Self-Improving Autonomic Systems for Antifragile Cyber Defence: Challenges and Opportunities

Mohan Baruwal Chhetri*, Anton V. Uzunov‡, Quoc Bao Vo*, Surya Nepal§, Ryszard Kowalczyk*†
*Faculty of Science, Engineering and Technology, Swinburne University of Technology, Melbourne, VIC 3122, Australia
†Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland
‡Defence Science and Technology Group, Edinburgh, SA 5111, Australia
§Data61, Marsfield, NSW 2122, Australia

***Abstract*—Antifragile systems enhance their capabilities and become stronger when exposed to adverse conditions, stresses or attacks, making *antifragility* a desirable property for cyber defence systems that operate in contested military environments. *Self-improvement* in autonomic systems refers to the improvement of their self-* capabilities, so that they are able to (a) better handle previously known (anticipated) situations, and (b) deal with previously unknown (unanticipated) situations. In this position paper, we present a vision of using self-improvement through learning to achieve antifragility in autonomic cyber defence systems. We first enumerate some of the major challenges associated with realizing *distributed self-improvement*. We then propose a reference model for middleware frameworks for self-improving autonomic systems and a set of desirable features of such frameworks.**

***Index Terms*—antifragility, autonomic systems, cyber defence, self-improvement, learning**

I. Introduction

Cyber security is a growing concern across Defence, Government and Industry, as cyber attacks become increasingly more powerful and sophisticated. It is not only important for cyber defence systems to *deploy the right capability* to protect cyber assets, but also to ensure that this capability *continues to operate* even under stress, especially in contested, military environments, and does so in a *self-governing* fashion.

Based on their ability to handle stress, systems can be loosely classified on a *fragility spectrum* as follows [1], [2]:

- A **fragile** system is *unable to withstand* stress and degrades immediately, *i.e.,* it goes from an *intended state*, where it is functioning correctly, to an *unintended state*, where it no longer functions correctly.
- A **robust** system is able to *withstand* stress, and remain in an intended state, up to a certain stress level, beyond which it becomes fragile, *i.e.,* it is fragile beyond some breaking point.
- A **resilient** system has the ability to *recover* from stress and *reconstitute*, *i.e.,* return from an unintended state to an intended state.
- An **antifragile** system is not only resilient to stress, but is also able to *evolve* and *improve* itself to become more robust as a result of stress.

As can be surmised from the preceding discussion, *antifragility* is a highly desirable – indeed, sometimes critical – property of cyber defence systems operating in contested environments.

Autonomic or self-adaptive[1] computing [3] is an attractive paradigm for achieving antifragility. Conceptually, an autonomic system comprises two key elements – the *domain functionality* and the *management functionality* (also called *self-management*, where the "self" notionally refers to the whole system) [4], [5]. Domain functionality deals with domain concerns, *i.e.,* achieving the goals for which the system was built, and comprises the application logic that helps realize it. Management functionality deals with adaptation concerns, *i.e.,* how the system achieves its goals under changing conditions, and comprises the adaptation logic. Most approaches for autonomic systems, as reported in the literature [5], [6], focus on *management of domain functionality* – by incorporating the key self-* functionalities including *self-configuration*, *self-healing*, *self-protection* and *self-optimization* (a.k.a self-CHOP) – using the de-facto MAPE-K reference model [7]. However, they tend to ignore *management of the management functionality itself*, which is equally important. This approach of focusing solely on domain management/adaptation leads to two problems:

- Firstly, as discussed in [8], one cannot rely on managing elements to be trustworthy and behave correctly at all times, as attacks on cyber systems can lead to compromise of both domain and management functionality. Any self-management mechanism has to consider the possibility of *compromise and/or failure of the management components*.
- Secondly, as identified in [5], most existing work on autonomic systems focuses on adaptation to meet *pre-specified goals* defined at design time, using *pre-specified plans*, also defined at design time. There is limited support for dealing with *changing goals* at runtime, including the addition and/or removal of goals, and the synthesis of new plans that comply with them [9], [10].

Moreover, most relevant research in the area takes a closed-world *reductionist approach* to the design of autonomic systems and assumes that the system operates in a known world with anticipated circumstances [11]. This assumption does not necessarily hold in contested, military environments, where

[1]We use the terms autonomic, self-governing, self-managing and self-adaptive interchangeably.

