

## Designing Cyber-Physical Systems for Runtime Self-Adaptation Knowing More about What We Miss...

Horváth, Imre; Tavčar, Jože

**DOI**

[10.3233/JID210030](https://doi.org/10.3233/JID210030)

**Publication date**

2021

**Document Version**

Accepted author manuscript

**Published in**

Journal of Integrated Design and Process Science

**Citation (APA)**

Horváth, I., & Tavčar, J. (2021). Designing Cyber-Physical Systems for Runtime Self-Adaptation: Knowing More about What We Miss... *Journal of Integrated Design and Process Science*, 25(2), 1-26.  
<https://doi.org/10.3233/JID210030>

**Important note**

To cite this publication, please use the final published version (if applicable).  
Please check the document version above.

**Copyright**

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

**Takedown policy**

Please contact us and provide details if you believe this document breaches copyrights.  
We will remove access to the work immediately and investigate your claim.

# 1 Designing Cyber-Physical Systems for Runtime Self- 2 Adaptation:

## 3 Knowing More about What We Miss ...

4 Imre Horváth<sup>a</sup> and Jože Tav ar<sup>b</sup>

5

6 <sup>a</sup> Professor emeritus, Faculty of Industrial Design Engineering, Delft University of Technology, Delft, the  
7 Netherlands.

8 <sup>b</sup> Senior lecturer at the Product Development Division of the Faculty of Engineering, University of Lund,  
9 Sweden.

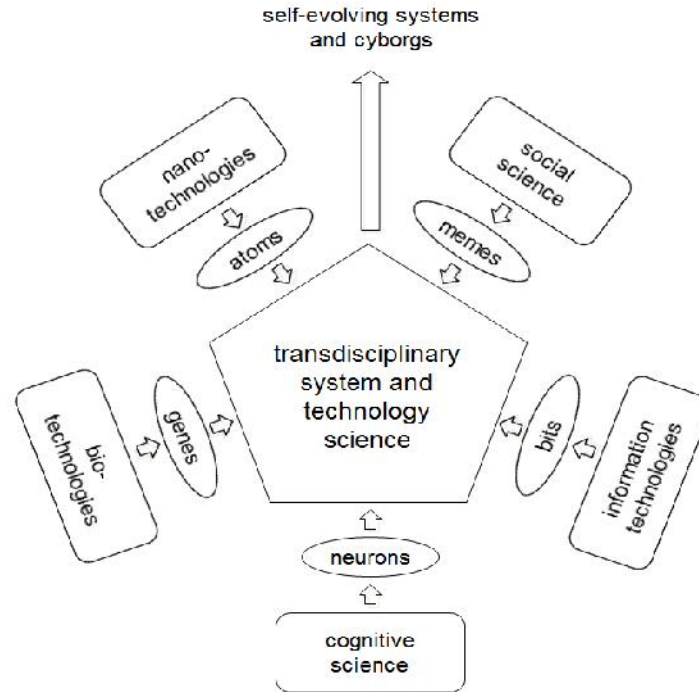
### 10 Abstract

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

### 13 1. First things first – Our view on cyber-physical systems

14 We live in the age of an extensive scientific, technological, and paradigmatic convergence [1]. One of  
15 the strongest current trends is the integration of social science, cognitive science, biotechnologies,  
16 information technologies, and nanotechnology (SCBIN) that enables fusion of bits, atoms, neurons, genes,  
17 and memes [2]. Graphically depicted in **Figure 1**, this accelerating merge process is often referred to as  
18 the bits-atoms-neurons-genes-memes (b.a.n.g.m.) revolution [3]. Cyber-physical systems (CPSs)  
19 represent practical examples of the integration of bits and atoms in human and social contexts, but they  
20 also make steps towards integration of neurons and genes into system implementations [4]. The move  
21 towards integration of neurons is exemplified by the interest in cyber-bio-physical (CBP) systems (e.g.  
22 assistive and corrective implants [5], and artificial limbs/augments [6]), while the results in the latter field  
23 are epitomized by gentelligent systems [7]. Consequently, engineered systems are going through a  
24 metamorphosis, and the significance of purely hardware (HW), software (SW), and cyberware (CW)  
25 systems is shrinking and their places are taken over quickly by heterogeneous and intellectualized systems.  
26 From the perspective of system adaptation, the current trends imply the need for a concurrent change of  
27 the HW, SW, and CW elements in runtime, in a synergic (compositional) manner. Theoretically, but also  
28 practically, the largest challenge in this context is that the operational changes of the HW constituents  
29 happen in the spatial-temporal space, the changes of the SW constituents in the logical-temporal space,  
30 and those of the CW constituents in the syntactic and semantic spaces.

31 In our view, software and data/knowledge integrated cyber-physical systems (CPSs): (i) include one or  
32 more independent (self-contained) or functionally networked actor nodes, (ii) are characterized by a deep  
33 penetration into real-life physical processes, (iii) operate based on multiple sensing-computing-adjusting



**Figure 1:** Merger of technologies and disciplines

34 loops or sensing-reasoning-learning-planning-adapting loops, (iv) provide tailored services and avail  
 35 resources dynamically in human, social, and industrial applications contexts, (v) have abilities to extend  
 36 their problem solving knowledge and computational mechanisms (system intelligence), (vi) may manifest  
 37 as part of a purposefully and synergistically arranged system of systems, and (vii) evolve through  
 38 generations [8]. Cybernetization of complex engineered systems seems to terminate with highly  
 39 intellectualized and autonomously operating, but cognitively and socially embedded systems [9]. If, in  
 40 sociotechnical systems, the technical parts manifest as CPSs, then researchers talk about social-cyber-  
 41 physical systems, whose adaptation may be according to the principles of centrality of the norms and  
 42 policy of autonomy, and not only to operational goals and affordances [10].

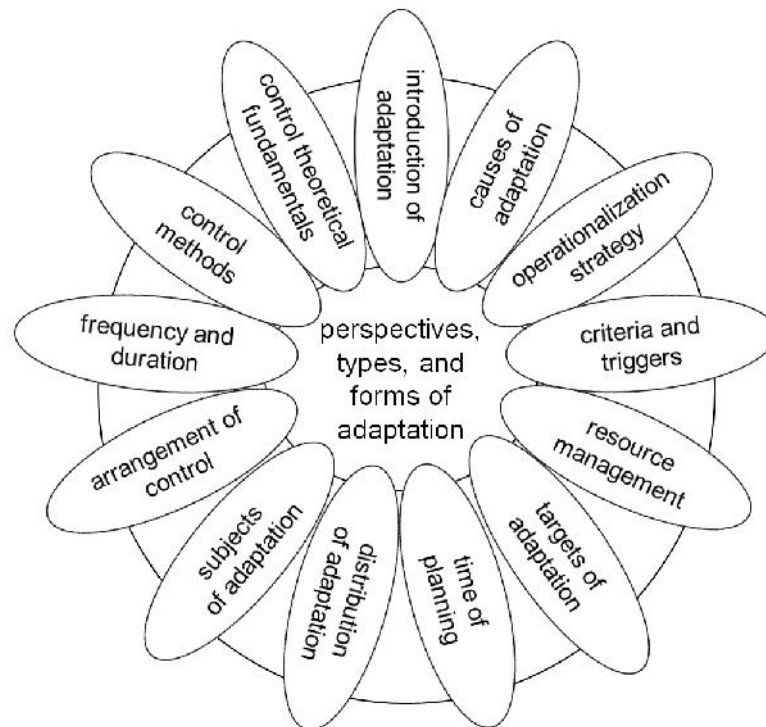
43 Though the complex phenomenon of system adaptation is a current hot issue, it is known only  
 44 partially in the case of complex engineered systems [11]. In the field of biology, adaptation has been  
 45 defined as the process of subsequent changes by which a living organism or a community of organisms  
 46 becomes better suited to its environment and increases its chances to survive [12]. Initially proposed for  
 47 natural systems, this interpretation implies four suppositions: (i) adaptation is towards a goal, purpose, or  
 48 situation, (ii) adaptation is not a one-time action, but a purposeful sequence of changes, (iii) adaptation is  
 49 done by the subjects of the changes themselves, and (iv) adaptation is to be put into the context of  
 50 interaction with the environment or a community of organisms. The same principles have been imposed  
 51 on engineered systems [13]. However, while biological adaptation is based on evolving bio-physiological  
 52 and cognitive mechanisms, there are no *ab ovo* granted or naturally evolving mechanisms in the case of  
 53 engineered systems [14]. Many experts believe that a deeper theoretical understanding of the phenomenon  
 54 of system adaptation will ultimately lead to the opportunity of developing autonomous systems and  
 55 adjustable autonomy.

56 The rest of this extended editorial is organized as follows. Section 2 summarizes the types and forms  
 57 of system control and adaptation, Section 3 introduces the scientific, engineering, and computational  
 58 fundamentals and issues of adaptation of first-generation cyber-physical systems (1G-CPSs). Section 4  
 59 discusses the phenomenon of self-adaptation of second-generation cyber-physical systems (2G-CPSs) and  
 60 its fundamental issues. Section 5 offers a (non-exhaustive) landscape of the concerns related to next-

61 generation cyber-physical systems (NG-CPSs). In addition, it discusses the milestone developments, and  
 62 elaborates on some open questions. Section 6 presents the short synopses of the papers contributed to this  
 63 special issue. Section 7 reflects on the major findings, what we apparently miss, and may consider as  
 64 opportunities for future research.

## 65 2. A brief overview of the types and forms of adaptation of systems

66 Natural evolution and selection of living organisms is a long term and strongly conditioned process.  
 67 The natural adaptation concerns many generations and favours to beings having a higher chance of  
 68 survival and a wide variation of heritable characteristics. Obviously, engineered systems cannot exhibit  
 69 such intricate mechanisms of progression. This is why systems science thinks differently about adaptation  
 70 of such kind of systems. Nevertheless, it assumes the potential and resources of adaptive systems to  
 71 change as well as the influence of the environment on the manifestation of changes. A birds-eye-view  
 72 image of the perspectives of system adaptation is shown in **Figure 2**. In general, four sources of the need  
 73 for adaptation are identified: (i) it is problematic to foreseen all requirements due to broadening and  
 74 complexification of using such systems in the society, (ii) it is difficult to predefine all system operation  
 75 and interaction modes due to growing uncertainties concerning applications and stakeholders, (iii) as a  
 76 consequence of unpredictable incidental effects and changes in the environment, it is difficult to achieve  
 77 overall resilience in the design phase, and (iv) owing to the emerging technological and servicing  
 78 affordances, it is often possible to achieve better performance than that the systems have been  
 79 programmed for. System adaptation can be relative to (i) a generally defined goal, (ii) a specifically  
 80 defined goal, (iii) a partially defined goal, or (iv) a non-defined goal of operation/servicing. Considering  
 81 these, adaptation is a means to (i) serve optimally for a purpose, (ii) maximize the fulfilment of  
 82 operational/servicing goals, (iii) achieve the best relation with the embedding environment, and (iv)  
 83 provide optimal interaction with other systems. In other words, it is about how something fits into  
 84 something else and what efforts does it make towards an overall optimum performance in runtime. The  
 85 action of adaptation may happen within a short operation period or over the entire lifecycle of engineered



**Figure 2:** Perspectives of system adaptation

86 systems [15].

87 The above similarities and differences triggered the interest towards a universal theory of adaptation of  
88 systems, but that is still a work in progress. From a control theoretical perspective the literature discusses  
89 (i) traditional control-based adaptation (PID-like or state-representation driven), (ii) advanced control-  
90 based adaptation (model-predictive, optimization-based, and stochastic), (iii) knowledge-based control  
91 (rule-based, fuzzy, heuristic, and analogical), and reasoning-based control (data-driven, learning-based,  
92 abductive, prognostic, twin-based) mechanisms [16]. It must be emphasized that these kinds of adaptation  
93 apply to complex software systems, rather than to resource-heterogeneous cyber-physical systems. In line  
94 with the current layering of information technological systems, the categories of (i) infrastructural  
95 (hardware and software) resource adaptation, (ii) reusable middleware adaptation, and domain specific  
96 application software adaptation are imposed [17]. In general, the criteria (trigger) for execution of  
97 adaptation may be (i) goal-related, (ii) task-orientated, or (iii) performance-based. Based on the  
98 operationalization of the adaptation agency, (i) reactive (after change event), (ii) active (concurrent with  
99 change event), and (iii) proactive (before change event) control strategies can be distinguished. Feedback-  
100 based control supports reactive strategies, whereas feed-forward control is usually active. Combinations  
101 of feedback and feed-forward control can detect disturbances and adjust the inputs before the disturbance  
102 affects the system outputs. Consequently, this combination implements a proactive strategy and can be  
103 used as a proactive control mechanism.

