Self-aware Cyber-Physical Systems

K. BELLMAN and C. LANDAUER, TopcyHouse Consulting, USA
N. DUTT, University of California–Irvine, USA
L. ESTERLE, Aarhus University, Denmark
A. HERKERSDORF, TU München, Germany
A. JANTSCH and N. TAHERINEJAD, TU Wien, Austria
P. R. LEWIS, Aston University, United Kingdom
M. PLATZNER, Paderborn University, Germany
K. TAMMEMÄE, Tallinn University of Technology, Estonia

38

In this article, we make the case for the new class of *Self-aware Cyber-physical Systems*. By bringing together the two established fields of cyber-physical systems and self-aware computing, we aim at creating systems with strongly increased yet managed autonomy, which is a main requirement for many emerging and future applications and technologies. Self-aware cyber-physical systems are situated in a physical environment and constrained in their resources, and they understand their own state and environment and, based on that understanding, are able to make decisions autonomously at runtime in a self-explanatory way. In an attempt to lay out a research agenda, we bring up and elaborate on five key challenges for future self-aware cyber-physical systems: (i) How can we build resource-sensitive yet self-aware systems? (ii) How to acknowledge situatedness and subjectivity? (iii) What are effective infrastructures for implementing self-awareness processes? (iv) How can we verify self-aware cyber-physical systems and, in particular, which guarantees can we give? (v) What novel development processes will be required to engineer self-aware cyber-physical systems? We review each of these challenges in some detail and emphasize that addressing all of them requires the system to make a comprehensive assessment of the situation and a continual introspection of its own state to sensibly balance diverse requirements, constraints, short-term and long-term objectives. Throughout, we draw on three examples of cyber-physical systems that may benefit from self-awareness: a multi-processor

The inception of this work goes back to discussions in the third international Self-Aware Cyber-Physical (SelPhyS) Workshop, held in 2018 at Aston University, Birmingham.
All authors contributed equally, and their names are listed in the alphabetical order of the first name appearing from each institute.
Authors' addresses: K. Bellman and C. Landauer, TopcyHouse Consulting, Thousand Oaks, 91362, USA; emails: bellmanhome@yahoo.com, topcycal@gmail.com; N. Dutt, University of California - Irvine, Department of Computer Science, Donald Bren School of Information and Computer Sciences, Irvine, CA , 92697- 3435, USA; email: dutt@uci.edu; L. Esterle, Aarhus University, Department of Engineering, Finlandsgade 22, Aarhus, 8000, Denmark; email: lukas.esterle@eng.au.dk; A. Herkersdorf, TU München, Chair of Integrated Systems, Theresienstr. 90, Munich, 80290, Germany; email: herkersdorf@tum.de; A. Jantschand N. TaheriNejad, TU Wien, Institute of Computer Technology, Gusshausstrasse 25-27, Vienna, 1040, Austria; emails: {nima.taherinejad, axel.jantschd}@tuwien.ac.at; P. R. Lewis, Aston University, Aston Lab for Intelligent Collectives Engineering, Aston Triangle, Birmingham, B4 7ET, United Kingdom; email: p.lewis@aston.ac.uk; M. Platzner, Paderborn University, Warburgerstrasse 100, Paderborn, 33098, Germany; email: platzner@upb.de; K. Tammemäe, Tallinn University of Technology, Dept. of Computer Systems, Akadeemia tee 15a, 12618, Tallinn, Estonia; email: kalle.tammemae@taltech.ee.

2378-962X/2020/06-ART38 $15.00
https://doi.org/10.1145/3375716

system-on-chip, a Mars rover, and an implanted insulin pump. These three very different systems nevertheless have similar characteristics: limited resources, complex unforeseeable environmental dynamics, high expectations on their reliability, and substantial levels of risk associated with malfunctioning. Using these examples, we discuss the potential role of self-awareness in both highly complex and rather more simple systems, and as a main conclusion we highlight the need for research on above listed topics.

CCS Concepts: • **Computer systems organization → Embedded and cyber-physical systems**; • **Computing methodologies → Artificial intelligence**; **Distributed artificial intelligence**; • **Computer systems organization →** *Robotic autonomy*;

Additional Key Words and Phrases: Self-awareness, cyber-physical systems, resource-sensitive, situatedness, subjectivity, organizational infrastructure, verification, guarantees, development processes

ACM Reference format:
K. Bellman, C. Landauer, N. Dutt, L. Esterle, A. Herkersdorf, A. Jantsch, N. TaheriNejad, P. R. Lewis, M. Platzner, and K. Tammemäe. 2020. Self-aware Cyber-Physical Systems. *ACM Trans. Cyber-Phys. Syst.* 4, 4, Article 38 (June 2020), 26 pages.
https://doi.org/10.1145/3375716

1 INTRODUCTION

Cyber-Physical Systems (CPSs) require us to think differently from how we consider classical computing systems, since they integrate computation with physical processes [1]. Externally, CPSs interact with the physical world in their locality, through sensors and actuators. Internally, CPSs are often heavily resource constrained, such that available resources can strongly constrain what the system is able to do. Further, CPSs come with a multitude of—often conflicting—design goals, including application-specific performance, price, energy efficiency, and functional safety. An important subclass of CPSs are reactive and real-time systems. Reactive systems execute and interact with their environment at a pace driven by that environment [2], while real-time systems need to satisfy timing constraints. In particular, for hard real-time systems we need formal guarantees for the timeliness of their responses, since not meeting timing constraints could result in catastrophe [3].

CPSs have an important role in contributing to many future and emerging technology visions, including Industry 4.0, Internet of Things (IoT), vehicular networks, and computing at the edge. Yet, the requirements of these applications, along with the characteristics above, lead to the development of CPSs being highly non-trivial. Typically, expertise from multiple diverse fields is required, as challenges include:

(1) Need to integrate computation not only with physical processes but also with psychological and social processes, especially those including communication and language, as such technologies also embed humans in the loop.
(2) Need unprecedented flexibility in both functional and non-functional behavior to respond to changes in the internal state, available resources, the local environment, and the "social" situation, so as to meet user goals.
(3) Need the ability to dynamically connect myriads of small edge devices for sensing and actuating, possibly also with available cloud resources to further enable data aggregation and complex processing (e.g., machine learning and reasoning) to support high-value decision making.

These challenges suggest a shift away from design taking place before deployment, based on assumptions about available resources and the environment, to systems making more decisions at

runtime. It is clear that increased autonomy will be needed to achieve this, but to provide guarantees, this autonomy will need to be deliberate, purposeful, and supervised. This leads us to the belief that *manageable degrees of autonomy* are key for future CPS to be able to answer these challenges.

We further believe that *self-aware computing* [4, 5] provides us with both the required adaptive autonomy as well as the capability to reason about and manage this autonomy during runtime. Self-aware computing is an established and growing field in the broad area of intelligent systems, concerned with computing systems that learn and reason about themselves and the environment they are in on an ongoing basis. This typically supports actions such as the generation of adaptive behavior and explanations for decisions taken during operation. In this article, we therefore make the case for *self-aware cyber-physical systems*, and identify challenges that arise when considering the intersection of self-aware computing with cyber-physical systems. It is our intention that this article serves as a roadmap for future research to integrate these technologies.

One may think of self-aware CPS as a subclass of adaptive and autonomous systems but with a greater emphasis on the ability of the self-aware system to explicitly reason and learn about its own components and capabilities and to adapt the use of those capabilities to different operational environments. Hence there is the implication that there will be more self-models, more types of self-instrumentation, more freedom to change and prioritize goals, and more reasoning capabilities for analyzing and characterizing one's own state and the environment.

Like many revolutions in computational and engineered systems, self-awareness is not adding just a component or a capability to existing systems; rather it changes these systems in fundamental ways. It changes how the CPS interacts with the world, with the human designers and users, and how it manages itself. Consider, for example, the revolution that occurred in real-time systems. Adding real-time capabilities was not just a matter of adding a clock to the system. Rather, it required new internal and external instrumentation and sensors for monitoring the environment and the real-time system and for acting upon the feedback it received. The fact that it had to be fast enough meant new types of algorithms for determining good enough responses and prioritizing responses. It often changed the design of mechanizms to be streamlined in certain ways and materials to be light enough and strong enough to be fast enough. Self-aware computing in CPSs has a similar array of widespread changes.

In this article, we first introduce some of the needs for self-awareness in complex CPSs and some of the challenges in developing self-aware CPS. As one will see, the motivations for self-awareness and the challenges in implementing self-awareness lead to the need to reconsider the design and development of such systems in a fundamental and pervasive manner. As the Self-Aware (SA) CPSs shift the design and development of a system from design to runtime operations, many development and test functions that were traditionally done by a human must now be implemented in a form that is machine processible and yet still accessible for review by human users. Making knowledge and reasoning processes to some degree explicit and accessible to processing by the CPS and the human is a hallmark of all self-aware systems, but now the additional challenge is to make such capabilities under the increasing developmental control of the CPS system.

Although SA CPSs come in an enormous variety, there are common challenges that must be addressed in all SA CPSs and, hence, common principles that underlie some of the new types of design and engineering processes that must be applied to all of them. In analogy to Design for Manufacturing or Design for Test, we are calling this Design for Self-awareness. After briefly defining CPSs and self-awareness, presenting the compelling motivations for self-awareness and presenting some of the common challenges, we will, starting in Section 4, describe what the new types of necessary design and engineering processes are for the Design for Self-awareness.

First, in Section 2 we elaborate on the idea of self-aware cyber-physical systems, providing a brief overview of self-aware computing research and highlighting the value that self-awareness

can bring to CPS. Integrating self-awareness and CPS also gives rise to new challenges, not yet considered in detail by either the CPS or self-aware computing communities, which we identify in Section 3. The following sections then cover some of the most relevant challenges: In Section 4, we present ideas on development processes and the engineering of self-aware CPS. Section 5 takes up the fact that CPS are resource-constrained and looks into approaches of implementing self-awareness techniques under limitations in resources. Section 6 covers the issues of situatedness and subjectivity that enable self-awareness to deal with the unique characteristics of each CPS. Section 7 focuses on the novel approach to designing and developing platforms and the required changes in software architectures for enabling adaptive self-aware CPS. Section 8 deals with verification and how to give some guarantees for autonomously acting CPS. Finally, Section 9 concludes the article, laying out key future research themes.

2 THE VISION OF SELF-AWARE CYBER PHYSICAL SYSTEMS

In this section, we first discuss challenges for CPSs using several relevant examples and argue that increased and manageable autonomy will be key for forthcoming systems. Then, we review the concept of self-awareness as it pertains to computing systems and describe the main features of computational self-awareness.

2.1 Challenges for Cyber-Physical Systems

The huge variety of CPSs can be classified along many dimensions, from tiny to room-filling physical form factors, from single nodes to massively distributed systems, and from easily accessible to systems operating in remote and hard-to-access locations. In the following, we present three example applications of CPSs to illustrate their diversity in challenges. Nevertheless, they all critically require autonomy in their individual operational contexts.