2474-0756/19/$31.00 ©2019 IEEE
DOI 10.1109/ICAC.2019.00013

cyber defence systems have to operate under uncertainty, *i.e.,* they have to deal with and adapt to unanticipated disruptions and/or emergent goals [12].

Dealing with the problems outlined above requires an additional, "special" self-* property – *self-improvement* [13] – also referred to as *meta-adaptation* [14], *self-self behaviours* [15] or *meta-management* [16]. Self-improvement in autonomic systems relates to management of the self-* capabilities and involves "*adjustment of the adaptation logic*" [13] so that a system is able to (a) perform better in known (anticipated) situations, and (b) handle previously unknown (unanticipated) situations. Simply put, it refers to the *improvement of self-* capabilities* and is strongly linked to the notion of (continuous) *learning*, *i.e.,* the process of explicit or implicit knowledge[2] acquisition, which includes *self-knowledge* and not just knowledge of the external world.

If the desirable property of resilience can be achieved through self-management or self-adaptivity, as argued elsewhere (e.g. [17]–[20]), then one can state that antifragility can be achieved through self-management and self-improvement – or simply through self-improvement, since self-improvement in effect subsumes self-management, as the former implies the latter. We will thus refer to systems that are self-improving and self-managing as "self-improving autonomic" systems, or just "self-improving systems" for short.

In this position paper, we present a *vision* for realizing self-improvement to achieve antifragility in cyber defence systems. More specifically, we propose a reference model (**first** contribution) and associated "solution architecture" consisting of a set of high-level features that we believe are particularly suitable (**second** contribution) for *middleware frameworks* for self-improving systems (*i.e.,* frameworks that are self-improving and self-managing and that transitively endow target systems, such as cyber defence systems, with self-improvement and self-management). These features are based on an enumeration of the major challenges associated with realizing distributed self-management and self-improvement (**third** contribution) (the former being considered in our prior work [8]), and building on our previous experiences with the AI-/multi-agent-based approach AWaRE [8], [20] and recent literature on the subject. The individual features not only capture plausible ways to address the aforementioned problems associated with realizing distributed self-improvement, but also help to define a set of research directions for future work in self-improvement (for the purpose of achieving antifragility) for the wider scientific community.

The rest of this paper is organized as follows. In Section II, we advance a cross-discipline, knowledge-centric view of antifragility (which can be considered our **fourth** contribution) and subsequently enumerate some of the challenges associated with realizing distributed self-improvement for antifragility. In Section III, we propose a reference model for middleware frameworks for self-improving autonomic systems, and a corresponding set of key desirable features for such frameworks. Section IV concludes the paper.

[2]We consider knowledge in a general sense so that it includes both factual knowledge as well as practical skills or expertise.

II. Background

A. Antifragility and Learning

Antifragility is a property that enhances the capabilities of a system in response to external stressors including failures, attacks and changing dynamics [1]. An antifragile system is not only able to recover from attacks and failures through the standard self-* capabilities, *i.e.,* the self-CHOP, but is also able to modify, through *learning*, the approach with which they are effected in future scenarios, through adaptation of self-CHOP – something especially useful when applied to configuration-/performance-related functions.

Taking a cross-discipline, knowledge-centric view of uncertainty [21], [22], learning occurs as follows:

- *Known knowns*: this represents the knowledge possessed by the system, *i.e.,* known and independently verifiable facts about itself and its environment that help with decision-making. Systems equipped with known knowns are able to detect, interpret and act upon observations in the environments. In this quadrant, learning involves the *improvement of existing knowledge*, including detection and removal of incorrect or obsolete knowledge, to ensure better decision-making.
- *Known unknowns*: this refers to gaps either in the factual knowledge or expertise that can pose a threat to the system due to ambiguity around its own behavior and state. For instance, a system may have a lot of observed data (about itself and its environment) but may be unable to interpret it. In this quadrant, there is scope for *learning* (from the observations) so that it is not only possible to detect system behavior but also explain it.
- *Unknown knowns:* this refers to the untapped (or implicit) knowledge that, if unlocked, can improve the quality of decision-making. A system can move from *unknown knowns* to *known knowns* through *knowledge sharing*, which can happen between system components, across independent systems, and between the system and a human oracle.
- *Unknown unknowns:* this refers to *latent uncertainty*, *i.e.,* uncertainty that the system has no knowledge of. The question that then arises is how can the system learn in this quadrant? One possible way is through *exploration* by considering variations to known events (that have already occurred), observing the outcomes of those unseen variations (both positive and negative), and learning from them.

