Self-aware Computing Systems: From Psychology to Engineering

Peter R. Lewis
Aston Lab for Intelligent Collectives Engineering (ALICE)
Aston University
Birmingham, UK
Email: p.lewis@aston.ac.uk

Abstract—**At the current time, there are several fundamental changes in the way computing systems are being developed, deployed and used. They are becoming increasingly large, heterogeneous, uncertain, dynamic and decentralised. These complexities lead to behaviours during run time that are difficult to understand or predict. One vision for how to rise to this challenge is to endow computing systems with increased self-awareness, in order to enable advanced autonomous adaptive behaviour. A desire for self-awareness has arisen in a variety of areas of computer science and engineering over the last two decades, and more recently a more fundamental understanding of what self-awareness concepts might mean for the design and operation of computing systems has been developed. This draws on self-awareness theories from psychology and other related fields, and has led to a number of contributions in terms of definitions, architectures, algorithms and case studies. This paper introduces some of the main aspects of self-awareness from psychology, that have been used in developing associated notions in computing. It then describes how these concepts have been translated to the computing domain, and provides examples of how their explicit consideration can lead to systems better able to manage trade-offs between conflicting goals at run time in the context of a complex environment, while reducing the need for a priori domain modelling at design or deployment time.**

I. Introduction

Twenty-first century computing systems, from those supporting cloud-based applications, to the Internet of Things, sensor networks or multi-core platforms, can increasingly be characterised in terms of high complexity. This complexity is often multi-faceted, and can include

1) **scale**, as the size of computing systems, or the number of distinct entities comprising them, is increasingly large;
2) **heterogeneity**, where individual entities, components or sub-systems comprising the system, vary from each other, for example in terms of capabilities, capacity or behaviour;
3) **uncertainty** surrounding the composition of the system itself, and/or the environment it is operating in;
4) **ongoing change** again in either the composition of the system or its operating environment; and
5) **decentralisation**, in terms of control, or increasingly even in ownership of constituent parts of the system.

Despite these complexities, modern systems typically have many stakeholders – individuals, communities or organisations for whom the runtime behaviour of the system matters. Classically, these concerns are captured as requirements, or in terms of goal states or objective functions. Increasingly, however, those interacting with or impacted by systems are not well-known until after deployment, and other parties such as those involved in development, operations, management and regulation roles also have concerns which must be considered, and factored in to decision-making.

Stakeholder concerns can vary to a high degree, and can include such qualities as task performance, throughput, flexibility, resource usage, costs, reliability, safety, security, understandability, and many more. This implies that any meaningful evaluation of a system's behaviour must inherently be multi-objective, where relationships between concerns are often characterised in terms of trade-offs, tensions or even full conflict. Classically, an analysis of how different behaviours or configurations can lead to desirable outcomes in a multi-objective space is tackled at design time (e.g., the field of multi-objective optimisation has contributed much to solving these sorts of problems as part of the design process [1]).

While analysis may provide insight into run-time behaviour, in a complex system operating in a poorly understood environment, such qualities are typically only observable at run time. Therefore, increasingly this process needs to be moved online, in order to enable continuous monitoring and, if necessary, redesign of the system, in order to remain effective, as the world changes. Such a process requires not just the designer to have an awareness of how the system behaves, but it requires the system itself to be able to experiment, model, hypothesise and adapt its own configuration and behaviour, in the context of its environment. In short, it requires the system to be aware of itself.

In this paper, I explore recent work that approaches the notion of self-awareness with a view to explicitly designing self-awareness capabilities into computing systems. I will discuss motivations for self-awareness from a variety of different areas of computer science and engineering. Then, I will discuss how theories from our understanding of self-awareness in humans and animals has been inspiring the explicit design of *computational* self-awareness capabilities in technical systems. I will discuss how computational self-awareness may include knowledge of internal state, history, social or physical environment, goals, and even a system's own way of representing and reasoning about these things.

II. Complexity Challenges

It is clear that in almost all cases, the computing systems of the 21st century operate on a **greater scale** than those from decades past, often comprising substantial numbers of components or sub-systems. As an example, in 2010 it was estimated by a Google engineer [2] that the execution of a single search query uses over 1000 computers. Meanwhile, multi- and many-core platforms are now found with numbers of cores in the thousands. In many domains, as engineers or users interacting with a system, we find ourselves no longer concerned with the specific behaviour of individual components or sub-systems. Instead, we are often more interested in the behaviour of the system as a whole, in the resulting behaviour of many interacting computational entities. This has been recognised in the emerging area of spatial computing [3], in which one considers how to program a large collective system, not at the level of individual entities (as in multi-agent systems), but at the level of the aggregate [4].

