Testing of Autonomous Systems - Challenges and Current State-of-the-Art

Conference Paper · July 2016

CITATIONS
14

READS
1,957

3 authors:

Philipp Helle
Airbus Central R&T
25 PUBLICATIONS **171** CITATIONS
SEE PROFILE

Wladimir Schamai
Danfoss
32 PUBLICATIONS **412** CITATIONS
SEE PROFILE

Carsten Strobel
Airbus Group
8 PUBLICATIONS **58** CITATIONS
SEE PROFILE

26th Annual INCOSE International Symposium (IS 2016)
Edinburg, Scotland, UK, July 18-21, 2016

Testing of Autonomous Systems - Challenges and Current State-of-the-Art

Philipp Helle
Airbus Group Innovations
Hamburg, Germany
philipp.helle@airbus.com

Wladimir Schamai
Airbus Group Innovations
Hamburg, Germany
wladimir.schamai@airbus.com

Carsten Strobel
Airbus Group Innovations
Munich, Germany
carsten.strobel@airbus.com

Abstract. Autonomous systems are on the rise. However, the challenge to test autonomous systems to ensure their safe and fault-free behaviour is not solved yet. This is especially critical when we consider the fact that autonomous systems are often safety-critical systems envisaged to interact with humans without explicit human supervision. This paper points out why testing autonomous systems is such a challenge and provides an overview of the current state-of-the-art theory and practice. The gathered information is then condensed into guiding points for the way forward.

Introduction

Without a doubt, autonomous systems represent a growing market: (Mallors, 2013) estimates an ”untapped short term market value of circa 7 billion per annum just for relatively low level autonomy products and services”, (Manyika, Chui, Bughin, Dobbs, Bisson, & Marrs, 2013) state, that the ”potential economic impact of autonomous cars and trucks could be $200 billion to $1.9 trillion per year by 2025”, (Autefage, Chaumette, & Magoni, 2015) estimate the total market for civilian robots at more than 10 billion euros in 2012 and continue that it ”should exceed 100 billion euros before 2020”, and the US DoD had planned to spend a $24 billion-plus total budget for unmanned systems in the 2007-2013 timeframe (Clapper, Young, Cartwright, & Grimes, 2007). The reasons for this growing trend differ with the application domain. For space applications, autonomy in satellites or other space vehicles reduces the need for human attendance especially during long-term missions and also reduces the reliance on the communication link between the system and the ground station, which is costly in terms of energy and has a large delay in long-distance missions (Pecheur, 2000) (Brat & Jonsson, 2005). For aerospace applications, such as Unmanned Air Vehicles (UAVs), autonomy with significant in-mission executive power allows systems to potentially replace humans in dangerous tasks (Alexander, Hall-May, & Kelly, 2007) and ensures that in the case of remotely piloted vehicles that the UAV remains safe and controllable in case of a disruption of the command and control communication link (Schumann & Visser, 2006). For automotive applications, autonomy

in consumer cars promises to reduce the commute burden, decrease traffic congestion, improve road safety and reduce carbon emission (Bernhart & Winterhoff, 2014).

Most of the research in the area is focused on the *development* of autonomous systems. This is the case despite the fact that even today over 25% of total development costs are spent on verification and validation (V&V), i.e. testing that the product works correctly and fulfils requirements. The testing effort is projected to increase disproportionately when systems become more complex through the integration of autonomy (Tallant, Buffington, Storm, Stanfill, & Krogh, 2006), which will make current test methods not cost-effective for next-generation autonomous control systems. Intuitively, autonomous systems are hard to test. Their flexible, context-aware behaviour implies unpredictability and can lead to emergent behaviour, i.e. behaviour that was not necessarily specified, intended or consciously implemented during the system design (Miles, et al., 2010). A key technology challenge for the research in the area of autonomous systems should thus be the verification and validation.

Testing every system state, as is often done today for safety-critical systems, is becoming an impossible task as these systems react to more environmental stimulus and have larger decision spaces (Barhorst, Paunicka, Stuart, & Hoffman, 2011), (Hinchman, Clark, Hoffman, Hulbert, & Snyder, 2012) or as (Thompson, 2008) puts it more blatantly "there is a common misconception in the testing industry that all unmanned autonomous systems can be tested using methodologies developed to test manned systems". A strong emphasis on developing methods to achieve V&V of complex, highly adaptive, autonomous systems will thus be essential to enabling the capabilities they can provide or as Macias says, "the evolutionary nature of UAS [..] must be met with evolutionary test capabilities yet to be discovered and developed" (Macias, 2008). And this will not be an easy task as Dahm writes that "developing certifiable V&V methods for highly adaptive autonomous systems is one of the major challenges facing the entire field of control science, and one that may require the larger part of a decade or more to develop a fundamental understanding of the underlying theoretical principles and various ways that these could be applied" (Dahm, 2010).

This paper provides an overview of the current state-of-the-art theory and practice regarding the stated challenge: V&V of autonomous systems.