B. Key Challenges for Realizing Self-Improvement

Achieving antifragility through self-improvement is challenging in general, but particularly so for *distributed cyber defence systems* that operate in contested environments. Below, we present some of the key challenges associated with realizing distributed self-improvement. They complement the challenges for distributed self-management presented in our previous work in [8].

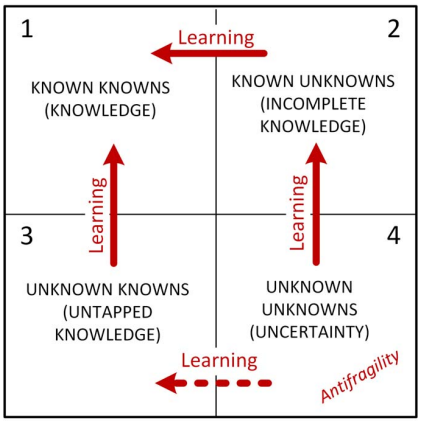

Fig. 1. Four-quadrants knowledge model (adapted from [21]).

1) Domain Modeling: A self-improving system has to learn on an on-going basis about itself and its environment, and subsequently improve its functionality w.r.t. some higher-level goals, which may also be subject to change [23], [24]. A key challenge for realizing self-improving systems is related to *defining appropriate models* that can represent a wide range of system (and environment) properties. The more precise these models are, the more effective they should be in supporting run-time analyses and decision processes.

2) Models/Mechanisms for Self-Assessment: Self-improving systems require appropriate models and mechanisms to assess how well they are doing w.r.t. *performance standards* defined w.r.t. both the *domain* and *self-improvement goals*. A number of challenges arise in the context of pre-defined performance standards, including: (a) design of an appropriate *learning component* that is able to assess the existing adaptation actions, and make suitable changes to improve them, (b) development of appropriate mechanisms to support exploration and exploitation of new/prior knowledge with the objective of augmenting it, and (c) assessing the appropriateness of the *decision making*.

3) Handling Uncertainty: A cyber defence system typically operates in an adversarial environment characterized by inherent uncertainty, and hence may have to deal with both *known* and *unknown unknowns* (cf. Section II-A). The former 'unknown' refers to situations where the system is aware of key missing knowledge, while the latter 'unknown' refers to situations that the system has no knowledge of or is unable to comprehend. The key challenge is to develop *an abstraction of uncertainty and unknowns* – both external and internal – where external uncertainty arises from the environment or domain in which the system is deployed, and internal uncertainty arises from within the self and is related to the adaptation logic and decisions [25]. A related challenge is to facilitate different mechanisms to manage these uncertanties.

4) Distributed/Decentralized Decision-Making: It is well-known that a centralized approach to realizing self-* capabilities is vulnerable to a single-point of failure/attack [5], which introduces fragility. There is thus a need for distributed/decentralized approaches for both realizing and managing self-* capabilities. This implies, however, that all the challenges of distributed/decentralized decision-making, including those related to decentralized control and coordination, pertain to both self-management and self-improvement. This includes the identification and development of appropriate system architectures and problem structures that support a secure, robust (self-)managing system that is both effective and efficient in achieving its goals and also satisfying other system constraints (*e.g.,* spatial, temporal, organizational, etc.). It also includes appropriate coordination mechanisms and interaction protocols for decentralized control and decision-making in the presence of incomplete knowledge and decision conflict (*e.g.,* due to simultaneous adaptations arising from self-management and self-improvement functions).

5) Knowledge Management: Learning, in general, is the process of explicit or implicit knowledge acquisition[3] and there are several challenges associated with managing the acquired knowledge. The first challenge is that of *concept drift* [26], *i.e.,* ensuring that knowledge does not become outdated and irrelevant. A related challenge is that of ensuring the *correctness* and *applicability* of the learned knowledge [27]. Incorrect knowledge can result in undesirable consequences and have a detrimental effect on new learning. Another challenge related to learning is that of *self-motivated learning* – for instance, a self-improving system should be capable of collecting its own training data and learning from it. This is something that current (supervised) learning algorithms, which use the closed world assumption, cannot do [28].

