Transactions of the SDPS:
Journal of Integrated Design and Process Science
25 (2), 2021, 1-26
DOI 10.3233/JID210030
http://www.sdpsnet.org

EDITORIAL

Designing Cyber-Physical Systems for Runtime Self-Adaptation:

Knowing More about What We Miss …

Imre Horváth[a*] and Jože Tavčar[b]

[a] *Professor Emeritus, Faculty of Industrial Design Engineering, Delft University of Technology, Delft, the Netherlands*
[b] *Senior Lecturer, Product Development Division, Faculty of Engineering, University of Lund, Sweden*

Keywords: Cyber-physical systems, programmed adaptation, runtime adaptation, self-adaptation, self-evolution, self-supervision, autonomous systems, open issues

1. First Things First – Our View on Cyber-Physical Systems

We live in an age of extensive scientific, technological, and paradigmatic convergence (Franz, 2011). One of the major current trends is the integration of social science, cognitive science, biotechnology, information technology, and nanotechnology, which enables the fusion of bits, atoms, neurons, genes, and memes (Yankovskaya and Kukushkin, 2019). This accelerating integration process, graphically depicted in **Figure 1**, is often referred to as the bits-atoms-neurons-genes-memes revolution (Wetter, 2006). Cyber-physical systems (CPSs) not only represent practical examples of the integration of bits and atoms in human and social contexts but also contribute to the integration of neurons and genes into system implementation (Horváth and Gerritsen, 2012). The move toward integration of neurons is exemplified by the interest in cyber-biophysical systems (e.g., assistive and corrective implants (Ospina et al., 2016) and artificial limbs/augmentations (Zhang et al., 2011), while the results in the latter field are epitomized by gentelligent systems (Denkena et al., 2021). Consequently, engineered systems are undergoing a metamorphosis, and the significance of purely hardware (HW), software (SW), and cyberware (CW) systems is shrinking—their places are taken over quickly by heterogeneous and intellectualized systems. From the perspective of system adaptation, the current trends imply the need for a concurrent change in the HW, SW, and CW elements at runtime, in a synergic (compositional) manner. Theoretically, but also practically, the biggest challenge in this context is that the operational changes of the HW constituents

* Corresponding author. Email: I.Horvath@tudelft.nl

 Published by IOS Press

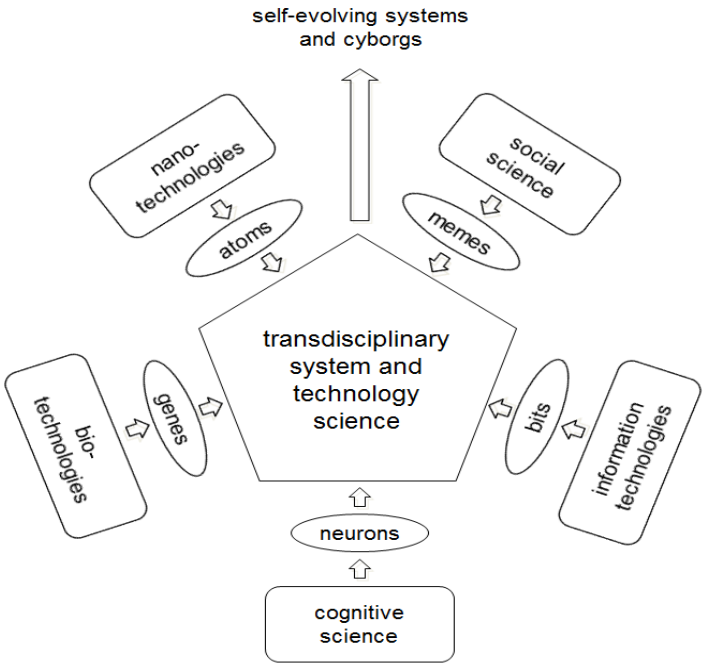

Figure 1. Merger of Technologies and Disciplines

occur in the spatial–temporal space, the changes of the SW constituents in the logical–temporal space, and those of the CW constituents in the syntactic and semantic spaces.

In our view, SW and data/knowledge integrated CPSs (i) include one or more independent (self-contained) or functionally networked actor nodes, (ii) are characterized by deep penetration into real-life physical processes, (iii) operate based on multiple sensing-computing-adjusting loops or sensing-reasoning-learning-planning-adapting loops, (iv) provide tailored services and avail resources dynamically in human, social, and industrial application contexts, (v) have the ability to extend their problem-solving knowledge and computational mechanisms (system intelligence), (vi) may manifest as part of a purposefully and synergistically arranged system of systems, and (vii) evolve through generations (Horváth et al., 2017). Cybernetization of complex engineered systems seems to terminate with highly intellectualized and autonomously operating but cognitively and socially embedded systems (Fitzgerald et al., 2014). If, in sociotechnical systems, the technical parts manifest as CPSs, then researchers talk about socio-CPSs (SCPSs), whose adaptation may also be based on the principles of the centrality of the norms and policy of autonomy and not only on operational goals and affordances (Zhang et al., 2018).

Although the complex phenomenon of system adaptation is currently a hot issue, it is known only partially in the case of complex engineered systems (Black et al., 2014). In biology, adaptation has been defined as the process of subsequent changes by which a living organism or a community of organisms becomes better suited to its environment and increases its chances of survival (Bock, 1980). Initially proposed for natural systems, this interpretation implies four suppositions: (i) adaptation is toward a goal, purpose, or situation; (ii) adaptation is not a one-time action but a purposeful sequence of changes; (iii) adaptation is performed by the subjects of the changes themselves; and (iv) adaptation is to be put into the context of interaction with the environment or a community of organisms. The same principles have been applied to engineered systems (Holland, 1992). However, while biological adaptation is based on evolving biophysiological and cognitive mechanisms, there are no *ab ovo* granted or naturally evolving mechanisms in engineered systems (Horváth et al., 2019). Many experts believe that a deeper theoretical understanding of system adaptation will ultimately lead to the development of autonomous systems and adjustable autonomy.

The remainder of this extended editorial is organized as follows. Section 2 summarizes the types and forms of system control and adaptation. Section 3 introduces the scientific, engineering, and computational fundamentals and issues of adaptation for first-generation CPSs (1G-CPSs). Section 4 discusses the self-adaptation phenomenon of second-generation CPSs (2G-CPSs) and its fundamental issues. Section 5 offers a non-exhaustive landscape of the concerns related to next-generation CPSs (NG-CPSs). In addition, it discusses milestone developments and elaborates on some open questions. Section 6 presents a short synopsis of papers that have contributed to this special issue. Finally, Section 7 reflects on the major findings. It discusses what we apparently miss and what we may consider as an opportunity for future research.

2. A Brief Overview of the Types and Forms of System Adaptation

The natural evolution and selection of living organisms are long-term and strongly conditioned processes. Natural adaptation involves many generations and favors beings with a higher chance of survival and a wide variation of heritable characteristics. Engineered systems do not exhibit these intricate mechanisms of progression. This is the reason that systems science thinks differently about the adaptation of such systems. Nevertheless, it assumes the potential and resources of adaptive systems to change as well as the influence of the environment on the manifestation of changes. A bird's-eye view image of the perspectives of system adaptation is illustrated in **Figure 2**. In general, four sources of the need for adaptation are identified: (i) it is problematic to foresee all requirements because of the broadening and complexification of using such systems in society; (ii) it is difficult to predefine all system operation and interaction modes owing to growing uncertainties concerning applications and stakeholders; (iii) as a consequence of unpredictable incidental effects and changes in the environment, it is difficult to achieve overall resilience in the design phase; and (iv) owing to the emerging technological and servicing affordances, it is often possible to achieve a better performance than that programmed for the systems. System adaptation can be relative to (i) a generally defined goal, (ii) a specifically defined goal, (iii) a partially defined goal, or (iv) an undefined operational/servicing goal. Considering this aspect, adaptation

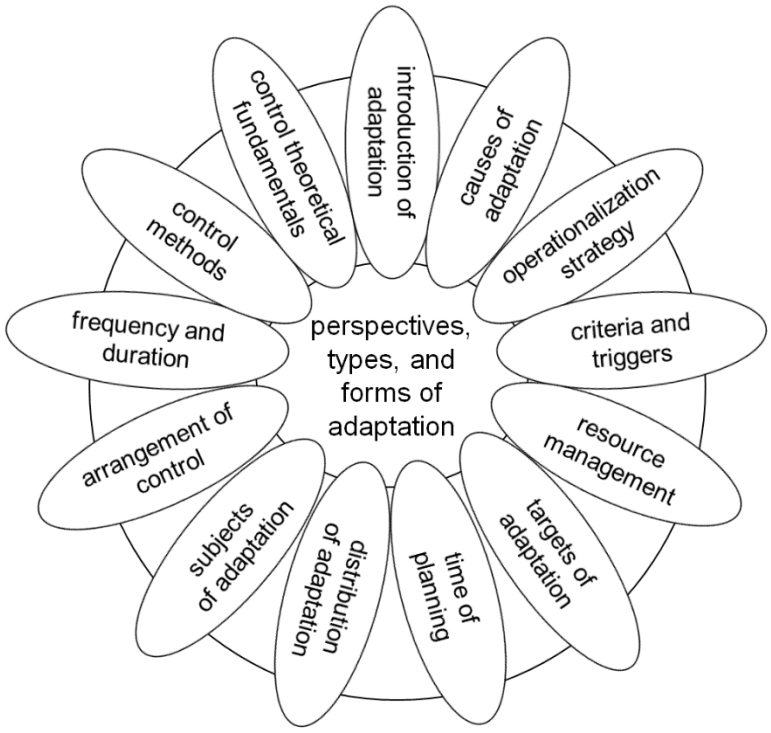

Figure 2. Perspectives of System Adaptation

is a means to (i) serve optimally for a purpose, (ii) maximize the fulfillment of operational/servicing goals, (iii) achieve the best relation with the embedding environment, and (iv) provide optimal interaction with other systems. In other words, it is about how something fits into something else and what efforts it makes toward an overall optimum runtime performance. Adaptation may occur within a short operation period or over the entire lifecycle of engineered systems (Tavčar and Horváth, 2018).

The above similarities and differences triggered the interest toward a universal theory of system adaptation, but this is still a work in progress. From a control theoretical perspective, the literature discusses (i) conventional control-based adaptation (proportional-integral-derivative (PID)-like or state-representation driven), (ii) advanced control-based adaptation (model-predictive, optimization-based, and stochastic), (iii) knowledge-based control (rule-based, fuzzy, heuristic, and analogical), and reasoning-based control (data-driven, learning-based, abductive, prognostic, and twin-based) mechanisms (Shevtsov et al., 2017). It must be emphasized that these types of adaptations apply to complex SW systems rather than to resource-heterogeneous CPSs. In line with the current layering of information technology systems, the categories of (i) infrastructural (HW and SW) resource adaptation, (ii) reusable middleware adaptation, and (iii) domain-specific application SW adaptation are applied (Salehie and Tahvildari, 2009). In general, the criteria (trigger) for the execution of adaptation may be (i) goal-related, (ii) task-oriented, or (iii) performance-based. Based on the operationalization of the adaptation agency, (i) reactive (after a change event), (ii) active (concurrent with a change event), and (iii) proactive (before a change event) control strategies can be distinguished. Feedback-based control supports reactive strategies, whereas feed-forward control is usually active. Combinations of feedback and feed-forward control can detect disturbances and adjust the inputs before the disturbance affects the system outputs. Consequently, this combination implements a proactive strategy that can be used as a proactive control mechanism.

Adaptation is usually not a single action of change but a logically/functionally related linear sequence or other patterns of change actions. Therefore, logical and procedural planning is required in the time dimension, in which various occurrences of adaptations have been identified. For instance, based on the occurrence frequency of adaptation, (i) consecutive (repetitive) and (ii) incidental (one-time) forms of adaptation are distinguished. Based on the duration of adaptation, periodic (repeated at fixed intervals) or permanent (lasting over a relatively long period of operation or the entire lifecycle of a system) are differentiated. In terms of the introduction of changes, adaptation can be made during idle time and/or runtime. In addition, adaptation can be (i) externally initiated (based on intervention or rules provided by an external controller or supervisor) or (ii) internally initiated (based on observed deviations from the intended goal, state, performance, and output, or change in input data). In terms of intentionality (the reason for initiating a specific event), (i) indispensable, (ii) planned, or (iii) self-decided adaptations are distinguished. Adaptations are planned during the (i) design time, (ii) runtime, or (iii) in both.

