2019 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
SYSTEMS ENGINEERING TECHNICAL SESSION
AUGUST 13-15, 2019 - NOVI, MICHIGAN

M&S AS THE KEY ENABLER FOR AUTONOMY DEVELOPMENT, ACQUISITION AND TESTING

John Brabbs[1], Scott Lohrer[2], Paul Kwashnak[3], Paul Bounker[1], Mark Brudnak, Ph.D.[1]

[1]U.S. Army Combat Capabilities Development Command, Ground Vehicle Systems Center, Warren, MI
[2]U.S. Army PdM ALUGS, Aberdeen, MD
[3]U.S. Army Test and Evaluation Center, Aberdeen, MD

ABSTRACT

This paper describes the role of Modeling and Simulation (M&S) as a critical tool which must be necessarily used for the development, acquisition and testing of autonomous systems. To be used effectively key aspects of development, acquisition and testing must adapt and change to derive the maximum benefit from M&S. We describe how development, acquisition and testing should leverage and use M&S. We furthermore introduce and explain the idea of testable autonomy and conclude with a discussion of the qualities and requirements that M&S needs to have to effectively function in the role that we envision.

Citation: J. Brabbs, S. Lohrer, P. Kwashnak, P. Bounker, M. Brudnak, "M&S as the Key Enabler for Autonomy Development, Acquisition and Testing", In *Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium* (GVSETS), NDIA, Novi, MI, Aug. 13-15, 2019.

1. INTRODUCTION

Autonomy and Artificial Intelligence (AI) have been identified as critical military technologies by Under Secretary of Defense Research and Engineering (USDR&E), but the current DoD infrastructure and capabilities are inadequate to safely test and evaluate Autonomous Systems performance. As the Army transforms under the new Army Operating Concept, the establishment of the Army Future's Command (AFC) demands innovation and responsiveness of Research, Development, Test and Evaluation (RDT&E) elements to accelerate the adoption of cutting-edge technologies.

To achieve this vision, the U.S. Army must undertake a holistic approach to the development of autonomous and intelligent systems. This approach cannot consider acquisition, development, testing, and simulation independently, but must develop an approach by which each of these mutually support each other. To achieve this, transformation must take place on two levels: 1) organizational/programmatic, and 2) technical. Multiple organizations must closely coordinate to include the Project Manager (PM), the Original Equipment Manufacturer (OEM), U.S. Army Test and Evaluation Command (ATEC) and Combat Capabilities Development Center (CCDC). As most organizational interfaces are, these will be implemented through the formal written documents of acquisition namely requirements, contracts, standards, TEMPs and other written agreements. Each of these will need to change to enable this holistic approach. Requirements may need to add aspects regarding the testability of autonomy. The contracts may need to incorporate the delivery of software, the use of M&S, and the use of standards. Standards may be needed for architecture. On the technical side, development, simulation and testing all need to mutually support each other. Development needs to focus on testability and M&S integration. Simulation needs to focus on automation, high-volume, scenario/environment modeling and use of real-

world data. Testing needs to focus on safety, confirmation/validation of M&S, controlled experimentation, data collection and discovery of unforeseen cases.

The U.S. Army will need to leverage a continuous spectrum of test technologies to thoroughly assess machine perception, reasoning, and behavior and determine the robustness of AI algorithms in a broad range of scenarios across numerous applications and mission areas. Advanced modeling and simulation tools and techniques will accelerate safety certification for earlier technology demonstration and Warfighter use; expedite acquisition and fielding by reducing developmental test iterations in live environments; and achieve statistical confidence in safe autonomous behaviors.

Weaving M&S toolsets into installed system test facilities (ISTF) to stress physical hardware and actuators in a controlled environment will further characterize system response and behavior prior to open air range testing. Intelligent safety controls will provide critical oversight while conducting system level demonstrations and test in live environments. Each successive layer builds trust and confidence in expected behavior and performance for the Warfighter.

The Combat Capabilities Development Center Ground Vehicle System Center (CCDC GVSC), Project Manager Force Projection (PM FP), and the U.S. Army Test & Evaluation Command (ATEC) are collaborating to develop a cohesive test approach built on M&S capability as the foundation. The RDT&E community needs the ability to rapidly develop and test autonomous systems prior to production and fielding, to include investments in M&S laboratories supporting software in the loop (SIL) and hardware in the loop (HIL) configurations. The collective goal is test assurance to reliably support development and validation of emerging prototypes and designs throughout the acquisition life cycle.

Early developmental lab-based testing utilizing M&S ensures the readiness of project management offices for field-testing and provides an opportunity for the system developer to make necessary changes prior to formal Test and Evaluation (T&E), thus reducing program and schedule risk. To achieve success in the future, the test model will need to adapt to address AI/Machine Learning (ML) data set generation, highly parallel automated simulation solutions, rapid high-fidelity terrain generation for both geo-specific and geo-typical and shared standards between Government and Industry. Early identification of a robust M&S framework will improve agility and adaptability for next-generation capability needs.

This paper addresses these needs by outlining recommended best practices in the area of technology development, acquisition and testing. It then proceeds to describe the quality of "testability" for autonomy code and concludes with a discussion of what a future M&S environment built for M&S should look like.