Heterogeneous Multiprocessor System-on-Chip (HMPSoC). Modern HMPSoC represent a microcosm of a larger CPS. They face extreme resource limitations but must deal with variability in multiple aspects of their own physical manifestation, such as tolerances in the manufactured device, temperature management in their operational environment, and the application stress caused by varying workloads. The prototypical CPS characteristic of sense-control-actuate loops for power, frequency, performance, load, and temperature management are performed at multiple levels of design abstraction, e.g., circuit, logic, micro-architecture, network-on-chip, runtime operating system, and end applications. The control or decision policies are often cross-layer, i.e., information is sensed across multiple layers of the abstraction stack, and actuations are effected at multiple layers of abstraction. Design constraints faced by HMPSoCs are multi-dimensional, spanning performance, peak power, energy consumption, peak operational temperatures, dynamic soft-error rates, ageing, and so on. Moreover, many of these constraints have complex, non-linear interrelationships that make it difficult to address one or few design constraints separately. These complex interactions between abstraction levels, interdependent constraints, inability to isolate side-effects of control/actuation policies, and tight compute and memory limitations pose unique challenges in guaranteeing predictable or bounded behaviors.

NASA Mars Rover Curiosity. Curiosity, launched in November 2011, is a mobile robot that supports scientific missions on the surface of Mars [6, 7]. It is the size of a station wagon and has over 50,000 parts, including a robotic arm to pick up desired samples, a plutonium energy source for generating electricity, and a host of different sensors such as a chemical camera that shoots lasers at materials to identify their constituents. Because Curiosity must operate far from human operators, with many minutes between subsequent command cycles (between 4 and 24 minutes of radio propagation delay, depending on orbital positions), and maneuver in often unknown parts of the

Mars surface, it has a high autonomy requirement to do the best observations possible in a limited time and to stay out of trouble. While Curiosity is a great success and has operated reliably for over 2,000 Mars days, it has limitations. For instance, it can only pursue limited classes of unexpected opportunities for new observations that are not commanded and controlled by human operators [6]. In a future rover mission, NASA plans to improve the rover capabilities to act autonomously and to be an effective partner in the scientific studies. One key self-awareness aspect particularly needed is the ability to abstract, summarize, and report to Earth what it has done and experienced. Also a mobile sensor, a helicopter, will be included in future missions that will greatly increase the rover's capacity to explore, observe, and measure the Mars surface. However, because the rover architecture lacks the ability to coordinate with other agents, the mobile helicopter is programmed to never return to the rover in order to avoid accidental collisions.

These examples show that the rover, notwithstanding its marvelous performance, has severe shortcomings in understanding and assessing its situation, and therefore it is kept under tight human control, resulting in limited autonomy, flexibility, and adaptivity.

Wearable Medical Devices. An insulin pump is a small, computerized device programmed to deliver insulin into the fatty tissue under the skin. An insulin pump must be durable and last for years, but the insulin supply and certain pump components such as the insulin reservoir, tubing, and the infusion set need to be changed every few days [8] (see also dtc.ucsf.edu). A pacemaker is placed into the chest to treat arhythmias [9, 10], i.e., abnormal heart rhythms, by generating electrical pulses to prompt the heart to beat at a normal rate.

As in all CPSs, wearable medical devices must deal with their physical limitations within the environment they operate, such as providing the required functionality with very limited computing and memory resources, dealing with material degradation and aging, and optimizing battery life time. Although many design issues have successfully used available CPS engineering methods, there are new challenges because of the local adaptations permitted by increasingly capable devices and their ability to be networked to external programs. In addition, these medical devices face many challenges that are quite familiar to mobile CPSs and that could benefit from self-awareness processes, including (1) adapting to incompletely known dynamic environments, (2) uncertainty in effectiveness the results of its plans, (3) need for rapid and effective "safing" of the system in the face of unexpected events, (4) the need for re-planning and learning for self-improvement, and, most importantly, (5) the need to recognize when it must ask for help. A rising threat for wearable medical devices are external cyber attacks that destroy the integrity of the device. In 2017, the USA FDA recalled over 400,000 implanted pacemakers potentially at risk of cyber attacks [9, 11]. Although the industry is working on patching these risks, as late as June 2019, Medtronic recalled their insulin pumps as the USA FDA announced hacking risks for this product. While such devices apparently demand cyber security to avoid leaking private information and being hacked, they also need to be accessed by appropriate personnel during emergencies or for routine check-ups. Such contradictory objectives between security and accessibility hold for many smaller CPS and IoT devices.

While a Mars rover, an insulin pump, and a complex HMPSoC are vastly different, the challenges to interpret sensory data and transform them into a goal-related understanding of the situation are very similar.

While it is perhaps easier to make the argument for self-awareness in complex autonomous systems operating in unfamiliar physical environments, we purposefully include the insulin pump in this article to explore a possible role for self-awareness in these simpler devices too. The role of minimal self-awareness in high-risk scenarios is a theme to which we will return in Section 9.1.

These examples illustrate the dynamic and unpredictable nature of the situations in which many CPSs operate, which render fixed design-time rules to have only limited value. Such CPSs would

greatly benefit from a higher degree of autonomy, including a comprehensive understanding of their situation in their environment, their own state, and their integrity and performance to make the best decisions for committing their limited resources to the most profitable actions. In summary, the main challenges faced by CPSs are as follows:

Environmental dynamics. The environment continuously changes and is only partially known. As a result, the effects of the system's actions is uncertain, its plans have to be continuously revised and reconsidered.

Situatedness and subjectivity. Each system's perspective is different, due to distinct context and positions from which sensing occurs, and since sensors themselves are unique. These factors should be considered in analyzing the sensory data.

Physical limitations and reliability. All resources are limited due to size, cost, computation power, and energy supply constraints. In addition, the quality of sensing, computing and communication is often limited and deteriorates further during the lifetime because of faults, failures, wear-out, and aging.

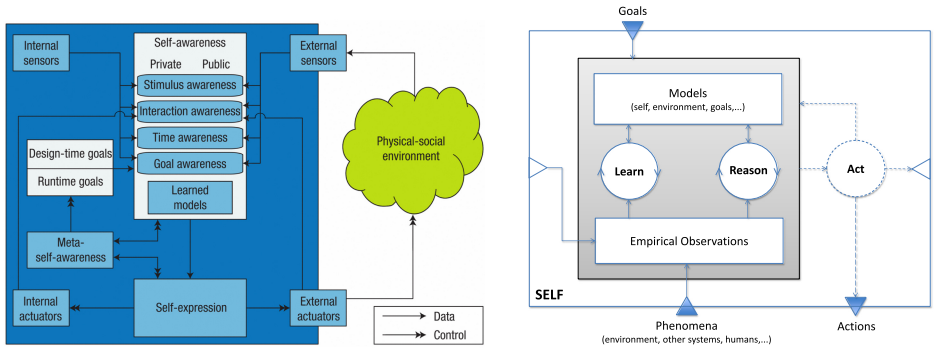
2.2 Computational Self-awareness

In the context of computing, the notion of self-awareness has been used to characterize systems able to build and use knowledge of aspects related to themselves, such as their own state, situation, behavior, and goals. This can also be applied recursively, so that a self-aware computing system can reason about its own processes for doing these things [12, 13]. Self-awareness has been proposed as a key enabling property for increasingly complex systems [14].

Generally speaking, self-aware computing research has been inspired by psychological concepts and processes (e.g., References [15, 16]). However, it aims to develop computational interpretations of self-awareness-related properties and capabilities, that are valuable for technical systems. Since this is an open-ended and complex set of characteristics, computational self-awareness is typically defined in terms of levels of self-awareness [15, 17], which may be present or desirable to different degrees. An important distinction for self-aware computing systems is that they sense their environment and their state at runtime, to extract knowledge that can then be used for reasoning and decision making at runtime. This implies a different new design procedure in which context- and runtime-dependent decisions are not integrated into the system at design-time by the designer (because that cannot be done adequately) but are made at runtime by the system itself.

Over the past two decades the research community has developed a number reference architectures for agent controllers as part of self-* systems with different emphases. Key examples include Observe-Decide-Act (ODA) [18] and Monitor-Analyze-Plan-Execute over a shared Knowledge base (MAPE-K) [19]. However, while systems built on these models often possess some level of self-awareness, the models do not explicitly capture self-awareness concerns. More recently, reference architectures have been developed to explicitly emphasise these concerns and provide an architectural perspective on a system's (possible) self-awareness capabilities. Two widely accepted reference architectures include the EPiCS reference architecture [4, 15] and the *Learn-Reason-Act* loop [20] (cf. Figure 1).

Research on self-aware computing systems has been supported by DARPA [21], as well as the European Commission through its Future & Emerging Technologies programme [22]. As of today, three books now collate thought and results from diverse research areas that have approached the idea of building self-aware computing systems [4, 5, 23], while five complementary surveys chart the progression of the research landscape over the past decade and more [14, 24–27]. From this literature, a number of key thematic properties of self-aware computing systems can be discerned:



(a) The EPiCS reference architecture [4, 15]. (b) The *Learn-Reason-Act* loop [20].

Fig. 1. Two reference architectures for self-aware computing systems.

Self-monitoring allows quicker and more accurate knowledge of how the system is being affected by the operational environment and how well its goals are being accomplished in the external environment.

Self-modeling allows the system to have access to explicit information of the limitations, best use and practices of all its components, as well as up to date information on the state of the system.

Learning allows the system to not only learn patterns about the external world, but how well its own components perform within different modes of operation and under different situations. When combined with reasoning processes, the system can also learn how to continually improve its planning and behavioral processes.

Self-analysis allows the system to use a variety of reasoning processes (including communication with other systems and human users) to diagnose any problems or anomalies, optimize or improve its performance, plan an adaptive response to a new situation and so forth. It does not have to wait for other systems or users to tell it how to interpret its monitored information or models and it can self-adapt its responses or behaviors. Self-analysis also includes self-testing and validation.

Self-reporting allows the system to inform the human user and other systems of the results of its monitoring, processing, learning and reasoning. In complex systems this is a critical capability as no external probes can be as fast or potentially as accurate.

It is important to note that self-awareness is not a binary property. We find many specific systems in the literature that either partially or entirely lack some of these abilities but that can still be considered self-aware to some degree. Indeed, capturing this spectrum is what motivated the development of computational frameworks based on levels of self-awareness [15].