With regard to the change in the system constituents (components), (i) constant resource-based and (ii) variable resource-based adaptations are implemented. From the perspective of the organization of the changes, (i) centralized and (ii) decentralized approaches are used. The target of adaptation can be (i) goals, (ii) functions, (iii) architecture, (iv) operation, (v) intellectualization, (vi) interactions, (vii) behaviors, or (viii) their combinations. Furthermore, (i) environment-centered adaptation (for proper interaction with a dynamic environment) and (ii) system-centered adaptation (guaranteeing the dependability of the states/operations/services) are differentiated.

As discussed by (Patikirikorala et al., 2012) control may have single or multiple objectives in the case of SW systems, and (i) basic or (ii) composite control schemes are implemented depending on the complexity of the objectives. The basic control schemes include (i) model-based fixed-gain control, (ii) model-based runtime dynamic-gain control, (iii) linear quadratic regulator, and (iv) model-based predictive horizon control. Composite control schemes include (i) cascaded (nested), (ii) rules-based gain scheduling, (iii) algorithm reconfiguring, (iv) top-down distributed (hierarchical), (v) decentralized independent, and (vi) combined event- and time-based (dynamic) controls (Swarnkar et al., 2014). The discussed control strategies are usually put under the conceptual umbrella of internal control, which entails some form of intertwining application functionality (logic) and control functionality (logic).

However, the literature is void with regard to (runtime) HW and CW adaptation issues that are particularly important in the case of transforming CPSs (Estrada-Jimenez et al., 2021).

The aforementioned strategies are typically model-based. Models are either predefined during the design time or generated at runtime. Although the current model engineering makes the creation of dynamic models possible, the range of adaptation is restricted to self-regulation and self-tuning in control-oriented models. When a fault or an unclear change in the environmental circumstances occurs, human intervention is expected. Often, this is referred to as mitigation adaptation, where designers define the (i) specific objectives to achieve, (ii) boundary conditions of operation, (iii) conditions of adaptation, (iv) mechanisms of adaptation, and (v) appropriateness criteria of adaptation (Khoramshahi and Billard, 2019). Another aspect of adaptation is its computational enabling, which can be (i) model-based, (ii) data-driven, (iii) awareness-based, or (iv) ontology-based. Model-based adaptation strategies involve the harmonization of various models such as (i) system, (ii) control, (iii) optimization, (iv) environment, (v) impact, and (vi) meta models.

Internally initiated adaptation is self-adaptation, a form of system operation in which the goals and rules of adaptation are not provided by external controllers. Traditionally, self-adaptation of systems was defined as the ability to make appropriate corrective actions based on information about the actions, which will have the best enhancement impact on the system during runtime. Recently, it has been reinterpreted as the capability of (i) setting a new goal at runtime for system-level problem solving, (ii) determining the most efficient strategy, plan, and execution of changes, and (iii) working accordingly to reach the initial or runtime set goal (Cámara et al., 2016). This multifaceted capability assumes sufficient awareness, reasoning, learning, planning, decision-making abilities, and mechanisms. For many researchers, the core of designing for adaptation is system-level modeling that (i) defines the relationship with the operational environment, (ii) monitors the objectives and the state of a system, and (iii) configures adaptation mechanisms and strategies during the design time of a system.

The above overview of the major adaptation aspects intends to shed light on the complexity of the phenomenon of system adaptation. Furthermore, it attempts to prove that the landscape of research and development activities toward the realization of system adaptation is extremely broad and varied. Two tangible reasons for this are (i) the current wide spectrum of system manifestations and (ii) the dynamic appearance of new generations of engineered systems.

3. Systems Science, Engineering, and Computational Issues of Adaptation of 1G-CPSs

The family of 1G-CPSs includes control-intensive plant-type systems in which the primary objective and logic of operation do not change during the lifespan. Coordinated control loops are essential for building this type of adaptive system, which is actually a result of functional enhancement by cyber-physical augmentation (i.e., supplementing physical systems with standalone or networked computational platforms) (Cómbita et al., 2015). The interfaces between the physical transformation processes and information computation processes are sensors and effectors (or their clusters). Another approach for realizing 1G-CPSs is to complement a digital network with physical objects (instruments, devices, robots, vehicles, etc.). This is a typical strategy of the Internet of Things (or Internet of Everything)–driven development efforts (Miraz et al., 2015). In view of the capabilities of rapidly progressing higher-level implementations, 1G-CPSs are regarded as low-end CPSs.

The functionality, architecture, and logic of operation of the 1G-CPSs are defined in the design phase, and they do not change throughout the lifespan of the system. In other words, this family of CPSs is supposed to adapt to known modes of change. This assumption makes it possible for designers to extensively use model-based engineering in their development. Usually, 1G-CPSs are equipped with conventional control mechanisms and can regulate the parameters of operation to a known degree through the system and control models. The end-user can adjust predefined adaptive control algorithms using preselected parameters. According to the latest reviews of industrially relevant control strategies, the most commonly used in practice are PID and model-based predictive controls. Such solutions are acceptable for many applications under predictable circumstances and working conditions. However, 1G-CPSs may

become unreliable or inefficient in situations that are not considered in the design phase and those that they are unable to adapt to.

The self-control implemented by 1G-CPSs may appear in multiple forms, such as self-regulation, self-healing, self-resilience, or self-tuning. Although these capabilities, like self-adaptiveness, are typically realized in a top-down manner, the literature considers them as a limited subset of capabilities that make CPSs self-adaptive (Brun et al., 2009). They are also differentiated from self-organization that, with a view to emergent functionalities and decentralization of their control, works in a bottom-up manner. We see self-organization as the mutual adaptation and co-evolution of the initially autonomous components of systems, namely, agents. In the literature, self-organization is a spontaneous process through which systems emerge and evolve, become increasingly complex, adaptive, and synergetic (Heylighen, 2011).

Internally initiated control intertwines the logics of application functions and adaptation functions. This approach is based on programming language features, such as conditional expressions, parameterization, and exceptions in SW systems (Floch et al., 2006). The sensors, effectors, and adaptation processes are combined with the application code. This often leads to poor scalability and maintainability and makes the system expensive to test and maintain/evolve. Using an external adaptation engine (or adaptation manager), external approaches of self-adaptive SW systems try to avoid these limitations by offering sophisticated adaptation processes. In addition, they offer reusability (customization and configuration for different systems) of the adaptation engine or processes tailored for various applications. An adaptation engine can implement both closed adaptation (using defined types/number of adaptive actions) and open adaptation (allowing new SW arrangements and behaviors during runtime) (Vogel and Giese, 2013).

Over the years, a dual-aspect solution has emerged in the form of a monitor-analyze-plan-execute (MAPE) approach (Kephart and Chess, 2003). This conceptual abstraction and generalization of the external feedback loop-based control realizes an adaptation logic that is significant for several reasons. For example, it (i) allows separating the concerns of the fulfillment of the system functionality and managing self-control; (ii) facilitates model-based adaptation control, even self-adaptation, by decomposing the control loop into four specific phases; (iii) supports the extension of control information with knowledge stored in a knowledge repository; and (iv) creates a methodological bridge between the self-control of 1G-CPSs and self-adaptation of 2G-CPSs. As discussed by Miller, the monitor, analyze, plan, and execute functions must share knowledge. Hence, this modeling approach is often referred to as MAPE-K (Miller, 2005). The use of formally specified MAPE-K templates that encode the design expertise for a family of self-adaptive systems was proposed by (Iglesias and Weyns, 2015). These include templates for behavioral specification and modeling the different components of a MAPE-K feedback loop, as well as property specification templates that support the verification of the correctness of the adaptation behaviors.

However, the MAPE-K approach is limited in terms of runtime variability, including the variable structure and functional systems. Furthermore, the issues of verifying the adaptation plans before executing and validating the results of the completed self-adaptation in context and the issue of resource generation and management during the lifecycle of the controlled system are not addressed specifically. It was argued that these functions should be included in the self-adaptation loop and proposed managing them in four logical steps, namely: (i) planning self-adaptation, (ii) verification before self-adaptation, (iii) operationalization of self-adaptation, and (iv) validation of self-adaptation, which extends MAPE-K into MAPVEV-K (Tavčar and Horváth, 2020).

After analyzing the current research on the methods and techniques for designing and engineering adaptive SW systems, Hidaka et al. argued that the effective development of self-adaptive systems could be achieved through the reuse and adaptation of existing models, such as MAPE-K loops (Hidaka et al., 2019). The survey completed by Muccini et al. found that the application layer and middleware layer (rather than the communication, service, or cloud layer) are the typical levels of system adaptation and that MAPE-RL (where RL stands for “reason and learn”), agents, and self-organization are the dominant adaptation mechanisms (Muccini et al., 2016). These functions are considered crucial elements for the self-supervised self-adaptation of CPSs.

An analysis (and comparison) of the architecture frameworks currently proposed for designing self-adaptive systems was presented by (Chandra et al., 2016). The analysis included (i) the observe-decide-act, (ii) MAPE-K, (iii) autonomic computing paradigm, and (iv) observer/controller architecture frameworks, which are rooted in organic computing research and are intended for different types of distributed systems, such as swarms, systems of systems, crowd computing arrangements, computing entity populations, and multi-agent systems. (Hummaida et al., 2016) presented a resource management strategy for clouds (that is based on allocation of a shared pool of configurable computing resources). This is seen as a typical example of demand-enabled system adaptation. As a concluding remark, we may claim that, despite the efforts, only useful pieces of an incomplete theory of system-level self-control of real-life systems are available and these do not include the agency of (intuitive and creative) heuristics or metaheuristics, which help solve a wide variety of application problems.

4. Systems Science, Engineering, and Computational Fundamentals of Self-Adaptation of 2G-CPSs

The above discussion is based on five main premises: (i) 1G-CPSs are designed for known modes of change and to implement self-tuning of their operation; (ii) 2G-CPSs are designed to handle partially or completely unknown modes of change and are equipped with the capability of operational self-adaptation; (iii) human stakeholders play an important role in the assurance of system operation and performance of 1G-CPSs, and there is a move toward partial automation of adaptation in the case of 2G-CPSs; (iv) the application and adaptation functions are purposefully separated in self-adaptive systems, whereas the application logic and adaptation logic are largely mixed in adaptive systems; and (v) research in self-adaptive systems distinguishes between internal and external adaptation mechanisms. These assumptions lend themselves not only to the distinction of various system generations but also to a natural demarcation of two major realms of system control: internal and external.

In principle, the goal of self-adaptation can be adapting: (i) the environment to maintain the targeted performance of the system, (ii) the system operations according to the environmental changes, or (iii) both in combination. Conventionally, adaptive systems are preprogrammed to realize the adaptation logic by means of closed feedback loops, while self-adaptive systems are preprogrammed to find a possible or the relative best adaptation logic by sophisticated computational mechanisms such as learning, reasoning, and abstracting (Cámara et al., 2020). In the case of self-adaptation, on the one hand, the designers define the (i) overall objectives to achieve, (ii) overall operational processes, (iii) possible resources to be used for adaptation, and (iv) scenario of realizing possible adaptations. On the other hand, the system determines the (v) necessity of adaptation, (vi) resources to be used for adaptation, (vii) concrete procedures of adaptation, and (viii) execution of adaptation. In the case of self-adaptive systems, it is possible to separate the parts of the system that deal with application concerns (i.e., the goals for which the system is built) from those that deal with self-adaptation concerns. Although this separation is useful for system engineering and computational reasons, the application-oriented subsystem and control-oriented subsystem are supposed to operate in a synergetic functional coupling. From a computational perspective, 2G-CPSs may exploit (i) search-based techniques, (ii) logical and uncertain reasoning techniques, and (iii) machine learning techniques to deal with unanticipated requests and uncertainties and prepare for change. By doing so, they implement various forms of autonomic computing (Sanchez et al., 2020).

Self-adaptation of (heterogeneous) CPSs is a more complicated task than that of SW systems. One obvious reason is that the control SW should adapt not only to itself but also to the HW and CW constituents. Another reason is that the planning of adaptation requires comprehensive context management. Many researchers see self-adaptation as a risk-mitigation strategy with regard to uncertainties caused by runtime changes in application-oriented subsystems. There is still a knowledge gap regarding the handling of real-time changes and constraints that account for context variability. Rodrigues et al. combined offline requirements and model checking with online data collection and assessment to guarantee the system goals by fine-tuning the adaptation policies toward the optimization of

quality attributes (Rodrigues et al., 2018). (Engelenburg et al., 2019) provided a method of identifying which elements of the environment are relevant contexts. The method involves three steps: (i) obtaining insight into the context, (ii) determining what components are required to sense and adapt to context, and (iii) determining the rules on how the system should adapt to different situations. Because both static and dynamic contexts must be managed in specific applications, Don et al. proposed an event-driven awareness mechanism (Don et al., 2012).