2. TECHNOLOGY DEVELOPMENT

2.1. Architecture

Within the Army, ground vehicle robotics has been evolving over the past several decades. From a research and development (R&D) perspective, programs have focused on the advancement and integration of sensor technology, development of knowledge bases and world modeling, as well as the definition and development of single and multiple unmanned ground vehicle (UGV) behaviors. While initial R&D efforts focused primarily on autonomous mobility, today's programs are focused on the development of integrated autonomous applications combining mobility; Intelligence, Surveillance, and Reconnaissance (ISR); lethality; and communications in support of manned-unmanned teaming in a wide variety of structured and unstructured environments. From an acquisition perspective, emphasis has migrated from commercial off-the-shelf (COTS) procurement to the definition of programs of record (PoR) that center on Army requirements and key performance parameters (KPPs) such that ground vehicle robotics can be seamlessly integrated with our operational forces. Program Executive Office (PEO) and Program Manager (PM) activities have also concentrated on defining and evolving standards to support and procure interoperable systems as part of the overarching acquisition strategy. From a commercial and academic perspective, advancements toward self-driving automobiles as well as increased attention on efforts such as smart cities, have sparked expansion in hardware/software technologies, modeling and simulation, standards, infrastructure, testing, and certification within the unmanned vehicle domain.

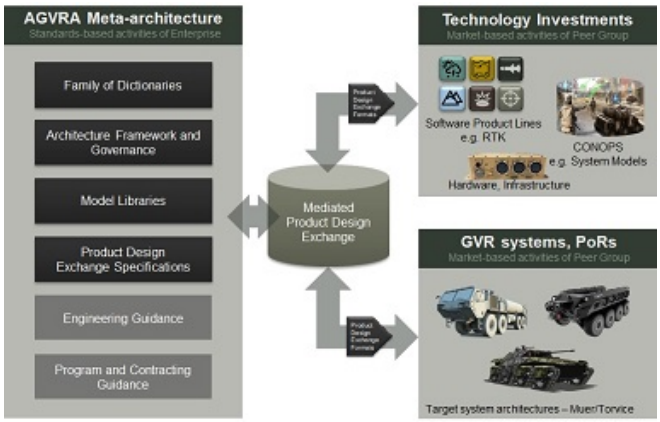

Figure 1: AGVRA ECOSYSTEM

The Army Robotic and Autonomous Systems (RAS) Strategy [1] describes how the Army will integrate new

technologies into future organizations to help ensure overmatch against increasingly capable enemies. The RAS Strategy focuses on five capability objectives to guide technology development of UGVs: increase situational awareness; lighten the Soldier's physical and cognitive workloads; sustain the Force with increased distribution, throughput, and efficiency; facilitate movement and maneuver; and protect the Force. Additionally, the RAS Strategy analyzes these capability objectives juxtaposed against near-, mid-, and far-term priorities as a means to identify and guide capability requirements and future investments.

The Autonomous Ground Vehicle Reference Architecture (AGVRA) [2] incorporates the term "Reference Architecture", its initial focus is to present and analyze current activities within the autonomous ground vehicle (AGV) domain, document a set of best practices, and at a high level, initiate an architectural definition based on the entirety of the domain, Figure 1. The architecture and architectural considerations in this document identify the AGVRA stakeholders, contexts, business models and processes, technical artifacts, and their relationships to provide guidance and contextualization to the set of AGVRA stakeholders (to include technologists, program managers, and policy makers). Ground robotics acquisition community has initiated efforts over the past several years to transition robotic systems procurement from the acquisition of commercial or specialized robotics systems to PoR where systems are developed or acquired to meet specific capabilities and KPPs. Additionally, the robotics acquisition community developed and governs a ground robotics interoperability profile (IOP) of the SAE Joint Architecture for Unmanned Systems (JAUS) to standardize the procurement and integration of robotic capabilities and platforms across their programs. Similarly, PEO Command, Control, Communications-Tactical (PEO C3T) has developed the Vehicular Integration for C4ISR/EW Interoperability (VICTORY) standard to govern the development and procurement of Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) mission equipment such that it can be integrated in a consistent manner within the ground vehicle community, to include ground vehicle robotic systems.

The acceleration of the vehicle autonomy market (commercial and Government) has led to continual rapid advances in platform, sensor, and safety technologies as well as autonomy architectures, behaviors, and software systems. The Robotic Operating System (ROS) is one such capability that has proliferated open source robotics software within the autonomy market and has been leveraged as a key technology specifically supporting the industrial and military communities. Government efforts such as the Robotics Collaborative Technology Alliance (RCTA) are defining key capabilities and architectures to support manned and unmanned teaming. These various architectures and initiatives represent the current state of AGV systems, which are being and will continue to be leveraged to support the Army RAS Strategy.

The U.S. Army Ground Vehicle Systems Center (GVSC) has developed this AGVRA as a means to better understand and inform the state of the domain and the manner in which these capabilities and architectures can be leveraged together or separately to support the RAS community. AGVRA encompasses all aspects of RAS ground systems to support individual and coordinated autonomous behaviors (for example, mobility, C4, lethality, survivability) within manned and unmanned teaming environments. To accomplish this, AGVRA considers the domain with respect to requirements, software, hardware, architectures, frameworks, systems of systems, and interfaces. Based on these considerations and analysis, the AGVRA provides a set of architectural guidelines to assist the RAS community in meeting the RAS Strategy objectives by referencing existing architectural and standards artifacts and identifying and recommending both business and technical best practices.[1]

In the same manner M&S must define standards, architectures and interfaces to get the community to build models, environments and tools that support the overall architecture, adhere to the standards and utilize the interfaces as defined. Having the community develop in this manner and where appropriate drive changes to the standards, architecture and interfaces is in the best interest of Army and autonomy development in general.