III. Vision for Self-Improving Autonomic Systems

On the basis of analyzing the challenges associated with realizing distributed self-improvement (see Section II and our previous work in [8], [20]), we propose below a reference model for middleware frameworks for self-improving autonomic systems, based on *multi-agent organizations* [29], [30] and the work of [15], [16], [31]; and a corresponding set of "desirable features" for such frameworks, building on a variety of ideas in autonomic/self-adaptive computing and AI. We explain the constituents of the model in detail in what follows, as part of our presentation of the aforementioned "desirable features" and the overview sub-section below.

A. Reference Model

Our reference model, whose structure is shown in Figure 2, is stratified into six layers according to function and abstraction. The first layer concerns system/infrastructure (domain) functions. The second layer contains middleware elements (in our case, agent-services, explained further below) that perform domain virtualization and are also capable of intelligent monitoring and effecting. The third layer contains *dual* middleware elements for autonomic sensing, such that they can monitor themselves and their *dual counterparts* in the layer below.

[3]Knowledge acquisition can happen through experience or through representation-based learning.

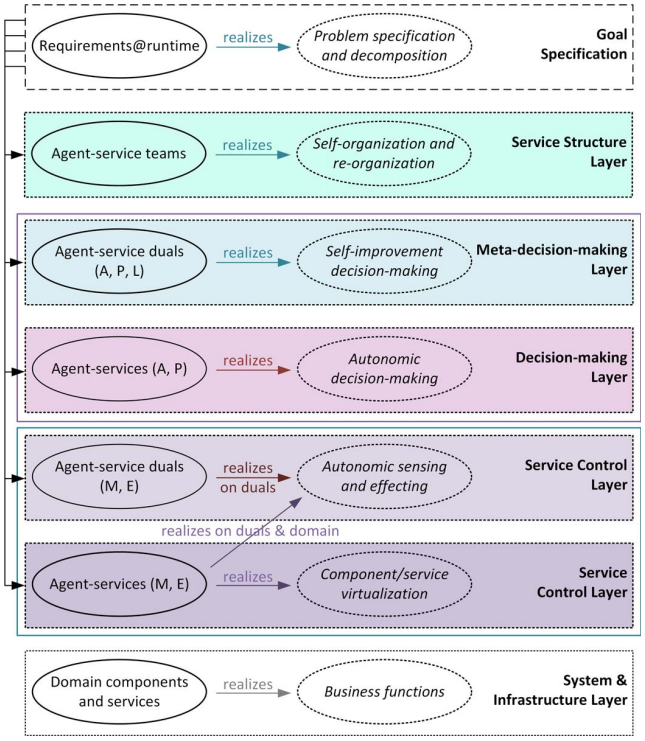

Fig. 2. Reference model for self-improving autonomic systems; overlaid ellipses indicate possible functions (dotted) and desirable features (solid)

The fourth layer contains decision-making elements realizing analysis and planning functions and, like the previous two layers, has a dual layer above (fifth layer) that performs meta-decision-making with respect to the autonomic functions. Just as for the service abstraction and control layers, the duality of the two decision-making layers implies that decision-making pertains to the self as well as underlying elements/services, thus avoiding infinitely stacked MAPE-K loops (cf. [16]). Finally, the sixth layer contains aggregates or organizations (in our case *teams* – see further below) of underlying middleware elements – aggregates, which are themselves self-contained middleware elements – supporting system self-organization and re-organization. Goals/objectives are specified across all layers except the bottom one, and used by the individual middleware elements and aggregates of these. In essence, the structure of our model implies (as indicated previously) a *transitive* approach to self-management and self-improvement, which occurs via virtualization and duality: conceptually, any middleware conformant to our model only ever manages, evolves and improves "itself" to achieve antifragility of both itself and some underlying system/infrastructure.