Heterogeneity, and other types of diversity, form a major challenge in cloud [5] and edge computing [6], and in sensor networks [7]. Yet heterogeneity also offers opportunities for greater efficiency and task performance, as entities within a system can specialise and be better adapted to their specific local situation, avoiding the constraint of a one-size-fits-all approach. Examples of such a benefit have been shown in a range of areas as diverse as multi-core systems [8], machine learning algorithms [9], circuit design [10] and, again, sensor networks [11]. Recently, we reviewed and analysed types of heterogeneity and other forms of diversity in computing systems [12], using examples of visual sensor networks, a population-based search algorithm, and a cloud load balancer. A more in-depth analysis of diversity in a visual sensor network application showed how a system comprising many self-aware entities may lead to increased heterogeneity, as the different entities learn to be different from each other, in line with their own perceptions of the world [13].

Uncertainty over operation is approaching being an assumption in many computing domains. This implies that the same behaviour may lead to different outcomes at different times, or the environment of a component (including other systems with which it needs to interact) will variedly be present or not. Cloud computing is a case in point of this, where for example in volunteer clouds, physical storage resources may or may not be available to satisfy a request, and even if storage is allocated, it may or may not be reliable [14], [15].

A further consideration that must be dealt with in many modern systems is **ongoing change**. This may be considered at the level of the operating environment (e.g,. workloads or other input may change in their characteristics over time, or in response to external factors, such as the economy, climate or political events). And ongoing change may also be present in the system itself, as available resources may not only be unknown until after deployment, but may continue to become available and unavailable, leading to a system that needs to be continuously formed and reformed on the fly. Major research efforts are already devoted to this issue in computer engineering (e.g., [8], [16]).

Finally, an increasingly prevalent characteristic of 21st century systems is **decentralisation**. This may be expressed in terms of a decentralisation of processing and decision-making, as the feasibility of routing data back to a centralised controller, responsible for making decisions about how the entire system should operate, is being questioned. Due to the vast amount of data involved, camera networks are one area where decentralisation of processing and decision-making is taking hold [17]. Here, a camera at the edge of the network is responsible for processing video data and also potentially for deciding whether what is observed should be communicated to the user or other cameras. A more extreme form of decentralisation exists when different parts of the system are not only operating in a decentralised manner, but may be owned or controlled by different individuals or organisations. For example, in cloud computing, resources required for an application may be owned by individuals with their own differing priorities and goals [14], and this leads to a diversity of purpose. In this case, not only must operational behaviour and data be coordinated without reference to a central node, but diverse goals must also either be reconciled, or else the system made tolerant to this fact of life.

III. A Desire for Self-awareness in Computing

Of course, having to deal with growing complexity in computing is nothing new. Referring to a previous "complexity challenge" [18], the autonomic computing effort sought to use greater levels of autonomy to provide systems with the ability to manage themselves [19]. In what has become known as IBM's "manifesto" for autonomic computing [18], Horn laid out what he saw as fundamental characteristic of autonomic systems: *"to be autonomic, a computing system needs to know itself."* Indeed, the autonomic computing literature has proposed [20]–[22] four supporting properties of autonomic systems, which are required in order to enable the activities above. These are self-awareness, environment-awareness (or self-situation), self-monitoring and self-adjustment.

Perhaps the earliest attempts to explicitly address self-awareness in a computing context began around 2004. A DARPA workshop on "Self-Aware Computer Systems" [23] discussed contributions from a range of researchers whose position papers proposed key challenges in understanding what self-awareness might mean in computing.