2.2. Standards

Standards are important and leveraged heavily in AGVRA as way to engage with the community of interest in Autonomy development and as way to leverage work that is being performed by other entities both inside and outside of the Department of Defense (DoD) development and procurement community.

M&S in the same manner as autonomy should leverage existing standards where practical and appropriate. Furthermore, for internal development where standards need to be enhanced or added to for a specific development effort, the effort should be taken to get the existing standard updated to reflect the changes needed for this development effort. This is important as the community will now be able to leverage this work and may also further enhance or add to this development work. This then could be beneficial in the future Army developments as the functional capabilities may be added to existing Army work in a way and manner that could

[1] Autonomous Ground Vehicle Reference Architecture (AGVRA) Concept Description Phase 0, 16 April, 2018

easily be adopted where applicable. This will provide the benefit of potentially reducing future programs cost and schedule.

2.3. Interfaces

The Department of Defense (DoD) has launched an initiative to identify and define interoperability standards to be organized and maintained within a Robotics and Autonomous Systems – Ground (RAS-G) Interoperability Profile (IOP). This IOP will be employed by Unmanned Ground Vehicle (UGV) acquisition managers in the acquisition of future Programs of Record, the upgrade of fielded systems, and the evaluation/acquisition of Commercial-Off-The-Shelf (COTS) products.

A primary goal of this initiative is to leverage existing and emerging standards within the Unmanned Vehicle (UxV) community such as the Society of Automotive Engineers (SAE) AS-4 Joint Architecture for Unmanned Systems (JAUS) standard and the Army Unmanned Aircraft Systems (UAS) Project Office IOPs.

- Facilitating interoperability among new UGV initiatives and legacy systems;
- Facilitating interoperability between controllers and UxV robotic system(s);
- Facilitating collaboration between UGV and UAS systems;
- Providing a path forward to standardized interoperable technology solutions
- Promoting payload and on-board subsystem modularity and commonality across the portfolio of UGV systems.

IOP is developed using a government/industry Working Integrated Product Team (WIPT) structure, and defines the interoperable interfaces and protocols necessary to enable interoperability and modularity to be introduced to the capabilities that have already been widely fielded. The exact set of capabilities addressed in IOP V3 is described in the RAS-G IOP V3 Capabilities Plan. The DoD intends to publish periodic revisions to the IOP in order to expand and evolve its scope as necessary, based on the evolution of Warfighter capability requirements and technological advances.[2]

The IOP is tested using the Conformance Verification Tool (CVT) which provides stimulus to subsystems defined in the IOP and verifies that the response is within the parameters defined within the IOP. This is provided to as aid to be utilized during internal development and testing of the IOP. M&S can be utilized in a similar manner as another tool to aid in development and testing. Systems and subsystems can be modeled in an environment and then stimulated using IOP commands from a controller much as they are done with finished systems. This provides the Army and developers a tool to validate that a system build to the specifications that are modeled in the virtual environment can perform a specific functional mission or task as required in a programs Performance Specification. To correctly validate the performance, the system must be modeled to the resolution needed for specific functional tasks and the environments interaction with the physics of the system must also be modeled to the same resolution needed for task validation. For example if your requirement is that your system detect objects 1 meter or larger diameter than you need to build objects at that diameter or larger.

3. ACQUISITION

Acquisition is changing. The need and desire to have systems fielded quickly is evidence by the Directed Requirements for two autonomous systems: SMET and Leader/Follower. These two programs are guinea pigs if you will to force the acquisition community and the responsible product office to derive a more streamlined process for transitioning technology and new systems to the soldier. The limited fielding of prototypes within 1-2 years for Operational Test Demonstrations (OTD) created a necessary uneasiness for these momentary troubles are achieving for us a larger goal that far outweighs them all. So we have had to fix our eyes on not what is familiar but embrace and explore the unfamiliar to meet the changing paradigm.

Leveraging new contracting mechanisms such as the OTA is one such mechanism used by both programs to quickly push contracts out and get the needed resources involved to support the programs. This quicker contracting is necessary in order to meet the limited fielding of the prototypes with the specified 1-2 year timelines. Because of these fielding timelines, a hard look at how the Army currently performs testing had to happen as there just isn't the time to perform all the previously required testing. One of the outcomes of this is the need to be able to rely on Modeling and Simulation to perform and inform some of this testing.

3.1. Requirements

As we begin to learn from these programs as to what worked well and what areas were bottlenecks or didn't work well we can derive new requirements for the contracts to hopefully create a more streamlined process and clarify expectations for the contractors.

As previously mentioned, Modeling and Simulation is recognized from the testing community at Aberdeen as a priority to implement. Efforts are underway to assess the gaps and to begin to develop and identify the necessary tools.

