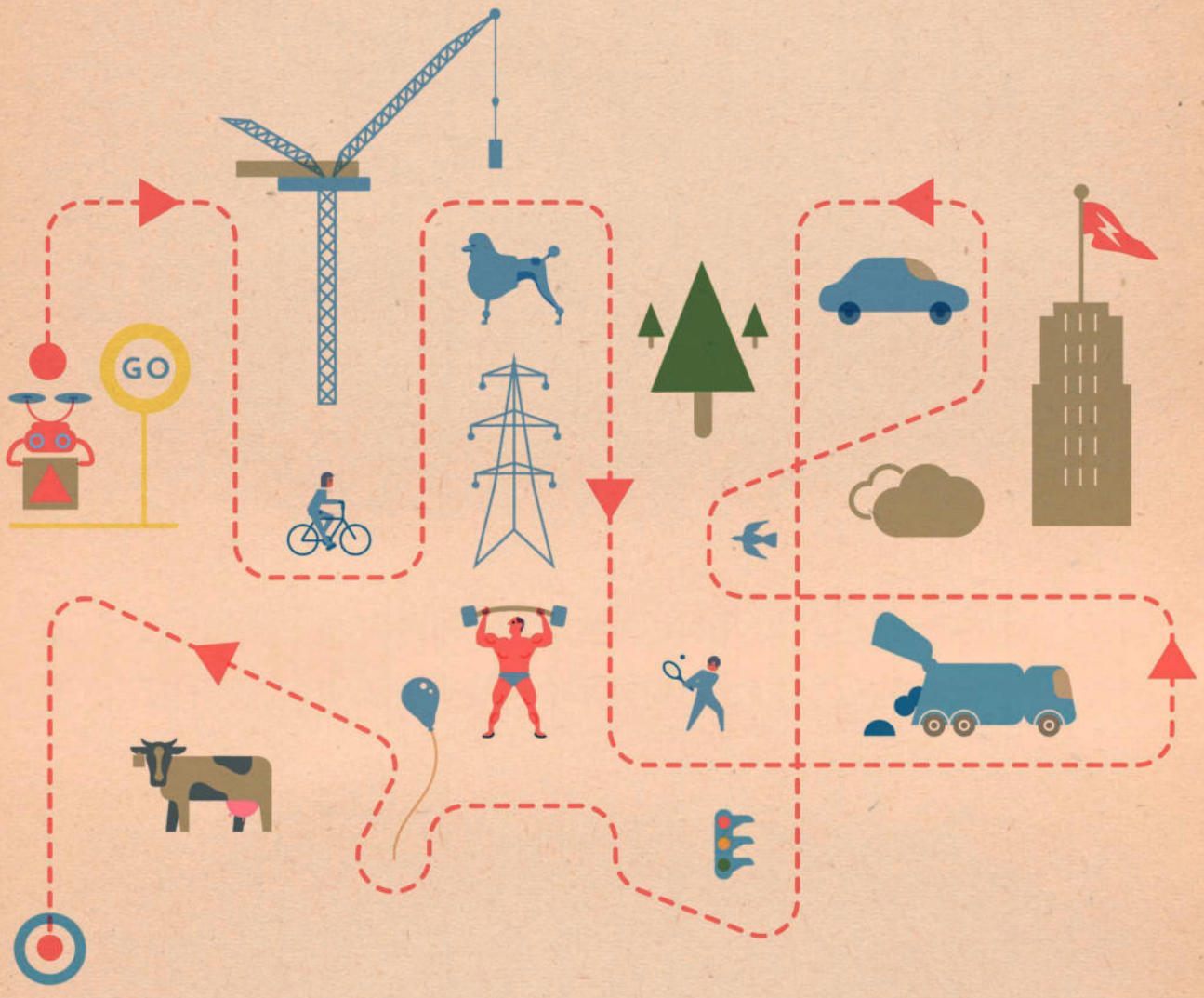

Intelligent Autonomous Systems

Atif Mashkoor, Software Competence Center Hagenberg GmbH and Johannes Kepler University

Paolo Arcaini, National Institute of Informatics

Angelo Gargantini, University of Bergamo

Digital Object Identifier 10.1109/MC.2020.3023158
Date of current version: 19 November 2020

0018-9162/20©2020IEEE

The bourgeoning industry of intelligent autonomous systems (IASs) presents numerous opportunities for cost-efficient automation. IASs also pose many challenges, including adapting to and overcoming uncertain situations they are likely to encounter.