2.3 Self-aware Cyber-Physical Systems

We contend that the concept of self-awareness and the runtime capabilities of self-aware computing systems can solve the autonomy challenges for future CPSs. For example, being able to maintain an understanding of its own state and the environment could enable the Mars rover to assess the effects of a well- or partially working chemical camera or an upcoming Mars storm. Moreover, sensing information over a longer period of time allows for creating and learning knowledge that can be used for reasoning and decision taking, for example, a specific workload that always leads to peak temperatures on the HMPSoC. To that end, sensor data must be analyzed and abstracted to a level where they can be related to the context. The context is ultimately defined by the goals of the

system. Hence, the system must come to understand how its current state and the environment, given the knowledge gained by experience, hinder or advance its goals.

As a result of bringing together the two fields of cyber systems and physical systems, we envision self-aware CPSs [28] with the following features:

- A self-aware CPS is *situated in a physical environment* by means of interaction through sensors and actuators and considers its specific *physical location and perspective* in the environment.
- A self-aware CPS is *resource constrained* and optimizes, at runtime, for multiple and often conflicting objectives, such as size, time, processing power, energy, cost, and temperature.
- A self-aware CPS maintains an *understanding of its state*, including the function and performance of its components such as sensors and actuators. The system knows relevant internal processes that define its behavior, its reactions and its capabilities. Based on that knowledge it is able to identify faults and deviations from expected behavior and performance.
- A self-aware CPS maintains an *understanding of its environment*, including the surrounding objects and other systems with which it interacts. The system is able to detect deviations from the expected behavior and performance of the environment, and it continuously learns and improves its understanding of the environment.
- A self-aware CPS maintains appropriate knowledge with respect to its *objectives and goals*, and uses that knowledge as an effective basis for resource allocation and dynamic decision making. The system is historic in the sense that it keeps track of behavior, performance and deviations of itself and of the environment over time to build up an assessment regarding the recent past, the long term past, or even the whole lifetime.
- A self-aware CPS *self-reports*, i.e., it can trace and explain its decisions and actions.

The above list constitutes our vision of a complete self-aware CPS. In spite of tight resource limitations, many CPSs would have significant computing capacity to realize some or all of these features, if done efficiently. Even if current systems do not fully implement these features, they may still qualify to be partially self-aware. In fact, even in the future CPSs we expect that specific systems will be only partially self-aware depending on an application's needs and constraints. Nevertheless, the abilities in this list contribute to self-awareness and should be considered in the processes of designing systems for being self-aware. We discuss some of the involved issues and challenges of designing for self-awareness in more details in the following sections.

3 CHALLENGES IN BRINGING SELF-AWARENESS TO CYBER-PHYSICAL SYSTEMS

Section 2 laid out the vision for self-awareness in computing systems in general and CPS in particular. However, while self-awareness is a promising vision, this class of systems possesses certain unique characteristics that make the application of self-awareness concepts non-trivial. Section 2.1 exposed a number of such characteristics, and in this section, we explore the intersection of these characteristics with what self-awareness promises. Five key research challenges arise in this exploration. These key challenges are then the focus of the rest of this article.

Building resource-sensitive self-awareness. First, we consider resource limitations and their implications for self-awareness. Obviously, self-awareness does not come for free, but the additional learning and behavior functions may save more overall resources than it takes to implement them. For this reason, it is important to always include the cost of self-awareness processes themselves in empirical evaluations of self-aware systems (as an example, see recent work on self-awareness in mobile robots [29]). In other cases, self-awareness does not result in reduced overall resource usage, but instead gains other qualities, such as flexibility or explainability. In these cases, it is important to conduct evaluation of a system in the context of the range of identified requirements.

Of course, for systems with higher levels of self-awareness, we desire that the system is able to balance such tradeoffs itself, making decisions concerning how to deploy its finite resources in pursuit of an efficient outcome in a tradeoff between such requirements. This leads to our first challenge for self-aware CPSs research: building resource-sensitive self-awareness.

Acknowledging situatedness and subjectivity. Next, we consider the interplay between environmental dynamics and self-monitoring/self-modelling. Environments vary, both between systems and in a given system over time, due to factors such as mobility, climate changes, and so on. A CPS's knowledge is necessarily *subjective* as far as it is derived from its own sensors, memory, and reasoning processes. This is important in the context of both a single CPS and in the integration of several systems. For example, the view of a Mars rover might be blocked by a hill and thus it may be unaware of interesting points. The accompanying helicopter, however, may have an unobstructed and different view. This leads to different perceptions due to different locations. This also applies temporally, and within an individual node. For example, a system able to reason not only about how its data change over time but also the perspective associated with each reading, means that it can make similar integrating judgments. It allows the system to arrive at conclusions about its state at a particular period of time. Longer term statements of this type may also identify failures in the sensing apparatus, prompting human repair or system reconfiguration in response. This also helps to address physical limitations and reliability demands. Therefore, our second challenge for self-aware CPSs is that self-awareness is subjective; its knowledge is dependent on the system's own perspective.

Creating an infrastructure for self-awareness processes. With infrastructure we mean here the mechanisms and facilities inside the CPS to enable dynamic and adaptive behavior. A self-aware agent deliberates the tradeoffs and allocates resources where they are most needed. However, this heavily depends on the goals and the current state of the system. Since the importance and priorities of goals can change depending on the situation, it is important the system can build goal models that map these higher-level goals into its current context and make use of different types of reasoning and learning processes. This also allows us to "program" systems in quite a different way, since the system possesses the ability to construct lower-level, perhaps intermediate goals from higher level ones. This allows us to describe desired system behavior in terms of high level goals, possibly at the level of the application, or a system's sustainability. Of course, this sort of activity itself will come at some resource cost, so an important question here is what infrastructure processes are needed to provide the system with such capabilities?

Verifying self-awareness and establishing guarantees. Lifelong adaptation on the part of CPSs can be expected to pose a challenge to their adoption. At a minimum, users will require that these systems with adaptation processes will have a minimum set of behavioral guarantees. Developing a CPS, self-aware or not, for which no guarantees at all can be provided, is a failure. Fortunately, self-awareness can help here, too. Since the system monitors and reflects over its behavior and adaptation processes, it can also generate explanations and hence build trust. Nevertheless, this discussion highlights the inherent challenge that exists in balancing the competing requirements of ongoing learning and adaptation, and establishing predictability and trustworthiness.

Developing new engineering processes. How, then, should we approach the creation of such systems? To build systems capable of such runtime reasoning and self-modification requires a substantial rethink in how we design CPS. Thus our final challenge for self-aware CPSs is to develop new design methods that provide general, trustworthy mechanisms for flexibly pursuing goals, based on self-monitoring and self-reflection. Two key design decisions are about how much adaptation

will be allowed/required on critical aspects of the system's functions and what instrumentation to insert to provide appropriate feedback to monitor behavior.

Thus, in summary we have five key challenges for achieving the self-aware CPSs vision:

(1) Developing new design and engineering processes (Section 4).
(2) Building resource-sensitive self-awareness (Section 5).
(3) Acknowledging situatedness and subjectivity of self-awareness (Section 6).
(4) Creating an infrastructure for self-awareness processes (Section 7).
(5) Verifying self-awareness and establishing guarantees (Section 8).

We believe that these challenges apply to self-aware systems in general, however they are more prominent in self-aware CPSs, due to the unique challenges that CPS face.

4 DEVELOPMENT PROCESSES AND ENGINEERING OF SELF-AWARE SYSTEMS

Self-awareness must be designed into a system from the beginning, partly because it is usually orthogonal to the other functionalities of the system, but mostly because it pervades many, if not all, parts, functions, components, and aspects of the system. In analogy to Design for Test and Design for Manufacturability, we call this design approach *Design for Self-awareness (DfSA)* and describe some of its necessary characteristics here.

One possible method for supporting this systematic approach is to provide blueprints of known and useful designs. Such blueprints can include reference architectures and design patterns that guide the engineer through all development phases. An example for such a reference architecture for SA computing systems has been proposed in Reference [4]. The reference architecture describes a SA system as being composed of several functionalities or blocks that explicitly represent knowledge about the system and its environment, and provide methods to gain this knowledge. The reference model has been applied to the development of rather diverse systems exhibiting self-awareness properties, such as runtime hardware reconfiguration in multi-core processors [30] and network protocol stacks [31], object tracking with smart cameras [32], industrial monitoring [33], heterogeneous high-performance computing systems [34], wearable healthcare systems [35], as well as single and multi-user active music environments [36]. Experience with these designs shows that a proper reference architecture is instrumental to developing effective SA systems, in particular, as it provides a common language for developers and supports appropriate separation of concerns.

Such a reference architecture and corresponding design patterns can be starting points also for Self-Aware Cyber-Physical Systems (self-aware CPSs), but in our view, DfSA has to be incorporated into all stages of CPS systems engineering, from concept engineering to component design to system integration, prototyping, development, deployment, and continual runtime validation.

First and foremost, the designers must decide how much flexibility and adaptability is expected/required in what parts of the system, how much those expectations require self-awareness, and how much the stakeholders are willing to pay for it, since self-awareness and autonomous decision making do not come for free. Note that this payment is not only in design cost; the corresponding costs are also in resources (e.g., areal footprint, computational effort, memory usage), additional efforts (in verification and certification), and in limitations or constraints on other parts of the system. These costs and benefits associated with the capability selections need quantification in appropriate metrics.

During concept design/engineering the most important decisions are to be taken, those that concern how much self-awareness will be in the resulting system, which features of self-awareness are included, and which parts and functions are covered by and used in self-awareness functions.

This includes deciding what information the SA features can access and what decisions are to be influenced by SA features.

A particularly difficult question is how to handle the "blind spots," i.e., those legacy (and other) components that have little or no self-awareness. The almost certain existence of these components implies that the SA parts have to be aware of more than just themselves, because they have to be aware of the context that includes the non-SA parts. Just like we adapt to our computers' limitations, these SA components will have to adapt to their other non SA components' limitations. That means that none of the SA parts can be designed in isolation.

The design decomposition and function allocation process must recognize that SA features may be distributed throughout the system, so the information detection, distribution and knowledge management functions must be carefully organized to guarantee that the SA functions get the data they need to make the decisions they will be expected to make.

This includes (1) parts more typically considered hardware and hardware support, such as sensors, effectors/actuators, data acquisition, proprioception sensors, and more complex processes such as sensor fusion, local action feedback, knowledge and model building; (2) parts typically not present without some self-awareness, such as fault and failure detection, diagnostic, and containment; and (3) parts typically not present at all, since they are generally managed by the operator organizations, such as resource allocation and management policies, and decision making policies in general. SA systems can and often will include all of these processes.

Such a comprehensive DfSA methodology will require supporting tools. Existing design tools need to be extended to incorporate the new self-awareness dimension in design, and new modeling methodologies and corresponding tools need to be developed. In any case it is paramount that the DfSA methodology and supporting tools builds upon and integrates well with established tool suites and design processes.

The necessity for a systematic consideration of self-awareness during the system design process, thus externalizing procedures, techniques and tools through DfSA for a community of potential adopters, differentiates our approach and thinking from already existing solutions with different degrees of self-awareness attributes.