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

116 With regard to the change of the system's constituents (components), (i) constant resource-based, and  
117 (ii) variable resource-based adaptations are implemented. From the perspective of organization of the  
118 changes (i) centralized and (ii) decentralized approaches are used. The target of adaptation can be (i) goal,  
119 (ii) functions, (iii) architecture, (iv) operation, (v) intellectualization, (vi) interactions, (vii) behaviour, and  
120 (viii) combined adaptation. Furthermore, (i) environment centred adaptation (for a proper interaction with  
121 a dynamic environment) and (ii) system centred adaptation (guaranteeing the dependability of the  
122 states/operations/services) are differentiated.

123 As discussed by Patikirikoralala et al., the control may have single objectives or multiple objectives in  
124 the case of software systems, and (i) basic or (ii) composite control schemes are implemented depending  
125 on the complexity of the objectives [18]. Basic control schemes are such as (i) model-based fixed-gain  
126 control, (ii) model-based runtime dynamic-gain control, (iii) linear quadratic regulator, and (iv) model-  
127 based predictive horizon control. Composite control schemes are, for example, (i) cascaded (nested)  
128 control: (ii) rules-based gain scheduling, (iii) algorithms reconfiguring control, (iv) top-down distributed  
129 (hierarchical) control, (v) decentralized independent control, and (vi) combined event- and time-based  
130 (dynamic) controls [19]. The discussed control strategies are usually put under the conceptual umbrella of  
131 internal control, which means some form of intertwining application functionality (logic) and control  
132 functionality (logic). However, the literature is void concerning (runtime) hardware and cyberware  
133 adaptation issues that are especially important in the case of transforming cyber-physical systems [20].

134 The abovementioned strategies are typically model-based. Models either are predefined in the design-  
135 time, or are generated in runtime. Though current model engineering makes the creation of dynamic  
136 models possible, the range of adaptation is restricted to self-regulation and self-tuning in the case of

control-oriented models. When a fault or an unclear change in the environmental circumstances happens, human intervention is expected. Often, this is referred to as mitigating adaptation. In the case of mitigating adaptation, designers define (i) the specific objectives to achieve, (ii) the boundary conditions of operation, (iii) the conditions of adaptation, (iv) the mechanisms of adaptation, and (v) the appropriateness criteria of adaptation [21]. Another aspect of adaptation is its computational enabling, which can be (i) model-based, (ii) data-driven, awareness-based, and ontology-based enablement. Model-based adaptation strategies involve harmonization of various models such as (i) system models, (ii) control models, (iii) optimization models, (iv) environment models, (v) impact models, and (vi) meta-models.

Internally initiated adaptation is self-adaptation - a form of system operation, for which the goals and rules of adaptation are not provided by external controllers. Traditionally, self-adaptation of systems was defined as the abilities to make appropriate corrective actions based on the information about the actions, which will have the best enhancement impact on the system in 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 according to this to reach the initially or runtime set goal [22]. This multifaceted capability assumes sufficient awareness, reasoning, learning, planning, and decision-making abilities and mechanisms. For many researchers, the core of designing for adaptation is system-level modelling 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 in the design-time of a system.

The above overview of the major adaptation aspects intended to shed light on the complexity of the phenomenon of system adaptation. In addition, it attempted to evidence that the landscape of research and development activities towards the realization of system adaptation is a very broad and varied. Two tangible reasons of 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 first-generation cyber-physical systems

The family of 1G-CPSs include control-intensive plant-type systems for which the primary objective and the logic of operation do not change during the life span. Coordinated control loops are essential to build this kind of adaptive systems, which are actually results of functional enhancement by cyber-physical augmentation (i.e. supplementing physical systems with stand-alone or networked computational platforms) [23]. The interfaces between the physical transformation processes and information computation processes are sensors and effectors (or clusters of these). Another approach to realization of 1G-CPSs is complementing a digital network with physical objects (instruments, devices, robots, vehicles, etc.). This is a typical strategy of the Internet of Things (or Internet Everything) driven development efforts [24]. In view to the capabilities of rapidly progressing higher-level implementations, 1G-CPSs are regarded as low-end cyber-physical systems.

The functionality, architecture, and the logic of operation of the 1G-CPSs are defined in the design phase and they do not change throughout the life span of the system. In other words, this family of CPSs is supposed to adapt to known modes of changes. This assumption makes it possible for the designers to use model-based engineering extensively in their development. Usually, 1G-CPSs systems are equipped with conventional control mechanisms and can regulate the parameters of operation to a known degree through the system model and control model. The end-user can adjust the predefined adaptive control algorithms with some preselected parameters. According to the latest reviews of industrially relevant control strategies, the ones most used in practice are proportional-integral-derivative (PID) control and model-based predictive control (MPC). Such solutions are acceptable for many applications with predictable circumstances and working conditions. However, 1G-CPSs may become unreliable or inefficient in situations that were not predicted in the design phase and they are unable to adapt to.

185 The self-control implemented by 1G-CPSs may appear in multiple forms such as self-regulation, self-  
186 healing, self-resilience or self-tuning. Though these, like self-adaptiveness, are realized typically in a top-  
187 down manner, the literature considers these as a limited sub-set of the capabilities that make CPSs self-  
188 adaptive [25]. The abovementioned capabilities are differentiated also from self-organization that, with a  
189 view to emergent functionalities and to decentralization of their control, works according to a bottom-up  
190 manner. We see self-organization as the mutual adaptation and co-evolution of the initially autonomous  
191 components of systems, namely, the agents. In the view of the related literature, self-organization is the  
192 spontaneous process through which systems emerge and evolve, becoming ever more complex, more  
193 adaptive, and more synergetic [26].

194 Internally initiated control intertwines the logic of application functions and the logic of adaptation  
195 functions. This approach is based on programming language features, such as conditional expressions,  
196 parametrization, and exceptions, in software systems [27]. The sensors, effectors, and adaptation  
197 processes are mixed with the application code. This often leads to poor scalability and maintainability,  
198 and the system is costly to test and maintain/evolve. Using external adaptation engine (or adaptation  
199 manager), external approaches of self-adaptive software system try to avoid these limitations by offering  
200 sophisticated adaptation processes. In addition, it offers reusability (customization and configuration for  
201 different systems) of the adaptation engine, or processes tailored for various applications. An adaptation  
202 engine can implement both closed adaptation (using defined type/number of adaptive actions) and open  
203 adaptation (allowing new software arrangements and behaviors during runtime) [28].

204 Over the years, a dual-aspect solution emerged in the form of the monitor-analyse-plan-execute  
205 (MAPE) approach [29]. This conceptual abstraction and generalization of the external feedback loop-  
206 based type of control realizes an adaptation logic that is significant for several reasons. For example, it: (i)  
207 allows to separate the concerns of fulfilment of the system functionality and the management of self-  
208 control, (ii) facilitates model-based adaptation control, even self-adaptation, by decomposing the control  
209 loop into four specific phases, (iii) supports the extension of control information with knowledge stored in  
210 a knowledge repository, and (iv) creates a methodological bridge between self-control of 1G-CPSs and  
211 self-adaptation of 2G-CPSs. As discussed by Miller, the monitor, analyse, plan, and execute functions  
212 must share knowledge. Hence, this modelling approach is often referred to as MAPE-K [30]. Iglesias and  
213 Weyns proposed to use formally specified MAPE-K templates that encode design expertise for a family  
214 of self-adaptive systems. These includes templates for behavioural specification and modelling the  
215 different components of a MAPE-K feedback loop, as well as property specification templates that  
216 support verification of the correctness of the adaptation behaviours [31].

217 However, the MAPE-K approach is limited in terms of runtime variability, including variable structure  
218 and functionality systems. Furthermore, the issues of verification of adaptation plans before execution and  
219 validation of the results of the completed self-adaptation in context, and the issue of resource generation  
220 and management during the lifecycle of the controlled system were not addressed specifically. Tav ar and  
221 Horváth argued that these functions should be included in the self-adaptation loop and proposed  
222 managing it in four logical steps: (i) planning self-adaptation, (ii) verification before self-adaptation, (iii)  
223 operationalization of self-adaptation, and (iv) validation of self-adaptation, which extends MAPE-K into  
224 MAPVEV-K [32].

225 Having analysed the current research on methods and techniques for designing and engineering of  
226 adaptive software systems, Hidaka et al. argued that effective development of self-adaptive systems could  
227 be achieved through the reuse and adaptation of existing models such as MAPE-K loops [33]. The survey  
228 completed by Muccini et al. (2016) explored that the application layer and the middleware layer (rather  
229 than the communication, service or cloud layer) are the typical levels of system adaptation and that  
230 MAPE-RL (where, RL stands for 'reason and learn'), agents, and self-organization are the dominant  
231 adaptation mechanisms [34]. Among others, these functions are seen as crucial elements for self-  
232 supervised self-adaptation of cyber-physical systems.

233 Chandra et al. (2016) analysed and compared architecture frameworks currently proposed for  
234 designing self-adaptive systems. The analysis included (i) the observe-decide-act (ODA), (ii) the MAPE-  
235 K, (iii) the autonomic computing paradigm (ACP), and (iv) the observer/controller architecture (OCA)

236 frameworks, which are rooted in organic computing research and are intended for different types of  
237 distributed systems, such as swarms, systems-of-systems, crowd computing arrangements, computing  
238 entity populations, and multi-agent systems [35].) As a typical example of demand-enabled system  
239 adaptation, Hummaida et al. (2016) presented a resource management strategy for clouds (allocation of a  
240 shared pool of configurable computing resources) [36]. As a concluding remark, we may claim that, in  
241 spite of the efforts, only useful pieces of an incomplete theory of system-level self-control of real life  
242 systems are available and those do not include the agency of (intuitive and creative) heuristics, or  
243 metaheuristics, that helps solve a wide variety of application problems.

#### 244 **4. Systems science, engineering, and computational fundamentals of self-adaptation of** 245 **second-generation cyber-physical systems**

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

257 In principle, the goal of self-adaptation can be either adapting the environment to maintain the targeted  
258 performance of the system, or adapting the system operations according to the environmental changes, or  
259 both in combination. Conventionally, adaptive systems are pre-programmed to realize the adaptation logic  
260 by means of closed feedback loops, while self-adaptive systems are pre-programmed to find a possible, or  
261 the relative best adaptation logic by sophisticated computational mechanisms such as learning, reasoning,  
262 and abstracting [37]. In the case of self-adaptation, on the one hand, the designers define (i) the overall  
263 objectives to achieve, (ii) the overall operational processes, (iii) the possible resources of adaptation, and  
264 (iv) the scenario of realizing possible adaptations. On the other hand, the system decides on: (v) the  
265 necessity of adaptation, (vi) the resources to be used for adaptation, (vii) the concrete procedures of  
266 adaptation, and (viii) the execution of adaptation. In the case of self-adaptive systems, it is possible to  
267 separate the parts of the system that deal with application concerns (i.e. the goals for which the system is  
268 built) from the parts that deal with the self-adaptation concerns. Though this separation is useful for  
269 system engineering and computational reasons, the application-oriented subsystem and the control-  
270 orientated subsystem are supposed to operate in a synergetic functional coupling. Approaching from a  
271 computational perspective, 2G-CPSs may exploit (i) search-based techniques, (ii) logical and uncertain  
272 reasoning techniques, and (iii) machine learning techniques to deal with unanticipated requests and  
273 uncertainties, and preparation for change. By doing so, they implement various forms of autonomic  
274 computing [38].

275 Self-adaptation of (heterogeneous) CPSs is a more complicated task than that of self-adaptation of  
276 software systems. One obvious reason is that the control software should adapt not only itself, also the  
277 hardware and cyberware constituents. Another reason is that that planning of the adaptation needs  
278 comprehensive context management. Many researchers see self-adaptation as a risk mitigation strategy  
279 with regard to the uncertainties caused by runtime changes on the application-oriented subsystem. There  
280 is still a knowledge gap with regard to handling real-time changes and constraints accounting for context  
281 variability. Rodrigues et al. combined off-line requirements and model checking with on-line data  
282 collection and assessment to guarantee the system's goals by fine-tuning the adaptation policies towards  
283 optimization of quality attributes [39]. Engelenburg et al. provided a method to identify what elements of  
284 the environment are relevant context, which involves three steps: (i) getting insight into context, (ii)



285 determining what components are needed to sense and adapt to context, and iii) determining the rules for  
286 how the system should adapt in different situations [40]. Since not only static context but also dynamic  
287 context is to be managed in specific applications, Don et al. proposed an event-driven awareness  
288 mechanism [41]. Another source of complication is that, beyond the change of the operational parameters,  
289 self-adaptation extends to changing elements of the system functionality (operations) and the system  
290 architecture (configuration and relations of components) in the runtime. Towards the orchestration of  
291 these, Braberman et al. proposed a reference architecture that allows for coordinated yet transparent and  
292 independent adaptation of system configuration and behaviour [42]. Cansado et al. proposed a formal  
293 framework that unifies behavioural adaptation and structural reconfiguration of components and showed  
294 the advantages in the context of reconfiguration of a client/server system in which the server has been  
295 replaced [43].

