RIACS Workshop on the Verification and Validation of Autonomous and Adaptive Systems

Charles Pecheur and Willem Visser
RIACS/NASA Ames

Reid Simmons
Carnegie Mellon University

April 10, 2001

1 Introduction

The long-term future of space exploration at NASA is dependent on the full exploitation of autonomous and adaptive systems: careful monitoring of missions from earth, as is the norm now, will be infeasible due to the sheer number of proposed missions and the communication lag for deep-space missions. Mission managers are however worried about the reliability of these more intelligent systems. The main focus of the workshop was to address these worries and hence we invited NASA engineers working on autonomous and adaptive systems and researchers interested in the verification and validation (V&V) of software systems. The dual purpose of the meeting was to (1) make NASA engineers aware of the V&V techniques they could be using and (2) make the V&V community aware of the complexity of the systems NASA is developing.

The workshop was held 5-7 December 2000 at the Asilomar Conference Center in Pacific Grove (near Carmel) California. It was co-organized by Charles Pecheur and Willem Visser from the Research Institute for Advanced Computer Science (RIACS) and Reid Simmons from Carnegie Mellon University. RIACS gave financial and administrative support, with Peggy Leising handling the local arrangements. We invited 42 participants to the workshop with 28 from the V&V community and 14 from the Autonomous and Adaptive system community; half of the participants were from NASA and the other half from universities and research labs.

The workshop was run over two days with the first being used for the presentation of four NASA autonomous and adaptive system development projects as well as one talk on the V&V of neural nets used in highway applications. The second day was used for three technology break-out sessions to discuss V&V issues of autonomous and adaptive systems. The workshop concluded with an open discussion on the results of the break-out sessions.

The five talks on the first day were the following:

- **Deploying Robust Autonomous Systems: Lessons Learned from the Remote Agent Experiment** by Nicola Muscettola from NASA Ames Research Center.
- **First Steps Towards Neural Net V&V** by Rodger Knaus from Instant Recall, Inc.
- **Stability Issues with Reconfigurable Flight Control Using Neural Generalized Predictive Flight Control** by Don Soloway from NASA Ames Research Center.
- **V&V of an Autonomous Agent for Mars Duty at KSC** by Peter Engrand from NASA Kennedy Space Center.
- **Distributive Adaptive Control for Advanced Life Support Systems** by David Kortenkamp from NASA Johnson Space Center.

The discussion topics for the break-out sessions were on V&V of Intelligent, Adaptive and Complex systems. In the rest of the report we first highlight some of the general issues that were raised during these three break-out sessions as well as in the wrap-up session that followed (section 2) and then give short summaries of each of the sessions in section 3 (Intelligent Systems), section 4 (Adaptive Systems) and section 5 (Complex Systems). Section 6 contains a short retrospective on the workshop and the future of the field. A full version of this report can be found online at http://ase.arc.nasa.gov/vv2000.

2 General Issues

Some of the issues that were discussed throughout the workshop range beyond autonomous and adaptive systems, into the more general fields of formal verification and software engineering. This section cites the more significant ones.

Scalability Lack of scalability is seen as a major obstacle of current verification methods such as model checking. There is definitely a need for improving and extending these methods in order to be able to address real-size systems. Since formal methods do not scale well it is most productive to apply formal methods to only the critical areas, where developers have least confidence in the correctness.

Software Engineering Practices Good software V&V starts with a good software engineering process, including clearly defined goals and requirements. Such practices are not as well established in the autonomy software community as in the mainstream software industry. Well-documented requirements are essential for driving the V&V work.

Metrics We have to define ways to measure and compare the utility of different verification efforts. For this, we need quantitative metrics that adequately address the different factors of the costs and benefits of each method. Such metrics are a necessity to clearly indicate "where you win" by using new verification approaches. It is noteworthy that most of the latest NASA project funding programs required the mention of such evaluation metrics.

Using Different Techniques It is rarely the case that a single V&V technique achieves good results on a real-world problem. In most cases, several techniques (model checking, testing, proofs, static analysis, etc) must be combined together to be able to tackle the complexity of the system to be verified. Progress in integrating different V&V techniques are therefore crucial.

Certification vs. Debugging V&V techniques can be used to achieve two complementary purposes: proving a system correct (certification), or proving it incorrect, i.e. finding errors (debugging). The terms "verification" vs. "falsification" have also been coined. Both eventually help to increase the confidence in the reliability of the system, but in different ways. Debugging is done in earlier stages, as part of the development, while certification is rather performed independently on the finished product. It should be noted that "easy" V&V techniques such as model checking are often limited to debugging, because the state space of real-size systems cannot be completely covered.