5 RESOURCE-SENSITIVE SELF-AWARE SYSTEMS

Systems generally are constrained in terms of the computing, memory, sensor and communication resources they have at their disposal. Furthermore, these resources have to maintain balance with the system's requirements in terms of its performance, safety, security, and agility. Edge-IoT and wearable medical CPS are more susceptible to such resource constraints and finding the right balance with their requirements might be fragile. Therefore, a self-aware CPS has to be aware of its overall resource limitations, and in particular, to the share of resources that can be allocated for ensuring self-awareness properties.

One approach that helps to overcome this resource challenge is to off-load resource-intensive processes, such as learning or reasoning, to systems with ample resources available, such as cloud servers (e.g., as in Reference [29]). However, in some cases this is not be possible due to infrastructure constraints (e.g., not being permanently or closely enough connected to cloud systems) as well as due to security and safety demands.

Let us consider machine learning (ML) algorithms, which are desired in virtually every application nowadays, as an example for the constrained resources problem. Many of today's embedded systems are not able to provide for the heavy processing power they require. For instance, AlexNet [37] has 650,000 neurons, 60-M parameters, and requires about 1-G floating point operations (FLOPs) for a classification. The implications of this requirement is reflected not only in the processing power but also in required memory, memory bandwidth, and the energy consumption.

VGG, GoogleNet, or ResNet require resources as extensive as AlexNet [38]. For embedded systems such as wearable health care devices, this might pose an insurmountable barrier. Nevertheless, larger systems, such as a Mars rover, are required to evaluate the availability and state of resources before committing to running desired algorithms, and to make a decision based on the resulting tradeoff. This has motivated self-aware ML algorithms such as in References [38–40] for Epileptic seizure detection, a 1,000-class image classification of ImageNet dataset, and Iris flower classification applications, respectively. In these works, the authors use the concept of confidence [41] to reduce the complexity and required resources of various ML algorithms while maintaining or improving their performance [38–40]. The mechanism enabling the success of these algorithms is attention. Attention is "selective allocation of limited resources to specific tasks" [42]. In the above mentioned algorithms confidence is used to decide about the employed classification processes. In other works such as References [43] and [44] simpler approaches are taken to implement attention to increase the efficiency of resource usage.

Another approach to address resource constraints is *approximate computation* [45–47]. Approximate computing, which has been widely explored in the field of image processing [48–50] and ML [50, 51] whenever accurate computation is not necessary or a certain amount of imprecision can be tolerated [46]. Approximate computing solutions are designed to outperform their accurate counterparts in speed, energy consumption, or the size of the silicon die. Certain self-aware solutions, such as References [52–54], monitor the environment (e.g., communication channel characteristics) and its changes and adapt their interface (here its internal impedance) at runtime to match the environment. Improved matching in the impedance leads to increased power injection to the network and stronger signal detection at the reception end. This improves the performance of the overall system (i.e., signal to noise ratio (SNR) and quality of communication). In contrast to these solutions, where the system tries to save power, in Reference [55], based on the awareness of the system about the user behavior (estimated plug-in time) the system may allow normal or increased power consumption despite the (relatively) low-battery status.

Additionally, previous works on engineering approaches for self-aware systems, such as in Reference [56], encourage us to ask the following question: How much self-awareness is needed, explicitly acknowledging that processes realizing self-awareness provide potential (future) benefits, but at some cost? Currently, there is a substantial room to develop minimal, yet meaningful and reusable forms of computational self-awareness, such as what will be required for heavily resource-constrained systems. Indeed, it is interesting to ask what "minimal" self-awareness might look like. Another important and rather open research question in this regard is how to measure the constraints of the system as well as the cost of various processes and functionalities, including that of self-awareness itself. Even though in some cases measurement mechanisms could be very simple and straightforward, in some cases they present considerable challenges. For example, the analysis of the extent of the sensitivity of the system regarding a specific resource can become particularly challenging when isolating the effect of that specific resource at runtime and at the presence of (internal and especially external) circumstances.

6 SITUATEDNESS AND SUBJECTIVITY

Every CPS is unique, in terms of both its hardware and situation. CPSs can also vary to a great degree in terms of available resources, both comparatively and individually over time. These differences in situation, sensing capabilities, and how a system decides to deploy its limited resources mean that each self-aware cyber-physical system has a different point of view, a unique and limited window on the world.

In their research roadmap for self-adaptive systems [57], Cheng et al. acknowledge that adaptation decisions will be made based on the system's *perception* of its environment and itself[1]. Since any perception is from a particular perspective, a key challenge will be to build a theory of how a system's own perspective influences the internal representations that it builds. In line with Duval and Wicklund's [58] distinction between forms of self-awareness that are objective and subjective, and its interpretation in computational self-awareness [4], we class this as a form of *subjective self-awareness*. This is in contrast to the notion of *objective self-awareness*, defined as awareness of oneself as an object in the world, and the broad theoretical area in which much recent work on computational self-awareness can be placed.

Using the *learn-reason-act* schematic loop defined by Kounev et al. [20] for self-aware computing systems as an illustrative tool, this new theory of *subjective computational self-awareness* could include: (i) how the available sensors determine what can be learned, (ii) how available computational resources and reasoning techniques determine the scope of possible results of reasoning, (iii) how the system capabilities map into the current operational environment, and (iv) therefore the scope of possible models available for the generation of actions. Such a theory could then be operationalised to provide technical systems with the capabilities to reason about the impact of their and others' perspectives.

In the single CPS case, subjective self-awareness plays an important role as it allows the system to explicitly model and reason about the perspective from which its knowledge is derived. For example, a camera sensor may provide knowledge about a geographic location, but if it is behind a wall of smoke, then its knowledge of what is beyond the smoke will be limited. Similarly, if smoke is not generally present in the environment, but is local to the camera, then the system's knowledge of the environment will be reduced due to local conditions at its sensor. Modelling and reasoning about its own local perspective allows a self-aware CPS to take action to modify its perspective, for example to improve its view of the world [59] or to request help from other systems perhaps more fortunately situated.

Similar issues arise in collective systems. If an aggregating process is tasked with combining data from multiple sensor nodes, then explicitly accounting for the perspective of those sensors allows richer multi-perspective modelling of the environment, and actions to be generated in response. For example, if individual contributing CPS may modify their position or sensing behavior not only to improve their own perspective, but also to provide more complementary readings. This explicit acknowledgement of perspectives associated with data lets us go beyond attributing such differences to "noise in the data," providing more contextually meaningful knowledge.

The above discussion concerns spatial perspectives, which are perhaps the most obvious form of perspective to consider. However this also applies temporally, and within an individual node. For example, if a system is able to reason not only about how its data changes over time but also the perspective associated with each reading, then it can make similar integrating judgements [28]. It allows the system to arrive at conclusions such as "this sensor is usually well-positioned to provide knowledge of body temperature, but right now this is temporarily blocked due to detachment" [60]. Longer term statements of this type may also identify failures in sensing apparatus, prompting human repair or system reconfiguration in response [44]. Therefore, this also helps to address the physical limitations and reliability challenge.

Considering further the case of time awareness, a system may be in a fast-changing environment, but if its sensors sample at a slow rate, then the system may be unable to model the dynamics sufficiently accurately. In extreme cases, it may have no sense of any pattern, or even no plausible reason to believe that there are environmental changes at all. The latter could be due to only

[1]Emphasis added.

being able to interpret varying sensor readings as noise, or having a sensor timing that coincides with cycles in what is sensed. And this is clearly not unique to time awareness; similar arguments can be constructed for other levels of self-awareness, where the key idea is that the perception machinery provides a limited window on what is being modelled, thus either hampering or even misleading the modelling process. It will be useful therefore to be able to understand the causal link between perception machinery and modelling, as this pertains to aspects of self-awareness.

However, what meta-self-aware systems can also do is to respond to their understanding of how perception affects their own knowledge. This may take the form of modifying their perception machinery, where this is possible, for example increasing the frequency with which a sensor reading is taken, or modifying a trigger to take one. Alternatively, it make take the form of building an understanding of the limits of understanding derived from its current perception machinery into its own models. For example, a self-aware system may build additional uncertainty or known "knowledge gaps" [61] explicitly into the model, allowing it to reason about and identify when it does not know something. To return to our time-awareness example, this is the difference between (i) making a judgment based on knowledge induced from an infrequent sensor reading, because that was the best fit for the available data, or (ii) knowing that the way the system observed the particular phenomenon means that there is a large gap in the hypothesis space, and therefore although its current data points in one direction, it is entirely plausible that something more complex and contradictory is going on. In the latter case, a fuller abstract understanding of how limitations in perception affect model building could be applied by the system to generate future hypotheses for testing, perhaps to build confidence in a hypothesis, or rule out plausible counter-factuals. The system can then identify further actions in order to achieve this, with the aim of gaining a more complete picture of the phenomenon in the future. In this case, and being able to reason about this process, the system's response might instead be "I am not sure yet, but I am going to find out."

Here we have focussed on the limitations of a system's knowledge based on the inevitable insufficiencies in its sensors, but similar arguments can be constructed regarding limitations on its reasoning processes. In some instances, such cases will be easy to fix, given that they have been identified, as they may just require more computation or the substitution of another algorithm. This may be easier than, say, integrating a new type of sensor for a more complete picture.

Situatedness factors will not all be physical: The system's membership of a group may lead to the availability of additional sources of information, from others. Of course, in receiving knowledge provided by other systems, it is important again to acknowledge the subjective way in which the received knowledge was created and factor this subjectivity into the reasoning process. For example, one may receive a strong assertion from a large group of other systems, which all agree on a particular fact about the world. This does not make that fact necessarily correct, if all the other systems have identically limited sensors or reasoning processes, and this is not acknowledged. To quote George Orwell [62], "sanity is not statistical."

Indeed, the importance of appropriate diversity between individual components of a collective (cyber-physical) system in achieving system-wide goals is well-known [63, 64], and the value of a diversity of opinions in machine learning is also long-established [65, 66], yet often non-trivial to achieve in practice [67, 68]. A collective system's awareness of the diversity of the perspectives held within it will be an important feature to develop, to account for and manage diversity of perspectives in an adaptive and effective way. In summary, in self-aware CPSs we are concerned with integrating multiple perspectives, not just multiple data points. This gives us an approach that leads to richer, smarter, and more grounded decision-making rather than simply assuming that we have noisy or partial data.

7 ORGANIZATIONAL INFRASTRUCTURE EXPECTATIONS

Designing and developing future SA CPS, where the system is in full control of all its resources, a more flexible infrastructure is required, supporting the type of adaptable and flexible self-aware processes discussed above.