296 It is well known by the software engineering community that the term 'architecture' refers to the  
297 conceptual model that defines the behaviour, structure, and characteristics of a software system that fulfils  
298 the given requirements. In software engineering, architecture is a bridge between requirements and  
299 computational codes [44]. It is conceived also as a formal description of the integrated, distributed, or  
300 hybrid arrangement and interconnection of the functional components. Involving qualitative judgment,  
301 architectural adaptation is a multi-faceted issue and implies modification on various levels [45].  
302 Understanding its guiding principles and possible forms is a central topic for research in self-adaptive  
303 systems. Villegas et al. posited that, besides the regular functional components of the system, the  
304 designed architectures must include components that enable self-awareness capabilities, such as  
305 monitoring and analyzing its own current state, as well planning and executing self-adaptation actions  
306 [46]. There are different possibilities for runtime architectural self-adaptation of composable and  
307 compositional systems. Kramer and Magee outlined a three-layer architectural reference model that  
308 provides the required level of abstraction and generality for self-management of composable architectures  
309 [47].

310 Compositional adaptation exchanges algorithmic or structural system components with others that  
311 improve the fit of the software to the state its current environment. Phan and Lee proposed a  
312 compositional multi-modal approach to model, analyse, and design adaptive CPS on a distributed  
313 architecture that facilitates adaptiveness, efficient use of resources, and incremental integration [48].  
314 Compositional adaptation is powerful, but its use without appropriate tools to automatically generate and  
315 verify code may negatively affect system integrity and security [49]. Compositional self-adaptation  
316 control systems should consider both static aspects (such as stability and availability) and dynamic  
317 properties (such as functional interconnections and transient change of variables). The dependable  
318 emergent ensembles of components (DEECo) framework, presented by Masrur et al., (i) allows modelling  
319 large-scale dynamic systems by a set of interacting components, (ii) provides mechanisms to describe  
320 transitory interactions between components, and (iii) supports reasoning about timing behaviour of the  
321 interacting components [50]. The motivation came from the hypothesis that components may  
322 automatically configure their interactions within self-managed software architectures in a way that is  
323 compatible with the overall architectural specification and can achieve the goals of the system. Another  
324 dimension of self-adaptation is self-adaptation of system of systems that is still in a premature stage of  
325 understanding and implementation [51].

326 Using models as the basis of self-adaptation is both a theoretical issue and a methodological one. The  
327 latter is concerned with the dynamic generation and adaptation of system and control models. Runtime  
328 models are based on abstractions of the system, while the goals serve as a driver and enabler for semi-  
329 automatic reasoning about system adaptations during operation [52]. Many researchers emphasized the  
330 role of software models at runtime (M@RT) as an extension of the adaptation control techniques to  
331 runtime contexts [53]. For instance, a key challenge for self-adaptive software systems is assurance. Some  
332 of the assurance tasks need to be performed at runtime. Towards this end, Cheng et al. argued that  
333 research into the use of M@RT is fundamental to the development of runtime assurance techniques and  
334 presented what information may be captured by M@RT for the purpose of assurance [54]. Bennaceur et

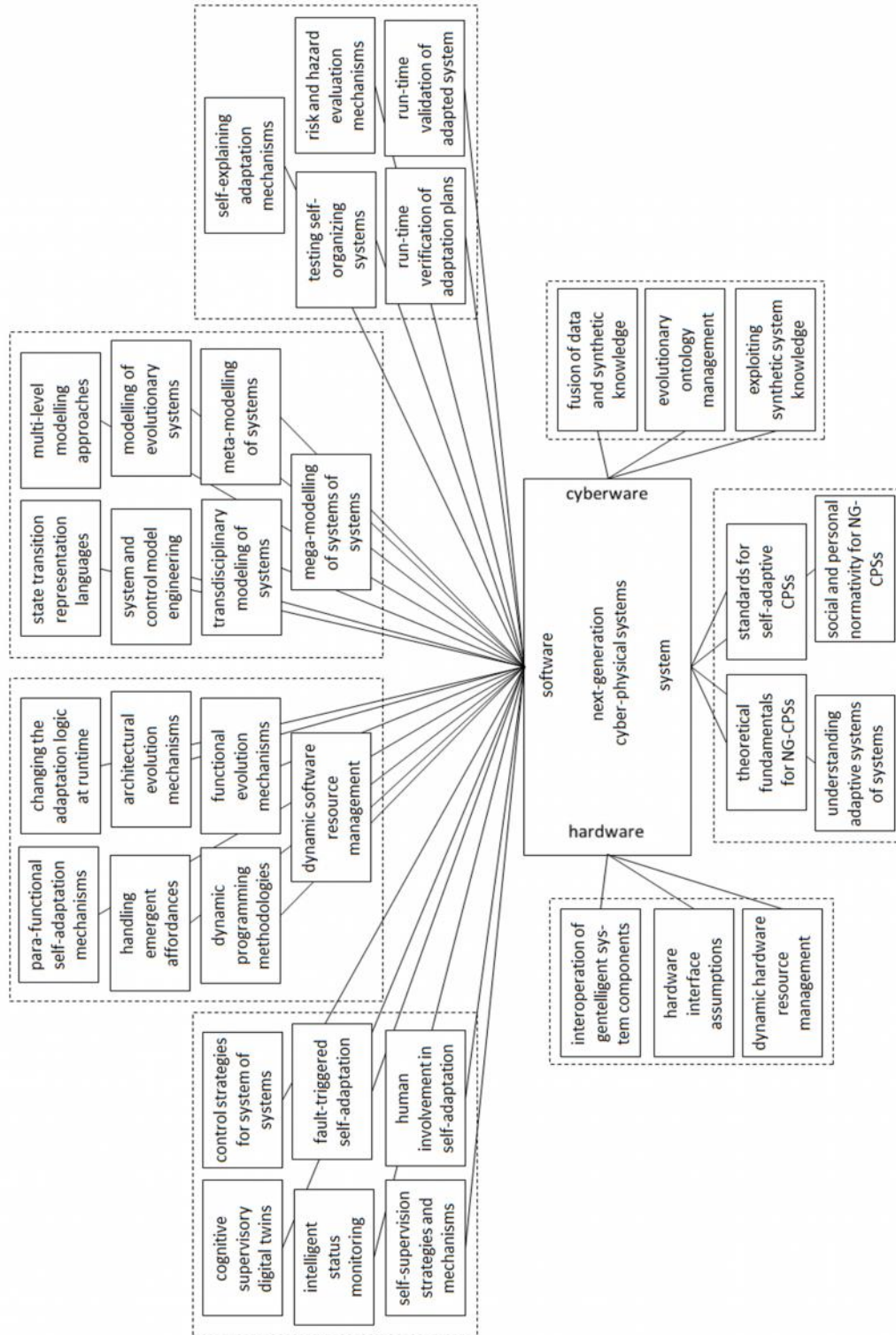
335 al. developed a four-layer partially causal conceptual M@RT reference model to provide a framework for  
336 the core concepts and to situate the computational mechanisms [55].

337 Klös et al. extended the MAPE-K feedback loop architecture by imposing requirements and a structure  
338 on the knowledge base and introducing a meta-adaptation layer. This enables (i) learning new adaptation  
339 rules based on executable runtime models, (ii) continuous evaluation of the accuracy of previous  
340 adaptations, and (iii) verification of the correctness of the adaptation logic in the current system context  
341 [56]. Hadj-Kacem constructed a formal model using a coloured Petri-net for an adaptive system to be  
342 trusted after adaptation. This model has sufficient abstraction of details, but still deal with the core of the  
343 protocol. This makes the model simpler and the analysis easier due to restricted state space size [57]. Also  
344 of theoretical significance is the three-phase approach for modeling and developing dynamically adaptive  
345 systems based on the combination of the runtime models technique and the aspect-oriented software  
346 development paradigm proposed by Loukil et al. The architecture of the software is specified in the first  
347 phase, the executable code is automatically generated in the second phase, and the running system is  
348 reconfigured and supervised in the third phase [58].

349 It is an intensifying trend to use artificial narrow intelligence techniques (in particular deep learning  
350 and machine learning) and fully-fledged digital twins in various runtime activities of system self-  
351 adaptation and dependable automation [59]. This on-going intellectualization concerns both the tasks  
352 related to solving application problems and the tasks related to self-adaptation and self-supervision related  
353 [60]. In both respects, both theoretical and practical issues are addressed. Integration of awareness  
354 building, machine learning, and ampliative reasoning mechanisms into software makes them capable to  
355 behave smartly and to handle not anticipated situations [61]. The latter efforts are justified by the growing  
356 need to autonomously detect and manage unanticipated or unknown situations and to plan the adaptation  
357 during runtime properly [62]. The inclusion of learning mechanisms in self-adaptive systems improves  
358 not only their flexibility, but also their reusability [63]. However, current computational learning allows  
359 self-adaptive CPSs to change their operation and/or configuration only up to specific limits or inside a  
360 goal-defined operational envelope. Furthermore, not only constrains, but also the usable resources are  
361 defined in the design phase [64]. Nevertheless, these technological augmentations of 2G-CPSs (i) transfer  
362 discrete functional and architectural adaptation approaches into a continuous (perpetual) self-adaptation,  
363 (ii) reduce reliance on human supervisors and increase the level of automation, and (iii) enhance the  
364 technological readiness for resource sensitive evolutionary self-adaptation. Three major issues are (i) the  
365 purpose-driven selective learning, (ii) the trustworthiness of the learnt data- and rule-driven models, and  
366 (iii) the scalability of the proposed solutions. Therefore, many researchers encouraged to gain experiences  
367 with industrial systems and applications [65].

## 368 5. Towards Next-Generation Cyber-Physical Systems

369 We made a (non-exhaustive) literature study with the intention to get insights in: (i) the trends of  
370 current research, (ii) the probable future developments, and (iii) the recognizable research/knowledge  
371 gaps in the field of next-generation cyber-physical systems (NG-CPSs). We focused on those seminal  
372 publications that presented front-end and road-paving research and development results, critical and  
373 conclusive overviews, or evidenced personal viewpoints. An important observation was that only a small  
374 portion of the studied journal articles and conference papers looked ahead to future CPSs, though the  
375 number of the related publications progressively increased in the last decade. Based on the selected  
376 publications, we attempted to sketch up a landscape of the major concerns of research and development  
377 towards NG-CPSs. Towards this end, we imposed an initial classification of the concerns according to  
378 what they were related to. The four categories of concerns were: (i) (holistic) system concerns ( ), (ii)  
379 software concerns (S), (iii) hardware concerns (H), and cyber concerns (C). We divided the system  
380 concerns into two sub-categories: (i) generic system concerns (  $\mathcal{C}_1$  ), and (ii) system supervision concerns  
381 (  $\mathcal{C}_2$  ). The obtained landscape is shown in **Figure 3**. Due to the abundance of the associated concerns, we  
382 allocated the software concerns to three sub-categories: (i) system modelling concerns (S<sub>1</sub>), (ii) software



**Figure 3:** Major concerns of next-generation cyber-physical systems

383 self-evolution concerns ( $S_2$ ), and (iii) software dependability concerns ( $S_3$ ). The sub-categories were  
 384 decomposed further into concern domains in the following way:

- 385 **1 Generic system concerns:**  
 386 (i) theoretical fundamentals for NG-CPSs [66] [67] [68], (ii) understanding adaptive systems of  
 387 systems [69] [70], (iii) standards for self-adaptive CPSs [71] [72], and (iv) social and personal  
 388 normativity for NG-CPSs [73] [74] [75].
- 389 **2 System supervision concerns:**  
 390 (i) self-supervision strategies, frameworks, and mechanisms [76] [77], (ii) human involvement in  
 391 self-adaptation [78] [79] [80], (iii) intelligent status monitoring [81] [82], (iv) fault-triggered self-  
 392 adaptation [83] [84], (v) cognitive supervisory digital twins [85] [86], and (vi) control strategies for  
 393 system of systems [87] [88].
- 394 **S<sub>1</sub> System modelling concerns:**  
 395 (i) state transition representation languages [89] [90] [91], (ii) multi-level modelling approaches [92]  
 396 [93] [94], (iii) modelling of evolutionary systems [95] [96] [97], (iv) system and control model  
 397 engineering and optimization [98] [99] [100], (v) transdisciplinary modelling of systems [101] [102],  
 398 (vi) meta-modelling of systems [103] [104] [105] [106], and (vii) mega-modelling of systems of  
 399 systems [107] [108] [109] [110].
- 400 **S<sub>2</sub> Software self-evolution concerns:**  
 401 (i) dynamic programming methodologies [111] [112], (ii) functional evolution mechanisms [113]  
 402 [114], (iii) architectural evolution mechanisms [115] [116], (iv) handling emergent affordances [117]  
 403 [118], (v) changing the adaptation logic at runtime [119] [120] [121], (vi) para-functional self-  
 404 adaptation mechanisms [122] [123] [124], and (vii) software resource management [125] [126] [127].
- 405 **S<sub>3</sub> Software dependability concerns:**  
 406 (i) run-time verification of self-adaptation plans [128] [129] [130], (ii) run-time validation of self-  
 407 adapted systems [131] [132] [133], (iii) testing self-organizing systems [134] [135], (iv) risk and  
 408 hazard evaluation mechanisms [136] [137] [138], (v) self-explaining adaptation mechanisms [139]  
 409 [140] [141] [142].
- 410 **H<sub>1</sub> Hardware management concerns:**  
 411 (i) dynamic hardware resource management [143] [144], (ii) hardware interface assumptions [145]  
 412 [146] [147], and (iii) interoperability of intelligent system components [148] [149] [150].
- 413 **C<sub>1</sub> Cyberware management concerns:**  
 414 (i) fusion of data and synthetic knowledge [151] [152] [153] [154], (ii) evolutionary ontology  
 415 management [155][156][157]), and (iii) exploiting synthetic system knowledge [158] [159] [160].