B. Feature Space

1) Agent-Services: Both multi-agent systems (MAS) and service-oriented architectures (SOA) have been used in the past for realizing complex, distributed and autonomic systems [32]. Of these two architectural styles, MAS is arguably the more natural and intuitive, especially in light of the vast literature on MAS self-organization, coordination, optimization, etc. [29], [30], [33]. However, there are several limitations with using MAS in operative settings, including (a) the programming abstractions for MAS are very different from other well-established programming paradigms, making the MAS paradigm a less attractive option for implementing practical systems [34], and (b) there is limited support for building MASs that are modular, maintainable, reusable and scalable [35].

Recently, several researchers have sought to address these issues by *combining* the MAS and SOA styles [36], [37], where services can enrich agents with additional modularity, maintainabability, reusability and scalability properties. We refer to a combination of agents and services as "*agent-services*", which retain all the desirable characteristics of agents, including goal-based planning, approaches and algorithms for flexible organizations (*e.g.,* [38], [39]), and various coordination and cooperation mechanisms [40], and, at the same time, foster reusability, modularity, orchestratability, service-based interactions, alignment and use of popular industry standards, and the widely prevalent software engineering approach of SOA [34]. We envision agent-services with various MAPE (+ Learning) capabilities realizing the functions spanning our reference model's second to fifth layers.

2) Agent-Service Teams: There are many ways to organize agents in traditional MAS – coalitions, federations, holarchies, matrices and others [39] – each with different strengths and weaknesses. One such type of organization that is particularly appropriate in the context of distributed self-management and self-improvement is *agent-service teams*. An agent-service team comprises multiple "cooperative" agent-services that work together to achieve a common goal; thus teams, not individual agent-services, carry the "objectives" of the system. We propose the use of agent-service teams that, contrary to traditional MAS ideas, are composed of both goal-driven agent-services and reactive services, which do not partake in agent-service communication protocols, but are still considered part of the team.

Thus, teams, as we envision them, could be composed of agent-services and domain services across abstraction levels of the reference model in Figure 2, considered as one entity, and could also re-form as required (in this sense, agent-service teams are similar to the idea of a "super-agent" from [41]). A key benefit of agent-service teams is that they enable collective learning and distributed problem solving. Additionally, self-awareness could be implemented for individual agent-based services, teams, and the whole system, leading to better/faster decision-making compared to building up self-awareness in a single (shared) model at run-time.

3) Flexible Agent-Service Architecture: In order to effectively participate in agent-service teams, the agent part of agent-services should possess flexibility across several key features to enable structural, behavioral and role adaptation. Some of these features are enumerated below:

- *Adjustable autonomy:* The system should support multiple levels of autonomy [42], [43] so that the proactivity of

agent-services can be increased/decreased to make them more "agent-like" or more "service-like". Additionally, incorporating adjustable autonomy into the agent-services allows, for instance, for human oversight, to augment the knowledge-base of agent-services performing specific self-* management functions.

- *Adjustable role assignment:* Agents should take on different roles and dynamically adjust them [29], [44] depending on the goal that needs to be achieved. Depending on the role they are playing, they should use different interaction/coordination patterns. Flexible redundancy should also be possible via multiple agent-services with a given role/purpose/sub-goal. Agent-services should be able to spawn, clone or merge themselves as required [33], [45].
- *Dynamic behavior selection:* Agent-services should dynamically adjust their behavior depending upon the role they are playing and the current state of the system. The *Strategy* pattern is a popular approach to realize dynamic behaviors in (multi-agent-based) autonomic systems [46], [47].

4) Learning in Agent-Services: An agent-service can use three well-known learning paradigms from the field of AI/ML to augment its knowledge: *supervised, unsupervised* and *reinforcement learning (RL)* [48], [49]. It can use supervised learning to build models from observations based on explicit feedback (labeled input-output pairs) and unsupervised learning to learn patterns (without any explicit feedback) from a set of observations. Both forms of learning can be applied in the second quadrant of the four-quadrants knowledge model from Section II-A, wherein models are learned *incrementally* from the observed data. Similarly, an agent-service could use RL to learn appropriate actions through trial and error interactions with its (dynamic) environment. With RL, the agent-service has to make appropriate *exploration/exploitation* trade-offs when collecting optimal training data from its environment. Such learning can happen either in the second or the fourth quadrant. Additionally, agent-service teams can use collaborative reinforcement learning (CRL) [50], [51] to learn in a more robust and scalable manner.