The main goal of this infrastructure is to allow any part of the system, including these same infrastructure components and processes, to be instrumented, analyzed, evaluated, and potentially replaced, according to current circumstances. For our purposes here, infrastructure does not only refer to the mechanisms for moving bits around within the system (often called "middleware"). It refers primarily to the expectations for and organization of processes that will use those bits to make choices about what the system should do. To make systems that can thrive in the uncertain and dynamically changing world [69], or even just survive for a sufficiently long period to be useful, we need to design them to have the self-awareness capabilities mentioned before.

A reflective architecture is defined [12, 70] as a system that can represent and reason about its own behavior, and use that reasoning to change the behavior effectively. Of course, the depth and pervasiveness of reflection should depend on how confident we are about our knowledge of the variability and predictability of both the operational environment and the system structure itself (and its proper behavior will depend on how correct we are about that knowledge). These basic properties lead to other advantages: reflective systems are

- Robust: They can be effective without change in a larger environment/system behavior space (they are resistant without change to more pressures); this results from an explicit awareness of environmental behaviors and the available choices of potential system responses;
- Resilient: They are flexible enough to make effective changes (they react properly to more varied pressures); this results from the substitutability of diverse resources;
- Explainable: Their decisions can be justified with relevant evidence, either before or after the fact; this results from the explicit knowledge of resources and their capabilities, and the diverse instrumentation available, and the decision processes that use that information;
- Capable of integration and cooperation: They can combine with others for a common enough goal (they can join systems of systems); the latter is intended for more loosely coupled systems, and the former for more tightly coupled systems; this results from the explicit self-models and the modeling capabilities available to model the other relevant systems.

An example of this kind of infrastructure is Wrappings [71–73]; here we describe the basic structure that this approach provides. We do not claim that it is the only way to maintain this flexibility, but it has been shown to be quite convenient for these challenges. A Wrapping is the meta-knowledge required to select and apply computational resources for particular problems in a specific context, and it is combined with the processes that use that meta-knowledge to organize how the resources are selected, adapted, integrated, explained and evaluated for different system goals.

First, there is a strong separation between the computational resources available and the tasks they are expected to perform. The "Problem Posing Paradigm" identifies all requests for computational or information services as *posed problems*, and all entities or data sources that can provide that information service as *resources*. As soon as these conceptually different name spaces are recognized, a system can map from problems to resources in a number of different and very flexible ways. Wrappings has chosen Context-Dependent Knowledge-Based Polymorphism, which computes the mapping based on current context, so that exactly the same problem can map to different resources in different contexts.

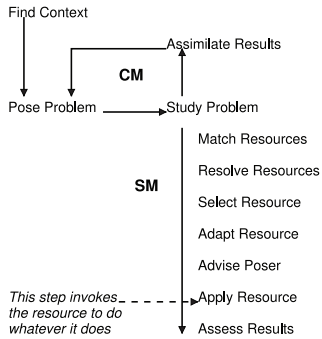


Fig. 2. CM and SM Steps.

Second, the processes that read the Wrappings to organize the context and compute these mappings are as important as the Wrappings themselves (declarative knowledge does nothing without an interpreter, and Wrappings makes the interpreter explicit and selectable in context). These *Problem Managers* organize the overall processing by collecting posed problems, addressing them using the Wrappings and the available resources, and assessing the results to change the relevant context. The key reflective aspect for Problem Managers is that they are also resources, they are also Wrapped, and they are also selected for processing and applied in exactly the same way as any other resource. This provides complete computational reflection to any desired level of detail, which means that any basic processing loop, such as *OODA* [74] or *MAPE-K* [75–77], is easily implemented in a very flexible way using this approach.

The main approach is shown in Figure 2, which shows the two default Problem Managers. The CM (Coordination Manager) is the basic heartbeat that moves the system through its processes. The SM (Study Manager) is the simplest planner that reads the Wrapping Knowledge Base(s) to choose what resources to apply (and replaced by other SMs when appropriate). The most important part of this approach is that these two default Problem Managers and the computational steps shown in the picture are also implemented by resources that are also Wrapped, so they can be changed at any time, according to context (and the rules in the Wrappings). Thus there are no privileged resources anywhere in the system. Full details may be found in the cited references.

The short term wins of this approach are as follows:

- There is an immediate separation from superficial implementations ("layers of symbol systems") via the Problem Posing Programming Paradigm;
- The system has the ability to "compile out" decisions that will be made the same way every time (partial evaluation) and still retain the option of revisiting that simplification when conditions change;
- There is a record of all internal activity (including decisions made and reasons for them) that can be queried or modeled for explanation.

The full depth of reflection defined in Wrappings is often more than is needed for particular applications [72, 78]. It is important to have explicit criteria or even guidelines for assessing how much reflection is required, based on the designers' expectations for flexibility and adaptability, and how much we as designers need to know about the problem, the environment, and the system to make those decisions. Since we as designers cannot always know enough to make that decision at design time, we need to build a system with pending design decisions that can only be constructed once the system is in its operational environment, and therefore we need to have our

system able to fill in the necessary information based on its own observations and models of the behavior of that environment.

There are several derived long term challenges in this approach:

- We need to develop implementable guidelines for the system to determine for itself at runtime how much reflection it needs just now; and methods for compiling infrastructure components in or out of the basic processing;
- We need appropriate languages and processes for specifying system behavior; for modeling external and internal phenomena, for maintaining model integrity, for changing representational mechanisms (and deciding when that is needed);
- The systems need careful knowledge management to account for limited resources [72].
- A designer has to consider the resources required to process and manage the knowledge as well as resources themselves during runtime.

These challenges were derived from one particular approach to the problem, but related challenges will occur for any approach to providing a robust and flexible infrastructure and architecture for a self-aware system. The engineering question is how much self-awareness is available for how much investment.

8 VERIFICATION OF SELF-AWARENESS AND SELF-AWARE SYSTEMS GUARANTEES

Verification of a system's ability to reach a specific state or to achieve given goals is the subject of research in the area of model checking [79] and runtime verification [80, 81]. This is done at runtime and is based on expected behavior and pre-defined *a priori* models of the states of the system and the environment. Verification of capabilities, resources, and behavior of a self-aware system, however, requires the system to learn these models continuously during runtime, since they can be expected to change at any time. While a system might want to verify its own capabilities during runtime in rapidly unfolding situations, it might also need to verify capabilities of other systems in its environment with which it might be interacting. Many verification techniques are applicable to self-aware CPSs [82] when it comes to resources and desired behavior. Therefore we focus on the specific issue here of verifying self-awareness capabilities of systems.

In a self-aware CPSs, the behavior is affected by the self-awareness capabilities of the system. To give guarantees and maintain them during runtime, the system also has to verify its own level of self-awareness and associate each level with respective guarantees. To achieve this, a system requires some form of meta-self-awareness, allowing it to distinguish different levels of self-awareness. A question arises whether such a meta-self-awareness can be pre-defined by a designer, specifically when self-aware capabilities can also be subject to change due to experiences during runtime?

Verifying the self-awareness capabilities of another system becomes invaluable when the self-aware CPS is expected to autonomously interact and potentially collaborate. This, in turn, will become more and more important with the rising number of systems in our environment relying on collaboration with others to overcome tasks even with constrained resources. For external verification of levels of self-awareness a designer can consider Black-Box and White-Box verification [83]. In addition to the question above, a system has to map observations of another systems on internal models of self-awareness. However, these mappings might be incorrect and lead to false conclusions. A designer, therefore, has to consider the implications of determining the wrong level of self-awareness.

But even constraining the degree of autonomy for a system to take actions unavoidably raises questions along the following line: Are the actions taken always in favour of the functional and extra-functional requirements and constraints of the system? If not, can unfavourable decisions

be reversed? Can the effects of unfavourable actions be bounded, or in general, what kind of guarantees can be expected from such a self-aware system?

CPS often consist of a heterogeneous mix of tasks with different criticality levels (mixed-critical systems). Usually, we find tasks that do have more or less stringent real-time, safety, security or power/energy requirements, but there are also tasks that have more of a best effort nature. There is no reason to assume that a self-aware system must apply the same effort and energy for autonomous decision making to all of them. Hence, the system can be more "exploratory" with those tasks, versus more conservative and restricted with real-time or safety critical tasks [84]. However, this assumption implies that mechanisms for proper isolation of applications (process, interconnect and memory virtualisation techniques, hypervisors) are in place. Thus, for critical tasks the same guarantees can be made as in conventionally designed systems. In turn, this means, self-awareness does not introduce an additional risk for them.

In a recent paper [85] it was argued that there are three classes of guarantees that can be developed for a self-aware CPS: (1) Fault detection, isolation, and recovery (FDIR) and top-down failure analysis; (2) Minimal self-protection; and (3) Minimal acceptable performance. These are briefly summarized below.

Fault detection, isolation, and recovery. This a subfield of control engineering that concerns itself with monitoring a system, identifying when a fault has occurred, and pinpointing the type of fault and its location. We are using it here as a shorthand for a large number of related activities, including analysing the system for both component and subsystem failures, as well as mission or enterprise-wide failures.

Just as in traditional systems, we identify all the paths that can cause critical failures and then show that these failures have been provably avoided or at least, mitigated, and that suitable instrumentation and failure recovery processes are in place. For self-aware CPSs, this will take some new thinking when developing novel fault detection and mitigation strategies because of the dynamic range of behaviors enabled by self-* capabilities. However, whether a critical failure occurs because of a broken component or badly implemented interface (as often occurs in traditional systems) or because of a badly implemented decision process leading to the wrong response (as might occur in a self-aware CPS), one can still enumerate the critical failures in the system. Here the sophisticated learning, reflection and reasoning of a self-aware system could prove to be an advantage in perceiving and reasoning about a potential failure state. In addition, a self-aware CPS with self-testing capabilities for checking in on certain states and capabilities as often as necessary based on the seriousness of a given domain would potentially allow to detect such failure states in a timely manner.

Minimal self-protection. The next major category is essentially analysis and testing that demonstrates that the system can "safe" itself appropriately. Space systems have a slew of behaviors that are critical to putting the system into a safe state when confronted by unexpected conditions in outer space or on other planets. Animals also have a version of safing in their reflexes (for fast, immediate responses) and in their behavioral patterns (e.g., fleeing, hiding, aggressive displays, etc.) with which they react to unexpected and potentially dangerous circumstances. By knowing what interactions can harm it, the self-aware system could actually help other systems to shape the solutions for their integration and combined performance. However, with the ability to adapt its own actions and interactions with others, a self-aware CPS needs to determine such harmful actions during runtime.

Minimal acceptable performance. The third class of guarantees will ensure that the self-aware system is able to do essential functions and minimal acceptable performance. Obviously this type of guarantee is closely related to the first class of critical failures. Unlike the question of how the

system breaks and fails, this is the identification of what the system needs to maintain among its own computational resources to perform its most essential functions. This might include information about minimal durations needed to perform given behaviors or energy required during a certain task. This knowledge then sets boundaries and constraints on what the system must retain when negotiating with other systems for possible collaboration, tradeoffs, and compromises. This information can also be communicated to other systems and will help again to constrain what sorts of tradeoffs and collaborative solutions are possible.