416 The references included above are only examples of typical publications orientated to the particular  
 417 concern domains. It must be fairly mentioned that the landscape shown in Figure 3 is probably incomplete  
 418 and subjective. The reasons of incompleteness are multiple. For instance, our literature analysis could  
 419 cover only a limited set of the abundant amount of relevant publications. Due to the obvious space  
 420 limitations, even less could be included in the above overview. It was also a technical issue that several  
 421 studied papers addressed multiple concerns or intended to contribute to multiple concern domains. We  
 422 made an attempt to sort them in the most relevant category. In addition to our personal interpretations,  
 423 views, and judgments, this also contributed to the subjective nature of the landscape. We have not done  
 424 any further research yet to validate its comprehensiveness and appropriateness, and to consolidate it in a  
 425 broader application context.

426 Notwithstanding these issues, the presented landscape is deemed a starting point for further  
 427 discussions and analyses. It can be observed that the number of concerns related to hardware, software,  
 428 and cyberware categories are largely different. The overwhelming majority of them are related to  
 429 software that plays multiple roles (such as integrator, driver, processor, mechanism, manager, and utility)  
 430 in current and future CPSs. The landscape also reflects certain trends, which are summarized in the  
 431 Conclusions section of this extended editorial. In the next section, we use it to position the contributed

432 papers in the most relevant concern domain.

## 433 **6. Short Synopses of the Contributed Papers**

434 This special issue is based on an open Call for papers initially presented on the journal's website. The  
435 Call attracted the attention of many potential authors. The selection of the submitted manuscripts for the  
436 peer review process and, after that, the best ones for publication was not a simple task. There were  
437 excellently written papers addressing somewhat conventional topics, and there were less well-elaborated  
438 papers addressing novel and essential topics. Concerning the whole of the submitted manuscripts, it is fair  
439 to mention that there was only a weak thematic coherence among them. For the above reasons, less than  
440 half of the reviewer papers could be considered for publication. It means that, in the end, six original  
441 contributions have been included in this special issue. Based on their actual objectives and contributions,  
442 these papers can be arranged into three general groups: (i) road-mapping for systems science and  
443 engineering (P1 and P2), (ii) methodological approaches to designing self-adaptive systems (P3 and P4),  
444 and (iii) enablers for realization of self-adaptive systems (P5 and P6). Below we briefly introduce these  
445 high quality papers.

446 The first paper, submitted by Danny Weyns, Jesper Andersson, Mauro Caporuscio, Francesco  
447 Flammini, Andreas Kerren, and Welf Lowe, proposes "*A Research Agenda for Smarter Cyber-Physical*  
448 *Systems*". This paper contributes to the conceptual framing and understanding of several concerns  
449 domains in the sub-categories of software self-evolution and software dependability concerns, as well as  
450 in the sub-categories of generic system concerns and system supervision concerns, and provides a broad  
451 and deep theoretical underpinning for next-generation cyber-physical systems. The work complements the  
452 existing perspectives on system smartness by taking a more holistic perspective that integrates systems  
453 operation with the processes to engineer them. The authors argue that both systems and the way they are  
454 engineered must become smarter. Systems and engineering processes must adapt themselves, and evolve  
455 based on stakeholders' input and from experience through a perpetual process that continuously improves  
456 their capabilities and utility to deal with environmental and operational uncertainties and amounts of data  
457 they face throughout their lifetime. The authors highlight key engineering areas (cyber-physical systems,  
458 runtime self-adaptation, data-driven technologies, and visual analytic reasoning), and outline some major  
459 challenges in each of them. They explain the synergies between these key areas. The second part of the  
460 paper presents the authors' proposal for a comprehensive research agenda. This addresses three themes: (i)  
461 assurances for unknowns (in the case of decentralised and smarter cyber-physical-systems that operate  
462 under uncertainty), (ii) self-explainability of autonomous decisions (concerning a lifelong self-learning  
463 and self-explainable cyber-physical systems), and (iii) smarter ecosystems for perpetual adaptation and  
464 evolution (including a unified modelling approach and self-governance for smarter cyber-physical  
465 systems). Exhibiting a high-level of autonomy, smarter cyber-physical ecosystems require reflective  
466 capabilities based on which they data about their utility and adjust according to their shifting operational  
467 goals. Recognizing the necessity of convergence, the research agenda calls for a multi-year concerted  
468 effort of research teams active in the different key areas of studying and developing novel solutions for  
469 trustworthy and sustainable cyber-physical systems.

470 The second paper, entitled "*Designing Runtime Evolution for Dependable and Resilient Cyber-*  
471 *Physical Systems Using Digital Twins*", presents the work and the results of Luis F. Rivera, Miguel  
472 Jimenez, Gabriel Tamura, Norha M. Villegas, and Hausi A. Muller. The main contribution of this paper  
473 belongs to the concern domain of cognitive supervisory digital twins in the system supervision concerns  
474 sub-category, but it also adds to the sub-category of software self-evolution concerns, more specifically,  
475 to concern domain of functional/architectural evolution mechanisms, and to the self-supervision strategies,  
476 frameworks, and mechanisms concern domain. The authors emphasize that designing of smart cyber-  
477 physical systems must address not only dependable autonomy, but also operational resiliency. Their goal  
478 was to implement reliable self-adaptation and self-evolution mechanisms and to include them in the  
479 design of SCPS. Their results are threefold: (i) a reference architecture for designing dependable and  
480 resilient SCPS that integrates concepts from the fields of digital twins, adaptive controls, and autonomic

481 computing, (ii) a model identification mechanism to guide self-evolution, evolutionary optimization, and  
482 dynamic simulation, and (iii) a gradient descent-based adjustment mechanism for self-adaptation to  
483 achieve operational resiliency. In addition to the model identification and the adjustment mechanisms, a  
484 featured contribution of this work is a so-called ‘reference architecture’ for designing digital twin-based  
485 autonomic control for dependable and resilient cyber-physical systems. The authors implemented  
486 prototypes and showed their viability using real data from a case study in the domain of intelligent  
487 transportation systems. The proposed execution adjustment mechanism finds appropriate control  
488 parameters so that the controller can enforce the control objectives in the CPS.

489 The next paper was submitted by Camille Salinesi, Asmaa Achtaich, Nissrine Souissi, Raul Mazo,  
490 Ounsa Roudies, and Angela Villota, under the title: “*State-Constraint Transition: A Language for the*  
491 *Formal Specification of Self-Adaptive Requirements*”. It offers a methodological approach to designing  
492 self-adaptive systems. The main contribution covers the concern domain of dynamic programming  
493 methodologies in the sub-category of software self-evolution concerns, and the concern domain of state-  
494 transition representation languages in the sub-category of system modelling concerns. The observation of  
495 the authors was that existing formal languages focus on the fulfilment of the users’ requirements by the  
496 designed system in the current context. However, they hardly consider runtime dynamically emerging  
497 requirements and context-sensitive requirements. Therefore, the authors introduced a state-constraint  
498 transition (SCT) modelling language to provide a solution to the problem of specifying dynamic  
499 requirements. An essential feature of this solution is the concept of configuration states, in which  
500 requirements are translated into constraints. The paper explains both the syntax and semantics of SCT and  
501 provides examples for reconfiguration scenarios. The authors realized the SCT requirement specification  
502 process relying on the finite-state machines (FSM) approach that provided the necessary computational  
503 power and expressiveness for constraint programming. Their preliminary evaluation explored both the  
504 benefits (expressiveness, scalability, domain independence) and the limitations (temporal constraints,  
505 scheduled reconfigurations, and validation of constraints) of SCT.

506 The fourth paper, entitled “*One-of-a-Kind Production in Cyber-Physical Production Systems*  
507 *Considering Machine Failures*”, presents the results of Guido Vinci Carlván and Daniel Alejandro  
508 Rossit. Though the topic of the paper is broader than a software concern, its scientific contribution can be  
509 related to the concern domain of ‘advanced control strategies for system of systems’ in the sub-category of  
510 ‘system supervision concerns’. Within customized production, the one-of-a-kind production (OKP)  
511 paradigm is the extreme case for production control and scheduling. Cyber-physical systems used in  
512 Industry 4.0 are supposed to facilitate the management of information related to each singular product, as  
513 well as the resolution of conflicts that may arise in processes with a very high variability. That is the  
514 reason why the authors studied the implementation of the constant work-in-progress (CONWIP) control  
515 logic in OKP systems from the perspective of productive job shop configurations in Industry 4.0  
516 environments. The CONWIP control logic was able to handle the challenging Industry 4.0 problem in an  
517 efficient manner, with a relatively low need of investment in CPS related equipment. However, they also  
518 found that the performance is sensitive to the stress of the scenario, i.e. the arrival rate of jobs - an issue  
519 closely related to the used dispatching rules. The general conclusion of the authors was that dispatching  
520 rules associated with due dates tend to improve the overall performance of the system, and the first-in,  
521 first-out (FIFO) rule has the worst performance in all experiments. Essential feature of their work is that  
522 simulation-based experimental studies were developed and their results have been compared  
523 systematically. As design concerns of the next-generation cyber-physical systems, Carlván and Rossit  
524 elaborated upon on intelligent status monitoring, fault-triggered self-adaptation, and system and control  
525 model engineering.

526 The title of the fifth paper is: “*Remote Runtime Failure Detection and Recovery Control for*  
527 *Quadcopters*”. The authors, Sajad Shahsavari, Mohammed Rabah, Eero Immonen, Mohammad-Hashem  
528 Haghbayan, and Juha Plosilab identified managing failures as a basic enabler for realization of  
529 dependable self-adaptive systems, such as quadcopter drones. This work contributes to the concern  
530 domains of fault-triggered self-adaptation and cognitive supervisory digital twins in the system  
531 supervision concerns sub-category. The authors implemented a distributed control system that includes: (i)

532 a local on-board PID-based control sub-system responsible for manoeuvring the drone in all conditions,  
533 (ii) a remote control sub-system responsible for detecting normal or failure states of the drone and  
534 communicating with the drone in real time, and (iii) a digital twin co-execution sub-system responsible  
535 for a real-time two-way data exchange between the above sub-systems. The measured RPM values of the  
536 quadcopter's motors are transmitted to a remote computer, which hosts the failure detection and recovery  
537 software platform. The control concept was implemented using the Simulink tool. The authors propose a  
538 modification of the Quad-Sim simulation model to represent motor failure situations. In addition, they  
539 offer a fast fault detection and recovery technique capable to work at run-time, and a two-way data-stream  
540 management facility. The experimental results obtained by using the MCX co-execution platform show  
541 the applicability and efficiency of the proposed approach in detecting failures and safely landing drones  
542 after failure detection.

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

## 562 **7. Some Conclusions about What We Miss ...**

563 Though significant progress has been achieved both in the research and development and in the  
564 theories and practices, there are still many open issues and unanswered questions. As our above analysis  
565 showed, this can be attributed to the extreme rapid shifts in the research phenomena and the academic  
566 interests. Below we attempt to pinpoint the open issues that are expedient to get resolved on a short notice:

- 567 1. Second-generation cyber-physical systems are based on a balanced utilization of hardware, software,  
568 and cyberware resources. Nevertheless, most of the research efforts focus on software challenges and  
569 issues. This can be explained by the dominance of research in information processing and smart  
570 reasoning systems, but self-adaptation of transformative (such as production, robotic, medical, and  
571 transportation) 2G-CPSs require sophisticated hardware and cyberware resource management  
572 potentials. Publications on their integral theoretical fundamentals and methodological approaches are  
573 scarce in the current literature.
- 574 2. As explained above, a functional motivation for self-adaptation is enabling systems to handle  
575 operational uncertainties that were difficult to foresee before deployment. At the same time, a non-  
576 functional motivation for self-adaptation is freeing system operators and administrators from the  
577 need of continuously monitoring and adjusting systems operating round-the-clock. Self-adaptation  
578 may introduce various levels of transformative operations such as (i) self-tuning, (ii) self-adaptation,  
579 (iii) self-conversion, and (iv) self-reproduction. In all cases, self-adaptive systems are inherently

580 nonlinear, as they possess parameters that are functions of their states and conditions. Thus, self-  
581 adaptive systems are simply a special class of nonlinear systems that either measure their own  
582 performance, operating environment, and operating conditions of the components and adapt their  
583 dynamics or those of their operating environments to ensure that measured performance is close to  
584 targeted performance or specifications.