Design for Verification V&V can be facilitated if all components are designed with verification in mind. For example, V&V of a fault diagnosis system is easier if the controlled system has mechanisms to inject or simulate faults.

Run-Time Verification Automatic verification techniques, such as model checking, can also be useful at run-time, during normal operation. For example, model checking can be used to check the results of a heuristic AI-based algorithm such as a planner. This combines the efficiency of heuristic search with the robustness and formality of verification.

3 V&V of Intelligent Systems

3.1 Attendance and Scope

This break-out group gathered thirteen people, six of them from NASA. The topic had been set to verification and validation of *intelligent systems*. This had been defined as *systems based on some form of artificial intelligence technique*, such as model-based, rule-based or knowledge-based systems. In accordance with the theme of the workshop, there was of course an interest in autonomous and adaptive systems, but the focus was specifically on the AI-related issues of such systems.

The moderator (Charles Pecheur) briefly introduced the topic and presented some proposed issues, then a lively, free-form debate ensued. The discussion turned out to be strongly focused on model-based systems. This did not stem from an intentional orientation or to a perception that such systems are more relevant to the topic, but rather from a strong involvement of model-based autonomy specialists in the discussion.

This section reports on discussion topics related to model-based systems. Many of the more general issues presented above (section 2) also arose along the discussion.

3.2 V&V of Model-Based Systems

For the purpose of this discussion, a model-based autonomous system is viewed as a *plant* (a spacecraft, a robot, etc.) driven by a *controller* through a command/sensor feedback loop. The controller itself is based on a generic, AI-based inference *engine* that peruses an abstract *model* of the plant. The engine infers the appropriate control actions based on the feedback it receives from the plant and its knowledge about the plant extracted from the model.

As an overall issue, there is a need to define and build experience in the software engineering process for model-based systems. What are the types of requirements that a customer would expect; how could these requirements be conveniently expressed and verified? Can we develop/specialize a theory and practice of testing for this kind of systems? In this prospect, abstract autonomy models could provide a good basis for automatic generation of test cases.

A natural approach is to decompose the V&V problem across the three core components of a model-based system: the plant, the engine and the model. V&V of the plant is outside the scope of this discussion. V&V of the engine is a complex task that needs to be addressed, but concerns the designers of that engine. From the point of view of the application designer, V&V focuses on the model and how it affects the operation of the whole system. This can be decomposed into two threads:

- How do we build and verify/validate autonomy models?
- Given a "valid" model, how do we verify/validate the resulting autonomous controller?

Note that because the model concentrates all the application-specific knowledge in a very abstract representation, it is potentially more amenable to V&V, even for larger systems.

There is even a hope that model-based autonomy is "correct by design": if the model directly captures the specification of the plant, then the correctness of the controller derives from the correctness of the logic inference principles implemented in the engine. However, experience shows that authoring autonomy models is still a difficult and error-prone task, and that full correctness (and completeness) of the engine is not always achieved or even desired, for

performance purposes. In practice, both threads above are needed and complementary.

Part of the difficulty resides in reliably writing complete and consistent models. Accordingly, tools for checking consistency and completeness would be useful. Some of this difficulty may be inherent to declarative specifications, but a part of it could be alleviated by richer modeling languages too.

The model itself can be further decomposed into the different aspects that it captures, such as the plant (e.g. the moving range of a camera), the operational constraints (e.g. do not point the camera towards the Sun) and the goals (e.g. minimize the delay when the camera moves). Though all three may be expressed in the same logic formalism, they entail different V&V activities and should thus be distinguished from each other. Another interesting issue is the fusion of partial models, involving conflict resolution principles.

Finally, a comparison can be made with the field of classical feedback control. For linear systems, one can, on mathematical grounds, extrapolate a limited set of observations to entire regions of the control space. We should investigate whether the high-level, uniform inference laws used in model-based reasoning would allow a similar reasoning. This is a very speculative idea, given the complexity and non-linearity of autonomous controllers, but it could dramatically decrease the cost of verification if it proved successful.

4 V&V of Adaptive Systems

4.1 Attendance and Scope

This break-out group gathered nine participants, four of them from NASA, around the topic of verification and validation of adaptive systems. In particular, the group focused on control systems that do learning, either off-line (pre-trained) or on-line. While the focus was fairly broad, much of the discussions centered around approaches based on neural networks.