Another source of complication is that, beyond the change in the operational parameters, self-adaptation extends to changing elements of the system functionality (operations) and system architecture (configuration and relations of components) during runtime. To deal with these issues, (Braberman et al., 2015) proposed a reference architecture that allows for coordinated yet transparent and independent adaptation of system configuration and behavior. (Cansado et al., 2010) proposed a formal framework that unifies behavioral adaptation and structural reconfiguration of components and showed the advantages in the context of reconfiguration of a client/server system in which the server has been replaced.

It is well known by the SW engineering community that the term "architecture" refers to a conceptual model that defines the behavior, structure, and characteristics of a SW system that fulfills the given requirements. In SW engineering, architecture is a bridge between requirements and computational codes (Garlan, 2001). It is also conceived as a formal description of the integrated, distributed, or hybrid arrangement and the interconnection of functional components. Architectural adaptation is a multifaceted issue involving qualitative judgment and implies modification at various levels (Cheng et al., 2006). Understanding its guiding principles and possible forms is a central topic for research on self-adaptive systems. (Villegas et al., 2017) posited that, in addition to the regular functional components of the system, the designed architectures must include components that enable self-awareness capabilities, such as monitoring and analyzing its own current state, as well as planning and executing self-adaptation actions. There are different possibilities for runtime architectural self-adaptation of composable and compositional systems. (Kramer and Magee, 2007) outlined a three-layer architectural reference model that provides the required level of abstraction and generality for self-management of composable architectures.

Compositional adaptation exchanges algorithmic or structural system components with others to improve the fit of the SW to the state of its current environment. (Phan and Lee, 2011) proposed a multimodal compositional approach to model, analyze, and design an adaptive CPS on a distributed architecture that facilitates adaptiveness, efficient use of resources, and incremental integration. Compositional adaptation is powerful, but its use without appropriate tools to automatically generate and verify codes may negatively affect the system integrity and security (McKinley al., 2004). Compositional self-adaptation control systems should consider both static aspects (such as stability and availability) and dynamic properties (such as functional interconnections and transient changes in variables). The dependable emergent ensembles of components framework presented by (Masrur et al., 2016) (i) allows modeling large-scale dynamic systems by a set of interacting components, (ii) provides mechanisms to describe transitory interactions between components, and (iii) supports reasoning about the timing behavior of the interacting components. The motivation came from the hypothesis that components may automatically configure their interactions within self-managed SW architectures in a manner that is compatible with the overall architectural specification and can achieve the goals of the system. Another dimension of self-adaptation is the self-adaptation of a system of systems, which is still in its early stage of development and implementation (Weyns and Andersson, 2013).

Using models as bases for self-adaptation is both a theoretical and methodological issue. The latter is concerned with the dynamic generation and adaptation of system and control models. Runtime models are based on abstractions of the system, while the goals serve as a driver and enabler for semi-automatic reasoning regarding system adaptations during operation (Blair et al., 2009). Many researchers have emphasized the role of SW models at runtime (M@RT) as an extension of adaptation control techniques to runtime context (Szvetits and Zdun, 2016). For example, a key challenge for self-adaptive SW systems is assurance. Assurance tasks must be performed at runtime. To this end, Cheng et al. argued that research

into the use of M@RT is fundamental to the development of runtime assurance techniques and presented what information may be captured by M@RT for the purpose of assurance (Cheng et al., 2014). (Bennaceur et al., 2014) developed a four-layer partially causal conceptual M@RT reference model to provide a framework for the core concepts and situate the computational mechanisms.

The MAPE-K feedback loop architecture was extended by (Klös et al., 2018). They imposed various requirements and a structure on the knowledge base and introduced a meta-adaptation layer. This enables (i) learning of new adaptation rules based on executable runtime models, (ii) continuous evaluation of the accuracy of previous adaptations, and (iii) verification of the correctness of the adaptation logic in the current system context. (Hadj-Kacem et al., 2009) constructed a formal model using a colored Petri net for an adaptive system to be trusted after adaptation. This model has sufficient abstraction of details but still deals with the core of the protocol. This makes the model simpler and easier to analyze owing to the restricted state-space size. Another study of theoretical significance is the three-phase approach for modeling and developing dynamically adaptive systems based on the combination of the runtime model technique and aspect-oriented SW development paradigm proposed by (Loukil et al., 2017). The SW architecture is specified in the first phase, the executable code is automatically generated in the second phase, and the running system is reconfigured and supervised in the third phase.

The use of artificial narrow intelligence techniques (in particular deep learning and machine learning) and full-fledged digital twins in various runtime activities of system self-adaptation and dependable automation is increasing (Müller et al., 2021). This ongoing intellectualization concerns both tasks related to solving application problems and tasks related to self-adaptation and self-supervision (Casadei et al., 2020). Both theoretical and practical issues are addressed in both respects. The integration of awareness building, machine learning, and ampliative reasoning mechanisms into SW enables them to behave smartly and handle unanticipated situations (Lee et al., 2020). The latter efforts are justified by the growing need to autonomously detect and manage unanticipated or unknown situations and properly plan adaptation during runtime (Zhou et al., 2019). The inclusion of learning mechanisms in self-adaptive systems improves not only their flexibility but also their reusability (Pearce et al., 2021). However, current computational learning allows self-adaptive CPSs to change their operation and/or configuration only up to specific limits or inside a goal-defined operational envelope. Furthermore, constraints and usable resources are defined in the design phase (Macher et al., 2019). Nevertheless, these technological augmentations of 2G-CPSs (i) transfer discrete functional and architectural adaptation approaches into a continuous (perpetual) self-adaptation, (ii) reduce reliance on human supervisors and increase the level of automation, and (iii) enhance the technological readiness for resource-sensitive evolutionary self-adaptation. Three major issues are (i) purpose-driven selective learning, (ii) trustworthiness of the learned data- and rule-driven models, and (iii) scalability of the proposed solutions. Therefore, many researchers have been encouraged to gain experience in industrial systems and applications (Kühnle and Bayanifar, 2017).

5. Toward Next-Generation Cyber-Physical Systems

We conducted a non-exhaustive literature study with the intention of obtaining insights into the (i) trends of current research, (ii) probable future developments, and (iii) recognizable research/knowledge gaps in the field of NG-CPSs. We focused on seminal publications that presented front-end and road-paving research and development results, critical and conclusive overviews, and evidence of personal viewpoints. An important observation was that only a small portion of the studied journal articles and conference papers looked ahead to future CPSs, although the number of related publications has progressively increased in the last decade. Based on the selected publications, we attempted to draw a landscape of the major concerns of the research and development toward NG-CPSs. To this end, we performed an initial classification of the concerns according to the system that they are related to. The four categories of concern are (i) (holistic) system (Σ), (ii) SW (S), (iii) HW (H), and CW (C) concerns. We divided the system concerns into two subcategories: (i) generic system concerns (Σ_1) and (ii) system supervision concerns (Σ_2). The obtained landscape is depicted in **Figure 3**. Owing to the abundance of

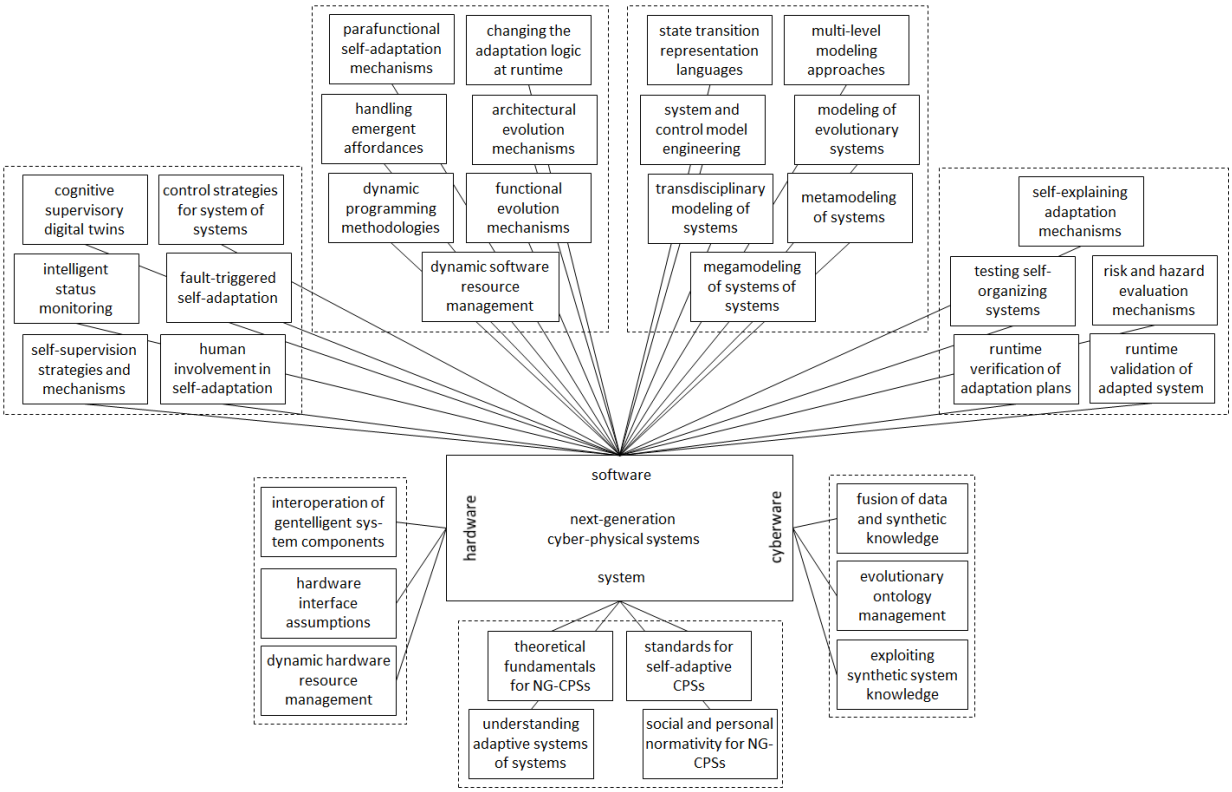

Figure 3. Major Concern-Domains of Next-Generation Cyber-Physical Systems

associated concerns, we classified the SW concerns to three subcategories: (i) system modeling (S_1), (ii) SW self-evolution (S_2), and (iii) SW dependability (S_3) concerns. The subcategories were further decomposed into concern-domains, as follows:

Σ_1 Generic system concerns:
(i) theoretical fundamentals for NG-CPSs (Tan and Chen, 2021) (Colombo et al., 2014) (Yilma et al., 2021), (ii) understanding adaptive systems of systems (Johnson and Hernandez, 2016) (Johnson, 2020), (iii) standards for self-adaptive CPSs (Ali et al., 2018) (Ahmadi et al., 2017), and (iv) social and personal normativity for NG-CPSs (Hohwy, 2020) (Yin et al., 2020) (Yilma et al., 2018).

Σ_2 System supervision concerns:
(i) self-supervision strategies, frameworks, and mechanisms (Klös et al., 2018) (Elkhodary et al., 2010), (ii) human involvement in self-adaptation (Cámara et al., 2017) (Cimini et al., 2020) (Firouznia et al., 2018), (iii) intelligent status monitoring (Cogliati et al., 2018) (Finogeev et al., 2019), (iv) fault-triggered self-adaptation (Krishna and Koren, 2013) (Ruíz Arenas, 2018), (v) cognitive supervisory digital twins (Eirinakis et al., 2020) (Müller et al., 2021), and (vi) control strategies for systems of systems (Ferrer et al., 2018) (Axelsson, 2019).

S_1 System modeling concerns:
(i) state transition representation languages (Du et al., 2020) (Buffoni et al., 2021) (Ruchkin, 2019), (ii) multilevel modeling approaches (Merelli et al., 2012) (Frank, 2014) (Kühne, 2018), (iii) modeling of evolutionary systems (Khakpour et al., 2012) (Hartsell et al., 2019) (Bonci et al., 2018), (iv) system and control model engineering and optimization (Zhang et al., 2019) (de Lamotte et al., 2008) (Fishwick, 1990), (v) transdisciplinary modeling of systems (Hashmi et al., 2021) (Mo and Beckett, 2021), (vi) metamodeling of systems (van Gigch, 1993) (Sangiovanni-Vincentelli et al.,

2009) (Odell et al., 2004) (Yang et al., 2018), and (vii) megamodeling of systems of systems (Villar et al., 2020) (Gašević et al., 2007) (Favre and Nguyen, 2005) (Vogel and Giese, 2012).