- 585 3. Facilitating systems' self-evolution and reaching autonomy seem to be two dominant tracks of  
586 developing next-generation cyber-physical systems. Adaptation turns to evolution when new  
587 resources are provided for a system runtime. Functional evolution and evolutionary adaptation  
588 assume extending the system resources (hardware, software, and cyberware) in runtime and adapting  
589 the system objectives, operation, performance, and relationships accordingly the obtained  
590 affordances. Autonomous adaptation has been interpreted as self-adaptation without any form of  
591 human interaction. In this case, the system itself is responsible for self-supervising the both the  
592 planning and the execution of adaptation, considering all risk factors and implications. The current  
593 literature offer neither robust underpinning theories, nor structured methodologies for evolutionary  
594 and autonomous self-adaptation.
- 595 4. Artificial narrow intelligent techniques (in particular, various mechanisms of computational learning)  
596 are increasingly used in self-control and self-adaptation of second-generation cyber-physical systems.  
597 Artificial neural network-based and other AI-based controller mechanisms extend the self-adaptation  
598 potential with additional functionality, but are not able to adapt to frequent requirements changes at  
599 runtime or to scale up to complex real life situations. Sections 2- 5 hinted at some open design issues  
600 that cannot be resolved since the knowledge they need is partly or entirely not available. To explore  
601 the knowledge gaps and eliminate the knowledge deficiencies, first the problems are to be correctly  
602 identified. Cognitive engineering will play an important role with regard to next generation systems.
- 603 5. Dynamic management of the operational and servicing goals of systems based on runtime emerging  
604 requirements is recognized as important topic for further studies, but dynamic development of goal  
605 models it is still in its infancy. The changes during the software lifecycle lead to software  
606 architecture erosion and make the management of software architecture evolution a complex task.  
607 Most existing computational approaches to architecture evolution enable evolution of early stage  
608 models only and fail to support the whole lifecycle of component-based software.
- 609 6. The fundamental mechanisms of automatic runtime (fine-)tuning of the adaptation logic to  
610 unanticipated conditions, runtime verification of adaptation plans, learning the impact of adaptation  
611 decisions on the goals of the system, and validation and testing the performance of self-adaptive  
612 systems after (multiple) adaptations are still concerns for research and development. These are  
613 especially relevant issues for networked times 2G-CPSs and mission critical systems.
- 614 7. A rapid shift can be observed in the literature from self-adaptive systems to self-supervised self-  
615 evolving systems, without providing complete solutions for the self-adaptation problem. The idea of  
616 layering was introduced in the design of self-adaptive software systems in order to separate the  
617 different types of concerns and to address various kinds of uncertainties. An interesting and  
618 important, but narrowly addressed research topic is functional emergence and utilization functional  
619 affordances in the case of NG-CPSs. Emergence may be a result of self-organisation, in particular in  
620 the case of multi-agent-based systems.
- 621 8. Designing CPSs requires an extensive collection of heterogeneous computational models, such as  
622 systems models, morphological models, physical models, structural models, hardware models,  
623 software model, information model, control models, reasoning models, and so forth, to enable deep  
624 semantic integration, simulation, and analysis. Models should interoperate and provide a sufficiently  
625 complete representation of the operation, structure, and behaviour of 2G-CPSs. In spite the efforts to  
626 introduce meta-models and mega-models, the currently used models (i) work in conceptually  
627 different engineering dimensions, (ii) are based on different abstractions, and (iii) involve different



- 628 representation formalisms. The methodology of coherent and consistent transdisciplinary and multi-  
629 dimensional system modelling and a cross-domain (hardware, software, and cyberware)  
630 representation formalisms need further attention in research. Formal criteria for structural and  
631 semantic consistency of modelling tools are not addressed with sufficient emphasis.
- 632 9. Several authors emphasize both the (restricted) necessity and feasibility of building self-explainable  
633 systems that monitor and analyse their behaviour and generate an explanation for human  
634 stakeholders involved in supervision based on explanation models. This approach however loses its  
635 significance in the case of systems with high level of autonomy.
- 636 10. Self-adaptive systems mostly consider parametric, functional, and architectural properties that  
637 capture concerns such as performance, reliability, and cost. A recent development in research is  
638 addressing non-functional or para-functional characteristics of NG-CPSs, such as trust, awareness,  
639 intellect, and emotions. These topics seem to be ready for immediate or near-future research.

## 640 Acknowledgments

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

## 648 References

- 649 [1] Franz, J.H. (2011). A Critique of Technology and Science: An Issue of Philosophy. *Southeast Asia:  
650 A Multidisciplinary Journal*, Vol 11, pp. 23-36.
- 651 [2] Yankovskaya, V.V., & Kukushkin, S.N. (2019). The Role of the High School in the “Triple Loop”  
652 Model: SCBIN Technologies. In: *IOP Conference Series: Earth and Environmental Science*. IOP  
653 Publishing, 274(1), p. 012115.
- 654 [3] Wetter, K.J. (2006). Implications of Technologies Converging at the Nano-Scale: Are We Ready  
655 for the Little BANG? In: *Nanotechnologien Nachhaltig Gestalten*. Institut für Kirche und  
656 Gesellschaft: Iserlohn, Germany, pp. 31-41.
- 657 [4] Horváth, I., & Gerritsen, B.H. (2012). Cyber-Physical Systems: Concepts, Technologies and  
658 Implementation Principles. In: *Proceedings of the International Symposium on Tools and Methods  
659 of Competitive Engineering - TMCE 2012*, Vol. 1, Delft university Press, Delft, pp. 19-36.
- 660 [5] Ospina, P.D., Díaz, M.C., & Lara, S. (2016). New Technologies in Eye Surgery - A Challenge for  
661 Clinical, Therapeutic, and Eye Surgeons. In: *Advances in Eye Surgery*. IntechOpen, pp. 1-20.
- 662 [6] Zhang, F., DiSanto, W., Ren, J., Dou, Z., Yang, Q., & Huang, H. (2011). A Novel CPS System for  
663 Evaluating a Neural-Machine Interface for Artificial Legs. In: *Proceedings of the IEEE/ACM  
664 Second International Conference on Cyber-Physical Systems*, IEEE, pp. 67-76.
- 665 [7] Denkena, B., Dittrich, M.A., Stamm, S., Wichmann, M., & Wilmsmeier, S. (2021). Gentelligent  
666 Processes in Biologically Inspired Manufacturing. *CIRP Journal of Manufacturing Science and  
667 Technology*, 32, pp. 1-15.
- 668 [8] Horváth, I., Rusák, Z., & Li, Y. (2017). Order Beyond Chaos: Introducing the Notion of Generation  
669 to Characterize the Continuously Evolving Implementations of Cyber-Physical Systems. In:  
670 *Proceedings of the International Design Engineering Technical Conferences*, Vol. 58110,  
671 American Society of Mechanical Engineers. p. V001T02A015.

- 672 [9] Fitzgerald, J., Larsen, P.G., & Verhoef, M. (2014). From Embedded to Cyber-Physical Systems:  
673 Challenges and Future Directions, In: *Collaborative Design for Embedded Systems*, Springer,  
674 Heidelberg New York, pp. 293-302.
- 675 [10] Zhang, J.J., Wang, F.Y., Wang, X., Xiong, G., Zhu, F., Lv, Y., ... & Lee, Y. (2018). Cyber-  
676 physical-social systems: The state of the art and perspectives. *IEEE Transactions on Computational*  
677 *Social Systems*, 5(3), pp. 829-840.
- 678 [11] Black, W.S., Haghi, P., & Ariyur, K.B. (2014). Adaptive Systems: History, Techniques, Problems,  
679 and Perspectives. *Systems*, 2(4), pp. 606-660.
- 680 [12] Bock, W.J. (1980). The Definition and Recognition of Biological Adaptation. *American Zoologist*,  
681 20(1), pp. 217-227.
- 682 [13] Holland, J.H. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with*  
683 *Applications to Biology, Control, and Artificial Intelligence*. MIT Press, Boston.
- 684 [14] Horváth, I., Suárez Rivero, J.P., & Hernández Castellano, P. M. (2019). Past, Present and Future of  
685 Behaviourally Adaptive Engineered Systems. *Journal of Integrated Design and Process Science*,  
686 23(1), pp. 1-15.
- 687 [15] Tav ar, J., & Horváth, I. (2018). A Review of the Principles of Designing Smart Cyber-Physical  
688 Systems for Runtime Adaptation: Learned Lessons and Open Issues. *IEEE Transactions on*  
689 *Systems, Man, and Cybernetics: Systems*, 49(1), pp. 145-158.
- 690 [16] Shevtsov, S., Berekmeri, M., Weyns, D., & Maggio, M. (2017). Control-Theoretical Software  
691 Adaptation: A Systematic Literature Review. *IEEE Transactions on Software Engineering*, 44(8),  
692 pp. 784-810.
- 693 [17] Salehie, M., & Tahvildari, L. (2009). Self-Adaptive Software: Landscape and Research Challenges.  
694 *ACM Transactions on Autonomous and Adaptive Systems*, 4(2), pp. 1-42.
- 695 [18] Patikirikorala, T., Colman, A., Han, J., & Wang, L. (2012). A Systematic Survey on the Design of  
696 Self-Adaptive Software Systems using Control Engineering Approaches. In: *Proceedings of the 7th*  
697 *International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, IEEE,  
698 pp. 33-42.
- 699 [19] Swarnkar, P., Jain, S.K., & Nema, R.K. (2014). Adaptive Control Schemes for Improving the  
700 Control System Dynamics: A Review. *IETE Technical Review*, 31(1), pp. 17-33.
- 701 [20] Estrada-Jimenez, L. A., Pulikottil, T., Peres, R. S., Nikghadam-Hojjati, S., & Barata, J. (2021).  
702 Complexity Theory and Self-Organization in Cyber-Physical Production Systems. *Procedia CIRP*,  
703 104, pp. 1831-1836.
- 704 [21] Khoramshahi, M., & Billard, A. (2019). A Dynamical System Approach to Task-Adaptation in  
705 Physical Human–Robot Interaction. *Autonomous Robots*, 43(4), pp. 927-946.
- 706 [22] Cámara, J., Lopes, A., Garlan, D., & Schmerl, B. (2016). Adaptation Impact and Environment  
707 Models for Architecture-Based Self-Adaptive Systems. *Science of Computer Programming*, 127,  
708 pp. 50-75.
- 709 [23] Cómbita, L.F., Giraldo, J., Cárdenas, A.A., & Quijano, N. (2015). Response and Reconfiguration of  
710 Cyber-Physical Control Systems: A Survey. In: *Proceedings of the 2nd Colombian Conference on*  
711 *Automatic Control*, IEEE, pp. 1-6.
- 712 [24] Miraz, M.H., Ali, M., Excell, P.S., & Picking, R. (2015). A Review on Internet of Things (IoT),  
713 Internet of Everything (IoE) and Internet of Nano Things (IoNT). In: *Proceedings of the 2015*  
714 *Internet Technologies and Applications Conference*, IEEE, pp. 219-224.
- 715 [25] Brun, Y., Di Marzo Serugendo, G., Gacek, C., Giese, H., Kienle, H., Litoiu, M., Muller, H., Pezze,  
716 M., & Shaw, M. (2009). Engineering Self-Adaptive Systems through Feedback Loops. In: *Software*  
717 *Engineering for Self-Adaptive Systems*, Springer-Verlag, Berlin, pp. 48-70.
- 718 [26] Heylighen, F. (2011). Self-Organization of Complex, Intelligent Systems: An Action Ontology for  
719 Transdisciplinary Integration. *Integral Review*, 134, pp. 1-39.
- 720 [27] Floch, J., Hallsteinsen, S., Stav, E., Eliassen, F., Lund, K., & Gjørven, E. (2006). Using  
721 Architecture Models For Runtime Adaptability. *IEEE Software*, 23(2), pp. 62-70.