This paper is structured as follows: Section *Autonomous Systems* will provide background information regarding autonomous systems and tries to give some key characteristics of autonomous system that influence the V&V challenge. Based on this, Section *Challenges for Testing Autonomous Systems* elaborates key challenges before Section *Current Approaches for Testing of Autonomous Systems* provides examples of currently used approaches for testing autonomous systems. The Section *Synopsis* will condense the information presented thus far into some guiding points and Section *Conclusion* wraps everything up with a conclusion.

Autonomous Systems

Autonomy, originating from *auto = self* and *nomos = law*, in the most basic definition means self-governance or freedom from external influence or authority (Nguyen, Miles, Perini, Tonella, Harman, & Luck, 2012). To avoid confusion it should be clarified that there is a significant difference between the words autonomous and automatic that are sometimes erroneously used interchangeably. Automatic means that a system will do exactly as programmed, without a choice. Autonomous means that a system can make choices without considering outside influence, i.e., "an autonomous system has free will" (Clough, 2002). Consider autonomy as "the ability to reason and make decisions to

reach given goals based on a systems current knowledge and its perception of the variable environment in which it evolves" (Rudd & Hecht, 2008). Autonomous systems must perform according to the specification under significant uncertainties in the operational environment for extended periods of time and they must be able to cope with a certain amount of system failures without external intervention (Antsaklis, Passino, & Wang, 1988). Hölzl et. al. (Wirsing, Hölzl, Koch, & Mayer, 2011) provide four major autonomous systems principles:

- **Knowledge:** The system knows facts about itself and its surroundings.
- **Adaptation:** The system can adapt its own behaviour dynamically to cope with changing surroundings.
- **Self-awareness:** The system can examine and reason about its own state.
- **Emergence:** Simple system elements construct complex entities.

Similar definitions are provided by (Micskei, Szatmári, Oláh, & Majzik, 2012) who write that "notable characteristics shared by the different kinds of autonomous systems include reasoning, learning, adaptation and context-awareness" and others such as (Watson & Scheidt, 2005) and (Brat & Jonsson, 2005). Of course not all autonomous systems exhibit the same level of autonomy. A variety of scales for describing the Level of Autonomy (LoA) achieved by a system have been proposed of which the most influential one probably was developed for military UAVs (Clough, 2002). Higher levels of autonomy of course typically pose a higher challenge for V&V. Another important preliminary point to make here is that in all modern systems, autonomy is realized in software (Schumann & Visser, 2006), which means that V&V of autonomous systems often boils down to software testing.

From the basic definition of autonomy we can develop common characteristics of autonomous systems. Adaptive Systems are systems whose function evolves over time, as they change their performance through learning. The advantage of adaptive systems is that they can, through judicious learning, react to situations for which the designer did not make specific provisions (Mili, Cukic, Liu, & Ayed, 2003).

Most autonomous systems use a model-based approach. In a model-based system, application-specific information, like the capabilities and limitations of the system, are described in a model of the application, not in implemented code. A general purpose reasoning system uses the model to operate the entity in question (Brat & Jonsson, 2005).

Automatic planning as a kind of predecessor of autonomy is an offline process which can most often be handled independent of system operation. Given an initial state, a goal and a set of possible operators and actions that can be used, the objective in automated planning is to derive an optimal plan to achieve the goal that can then be executed automatically. This problem is quite different when we consider autonomous systems, which operate in open environments, which cannot be completely foreseen at design time. In this case, planning must happen continuously and in parallel to its execution so that previously unanticipated events that occur during the plan execution can be considered in the next planning step.