4.2 Introduction

Initial discussion centered on how verification of adaptive systems differs from verification of traditional control systems. One point was made that adaptive systems have more potential fault modes, and so can behave more unpredictably. Another point is that most commercial V&V products are based on the software engineering *process*, and so are not really appropriate to learning systems, where the development of the learning program is often secondary to the way it is trained. It is also the case that most current coverage criteria are process-oriented, and not product-oriented (this is a problem even for V&V of object-oriented programs!).

Adaptive systems are mainly used when there is no good model of the plant – thus it is hard to determine what to verify against. It was thought that it is often very difficult to specify requirements or acceptability criteria for complex

adaptive systems. One recurring theme is that adaptive systems often do not have a good way of telling when they are outside their range of expertise. It was suggested that other methods (such as putting "wrappers" around the neural net code) are needed to prevent such systems from trying to operate outside of their range.

Problems exist for V&V of both *off-line* and *on-line* adaptive systems. For the former, the idea is to train a system and *then* verify it. For the latter, the question is how to do verification when the system can evolve many times after it is deployed. It was agreed that V&V for on-line adaptive systems is much harder. We will discuss both, in turn.

4.3 Off-Line Adaptive Systems

It was generally agreed that the best current methods for V&V of off-line adaptive systems are blackbox testing and statistical techniques. While useful, these techniques are not very satisfying since they cannot make any guarantees about stability, coverage, etc. There is also the problem of trace-ability – when a bug is found, it is often difficult to "point a finger" at the part (or parts) of the adaptive system that is responsible. While analysis is possible, the non-linear nature of most adaptive systems makes formal analysis very difficult. Current approaches are either intractable, or make very strict assumptions about the form of the plant, which are typically not valid.

We discussed several interesting options. One is to try to prove weaker mathematical results than "standard" stability (e.g., plain stability rather than the stronger convergence results typically proved for linear systems).

A big problem for adaptive systems is the question of collecting representative data – how to sample and how to determine whether there are "holes" in the test data set. One suggestion was to analyze the learned functions to find partitions of the operating regions where the chosen actions are "similar" and then devise tests for those regions. This could enable guarantees of coverage for statistical testing. In general, the methodology might be iterative: One would train a system, analyze it to determine how to choose test data, re-train if it fails any of those tests, analyze again, etc.

Another option is to investigate learning techniques that may be more amenable to formal analysis. Neural nets are very widely used, but they are just one of a whole family of function approximators that can be learned. Different families of functions have different characteristics in terms of learnability, expressiveness, sensitivity to noise, etc. It may be worthwhile investigating whether there are classes of function approximators that are more easily analyzed, and hence could lead to formal guarantees of safety. For instance, one technique described at the workshop uses hyperplanes to approximate the functions of interest. A neural net constructed from a hierarchy of such structures may be both expressive, yet tractable enough to lend itself to rigorous analysis of its properties. Such analyses may also aid us in determining how to incorporate domain knowledge into building such function approximators.

4.4 On-Line Adaptive Systems

Three options were discussed for V&V of on-line adaptive systems:

1. Continually doing V&V as the system evolves (on-line V&V).
2. Verifying that the learning technique cannot move from "safe" areas. The idea here is to demonstrate some sort of monotonicity property – if the system starts out being safe (shown via off-line V&V), then prove that it cannot become unsafe.
3. Certifying classes of systems rather than single instances. The idea here is that if one could show that a particular structure of neural net is "safe", no matter what training data it receives, then one can have it adapt without worry.

In general, we agreed that this is a very hard problem, and that we did not even understand well what are the desired requirements for on-line adaptive systems. For instance, it is not clear how to specify the failure modes of the system in advance, and so it was clear that monitoring plays an important part in maintaining safety (although it was not clear exactly what that role is). It was suggested that we may need to restrict the types of learning to keep the system safe (e.g., not changing the weights of the neural net too rapidly). By explicitly modeling the adaptation process and the process of environmental change, we may be able to estimate the parameters needed to ensure that such safety conditions hold at all times.

The problem may even be unsolvable as stated: if things are changing rapidly, while it may be feasible to use on-line statistical techniques to detect when dramatic changes have occurred, it may not be possible to guarantee that the system remains safe at all times, since adaptation cannot be instantaneous. For instance, while adapting to hardware failures, the system might, for a short time, become unstable or unsafe. Is that acceptable, or not? We might want to require that the system reenters a safe state within T seconds, or that it adapts at a given speed. This harks back to the point that we do not really understand what the requirements are for on-line adaptive systems.