5) Requirements@Runtime: Models play a central role in the realization of autonomic systems, with many self-adaptation mechanisms leveraging the *models@runtime* paradigm [52]. If systems are designed merely to recover from known adverse conditions, *i.e.,* meet "known requirements", then the system goals tend to be static/fixed at runtime. However, autonomic systems designed to evolve and improve not only have to support system (re-)configuration (to meet pre-defined goals), but also modification of existing goals, requirements and constraints, or completely new requirements altogether [9], [10]. A *requirements@runtime* model can offer a vehicle for this, complementing the use of *reflection models* for self-representation. Using constraint models is a particularly powerful approach for this purpose, as it enables the specification of both *adaptation models* [53] and requirements/goals that can be adjusted at runtime, thereby supporting (top-down) re-organization and (bottom-up) self-organization of agent-service teams.

IV. Conclusion

In this paper, we presented our vision for achieving *antifragility* in cyber defence systems via self-improvement. In particular, we argued that self-improvement is best realized by AI-/multi-agent-based middleware frameworks that are competent at two critical aspects: (i) achieving their mission objectives, namely, realizing self-* functions in a target system, and (ii) realizing self-* functions with respect to themselves, *i.e.,* being self-managing and self-improving within the environment in which they are embedded. Based on this argument and a consideration of associated challenges, we proposed a reference model and briefly outlined a set of high-level "desirable features" that we believe are particularly suitable for (reference-model-conformant) middleware frameworks for self-improving systems. These features represent a challenge and, we hope, stimulant for the research community to develop concrete feature realizations, and thus whole families of middleware frameworks, that can fulfil the presented vision. We ourselves are currently working towards this by porting our AWaRE framework to Jadex [34] and suitably extending it in accord with the presented desirable features.

Acknowledgments

This research is supported by a Collaborative Research Project on *"Autonomic Computing for Resilient Cyber Operations in Contested Environments"* between the Defence Science and Technology (DST) Group, Data61 and Swinburne University of Technology, funded by the Department of Defence Next Generation Technology initiative. The authors would like to thank Michael Docking (DST Group) and Seyit Camtepe (CSIRO Data61) for their feedback on this paper.

References

[1] N. N. Taleb, *Antifragile: Things that gain from disorder*. Random House Incorporated, 2012, vol. 3.

[2] D. Bodeau and R. Graubart, "Cyber resiliency design principles: selective use throughout the lifecycle and in conjunction with related disciplines," *McClean (VA): The MITRE Corporation*, pp. 2017–0103, 2017.

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

[4] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 4, no. 2, p. 14, 2009.

[5] D. Weyns, "Software engineering of self-adaptive systems: an organised tour and future challenges," *Chapter in Handbook of Software Engineering*, 2017.

[6] N. Villegas, G. Tamura, and H. Müller, "Architecting software systems for runtime self-adaptation: Concepts, models, and challenges," in *Managing Trade-offs in Adaptable Software Architectures*. Elsevier, 2017, pp. 17–43.

[7] IBM Corporation, "An architectural blueprint for autonomic computing," *IBM White Paper*, vol. 31, 2006.

[8] M. Baruwal Chhetri, A. V. Uzunov, Q. B. Vo *et al.*, "AWaRE – Towards Distributed Self-Management for Resilient Cyber Systems," in *23rd International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2018, pp. 185–188.

[9] P. Sawyer, N. Bencomo, J. Whittle, E. Letier, and A. Finkelstein, "Requirements-aware systems: A research agenda for re for self-adaptive systems," in *Requirements Engineering Conference (RE), 2010 18th IEEE International*. IEEE, 2010, pp. 95–103.

[10] N. Bencomo, J. Whittle, P. Sawyer, A. Finkelstein, and E. Letier, "Requirements reflection: requirements as runtime entities," in *Software Engineering, 2010 ACM/IEEE 32nd International Conference on*, vol. 2. IEEE, 2010, pp. 199–202.

[11] K. H. Jones, "Engineering antifragile systems: A change in design philosophy," *Procedia computer science*, vol. 32, pp. 870–875, 2014.

[12] C. J. Marshalla, B. Roberts, and M. Grenn, "Adaptive and automated reasoning for autonomous system resilience in uncertain worlds," in *Disciplinary Convergence in Systems Engineering Research*. Springer, 2018, pp. 799–812.