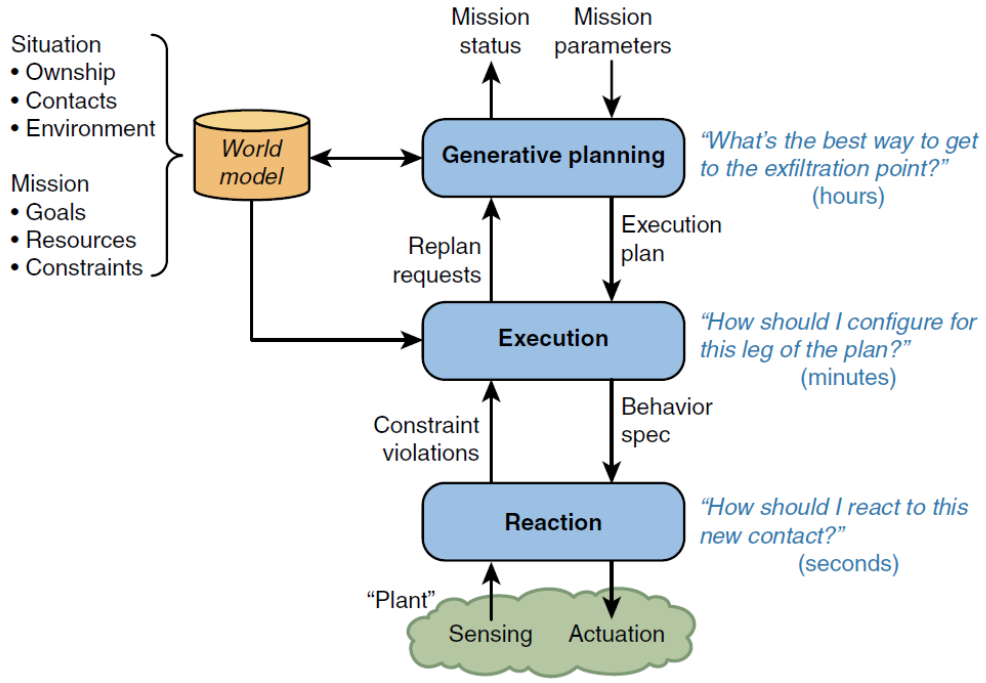

Figure 1. Layered Autonomy Model (Watson & Scheidt, 2005)

To reflect this, a canonical architecture has emerged in the research community based on the notion of layered control loops (Gat, 1998) (Albus & Meystel, 1996). Figure 1 reproduced from (Watson & Scheidt, 2005) shows a simplified example of this architecture, where the lowest-level control loops are used to provide feedback control with deterministic responsiveness. This control is reactive in the sense that the system will be driven to some local optima with respect to the overall behaviour goals, e.g., maintaining system safety. This local control set point is determined by the next level of the architecture, which is called the plan executor. This loop uses filtered perception data to assess the progress of the system through a pre-planned sequence of states. Responsiveness at this level is limited to contingency plans that have been prepared in advance and the event conditions that trigger them. At the highest level of control, a deliberative planning process uses explicit, declarative models of the system and its environment, combined with state information from the plan execution layer, to determine if the current situation requires global re-planning. All layers operate asynchronously and in parallel to produce the controlled behaviour.

Challenges for Testing Autonomous Systems

There is a consensus among researchers that if testing complex systems is hard, then testing complex autonomous systems is even harder and as stated in a recent paper, "testing autonomous systems is still an unsolved key area" (Weiss, 2011) or as Brat and Jonsson (Brat & Jonsson, 2005) state, our current validation techniques struggle with existing mission systems and now we are faced with validating autonomous systems that can exhibit a much larger set of behaviours.

There are a number of causes for this:

- **Complex environment:** The larger size and higher complexity of the valid input space and the context in which the system operates, both of which might be partially unknown at design time. This makes exhaustive testing - as it is done traditionally - impossible (Brat & Jonsson,

2005), (Schumann & Visser, 2006), (Clapper, Young, Cartwright, & Grimes, 2007), (Micskei, Szatmári, Oláh, & Majzik, 2012), (Brun, 2009) and (Pouly & Jouanneau, 2012).

- **Complex software:** The complexity of the program logic, which is caused by characteristics that are inherent in autonomous systems such as assembling their own course of actions for a given goal, adapting to different environments, learning, diagnosing themselves and reconfiguring themselves to maintain their functionality (Schumann & Visser, 2006), (Mikaelian, 2010) and (Brat & Jonsson, 2005). (Brat & Jonsson, 2005) observe that "verifying a planner is an enormous challenge considering that planners are meant to find intricate solutions in very large state spaces".

- **Non-deterministic behaviour:** Since often the autonomous behaviour is based on some kind of learning mechanism, autonomous systems may react differently to the same inputs over time, since their knowledge on which their behaviour is based changes over time (Nguyen, Miles, Perini, Tonella, Harman, & Luck, 2012), (Mikaelian, 2010), (Schumann & Visser, 2006), (Cukic, 2001) and (Menzies & Pecheur, 2005). This also means that one successful test does not guarantee that the system will pass the same test on the next test run. (Kurd, 2005) identifies a key challenge as being the difficulty of understanding the model that the system has learned (behaviour transparency and representation).

There is a second angle that makes testing autonomous systems even more important. Since autonomous systems are by design not always under human control or supervision, it is crucial to ensure their behaviour is safe (Mikaelian, 2010) especially since "society holds robots to a higher standard and has a lower tolerance for their errors" (Weiss, 2011). This is a major impediment for the certification of autonomous system as (Dahm, 2010) acknowledges that "it is possible to develop systems having high levels of autonomy, but it is the lack of V&V methods that prevents all but relatively low levels of autonomy from being certified for use". Since autonomous systems are often safety-critical (e.g. flight control) systems, they have to adhere to strict certification standards, leaving a wide technological gap between the requirements of the application domain and the capabilities of available technologies. (Heitmeyer & Leonard, 2015) see a major problem for autonomous systems in the "Human Mistrust of Automation/Autonomy" and (Zwillinger, Palmer, & Selwyn, 2014) state that two kinds of trust are needed so that a user accepts an autonomous system:

- **System trust:** Human confidence that the system behaves as intended. Achieving this trust requires a high assurance that the system satisfies its requirements, i.e. the traditional V&V challenges.