S_2 SW self-evolution concerns:
(i) dynamic programming methodologies (Bayne, 1995) (Bickhard, 2019), (ii) functional evolution mechanisms (Gleizes et al., 2007) (Alcocer et al., 2019), (iii) architectural evolution mechanisms (Garces et al., 2020) (Mokni et al., 2016), (iv) handling emergent affordances (Ballesteros et al., 2020) (Zhou et al., 2019), (v) changing the adaptation logic at runtime (Epifani et al., 2009) (Roth et al., 2015) (Pasquale et al., 2011), (vi) parafunctional self-adaptation mechanisms (Basich et al., 2020) (Löffler et al., 2018) (Zhu and Li, 2019), and (vii) SW resource management (Catuogno et al., 2018) (Liu et al., 2020) (Wan et al., 2018).

S_3 SW dependability concerns:
(i) runtime verification of self-adaptation plans (Hachicha et al., 2019) (Nia et al., 2020) (Redfield, and Seto, 2017), (ii) runtime validation of self-adapted systems (Eze et al., 2011) (Dewasurendra et al., 2019) (Cardozo et al., 2014), (iii) testing self-organizing systems (Abuseta and Swesi, 2015) (Eberhardinger et al., 2014), (iv) risk and hazard evaluation mechanisms (Schierman et al., 2008) (Marshall et al., 2018) (Cámara et al., 2018), and (v) self-explaining adaptation mechanisms (Drechsler et al., 2018) (Blumreiter et al., 2019) (Wickramasinghe et al., 2021) (Daglarli, 2021).

H_1 HW management concerns:
(i) dynamic HW resource management (Huang et al., 2009) (Schmidt et al., 2014), (ii) HW interface assumptions (González-Nalda et al., 2017) (Landolfi et al., 2018) (Shin and Park, 2020), and (iii) interoperation of gentelligent system components (Denkena et al., 2010) (Lachmayer et al., 2016) (Möhring et al., 2020).

C_1 CW management concerns:
(i) fusion of data and synthetic knowledge (Cuevas and Guzman-Arenas, 2010) (Zhang et al., 2021) (Sahu et al., 2021) (Petrovska et al., 2020), (ii) evolutionary ontology management (Bürger et al., 2020) (Kotis et al., 2020) (Ferranti et al., 2021)), and (iii) exploiting synthetic system knowledge (Horváth, 2020) (Gembarski, 2019) (Abel, 2014).

The references included above are only examples of typical publications oriented to the particular concern-domains. It should be noted that the landscape displayed in **Figure 3** is incomplete and subjective. There are several reasons for this deficiency. For instance, our literature analysis covered only a limited set of the numerous relevant publications. Owing to the obvious space limitations, only a few could be included in the above overview. There was also a technical issue that several studies addressed multiple concerns or intended to contribute to multiple concern-domains. We attempted to sort them into the most relevant category. Our personal interpretations, views, and judgments contributed to the subjective nature of the landscape. We have not yet conducted any further research to validate its comprehensiveness and appropriateness and to consolidate it in a broader application context.

Notwithstanding these issues, the presented landscape is considered a starting point for further discussions and analyses. It can be observed that the number of concerns related to the HW, SW, and CW categories is significantly different. The overwhelming majority of them are related to SW that plays multiple roles (such as integrators, drivers, processors, mechanisms, managers, and utilities) in the current and future CPSs. The landscape also reflects certain trends, which are summarized in the conclusion section of this extended editorial. In the next section, we use it to position the contributing papers in the most relevant concern-domain.

6. Short Synopses of the Contributed Papers

This special issue is based on an open call for papers initially presented on a journal's website. The call attracted the attention of many potential authors. The process of selecting the submitted manuscripts

for peer review and subsequently screening the best ones for publication was not simple. Many excellently written papers were addressing somewhat conventional topics, but there were fewer well-elaborated papers addressing novel and essential topics. With regard to all the submitted manuscripts, it is fair to mention that there was weak thematic coherence among them. For the above reasons, less than half of the reviewed papers were considered for publication, and finally, only six original contributions were included in this special issue. Based on their actual objectives and contributions, these papers can be categorized into three general groups: (i) road mapping for systems science and engineering (P1 and P2), (ii) methodological approaches to designing self-adaptive systems (P3 and P4), and (iii) enablers for the realization of self-adaptive systems (P5 and P6). Below, we briefly introduce these high-quality papers.

The first paper, submitted by Danny Weyns, Jesper Andersson, Mauro Caporuscio, Francesco Flammini, Andreas Kerren, and Welf Lowe, proposes "*A Research Agenda for Smarter Cyber-Physical Systems*." This paper contributes to the conceptual framing and understanding of several concern-domains in the subcategories of SW self-evolution and SW dependability, as well as in the subcategories of generic system and system supervision. It also provides a broad and deep theoretical underpinning for NG-CPSs. The work complements the existing perspectives on system smartness by taking a more holistic perspective that integrates system operation with the processes to engineer them. The authors argue that both the systems and the way they are engineered must become smarter. Systems and engineering processes must adapt themselves and evolve based on stakeholder input and experience through a perpetual process that continuously improves their capabilities and utility to deal with environmental and operational uncertainties and amounts of data they encounter throughout their lifetime. The authors highlight key engineering areas (CPSs, runtime self-adaptation, data-driven technologies, and visual analytic reasoning) and outline some major challenges in each of them. They explain the synergies between these key areas. The second part of the paper presents the authors' proposal for a comprehensive research agenda. This addresses three themes: (i) assurances for unknowns (in the case of decentralized and smart CPSs that operate under uncertainty), (ii) self-explainability of autonomous decisions (concerning lifelong self-learning and self-explainable CPSs), and (iii) smart ecosystems for perpetual adaptation and evolution (including a unified modeling approach and self-governance for smart CPSs). Exhibiting a high level of autonomy, smarter cyber-physical ecosystems require reflective capabilities based on which they can improve their utility and adjust themselves according to their shifting operational goals. Recognizing the necessity of convergence, the research agenda calls for a multi-year concerted effort by research teams active in the different key areas of studying and developing novel solutions for trustworthy and sustainable CPSs.

The second paper, titled "*Designing Runtime Evolution for Dependable and Resilient Cyber-Physical Systems Using Digital Twins*," presents the work of Luis F. Rivera, Miguel Jimenez, Gabriel Tamura, Norha M. Villegas, and Hausi A. Muller. The main contribution of this paper is on the concern-domain of cognitive supervisory digital twins in the system supervision subcategory. Moreover, it adds to the subcategory of SW self-evolution, more specifically, to the concern-domain of functional/architectural evolution mechanisms, and to the self-supervision strategies, frameworks, and mechanisms. The authors emphasize that designing smart CPSs must address not only dependable autonomy but also operational resiliency. Their goal was to implement reliable self-adaptation and self-evolution mechanisms and to include them in the design of SCPSs. Their results are threefold: (i) a reference architecture for designing dependable and resilient SCPSs that integrates concepts from the fields of digital twins, adaptive controls, and autonomic computing; (ii) a model identification mechanism to guide self-evolution, evolutionary optimization, and dynamic simulation; and (iii) a gradient descent-based adjustment mechanism for self-adaptation to achieve operational resiliency. In addition to the model identification and adjustment mechanisms, a featured contribution of this work is the so-called "reference architecture" for designing digital twin-based autonomic control for dependable and resilient CPSs. The authors implemented prototypes and demonstrated their viability using real data from a case study in the domain of intelligent transportation systems. The proposed execution adjustment mechanism determines the appropriate control parameters such that the controller can enforce the control objectives in the CPS.

The next paper was submitted by Camille Salinesi, Asmaa Achtaich, Nissrine Souissi, Raul Mazo, Ounsa Roudies, and Angela Villota, under the title "*State-Constraint Transition: A Language for the Formal Specification of Self-Adaptive Requirements.*" It offers a methodological approach to designing self-adaptive systems. The main contribution covers the concern-domain of dynamic programming methodologies in the subcategory of SW self-evolution and the concern-domain of state-transition representation languages in the subcategory of system modeling. The authors observed that existing formal languages focus on the fulfillment of user requirements by the designed system in the current context. However, they hardly consider dynamically emerging runtime requirements and context-sensitive requirements. Therefore, the authors introduced a state-constraint transition (SCT) modeling language to provide a solution to the problem of specifying dynamic requirements. An essential feature of this solution is the concept of configuration states, in which requirements are translated into constraints. The paper explains both the syntax and semantics of the SCT and provides examples of reconfiguration scenarios. The authors realized the SCT requirement specification process relying on a finite-state machine approach that provided the necessary computational power and expressiveness for constraint programming. Their preliminary evaluation explored both the benefits (expressiveness, scalability, and domain independence) and limitations (temporal constraints, scheduled reconfigurations, and validation of constraints) of the SCT.

The fourth paper, titled "*One-of-a-Kind Production in Cyber-Physical Production Systems Considering Machine Failures*," presents the study results of Guido Vinci Carlavan and Daniel Alejandro Rossit. Although the topic of the paper is broader than the SW concern, its scientific contribution can be related to the concern-domain of advanced control strategies for system of systems in the subcategory of system supervision. Within customized production, the one-of-a-kind production (OKP) paradigm is an extreme case for production control and scheduling. The CPSs used in Industry 4.0 are supposed to facilitate the management of information related to each singular product as well as the resolution of conflicts that may arise in processes with very high variability. For this reason, the authors studied the implementation of the constant work-in-progress (CONWIP) control logic in OKP systems from the perspective of productive job shop configurations in Industry 4.0. The CONWIP control logic was able to handle the challenging Industry 4.0 problem in an efficient manner, with a relatively low requirement for investment in CPS-related equipment. However, they also found that the performance is sensitive to the stress of the scenario, that is, the arrival rate of jobs, which is an issue closely related to the dispatching rules used. The general conclusion of the authors was that the dispatching rules associated with due dates tend to improve the overall performance of the system, and the first-in, first-out rule has the worst performance in all experiments. An essential feature of their work is the development of simulation-based experimental studies, whose results were compared systematically. As design concerns of the NG-CPSs, Carlavan and Rossit elaborated on intelligent status monitoring, fault-triggered self-adaptation, and system and control model engineering.

The title of the fifth paper is "*Remote Runtime Failure Detection and Recovery Control for Quadcopters.*" The authors, Sajad Shahsavari, Mohammed Rabah, Eero Immonen, Mohammad-Hashem Haghbayan, and Juha Plosilab, identified managing failures as a basic enabler for the realization of dependable self-adaptive systems such as quadcopter drones. Their work contributes to the concern-domains of fault-triggered self-adaptation and cognitive supervisory digital twins in the system supervision subcategory. The authors implemented a distributed control system that includes (i) a local onboard PID-based control subsystem responsible for maneuvering the drone in all conditions, (ii) remote control subsystem responsible for detecting normal or failure states of the drone and communicating with the drone in real-time, and (iii) digital twin co-execution subsystem responsible for real-time two-way data exchange between the above subsystems. The measured RPM values of the quadcopter motors are transmitted to a remote computer, which hosts the failure detection and recovery SW platform. The control concept was implemented using the Simulink tool. The authors propose a modification of the Quad-Sim simulation model to represent motor failure situations. In addition, they offer a fast fault detection and recovery technique capable of working at runtime and a two-way data stream management facility. The experimental results obtained using the MCX co-execution platform demonstrate the

applicability and efficiency of the proposed approach in detecting failures and safely landing of drones after failure detection.

Included as last in this special issue, the work of Amal Ahmed Anda and Daniel Amyot mainly addresses the concern-domain of 'system and control model engineering and optimization' in the subcategory of software 'system modeling concerns'. Nevertheless, their paper, entitled, "*Goal and Feature Model Optimization for the Design and Self-Adaptation of Socio-Cyber-Physical Systems*", also contributes to the concern-domains of runtime validation of the adapted system, functional evolution mechanisms, intelligent status monitoring, and human involvement in self-adaptation. The presented optimization method provides design-time and runtime solutions for goal-based self-adaptation of SCPSs, while supporting the validation of their design models. The goal satisfaction is supported by a simultaneous monitoring the system's environment and operational qualities, while constraints enforcing correctness are specified in the feature model. The arithmetic functions are generated automatically from goal and feature models. The generated goal-feature model is solved by an optimization tool, which calculates optimal adaptation solutions for foreseen common situations at design-time. In addition, runtime optimization is used also by the system in order to adapt to situations unanticipated in the design-time. To assess how well the proposed approach could be used to manage selection among alternatives while solving emergent conflicts, it was applied to a smart home management system. The optimized performance of the system was assessed through the fulfillment of time, total programing time, memory usage, and program memory usage goals/constraints. The approach proposed by Anda and Amyot facilitates iterative processes, reduces design errors, and increases system reliability.

