Autonomic Plan Monitoring

Braulio Coronado, Dr. Douglas S. Lange
Space and Naval Warfare Systems Center - Pacific
San Diego, California
UNITED STATES OF AMERICA

{braulio.coronado, doug.lange}@navy.mil

ABSTRACT

Autonomous agents have their own plans and monitoring their plans is among the essential tasks of command and control. As we move towards forces of large numbers of autonomous entities, we must decide at what level of detail we will monitor and alter plans. Part of that decision will depend on how well we can provide monitoring tools to the human-autonomy team conducting command and control (C2). In the Intelligent Multi-UxV Planner with Adaptive Collaborative Control Technologies (IMPACT) project, we employed an autonomics framework to manage the plan monitoring task. Fundamental to this approach was the realization that plans could be represented as networks, which are the target of the framework's capabilities. Useful to human-autonomy teaming, the strategies employed to respond to performance issues within a network form abstractions of the situation that can be acted upon or communicated to human operators.

1.0 INTRODUCTION

Future C2 operations will involve large numbers of heterogenous autonomous entities executing detailed mission plans towards mission objectives. As battlespace complexity increases, it will become increasingly difficult for the human-autonomy team conducting C2 operations to track the progress and performance of entities executing plans. A human C2 operator will have high-level supervisory tasks to perform including the monitoring of entity progress. Depending on the operational situation, the human will need to prioritize tasks through traditional manual means or with the help of task-managing automated assistants [1]. The cognitive load on the human will be increased as larger numbers of autonomous entities will need their performance monitored which will draw attention away from other mission critical operations.

Decision-making in C2 is the selection of an action aimed at improving the tactical situation, and is achieved by the human through past experience, judgment, and intuition; or by autonomous Intelligent Agents (IA) through automated reasoning. In both cases, decision-making is performed based on information obtained through tools available to the human-autonomy team. Optimal Situational Awareness (SA) is critical for effective decision-making, as the human-autonomy team can only make decisions to address issues they are aware of and understand.

In this paper we describe our approach to improve SA and decision-making by utilizing autonomic computing techniques to track the performance of autonomous entities executing mission plans. An enabling factor towards this approach is the selection of an autonomics framework that can model mission plans as networks. By modeling plans as networks, we can provide continuous plan quality evaluation for all performing autonomous entities. In addition to evaluating mission plans, we can monitor changes in the environment outside of mission plans to offer courses of action. The goal is to enhance the capabilities of the human-autonomy team by providing continuous feedback on the current tactical posture, and providing mission plan corrections as required by the operational situation.

2.0 BACKGROUND

Highly automated unmanned systems have shifted the role of the human conducting C2 from platform-level control to supervisory control of teams of unmanned systems. Advances in physical systems, their capabilities, and onboard automation have abstracted manual control of vehicle sensors and actuators to high-level commands that human supervisors can issue remotely. Research conducted in the Intelligent Multi-UxV Planner with Adaptive Collaborative Control Technologies (IMPACT) prototype testbed explores the use of a "playbook" control paradigm approach towards C2 aimed at inverting the human to unmanned system staffing ratio [2]. An IMPACT operator can call a "play", a high-level control abstraction, that encapsulates the constraints and behaviors in a mission plan for unmanned systems to perform. Supervisory control of large numbers of unmanned systems will require increased automation to help manage complexity in the C2 environment. Control of multiple unmanned systems can be represented by hierarchical control loops [3].

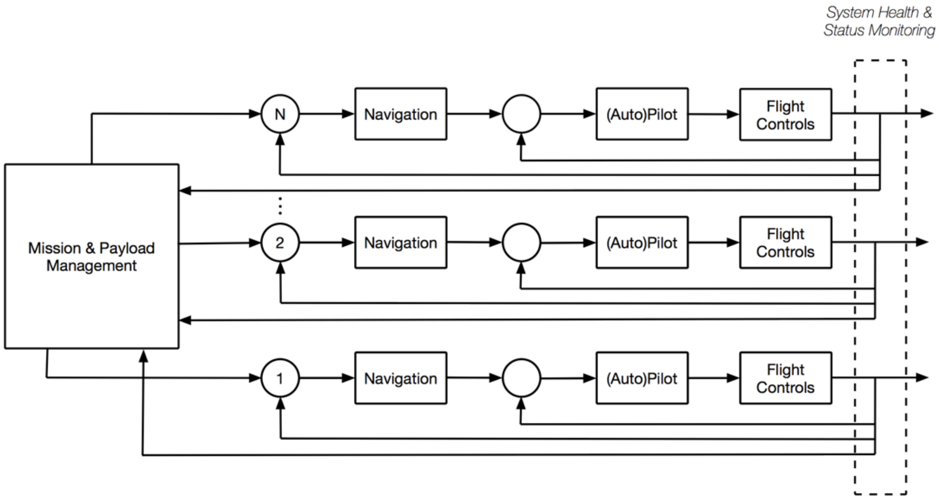

Figure 1: Hierarchical Control for Multiple Unmanned Vehicles. (Adapted from [3])

In Figure 1, we observe a hierarchy of control loops towards C2 of multiple unmanned systems. The Mission & Payload Management loop is shared across all systems, and is the control loop that orchestrates plans and tasks towards mission objectives. The Navigation control loop concerns route waypoints and tasking which set the constraints that define a successful mission. The Piloting and Flight Controls control loop manages the physical system's motion through control of actuators. Finally, the System Health & Monitoring control loop ensures the overall system operates normally. This control loop is a dashed line because it represents a highly intermittent loop from the human's perspective. The human will interact with this control loop the least, as the other loops take priority. In general, the outermost control loops will require the most attention by the human, as they require human experience, judgment, and intuition. The inner control loops concern more technical details of physical system capabilities, and are increasingly amenable to control by automation.