- **Operational trust:** Human confidence that system helps the user to perform the assigned tasks. Achieving this trust requires a high assurance that the scenarios for which the system was designed are useful.

US Air Force (USAF) Chief scientist Werner Dahm identified control science, i.e., the systematic way to study certifiable validation and verification (V&V) methods and tools to allow humans to trust decisions made by autonomous systems as a top priority for the USAF saying that "the major barrier that prevents the USAF from gaining more capability from autonomous systems is the lack of V&V methods and tools" (Dahm, 2010).

So, why can current testing methods not cope with the challenge? The currently used methods for software testing can be classified into three categories:

- **Fault Avoidance**, which is built on the premise that it is possible to build fault-free systems by design.
- **Fault Removal,** which is an ex post facto approach, which is based on the assumption that we can remove all faults from systems by extensive testing and corrections.
- **Fault Tolerance**, which concedes that a fault-free system is unrealistic in practice and tries to ensure that residual faults do not lead to system failures.

Unfortunately, none of these three methods can cope with the challenges of autonomous system testing at the moment for different reasons (Mili, Cukic, Liu, & Ayed, 2003):

- **Fault Avoidance:** Formal design methods exist for current systems that can proof that design and implementation formally adhere to the requirements. This does not work for autonomous systems where the behaviour not only depends on design and implementation but also on the acquired knowledge of the system, which could not have been included in the formal verification process during design time.
- **Fault Removal:** All testing methods are built on the assumption that the system under test will behave in the same way in the field as they do in the test environment and, consequently, that if a system passes all tests then it will work flawlessly during operation. And, as said before, this does not hold for autonomous systems, whose behaviour can change over time (Alexander, DelGobbo, Cortellessa, Mili, & Napolitano, 2000).
- **Fault Tolerance:** Fault tolerance techniques are based on the assumption that we can formalize the required behaviour of software in a way that we can use them to design error detection and recovery capabilities. With adaptive systems, it is not possible to formulate such expectations because the full required behaviour in all working conditions is not known at design time.

Current Approaches for Testing of Autonomous Systems

Having stated the specific problems that come with the testing of autonomous systems, this section provides examples of existing approaches to these challenges. The goal is not to present the approaches in detail but to identify which methods and tools are used in order to form a comprehensive list of technologies that are useful for testing autonomous systems. It should be noted, that most of the presented approaches are from research projects.

(Pouly & Jouanneau, 2012) present an approach for development and testing of an autonomous satellite that was elaborated in the frame of the AGATA project (Charmeau & Bensana, 2005). The approach relies on using a Model-based Systems Engineering approach based on the Unified Modelling Language (UML) with automatic code generation and uses a model-based testing approach, i.e., automatic generation of test cases from a model, to support the verification of the on-board software. It reshapes the traditional V-shaped development process into a Y-shaped production cycle, which enables an incremental development process in which the software

validation can start much earlier. They further state that "current works are exploring how to integrate model-checking in our process" (Pouly & Jouanneau, 2012).

(Hölzl, et al., 2011) present results from the ASCENS (Autonomic Service-Component Ensembles) project which targets "ensembles: systems with massive numbers of nodes, operating in open and non-deterministic environments with complex interactions, and the need to dynamically adapt to new requirements, technologies or environmental conditions without redeployment and without interruption of the systems functionality". In their effort to develop a "coherent, integrated set of methods and tools to build software for ensembles" they are combining several methods: A Model-based approach with different domain-specific languages (DSLs) built on components for the construction and run-time monitoring including predictive analysis to support the verification of the systems.

(Lill & Saglietti, 2012) present a model-based testing approach for testing the cooperative behaviour of software controlled autonomous systems using Coloured Petri Nets (CPN) (Jensen & Kristensen, 2009) in order to identify adequate test scenarios and provide objectively measurable test stopping rules for testing especially the interaction between autonomous systems.

(Horányi, Micskei, & Majzik, 2013) propose a model-based testing approach using UML for testing the control module of autonomous robots. They use a monitor-based testing approach where requirements are formalized into "observer automata", which are then used as test oracles and test data is automatically generated from a context model of the system.

A similar approach is presented by (Nguyen, Miles, Perini, Tonella, Harman, & Luck, 2012). They use meta-heuristic search techniques for online test generation, calculating the fitness of potential test data on the basis of evaluating the execution of the system under test and optimization to generate demanding test cases. They conclude that this evolutionary test approach "can test agents in a greater range of contexts than standard tests, thereby accounting for their autonomy to act differently in each such context".

(Berger & Rumpe, 2012) report experience from their participation at the 2007 DARPA Urban Challenge. They used a mostly traditional software development approach only partially supported by some UML modelling activities. Regarding testing, they developed a complete virtual test environment that could supply the control software with virtual raw sensor inputs to enable automatic testing as early as possible in a "simulate first approach" where all tests are conducted in the virtual test environment before they are conducted in the real world.