7. Some Conclusions about What We Miss …

Although significant progress has been made both in research and development and in theory and practice, many open issues and unanswered questions still exist. As the above analysis shows, this can be attributed to the extremely rapid shifts in research phenomena and academic interests. Below, we attempt to identify the open issues that are expedient to be resolved immediately:

1. Second-generation CPSs are based on balanced utilization of HW, SW, and CW resources. Nevertheless, most research has focused on SW challenges and issues. This can be explained by the dominance of research in information processing and smart reasoning systems, but self-adaptation of transformative (such as production, robotic, medical, and transportation) 2G-CPSs requires sophisticated HW and CW resource management. Publications on their integral theoretical fundamentals and methodological approaches are scarce in the current literature.

2. As explained above, a functional motivation for self-adaptation is enabling systems to handle operational uncertainties that are difficult to foresee before deployment. At the same time, a nonfunctional motivation for self-adaptation is freeing system operators and administrators from the need to continuously monitor and adjust systems that are operating round-the-clock. Self-adaptation may introduce various levels of transformative operations, such as (i) self-tuning, (ii) self-adaptation, (iii) self-conversion, and (iv) self-reproduction. In all cases, self-adaptive systems are inherently nonlinear because they possess parameters that are functions of their states and conditions. Thus, self-adaptive systems are simply a special class of nonlinear systems that measure their own performance, interact with the operating environment, monitor the operating conditions of the components, and adapt their dynamics. The goal of interaction with their operating environment is to ensure that the measured performance is close to the targeted performance or specifications.

3. Section 4, titled "Systems Science, Engineering, and Computational Fundamentals of Self-Adaptation for 2G-CPSs", and Section 5, titled "Towards Next-Generation Cyber-Physical Systems", discuss the already observable and foreseeable differences in the adaptive potentials of 2G-CPSs and NG-CPSs. Similar to some of the contributed papers, this extended editorial proposes that CPSs are moving from self-adaptation (typical of 2G-CPSs) toward self-evolution and even to a not

preprogrammed but self-supervised automated operation (regarded as typical of 3G-CPSs and higher generations). As indicated in **Figure 4**, the latter raises the need for additional supra-disciplinary research on the intellectualization, organization, and socialization dimensions of NG-CPSs.

4. Facilitating the system self-evolution and reaching autonomy seem to be the two dominant tracks for developing NG-CPSs. Adaptation evolves when new resources are provided for the system runtime. Functional evolution and evolutionary adaptation assume extending the system resources (HW, SW, and CW) at runtime and adapting the system objectives, operation, performance, and relationships according to the obtained affordances. Autonomous adaptation has been interpreted as self-adaptation without any form of human interaction. In this case, the system is responsible for self-supervising both the planning and execution of adaptation, considering all risk factors and implications. The current literature offers neither robust underlying theories nor structured methodologies for evolutionary and autonomous self-adaptation.
5. Artificial narrow intelligent techniques (in particular, various mechanisms of computational learning) are increasingly being used in the self-control and self-adaptation of 2G-CPSs. Artificial neural network-based and other AI-based controller mechanisms extend the self-adaptation potential with additional functionality but are not able to adapt to frequent requirement changes at runtime or to scale up to complex real-life situations. Sections 2–5 hint at some open design issues that cannot be resolved because the knowledge they need is partly or entirely unavailable. To explore the knowledge gaps and eliminate knowledge deficiencies, the problems must first be correctly identified. Cognitive engineering plays an important role in the development of next-generation systems.
6. Dynamic management of the operational and servicing goals of systems based on emerging runtime requirements is recognized as an important topic for further studies; however, the dynamic development of goal models is still in its infancy. Changes during the SW lifecycle lead to SW architecture erosion and make the management of SW architecture evolution a complex task. Most existing computational approaches to architecture evolution only enable the evolution of early stage models and fail to support the entire lifecycle of component-based SW.
7. The fundamental mechanisms of automatic runtime (fine-) tuning of the adaptation logic to

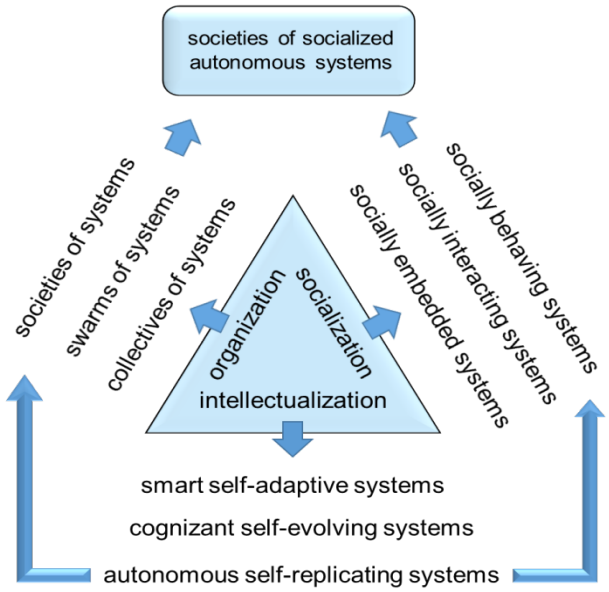

Figure 4. Dimensions of Thinking about Next-Generation Cyber-physical Systems

unanticipated conditions, runtime verification of adaptation plans, learning the impact of adaptation decisions on the goals of the system, and validation and testing of the performance of self-adaptive systems after (multiple) adaptations are still concerns for research and development. These are particularly relevant issues for networked time 2G-CPSs and mission-critical systems.

8. A rapid shift can be observed in the literature from self-adaptive systems to self-supervised self-evolving systems without providing complete solutions for the self-adaptation problem. The idea of layering was introduced in the design of self-adaptive SW systems to separate different types of concerns and address various types of uncertainties. An interesting and important but narrowly addressed research topic is the functional emergence and utilization of functional affordances in the case of NG-CPSs. Emergence may be a result of self-organization, particularly in the case of multi-agent-based systems.
9. Designing CPSs requires an extensive collection of heterogeneous computational models, such as systems, morphological, physical, structural, HW, SW, information, control, and reasoning models, to enable deep semantic integration, simulation, and analysis. Models should interoperate and provide a sufficiently complete representation of the operation, structure, and behavior of 2G-CPSs. Despite efforts to introduce metamodels and megamodels, the currently used models (i) work in conceptually different engineering dimensions, (ii) are based on different abstractions, and (iii) involve different representation formalisms. The methodology of coherent and consistent transdisciplinary and multidimensional system modeling and cross-domain (HW, SW, and CW) representation formalisms require further research attention. Formal criteria for the structural and semantic consistency of modeling tools have not been addressed with sufficient emphasis.
10. Several authors emphasize both the (restricted) necessity and feasibility of building self-explainable systems that monitor and analyse their behavior and generate an explanation for human stakeholders involved in supervision based on explanation models. However, this approach loses significance in systems with high levels of autonomy.
11. Self-adaptive systems mostly consider parametric, functional, and architectural properties that capture concerns, such as performance, reliability, and cost. Recent research has addressed the nonfunctional or parafunctional characteristics of NG-CPSs, such as trust, awareness, intellect, and emotions. These topics seem to be ready for immediate or near-future research.

Acknowledgments

The guest editors are very grateful to everybody who supported and/or contributed to the realization of this special issue. We sincerely appreciate the understanding cooperation of the authors in the rather long-lasting, multicycle review and publication process. The subsequent revisions of the submitted manuscripts helped a lot to consolidate the contents of the papers and achieve the best possible results. We thank the valuable contribution of the peer reviewers who expressed their opinions critically but very constructively. Many thanks also go to the fellow editor colleagues for their inspiration and professional support.

References

[1] Franz, J.H. (2011). A Critique of Technology and Science: An Issue of Philosophy. *Southeast Asia: A Multidisciplinary Journal*, Vol 11, pp. 23-36.

[2] Yankovskaya, V.V., & Kukushkin, S.N. (2019). The Role of the High School in the “Triple Loop” Model: SCBIN Technologies. In: *IOP Conference Series: Earth and Environmental Science*. IOP Publishing, 274(1), p. 012115.

[3] Wetter, K.J. (2006). Implications of Technologies Converging at the Nano-Scale: Are We Ready for the Little BANG? In: *Nanotechnologien Nachhaltig Gestalten*. Institut für Kirche und Gesellschaft: Iserlohn, Germany, pp. 31-41.

[4] Horváth, I., & Gerritsen, B.H. (2012). Cyber-Physical Systems: Concepts, Technologies and Implementation Principles. In: *Proceedings of the International Symposium on Toools and Methods of Competitive Engineering - TMCE 2012*, Vol. 1, Delft university Press, Delft, pp. 19-36.

[5] Ospina, P.D., Díaz, M.C., & Lara, S. (2016). New Technologies in Eye Surgery - A Challenge for Clinical, Therapeutic, and Eye Surgeons. In: *Advances in Eye Surgery*. IntechOpen, pp. 1-20.

[6] Zhang, F., DiSanto, W., Ren, J., Dou, Z., Yang, Q., & Huang, H. (2011). A Novel CPS System for Evaluating a Neural-Machine Interface for Artificial Legs. In: *Proceedings of the IEEE/ACM Second International Conference on Cyber-Physical Systems*, IEEE, pp. 67-76.

[7] Denkena, B., Dittrich, M.A., Stamm, S., Wichmann, M., & Wilmsmeier, S. (2021). Gentelligent Processes in Biologically Inspired Manufacturing. *CIRP Journal of Manufacturing Science and Technology*, 32, pp. 1-15.

[8] Horváth, I., Rusák, Z., & Li, Y. (2017). Order Beyond Chaos: Introducing the Notion of Generation to Characterize the Continuously Evolving Implementations of Cyber-Physical Systems. In: *Proceedings of the International Design Engineering Technical Conferences*, Vol. 58110, American Society of Mechanical Engineers. p. V001T02A015.

[9] Fitzgerald, J., Larsen, P.G., & Verhoef, M. (2014). From Embedded to Cyber-Physical Systems: Challenges and Future Directions, In: *Collaborative Design for Embedded Systems*, Springer, Heidelberg New York, pp. 293-302.

[10] Zhang, J.J., Wang, F.Y., Wang, X., Xiong, G., Zhu, F., Lv, Y., ... & Lee, Y. (2018). Cyber-physical-social systems: The state of the art and perspectives. *IEEE Transactions on Computational Social Systems*, 5(3), pp. 829-840.

[11] Black, W.S., Haghi, P., & Ariyur, K.B. (2014). Adaptive Systems: History, Techniques, Problems, and Perspectives. *Systems*, 2(4), pp. 606-660.

[12] Bock, W.J. (1980). The Definition and Recognition of Biological Adaptation. *American Zoologist*, 20(1), pp. 217-227.

[13] Holland, J.H. (1992). Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press, Boston.

[14] Horváth, I., Suárez Rivero, J.P., & Hernández Castellano, P. M. (2019). Past, Present and Future of Behaviourally Adaptive Engineered Systems. *Journal of Integrated Design and Process Science*, 23(1), pp. 1-15.

[15] Tavčar, J., & Horváth, I. (2018). A Review of the Principles of Designing Smart Cyber-Physical Systems for Runtime Adaptation: Learned Lessons and Open Issues. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(1), pp. 145-158.

[16] Shevtsov, S., Berekmeri, M., Weyns, D., & Maggio, M. (2017). Control-Theoretical Software Adaptation: A Systematic Literature Review. *IEEE Transactions on Software Engineering*, 44(8), pp. 784-810.

[17] Salehie, M., & Tahvildari, L. (2009). Self-Adaptive Software: Landscape and Research Challenges. *ACM Transactions on Autonomous and Adaptive Systems*, 4(2), pp. 1-42.

[18] Patikirikorala, T., Colman, A., Han, J., & Wang, L. (2012). A Systematic Survey on the Design of Self-Adaptive Software Systems using Control Engineering Approaches. In: *Proceedings of the 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, IEEE, pp. 33-42.