[2] Robotic Autonomous Systems – Ground Interoperability Profile Version 3 Overarching Document, March 2018

This potentially can impact how an autonomy developer develops their software. They may have to implement ways to control the speed at which their software runs for example or abstract layers in order to receive virtual inputs and provide access to internal variables or state data.

Data collection is another area that may require requirements to support. The Leader/Follower program for example recognized that they would have to forego much of what normally is part of Development Testing (DT) and rely on more of the operational testing. This again was necessary to achieve the required schedule. Going forward it could become a requirement that vendors collect and provide data that as part of their internal development to create an initial pedigree of data and share their process of development to create confidence in the safety of their solution.

Another potential requirement that may come from the Leader Follower program is the use of government owned software. The advantage here is that the product office and the testing community can be proactive instead of reactive. If the government has the software solution or at least a good portion of it, there is now history with that software and continual testing can achieve a trusted and harden product.

Requirements for the vendors to submit their software to the repository as well as supporting test data will benefit the timely fielding of new systems.

Automated testing or unit testing standards is another area to explore to gain confidence in a vendor solution as part of the repository.

As these new systems are reliant on technology and software to operate properly, a method of updating is yet another concern. Commercial autonomous vehicle companies have the luxury of leveraging large amounts of test data because of their larger fleets of vehicles and the use of cloud data storage. Consumers that purchase a Tesla vehicle for example can receive software and firmware updates from the Tesla Cloud. So as the company is continually testing and making updates the customers can also receive these benefits. We all have experienced how rapidly technology can change. The Army may have to address how a fleet of autonomous vehicles will receive software updates and look to some of these network enabled technologies can fit into a program.

4. TEST

Autonomy delivers an expansive test space, presenting challenges to evaluate deliberate autonomous behaviors over a wide range of conditions with sufficient fidelity and accuracy. Comprehensive M&S serves as a crucial enabler to address this intractably large test space to facilitate effective and affordable test and evaluation (T&E) of autonomy perception and comprehension. [3]

4.1. Digital Test Ecosystem

The T&E community requires a robust M&S test capability that ensures the autonomous system will operate safely and as intended. Current methods rely on observing developer testing or conducting a minimal number of live scenarios. This is insufficient to fully validate and "build trust" that the system will operate as intended.

In order to achieve a statistical confidence for safe operation, the autonomy software can be stressed in a virtual test environment in a wide array of scenarios to calculate the probability of anomalies by varying sensor inputs, weather conditions, obstacles, and degraded communications. The autonomy software is coupled with kinematic models of the system under test (SUT) and sensor models to populate and interact with the digital test environment, Figure 2. This environment allows for precisely controlled inputs to stimulate the decision engine and improved repeatability through known initial conditions.

By characterizing autonomous behavior, virtual testing aids in identifying economical test scenarios, understanding indicators of catastrophic or critical anomalies, and reducing risk of failure. Stakeholders are able to conduct a safety review based on the findings, identify issues that require corrective action, or recommend additional scenarios that need concentrated testing.

4.2. Integrated Virtual Environment

Open air ranges (OAR) are digitally constructed using data collected from Light Detection and Ranging (LIDAR) scans and photographic images to replicate test infrastructure and conditions such as existing Major Range Test Facility Bases (MRTFB). Digital assets are then integrated into the virtual courses to stimulate autonomy perception and decision making algorithms to observe and record resultant behavior.

The virtual environment is built for the scenarios that test the autonomy functions using the objectives, hazards and test constraints given the concept of operations for each system. Test scenarios are built in general accordance to existing test operations procedures for virtual and live test environments.

Prior to testing, a state space model of the autonomy must be defined to determine measured input-output data. This state space analysis provides inputs enabling predictive modeling tools to scan the state space and predict boundary conditions with higher risk of failure. The process results in delivery of targeted scenarios warranting closer examination. These inputs are utilized to comprehensively explore decision making logic by modifying a selection of test variables as part of the test selection methodology.

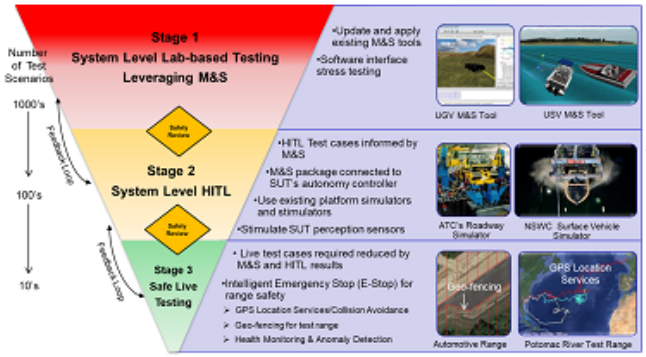

Figure 2: Conceptual operational view

A dynamic virtual scenario engine can be used to support mission development, execution, and playback functions to simulate the environment.

The test community will benefit from the creation of a simulation architecture standard to support a spectrum of autonomy testing from early developer tests to full-up system level testing. By implementing an enterprise framework, test agencies can define a system of interfaces to integrate key architectures via interface control documents (ICD) and data standards. The proliferation of service-oriented architectures (SOA) for robotic system design provide additional utility in exploring Modular Open System Architecture (MOSA) solutions.