- 722 [28] Vogel, T., & Giese, H. (2013). Model-Driven Engineering of Adaptation Engines for Self-Adaptive  
723 Software: Executable Runtime Megamodels. *Technische Berichte No. 66*. Universitätsverlag  
724 Potsdam. 1-59.
- 725 [29] Kephart, J.O., & Chess, D.M. (2003). The Vision of Autonomic Computing. *Computer*. 36(1), pp.  
726 41–50.
- 727 [30] Miller, B. (2005). The autonomic computing edge: The Role of Knowledge in Autonomic Systems.  
728 DB2 Developer Domain. <http://www128.ibm.com/developerworks/autonomic/library/ac-edge6/>.
- 729 [31] Iglesia, D.G., & Weyns, D. (2015). MAPE-K Formal Templates to Rigorously Design Behaviors  
730 for Self-Adaptive Systems. *ACM Transactions on Autonomous and Adaptive Systems*, 10(3), pp. 1-  
731 31.
- 732 [32] Tav ar, J., & Horváth, I. (2020), How Can a Smart Cyber-Physical System Validate its Runtime  
733 Adaptation Actions Before and After Executing Them?, In: *Proceedings of the Thirteenth  
734 International Symposium on Tools and Methods of Competitive Engineering Symposium*, Delft  
735 University of Technology, pp. 103-114.
- 736 [33] Hidaka, S., Hu, Z., Litoiu, M., Liu, L., Martin, P., Peng, X., ... & Yu, Y. (2019). Design and  
737 Engineering of Adaptive Software Systems. In: *Engineering Adaptive Software Systems*, Springer,  
738 Singapore, pp. 1-33.
- 739 [34] Muccini, H., Sharaf, M., & Weyns, D. (2016). Self-Adaptation for Cyber-Physical Systems: A  
740 Systematic Literature Review. In: *Proceedings of the 11th International Symposium on Software  
741 Engineering for Adaptive and Self-Managing Systems*, pp. 75-81.
- 742 [35] Chandra, A., Lewis, P.R., Glette, K., & Stilkerich, S.C. (2016). Reference Architecture for Self-  
743 Aware and Self-Expressive Computing Systems. In: *Self-Aware Computing Systems*, Springer,  
744 Cham. pp. 37-49.
- 745 [36] Hummida, A.R., Paton, N.W., & Sakellariou, R. (2016). Adaptation in Cloud Resource  
746 Configuration: A Survey. *Journal of Cloud Computing*, 5(1), pp. 1-16.
- 747 [37] Cámara, J., Papadopoulos, A.V., Vogel, T., Weyns, D., Garlan, D., Huang, S., & Tei, K. (2020).  
748 Towards Bridging the Gap between Control and Self-Adaptive System Properties. In: *Proceedings  
749 of the 15th International Symposium on Software Engineering for Adaptive and Self-Managing  
750 Systems, IEEE/ACM*, pp. 78-84.
- 751 [38] Sanchez, M., Exposito, E., Aguilar, J. (2020). Autonomic Computing in Manufacturing Process  
752 Coordination in Industry 4.0 Context. *Journal of Industrial Information Integration*, 19, pp.1-16.
- 753 [39] Rodrigues, A., Caldas, R.D., Rodrigues, G.N., Vogel, T., & Pelliccione, P. (2018). A Learning  
754 Approach to Enhance Assurances for Real-Time Self-Adaptive Systems. In: *Proceedings of 13th  
755 IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing  
756 Systems*, IEEE, pp. 206-216.
- 757 [40] Engelenburg, S., Janssen, M., & Klievink, B. (2019), Designing Context-Aware Systems: A  
758 Method for Understanding and Analysing Context in Practice. *Journal of Logical and Algebraic  
759 Methods in Programming*, 103(2019), pp. 79-104.
- 760 [41] Don, S., Choi, E., & Min, D. (2012). Event Driven Adaptive Awareness System for Medical Cyber  
761 Physical Systems. In: *Proceedings of the 4th International Conference on Awareness Science and  
762 Technology*. IEEE, pp. 238-242.
- 763 [42] Braberman, V., D'Ippolito, N., Kramer, J., Sykes, D., & Uchitel, S. (2015), MORPH: A Reference  
764 Architecture for Configuration and Behaviour Self-Adaptation. In: *Proceedings of the 1st  
765 International Workshop on Control Theory for Software Engineering*, August 2015 pp. 9-16.
- 766 [43] Cansado, A., Canal, C., Salaün, G., & Cubo, J. (2010). A Formal Framework for Structural  
767 Reconfiguration of Components under Behavioural Adaptation. *Electronic Notes in Theoretical  
768 Computer Science*, 263, pp. 95-110.
- 769 [44] Garlan, D. (2001). Software Architecture. In: *Wiley Encyclopedia of Software Engineering*. John  
770 Wiley & Sons, Inc., New York. pp. 1-10.

- 771 [45] Cheng, S.W., Garlan, D., & Schmerl, B. (2006). Architecture-Based Self-Adaptation in the  
772 Presence of Multiple Objectives. In: *Proceedings of the 2006 International Workshop on Self-*  
773 *Adaptation and Self-Managing Systems*, pp. 2-8.
- 774 [46] Villegas, N.M., Tamura, G., & Müller, H.A. (2017). Architecting Software Systems for Runtime  
775 Self-Adaptation: Concepts, Models, and Challenges. In: *Managing Trade-offs in Adaptable*  
776 *Software Architectures*, Morgan Kaufmann, x, pp. 17-43.
- 777 [47] Kramer J., & Magee, J. (2007). Self-Managed Systems: An Architectural Challenge. In:  
778 *Proceedings of the Conference on the Future of Software Engineering*. Washington, DC, IEEE  
779 Computer Society, pp. 259-268.
- 780 [48] Phan, L.T., & Lee, I. (2011). Towards a Compositional Multi-Modal Framework for Adaptive  
781 Cyber-Physical Systems. In: *Proceedings of the 17th International Conference on Embedded and*  
782 *Real-Time Computing Systems and Applications*, Vol. 2, IEEE, pp. 67-73.
- 783 [49] McKinley, P.K., Sadjadi, S.M., Kasten, E.P., & Cheng, B.H. (2004). Composing Adaptive  
784 Software. *Computer*, 37(7), pp. 56-64.
- 785 [50] Masrur, A., Kit, M., Mat na, V., Bureš, T., & Hardt, W. (2016). Component-Based Design of  
786 Cyber-Physical Applications with Safety-Critical Requirements. *Microprocessors and*  
787 *Microsystems*, 42, pp. 70-86.
- 788 [51] Weyns, D., & Andersson, J. (2013). On the Challenges of Self-Adaptation in Systems of Systems.  
789 In: *Proceedings of the First International Workshop on Software Engineering for Systems-of-*  
790 *Systems*, pp. 47-51.
- 791 [52] Blair, G., Bencomo, N., & France, R.B. (2009). “Models@run.time,” *Computer*, 42(10), pp. 22-27.
- 792 [53] Szvetits, M., & Zdun, U. (2016). Systematic Literature Review of the Objectives, Techniques,  
793 Kinds, and Architectures of Models at Runtime. *Software & Systems Modeling*, 15(1), pp. 31-69.
- 794 [54] Cheng, B.H., Eder, K.I., Gogolla, M., Grunske, L., Litoiu, M., Müller, H.A., ... & Villegas, N.M.  
795 (2014). Using Models at Runtime to Address Assurance for Self-Adaptive Systems. In:  
796 *Models@Run.time*. Lecture Notes in Computer Science, Vol. 8378, Springer, Cham. pp. 101-136.
- 797 [55] Bennaceur, A., France, R., Tamburrelli, G., Vogel, T., Mosterman, P. J., Cazzola, W., ... & Redlich,  
798 D. (2014). Mechanisms for Leveraging Models at Runtime in Self-Adaptive Software. In:  
799 *Models@Run.time*. Lecture Notes in Computer Science, Vol. 8378, Springer, Cham. pp. 19-46.
- 800 [56] Klös, V., Göthel, T., & Glesner, S. (2018). Comprehensible and Dependable Self-Learning Self-  
801 Adaptive Systems. *Journal of Systems Architecture*, 85, pp. 28-42.
- 802 [57] Hadj-Kacem, N., Kacem, A.H., & Drira, K. (2009). A Formal Model of a Multi-Step Coordination  
803 Protocol for Self-Adaptive Software Using Coloured Petri Nets. *International Journal of*  
804 *Computing and Information Sciences*, 7(1), pp. 1-15.
- 805 [58] Loukil, S., Kallel, S., Jmaiel, M. (2017). An Approach Based on Runtime Models for Developing  
806 Dynamically Adaptive Systems. *Future Generation Computer Systems*, 68(3), pp. 365-375.
- 807 [59] Müller, M., Müller, T., Talkhestani, B.A., Marks, P., Jazdi, N., & Weyrich, M. (2021). Industrial  
808 Autonomous Systems: A Survey on Definitions, Characteristics and Abilities. *AT-*  
809 *Automatisierungstechnik*, 69(1), pp. 3-13.
- 810 [60] Casadei, R., Pianini, D., Placuzzi, A., Viroli, M., & Weyns, D. (2020). Pulverization in Cyber-  
811 Physical Systems: Engineering the Self-Organizing Logic Separated from Deployment. *Future*  
812 *Internet*, 12(11), 203, pp. 1-28.
- 813 [61] Lee, J., Azamfar, M., Singh, J., & Siahpour, S. (2020). Integration of digital twin and deep learning  
814 in cyber-physical systems: Towards smart manufacturing. In: *IET Collaborative Intelligent*  
815 *Manufacturing*, 2(1), pp. 34-36.
- 816 [62] Zhou, P., Zuo, D., Hou, K.M., Zhang, Z., Dong, J., Li, J., & Zhou, H. (2019). A Comprehensive  
817 Technological Survey on the Dependable Self-Management CPS: From Self-Adaptive Architecture  
818 to Self-Management Strategies. *Sensors*, 19(5), p. 1033.
- 819 [63] Pearce, H., Yang, X., Pinisetty, S., & Roop, P.S. (2021). Runtime Interchange for Adaptive Re-use  
820 of Intelligent Cyber-Physical System Controllers. arXiv preprint arXiv:2110.01974.

- 821 [64] Macher, G., Diwold, K., Veledar, O., Armengaud, E., & Römer, K. (2019). The Quest for  
822 Infrastructures and Engineering Methods Enabling Highly Dynamic Autonomous Systems. In:  
823 *Proceedings of the European Conference on Software Process Improvement*. Springer, Cham, pp.  
824 15-27.
- 825 [65] Kühnle, H., & Bayanifar, H. (2017). An Approach to Develop an Intelligent Distributed  
826 Dependability and Security Supervision and Control For Industry 4.0 Systems. *Asian Journal of*  
827 *Information and Communications*, 9, pp. 8-16.
- 828 [66] Tan, T., & Chen, J. (2021). Model Predictive and Inversive Control for State Transition of  
829 Dynamics Systems. *Advances in Astronautics Science and Technology*, pp. 1-6.
- 830 [67] Colombo, A.W., Karnouskos, S., & Bangemann, T. (2014). Towards the Next Generation of  
831 Industrial Cyber-Physical Systems. In: *Industrial Cloud-Based Cyber-Physical Systems*. Springer,  
832 Cham, pp. 1-22.
- 833 [68] Yilma, B.A., Panetto, H., & Naudet, Y. (2021). Systemic Formalisation of Cyber-Physical-Social  
834 System (CPSS): A Systematic Literature Review. *Computers in Industry*, 129, pp. 1-25.
- 835 [69] Johnson, B., & Hernandez, A. (2016). Exploring Engineered Complex Adaptive Systems of  
836 Systems. *Procedia Computer Science*, 95, pp. 58-65.
- 837 [70] Johnson, B. (2020). Intelligent and Adaptive Systems of Systems for Engineering Desired Self-  
838 organization and Emergent Behavior. In: *Proceedings of the Future Technologies Conference*,  
839 Springer, Cham, pp. 126-146.
- 840 [71] Ali, S., Al Balushi, T., Nadir, Z., & Hussain, O. K. (2018). Standards for CPS. In: *Cyber Security*  
841 *for Cyber Physical Systems*. Springer, Cham, pp. 161-174.
- 842 [72] Ahmadi, A., Cherifi, C., Cheutet, V., & Ouzrout, Y. (2017). A Review of CPS 5 Components  
843 Architecture for Manufacturing Based on Standards. In: *Proceedings of the 11th International*  
844 *Conference on Software, Knowledge, Information Management and Applications*, IEEE, pp. 1-6.
- 845 [73] Hohwy, J. (2020). Self-Supervision, Normativity and the Free Energy Principle. *Synthese*, 10. pp.  
846 x-x
- 847 [74] Yin, D., Ming, X., & Zhang, X. (2020). Understanding Data-Driven Cyber-Physical-Social System  
848 (D-CPSS) Using a 7C Framework in Social Manufacturing Context. *Sensors*, 20(18), p. 5319.
- 849 [75] Yilma, B. A., Naudet, Y., & Panetto, H. (2018). Introduction to Personalisation in Cyber-Physical-  
850 Social Systems. In: *Proceedings of the OTM Confederated International Conferences on the Move*  
851 *to Meaningful Internet Systems*, Springer, Cham, pp. 25-35.
- 852 [76] Klös, V., Göthel, T., Glesner, S. (2018). Runtime Management and Quantitative Evaluation of  
853 Changing System Goals in Complex Autonomous Systems. *Journal of Systems and Software*, Vol.  
854 144, October 2018, pp. 314-327.
- 855 [77] Elkhodary, A., Esfahani, N., & Malek, S. (2010). FUSION: A Framework for Engineering Self-  
856 Tuning Self-Adaptive Software Systems. In: *Proceedings of the Eighteenth ACM SIGSOFT*  
857 *International Symposium on Foundations of Software Engineering*, pp. 7-16.
- 858 [78] Cámara, J., Garlan, D., Moreno, G.A., & Schmerl, B. (2017). Evaluating Trade-Offs of Human  
859 Involvement in Self-Adaptive Systems. In: *Managing Trade-Offs in Adaptable Software*  
860 *Architectures*, Morgan Kaufmann, pp. 155-180.
- 861 [79] Cimini, C., Pirola, F., Pinto, R., & Cavalieri, S. (2020). A Human-in-the-Loop Manufacturing  
862 Control Architecture for the Next Generation of Production Systems. *Journal of Manufacturing*  
863 *Systems*, 54, pp. 258-271.
- 864 [80] Firouznia, M., Peng, C., & Hui, Q. (2018). Toward Human-in-the-Loop Supervisory Control for  
865 Cyber-Physical Networks. arXiv preprint arXiv:1805.02611.
- 866 [81] Cogliati, D., Falchetto, M., Pau, D., Roveri, M., & Viscardi, G. (2018). Intelligent Cyber-Physical  
867 Systems for Industry 4.0. In: *Proceedings of the First International Conference on Artificial*  
868 *Intelligence for Industries*, IEEE, pp. 19-22.
- 869 [82] Finogeev, A., Finogeev, A., Fionova, L., Lyapin, A., Lychagin K. A. (2019), Intelligent Monitoring  
870 System for Smart Road Environment, *Journal of Industrial Information Integration*, 15(9), pp. 15-  
871 20.