As can be seen from this brief discussion, a self-aware system does not suddenly change the need for careful design and analysis of a system; one still needs to carefully define *a priori* the overall purposes for the system and what behaviors it must and must not do. Not enumerating all possible operational situations ahead of time does not mean that the system has no constraints or rules that it must operate by. To the contrary, its range of acceptable behaviors and responses are carefully specified. However, when accomplished, the self-awareness of the system can in fact contribute to identifying failures and facilitate sensible responses.

9 CONCLUSIONS AND DISCUSSION

We have argued the need for techniques that embody the notion of self-awareness in resource constrained CPSs. Self-awareness in computing systems has been researched for over 30 years, leading to a range of concepts and techniques that have also been applied in many application domains. We reviewed very briefly this state of the art in Section 2.2 and argued then in Section 2.3 that CPSs pose specific challenges that are different from many other computing systems. In Section 3 we detail the CPS challenges and conclude with identifying five key issues: (1) Developing new design and engineering processes, (2) Building resource-sensitive self-awareness, (3) Acknowledging situatedness and subjectivity of self-awareness, (4) Creating an infrastructure for self-awareness processes, and (5) Verifying self-awareness and establishing guarantees.

These specific topics were then elaborated in Sections 4 through 8.

In the following we briefly discuss the domain-specific risks in combination with self-awareness and propose a road-map to achieve the challenges outlined in this article.

9.1 Domain-Appropriate Self-awareness and Risk

We motivated the challenges and contextualised the discussion in this article using three examples: an implanted insulin pump, a Mars rover, and a multicore system on chip. These examples are very different, but they share with many other CPSs the challenges of limited resources, complex unforeseeable environmental dynamics, and high expectations on their reliability. They also share the feature of substantial levels of risk, arising from potential malfunctions or action insensitive to the current context, and their entanglement with the outside world. In any design process, engineers must confront questions of risk. Typically, these questions are of the form: What are the consequences of the system malfunctioning (in different ways) within certain anticipated environmental conditions? This leads to a process designed to ensure such malfunctioning cannot occur, or minimise the probability or impact of a malfunction to within acceptable limits. However, as engineers must also ask: what are the consequences of a system continuing to operate in the way it was designed to, in unanticipated environmental conditions?

In the former question, a well-formed design and test process can ensure that the system is able to recover from errors, or perhaps disengage or deactivate when it enters an anticipated failure state. But it is in answering the latter question that we are encouraged to think differently: if the system enters operating conditions that were not foreseen, is the correct action to continue to operate as before? Or is the correct action to stop normal behavior and deactivate? It is easy to see why there is no simple answer. In an unknown environmental state, the mars rover, disconnected

from human operators, might pose danger to itself when continuing to operate. Halting all operations and waiting for additional assessment the situation and more information may or may not allow the rover to avoid harming itself. Conversely, for a pacemaker or insulin pump to deactivate upon entering an unanticipated state may pose serious dangers to life. But if that unanticipated state is a zero-day attack from a hostile actor, then either shutting down or continuing to operate normally may be equally risky.

As recently as August 31, 2017, the USA FDA has been announcing voluntary recalls of pacemakers because of cyber-attack potentials [86]. The FDA stated "This [cyber-attack] access could be used to modify programming commands to the implanted pacemaker, which could result in patient harm from rapid battery depletion or administration of inappropriate pacing." These recalls included over 400,000 implanted heart devices. Although the industry is working on patching these risks, as late as June 2019, Medtronic recalled their insulin pumps as the USA FDA announced additional hacking risks [87].

While such devices apparently demand cyber security to avoid leaking private information and being hacked, they also need to be accessed by appropriate personnel during emergencies or for routine check-ups. Such contradictory objectives between security and accessibility hold for many smaller CPS and IoT devices, and may be amenable to some simple local decision-making by the device. For example, a simple set of rules could allow the device to respond to abnormal command patterns by going into a safe mode and reporting the abnormality.

This argument leaves us with two options. The first is that we permit the use of devices, and thus certify them, only when all operating conditions can be anticipated, enumerated, and factored in to the design. And to be clear, a complete bespoke system will always be better than a generic adaptive one: asymptotically at least, it is always possible to build a better bespoke system, but as complexity continues to increase, we will tend toward a world full of individual complex bespoke systems, potentially interacting with each other. Alternatively, our second option is based on the assumption that, practically, this cannot be achieved due to environmental complexity and uncertainty. As a result, a new requirement on these devices emerges: the ability for the state of the device and its behavior in its environment to be monitored in real time. While it is conceivable, at least in principle, that infrastructure for a centralised and remote form of this monitoring could be constructed, this adds both a layer of communication overhead and an additional attack vector to the system that can be avoided by taking a *local self-monitoring* approach. This is what leads us to begin to consider self-awareness even for simpler CPSs and even (or perhaps especially) in high-risk deployments.

In this article, we do not make the argument that insulin pumps *ought*, at present, to contain at least a minimal form of self-awareness. This is beyond the scope of this article. However, we believe it is important in the context of this discussion to outline *how such decisions ought to be made* and justified. Indeed, outlining one way to make such arguments in a well-founded way is an important task in itself, given how statements concerning the use, suitability, and acceptance of self-awareness and intelligent decision-making in high risk applications often lead to controversy and anxiety.

Specifically, consider the very many cases where traditionally, certification for use of a system involves a full understanding of the behavior of a system, assuming an operating envelope. Should such a system ever, in principle, be augmented with self-awareness? At first glance, it appears obvious that doing so serves to increase uncertainty and risk, and as a result such self-aware CPSs *ought not* to be certified. But on reflection, we can also ask the following question: can we conceive of an operating environment where a certifying authority *ought not* to authorise an insulin pump unless it had at least some form of self-monitoring? If so, then what does that operating environment look like, and what sort of self-monitoring would be required to bring the device back to

operating within an acceptable level of risk? Moreover, how can we *guarantee* that the system will ask for help when it does not know what else to do?

Indeed, we believe it is incumbent upon engineers to ask: could environmental conditions occur during operation in which continued application of the designed rules increases risk? In many complex human scenarios, it is commonly acknowledged that situations can occur that could not have been foreseen when the rules were written. This is a common feature of constitutional law, where the word "normally" is often used to condition the application of a rule, and discretion is expected to be applied in its application. Of course, the application of discretion on a decision in a particular context is a complex cognitive function; many advanced intelligent and self-aware systems will have similar features, but possessing such a feature is not required for the sort of minimal self-awareness we are discussing in the case of the insulin pump. Indeed, in the insulin pump case, one possible and simple approach would be to augment current controllers with a layer of self-monitoring, able to identify unusual patterns in behavior and operating conditions, indicative of an unanticipated state, and alerting a human in that case. If such operating conditions can exist, then when considering certification for use of such a device *today*, we should also ask what are the features of these conditions, and most importantly, are those features already present?

Thus, while the inclusion of a simple and high-risk case like an insulin pump or pacemaker in this article may seem an unlikely choice, it is clear how consideration of this extreme of the spectrum of self-aware CPS opens up new considerations. It is our contention that one ought not to avoid discussion of self-awareness in these cases on the basis that introducing adaptive behavior will automatically increase risk. As per the argument that we have sketched above, this may or may not be the case, and adaptive behavior based on self-monitoring may in fact turn out to be a route to decreasing risk in an uncertain operating environment.

9.2 Research Agenda

Although there exist plenty of excellent technical solutions to many of the specific sub-problems for future CPSs, the key point to realize is that we cannot hard-code these solutions into CPS and deploy them independently of each other or of the application domain. At their system level, future CPS need to reconcile all individual demands and sub-system solutions with each other. Subsystems should always contribute to the global objectives (or to secondary agenda or to longer-term less immediate agenda, etc.), and short-term urgencies have to be traded off against long-term goals.

We argue that only a comprehensive assessment of the situation allows the system to take all relevant aspects into account according to their respective importance. Such an assessment obviously requires reasoning capabilities and must take place at runtime to achieve the required flexibility. The overall system, including its goals and its expectations about its environment, should be subject to the reasoning processes. While full self-awareness may turn out to be the most cost efficient solution for some CPSs despite its overhead, for other applications it is clear that only some of the self-awareness techniques will be relevant and useful. Rather than engineering aspects of self-awareness into specific applications and systems, we strongly advocate for studying more general approaches and methods that can be applicable to a broad spectrum of self-aware CPSs. As the main conclusion, we thus propose *Design for Self-awareness (DfSA)* as opposed to designing individual systems to be self-aware. To this end, we contend that we need to launch and reinforce research efforts addressing the following important topics, with the goal to better understand the challenges of and deliver on the promise for self-aware CPSs:

- **Developing new design and engineering processes:** By moving decisions from design time to runtime, we change the nature of design questions toward those relevant to when, how, and how much to integrate different forms of self-awareness, and how to design for flexibility. While design templates, reference architectures, and generic engineering processes are available for self-aware systems, they need to be adapted to self-aware CPSs.
- **Building resource-sensitive self-awareness:** Besides the classic consideration of resource constraints during the design of self-aware CPSs, we additionally need to introduce awareness of operating under limited resources to the system itself. Approaches must be explored that enable a system to flexibly map required resources to different processes at runtime, including the processes that allow it to reflect on itself.
- **Acknowledging situatedness and subjectivity of self-awareness:** We require techniques enabling systems to reflect on their own perspective to ground what they know and factor this into decision making. This will also allow them to harness the value of multiple perspectives.
- **Creating an infrastructure for self-awareness processes:** Generic infrastructure platforms and components need to be developed to facilitate the development of self-aware CPSs in a systematic, resusable, and reliable way. Monitoring, reflection, reasoning and adaptation will all benefit from implementation support and corresponding tooling, even in the simplest of systems. Returning again to the insulin pump example, self-awareness-ready platforms can help raise awareness of and mitigate risks introduced by unanticipated environmental threats and by added freedom in behavior space. For example, when it is unacceptable for an insulin pump to behave outside of the doctor's prescribed range of parameters then it is obviously welcomed when the system has abilities to alert humans about any unexpected dynamics in multidimensional measurement data space. The discovery of these requires elevated awareness of the internal and external environment, as discussed in Section 9.1.
- **Verifying self-awareness and establishing guarantees:** Systems that change their own behavior fundamentally at runtime pose novel challenges for verification and might even require a re-thinking of how guarantees can be defined. Instead of verifying or giving guarantees for the overall system over time, current approaches strive to provide guarantees for sub-processes and situations where guarantees are required (e.g., safety- or time-critical situations) or to guarantee boundaries that the system will not overstep. Importantly, the system itself is required to have an understanding and an awareness of its own guarantees to ensure it operates within.