[13] C. Krupitzer, F. M. Roth, M. Pfannemüller, and C. Becker, "Comparison of approaches for self-improvement in self-adaptive systems," in *2016 IEEE International Conference on Autonomic Computing (ICAC)*. IEEE, 2016, pp. 308–314.

[14] N. Gui and V. De Florio, "Towards meta-adaptation support with reusable and composable adaptation components," in *Self-Adaptive and Self-Organizing Systems (SASO), 2012 IEEE Sixth International Conference on*. IEEE, 2012, pp. 49–58.

[15] S. Bouchenak, F. Boyer, B. Claudel, N. De Palma, O. Gruber, and S. Sicard, "From Autonomic to Self-Self Behaviors: The JADE Experience," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 6, no. 4, p. 28, 2011.

[16] C. M. Kennedy, "Distributed meta-management for self-protection and self-explanation," *School of Computer Science Research Reports-University of Birmingham CSR*, vol. 3, 2008.

[17] P. Pal, R. Schantz, A. Paulos, and et al., "A3: An environment for self-adaptive diagnosis and immunization of novel attacks," in *Procs. 6th IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW)*. IEEE, 2012, pp. 15–22.

[18] B. Benyo, P. Pal, R. Schantz, and et al., "Automated self-adaptation for cyber-defense – pushing adaptive perimeter protection inward," in *Procs. 7th IEEE International Conference on Self-Adaptation and Self-Organizing Systems Workshops (SASOW)*. IEEE, 2013, pp. 47–52.

[19] M. Docking, A. V. Uzunov, C. Fiddyment, R. Brain, S. Hewett, and L. Blucher, "UNISON: Towards a middleware architecture for autonomous cyber defence," in *Procs. 24th Australasian Software Engineering Conference (ASWEC)*. IEEE, 2015, pp. 203–212.

[20] M. Baruwal Chhetri, H. Luong, A. V. Uzunov, Q. B. Vo, R. Kowalczyk, S. Nepal, and I. Rajapakse, "ADSL: An embedded domain-specific language for constraint-based distributed self-management," in *2018 25th Australasian Software Engineering Conference (ASWEC)*. IEEE, 2018, pp. 101–110.

[21] D. Cleden, *Managing project uncertainty*. Routledge, 2017.

[22] J. Casti and L. IIASA, "Four faces of tomorrow," *OECD International Futures Project on Future Global Shock*, 2011.

[23] B. G. Buchanan, T. M. Mitchell, R. G. Smith, and C. R. Johnson Jr, "Models of learning systems." Stanford Univ Calif Dept of Computer Science, Tech. Rep., 1979.

[24] S. Kounev, P. Lewis, K. L. Bellman *et al.*, "The notion of self-aware computing," in *Self-Aware Computing Systems*. Springer, 2017, pp. 3–16.

[25] S. Mahdavi-Hezavehi, P. Avgeriou, and D. Weyns, "A classification framework of uncertainty in architecture-based self-adaptive systems with multiple quality requirements," in *Managing Trade-Offs in Adaptable Software Architectures*. Elsevier, 2017, pp. 45–77.

[26] N. Kühl, M. Goutier, R. Hirt, and G. Satzger, "Machine learning in artificial intelligence: Towards a common understanding," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019.

[27] Z. Chen and B. Liu, "Lifelong machine learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 10, no. 3, pp. 1–145, 2016.

[28] G. Fei, S. Wang, and B. Liu, "Learning cumulatively to become more knowledgeable," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 1565–1574.

[29] M. Hoogendoorn and J. Treur, "An adaptive multi-agent organization model based on dynamic role allocation," *International journal of knowledge-based and intelligent engineering systems*, vol. 13, no. 3-4, pp. 119–139, 2009.

[30] G. Picard, J. F. Hübner, O. Boissier, and M.-P. Gleizes, "Reorganisation and self-organisation in multi-agent systems," in *1st International Workshop on Organizational Modeling, ORGMOD*, 2009, pp. 66–80.

[31] X. Mao, M. Dong, and H. Zhu, "Towards multiple-layer self-adaptations of multi-agent organizations using reinforcement learning," in *Novel Design and Applications of Robotics Technologies*. IGI Global, 2019, pp. 66–95.