- 872 [83] Krishna, C.M., & Koren, I. (2013). Adaptive Fault-Tolerance for Cyber-Physical Systems. In:  
873 *Proceedings of the International Conference on Computing, Networking and Communications*.  
874 IEEE, pp. 310-314.
- 875 [84] Ruíz Arenas, S. (2018). Exploring the role of system operation modes in failure analysis in the  
876 context of first generation cyber-physical systems (*Doctoral dissertation*, Delft University of  
877 Technology, Delft, the Netherlands).
- 878 [85] Eirinakis, P., Kalaboukas, K., Lounis, S., Mourtos, I., Rožanec, J. M., Stojanovic, N., & Zois, G.  
879 (2020). Enhancing Cognition for Digital Twins. In: *Proceedings of the International Conference on*  
880 *Engineering, Technology and Innovation*, IEEE, pp. 1-7.
- 881 [86] Müller, H.A., Rivera, L.F., Jiménez, M., Villegas, N.M., Tamura, G., Akkiraju, R., ... & Erpenbach,  
882 E. (2021). Proactive AIOps Through Digital Twins. In: *Proceedings of the 31st Annual*  
883 *International Conference on Computer Science and Software Engineering*, pp. 275-276.
- 884 [87] Ferrer, B.R., Mohammed, W.M., Lastra, J.L., Villalonga, A., Beruvides, G., Castaño, F., & Haber,  
885 R.E. (2018). Towards the Adoption of Cyber-Physical Systems of Systems Paradigm in Smart  
886 Manufacturing Environments. In: *Proceedings of the 16th International Conference on Industrial*  
887 *Informatics*, IEEE, pp. 792-799.
- 888 [88] Axelsson, J. (2019). Game Theory Applications in Systems-of-Systems Engineering: A Literature  
889 Review and Synthesis. *Procedia Computer Science*, 153, pp. 154-165.
- 890 [89] Du, D., Guo, T., & Wang, Y. (2020). DSML4CS: An Executable Domain-Specific Modeling  
891 Language for Co-Simulation Service in CPS. *International Journal of Web Services Research*,  
892 17(2), pp. 59-75.
- 893 [90] Buffoni, L., Ochel, L., Pop, A., Fritzon, P., Fors, N., Hedin, G., ... & Sjölund, M. (2021). Open  
894 Source Languages and Methods for Cyber-Physical System Development: Overview and Case  
895 Studies. *Electronics*, 10(8), pp. 902-x.
- 896 [91] Ruchkin, I. (2019). Integration of Modeling Methods for Cyber-Physical Systems (Doctoral  
897 dissertation (*PhD thesis*, Institute for Software Research School of Computer Science, Carnegie  
898 Mellon University, Pittsburgh, CMU-ISR-18-107).
- 899 [92] Merelli, E., Paoletti, N., & Tesei, L. (2012). A Multi-Level Model for Self-Adaptive Systems.  
900 arXiv preprint arXiv:1209.1628.
- 901 [93] Frank, U. (2014). Multilevel Modeling - Toward a New Paradigm of Conceptual Modeling and  
902 Information Systems Design. *Business & Information Systems Engineering*, 6(6), pp. 319-337.
- 903 [94] Kühne, A. (2018). Story of Levels. In: *Proceedings of the 5th International Workshop on Multi-*  
904 *Level Modelling*, Volume 2245 of CEUR Workshop Proceedings, pp. 673-682.
- 905 [95] Khakpour, N., Jalili, S., Talcott, C., Sirjani, M., & Mousavi, M. (2012). Formal Modeling of  
906 Evolving Self-Adaptive Systems. *Science of Computer Programming*, 78(1), pp. 3-26.
- 907 [96] Hartsell, C., Mahadevan, N., Ramakrishna, S., Dubey, A., Bapty, T., Johnson, T., ... & Karsai, G.  
908 (2019). Model-Based Design for CPS with Learning-Enabled Components. In: *Proceedings of the*  
909 *Workshop on Design Automation for CPS and IoT*, pp. 1-9.
- 910 [97] Bonci, A., Pirani, M., Bianconi, C., & Longhi, S. (2018). RMAS: Relational Multiagent System for  
911 CPS Prototyping and Programming. In: *Proceedings of the 14th IEEE/ASME International*  
912 *Conference on Mechatronic and Embedded Systems and Applications*, IEEE, pp. 1-6.
- 913 [98] Zhang, L., Zeigler, B.P., & Laili, Y. (2019). Introduction to Model Engineering for Simulation. In:  
914 *Model Engineering for Simulation*, Academic Press, pp. 1-23.
- 915 [99] de Lamotte, F.F., Berruet, P., Rossi, A., & Philippe, J.L. (2008). Control Code Generation using  
916 Model Engineering for an Electric Train. *IFAC Proceedings Volumes*, 41(2), pp. 8327-8332.
- 917 [100] Fishwick, P.A. (1990). Toward an integrated approach to simulation model engineering.  
918 *International Journal of General System*, 17(1), pp. 1-19.
- 919 [101] Hashmi, M.A., Mo, J.P., & Beckett, R. (2021). Transdisciplinary Systems Approach to Realization  
920 of Digital Transformation. *Advanced Engineering Informatics*, 49, p. 101316.

- 921 [102] Mo, J.P., & Beckett, R.C. (2021). Transdisciplinary System of Systems Development in the Trend  
922 to X4.0 for Intelligent Manufacturing. *International Journal of Computer Integrated*  
923 *Manufacturing*, pp. 1-15.
- 924 [103] van Gigch, J.P. (1993). Metamodeling: The Epistemology of System Science. *Systems Practice*,  
925 6(3), pp. 251-258.
- 926 [104] Sangiovanni-Vincentelli, A., Shukla, S.K., Sztipanovits, J., Yang, G., & Mathaikutty, D. A. (2009).  
927 Metamodeling: An Emerging Representation Paradigm for System-Level Design. *IEEE Design &*  
928 *Test of Computers*, 26(3), pp. 54-69.
- 929 [105] Odell, J., Nodine, M., & Levy, R. (2004). A Metamodel for Agents, Roles, and Groups. In:  
930 *Proceedings of the International Workshop on Agent-Oriented Software Engineering*, Springer,  
931 Berlin, Heidelberg, pp. 78-92.
- 932 [106] Yang, Z., Eddy, D., Krishnamurty, S., Grosse, I., & Lu, Y. (2018). A Super-Metamodeling  
933 Framework to Optimize System Predictability. In: *Proceedings of the International Design*  
934 *Engineering Technical Conferences*, Vol. 51722, American Society of Mechanical Engineers, p.  
935 V01AT02A009.
- 936 [107] Villar, E., Merino, J., Posadas, H., Henia, R., & Rioux, L. (2020). Mega-Modeling of Complex,  
937 Distributed, Heterogeneous CPS Systems. *Microprocessors and Microsystems*, 78, p. 103244.
- 938 [108] Gašević, D., Kaviani, N., & Hatala, M. (2007). On Metamodeling in Megamodels. In: *Proceedings*  
939 *of the International Conference on Model Driven Engineering Languages and Systems*, Springer,  
940 Berlin, Heidelberg, pp. 91-105.
- 941 [109] Favre, J. M., & Nguyen, T. (2005). Towards a Megamodel to Model Software Evolution Through  
942 Transformations. *Electronic Notes in Theoretical Computer Science*, 127(3), pp. 59-74.
- 943 [110] Vogel, T., & Giese, H. (2012). A Language for Feedback Loops in Self-Adaptive Systems:  
944 Executable Runtime Megamodels. In: *Proceedings of the 7th International Symposium on Software*  
945 *Engineering for Adaptive and Self-Managing Systems*, IEEE, pp. 129-138.
- 946 [111] Bayne, J.S. (1995). A Distributed Programming Model for Real-Time Industrial Control. *Control*  
947 *Engineering Practice*, 3(8), pp. 1133-1138.
- 948 [112] Bickhard, M.H. (2019). Dynamics Is Not Enough: An Interactivist Perspective. *Human*  
949 *Development*, 63(3-4), pp. 227-244.
- 950 [113] Gleizes, M.-P., Camps, V., George, J.-P., & Capera, D. (2007). Engineering Systems Which  
951 Generate Emergent Functionalities. In: *Proceedings of the Satellite Conference on Engineering*  
952 *Environment-Mediated Multiagent Systems*. Dresden, Germany, pp. x-x.
- 953 [114] Alcocer, J.P., Beck, F., & Bergel, A. (2019). Performance Evolution Matrix: Visualizing  
954 Performance Variations Along Software Versions. In: *Proceedings of the Working Conference on*  
955 *Software Visualization*, IEEE, pp. 1-11.
- 956 [115] Garces, L., Martinez-Fernandez, S., Neto, V.V. & Nakagawa, E.Y. (2020). Architectural Solutions  
957 for Self-Adaptive Systems. *Computer*, 53(12), pp. 47-59.
- 958 [116] Mokni, A., Urtado, C., Vauttier, S., Huchard, M., & Zhang, H.Y. (2016). A Formal Approach for  
959 Managing Component-Based Architecture Evolution. *Science of Computer Programming*, 127, pp.  
960 24-49.
- 961 [117] Ballesteros, J., Ayala, I., Caro-Romero, J.R., Amor, M., Fuentes, L. (2020), Evolving Dynamic  
962 Self-Adaptation Policies of mHealth Systems for Long-Term Monitoring, *Journal of Biomedical*  
963 *Informatics*, pp. x-x.
- 964 [118] Zhou, P., Zuo, D., Hou, K.M., Zhang, Z., Dong, J., Li, J., & Zhou, H. (2019). A Comprehensive  
965 Technological Survey on the Dependable Self-Management CPS: From Self-Adaptive Architecture  
966 to Self-Management Strategies. *Sensors*, 19(5), 1033, x-x.
- 967 [119] Epifani, I., Ghezzi, C., Mirandola, R., & Tamburrelli, G. (2009). Model Evolution by Run-Time  
968 Parameter Adaptation. In: *Proceedings of the 31st International Conference on Software*  
969 *Engineering*, IEEE, pp. 111-121.