4.3. Test Selection Methodology

Manual experimental design to determine applicable combinations of input variables to trigger emergent or unanticipated behavior would be resource intensive and time consuming. As dimensionality of the test space increases, intelligent sampling techniques are required to test thoroughly. To conduct an exhaustive test with statistical rigor, capabilities must be matured to systematically identify sequences of input conditions to characterize autonomous behavior.

Test case analysis would identify edge cases where the system may pass or fail over numerous iterations of a given test scenario. These results provide a selection of test cases where definitive outcomes are uncertain. For further assessment, these test cases can be transitioned to the live environment, thereby improving cost and schedule efficiency by focusing effort.

This analysis can be achieved by automating execution of simulations with the appropriate level of fidelity based on configured combinations of inputs. These variables can include environmental conditions, dynamic or static obstacles, sensor degradation, degraded communications, or other faults. A high volume of test simulations will be conducted to estimate probabilities of mission and safety failures using a defined scoring space.

High performance computing (HPC) and cluster enablement are key in accomplishing this simulation framework. Templated simulations and variables will be loaded and executed in parallel to reduce test time and using a batch mode to reduce operator involvement. This will provide the high throughput necessary to generate a significant number of test runs. As autonomy software matures and becomes time-managed, this process can be accelerated by implementing a faster than real-time framework, currently limited by sensor models, non-player character (NPC) generation, and the autonomy code which are not typically multi-threaded.

4.4. Data Analysis Framework

Data products provide the system developer and evaluators with information that can be used to reduce risks during research and development, developmental testing, operational testing, and during sustainment. The end result is the fielding of autonomous systems which have been characterized with a high degree of trust and will operate as intended in the battlefield.

Implementation of data acquisition tools will ensure data from all sources are read and interpreted consistently. Advancement instrumentation and software tools will provide capability to collect and aggregate large data sets resulting from sensor and autonomy outputs.

Once M&S testing is completed stakeholders will conduct a safety review using automated data intelligence to analyze the data generated. This analysis will generate the statistical data for safety risks, corrective action, and recommendations for additional testing. This provides critical information to stakeholders for test cases that would be unachievable or high risk using OARs.

Making this data framework available throughout the testing continuum supports system developers to adopt continuous integration and delivery of capability through increasingly automated testing and analysis. Evaluators can enact a model-test-fix approach as a cost-effective method to overcome test constraints.

4.5. Validation

Verification and validation (V&V) of the models and simulations are required to ensure the results are accurate and provide confidence to the test community. A verification and validation strategy that is modular and designed to ensure that all components are checked including the virtual world, environmental conditions, interfaces, and the simulations will need to be implemented.

As the decision engine is reliant upon input from the perception engines, there is no relative difference between the live and virtual abstractions to the autonomy engine. Therefore, test strategies are optimized by using Hardware in

the Loop (HITL) and live testing to validate the perception components and virtual test results rather than relying on physical components to explore the test space. [4]

Real-world testing will establish a feedback loop to validate the simulated events. This is facilitated by designing and integrating data inputs and formats to synchronize virtual and live test environments. As the models and simulations become more developed, early verification and validation of capabilities to include prototypes can be achieved in higher fidelity environments.

5. TESTABLE AUTONOMY

A key requirement for the development of autonomy software is that it be developed in such a way that it can easily be executed either on a platform or in a partially or wholly simulated environment. Furthermore, the autonomy system should have functions or capabilities which enable its evaluation in a testing environment whether in pure simulation, a SIL/HIL environment or live. Without an explicit testability requirement, the integration of autonomy code into a simulation system is an arduous process taking weeks if not months. If not explicitly designed in, integration boundaries between autonomy and other systems can be complex and highly coupled. Integration in these cases requires modification of the autonomy code, writing wrapper, marshalling data flowing into and out of the system and mocking certain aspects of the system.

This section describes a few aspects of testability as it relates to the development of autonomous or semi-autonomous systems. Ordinarily, "testability" is associated with hypotheses in the scientific method, in our case we will define testability for autonomous systems as the design quality of the autonomy system which enables integration with M&S and test environments and evaluation by such an environment. (In short, autonomy must be designed to be tested.) Some aspects of testability may be addressed by the architecture. For those that are not, the quality of testability should be included as a requirement, standard or specification. The following sections each begin with a statement as to why it is important and then discuss specific guidelines which enable testability in that particular domain.

5.1. Time Management

Depending on the type of M&S being employed, the nature of the evaluation may require that the simulation run faster than, in pace with, or slower than real time. Faster than real time may provide an opportunity to accelerate the exposure of the system to relevant miles, events, time, etc. because the relevant simulation does not require excessive processing. In-pace with real time is typical of M&S in a HIL/SIL environment where real hardware is integrated into the evaluation system. Slower than real time is often associated with highly detailed physics of a particular sensor or sensors which are too complex to execute in real time (e.g. ray tracing, environmental effects, multi-path radar). It is likely that all three scenarios may be used in an autonomy development program over its lifetime.

For a system to be testable with respect to time management, an autonomy system must not manage its own time. Time management may be implicit or explicit. Implicit time management is the reliance on particular events or phenomenon which regulate the execution of the code. This may be as simple as reliance on a particular routine or loop to take some amount of processing time or may include response to hardware events such as interrupts. Explicit time management is the use of a timer to regulate and pace the execution of code, usually on a periodic basis. Testable autonomy must not internally depend on implicit or explicit time management, but must be designed in such a way that time management is handled at the interface of the autonomy module. This aspect could be addressed in the architecture. If it is not, then time may be managed by the response to external events or by injecting a time management dependency into the autonomy module.