A similar stance is taken by (Mutter, et al., 2011) and (Bayha, Grüneis, & Schätz, 2012) from the domain of autonomous aerial vehicles. They worked on a virtual testing environment for autonomous UAVs that allows to "test the software as-is, i.e., without any modifications or instrumentation for testing purposes" (Bayha, Grüneis, & Schätz, 2012). They state that "virtual integration of the system makes it possible to test a software implementation without endangering the hardware or the environment" (Mutter, et al., 2011).

(Thompson, 2008) also advocates testing in virtual environments and takes this even further when he states that "virtual testing must become a standard complement to field-testing UASs if the testing community is ever going to be able to test an intelligent UAS safely and comprehensively" and that "hardware-in-the-loop is a necessity of any virtual test environment".

(Scrapper, Balakirsky, & Messina, 2006) also report on virtual environments for testing of autonomous mobile robot systems. And while they advocate the usage of virtual testing they also warn that not everything can be tested virtually, at least not with today's simulation capabilities, and that "much further testing will be needed in the real world". They see deficits in the representation of complexity and noise found in the real world, the approximation to reality of simulated sensor feeds, the complete and accurate modelling of the system under test's physical behaviour.

(Brat & Jonsson, 2005) present an approach that relies heavily on composition for design and verification. Their design process relies on creating new systems based on known building blocks in an assume-guarantee framework. For the verification of autonomous systems they combine "advanced verification techniques (static analysis, model checking, compositional verification, and automated test generation)" with automatic generation of certified code.

Further advocates of formal verification, i.e., static analysis, model checking and runtime verification, for testing autonomous systems include (Schumann & Visser, 2006), (Feather, Fesq, Ingham, Klein, & Nelson, 2004), (Cheng, et al., 2014), (De Lemos, et al., 2013) and (Simmons, Pecheur, & Srinivasan, 2000).

Synopsis

From the characteristics of autonomous systems, the challenges that they present for V&V and the already existing approaches to solve these challenges, this section tries to provide a synopsis in the form of a list of "things to do" that will support successful autonomous systems development and testing.

1. **Use models:** It has been shown that using a Model-based Systems Engineering (MBSE), e.g., (Helle & Schamai, 2014), (Estefan, 2008), approach can significantly reduce the number of defects that are introduced into a development project, especially in the early development stage (Saunders, 2011). This benefit increases in the scope of autonomous systems development, where exhaustive testing is not possible and it is therefore more difficult to ensure that a product is fault-free. Furthermore, the operational models that are part of most MBSE processes can be used to communicate the intended behaviour of the system to the potential users more easily and earlier on, which helps to build the operational trust and thereby the acceptance of the system.
2. **Be formal:** One common trend in the whole complex systems development area that could also be shown in the existing autonomous system testing approaches discussed in this paper is to move towards formal methods, which can be used to formally prove the absence of faults in specifications and/or code. The potential of formal methods have never been doubted but the widespread usage of formal methods so far have been hindered by two facts: Firstly, ease of use for non-experts and secondly, dealing with large state-spaces. The latter becomes even more of a challenge when we consider the fact that the state-space in autonomous systems development is typically bigger than for other systems. So, there is still work to be done. Nevertheless, formal methods can be used today and will help people building better systems.
3. **Automate:** Automatic test execution is already the current state-of-practice in industry. What is not, however, is automatic generation of test scenarios, i.e., model-based testing (MBT). It has been shown that MBT can significantly reduce the effort for testing (Kläs,

Bauer, Söderqvist, Dereani, & Helle, 2015) and a number of the presented approaches in the previous section consider it a useful tool for testing autonomous systems.

4. **Test early:** It has been shown before that fixing defects becomes more expensive the later the defect is discovered in the product lifecycle. (Laycock, 1993) claims that ”the effort needed to produce test cases during each phase will be less than the effort needed to produce one huge set of test cases of equal effectiveness on a separate lifecycle phase just for testing”. Especially, if we found our development on models, as advised by the proposition "Use models", we can start the testing of these models as soon as they are built thereby preventing errors from trickling down and snowball into a big problem during a separate testing phase.
5. **Test continuously:** Runtime monitoring can be used to continue monitoring the intended behaviour of a system even in service (Leucker & Schallhart, 2009), (Levy, Saïdi, & Uribe, 2002). Monitor-based testing, e.g., (Schamai, Helle, Fritzson, & Paredis, 2011), is a similar approach where we do not have to care so much about the resource consumption of the monitors on-board the system because they are running on the test bench. What both approaches have in common, is that they enable the testing of certain properties of a system in any context, even unknown ones and this makes them ideal approaches for testing of autonomous systems.
6. **Test virtually:** Google relies on extensive road testing of their autonomous vehicle and they have clocked more than 700000 miles so far (Johanning & Mildner, 2015). This is done in an effort to gain trust in the system and prove to the world that autonomous driving is possible today. This approach of testing extensively in real-life service is not feasible for some autonomous systems, e.g., in the space domain, and becomes very costly and time-consuming and therefore impractical if autonomy becomes a regular feature. Virtual testing, which is not a new topic, can help with that. Firstly, virtual testing can start with imperfect systems that fail frequently and which we would not want to test in a real environment because nothing serious can happen. Secondly, virtual testing scales better because it can be done in parallel on several virtual test benches. It is only limited by the available computing power and Moore’s Law still holds. Thirdly, coming back to the third bullet point, virtual testing can be started before the first metal is cut. The challenges for virtual testing when it comes to autonomous systems testing is that the virtual test environment needs to be representative of the intended operational environment of the system under test and, as said before, this can potentially be the whole world with a very large state space.
7. **Start by testing the correctness of the autonomy capability:** Autonomy introduces a new level of abstraction - a meta-level for design and V&V. When designing a deterministic system, engineers decide how the system should react in particular situations. Autonomous systems are able to learn and adapt their behaviour to some extent. This implies that, now, the system itself will make decisions based on the data learnt so far, its current state and the state of its environment. Taking such decision requires some reasoning capability that will take into account reasoning rules, constraints or optimization objectives. Dealing with the challenge of ensuring that the system will be safe and reliable therefore means dealing with the verification of the reasoning component of the system in the first place. Having the confidence that the system will reason correctly,