One area of self-awareness that has received some focus has been those that form part of meta-cognition [24] (thinking about thinking), long seen as a challenge in progressing towards more advanced artificial intelligence [25]. A system that engages in meta-cognition can be considered to be a special case of one engaged in cognition more generally, since in the meta-cognitive case, the domain of a cognitive system's knowledge is the system itself [26]. A meta-cognitive system can learn and reason about, and therefore act on, its knowledge, beliefs and own reasoning processes. Cox [27] highlights that a meta-cognitive feedback loop is required for self-awareness; he argues that being aware of oneself is not merely about possessing information, but also about using that information in order to modify goals, including concerning what information to gather in future. Schubert [25] and Cox [28] both highlight another benefit that self-awareness can bring to systems, beyond adaptation: self-explanation. Self-aware systems will be able to explain or justify themselves to external entities, such as humans or other systems, based on their self-awareness.

While meta-cognition and meta-self-awareness have been a focus of artificial intelligence researchers, computer engineers have also identified a role for self-awareness in the form of systems that use run-time knowledge about themselves. Agarwal and others [16], [29] propose a shift in how design is carried out, away from the classic case where the behaviour of the system is specified at design time, in favour of designing systems capable of adapting to their context at run time. This has the effect of delaying decisions until run time, particularly concerning how resources should be deployed. Since, in a complex world, it is unclear exactly which resources will be available in a given context, designs are favoured in which systems can discover resources and make decisions about how to effectively and efficiently allocate them during operation.

One benefit of this approach is that it can lead to less programming effort, since if a system can automatically discover how to meet its goals based on what it finds available during operation, then designers are no longer required to solve problems of resource allocation for every envisaged situation themselves. Important properties of self-aware systems, Agarwal argues [16], include introspection, adaptation, self-healing, goal-orientation and the ability to carry out computations to a desired approximation.

In the area of software engineering for service-based systems, Kounev [30] also propose a role for self-awareness. Here the problem tackled is that of how to meet quality of service requirements, despite changes in the operating environment. Here, the focus is on endowing the system with the capability to build models of the system's architecture and its interactions with its environment. These self-models are used to enable run-time reasoning and adaptation. Properties highlighted in Kounev et al's work [31] include self-reflection (awareness of hardware and software infrastructure, execution environment, and operational goals), self-prediction (the ability to predict the effects of environmental changes and of actions) and self-adaptation. A challenge is highlighted around the need for systematic engineering methodologies for self-aware systems.

Unsurprisingly, self-awareness has also received attention in robotics research. Here, much work has focussed on how to replicate forms of self-awareness as seen in humans. For example, the robot *Nico* [32] learns about its body and attempts to mimic an infant. Winfield has argued for a vital role for self-awareness in robots that interact with humans [33]. His thesis is that the possession of internal self-models are essential for safe and ethical robot behaviour. In order to avoid unsafe or unethical outcomes, robots will need to construct and evaluate a model, perhaps a simulation of itself in its current surroundings. Such a model may then be used to moderate the robot's actions.

In collective robotics, self-awareness has also been proposed as a means to enable a swarm of robots to recognise situations during operation that require self-adaptive actions. A particular focus here has been the use of self-awareness to intentionally modify the structure of the swarm [34]. Here it is argued that self-awareness should not only be concerned with what is currently happening, but also what might happen in the future. Such predictions typically focus on how the environment may change over time, but they might also include what the swarm has the potential to become through adaptation, or how individuals in the swarm might affect each other.

Organic computing, an effort to build so-called "life-like" computing systems, has also identified a key role for self-awareness [35] in decentralised systems. A key target of the organic computing initiative is to engineer desirable global behaviour in self-organising systems, through the use of (often global) observation of the system, and resulting control [36]. Thus, both the benefits of self-organisation and decentralisation can be realised, but in a controlled way that can provide assurances.

At the other end of the spectrum, self-awareness has also been recognised as a valuable property in small, resource constrained systems. One example of this is cognitive radio [37], where the drive to improve communication efficiency has led to the development of devices that engage in self-monitoring, and use this to negotiate changes in parameter settings. A further example is cognitive packet networks (CPNs) [38], where there is the challenge to remain highly resilient to network attacks, while also dealing with changing quality of service requirements [39]. Here, a self-awareness loop provides nodes on a network with the ability to monitor the effect of using different routes. Based on a simple learning scheme, routes between a particular source and destination are adapted on an ongoing basis.

Five years ago, we presented a survey [40] of the various research areas that had considered self-awareness as part of technical systems. This survey found that despite the prevalence of the term, what self-awareness meant in a technical context was not well understood. Further, despite the discussion of self-awareness in various areas of computer science and engineering, there was a distinct lack of interaction between the literatures discussing the concept, and assumptions and understandings often varied or contradicted each other.