5.2. Modularity

It understood that software should be written in a modular way. For our purposes this section addresses modularity so as to allow the autonomy code to either drop into a platform or a HIL/SIL or an M&S environment. As discussed earlier, unless explicitly designed this way, the autonomy code may have internal dependencies on non-autonomy code. For example, some sensor data may be read internally. To make autonomy modular for the purposes of testability, the autonomy must be decoupled from the rest of the system. From the point of view of data flow, these interfaces should act with and use the same data structures in both live and M&S environments. For more tightly coupled interactions such as with services, these should be handled using existing modularization approaches such as dependency injection. The form of the interface can be flexible as long as it is the same on the platform and in M&S. It may be a binary API, an IPC based publish-subscribe or a service based architecture. The measure of success in this regard is that the autonomy code cannot internally detect whether it is running on a platform or in an M&S environment. This requirement may stand on its own or be part of a larger requirement or standard as part of the architecture.

5.3. Instrumentation

A key aspect of simulation and testing is the ability to measure to obtain an assessment of how a system performs in a particular circumstance. In data acquisition for ordinary vehicles, testers often record data being transmitted on an

internal data bus (i.e. CAN, IEEE 1553, etc.). In analogy, an autonomy system must provide an ability to be instrumented as well. In these cases, an autonomy system must provide a facility or interface for instrumentation of its internal state. Such an interface may be elective in that it is not always logged, but particular and important stages of the processing should be exposed for potential logging. Given that autonomy is internally broken into logical pieces or stages, it is important that each of the logical pieces or stages be equipped for logging of inputs, outputs and events. Minimally, this would include processing of raw sensor input, sensor fusion, world modeling and planning and control functions. Like the other aspects of testability, these functions may be specified as part of a standard or in the system architecture.

5.4. Target Hardware and OS

A platform and an M&S environment may run on different hardware. Platforms often make hardware decisions based on cost, performance, power consumption, packaging or some other metric. It is entirely likely that an autonomy hardware solution employ some mixture of general purpose CPU for overall management and high-level execution. These may draw from Intel or ARM or some other general purpose CPU vendor. They will also likely contain specialized hardware for highly complex computations. Depending on the application these could be DSPs, GPUs, FPGAs, ASICs, or AI co-processors. In M&S and SILs, simulation hardware normally consists of Intel x86 or x64 general purpose processor along with GPU facilities. They do not ordinarily contain other specialized embedded hardware. Testability requires that autonomy code run in the M&S environment, which may create challenges. To the degree possible, testability requires that general purpose code be portable to the x64 instruction set and be validated to execute there. For other specialized hardware, the ideal is that all code execute on the simulation environment's CPUs and GPUs. This may require hardware emulation and slower than real-time performance in the M&S environment.

In a similar way, the target OS may differ. On a platform, the autonomy code may be built against a real-time operating system such as VxWorks, QNX, RTOS, etc. In simulation the OS is likely Windows or a variant of Linux. In these cases the autonomy code should minimize dependence on the underlying OS and rely on external interfaces to provide data, events, timing and process/thread control and other OS related services.

We understand that these are all complex and difficult choices and that in some cases it may not be possible to port/emulate all functions to an x64 based simulation environment. Ultimately, these are architectural decisions and to reap the full benefit of M&S in autonomy development, these choices should be made with testability in mind.

6. SIMULATION QUALITIES

The Robotic and Autonomous Systems (RAS) simulation needs the ability to support developing RAS from concept through production. The RAS simulation architecture will be one of the keys to accelerating the development and fielding of RAS. One of the leading companies in self driving vehicle technology Waymo says "In simulation, we rigorously test any changes or updates to our software before they're deployed in our fleet. We identify the most challenging situations our vehicles have encountered on public roads, and turn them into virtual scenarios for our self-driving software to practice in simulation." [5] Waymo has backed this up over the last 10 years by driving over 7 billion miles in simulation [6]. The US Army is funding the development of the multiple simulation programs (e.g. Continuous Autonomy Simulation Test Laboratory Environment (CASTLE) figure 6, Autonomous System Test Capability (ASTC) …) in support of using simulation to augment live testing. These programs are laying the foundation as well as learning how simulation needs to be architected to support RAS.

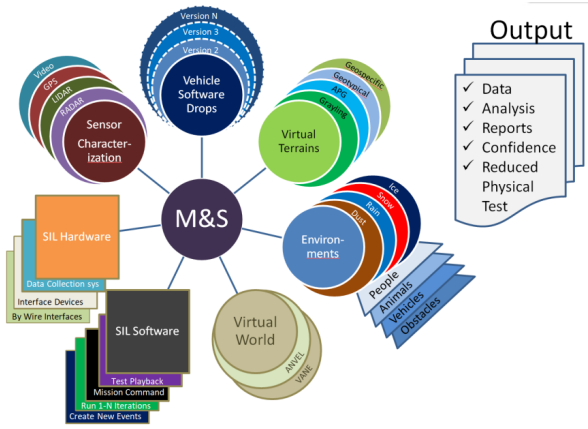

Figure 3: Example simulation architecture

6.1. Architecture

The simulation architecture as shown in Figure 3 needs to be able to support Software in the Loop (SWITL), Hardware in the Loop (HITL), Human in the Loop (HUMITL) and Vehicle in the Loop (VITL). The simulation architecture needs to be flexible enough to accommodate the different modes depending on how the autonomous vehicle (AV) software (SW) is being tested and evaluated. The simulation architecture needs to be developed to be modular so the software can be composed of a component and/or plugin system to allow for expanding, detracting and upgrading features that will be needed as the RAS gets more complex. The simulation architecture needs to be scalable so it can handle more than one vehicle, terrain environments from