[32] F. M. Brazier, J. O. Kephart, H. Van Dyke Parunak, and M. N. Huhns, "Agents and service-oriented computing for autonomic computing: A research agenda," *IEEE Internet Computing*, vol. 13, no. 3, p. 82, 2009.

[33] S. Kamboj, "Analyzing the tradeoffs between breakup and cloning in the context of organizational self-design," in *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*. International Foundation for Autonomous Agents and Multiagent Systems, 2009, pp. 829–836.

[34] A. Pokahr, L. Braubach, and K. Jander, "The Jadex project: Programming model," in *Multiagent Systems and Applications*. Springer, 2013, pp. 21–53.

[35] I. Nunes, D. Cowan, E. Cirilo, and C. J. De Lucena, "A case for new directions in agent-oriented software engineering," in *International Workshop on Agent-Oriented Software Engineering*. Springer, 2010, pp. 37–61.

[36] R. L. Hartung, "Service oriented architecture and agents: Parallels and opportunities," in *Agent and Multi-agent Technology for Internet and Enterprise Systems*. Springer, 2010, pp. 25–48.

[37] D. I. Tapia, J. Bajo, and J. M. Corchado, "Distributing functionalities in a SOA-based multi-agent architecture," in *7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2009)*. Springer, 2009, pp. 20–29.

[38] M. Hannebauer, *Autonomous dynamic reconfiguration in multi-agent systems: improving the quality and efficiency of collaborative problem solving*. Springer-Verlag, 2002.

[39] B. Horling and V. Lesser, "A survey of multi-agent organizational paradigms," *The Knowledge engineering review*, vol. 19, no. 4, pp. 281–316, 2004.

[40] W. Ren, R. W. Beard, and E. M. Atkins, "A survey of consensus problems in multi-agent coordination," in *American Control Conference, 2005. Proceedings of the 2005*. IEEE, 2005, pp. 1859–1864.

[41] K. Stathis, "Autonomic computing with self-governed super-agents," in *Procs. 4th IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshop (SASOW)*. IEEE, 2010, pp. 76–79.

[42] K. Barber and C. Martin, "Agent autonomy: Specification, measurement, and dynamic adjustment," in *Proceedings of the autonomy control software workshop at autonomous agents*, 1999, pp. 8–15.

[43] S. A. Mostafa, M. S. Ahmad, M. Annamalai, A. Ahmad, and S. S. Gunasekaran, "A conceptual model of layered adjustable autonomy," in *Advances in Information Systems and Technologies*. Springer, 2013, pp. 619–630.

[44] K. Sycara, M. Paolucci, M. Van Velsen, and J. Giampapa, "The RETSINA MAS infrastructure," *Autonomous agents and multi-agent systems*, vol. 7, no. 1-2, pp. 29–48, 2003.

[45] D. Ye, M. Zhang, and D. Sutanto, "Cloning, resource exchange, and relation adaptation: An integrative self-organisation mechanism in a distributed agent network," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 4, pp. 887–897, 2014.

[46] E. A. Kendall, M. T. Malkoun, and C. H. Jiang, "Multiagent system design based on object-oriented patterns," *JOOP*, vol. 10, no. 3, pp. 41–47, 1997.

[47] L. Sabatucci, M. Cossentino, and A. Susi, "A goal-oriented approach for representing and using design patterns," *Journal of Systems and Software*, vol. 110, pp. 136–154, 2015.

[48] T. Mitchell, W. Cohen, E. Hruschka *et al.*, "Never-ending learning," *Communications of the ACM*, vol. 61, no. 5, pp. 103–115, 2018.

[49] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[50] J. Dowling, R. Cunningham, E. Curran, and V. Cahill, "Collaborative reinforcement learning of autonomic behaviour," in *Proceedings. 15th International Workshop on Database and Expert Systems Applications, 2004.* IEEE, 2004, pp. 700–704.

[51] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 38, no. 2, 2008.

[52] G. Blair, N. Bencomo, and R. B. France, "Models@run.time," *Computer*, vol. 42, no. 10, 2009.

[53] T. Vogel, A. Seibel, and H. Giese, "The role of models and megamodels at runtime," in *Models in Software Engineering*, J. Dingel and A. Solberg, Eds. Springer Berlin Heidelberg, 2011, pp. 224–238.