development exploiting IoT Vision and Model Driven programming. In *2015 8th International Conference on Intelligence in Next Generation Networks*, 2015; pp 168-175.

220. Broman, D.; Lee, E. A.; Tripakis, S.; Törngren, M. In *Viewpoints, formalisms, languages, and tools for cyber-physical systems*, Proceedings of the 6th International Workshop on Multi-Paradigm Modeling, ACM: 2012; pp 49-54.

221. Berry, G., Formally Unifying Modeling and Design for Embedded Systems - A Personal View. In *Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications, Isola 2016, Pt Ii*, Margaria, T.; Steffen, B., Eds. 2016; Vol. 9953, pp 134-149.

222. Jambli, M. N.; Lenando, H.; Zen, K.; Suhaili, S. M.; Tully, A.; Ieee, Simulation Tools for Mobile Ad-hoc Sensor Networks: A State-of-the-Art Survey. In *2012 International Conference on Advanced Computer Science Applications and Technologies*, 2012; pp 1-6.

223. Kang, S.; Aldwairi, M.; Kim, K.-I., A survey on network simulators in three-dimensional wireless ad hoc and sensor networks. *International Journal of Distributed Sensor Networks* **2016,** *12* (10).

224. 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. *Computers in Industry* **2016,** *82*, 273-289.

225. Kroiss, C.; Bures, T., Logic-based modeling of information transfer in cyber-physical multi-agent systems. *Future Generation Computer Systems-the International Journal of Escience* **2016,** *56*, 124-139.

226. Heimdahl, M. P.; Duan, L.; Murugesan, A.; Rayadurgam, S. In *Modeling and requirements on the physical side of cyber-physical systems*, Twin Peaks of Requirements and Architecture (TwinPeaks), 2013 2nd International Workshop on the, IEEE: 2013; pp 1-7.

227. Goncalves, F. S.; Becker, L. B.; Ieee, Model Driven Engineering Approach to Design Sensing and Actuation Subsystems. In *2016 Ieee 21st International Conference on Emerging Technologies and Factory Automation*, 2016.

228. Xi, Z.; Ieee, Physically Informed Assertions for Cyber Physical Systems Development and Debugging. *2014 Ieee International Conference on Pervasive Computing and Communications Workshops (Percom Workshops)* **2014**, 181-183.

229. Passarini, R. F.; Farines, J.-M.; Fernandes, J. M.; Becker, L. B., Cyber-physical systems design: transition from functional to architectural models. *Design Automation for Embedded Systems* **2015,** *19* (4), 345-366.

230. Rajhans, A.; Bhave, A.; Ruchkin, I.; Krogh, B. H.; Garlan, D.; Platzer, A.; Schmerl, B., Supporting Heterogeneity in Cyber-Physical Systems Architectures. *Ieee Transactions on Automatic Control* **2014,** *59* (12), 3178-3193.

231. Bures, T.; Hnetynka, P.; Kofron, J.; Al Ali, R.; Skoda, D. In *Statistical approach to architecture modes in smart cyber physical systems*, Software Architecture (WICSA), 2016 13th Working IEEE/IFIP Conference on, IEEE: 2016; pp 168-177.

232. Ballagny, C.; Hameurlain, N.; Barbier, F.; Ieee, MOCAS: a State-Based Component Model for Self-Adaptation. In *Saso: 2009 3rd Ieee International Conference on Self-Adaptive and Self-Organizing Systems*, 2009; pp 206-215.

233. Menczel, A.; Weiss, G.; Fried, D. In *On Modeling, Complexities, and Automatic Configuration of Wireless Industrial Control Networks*, Software Science, Technology and Engineering (SWSTE), 2016 IEEE International Conference on, IEEE: 2016; pp 125-134.

234. Lopes, F. A.; Lima, L.; Santos, M.; Fidalgo, R.; Fernandes, S., High-Level Modeling and Application Validation for SDN. In *Noms 2016 - 2016 Ieee/Ifip Network Operations and Management Symposium*, Badonnel, S. O.; Ulema, M.; Cavdar, C.; Granville, L. Z.; DosSantos, C. R. P., Eds. 2016; pp 197-205.

235. Rajhans, A.; Krogh, B. H. In *Heterogeneous verification of cyber-physical systems using behavior relations*, Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control, ACM: 2012; pp 35-44.

236. Clarke, E. M.; Zuliani, P., Statistical Model Checking for Cyber-Physical Systems. In *Automated Technology for Verification and Analysis*, Bultan, T.; Hsiung, P. A., Eds. 2011; Vol. 6996, pp 1-12.

237. Djoudi, B.; Bouanaka, C.; Zeghib, N., A formal framework for context-aware systems specification and verification. *Journal of Systems and Software* **2016,** *122*, 445-462.

238. Cheng, B.; Wang, X.; Liu, J.; Du, D., Modana: An Integrated Framework for Modeling and Analysis of Energy-Aware CPSs. In *39th Annual Ieee Computers, Software and Applications Conference*, Ahamed, S. I.; Chang, C. K.; Chu, W.; Crnkovic, I.; Hsiung, P. A.; Huang, G.; Yang, J. W., Eds. 2015; pp 127-136.

239. Anier, A.; Vain, J.; Tsiopoulos, L., DTRON: a tool for distributed model-based testing of time critical applications. *Proceedings of the Estonian Academy of Sciences* **2017,** *66* (1), 75-88.

240. Schenk, T.; Gilg, A. B.; Mühlbauer, M.; Rosen, R.; Wehrstedt, J. C., Architecture for modeling and simulation of technical systems along their lifecycle. *Computing and Visualization in Science* **2015,** *17* (4), 167-183.