small high fidelity to larger with less fidelity, numbers of independent actors for supporting vehicle traffic, pedestrians/crowds, obstacles, animals, combat scenarios with friendly and enemy forces to support creating realistic scenarios. The simulation architecture needs the ability to run slower than real time, real time and faster than real-time

The autonomous Modeling and Simulation architecture needs to provide the autonomous vehicle (AV) software (SW) with all the inputs and outputs (I/O) that the AV SW would receive if it was in the real world. The I/O will include the data from the Sensors models (e.g. LIDAR, RADAR, Cameras …), Localization simulation (e.g. GPS, IMU …), vehicle model (e.g. by-wire, wheel sensor..), terrain environments, communication models (e.g. Radio ..). The interfaces (e.g. Ethernet, CAN, Radio …) may use real hardware interfaces and/or be represented virtually. This can be done by having an extendable simulation architecture that allows the user to incorporate vehicle, sensor, controllers and communication models into a simulation that support the specific RAS.

The simulation architecture should incorporate the latest in gaming technology by utilizing a modern gaming engine like Unreal Engine or Unity. This allows the simulation architecture to take advantage of the extensive investment that the gaming engine companies are making into visualization and physics as new features and capabilities are being rolled out multiple times a year. A number of open source and industry AV simulation projects are already utilizing the gaming engines e.g. AirSim, CARLA, LGSVL Simulator, SynCity, Metamodo and Nvidia Constellation. Commercial AV simulation software are also using the gaming engines. By utilizing a gaming engine like Unreal Engine as part of the simulation architecture it allows developers and users to have access to a wealth of documentation and tutorials from the Epic and other developers; vehicle models, terrain assets and plugins from the Unreal market place at an affordable price; independent developers and video game studios to extend the capability and features of Unreal. Selecting a gaming engine like Unreal also allows an organization like the Ground Vehicle System Center (GVSC) to have a common platform for visualization across multiple simulation projects like Virtual Experiments, Manned Unmanned Teaming Simulation Framework and the AV M&S SILs so that model and terrain assets; scenarios; and interfaces can be shared between the different simulation projects to speed up the platform development.

The simulation architecture will also need the ability to rapidly generate terrain databases with models for buildings, signs, trees, bushes … that are geo-typical or geo-specific or have the ability to receive or purchase the terrains/models from a gaming market place. The terrains and models will need to provide the data necessary for the sensor models to perform correctly e.g. material properties.

The simulation architecture will need to be able to interface with the AV SW if it is running on the real vehicle computer whether in a HWIL or on the actual autonomous vehicle. The simulation architecture will also need to be able to support interfacing with the AV SW running on local computer hardware, a virtual machine, cloud infrastructure and/or high-performance computers (HPC).

The simulation architecture needs to know how the AV SW runs in the real world so that the simulation can support the infrastructure correctly e.g. performance on real hardware vs performance running in virtual machine running on a PC Workstation.

A key to the AV simulation architecture will be developing a Simulation/AV SW Interface Control Document (ICD) that defines the interface between the simulation and the AV SW for the sensors, localization, vehicle, communication and so on as shown in Figure 4. The Sim/AV ICD will help accelerate the implementation of the simulation with the AV SW as new releases are available by defining how changes in the simulation or AV SW are documented and communicated.

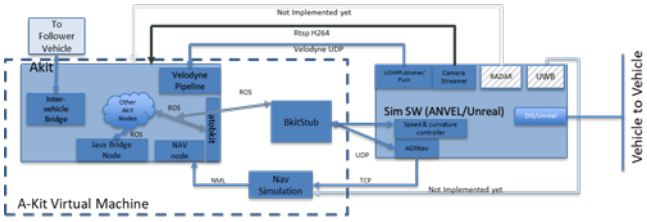

Figure 4: Leader Follower Autonomy Vehicle Software running with simulation software.

6.2. Drop-in Autonomy

The simulation architecture and the AV SW both need to be designed to support dropping the AV SW into a simulation so that the AV SW does not know whether input and outputs are being provided from the real world or from a virtual world or a combination of both depending on the SWIL, HWIL and/or VITL configuration that is being used.

This drop-in autonomy capability will support the rapidly testing of the AV SW during development, testing and production. The Simulation architecture using the Sim/AV ICD will provide a method for quickly evaluating multiple versions of the AV SW using simulation.

The Sim/AV SW ICD will again be important for this capability to work properly by making sure any changes on either the simulation SW or AV SW is identified and updated.