REFERENCES

[1] E. A. Lee. 2007. Computing Foundations and Practice for Cyber Physical Systems: A Preliminary Report. Technical Report. Technical Report No. UCB/EECS-2007-72.
[2] Jean-Michel Bergé, Oz Levia, and Jacques (editors) Rouillard. 1995. *High-level System Modeling: Specification Languages*. Springer.
[3] H. Kopetz. 1997. *Real-Time Systems: Design Principles for Distributed Embedded Applications*. Springer.
[4] Peter R. Lewis, Marco Platzner, Bernhard Rinner, Jim Torresen, and Xin Yao (Eds.). 2016. *Self-aware Computing Systems: An Engineering Approach*. Springer.
[5] Samuel Kounev, Jeffrey O. Kephart, Aleksandar Milenkowski, and Xiaoyun Zhu (Eds.). 2017. *Self-aware Computing Systems*. Springer.
[6] Steve Chien and Kiri L. Wagstaff. 2017. Robotic space exploration agents. *Sci. Robot.* 2, 7 (2017).
[7] Anna Nowogrodzki. 2016. How does Mars rover Curiosity's new AI system work? Retrieved August 2016 from http://www.astronomy.com/news/2016/08/how-does-mars-rover-curiositys-new-ai-system-work.
[8] W. Burleson, S. S. Clark, B. Ransford, and K. Fu. 2012. Design challenges for secure implantable medical devices. In *Proceedings of the Design Automation Conference (DAC'12)*. 12–17.

[9] Benjamin Ransford, Daniel B. Kramer, Denis Foo Kune, Julio Auto de Medeiros, Chen Yan, Wenyuan Xu, Thomas Crawford, and Kevin Fu. Cybersecurity and medical devices: A practical guide for cardiac electrophysiologists. *Pacing Clin. Electrophysiol.* 40, 8, 913–917.

[10] D. Halperin, T. S. Heydt-Benjamin, K. Fu, T. Kohno, and W. H. Maisel. 2008. Security and privacy for implantable medical devices. *IEEE Perv. Comput.* 7, 1 (Jan. 2008), 30–39.

[11] Sarbari Gupta. 2012. Implantable medical devices – Cyber risks and mitigation approaches. In *Proceedings of the NIST Cyber Physical Systems Workshop.*

[12] Christopher Landauer and Kirstie L. Bellman. 2000. Reflective infrastructure for autonomous systems. In *Proceedings of the 15th European Meeting on Cybernetics and Systems Research (EMCSR'00).* 671–676.

[13] M. T. Cox. 2005. Metacognition in computation: A selected research review. *Art. Int.* 169, 2 (2005), 104–141.

[14] Peter R. Lewis. 2017. Self-aware computing systems: From psychology to engineering. In *Proceedings of the 2017 Design, Automation & Test in Europe Conference & Exhibition (DATE'17).* 1044–1049.

[15] Peter R. Lewis, Arjun Chandra, Funmilade Faniyi, Kyrre Glette, Tao Chen, Rami Bahsoon, Jim Torresen, and Xin Yao. 2015. Architectural aspects of self-aware and self-expressive computing systems. *IEEE Comput.* 48, 8 (2015), 62–70.

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

[17] Peter Lewis, Kirstie Bellman, Chris Landauer, Lukas Esterle, Kyrre Glette, Ada Diaconescu, and Holger Giese. 2017. Towards a framework for the levels and aspects of self-aware computing systems. In *Self-Aware Computing Systems*, Samuel Kounev, Jeffrey O. Kephart, Aleksandar Milenkoski, and Xiaoyun Zhu (Eds.). Springer, 51–85.

[18] H. Hoffmann et al. 2013. A generalized software framework for accurate and efficient management of performance goals. In *Proceedings of the International Conference on Embedded Software.* 1–10.

[19] IBM Corporation. 2006. An Architectural Blueprint for Autonomic Computing. IBM White Paper.

[20] Samuel Kounev, Peter Lewis, Kirstie Bellman, Nelly Bencomo, Javier Camara, Ada Diaconescu, Lukas Esterle, Kurt Geihs, Holger Giese, Sebastian Götz, Paola Inverardi, Jeffrey Kephart, and Andrea Zisman. 2017. The notion of self-aware computing. In *Self-Aware Computing Systems*, Samuel Kounev, Jeffrey O. Kephart, Aleksandar Milenkoski, and Xiaoyun Zhu (Eds.). Springer, 3–16.

[21] L. D. Paulson. 2003. DARPA creating self-aware computing. *IEEE Comput/* 36, 3 (Mar. 2003), 24.

[22] European Commission. 2013. Self-Awareness in Autonomic Systems.

[23] Jeremy Pitt (Ed.). 2014. *The Computer After Me.* Imperial College Press/World Scientific Book.

[24] Peter R. Lewis, Arjun Chandra, Shaun Parsons, Edward Robinson, Kyrre Glette, Rami Bahsoon, Jim Torresen, and Xin Yao. 2011. A survey of self-awareness and its application in computing systems. In *Proceedings of the International Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW'11).* IEEE Computer Society, 102–107.

[25] J. Schaumeier, J. Jeremy Pitt, and G. Cabri. 2012. A tripartite analytic framework for characterising awareness and self-awareness in autonomic systems research. In *Proceedings of the 6th IEEE Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW'12).* IEEE Computer Society, 157–162.

[26] Axel Jantsch, Nikil Dutt, and Amir M. Rahmani. 2017. Self-awareness in systems on chip—A survey. *IEEE Des. Test* 34, 6 (2017), 8–26.

[27] Tao Chen, Rami Bahsoon, and Xin Yao. 2018. A survey and taxonomy of self-aware and self-adaptive cloud autoscaling systems. *Comput. Surv.* 51, 06 (2018).

[28] Lukas Esterle and Radu Grosu. 2016. Cyber-physical systems: Challenge of the 21st century. *Elektrotech. Informationstech.* 133, 7 (01 Nov 2016), 299–303.

[29] Aamir Akbar and Peter R. Lewis. 2018. Self-adaptive and self-aware mobile-cloud hybrid robotics. In *Proceedings of the 4th International Workshop on Mobile Cloud Computing systems, Management, and Security (MCSMS'18).* To appear.

[30] Andreas Agne, Markus Happe, Achim Lösch, Christian Plessl, and Marco Platzner. 2014. Self-awareness as a model for designing and operating heterogeneous multicores. *ACM Trans. Reconfig. Technol. Syst.* 7, 2 (Jul. 2014), 13:1–13:18.

[31] Ariane Keller, Daniel Borkmann, Stephan Neuhaus, and Markus Happe. 2014. Self-awareness in computer networks. *Int. J. Reconfig. Comput.* 2014, Article 10 (Jan. 2014), 1 pages.

[32] Bernhard Rinner, Lukas Esterle, Jennifer Simonjan, Georg Nebehay, Roman Pflugfelder, Peter R. Lewis, and Gustavo Fernandez Dominguez. 2015. Self-aware and self-expressive camera networks. *IEEE Comput.* 48, 7 (2015), 21–28.

[33] M. Götzinger, N. TaheriNejad, H. A. Kholerdi, A. Jantsch, E. Willegger, T. Glatzl, A. M. Rahmani, T. Sauter, and P. Liljeberg. 2019. Model-free monitoring with confidence. *Int. J. Comput. Integr. Manufact.* 32, 4–5 (2019), 466–481. DOI: http://dx.doi.org/10.1080/0951192X.2019.1605201 arXiv:https://doi.org/10.1080/0951192X.2019.1605201

[34] Maciej Kurek, Tobias Becker, Ce Guo, Stewart Denholm, Andreea-Ingrid Funie, Mark Salmon, Tim Todman, and Wayne Luk. 2016. Self-aware hardware acceleration of financial applications on a heterogeneous cluster. In *Self-aware Computing Systems: An Engineering Approach.* Springer.

[35] M. Götzinger, A. Anzanpour, I. Azimi, N. TaheriNejad, A. Jantsch, A. Rahmani, and P. Liljeberg. 2019. Confidence-enhanced early warning score based on fuzzy logic. *ACM/Springer Mobile Networks and Applications* (August 2019), 1–18. DOI: http://dx.doi.org/10.1007/s11036-019-01324-5

[36] K. Nymoen, A. Chandra, and J. Torresen. 2016. Self-awareness in active music systems. In *Self-aware Computing Systems: An Engineering Approach*. Springer.

[37] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS'12)*, Vol. 1. Curran Associates, 1097–1105.

[38] K. Neshatpour, F. Behnia, H. Homayoun, and A. Sasan. 2018. ICNN: An iterative implementation of convolutional neural networks to enable energy and computational complexity aware dynamic approximation. In *Proceedings of the 2018 Design, Automation Test in Europe Conference Exhibition (DATE'18)*. 551–556.

[39] Farnaz Forooghifar, Amir Aminifar, and David Atienza Alonso. 2018. Self-aware wearable systems in epileptic seizure detection. In *21st Euromicro Conference on Digital System Design (DSD'18)*. IEEE, 426–432.

[40] H. A. Kholerdi, N. TaheriNejad, and A. Jantsch. 2018. Enhancement of classification of small data sets using self-awareness; an iris flower case-study. In *Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS'18)*. 1–5.

[41] N. TaheriNejad and A. Jantsch. 2019. Improved machine learning using confidence. In *Proceedings of the 2019 IEEE 32nd Canadian Conference on Electrical and Computer Engineering (CCECE'19)*. 1–5.

[42] N. TaheriNejad, A. Jantsch, and D. Pollreisz. 2016. Comprehensive observation and its role in self-awareness; an emotion recognition system example. In *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS'16)*.

[43] N. TaheriNejad, M. A. Shami, and S. M. P. D. 2017. Self-aware sensing and attention-based data collection in multi-processor system-on-chips. In *Proceedings of the 2017 15th IEEE International New Circuits and Systems Conference (NEWCAS'17)*. 81–84.

[44] Arman Anzanpour, Iman Azimi, Maximilian Götzinger, Amir M. Rahmani, Nima TaheriNejad, Pasi Liljeberg, Axel Jantsch, and Nikil Dutt. 2017. Self-awareness in remote health monitoring systems using wearable electronics. In *Proceedings of the Design and Test Europe Conference (DATE'17)*.

[45] Sparsh Mittal. 2016. A survey of techniques for approximate computing. *Comput. Surv.* 48, 4 (Mar. 2016).

[46] Qiang Xu, Todd Mytkowicz, and Nam Sung Kim. 2016. Approximate computing: A survey. *IEEE Des. Test* 33, 1 (2016), 8–22.

[47] Armin Alaghi and John P. Hayes. 2013. Survey of stochastic computing. *ACM Trans. Embed. Comput. Syst.* 12, 2 (May 2013).