traditional V&V approaches can still be used for ensuring that, after taking the correct decision, the system will be able to accomplish its mission.

8. **Think ahead:** All the points made so far have one thing in common: They have not made it into the industrial practice for complex systems development in a widespread fashion, yet. And, while they exist and have the potential to support the testing of autonomous systems, they do not have all the necessary features, yet, e.g. formal methods need improvement regarding state-space explosion, virtual test environments need to represent reality in a more detailed fashion, and so on. So, successful autonomous systems development and testing requires using methods and tools that are on the cutting edge of technology. As a result, this requires thinking ahead and forecasting, which of these cutting edge technologies are here to stay and which of them lead to a dead-end.

Conclusion

In the face of the rising complexity of safety, testing becomes one of the biggest challenges. This paper showed that and why this is even more true when considering testing of autonomous systems. Non-deterministic behaviour that depends on the acquired knowledge of the system during operation and the fact that autonomous system often have to operate in a large environment that is not necessarily fully known at design time are just two of the aspects that are responsible for this. A lot of effort is spent on research to support the testing of autonomous system and this paper gave an overview of what has been done and condensed this information into some guiding points for going forward into the future. Autonomous systems are on the rise and we need to make sure that we can test these systems and ensure their safe behaviour before they can be put into operation where they may interact with humans without explicit human supervision.

References

Albus, J. S., & Meystel, A. M. (1996). A reference model architecture for design and implementation of intelligent control in large and complex systems. *International Journal of Intelligent Control and Systems*, pp. 15-30.

Alexander, C., DelGobbo, D., Cortellessa, V., Mili, A., & Napolitano, M. (2000). Modeling the fault tolerant capability of a flight control system: An exercise in SCR specifications. *Proceedings of the Langley Formal Methods Conference.*

Alexander, R. D., Hall-May, M., & Kelly, T. P. (2007). *Certification of autonomous systems under UK military safety standards.* University of York.

Antsaklis, P. J., Passino, K. M., & Wang, S. J. (1988). Autonomous control systems: Architecture and fundamental issues. *American Control Conference* (pp. 602-607). IEEE.

Autefage, V., Chaumette, S., & Magoni, D. (2015). Comparison of time synchronization techniques in a distributed collaborative swarm system. *European Conference on Networks and Communications (EuCNC)* (pp. 455-459). IEEE.

Barhorst, J. F., Paunicka, J. L., Stuart, D. A., & Hoffman, J. (2011). Emerging Directions in Aerospace Software V&V. *Proceedings of Infotech@ Aerospace Conference*, (pp. 1507-1512).

Bayha, A., Grüneis, F., & Schätz, B. (2012). Model-based software in-the-loop-test of autonomous systems. *Proceedings of the 2012 Symposium on Theory of Modeling and Simulation-DEVS Integrative M&S Symposium.* Society for Computer Simulation International.

Berger, C., & Rumpe, B. (2012). Engineering autonomous driving software. In C. Rouff, & M. Hinchey, *Experience from the DARPA Urban Challenge* (pp. 243-271). Springer.

Bernhart, W., & Winterhoff, M. (2014). Autonomous Driving: Disruptive Innovation that Promises to Change the Automotive Industry as We Know It. In J. Langheim, *Energy Consumption and Autonomous Driving - Proceedings of the 3rd CESA Automotive Electronics Congress, Paris, 2014* (pp. 3-10). Springer.

Brat, G., & Jonsson, A. (2005). Challenges in verification and validation of autonomous systems for space exploration. *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN'05)* (pp. 2909--2914). IEEE.

Brun, Y. (2009). Software Engineering for Self-Adaptive Systems: A Research Roadmap. In B. Cheng, R. de Lemos, P. Inverardi, & J. Magee, *Software Engineering for Self-Adaptive Systems* (pp. 48-70). Springer.