Nevertheless, our analysis led us to the view that these disparate communities had all arrived, albeit sometimes implicitly present, at something approximating the same high-level hypothesis: **systems that engage in self-awareness can better manage trade-offs between goals at run time, in complex, uncertain and dynamic environments**.

IV. A Common Framework for Computational Self-awareness

In order to develop a theory underpinning self-awareness in computing systems, it is beneficial to first consider self-awareness in humans. In a recent article [41], we introduced the notion of *computational self-awareness*, a interpretation of self-awareness as it pertains to computing systems. We proposed a conceptual framework for computational self-awareness, including a design approach based on a reference architecture, which was inspired by understanding of human self-awareness in psychology.

Our starting point is psychologist Alain Morin's definition of self-awareness: *"the capacity to become the object of one's own attention"* [42]. Morin's definition requires the ability to consider oneself as an object, often called objective or explicit self-awareness. This permits an individual to focus its attention on itself, to consider itself as an entity within the world, and to observe and consider its own behaviour. This often involves an awareness of the individual's public self, that visible to the rest of the world. However, another facet of self-awareness is subjective or implicit. This is concerned not with the self as the object "me", but rather as "I", as the subject of experiences. Here, the individual is aware of its experiences within the world, and that these are its own experiences, subjective and unique. These experiences are private to the individual, and are typically not externally observable. This highlights the first concept in our framework: the distinction between *public* and *private* self-awareness processes, classes which are concerned with knowledge based on phenomena external and internal to the individual respectively. Similarly, systems possessing computational self-awareness should not only be concerned with knowledge of their own internals, but also of their experiences of, impact on, and role within the world.

The second concept in the framework is the existence of different *levels of self-awareness* [43], ranging from basic awareness of environmental stimuli through awareness of interactions and time, up to awareness of one's own thoughts. Advanced organisms also engage in *meta-self-awareness* [42], an awareness that they themselves are self-aware. Self-aware computing systems may similarly vary a great deal in their complexity. Of the various descriptions in psychology, Neisser's [43] levels of self-awareness were chosen as a source of inspiration from which to build concepts of computational self-awareness, since they include the broadest range of forms of self-awareness identified in humans. Accordingly, we developed a set of *levels of computational self-awareness* [44], inspired by Neisser's levels for humans but translated appropriately for describing the capabilities of computer systems. By translating concepts such as this to the computing domain, designers are then able to adopt a common language in considering the various self-awareness capabilities which their systems may or may not possess. While "full-stack" computational self-awareness may often be beneficial, with several processes responsible for one or more levels of self-awareness, there are also cases where a more minimal approach is appropriate.

The third concept in the framework is that self-awareness can be a property of collective systems, even when there is no single component with a global awareness of the whole system [45]. This is a key observation which can contribute to the architecture of self-aware systems: one need not require that a self-aware system possess a global component "responsible" for doing the self-awareness, or for holding the self-knowledge.

This conceptual framework was developed and validated primarily in the context of the EU-funded EPiCS project, the vision of which was laid out in a short article in 2012 [46]. The aim of this project was to take inspiration from self-awareness theories in psychology, to develop new methodologies for the engineering of complex computing systems. A range of different case studies were used in developing the framework and reference architecture, case studies which are concerned with both the platform level (e.g., [47]) and the application level (e.g., [48]). How these relate to the EPiCS vision was laid out in a summary paper [49], and the culmination of the project's results are documented in a 2015 book [50].

V. State of the Art

In the years since our 2011 survey [40], research on self-aware computing has flourished. Notable publications include a 2015 special issue of *Computer* [51], and four books, edited by Pitt [52], Lewis et al. [50], Kounev et al. [53], and Wirsing et al. [54] respectively. New applications of self-awareness ideas are being found in fog and mist computing [55], virtualised data centres [56], heterogeneous multi-core platforms for *on-the-fly* computing [8], interactive music devices [57], sensor networks [48], cloud computing [58] and the Internet of Things [59]. In all of these, self-awareness has been used to help deal with the issue of run-time resource constraints, and it is notable that self-awareness is providing benefits particularly in time- and space-constrained systems.