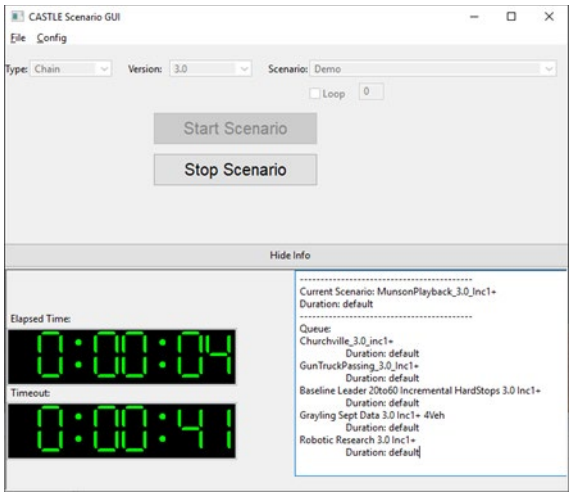

Figure 5: CASTLE NAP GUI

6.3. Automation

The simulation architecture needs to provide a way to run 100's and 1000's of scenarios using automation to evaluate edge cases, do regression testing, verify software updates provided via a new feature, address an issue or fix a problem. This can be done by developing a way to automate the tasks that a soldier or tester will need to do for fully exercising the capability and features of the RAS during test and evaluation. The Ground Vehicle System Center's (GVSC) Autonomous M&S SIL teams are developing and implementing the Network Automation Program (NAP) to support this capability. NAP provides the ability for the simulation architecture to launch, execute, stop and restart all the software that will be used during a simulation (e.g. simulation SW, the AV SW …) in its desired configuration for each scenario, so that test after test can be executed without a person or soldier needed, Figure 5 shows an example of the CASTLE NAP Graphical User Interface (GUI). Each scenario will be designed to support testing a capability or feature of the RAS. The test engineer will be able to develop scripts that support the desired configuration to be tested and evaluated. An example of this could be the desire to test the gap distance of a Leader Follower (LF) Convoy where the tester wants to determine how the LF convoy performs using different gap distances at different speeds on different courses. The scripts will allow the tester to easily change one parameter at a time or change all the parameters to run 100's or 1000's of scenarios testing all the different combinations or only the edge cases.


Figure 6: CASTLE

6.4. Data Collection & Analysis

The simulation architecture will also need the ability to monitor and collect data from both the virtual world and from the AV software in real-time. The ability to watch the simulation and AV SW in real-time is needed since there is no human tester to determine if the simulation and/or AV SW is running into an issue. The real-time monitoring will allow automation SW like NAP to stop and/or restart a scenario if it determines there is an issue as well as save the data collected during a scenario run. The saved data can then be shared with either the autonomy and/or simulation developers to troubleshoot the issue. An example of this capability is being developed as part of the CASTLE project. CASTLE is currently implementing the ability to have both a Simulation listener and a ROS listener these listeners allow the automaton software e.g. NAP to monitor and watch the state of both the simulation and the AV SW to determine how the simulation and RAS are performing; whether the test was successful or failed; and to collect all the necessary data and save it.

After the 100s to 1000s of scenario tests are completed the data will need to be analyzed to determine whether the test was successful, unknown or a failure. Once the data has been analyzed a report will need to be generated that contain information on whether the test was successful or a failure along with additional information that explains why. The US Army ATEC has an ADMAS system that is currently being used in support of AGR/ExLF live vehicle development and testing as the black box data collection system. The data is then sent back to ATEC for analysis. CASTLE is using NAP to automate the monitoring and data collection. The ASTC project is looking to expand on the current ADMAS and data collection analysis capability. These three efforts can help shape how automated real-time monitoring, data collection and analysis can support autonomous vehicle M&S efforts.

7. Conclusion

This paper we described the role of Modeling and Simulation (M&S) as a critical tool which must be necessarily used for the development, acquisition and testing of

autonomous systems. We furthermore outlined key aspects of development, acquisition and testing that must adapt and change to derive the maximum benefit from M&S. We described how development, acquisition and testing should leverage and use M&S. We furthermore introduced and explained the idea of testable autonomy and concluded with a discussion of the qualities and requirements that M&S needs to have to effectively function in the role that we envisioned.

8. References

[1] US Army TRADOC, "The US Army Robotic and Autonomous Systems Strategy," March 2017. [Online]. Available: https://www.tradoc.army.mil/Portals/14/Documents/RAS_Strategy.pdf.

[2] C. Cheung, "Autonomous Ground Vehicle Reference Architecture," in *2018 GVSETS Conference*, Novi, MI, 2018.

[3] Georgia Tech Research Institute, "Autonomy Test and Evaluation Infrastructure Gap Identification Report," Test Resource Management Center, Washington, DC, 2017.

[4] Department of Defense, "FY 2018-FY 2028 Strategic Plan for DoD T&E Resources," Washington, DC, 2017.

[5] Waymo, "Waymo Safety Report - On the Road to Fully Self-Driving," 2018. [Online]. Available: https://storage.googleapis.com/sdc-prod/v1/safety-report/Safety%20Report%202018.pdf. [Accessed 23 May 2019].

[6] D. Hersman, "Safety at Waymo | Self-Driving cars & other road users," Medium, 1 May 2019. [Online]. Available: https://medium.com/waymo/safety-at-waymo-self-driving-cars-other-road-users-d3b33e57e994. [Accessed 23 May 2019].