Charmeau, M.-C., & Bensana, E. (2005). AGATA: A Lab Bench Project for Spacecraft Autonomy. *International Symposium on Artificial Intelligence Robotics and Automation in Space (iSAIRAS).*

Cheng, B. H., Eder, K. I., Gogolla, M., Grunske, L., Litoiu, M., Müller, H. A., et al. (2014). Using models at runtime to address assurance for self-adaptive systems. In *Models@run.time* (pp. 101-136). Springer.

Clapper, J., Young, J., Cartwright, J., & Grimes, J. (2007). *Unmanned systems roadmap 2007-2032.* Office of the Secretary of Defense.

Clough, B. T. (2002). *Metrics, schmetrics! How the heck do you determine a UAV's autonomy anyway.* Air Force Research Laboratory.

Cukic, B. (2001). The need for verification and validation techniques for adaptive control system. *Proceedings of the 5th International Symposium on Autonomous Decentralized Systems* (pp. 297-298). IEEE.

Dahm, W. J. (2010). *Technology Horizons a Vision for Air Force Science & Technology During 2010-2030.* Office of the US Air Force Chief Scientist.

De Lemos, R., Giese, H., Müller, H. A., Shaw, M., Andersson, J., Litoiu, M., et al. (2013). Software engineering for self-adaptive systems: A second research roadmap. In *Software Engineering for Self-Adaptive Systems II* (pp. 1-32). Springer.

Estefan, J. (2008). *Survey of Model-Based Systems Engineering (MBSE) methodologies.* International Council on Systems Engineering.

Feather, M. S., Fesq, L. M., Ingham, M. D., Klein, S. L., & Nelson, S. D. (2004). Planning for V&V of the Mars Science Laboratory rover software. *Proceedingsof the 2004 IEEE Aerospace Conference.* IEEE.

Gat, E. (1998). On three-layer architectures. *Artificial intelligence and mobile robots*.

Heitmeyer, C. L., & Leonard, E. I. (2015). Obtaining trust in autonomous systems: tools for formal model synthesis and validation. *2015 IEEE/ACM 3rd FME Workshop on Formal Methods in Software Engineering (FormaliSE)* (pp. 54-60). IEEE.

Helle, P., & Schamai, W. (2014). Towards an integrated methodology for the development and testing of complex systems - with example. *International Journal on Advances in Systems and Measurements*, pp. 129-149.

Hinchman, J., Clark, M., Hoffman, J., Hulbert, B., & Snyder, C. (2012). Towards Safety Assurance of Trusted Autonomy in Air Force Flight Critical Systems. *Computer Security Applications Conference, Layered Assurance Workshop.*

Hölzl, M., Wirsing, M., Klarl, A., Koch, N., Reiter, S., Tribastone, M., et al. (2011). *Engineering Ensembles: A White Paper of the ASCENS Project.*

Horányi, G., Micskei, Z., & Majzik, I. (2013). Scenario-based Automated Evaluation of Test Traces of Autonomous Systems. *SAFECOMP 2013 - Workshop DECS (ERCIM/EWICS Workshop on Dependable Embedded and Cyber-physical Systems) of the 32nd International Conference on Computer Safety, Reliability and Security.*

Jensen, K., & Kristensen, L. M. (2009). *Coloured Petri nets: modelling and validation of concurrent systems.* Springer.

Johanning, V., & Mildner, R. (2015). *Car IT kompakt: Das Auto der Zukunft--Vernetzt und autonom fahren.* Springer.

Kläs, M., Bauer, T., Söderqvist, T., Dereani, A., & Helle, P. (2015). A Large-Scale Technology Evaluation Study: Effects of Model-based Analysis and Testing. *Proceedings of the 37th International Conference on Software Engineering (ICSE 2015).* IEEE.

Kurd, Z. (2005). *Artificial neural networks in safety-critical applications.* University of York.

Laycock, G. T. (1993). *The theory and practice of specification based testing.* University of Sheffield.

Leucker, M., & Schallhart, C. (2009). A brief account of runtime verification. *Journal of Logic and Algebraic Programming*, pp. 293-303.

Levy, J., Saïdi, H., & Uribe, T. E. (2002). Combining monitors for runtime system verification. *Electronic Notes in Theoretical Computer Science*, pp. 112–127.

Lill, R., & Saglietti, F. (2012). Test Coverage Criteria for Autonomous Mobile Systems based on Coloured Petri Nets. *9th Symposium on Formal Methods for Automation and Safety in Railway and Automotive Systems (FORMS/FORMAT 2012)*, (pp. 155-162).

Macias, F. (2008). The test and evaluation of unmanned and autonomous systems. *ITEA Journal*, pp. 388–395.