[48] E. P. Kim and N. R. Shanbhag. 2014. Energy-efficient accelerator architecture for stereo image matching using approximate computing and statistical error compensation. In *Proceedings of the IEEE Global Conference Signal and Information Processing (GlobalSIP'14)*.

[49] Fei Qiao, Ni Zhou, Yuanchang Chen, and Huazhong Yang. 2015. Approximate computing in chrominance cache for image/video processing. In *Proceedings of the IEEE International Conference on Multimedia Big Data*. 180–183.

[50] Zidong Du, Robert Fasthuber, Tianshi Chen, Paolo Ienne, Ling Li, Xiaobing Feng, Yunji Chen, and Olivier Temam. 2015. ShiDianNao: Shifting vision processing closer to the sensor. In *Proceedings of IEEE/ACM International Symposium on Computer Architecture*.

[51] Jiachao Deng, Yuntan Fang, Zidong Du, Ying Wang, and Huawei L. 2015. Retraining-based timing error mitigation for hardware neural networks. In *Proceedings of the Conference on Design, Automation and Test in Europe*. 593–596.

[52] N. Taherinejad, L. Lampe, and S. Mirabbasi. 2014. Adaptive impedance matching for vehicular power line communication systems. In *Proceedings of the 18th IEEE International Symposium on Power Line Communications and Its Applications*. 214–219.

[53] Z. Sheng, A. Kenarsari-Anhari, N. Taherinejad, and V. C. M. Leung. 2016. A multichannel medium access control protocol for vehicular power line communication systems. *IEEE Trans. Vehic. Technol.* 65, 2 (Feb. 2016), 542–554.

[54] N. Taherinejad, L. Lampe, and S. Mirabbasi. 2017. An adaptive impedance-matching system for vehicular power line communication. *IEEE Trans. Vehic. Technol.* 66, 2 (Feb. 2017), 927–940.

[55] E. Shamsa, A. Kanduri, N. Taherinejad, A. Proebstl, S. Chakraborty, A. M. Rahmani, and P. Liljeberg. 2020. User-centric resource management for embedded multi-core processors. In *Proceedings of 33rd IEEE Internation Conference on VLSI Design and Embedded Systems (VLSID'20)*. 1–6.

[56] Tao Chen, Funmilade Faniyi, Rami Bahsoon, Peter R. Lewis, Xin Yao, Leandro L. Minku, and Lukas Esterle. 2014. The handbook of engineering self-aware and self-expressive systems. *CoRR* abs/1409.1793 (2014).

[57] Betty H. C. Cheng, Rogério de Lemos, Holger Giese, Paola Inverardi, Jeff Magee, Jesper Andersson, Basil Becker, Nelly Bencomo, Yuriy Brun, Bojan Cukic, Giovanna Di Marzo Serugendo, Schahram Dustdar, Anthony Finkelstein, Cristina Gacek, Kurt Geihs, Vincenzo Grassi, Gabor Karsai, Holger M. Kienle, Jeff Kramer, Marin Litoiu, Sam Malek, Raffaela Mirandola, Hausi Müller, Sooyong Park, Mary Shaw, Matthias Tichy, Massimo Tivoli, Danny Weyns, and Jon Whittle. 2009. Software engineering for self-adaptive systems: A research roadmap. In *Software Engineering for*

Self-Adaptive Systems, Betty H. C. Cheng, Rogério de Lemos, Holger Giese, Paola Inverardi, and Jeff Magee (Eds.). Lecture Notes in Computer Science (LNCS), Vol. 5525. Springer, 1–26.

[58] Shelley Duval and Robert A. Wicklund. 1972. *A Theory of Objective Self Awareness*. Academic Press.

[59] Lukas Esterle, Peter R. Lewis, Richie McBride, and Xin Yao. 2017. The future of camera networks: Staying smart in a chaotic world. In *Proceedings of the 11th International Conference on Distributed Smart Cameras (ICDSC'17)*. ACM, New York, NY, 163–168.

[60] Maximilian Götzinger, Arman Azanpour, Iman Azimi, Nima Taherinejad, and Amir M Rahmani. 2017. Enhancing the self-aware early warning score system through fuzzified data reliability assessment. In *Proceedings of the International Conference on Wireless Mobile Communication and Healthcare*. Springer.

[61] Anthony Stein, Sven Tomforde, Ada Diaconescu, Jörg Hähner, and Christian Müller-Schloer. 2018. A concept for proactive knowledge construction in self-learning autonomous systems. In *Proceedings of the 2018 IEEE 3rd International Workshops on Foundations and Applications of Self* Systems (FAS*W'18)*.

[62] George Orwell. 1949. *Nineteen Eighty-Four. A Novel*.Secker & Warburg, London.

[63] Peter R. Lewis, Lukas Esterle, Arjun Chandra, Bernhard Rinner, and Xin Yao. 2013. Learning to be different: Heterogeneity and efficiency in distributed smart camera networks. In *Proceedings of the 7th IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO'13)*. IEEE Press, 209–218.

[64] Peter R. Lewis, Harry Goldingay, and Vivek Nallur. 2014. It's good to be different: Diversity, heterogeneity and dynamics in collective systems. In *Proceedings of the 8th IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASO'14)*. IEEE Computer Society Press, 84–89.

[65] Yong Liu and Xin Yao. 1999. Ensemble learning via negative correlation. *Neur. Netw.* 12, 10 (1999), 1399–1404.

[66] Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. 2005. Diversity creation methods: A survey and categorisation. *Inf. Fus.* 6, 1 (2005), 5–20.

[67] Georg Nebehay, Walter Chibamu, Peter R. Lewis, Arjun Chandra, Roman Pflugfelder, and Xin Yao. 2013. Can diversity amongst learners improve online object tracking? In *Multiple Classifier Systems*. Springer, Berlin, 212–223.

[68] Harry Goldingay and Peter R. Lewis. 2014. A taxonomy of heterogeneity and dynamics in particle swarm optimisation. In *Parallel Problem Solving from Nature–PPSN XIII*,Lecture Notes in Computer Science, Vol. 8672. Springer, 171–180.

[69] Edwin Hutchins. 1995. *Cognition in the Wild*. MIT Press.

[70] Pattie Maes. 1987. Concepts and experiments in computational reflection. In *Proceedings of the Conference Proceedings on Object-oriented Programming Systems, Languages and Applications (OOPSLA'87)*. ACM, New York, NY, 147–155.

[71] C. Landauer and K. L. Bellman. 1999. Generic programming, partial evaluation, and a new programming paradigm. In *Software Process Improvement*,Gene McGuire (ed.). Idea Group Publishing, 108–154.

[72] C. Landauer. 2017. Mitigating the inevitable failure of knowledge representation. In *Proceedings of the 2017 IEEE International Conference on Autonomic Computing (ICAC'17)*. 239–246.

[73] C. Landauer and K. L. Bellman. 2017. Integration by negotiated behavior restrictions. In *Proceedings of the 2017 IEEE 2nd International Workshops on Foundations and Applications of Self* Systems (FAS*W'17)*. 117–121.

[74] David G. Ullman. 2017. OO-OO-OO!" the sound of a broken OODA loop. *CrossTalk-The Journal of Defense Software Engineering* (2007), 22–25.

[75] Yuriy Brun, Giovanna Di Marzo Serugendo, Cristina Gacek, Holger Giese, Holger Kienle, Marin Litoiu, Hausi Müller, Mauro Pezzè, and Mary Shaw. 2009. *Engineering Self-Adaptive Systems through Feedback Loops*. Springer, Berlin, 48–70.

[76] Danny Weyns, Sam Malek, and Jesper Andersson. 2010. On decentralized self-adaptation: Lessons from the trenches and challenges for the future. In *Proceedings of the 2010 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS'10)*. ACM, New York, NY, 84–93.

[77] Yuriy Brun, Ron Desmarais, Kurt Geihs, Marin Litoiu, Antonia Lopes, Mary Shaw, and Michael Smit. 2013. *A Design Space for Self-Adaptive Systems*. Springer, Berlin, 33–50.

[78] Christopher Landauer and Kirstie L. Bellman. 2003. Managing self-modeling systems. In *Proceedings of the 3rd International Workshop on Self-Adaptive Software*, R. Laddaga and H. Shrobe (Eds.).

[79] Marta Kwiatkowska, Gethin Norman, and David Parker. 2007. *Stochastic Model Checking*. Springer, Berlin, 220–270.

[80] Scott D. Stoller, Ezio Bartocci, Justin Seyster, Radu Grosu, Klaus Havelund, Scott A. Smolka, and Erez Zadok. 2012. Runtime verification with state estimation. In *Runtime Verification*, Sarfraz Khurshid and Koushik Sen (Eds.). Springer, Berlin, 193–207.

[81] Ezio Bartocci, Radu Grosu, Atul Karmarkar, Scott A. Smolka, Scott D. Stoller, Erez Zadok, and Justin Seyster. 2013. Adaptive runtime verification. In *Runtime Verification*, Shaz Qadeer and Serdar Tasiran (Eds.). Springer, Berlin, 168–182.

[82] Denise Ratasich, Faiq Khalid, Florian Geissler, Radu Grosu, Muhammad Shafique, and Ezio Bartocci. 2019. A Roadmap toward the resilient internet of things for cyber-physical systems. In *IEEE Access*, vol. 7. 13260–13283. DOI: 10.1109/ACCESS.2019.2891969

[83] Lukas Esterle, Kirstie L. Bellman, Steffen Becker, Anne Koziolek, Christopher Landauer, and Peter Lewis. 2017. *Assessing Self-awareness*. Springer International Publishing, Cham, 465–481.

[84] Nikil Dutt, Fadi J. Kurdahi, Rolf Ernst, and Andreas Herkersdorf. 2016. Conquering MPSoC complexity with principles of a self-aware information processing factory. In *Proceedings of the 11th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*. ACM, 37.

[85] Kirstie L. Bellman. 2018. What reasonable guarantees can we make for a SISSY system. In *Proceedings of the 2018 IEEE 3rd International Workshops on Foundations and Applications of Self* Systems (FAS*W'18)*.

[86] US Food and Drug Administration. 2017. Cybersecurity Vulnerabilities Identified in St. Jude Medical's Implantable Cardiac Devices and Merlin@home Transmitter: FDA Safety Communication. Retrieved December 5, 2019 from https://www.fda.gov/medical-devices/safety-communications/cybersecurity-vulnerabilities-identified-st-jude-medicals-implantable-cardiac-devices-and-merlinhome accessed: 2019-12-05.

[87] US Food and Drug Administration. 2019. Cybersecurity Vulnerabilities Affecting Medtronic Implantable Cardiac Devices, Programmers, and Home Monitors: FDA Safety Communication. Retrieved December 5, 2019 from https://www.fda.gov/medical-devices/safety-communications/cybersecurity-vulnerabilities-affecting-medtronic-implantable-cardiac-devices-programmers-and-home.

Received November 2019; revised December 2019; accepted December 2019