241. Lee, K. H.; Hong, J. H.; Kim, T. G., System of systems approach to formal modeling of cps for simulation-based analysis. *ETRI Journal* **2015,** *37* (1), 175-185.

242. Hou, Y.; Zhao, Y.; Wagh, A.; Zhang, L.; Qiao, C.; Hulme, K. F.; Wu, C.; Sadek, A. W.; Liu, X., Simulation-Based Testing and Evaluation Tools for Transportation Cyber-Physical Systems. *Ieee Transactions on Vehicular Technology* **2016,** *65* (3), 1098-1108.

243. Brito, A. V.; Negreiros, A. V.; Roth, C.; Sander, O.; Becker, J.; Ieee, Development and Evaluation of Distributed Simulation of Embedded Systems using Ptolemy and HLA. In *17th Ieee/Acm International Symposium on Distributed Simulation and Real Time Applications*, 2013; pp 189-196.

244. Slåtten, V.; Herrmann, P.; Kraemer, F. A., Model-Driven Engineering of Reliable Fault-Tolerant Systems-A State-of-the-Art Survey. *Advances in Computers* **2013,** *91*, 119-205.

245. Solé, M.; Muntés-Mulero, V.; Rana, A. I.; Estrada, G., Survey on Models and Techniques for Root-Cause Analysis. *arXiv preprint arXiv:1701.08546* **2017**.

246. Esteve, M.-A.; Katoen, J.-P.; Viet Yen, N.; Postma, B.; Yushtein, Y., Formal Correctness, Safety, Dependability, and Performance Analysis of a Satellite. In *2012 34th International Conference on Software Engineering*, Glinz, M.; Murphy, G.; Pezze, M., Eds. 2012; pp 1022-1031.

247. Marashi, K.; Sarvestani, S. S. In *Towards comprehensive modeling of reliability for smart grids: Requirements and challenges*, High-Assurance Systems Engineering (HASE), 2014 IEEE 15th International Symposium on, IEEE: 2014; pp 105-112.

248. Hu, X.; Liu, S.; Chen, G.; Jiang, C. In *Dependability Modelling and Evaluation of Cyber-Physical Systems: A Model-Driven Perspective*, 1st International Workshop on Cloud Computing and Information Security, Atlantis Press: 2013.

249. Sanislav, T.; Mois, G.; Miclea, L., An approach to model dependability of cyber-physical systems. *Microprocessors and Microsystems* **2016,** *41*, 67-76.

250. Di Giandomenico, F.; Kwiatkowska, M.; Martinucci, M.; Masci, P.; Qu, H., Dependability Analysis and Verification for CONNECTed Systems. In *Leveraging Applications of Formal Methods, Verification, and Validation, Pt Ii*, Margaria, T.; Steffen, B., Eds. 2010; Vol. 6416, pp 263-+.

251. Dal Lago, L.; Ferrante, O.; Passerone, R.; Ferrari, A., Dependability Assessment of SOA-based CPS with Contracts and Model-Based Fault Injection. *IEEE Transactions on Industrial Informatics* **2017**.

252. Ghezzi, C., Dependability of Adaptable and Evolvable Distributed Systems. In *Formal Methods for the Quantitative Evaluation of Collective Adaptive Systems*, Springer: 2016; pp 36-60.

253. Hu, F.; Lu, Y.; Vasilakos, A. V.; Hao, Q.; Ma, R.; Patil, Y.; Zhang, T.; Lu, J.; Li, X.; Xiong, N. N., Robust cyber–physical systems: Concept, models, and implementation. *Future Generation Computer Systems* **2016,** *56*, 449-475.

254. Tundis, A.; Buffoni, L.; Fritzson, P.; Garro, A., Model-Based Dependability Analysis of Physical Systems with Modelica. *Modelling and Simulation in Engineering* **2017,** *2017*.

255. Provan, G., A Contracts-Based Framework for Systems Modeling and Embedded Diagnostics. In *Software Engineering and Formal Methods, Sefm 2014*, Canal, C.; Idani, A., Eds. 2015; Vol. 8938, pp 131-143.

256. Silva, L. C.; Almeida, H. O.; Perkusich, A.; Perkusich, M., A Model-Based Approach to Support Validation of Medical Cyber-Physical Systems. *Sensors* **2015,** *15* (11), 27625-27670.

257. Wan, K.; Alagar, V. In *Achieving dependability of cyber physical systems with autonomic covering*, Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference on, IEEE: 2014; pp 139-145.

258. Asplund, F., The future of software tool chain safety qualification. *Safety Science* **2015,** *74*, 37-43.

259. Pirvu, B.-C.; Zamfirescu, C.-B.; Gorecky, D., Engineering insights from an anthropocentric cyber-physical system: A case study for an assembly station. *Mechatronics* **2016,** *34*, 147-159.

260. Bradley, J. M.; Atkins, E. M., Optimization and control of cyber-physical vehicle systems. *Sensors* **2015,** *15* (9), 23020-23049.

261. Boano, C. A.; Römer, K.; Bloem, R.; Witrisal, K.; Baunach, M.; Horn, M., Dependability for the Internet of Things—from dependable networking in harsh environments to a holistic view on dependability. *e & i Elektrotechnik und Informationstechnik* **2016,** *133* (7), 304-309.

262. Teodorov, C.; Dhaussy, P.; Le Roux, L., Environment-driven reachability for timed systems. *International Journal on Software Tools for Technology Transfer* **2017,** *19* (2), 229-245.