Finally, we briefly touched on the issue of adapting to slow degradation in the controlled system, as opposed to qualitative, topological changes in the plant (e.g., due to hardware failures). It was agreed that the latter is generally a much harder problem to deal with, both for adaptation and for V&V. For instance, it was suggested that if training occurs even during the performance phase, perhaps using decaying values of the learning parameters or simulated annealing, then the system could slowly adapt to such changes. However, there are well-known problems where neural nets can "forget" old responses, especially when they are not being trained with a statistically valid sampling of the input space. It was suggested that this is an area in which formal proofs could give us insight into some of the design issues for adaptive systems.

5 V&V of Complex Systems

5.1 Attendance and Scope

The group contained sixteen people, with a heavy bias towards Verification and Validation - only two NASA researchers from the autonomy and adaptive field were present. The original topic put forward for discussion was the V&V of systems with many interacting components, either within one location (e.g. layered control architectures) and among several locations (homogeneous or heterogeneous multi-agent systems).

5.2 Introduction

The discussion took an interesting turn before any meaningful progress could be made on the stated subject. Basically, a debate ensued on the merits of V&V in general, rather than just limiting it to complex systems. In the words of one participant, we spent the whole time justifying the use of formal methods to the two non-V&V participants of the group.

5.3 Issues

In this section some of the highlights of this discussion will be recounted, but the majority of the issues was presented in section 2 where we addressed general issues from the workshop. The discussion was in the form of questions being raised by the NASA researchers followed by intense discussions. The output of the session was a list of issues (17 in all) from which we list a selection below.

System Engineering problems are often addressed as Software Engineering "It is like addressing architectural problems that arise with the construction of a bridge as issues of how to engineer blocks of concrete" - Gerard Le Lann (workshop participant). We should try and learn from other engineering disciplines, especially in how, for example civil engineers, seem to learn far better from their mistakes than software engineers do.

V&V is not possible without a formal specification of requirements for the system under analysis. Stating formal requirements for autonomous and adaptive systems is hard, and as such, not something often done during system development at NASA.

Implementation should not be attempted without a provably correct design. Doing mathematical proofs are hard, even using a theorem prover, but is worth the effort for critical parts of a system. It was also interesting that comments made after the workshop seemed to indicate that many participants felt that proofs of correctness are being shunned in favor of automated error-finding techniques such as model checking and that this trend is worrying and should be addressed.

Formal methods must be customized to specific domains in order to get maximum benefit from exploiting domain specific information. Domain specific knowledge is one of the best ways to attack the scalability issue of formal methods.

Compositional Techniques. Just as systems are built up from smaller pieces one should use compositional reasoning to reduce the complexity of applying formal methods to a complete system. Unfortunately, it is often the case that only a global system property is to be verified and then it is very hard to decompose this property into ones to be shown for components. It is often easier to compose properties known to hold for local components that hopefully will imply the desired global property is valid.

Challenge Problem The members of the group felt that one of the most important issues was for NASA to provide examples of autonomous and/or adaptive systems that need to be verified in order for the V&V community to try out their numerous techniques. This would allow both communities to benefit: NASA will be in a better position to defend the use of complex systems to mission managers, and V&V techniques would be improved and evaluated with respect to new challenging problems from the autonomous/adaptive domain.

6 Conclusions

The feedback after the workshop was very positive, but most of all it was clear that the problem of V&V of Autonomous and Adaptive systems is a hard one to solve - especially given that it is even unclear whether V&V of "normal" systems can be done with any degree of success with current techniques. Given the importance of this field we believe this area will be a fruitful research field for some time to come. Interestingly, in a unrelated event (High Dependability Computing Consortium Kick-off workshop at NASA Ames 10-12 January 2000) one of the main observations made by the Formal Methods working group was that V&V of Autonomous and Adaptive systems is a long term problem with at least a 20 year horizon. This independent assessment of the field closely mirrors this workshop's view.

Acknowledgments The organizers of the workshop are very grateful to all the workshop participants whose comments helped to improve this report. Special thanks go to James Caldwell (University of Wyoming), for his general comments, and Rodger Knaus (Instant Recall, Inc), who provided very detailed feedback on V&V of adaptive systems. We would also like to thank Peggy Leising for her hard work to make this workshop a reality.