[19] Swarnkar, P., Jain, S.K., & Nema, R.K. (2014). Adaptive Control Schemes for Improving the Control System Dynamics: A Review. *IETE Technical Review*, 31(1), pp. 17-33.

[20] Estrada-Jimenez, L. A., Pulikottil, T., Peres, R. S., Nikghadam-Hojjati, S., & Barata, J. (2021). Complexity Theory and Self-Organization in Cyber-Physical Production Systems. *Procedia CIRP*, 104, pp. 1831-1836.

[21] Khoramshahi, M., & Billard, A. (2019). A Dynamical System Approach to Task-Adaptation in Physical Human–Robot Interaction. *Autonomous Robots*, 43(4), pp. 927-946.

[22] Cámara, J., Lopes, A., Garlan, D., & Schmerl, B. (2016). Adaptation Impact and Environment Models for Architecture-Based Self-Adaptive Systems. S*cience of Computer Programming*, 127, pp. 50-75.

[23] Cómbita, L.F., Giraldo, J., Cárdenas, A.A., & Quijano, N. (2015). Response and Reconfiguration of Cyber-Physical Control Systems: A Survey. In: *Proceedings of the 2nd Colombian Conference on Automatic Control*, IEEE, pp. 1-6.

[24] Miraz, M.H., Ali, M., Excell, P.S., & Picking, R. (2015). A Review on Internet of Things (IoT), Internet of Everything (IoE) and Internet of Nano Things (IoNT). In: *Proceedings of the 2015 Internet Technologies and Applications Conference*, IEEE, pp. 219-224.

[25] Brun, Y., Di Marzo Serugendo, G., Gacek, C., Giese, H., Kienle, H., Litoiu, M., Muller, H., Pezze, M., & Shaw, M. (2009), Engineering Self-Adaptive Systems through Feedback Loops. In: *Software Engineering for Self-Adaptive Systems*, Springer-Verlag, Berlin, pp. 48-70.

[26] Heylighen, F. (2011). Self-Organization of Complex, Intelligent Systems: An Action Ontology for Transdisciplinary Integration. *Integral Review*, 134, pp. 1-39.

[27] Floch, J., Hallsteinsen, S., Stav, E., Eliassen, F., Lund, K., & Gjorven, E. (2006). Using Architecture Models For Runtime Adaptability. *IEEE Software,* 23(2), pp. 62-70.

[28] Vogel, T., & Giese, H. (2013). Model-Driven Engineering of Adaptation Engines for Self-Adaptive Software: Executable Runtime Megamodels. *Technische Berichte No. 66*. Universitätsverlag Potsdam. 1-59.

[29] Kephart, J.O., & Chess, D.M. (2003). The Vision of Autonomic Computing. *Computer*. 36(1), pp. 41–50.

[30] Miller, B. (2005). The autonomic computing edge: The Role of Knowledge in Autonomic Systems. DB2 Developer Domain. http://www128.ibm.com/developerworks/autonomic/library/ac-edge6/.

[31] Iglesia, D.G., & Weyns, D. (2015). MAPE-K Formal Templates to Rigorously Design Behaviors for Self-Adaptive Systems. *ACM Transactions on Autonomous and Adaptive Systems*, 10(3), pp. 1-31.

[32] Tavčar, J., & Horváth, I. (2020), How Can a Smart Cyber-Physical System Validate its Runtime Adaptation Actions Before and After Executing Them?, In: *Proceedings of the Thirteenth International Symposium on Tools and Methods of Competitive Engineering Symposium,* Delft University of Technology, pp. 103-114.

[33] Hidaka, S., Hu, Z., Litoiu, M., Liu, L., Martin, P., Peng, X., ... & Yu, Y. (2019). Design and Engineering of Adaptive Software Systems. In: *Engineering Adaptive Software Systems,* Springer, Singapore, pp. 1-33.

[34] Muccini, H., Sharaf, M., & Weyns, D. (2016). Self-Adaptation for Cyber-Physical Systems: A Systematic Literature Review. In: *Proceedings of the 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pp. 75-81.

[35] Chandra, A., Lewis, P.R., Glette, K., & Stilkerich, S.C. (2016). Reference Architecture for Self-Aware and Self-Expressive Computing Systems. In: *Self-Aware Computing Systems,* Springer, Cham. pp. 37-49.

[36] Hummaida, A.R., Paton, N.W., & Sakellariou, R. (2016). Adaptation in Cloud Resource Configuration: A Survey. *Journal of Cloud Computing*, 5(1), pp. 1-16.

[37] Cámara, J., Papadopoulos, A.V., Vogel, T., Weyns, D., Garlan, D., Huang, S., & Tei, K. (2020). Towards Bridging the Gap between Control and Self-Adaptive System Properties. In: *Proceedings of the 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, *IEEE/ACM,* pp. 78-84.

[38] Sanchez, M., Exposito, E., Aguilar, J. (2020). Autonomic Computing in Manufacturing Process Coordination in Industry 4.0 Context. *Journal of Industrial Information Integration*, 19, pp.1-16.

[39] Rodrigues, A., Caldas, R.D., Rodrigues, G.N., Vogel, T., & Pelliccione, P. (2018). A Learning Approach to Enhance Assurances for Real-Time Self-Adaptive Systems. In: *Proceedings of 13th IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, IEEE, pp. 206-216.

[40] Engelenburg, S., Janssen, M., & Klievink, B. (2019), Designing Context-Aware Systems: A Method for Understanding and Analysing Context in Practice. *Journal of Logical and Algebraic Methods in Programming*, 103(2019), pp. 79-104.

[41] Don, S., Choi, E., & Min, D. (2012). Event Driven Adaptive Awareness System for Medical Cyber Physical Systems. In: *Proceedings of the 4th International Conference on Awareness Science and Technology*. IEEE, pp. 238-242.

[42] Braberman, V., D'Ippolito, N., Kramer, J., Sykes, D., & Uchitel, S. (2015), MORPH: A Reference Architecture for Configuration and Behaviour Self-Adaptation. In: *Proceedings of the 1st International Workshop on Control Theory for Software Engineering*, August 2015 pp. 9-16.

[43] Cansado, A., Canal, C., Salaün, G., & Cubo, J. (2010). A Formal Framework for Structural Reconfiguration of Components under Behavioural Adaptation. *Electronic Notes in Theoretical Computer Science*, 263, pp. 95-110.

[44] Garlan, D. (2001). Software Architecture. In: *Wiley Encyclopedia of Software Engineering*. John Wiley & Sons, Inc., New York. pp. 1-10.

[45] Cheng, S.W., Garlan, D., & Schmerl, B. (2006). Architecture-Based Self-Adaptation in the Presence of Multiple Objectives. In: *Proceedings of the 2006 International Workshop on Self-Adaptation and Self-Managing Systems*, pp. 2-8.

[46] Villegas, N.M., Tamura, G., & Müller, H.A. (2017). Architecting Software Systems for Runtime Self-Adaptation: Concepts, Models, and Challenges. In: *Managing Trade-offs in Adaptable Software Architectures*, Morgan Kaufmann, x, pp. 17-43.

[47] Kramer J., & Magee, J. (2007). Self-Managed Systems: An Architectural Challenge. In: *Proceedings of the Conference on the Future of Software Engineering*. Washington, DC, IEEE Computer Society, pp. 259-268.

[48] Phan, L.T., & Lee, I. (2011). Towards a Compositional Multimodal Framework for Adaptive Cyber-Physical Systems. In: *Proceedings of the 17th International Conference on Embedded and Real-Time Computing Systems and Applications*, Vol. 2, IEEE, pp. 67-73.

[49] McKinley, P.K., Sadjadi, S.M., Kasten, E.P., & Cheng, B.H. (2004). Composing Adaptive Software. *Computer*, 37(7), pp. 56-64.

[50] Masrur, A., Kit, M., Matěna, V., Bureš, T., & Hardt, W. (2016). Component-Based Design of Cyber-Physical Applications with Safety-Critical Requirements. *Microprocessors and Microsystems*, 42, pp. 70-86.

[51] Weyns, D., & Andersson, J. (2013). On the Challenges of Self-Adaptation in Systems of Systems. In: *Proceedings of the First International Workshop on Software Engineering for Systems-of-Systems*, pp. 47-51.

[52] Blair, G., Bencomo, N., & France, R.B. (2009). "Models@run.time," *Computer*, 42(10), pp. 22-27.

[53] Szvetits, M., & Zdun, U. (2016). Systematic Literature Review of the Objectives, Techniques, Kinds, and Architectures of Models at Runtime. *Software & Systems Modeling*, 15(1), pp. 31-69.

[54] Cheng, B.H., Eder, K.I., Gogolla, M., Grunske, L., Litoiu, M., Müller, H.A., ... & Villegas, N.M. (2014). Using Models at Runtime to Address Assurance for Self-Adaptive Systems. In: *Models@Run.time*. Lecture Notes in Computer Science, Vol. 8378, Springer, Cham. pp. 101-136.

[55] Bennaceur, A., France, R., Tamburrelli, G., Vogel, T., Mosterman, P. J., Cazzola, W., ... & Redlich, D. (2014). Mechanisms for Leveraging Models at Runtime in Self-Adaptive Software. In: *Models@Run.time*. Lecture Notes in Computer Science, Vol. 8378, Springer, Cham. pp. 19-46.

[56] Klös, V., Göthel, T., & Glesner, S. (2018). Comprehensible and Dependable Self-Learning Self-Adaptive Systems. *Journal of Systems Architecture*, 85, pp. 28-42.

[57] Hadj-Kacem, N., Kacem, A.H., & Drira, K. (2009). A Formal Model of a Multi-Step Coordination Protocol for Self-Adaptive Software Using Coloured Petri Nets. *International Journal of Computing and Information Sciences*, 7(1), pp. 1-15.

[58] Loukil, S., Kallel, S., Jmaiel, M. (2017). An Approach Based on Runtime Models for Developing Dynamically Adaptive Systems. *Future Generation Computer Systems*, 68(3), pp. 365-375.

[59] Müller, M., Müller, T., Talkhestani, B.A., Marks, P., Jazdi, N., & Weyrich, M. (2021). Industrial Autonomous Systems: A Survey on Definitions, Characteristics and Abilities. *AT-Automatisierungstechnik*, 69(1), pp. 3-13.

[60] Casadei, R., Pianini, D., Placuzzi, A., Viroli, M., & Weyns, D. (2020). Pulverization in Cyber-Physical Systems: Engineering the Self-Organizing Logic Separated from Deployment. *Future Internet*, 12(11), 203, pp. 1-28.

[61] Lee, J., Azamfar, M., Singh, J., & Siahpour, S. (2020). Integration of digital twin and deep learning in cyber-physical systems: Towards smart manufacturing. In: *IET Collaborative Intelligent Manufacturing*, 2(1), pp. 34-36.

[62] Zhou, P., Zuo, D., Hou, K.M., Zhang, Z., Dong, J., Li, J., & Zhou, H. (2019). A Comprehensive Technological Survey on the Dependable Self-Management CPS: From Self-Adaptive Architecture to Self-Management Strategies. *Sensors*, 19(5), p. 1033.

[63] Pearce, H., Yang, X., Pinisetty, S., & Roop, P.S. (2021). Runtime Interchange for Adaptive Reuse of Intelligent Cyber-Physical System Controllers. arXiv preprint arXiv:2110.01974.

[64] Macher, G., Diwold, K., Veledar, O., Armengaud, E., & Römer, K. (2019). The Quest for Infrastructures and Engineering Methods Enabling Highly Dynamic Autonomous Systems. In: *Proceedings of the European Conference on Software Process Improvement*. Springer, Cham, pp. 15-27.

[65] Kühnle, H., & Bayanifar, H. (2017). An Approach to Develop an Intelligent Distributed Dependability and Security Supervision and Control For Industry 4.0 Systems. *Asian Journal of Information and Communications*, 9, pp. 8-16.

[66] Tan, T., & Chen, J. (2021). Model Predictive and Inversive Control for State Transition of Dynamics Systems. *Advances in Astronautics Science and Technology*, pp. 1-6.

[67] Colombo, A.W., Karnouskos, S., & Bangemann, T. (2014). Towards the Next Generation of Industrial Cyber-Physical Systems. In: *Industrial Cloud-Based Cyber-Physical Systems*. Springer, Cham, pp. 1-22.

[68] Yilma, B.A., Panetto, H., & Naudet, Y. (2021). Systemic Formalisation of Cyber-Physical-Social System (CPSS): A Systematic Literature Review. *Computers in Industry*, 129, pp. 1-25.