Further, fundamental aspects of computational self-awareness continue to be explored. For example, Preden et al. [55] highlight the key relationship between self-awareness and attention, which is also prevalent in the psychology literature. This will be an increasingly important avenue to explore, as resource-constrained systems must determine, for themselves, how to direct their limited resources, given the vast set of possible things they could attend to.

Finally, new ways of realising aspects of self-awareness are being proposed. In addition to our own approach to the architecture of self-aware systems [41], efforts exist to collate common features of self-aware systems [60], as well as common learning techniques for realising self-awareness in its various forms [61]. Meanwhile, the issue of collective self-awareness in distributed systems has been explored, using mechanisms based on hierarchies of self-aware components [62], [63]. A Dagstuhl seminar on self-aware computing systems in 2015 identified links between self-aware computing and other ongoing work in self-adaptive systems, run-time modelling, reflective architectures and autonomic computing [64].

VI. Conclusion

In summary, computing systems of various types now face the challenges of performing effectively, efficiently, robustly and flexibly in a complex interconnected world, where they interact with people, each other, the economy and the climate. This leads to uncertainties that are difficult to understand and predict, and where decisions need to be made and revisited on an ongoing basis, based on information only available at run time. *Self-aware computing* refers to systems that are inspired by self-awareness concepts in humans and animals, and is a direct attempt to face this challenge.

Systems with *computational self-awareness* learn and adapt during their lifetime on an ongoing basis, based on their sensed experiences and the internal models that they build. In doing so, they develop an awareness of themselves and their experience of the world they inhabit. Systems engaging in advanced self-awareness are also meta-self-aware, that is they are aware of the way they themselves are aware of these things, and of the way in which they make decisions, based on this self-knowledge. Further, due to the presence of internal self-models, they can engage in self-explanation, a form of reporting in which the reasons behind action (or inaction) are made clear. Increased self-awareness therefore allows systems to express themselves in dynamic ways. Highly adaptive behaviour can help systems to meet their goals, and the goals of those who have a stake in the effects of the behaviour of the system, under unpredictably changing conditions.

Acknowledgment

The author would like to thank the partners in the EU-funded EPiCS project, by and with whom much of the work reviewed in this paper was carried out. Additional thanks go to those who have contributed to many stimulating and challenging discussions on the nature of computational self-awareness, including the participants of the EU-funded Awareness Project, several workshops at the IEEE International Conference on Self-adaptive and Self-organising Systems (SASO), the participants of the Dagstuhl Workshop 15041, *Model-driven Algorithms and Architectures for Self-Aware Computing Systems*, and the participants of the Cyber-Physical Systems Week workshop on Self-aware Cyber-Physical Systems (SelPhyS) in 2016.

References

[1] R. Marler and J. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[2] J. Dean, "Building Software Systems At Google and Lessons Learned," talk at Stanford University, November 10, 2010, retrieved, 1st December 2016. [Online]. Available: https://www.youtube.com/watch?v=modXC5IWTJI

[3] J. Beal, O. Michel, and U. P. Schultz, "Spatial computing: Distributed systems that take advantage of our geometric world," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 6, no. 2, pp. 11:1–11:3, 2011.

[4] J. Beal and M. Viroli, "Aggregate programming: From foundations to applications," in *Formal Methods for the Quantitative Evaluation of Collective Adaptive Systems: 16th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2016, Bertinoro, Italy, June 20-24, 2016, Advanced Lectures*, M. Bernardo, R. De Nicola, and J. Hillston, Eds. Springer, 2016, pp. 233–260.

[5] S. Crago, K. Dunn, P. Eads, L. Hochstein, D. I. Kang, M. Kang, D. Modium, K. Singh, J. Suh, and J. P. Walters, "Heterogeneous cloud computing," in *2011 IEEE International Conference on Cluster Computing*, 2011, pp. 378–385.

[6] D. Schäfer, J. Edinger, S. VanSyckel, J. M. Paluska, and C. Becker, "Tasklets: Overcoming heterogeneity in distributed computing systems," in *Distributed Computing Systems Workshops (ICDCSW), 2016 IEEE 36th International Conference on*. IEEE, 2016, pp. 156–161.

[7] I. Amundson, B. Kusy, P. Volgyesi, X. Koutsoukos, and A. Ledeczi, "Time synchronization in heterogeneous sensor networks," in *Distributed Computing in Sensor Systems: 4th IEEE International Conference, DCOSS 2008 Santorini Island, Greece, June 11-14, 2008 Proceedings*, S. E. Nikoletseas, B. S. Chlebus, D. B. Johnson, and B. Krishnamachari, Eds. Springer, 2008, pp. 17–31.