Mallors, R. L. (2013). Autonomous systems: Opportunities and challenges for the UK. *IET Seminar on UAVs in the Civilian Airspace.*

Manyika, J., Chui, M., Bughin, J., Dobbs, R., Bisson, P., & Marrs, A. (2013). *Disruptive technologies: Advances that will transform life, business, and the global economy.* McKinsey Global Institute San Francisco, CA, USA.

Menzies, T., & Pecheur, C. (2005). Verification and validation and artificial intelligence. *Advances in Computers*, pp. 153-201.

Micskei, Z., Szatmári, Z., Oláh, J., & Majzik, I. (2012). A concept for testing robustness and safety of the context-aware behaviour of autonomous systems. In *Agent and Multi-Agent Systems. Technologies and Applications* (pp. 504-513). Springer.

Mikaelian, T. (2010). *A Real Options Approach to Testing.* MIT.

Miles, S., Winikoff, M., Cranefield, S., Nguyen, C. D., Perini, A., Tonella, P., et al. (2010). *Why testing autonomous agents is hard and what can be done about it.*

Mili, A., Cukic, B., Liu, Y., & Ayed, R. B. (2003). Towards the verification and validation of online learning adaptive systems. In *Software Engineering with Computational Intelligence* (pp. 173-203). Springer.

Mutter, F., Gareis, S., Schätz, B., Bayha, A., Grüneis, F., Kanis, M., et al. (2011). *18th IEEE International Conference and Workshops on Engineering of Computer Based Systems (ECBS)* (pp. 269-275). IEEE.

Nguyen, C. D., Miles, S., Perini, A., Tonella, P., Harman, M., & Luck, M. (2012). Evolutionary testing of autonomous software agents. *Autonomous Agents and Multi-Agent Systems*, pp. 260-283.

Pecheur, C. (2000). *Verification and validation of autonomy software at NASA.* National Aeronautics and Space Administration.

Pouly, J., & Jouanneau, S. (2012). Model-based specification of the flight software of an autonomous satellite. *Embedded Real Time Software Systems (ERTS 2012).*

Rudd, L., & Hecht, H. (2008). *Certification techniques for advanced flight critical systems.* WPAFB.

Saunders, S. (2011). Does a Model Based Systems Engineering Approach Provide Real Program Savings? Lessons Learnt. *Informal Symposium on Model-Based Systems Engineering.* DSTO.

Schamai, W., Helle, P., Fritzson, P., & Paredis, C. J. (2011). Virtual verification of system designs against system requirements. In J. Dingel, & A. Solberg, *Models in Software Engineering* (pp. 75-89). Springer.

Schumann, J., & Visser, W. (2006). Autonomy software: V&V challenges and characteristics. *Proceedings of the 2006 IEEE Aerospace Conference* (pp. 1233--1249). IEEE.

Scrapper, C., Balakirsky, S., & Messina, E. (2006). MOAST and USARSim: a combined framework for the development and testing of autonomous systems. *Defense and Security Symposium.* International Society for Optics and Photonics.

Simmons, R., Pecheur, C., & Srinivasan, G. (2000). Towards automatic verification of autonomous systems. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)* (pp. 1410-1415). IEEE.

Tallant, G. S., Buffington, J. M., Storm, W. A., Stanfill, P. O., & Krogh, B. H. (2006). *Validation & Verification for emerging avionic systems.*

Thompson, M. (2008). Testing the Intelligence of Unmanned Autonomous Systems. *ITEA Journal*, pp. 380–387.

Watson, D. P., & Scheidt, D. H. (2005). Autonomous systems. *Johns Hopkins APL technical digest*, pp. 368-376.

Weiss, L. G. (2011). Autonomous robots in the fog of war. *IEEE Spectrum*, pp. 30-57.

Wirsing, M., Hölzl, M., Koch, N., & Mayer, P. (2011). *Software Engineering for Collective Autonomic Systems - The ASCENS Approach.* Springer.

Zwillinger, D., Palmer, G., & Selwyn, A. (2014). *The Trust V - Building and measuring trust in autonomous systems.* Raytheon.

Biography

Philipp Helle joined Airbus Group Innovations in 2003 and is currently a member of the team *Model-based Systems and Software Engineering*. He studied linguistics and computer science and received his MA from the University of Hamburg. Since 2005, he is actively involved in research concerning model-based systems engineering including model-based testing and model-based safety analysis and the deployment of these approaches in the Airbus Group business units. Philipp is a member of GfSE, the German INCOSE chapter, and a certified Project Management Professional (PMP).

Dr. Wladimir Schamai is a researcher in the Systems Engineering department of Airbus Group Innovations. After studying computer science he worked as software developer for several years. He received his Ph.D. in Computer and Information Science from Linköping University in Sweden. Since 2005, he is actively involved in research concerning Systems Engineering.

Carsten Strobel is head of the Model-based System and Software Engineering research team of Airbus Group Innovations. He studied Business Engineering at the Karlsruhe Institute of Technology. Since joining Airbus Group in 2006 he is working in the field of MBSE at projects for all Airbus Group Divisions, mainly focusing on integrated model-based specification, design and testing.