[69] Johnson, B., & Hernandez, A. (2016). Exploring Engineered Complex Adaptive Systems of Systems. *Procedia Computer Science*, 95, pp. 58-65.

[70] Johnson, B. (2020). Intelligent and Adaptive Systems of Systems for Engineering Desired Self-organization and Emergent Behavior. In: *Proceedings of the Future Technologies Conference*, Springer, Cham, pp. 126-146.

[71] Ali, S., Al Balushi, T., Nadir, Z., & Hussain, O. K. (2018). Standards for CPS. In: *Cyber Security for Cyber Physical Systems*. Springer, Cham, pp. 161-174.

[72] Ahmadi, A., Cherifi, C., Cheutet, V., & Ouzrout, Y. (2017). A Review of CPS 5 Components Architecture for Manufacturing Based on Standards. In: *Proceedings of the 11th International Conference on Software, Knowledge, Information Management and Applications*, IEEE, pp. 1-6.

[73] Hohwy, J. (2020). Self-Supervision, Normativity and the Free Energy Principle. *Synthese*, 199(1), pp. 29-53.

[74] Yin, D., Ming, X., & Zhang, X. (2020). Understanding Data-Driven Cyber-Physical-Social System (D-CPSS) Using a 7C Framework in Social Manufacturing Context. *Sensors*, 20(18), p. 5319.

[75] Yilma, B. A., Naudet, Y., & Panetto, H. (2018). Introduction to Personalisation in Cyber-Physical-Social Systems. In: *Proceedings of the OTM Confederated International Conferences on the Move to Meaningful Internet Systems*, Springer, Cham, pp. 25-35.

[76] Klös, V., Göthel, T., Glesner, S. (2018). Runtime Management and Quantitative Evaluation of Changing System Goals in Complex Autonomous Systems. *Journal of Systems and Software*, Vol. 144, October 2018, pp. 314-327.

[77] Elkhodary, A., Esfahani, N., & Malek, S. (2010). FUSION: A Framework for Engineering Self-Tuning Self-Adaptive Software Systems. In: *Proceedings of the Eighteenth ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pp. 7-16.

[78] Cámara, J., Garlan, D., Moreno, G.A., & Schmerl, B. (2017). Evaluating Trade-Offs of Human Involvement in Self-Adaptive Systems. In: *Managing Trade-Offs in Adaptable Software Architectures*, Morgan Kaufmann, pp. 155-180.

[79] Cimini, C., Pirola, F., Pinto, R., & Cavalieri, S. (2020). A Human-in-the-Loop Manufacturing Control Architecture for the Next Generation of Production Systems. *Journal of Manufacturing Systems*, 54, pp. 258-271.

[80] Firouznia, M., Peng, C., & Hui, Q. (2018). Toward Human-in-the-Loop Supervisory Control for Cyber-Physical Networks. arXiv preprint arXiv:1805.02611.

[81] Cogliati, D., Falchetto, M., Pau, D., Roveri, M., & Viscardi, G. (2018). Intelligent Cyber-Physical Systems for Industry 4.0. In: *Proceedings of the First International Conference on Artificial Intelligence for Industries*, IEEE, pp. 19-22.

[82] Finogeev, A., Finogeev, A., Fionova, L., Lyapin, A., Lychagin K. A. (2019), Intelligent Monitoring System for Smart Road Environment, *Journal of Industrial Information Integration*, 15(9), pp. 15-20.

[83] Krishna, C.M., & Koren, I. (2013). Adaptive Fault-Tolerance for Cyber-Physical Systems. In: *Proceedings of the International Conference on Computing, Networking and Communications*. IEEE, pp. 310-314.

[84] Ruíz Arenas, S. (2018). Exploring the role of system operation modes in failure analysis in the context of first generation cyber-physical systems (*Doctoral dissertation*, Delft University of Technology, Delft, the Netherlands).

[85] Eirinakis, P., Kalaboukas, K., Lounis, S., Mourtos, I., Rožanec, J. M., Stojanovic, N., & Zois, G. (2020). Enhancing Cognition for Digital Twins. In: *Proceedings of the International Conference on Engineering, Technology and Innovation*, IEEE, pp. 1-7.

[86] Müller, H.A., Rivera, L.F., Jiménez, M., Villegas, N M., Tamura, G., Akkiraju, R., ... & Erpenbach, E. (2021). Proactive AIOps Through Digital Twins. In: *Proceedings of the 31st Annual International Conference on Computer Science and Software Engineering*, pp. 275-276.

[87] Ferrer, B.R., Mohammed, W.M., Lastra, J.L., Villalonga, A., Beruvides, G., Castaño, F., & Haber, R.E. (2018). Towards the Adoption of Cyber-Physical Systems of Systems Paradigm in Smart Manufacturing Environments. In*: Proceedings of the 16th International Conference on Industrial Informatics*, IEEE, pp. 792-799.

[88] Axelsson, J. (2019). Game Theory Applications in Systems-of-Systems Engineering: A Literature Review and Synthesis. *Procedia Computer Science*, 153, pp. 154-165.

[89] Du, D., Guo, T., & Wang, Y. (2020). DSML4CS: An Executable Domain-Specific Modeling Language for Co-Simulation Service in CPS. *International Journal of Web Services Research*, 17(2), pp. 59-75.

[90] Buffoni, L., Ochel, L., Pop, A., Fritzson, P., Fors, N., Hedin, G., ... & Sjölund, M. (2021). Open Source Languages and Methods for Cyber-Physical System Development: Overview and Case Studies. *Electronics*, 10(8), pp. 902.

[91] Ruchkin, I. (2019). Integration of Modeling Methods for Cyber-Physical Systems (Doctoral dissertation (*PhD thesis*, Institute for Software Research School of Computer Science, Carnegie Mellon University, Pittsburgh, CMU-ISR-18-107).

[92] Merelli, E., Paoletti, N., & Tesei, L. (2012). A Multi-Level Model for Self-Adaptive Systems. arXiv preprint arXiv:1209.1628.

[93] Frank, U. (2014). Multilevel Modeling - Toward a New Paradigm of Conceptual Modeling and Information Systems Design. *Business & Information Systems Engineering*, 6(6), pp. 319–337.

[94] Kühne, A. (2018). Story of Levels. In: *Proceedings of the 5th International Workshop on Multi-Level Modelling*, Volume 2245 of CEUR Workshop Proceedings, pp. 673–682.

[95] Khakpour, N., Jalili, S., Talcott, C., Sirjani, M., & Mousavi, M. (2012). Formal Modeling of Evolving Self-Adaptive Systems. *Science of Computer Programming*, 78(1), pp. 3-26.

[96] Hartsell, C., Mahadevan, N., Ramakrishna, S., Dubey, A., Bapty, T., Johnson, T., ... & Karsai, G. (2019). Model-Based Design for CPS with Learning-Enabled Components. In: *Proceedings of the Workshop on Design Automation for CPS and IoT*, pp. 1-9.
[97] Bonci, A., Pirani, M., Bianconi, C., & Longhi, S. (2018). RMAS: Relational Multiagent System for CPS Prototyping and Programming. In: *Proceedings of the 14th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*, IEEE, pp. 1-6.
[98] Zhang, L., Zeigler, B.P., & Laili, Y. (2019). Introduction to Model Engineering for Simulation. In: *Model Engineering for Simulation*, Academic Press, pp. 1-23.
[99] de Lamotte, F.F., Berruet, P., Rossi, A., & Philippe, J.L. (2008). Control Code Generation using Model Engineering for an Electric Train. *IFAC Proceedings Volumes*, 41(2), pp. 8327-8332.
[100] Fishwick, P.A. (1990). Toward an integrated approach to simulation model engineering. *International Journal of General System*, 17(1), pp. 1-19.
[101] Hashmi, M.A., Mo, J.P., & Beckett, R. (2021). Transdisciplinary Systems Approach to Realization of Digital Transformation. *Advanced Engineering Informatics*, 49, p. 101316.
[102] Mo, J.P., & Beckett, R.C. (2021). Transdisciplinary System of Systems Development in the Trend to X4.0 for Intelligent Manufacturing. *International Journal of Computer Integrated Manufacturing*, pp. 1-15.
[103] van Gigch, J.P. (1993). Metamodeling: The Epistemology of System Science. *Systems Practice*, 6(3), pp. 251-258.
[104] Sangiovanni-Vincentelli, A., Shukla, S.K., Sztipanovits, J., Yang, G., & Mathaikutty, D. A. (2009). Metamodeling: An Emerging Representation Paradigm for System-Level Design. *IEEE Design & Test of Computers*, 26(3), pp. 54-69.
[105] Odell, J., Nodine, M., & Levy, R. (2004). A Metamodel for Agents, Roles, and Groups. In: *Proceedings of the International Workshop on Agent-Oriented Software Engineering*, Springer, Berlin, Heidelberg, pp. 78-92.
[106] Yang, Z., Eddy, D., Krishnamurty, S., Grosse, I., & Lu, Y. (2018). A Super-Metamodeling Framework to Optimize System Predictability. In: *Proceedings of the International Design Engineering Technical Conferences*, Vol. 51722, American Society of Mechanical Engineers, p. V01AT02A009.
[107] Villar, E., Merino, J., Posadas, H., Henia, R., & Rioux, L. (2020). Mega-Modeling of Complex, Distributed, Heterogeneous CPS Systems. *Microprocessors and Microsystems*, 78, p. 103244.
[108] Gašević, D., Kaviani, N., & Hatala, M. (2007). On Metamodeling in Megamodels. In: *Proceedings of the International Conference on Model Driven Engineering Languages and Systems*, Springer, Berlin, Heidelberg, pp. 91-105.
[109] Favre, J. M., & Nguyen, T. (2005). Towards a Megamodel to Model Software Evolution Through Transformations. *Electronic Notes in Theoretical Computer Science*, 127(3), pp. 59-74.
[110] Vogel, T., & Giese, H. (2012). A Language for Feedback Loops in Self-Adaptive Systems: Executable Runtime Megamodels. In: Proceedings of the 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, IEEE, pp. 129-138.
[111] Bayne, J.S. (1995). A Distributed Programming Model for Real-Time Industrial Control. *Control Engineering Practice*, 3(8), pp. 1133-1138.
[112] Bickhard, M.H. (2019). Dynamics Is Not Enough: An Interactivist Perspective. *Human Development*, 63(3-4), pp. 227-244.
[113] Gleizes, M.-P., Camps, V., George, J.-P., & Capera, D. (2007). Engineering Systems Which Generate Emergent Functionalities. In: *Proceedings of the Satellite Conference on Engineering Environment-Mediated Multiagent Systems*. Dresden, Germany, pp. 58-75.
[114] Alcocer, J.P., Beck, F., & Bergel, A. (2019). Performance Evolution Matrix: Visualizing Performance Variations Along Software Versions. In: *Proceedings of the Working Conference on Software Visualization*, IEEE, pp. 1-11.
[115] Garces, L., Martinez-Fernandez, S., Neto, V.V. & Nakagawa, E.Y. (2020). Architectural Solutions for Self-Adaptive Systems. *Computer*, 53(12), pp. 47-59.