[8] M. Platzner, "On-the-fly computing: Self-aware heterogeneous multicores," in *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, ser. CODES '16. ACM, 2016, pp. 36:1–36:2.

[9] L. L. Minku and X. Yao, "DDD: A new ensemble approach for dealing with concept drift," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 4, pp. 619–633.

[10] T. Schnier and X. Yao, "Using negative correlation to evolve fault-tolerant circuits," in *In Proceedings of the 5th International Conference on Evolvable Systems: From Biology to Hardware (ICES2003), Lecture Notes in Computer Science*, vol. 2606. Springer, 2003, pp. 35–46.

[11] P. R. Lewis, L. Esterle, A. Chandra, B. Rinner, J. Torresen, and X. Yao, "Static, Dynamic, and Adaptive Heterogeneity in Distributed Smart Camera Networks," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 10, no. 2, pp. 8:1–8:30, Jun. 2015.

[12] P. R. Lewis, H. Goldingay, and V. Nallur, "Its Good to be Different: Diversity, Heterogeneity and Dynamics in Collective Systems," in *Eighth IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2014) Workshops Proceedings*. IEEE Computer Society Press, 2014, pp. 84–89.

[13] P. R. Lewis, L. Esterle, A. Chandra, B. Rinner, and X. Yao, "Learning to be different: Heterogeneity and efficiency in distributed smart camera networks," in *Proceedings of the Seventh IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*. IEEE Press, 2013, pp. 209–218.

[14] A. Elhabbash, R. Bahsoon, P. Tino, and P. R. Lewis, "Self-adaptive volunteered services composition through stimulus- and time-awareness," in *Proceedings of the IEEE International Conference on Web Services (ICWS) 2015*, 2015, pp. 57–64.

[15] R. B. Abdessalam Elhabbash and P. Tino, "Interaction-awareness for self-adaptive volunteer computing," in *Proceedings of the 10th IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2016)*, 2016.

[16] A. Agarwal, J. Miller, J. Eastep, D. Wentziaff, and H. Kasture, "Self-aware computing," MIT, Tech. Rep. AFRL-RI-RS-TR-2009-161, 2009.

[17] B. Rinner and W. Wolf, "Introduction to Distributed Smart Cameras," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1565–1575, 2008.

[18] P. Horn, "Autonomic computing: IBMs perspective on the state of information technology," Armonk, NY, USA. International Business Machines Corporation., 2001.

[19] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[20] R. Sterritt, M. Parashar, H. Tianfield, and R. Unland, "A concise introduction to autonomic computing," *Advanced Engineering Informatics*, vol. 19, no. 3, pp. 181–187, 2005.

[21] M. Parashar and S. Hariri, "Autonomic computing: an overview," in *Proceedings of the 2004 international conference on Unconventional Programming Paradigms*. Springer, 2005, pp. 257–269.

[22] S. Dobson, R. Sterritt, P. Nixon, and M. Hinchey, "Fulfilling the vision of autonomic computing," *Computer*, vol. 43, no. 1, pp. 35 –41, 2010.

[23] E. Amir, M. L. Anderson, and V. K. Chaudhri, "Report on DARPA workshop on self-aware computer systems," UIUC Comp. Sci., Tech. Rep. UIUCDCS-R-2007-2810, 2007.

[24] M. L. Anderson and D. R. Perlis, "Logic, self-awareness and self-improvement: The metacognitive loop and the problem of brittleness." *Journal of Logic and Computation*, vol. 15, no. 1, pp. 21–40, 2005.

[25] L. Schubert, "Some knowledge representation and reasoning requirements for self-awareness," 2005.

[26] P. Maes, "Introspection in knowledge representation," *Advances in Artificial Intelligence II*, pp. 249–262, 1987.

[27] M. Cox, "Metacognition in computation: A selected research review," *Art. Int.*, vol. 169, no. 2, pp. 104–141, 2005.

[28] ——, "Metareasoning, monitoring, and self-explanation." in *Metareasoning: Thinking about thinking*, M. Cox and A. Raja, Eds. Cambridge, MA, USA: MIT Press, 2011, pp. 131–149.