Modern systems, such as cooperative robotic systems, mobile computing systems, unmanned aerial vehicles, and financial systems, are becoming distributed, ubiquitous, and systems of systems composed of autonomous entities. They have to operate in highly dynamic and volatile environments where physical infrastructure, social and societal context, network topologies, and workloads are continuously fluctuating. Consequently, intelligent and autonomous software-intensive behaviors become indispensable characteristics of such systems.

Intelligent autonomous systems (IASs)[1] are composed of communicating autonomous components whose behavior may be volatile, that is, components may break down, become unavailable due to network problems, or change their behavior (examples of IASs are shown in Figure 1). Consequently, the system has to be intelligent enough to recognize the faulty behavior, adapt itself to new arising situations if possible, and return to its original processing in case the cause of the problem has been removed. Thus, monitoring the system's environment and adapting its behavior to critical situations are other defining characteristics of IASs. Apart from being aware of their capabilities and limitations, IASs are also capable of reasoning over a diverse body of knowledge. The core of IASs is software shaping the behavior of related industries, and the recent advances in the areas of artificial intelligence, machine learning, and deep learning provide IASs with further improved robustness and flexibility.

According to some of the world's leading advisers on business strategy, for example, McKinsey,[2] Boston Consulting Group,[3] and Roland Berger,[4] billions of dollars will be spent worldwide on IASs in years to come, and software will be the linchpin. In the future, using software enhanced through machine vision, motion sensors, image and voice recognition, and artificial intelligence, IASs will be able to handle increasingly intelligent work, including interacting with and continuously learning from their environment and especially from people. Although these estimates provide a glimpse of the potential of such systems, they also show many challenges that may take longer to surmount than the enthusiastic early projections suggest. Some of those challenges are the following:

- *Safety and security assurance*: Classical safety and security assurance is usually performed during the design and development phases. However, the evolving nature of IASs renders design-time safety and security assurance infeasible and calls for approaches supporting continuous on-the-fly safety and security considerations.[5,6]
- *Modern design and development strategies*: IASs need to be able to adapt and reconfigure at runtime to handle environmental changes, recover from faults,