[116] Mokni, A., Urtado, C., Vauttier, S., Huchard, M., & Zhang, H.Y. (2016). A Formal Approach for Managing Component-Based Architecture Evolution. *Science of Computer Programming*, 127, pp. 24-49.
[117] Ballesteros, J., Ayala, I., Caro-Romero, J.R., Amor, M., Fuentes, L. (2020), Evolving Dynamic Self-Adaptation Policies of mHealth Systems for Long-Term Monitoring, *Journal of Biomedical Informatics*, pp. x-x.
[118] Zhou, P., Zuo, D., Hou, K.M., Zhang, Z., Dong, J., Li, J., & Zhou, H. (2019). A Comprehensive Technological Survey on the Dependable Self-Management CPS: From Self-Adaptive Architecture to Self-Management Strategies. *Sensors*, 19(5), 1033, x-x.
[119] Epifani, I., Ghezzi, C., Mirandola, R., & Tamburrelli, G. (2009). Model Evolution by Run-Time Parameter Adaptation. In: *Proceedings of the 31st International Conference on Software Engineering*, IEEE, pp. 111-121.
[120] Roth, F.M., Krupitzer, C., & Becker, C. (2015). Runtime Evolution of the Adaptation Logic in Self-Adaptive Systems. In: *Proceedings of the International Conference on Autonomic Computing*, IEEE, pp. 141-142.
[121] Pasquale, L., Baresi, L., & Nuseibeh, B. (2011). Towards Adaptive Systems Through Requirements@Runtime. In: *Proceedings of the 6th Workshop on Models@run.time*, p. 39.
[122] Basich, C., Svegliato, J., Wray, K.H., Witwicki, S., Biswas, J., & Zilberstein, S. (2020). Learning to Optimize Autonomy in Competence-Aware Systems. arXiv preprint arXiv:2003.07745.
[123] Löffler, D., Schmidt, N., & Tscharn, R. (2018). Multimodal expression Of Artificial Emotion In Social Robots Using Color, Motion and Sound. In: *Proceedings of the International Conference on Human-Robot Interaction*, ACM/IEEE, pp. 334-343.
[124] Zhu, Y.B., & Li, J.S. (2019). Collective Behavior Simulation Based nn Agent with Artificial Emotion. *Cluster Computing*, 22(3), pp. 5457-5465.
[125] Catuogno, L., Galdi, C., & Pasquino, N. (2018). An Effective Methodology For Measuring Software Resource Usage. *IEEE Transactions on Instrumentation and Measurement*, 67(10), pp. 2487-2494.
[126] Liu, Q., Li, Q., Li, Z., & Wu, H. (2020). Research on Software Resource Management Model Based on Cloud Service. In: *Proceedings of the 8th International Conference on Information Technology: IoT and Smart City*, pp. 65-68.
[127] Wan, J., Chen, B., Imran, M., Tao, F., Li, D., Liu, C., & Ahmad, S. (2018). Toward Dynamic Resources Management tor IoT-Based Manufacturing. *IEEE Communications Magazine*, 56(2), pp. 52-59.
[128] Hachicha, M., Halima, R.B., & Kacem, A.H. (2019). Formal Verification Approaches of Self-Adaptive Systems: A Survey. *Procedia Computer Science*, 159, pp. 1853-1862.
[129] Nia, M.A., Kargahi, M., & Faghih, F. (2020). Probabilistic Approximation of Runtime Quantitative Verification in Self-Adaptive Systems. *Microprocessors and Microsystems*, 72, p. 102943.
[130] Redfield, S.A., & Seto, M.L. (2017). Verification Challenges for Autonomous Systems. In: *Autonomy and Artificial Intelligence: A Threat or Savior?*, Springer, Cham, pp. 103-127.
[131] Eze, T., Anthony, R.J., Walshaw, C., & Soper, A. (2011). The Challenge of Validation for Autonomic and Self-Managing Systems. In: *Proceedings of the Seventh International Conference on Autonomic and Autonomous Systems*, pp. 128-133.
[132] Dewasurendra, S.D., Vidanapathirana, A.C., & Abeyratne, S.G. (2019). Integrating Runtime Validation and Hardware-in-the-Loop (HiL) testing with V&V in Complex Hybrid Systems. *Journal of the National Science Foundation of Sri Lanka*, 47, 4.
[133] Cardozo, N., Christophe, L., De Roover, C., & De Meuter, W. (2014). Runtime Validation of Behavioral Adaptations. In: *Proceedings of 6th International Workshop on Context-Oriented Programming*, pp. 1-6.
[134] Abuseta, Y., & Swesi, K. (2015). Towards a Framework for Testing and Simulating Self-Adaptive Systems. In: *Proceeding of the 6th International Conference on Software Engineering and Service Science*, IEEE, pp. 70-76.

[135] Eberhardinger, B., Seebach, H., Knapp, A., & Reif, W. (2014). Towards Testing Self-Organizing, Adaptive Systems. In: *Proceedings of the IFIP International Conference on Testing Software and Systems*. Springer, Berlin, Heidelberg, pp. 180-185.

[136] Schierman, J., Ward, D., Dutoi, B., Aiello, A., Berryman, J., DeVore, M., ... & Wadley, J. (2008). Runtime Verification and Validation for Safety-Critical Flight Control Systems. In: *Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit*, p. 6338.

[137] Marshall, A., Jahan, S., & Gamble, R. (2018). Toward Evaluating the Impact of Self-Adaptation on Security Control Certification. In: *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems*, pp. 149-160.

[138] Cámara, J., Peng, W., Garlan, D., & Schmerl, B. (2018). Reasoning about Sensing Uncertainty and its Reduction in Decision-Making for Self-Adaptation. *Science of Computer Programming*, 167, pp. 51-69.

[139] Drechsler, R., Lüth, C., Fey, G., & Güneysu, T. (2018). Towards Self-Explaining Digital Systems: A Design Methodology for the Next Generation. In: *Proceedings of the 3rd International Verification and Security Workshop*, IEEE, pp. 1-6.

[140] Blumreiter, M., Greenyer, J., Garcia, F.J., Klös, V., Schwammberger, M., Sommer, C., ... & Wortmann, A. (2019). Towards Self-Explainable Cyber-Physical Systems. In: *Proceedings of the 22nd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion*, IEEE, pp. 543-548.

[141] Wickramasinghe, C.S., Amarasinghe, K., Marino, D.L., Rieger, C., & Manic, M. (2021). Explainable Unsupervised Machine Learning for Cyber-Physical Systems. *IEEE Access*, 9, pp. 131824-131843.

[142] Daglarli, E. (2021). Explainable Artificial Intelligence (xAI) Approaches and Deep Meta-Learning Models for Cyber-Physical Systems. In: *Artificial Intelligence Paradigms for Smart Cyber-Physical Systems*, IGI Global, pp. 42-67.

[143] Huang, J., Bastani, F., Yen, I.L., & Jeng, J.J. (2009). Toward a Smart Cyber-Physical Space: A Context-Sensitive Resource-Explicit Service Model. In: *Proceedings of the 33rd Annual International Computer Software and Applications Conference*, Vol. 2, IEEE, pp. 122-127.

[144] Schmidt, D.C., White, J., & Gill, C.D. (2014). Elastic Infrastructure to Support Computing Clouds for Large-Scale Cyber-Physical Systems. In: *Proceedings of the 17th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing*, IEEE, pp. 56-63.

[145] González-Nalda, P., Etxeberria-Agiriano, I., Calvo, I., & Otero, M. C. (2017). A Modular CPS Architecture Design Based on ROS and Docker. *International Journal on Interactive Design and Manufacturing*, 11(4), pp. 949-955.

[146] Landolfi, G., Barni, A., Menato, S., Cavadini, F.A., Rovere, D., & Dal Maso, G. (2018). Design of a multi-sided platform supporting CPS deployment in the automation market. In: *Proceedings of the 2018 IEEE Industrial Cyber-Physical Systems*, IEEE, pp. 684-689.

[147] Shin, D.Y., & Park, J.W. (2020). Modularization for Personal Social Service Robots. *The Journal of the Convergence on Culture Technology*, 6(2), pp. 349-355.

[148] Denkena, B., Henning, H., & Lorenzen, L.E. (2010). Genetics and Intelligence: New Approaches in Production Engineering. *Production Engineering*, 4(1), pp. 65-73.

[149] Lachmayer, R., Mozgova, I., & Scheidel, W. (2016). An Approach to Describe Gentelligent Components in Their Life Cycle. *Procedia Technology*, 26, pp. 199-206.

[150] Möhring, H.C., Wiederkehr, P., Erkorkmaz, K., & Kakinuma, Y. (2020). Self-optimizing Machining Systems. *CIRP Annals*, 69(2), pp. 740-763.

[151] Cuevas, A.D., & Guzman-Arenas, A. (2010). Automatic Fusion of Knowledge Stored in Ontologies. *Intelligent Decision Technologies*, 4(1), pp. 5-19.

[152] Zhang, S., Yang, L.T., Feng, J., Wei, W., Cui, Z., Xie, X., & Yan, P. (2021). A Tensor-Network-Based Big Data Fusion Framework for Cyber-Physical-Social Systems (CPSS). *Information Fusion*, 76(2021), pp. 337-354.

[153] Sahu, A., Mao, Z., Wlazlo, P., Huang, H., Davis, K., Goulart, A., & Zonouz, S. (2021). Multi-Source Multi-Domain Data Fusion for Cyberattack Detection in Power Systems. *IEEE Access*, 9, pp. 119118-119138.

[154] Petrovska, A., Quijano, S., Gerostathopoulos, I., & Pretschner, A. (2020). Knowledge Aggregation with Subjective Logic in Multi-Agent Self-Adaptive Cyber-Physical Systems. In: *Proceedings of the 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, IEEE/ACM, pp. 149-155.

[155] Bürger, J., Kehrer, T., & Jürjens, J. (2020). Ontology Evolution in the Context of Model-Based Secure Software Engineering. In: *Proceedings of the International Conference on Research Challenges in Information Science, Springer*, Cham, pp. 437-454.

[156] Kotis, K.I., Vouros, G.A., & Spiliotopoulos, D. (2020). Ontology Engineering Methodologies for the Evolution of Living and Reused Ontologies: Status, Trends, Findings and Recommendations. *The Knowledge Engineering Review*, 35(4), pp. 1-34.

[157] Ferranti, N., Soares, S.S., & de Souza, J.F. (2021). Metaheuristics-Based Ontology Meta-Matching Approaches. *Expert Systems with Applications*, 173, 114578.

[158] Horváth, I. (2020). On Reasonable Inquiry and Analysis Domains of Sympérasmology. *Journal of Integrated Design and Process Science*, 24(1), pp. 5-43.

[159] Gembarski, P.C. (2019). The Meaning of Solution Space Modelling and Knowledge-Based Product Configurators for Smart Service Systems. In: *Proceedings of the International Conference on Information Systems Architecture and Technology*, Springer, Cham, pp. 28-37.

[160] Abel, G. (2014). Systematic Knowledge Research. Rethinking Epistemology. *Rivista Internazionale di Filosofia e Psicologia*, 5(1), pp. 13-28.

Author Biographies

Dr. Imre Horváth is a professor emeritus of the Faculty of Industrial Design Engineering, Delft University of Technology, the Netherlands. In recent years, his research group has focused on the research and education of smart cyber-physical system design, with special attention to cognitive engineering and prototype systems for personal and social applications. Prof. Horváth is also interested in systematic design research methodologies. He was a promoter of more than 25 Ph.D. students. He is the first author or co-author of more than 440 publications. He has received five best paper awards for his scientific work and has a wide range of society memberships and contributions. He was the past chair of the Executive Committee of the CIE Division of ASME. Since 2011, he has been a fellow of ASME. He is a member of the Royal Dutch Institute of Engineers and also involved in the Society for Design and Process Science. He received honorary doctor titles from two universities and the Pahl-Beitz ICONNN award for outstanding international contributions to design science and education. He was conferred the Lifetime Achievement Award by ASME's CIE Division in 2019. He has served several international journals as an editor and initiated the International Tools and Methods of Competitive Engineering (TMCE) Symposia series. He was editor-in-chief of the Journal Computer Aided Design over 12 years. He is an associate editor of the Journal of Engineering Design, a co-editor-in-chief of the Journal of Integrated Design and Process Science, and served more than 10 journals as registered reviewer. His current research interests include various philosophical, methodological, and computational aspects of smart products, systems, and service design, as well as synthetic knowledge science, sympérasmology, and the development of self-evolving systems.

Dr. Jože Tavčar has been working as a senior lecturer at the Product Development Division, LTH, University of Lund, Sweden, since 2020. He earned his B.Sc., M.Sc., and Ph.D. degrees in mechanical engineering from the University of Ljubljana in 1991, 1994, and 1999, respectively. He started his research career in 1991, focusing on technical information systems, information flow in product development, process re-engineering, and design methodology. Starting in 2001, he spent a decade as the Head of the Noise and Vibration Lab at the Domel Company and as the Head of the Quality Department at Iskra Mehanizmi. During this period, he was involved in several product development teams of international corporations such as Philips, Electrolux, and Rowenta. He coordinated the development of a motor diagnostic system and studied various noise reduction and vibration topics. He also developed quality systems for the automotive industry (ISO/IATF 16949) and medical devices (ISO 13485, FDA). Between 2011 and 2020, he was a lecturer at the University of Ljubljana in various courses, such as Design Methodology, Engineering Design Techniques, Engineering Design using Non-Metallic Materials, and Technical Information Systems (PLM). He was a guest researcher at ProSTEP AG, Germany in 1995; at the University of Karlsruhe, Germany in 1996; and at the Delft University of Technology, the Netherlands in 2016. He has a unique combination of concrete product development experience and a holistic view on system approach. His current research topics include designing smart cyber-physical systems, design for additive manufacturing, data mining and big data analytics, and application of agility methods and knowledge management in product development processes. He has published over 40 SCI papers, over 50 conference papers, 5 book chapters, and over 80 technical reports.