- 970 [120] Roth, F.M., Krupitzer, C., & Becker, C. (2015). Runtime Evolution of the Adaptation Logic in  
971 Self-Adaptive Systems. In: *Proceedings of the International Conference on Autonomic Computing*,  
972 IEEE, pp. 141-142.
- 973 [121] Pasquale, L., Baresi, L., & Nuseibeh, B. (2011). Towards Adaptive Systems Through  
974 Requirements@Runtime. In: *Proceedings of the 6th Workshop on Models@run.time*, p. 39.
- 975 [122] Basich, C., Svegliato, J., Wray, K.H., Witwicki, S., Biswas, J., & Zilberstein, S. (2020). Learning  
976 to Optimize Autonomy in Competence-Aware Systems. arXiv preprint arXiv:2003.07745.
- 977 [123] Löffler, D., Schmidt, N., & Tscharn, R. (2018). Multimodal expression Of Artificial Emotion In  
978 Social Robots Using Color, Motion and Sound. In: *Proceedings of the International Conference on*  
979 *Human-Robot Interaction*, ACM/IEEE, pp. 334-343.
- 980 [124] Zhu, Y.B., & Li, J.S. (2019). Collective Behavior Simulation Based nn Agent with Artificial  
981 Emotion. *Cluster Computing*, 22(3), pp. 5457-5465.
- 982 [125] Catuogno, L., Galdi, C., & Pasquino, N. (2018). An Effective Methodology For Measuring  
983 Software Resource Usage. *IEEE Transactions on Instrumentation and Measurement*, 67(10), pp.  
984 2487-2494.
- 985 [126] Liu, Q., Li, Q., Li, Z., & Wu, H. (2020). Research on Software Resource Management Model  
986 Based on Cloud Service. In: *Proceedings of the 8th International Conference on Information*  
987 *Technology: IoT and Smart City*, pp. 65-68.
- 988 [127] Wan, J., Chen, B., Imran, M., Tao, F., Li, D., Liu, C., & Ahmad, S. (2018). Toward Dynamic  
989 Resources Management for IoT-Based Manufacturing. *IEEE Communications Magazine*, 56(2), pp.  
990 52-59.
- 991 [128] Hachicha, M., Halima, R.B., & Kacem, A.H. (2019). Formal Verification Approaches of Self-  
992 Adaptive Systems: A Survey. *Procedia Computer Science*, 159, pp. 1853-1862.
- 993 [129] Nia, M.A., Kargahi, M., & Faghih, F. (2020). Probabilistic Approximation of Runtime Quantitative  
994 Verification in Self-Adaptive Systems. *Microprocessors and Microsystems*, 72, p. 102943.
- 995 [130] Redfield, S.A., & Seto, M.L. (2017). Verification Challenges for Autonomous Systems. In:  
996 *Autonomy and Artificial Intelligence: A Threat or Savior?*, Springer, Cham, pp. 103-127.
- 997 [131] Eze, T., Anthony, R.J., Walshaw, C., & Soper, A. (2011). The Challenge of Validation for  
998 Autonomic and Self-Managing Systems. In: *Proceedings of the Seventh International Conference*  
999 *on Autonomic and Autonomous Systems*, pp. 128-133.
- 1000 [132] Dewasurendra, S.D., Vidanapathirana, A.C., & Abeyratne, S.G. (2019). Integrating Runtime  
1001 Validation and Hardware-in-the-Loop (HiL) testing with V&V in Complex Hybrid Systems.  
1002 *Journal of the National Science Foundation of Sri Lanka*, 47, 4.
- 1003 [133] Cardozo, N., Christophe, L., De Roover, C., & De Meuter, W. (2014). Run-Time Validation of  
1004 Behavioral Adaptations. In: *Proceedings of 6th International Workshop on Context-Oriented*  
1005 *Programming*, pp. 1-6.
- 1006 [134] Abuseta, Y., & Swesi, K. (2015). Towards a Framework for Testing and Simulating Self-Adaptive  
1007 Systems. In: *Proceeding of the 6th International Conference on Software Engineering and Service*  
1008 *Science*, IEEE, pp. 70-76.
- 1009 [135] Eberhardinger, B., Seebach, H., Knapp, A., & Reif, W. (2014). Towards Testing Self-Organizing,  
1010 Adaptive Systems. In: *Proceedings of the IFIP International Conference on Testing Software and*  
1011 *Systems*. Springer, Berlin, Heidelberg, pp. 180-185.
- 1012 [136] Schierman, J., Ward, D., Dutoi, B., Aiello, A., Berryman, J., DeVore, M., ... & Wadley, J. (2008).  
1013 Run-Time Verification and Validation for Safety-Critical Flight Control Systems. In: *Proceedings*  
1014 *of the AIAA Guidance, Navigation and Control Conference and Exhibit*, p. 6338.
- 1015 [137] Marshall, A., Jahan, S., & Gamble, R. (2018). Toward Evaluating the Impact of Self-Adaptation on  
1016 Security Control Certification. In: *Proceedings of the 13th International Conference on Software*  
1017 *Engineering for Adaptive and Self-Managing Systems*, pp. 149-160.
- 1018 [138] Cámara, J., Peng, W., Garlan, D., & Schmerl, B. (2018). Reasoning about Sensing Uncertainty and  
1019 its Reduction in Decision-Making for Self-Adaptation. *Science of Computer Programming*, 167, pp.  
1020 51-69.



- 1021 [139] Drechsler, R., Lüth, C., Fey, G., & Güneysu, T. (2018). Towards Self-Explaining Digital Systems:  
1022 A Design Methodology for the Next Generation. In: *Proceedings of the 3rd International*  
1023 *Verification and Security Workshop*, IEEE, pp. 1-6.
- 1024 [140] Blumreiter, M., Greenyer, J., Garcia, F.J., Klös, V., Schwammberger, M., Sommer, C., ... &  
1025 Wortmann, A. (2019). Towards Self-Explainable Cyber-Physical Systems. In: *Proceedings of the*  
1026 *22nd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*  
1027 *Companion*, IEEE, pp. 543-548.
- 1028 [141] Wickramasinghe, C.S., Amarasinghe, K., Marino, D.L., Rieger, C., & Manic, M. (2021).  
1029 Explainable Unsupervised Machine Learning for Cyber-Physical Systems. *IEEE Access*, 9, pp.  
1030 131824-131843.
- 1031 [142] Daglarli, E. (2021). Explainable Artificial Intelligence (xAI) Approaches and Deep Meta-Learning  
1032 Models for Cyber-Physical Systems. In: *Artificial Intelligence Paradigms for Smart Cyber-*  
1033 *Physical Systems*, IGI Global, pp. 42-67.
- 1034 [143] Huang, J., Bastani, F., Yen, I.L., & Jeng, J.J. (2009). Toward a Smart Cyber-Physical Space: A  
1035 Context-Sensitive Resource-Explicit Service Model. In: *Proceedings of the 33rd Annual*  
1036 *International Computer Software and Applications Conference*, Vol. 2, IEEE, pp. 122-127.
- 1037 [144] Schmidt, D.C., White, J., & Gill, C.D. (2014). Elastic Infrastructure to Support Computing Clouds  
1038 for Large-Scale Cyber-Physical Systems. In: *Proceedings of the 17th International Symposium on*  
1039 *Object/Component/Service-Oriented Real-Time Distributed Computing*, IEEE, pp. 56-63.
- 1040 [145] González-Nalda, P., Etxeberria-Agiriano, I., Calvo, I., & Otero, M. C. (2017). A Modular CPS  
1041 Architecture Design Based on ROS and Docker. *International Journal on Interactive Design and*  
1042 *Manufacturing*, 11(4), pp. 949-955.
- 1043 [146] Landolfi, G., Barni, A., Menato, S., Cavadini, F.A., Rovere, D., & Dal Maso, G. (2018). Design of  
1044 a multi-sided platform supporting CPS deployment in the automation market. In: *Proceedings of*  
1045 *the 2018 IEEE Industrial Cyber-Physical Systems*, IEEE, pp. 684-689.
- 1046 [147] Shin, D.Y., & Park, J.W. (2020). Modularization for Personal Social Service Robots. *The Journal*  
1047 *of the Convergence on Culture Technology*, 6(2), pp. 349-355.
- 1048 [148] Denkena, B., Henning, H., & Lorenzen, L.E. (2010). Genetics and Intelligence: New Approaches in  
1049 Production Engineering. *Production Engineering*, 4(1), pp. 65-73.
- 1050 [149] Lachmayer, R., Mozgova, I., & Scheidel, W. (2016). An Approach to Describe Gentelligent  
1051 Components in Their Life Cycle. *Procedia Technology*, 26, pp. 199-206.
- 1052 [150] Möhring, H.C., Wiederkehr, P., Erkorkmaz, K., & Kakinuma, Y. (2020). Self-optimizing  
1053 Machining Systems. *CIRP Annals*, 69(2), pp. 740-763.
- 1054 [151] Cuevas, A.D., & Guzman-Arenas, A. (2010). Automatic Fusion of Knowledge Stored in  
1055 Ontologies. *Intelligent Decision Technologies*, 4(1), pp. 5-19.
- 1056 [152] Zhang, S., Yang, L.T., Feng, J., Wei, W., Cui, Z., Xie, X., & Yan, P. (2021). A Tensor-Network-  
1057 Based Big Data Fusion Framework for Cyber-Physical-Social Systems (CPSS). *Information Fusion*,  
1058 76(2021), pp. 337-354.
- 1059 [153] Sahu, A., Mao, Z., Wlazlo, P., Huang, H., Davis, K., Goulart, A., & Zonouz, S. (2021). Multi-  
1060 Source Multi-Domain Data Fusion for Cyberattack Detection in Power Systems. *IEEE Access*, 9,  
1061 pp. 119118-119138.
- 1062 [154] Petrovska, A., Quijano, S., Gerostathopoulos, I., & Pretschner, A. (2020). Knowledge Aggregation  
1063 with Subjective Logic in Multi-Agent Self-Adaptive Cyber-Physical Systems. In: *Proceedings of*  
1064 *the 15th International Symposium on Software Engineering for Adaptive and Self-Managing*  
1065 *Systems*, IEEE/ACM, pp. 149-155.
- 1066 [155] Bürger, J., Kehrler, T., & Jürjens, J. (2020). Ontology Evolution in the Context of Model-Based  
1067 Secure Software Engineering. In: *Proceedings of the International Conference on Research*  
1068 *Challenges in Information Science*, Springer, Cham, pp. 437-454.
- 1069 [156] Kotis, K.I., Vouros, G.A., & Spiliotopoulos, D. (2020). Ontology Engineering Methodologies for  
1070 the Evolution of Living and Reused Ontologies: Status, Trends, Findings and Recommendations.  
1071 *The Knowledge Engineering Review*, 35(4), pp. 1-34.

- 1072 [157] Ferranti, N., Soares, S.S., & de Souza, J.F. (2021). Metaheuristics-Based Ontology Meta-Matching  
1073 Approaches. *Expert Systems with Applications*, 173, 114578.
- 1074 [158] Horváth, I. (2020). On Reasonable Inquiry and Analysis Domains of Sympérasology. *Journal of*  
1075 *Integrated Design and Process Science*, 24(1), pp. 5-43.
- 1076 [159] Gembarski, P.C. (2019). The Meaning of Solution Space Modelling and Knowledge-Based Product  
1077 Configurators for Smart Service Systems. In: *Proceedings of the International Conference on*  
1078 *Information Systems Architecture and Technology*, Springer, Cham, pp. 28-37.
- 1079 [160] Abel, G. (2014). Systematic Knowledge Research. Rethinking Epistemology. *Rivista*  
1080 *Internazionale di Filosofia e Psicologia*, 5(1), pp. 13-28.
- 1081

## 1082 **Author Biographies**

1083 **Dr. Imre Horváth** is emeritus professor of the Faculty of Industrial Design Engineering, Delft  
1084 University of Technology, the Netherlands. In the last years, his research group focused on research,  
1085 development, and education of smart cyber-physical system design, with special attention to cognitive  
1086 engineering. Prof. Horváth is also interested in systematic design research methodologies. He was the  
1087 promotor of more than 20 Ph.D. students. He was first author or co-author of more than 430 publications.  
1088 His scientific work was recognized by five best paper awards. He has a wide range of society  
1089 memberships and contribution. He is past chair of the Executive Committee of the CIE Division of  
1090 ASME. Since 2011, he is a fellow of ASME. He is member of the Royal Dutch Institute of Engineers. He  
1091 received honorary doctor titles from two universities, and the Pahl-Beitz ICONNN award for  
1092 internationally outstanding contribution to design science and education. He was distinguished with the  
1093 Lifetime Achievement Award by the ASME's CIE Division in 2019. He has served several international  
1094 journals as editor. He was the initiator of the series of International Tools and Methods of Competitive  
1095 Engineering (TMCE) Symposia. His current research interests are in various philosophical,  
1096 methodological, and computational aspects of smart product, system, and service design, as well as in  
1097 synthetic knowledge science and development of self-adaptive systems.

1098 **Dr. Jože Tav ar** has been working as senior lecturer at Product Development Division, LTH, of the  
1099 University of Lund, Sweden, since 2020. He earned B.Sc., M.Sc., and Ph.D. degrees in mechanical  
1100 engineering from the University of Ljubljana, in 1991, 1994, and 1999, respectively. He started his  
1101 research career in 1991, focusing on technical information systems, information flow in product  
1102 development, processes re-engineering, and methodology of design. Starting in 2006, he spent a decade in  
1103 industry as the Head of the Noise and Vibration Lab at the Domel Company, and as the Head of the  
1104 Quality Department at Iskra Mehanizmi, respectively. During this period he was involved in several  
1105 product-development teams of international corporations such as Philips, Electrolux, and Rowenta. He  
1106 co-ordinated the development of a motor diagnostic system and studied various topics of noise-reduction  
1107 and vibrations. He was also developing quality systems for the automotive industry (ISO/IATF 16949)  
1108 and for medical devices (ISO 13485, FDA). Between 2011 and 2020, he was lecturing at the University of  
1109 Ljubljana in various courses such as Design Methodology, Engineering Design Techniques, and  
1110 Engineering Design from Non-Metallic Materials. He was guest researcher at the ProSTEP AG, Germany,  
1111 in 1995, at the University Karlsruhe, Germany, in 1996, and at the Delft University of Technology, the  
1112 Netherlands, in 2016. Now, he has a unique combination of concrete product development experiences  
1113 and a holistic view on system approach. His current research topics are designing smart cyber-physical  
1114 systems, data mining and big-data analysis, and application of agility methods and knowledge  
1115 management in product-development processes. He published over 40 SCI papers, over 50 conference  
1116 papers, 5 book chapters, and over 80 technical reports.