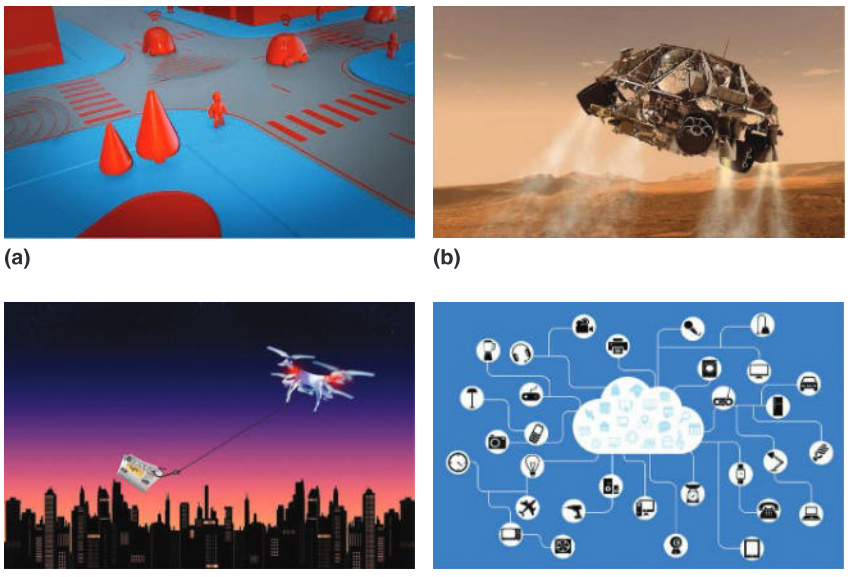

FIGURE 1. Examples of IASs. (a) Autonomous vehicles (source: Julien Tromeur, Pixabay), (b) a rover (source: David Mark, Pixabay), (c) a drone (source: Jan Alexander, Pixabay), and (d) the Internet of Things (source: jeferrb, Pixabay).

and so forth. Moreover, IASs are usually distributed, and the decision process is decentralized over different components. The traditional architectural and development strategies are not suitable for these kinds of systems anymore, and more modern/futuristic development practices are needed.

- *Dealing with uncertainties*: At runtime, IASs must be able to deal with unknown situations and possibly adapt to them. This requires the use of specification formalisms, such as state-based formal methods,[7] providing guarantees for behavioral correctness and the ability to capture or adapt to unknown situations, and the proper learning of resynthesis methods.
- *Timeliness*: At runtime, IASs must be able to monitor the environment and react in a timely manner. To this end, efficient techniques for runtime monitoring and online data processing are needed.

IN THIS ISSUE

For this issue, we received 11 submissions from around the globe. After thorough and stringent reviews, we selected six articles that provide relevant contributions to the topics covered in this issue.

In "Computational Intelligence for the Safety Assurance of Cooperative Systems of Systems" Kabir and Papadopoulos tackle the problem of providing, during operation, continuous safety assurance of autonomous systems composed of multiple independent subsystems that connect and cooperate dynamically at runtime. The authors propose a framework in which different intelligent agents observe and enforce the dependability of the individual subsystems. Each agent can make mitigation actions on its respective monitored subsystem and share information with other agents to cooperate, ensuring the safety of the whole system. The authors demonstrate the proposed approach on an autonomous production cell system.

Asaadi et al. propose using assurance cases for ensuring the trust of autonomous systems that depend on machine learning components capable of changing over time, either because of continuous learning or retraining in "Dynamic Assurance Cases: A Pathway to Trusted Autonomy." However, classical assurance cases are usually assessed offline before system development; as such, they are not suitable to handle the dynamic nature of autonomous systems. Therefore, the authors propose dynamic assurance cases as the combination of static assurance artifacts and assurance measures that provide quantitative values to estimate the confidence in the satisfaction of the assurance property. The approach is evaluated on an unmanned aircraft system embedding a generic autonomous taxiing capability.

In "Architectural Solutions for Self-Adaptive Systems," Garcés et al. tackle the problem of designing a self-adaptive system that can, at runtime, reconfigure its architecture to adapt to environmental changes, internal faults, unexpected behaviors, and so on. Designing such systems is particularly challenging and usually, a typical solution cannot be reused in different contexts. The article proposes four domain-independent solutions that should guide and facilitate the design of self-adaptive systems. The approach is demonstrated on the design of a river monitoring system.

Christie et al. consider Internet of Things (IoT) applications that involve different autonomous parties cooperating to achieve their goals in "Protocols Over Things: A Decentralized Programming Model for the Internet of Things." In such applications, there are multiple loci of control distributed across different components. Main programming models are not suitable for this kind of system as they mainly support a single locus of control. Therefore, the article proposes a programming model, Protocols Over Things, that supports both distribution and decentralization.

AT RUNTIME, IASs MUST BE ABLE TO MONITOR THE ENVIRONMENT AND REACT IN A TIMELY MANNER.

In "Controller Resynthesis for a Multirobot System When Changes Happen," Shi et al. tackle the problem of resynthesizing multirobot systems at runtime when some unexpected environmental change occurs; such runtime resynthesis is needed for systems that cannot be halted for redeployment. The authors show how the specification must be evolved enough to consider the environmental change (for example, sensors or

actuators that are no longer available) and propose a two-step controller synthesis method that attempts to synthesize a controller, satisfying the evolved specification as much as possible.