[29] A. Agarwal and B. Harrod, "Organic computing," MIT and DARPA, Tech. Rep. White paper, 2006.

[30] S. Kounev, "Self-Aware Software and Systems Engineering: A Vision and Research Roadmap," in *GI Softwaretechnik-Trends, 31(4), November 2011, ISSN 0720-8928*, Karlsruhe, Germany, 2011. [Online]. Available: http://pi.informatik.uni-siegen.de/stt/31_4/index.html

[31] ——, "Engineering of Self-Aware IT Systems and Services: State-of-the-Art and Research Challenges," in *Proceedings of the 8th European Performance Engineering Workshop (EPEW'11), Borrowdale, The English Lake District, October 12–13*, 2011, (Keynote Talk).

[32] J. W. Hart and B. Scassellati, "Robotic self-modeling," in *The Computer After Me*, J. Pitt, Ed. Imperial College Press / World Scientific Book, 2014, ch. 14.

[33] A. F. T. Winfield, "Robots with internal models: a route to self-aware and hence safer robots," in *The Computer After Me*, J. Pitt, Ed. Imperial College Press / World Scientific Book, 2014, ch. 16.

[34] F. Zambonelli, N. Bicocchi, G. Cabri, L. Leonardi, and M. Puviani, "On self-adaptation, self-expression, and self-awareness in autonomic service component ensembles," in *Proceedings of the Fifth IEEE Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW)*, 2011, pp. 108–113.

[35] D. Kramer, R. Buchty, and W. Karl, "Monitoring and self-awareness for heterogeneous, adaptive computing systems," in *Organic Computing — A Paradigm Shift for Complex Systems*, C. Müller-Schloer, H. Schmeck, and T. Ungerer, Eds. Springer, 2011, pp. 163–177.

[36] C. Müller-Schloer, H. Schmeck, and T. Ungerer, Eds., *Organic computing: a paradigm shift for complex systems*. Springer, 2011.

[37] J. Wang, D. Brady, K. Baclawski, M. Kokar, and L. Lechowicz, "The use of ontologies for the self-awareness of the communication nodes," in *Proceedings of the Software Defined Radio Technical Conference SDR*, vol. 3, 2003.

[38] G. Sakellari, "The cognitive packet network: A survey," *The Computer Journal*, vol. 53, 2010.

[39] E. Gelenbe and G. Loukas, "A self-aware approach to denial of service defence," *Computer Networks*, vol. 51, pp. 1299–1314, 2007.

[40] P. R. Lewis, A. Chandra, S. Parsons, E. Robinson, K. Glette, R. Bahsoon, J. Torresen, and X. Yao, "A survey of self-awareness and its application in computing systems," in *Proceedings of the Fifth IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW)*. IEEE Computer Society Press, 2011, pp. 102–107.

[41] P. R. Lewis, A. Chandra, F. Faniyi, K. Glette, T. Chen, R. Bahsoon, J. Torresen, and X. Yao, "Architectural aspects of self-aware and self-expressive systems: From psychology to engineering," *Computer*, vol. 48, no. 8, 2015.

[42] A. Morin, "Levels of consciousness and self-awareness : A comparison and integration of various neurocognitive views," *Consiousness and Cognition*, vol. 15, no. 2, pp. 358–71, 2006.

[43] U. Neisser, "The roots of self-knowledge: Perceiving self, it, and thou," *Annals of the New York Academy of Sciences*, vol. 818, pp. 19–33, 1997.

[44] F. Faniyi, P. R. Lewis, R. Bahsoon, and X. Yao, "Architecting self-aware software systems," in *Proceedings of the Working IEEE/IFIP Conference on Software Architecture (WICSA) 2014*. IEEE, 2014, pp. 91–94.

[45] M. Mitchell, "Self-awareness and control in decentralized systems," in *Working Papers of the AAAI 2005 Spring Symposium on Metacognition in Computation*. Menlo Park, CA, USA: AAAI Press, 2005, pp. 80–85.

[46] P. R. Lewis, M. Platzner, and X. Yao, "An outlook for self-awareness in computing systems," *Awareness Magazine*, March 2012.

[47] A. Agne, M. Happe, A. Lsch, C. Plessl, and M. Platzner, "Self-awareness as a model for designing and operating heterogeneous multicores," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 7, no. 2, Jun. 2014.