In "Real-Time Object Processing and Routing for Intelligent Drones: A Novel Approach," Sarkar et al. consider the problem of object processing and routing for unmanned aerial vehicles (UAVs) in which a route for a UAV must be generated that covers all of the objects under consideration with minimal cost. The authors propose a framework in which the UAV uses a camera for top-view imaging and an object-recognition algorithm to determine the image coordinates of object locations. The framework then maps image coordinates to real-world locations and calculates the shortest route between the coordinates of detected objects. The framework is experimented on a parking lot scenario.

ABOUT THE AUTHORS

ATIF MASHKOOR is a senior research scientist at the Software Competence Center Hagenberg GmbH and at Johannes Kepler University Linz, Austria. His research interests include systems and software engineering. Maskhoor received a Ph.D. in computer science from the University of Lorraine. Contact him at atif.mashkoor@scch.at.

PAOLO ARCAINI is project associate professor at the National Institute of Informatics, Japan. His main research interests are in search-based testing, fault-based testing, model-based testing, software product lines, and automatic repair. Arcaini received a Ph.D. in computer science from the University of Milan. He is a Member of IEEE. Contact him at arcaini@nii.ac.jp.

ANGELO GARGANTINI is associate professor at the University of Bergamo, Italy. His research interests include rigorous specification of critical systems, certification of medical systems and software, model-based testing, and neural networks testing. Gargantini received a Ph.D. in computer engineering from the Politecnico of Milan. He is a Member of IEEE. Contact him at angelo.gargantini@unibg.it.

ACKNOWLEDGMENTS

The research reported in this article was partly funded by the LIT Secure and Correct Systems Lab; the Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology; the Federal Ministry for Digital and Economic Affairs; and the Province of Upper Austria in the framework of the Competence Centers for Excellent Technologies, a program managed by the Austrian Research Promotion Agency FFG. Paolo Arcaini is supported by ERATO HASUO Metamathematics for the Systems Design Project (JPMJER1603) and the JST (10.13039/501100009024 ERATO).

REFERENCES

1. S. G. Tzafestas, *Advances in Intelligent Autonomous Systems*, vol. 18. New York: Springer-Verlag, 2012.
2. J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, and Alex Marrs, "Disruptive technologies: Advances that will transform life, business, and the global economy," McKinsey Global Institute, Boston, May 1, 2013. Accessed: Sept. 30, 2020. [Online]. Available: https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/disruptive-technologies
3. M. Wolfgang, V. Lukic, A. Sander, J. Martin, and D. Küpper, "Gaining robotics advantage," BCG, June 14, 2017. Accessed: Sept. 30, 2020. [Online]. Available: https://www.bcg.com/publications/2017/strategy-technology-digital-gaining-robotics-advantage
4. W. Bernhart et al., "Autonomous Driving," Nov. 2014, Roland Berger, Munich. Accessed: Sept. 30, 2020. [Online]. Available: https://www.rolandberger.com/publications/publication_pdf/roland_berger_tab_autonomous_driving.pdf
5. M. Biró, A. Mashkoor, J. Sametinger, and R. Seker, "Software safety and security risk mitigation in cyber-physical systems," *IEEE Softw.*, vol. 35, no. 1, pp. 24–29, 2018. doi: 10.1109/MS.2017.4541050.
6. A. Mashkoor, J. Sametinger, M. Biró, and A. Egyed, "Security- and safety-critical cyber-physical systems," *J. Softw. Evol. Process*, vol. 32, no. 2, p. e2239, 2020. doi: 10.1002/smr.2239.
7. A. Mashkoor, F. Kossak, and A. Egyed, "Evaluating the suitability of state-based formal methods for industrial deployment," *Softw. Pract. Exp.*, vol. 48, no. 12, pp. 2350–2379, 2018. doi: 10.1002/spe.2634.