[48] B. Rinner, L. Esterle, J. Simonjan, G. Nebehay, R. Pflugfelder, G. Fernandez Dominguez, and P. R. Lewis, "Self-aware and self-expressive camera networks," *Computer*, vol. 48, no. 7, pp. 21–28, July 2015.

[49] T. Becker, A. Agne, P. R. Lewis, R. Bahsoon, F. Faniyi, L. Esterle, A. Keller, A. Chandra, A. R. Jensenius, and S. C. Stilkerich, "EPiCS: Engineering proprioception in computing systems," in *Proceedings of the 15th IEEE International Conference on Computational Science and Engineering (CSE)*. IEEE Press, 2012, pp. 353–360.

[50] P. R. Lewis, M. Platzner, B. Rinner, J. Torresen, and X. Yao, Eds., *Self-Aware Computing Systems: An Engineering Approach*. Springer, 2016.

[51] J. Torresen, C. Plessl, and X. Yao, "Self-aware and self-expressive systems," *Computer*, vol. 48, no. 7, pp. 18–20.

[52] J. Pitt, Ed., *The Computer After Me: Awareness and Self-awareness in Autonomic Systems*. Imperial College Press, 2014.

[53] S. Kounev, J. O. Kephart, A. Milenkoski, and X. Zhu, Eds., *Self-Aware Computing Systems*. Springer, 2017.

[54] M. Wirsing, M. Hölzl, N. Koch, and P. Mayer, *Software Engineering for Collective Autonomic Systems: The ASCENS Approach*, ser. Lecture Notes in Computer Science. Springer, 2015, vol. 8998.

[55] J. S. Preden, K. Tammeme, A. Jantsch, M. Leier, A. Riid, and E. Calis, "The benefits of self-awareness and attention in fog and mist computing," *Computer*, vol. 48, no. 7, pp. 37–45, 2015.

[56] V. van Beek, J. Donkervliet, T. Hegeman, S. Hugtenburg, and A. Iosup, "Self-expressive management of business-critical workloads in virtualized datacenters," *Computer*, vol. 48, no. 7, pp. 46–54, 2015.

[57] K. Nymoen, A. Chandra, and J. Torresen, "Self-awareness in active music systems," in *Self-aware Computing Systems: An Engineering Approach*, P. R. Lewis, M. Platzner, B. Rinner, J. Tørresen, and X. Yao, Eds. Springer, 2016, pp. 279–296.

[58] T. Chen and R. Bahsoon, "Toward a smarter cloud: Self-aware autoscaling of cloud configurations and resources," *Computer*, vol. 48, no. 9, pp. 93–96, 2015.

[59] M. Möstl, J. Schlatow, R. Ernst, H. Hoffmann, A. Merchant, and A. Shraer, "Self-aware systems for the internet-of-things," in *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, ser. CODES '16. ACM, 2016, pp. 21:1–21:9.

[60] L. L. Minku, L. Esterle, G. Nebehay, and R. Chen, "Knowledge representation and modelling: Structures and trade-offs," in *Self-aware Computing Systems: An Engineering Approach*, P. R. Lewis, M. Platzner, B. Rinner, J. Tørresen, and X. Yao, Eds. Springer, 2016, pp. 79–111.

[61] S. Wang, G. Nebehay, L. Esterle, K. Nymoen, and L. L. Minku, "Common techniques for self-awareness and self-expression," in *Self-aware Computing Systems: An Engineering Approach*, P. R. Lewis, M. Platzner, B. Rinner, J. Tørresen, and X. Yao, Eds. Springer, 2016, pp. 113–142.

[62] M. Amoretti and S. Cagnoni, "Toward collective self-awareness and self-expression in distributed systems," *Computer*, vol. 48, no. 7, pp. 29–36, 2015.

[63] L. Guang, J. Plosila, and H. Tenhunen, "From self-aware building blocks to self-organizing systems with hierarchical agent-based adaptation," in *Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis*, ser. CODES '14. ACM, 2014, pp. 23:1–23:3.

[64] S. Kounev, X. Zhu, J. O. Kephart, and M. Kwiatkowska, "Model-driven Algorithms and Architectures for Self-Aware Computing Systems (Dagstuhl Seminar 15041)," *Dagstuhl Reports*, vol. 5, no. 1, pp. 164–196, 2015. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2015/5038