While monitoring of system health and status can be done by the human, it is an opportunity to investigate how the autonomy can provide this functionality. Our plan monitoring agent, the Plan Monitor, has been implemented in the IMPACT testbed to help manage the monitoring of unmanned system plans.

3.0 THE RAINBOW AUTONOMICS FRAMEWORK

A major problem with many autonomics approaches is they are tied to particular system implementations making scalability an issue for complex systems. For example, being tied to a particular architecture means a lack of flexibility to express new additions that may have not been designed for. Rainbow, the method in this paper, addresses this problem through abstraction of architecture into a formal architecture description language, Acme [4]. The Acme language allows Rainbow to express any system that can be expressed formally.

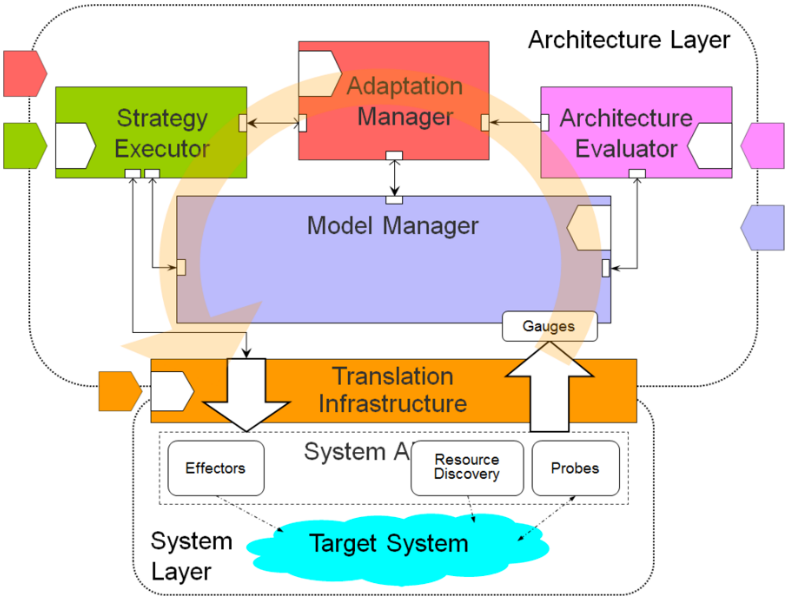

Figure 2: The Rainbow autonomics framework. (From [5], used with author's permission)

An Architecture Evaluator analyzes the model to determine violations of constraints that were defined for the target system. If constraints are violated, then an Adaptation Manager is triggered, which evaluates various strategies defined in a logic-based language (Stitch [6]) to determine the best approach to solving the problem that triggered the need for adaptation. The Adaptation Manager sends the chosen strategy to the Strategy Executor that is tasked with programmatically carrying out the strategy and effecting changes to the target system. The Rainbow framework interacts with the system through Probes, which send data to Rainbow, Gauges, which receive data from Probes to manipulate the model, and Effectors, which are actionable items that cause changes on the target system [5].

4.0 IMPLEMENTATION IN IMPACT

IMPACT allows for a human-autonomy team to supervise multiple heterogenous UxVs through high level goal-oriented plays, which Intelligent Agents (IA) translate into plans to achieve mission goals [2]. By modeling these plans as networks, we can use Rainbow's evaluation and adaptation tools and languages to effect plans and the systems that are executing them. To utilize Rainbow, we must supply certain software:

- Probes into the tactical data.
- Gauges that aggregate tactical data into useful information for the evaluation of performance.
- Translation of plans into network models.
- Strategies for repairing plans or communicating critical data to the operator.
- Effectors for executing strategies.

Among the most interesting pieces are the strategies. Strategies can range from simply notifying the operator to autonomously re-planning a mission. When notifying an operator, the very fact that a particular strategy was selected is a useful abstraction of the tactical situation. Rather than providing details about movement, the strategy is essentially communicating the final classification of the situation by the system. That the monitor may be permitted to autonomously make changes provides the other end of the full spectrum of contributions that the monitor makes as a teammate of the human operator.

4.1 Probes

An IMPACT instantiation features software modules communicating through a central messaging hub [2]. Data flowing through the hub composes the environment where the human autonomy team conduct C2. Entity types such as vehicle states, plans, tasks, and areas of interest are published to the hub and are used by IAs to handle route planning, resource allocation, and task management [7]. In order for the plan monitor to be an effective teammate, it needs information about the environment; from single events to data streams. Rainbow probes provide a means to sense the environment and feed information to gauges for modeling. In the plan monitor, we employ the ZeroMQ distributed messaging framework within probes to subscribe to messages that can help model the environment. Probes and the messages they subscribe to are:

- Vehicle Probe – Air, Ground, Surface and Track vehicle state messages. Includes vehicle telemetry and is generally frequent with multiple states per vehicle per second.
- Task Probe – Task-related messages which are important when monitoring plan quality for vehicles performing a task as part of a plan.
- Geo Probe – Messages that describe a location in the environment. Areas of interest, lines of interest, points of interest, and restricted zones can potentially be a target or obstacle in a plan and thus important for measuring performance.
- Plan Probe – Messages related to IMPACT play calls. Play automation requests and responses include critical information such as route waypoints that the plan monitor ingests to establish baseline plan quality. This probe will also process messages announcing play activation, cancellation, pause, resume, and completion.
- Critical Area Probe – Messages with information relating to vehicle ability to reach a critical area in the environment. For IMPACT scenarios, a critical area may be a flightline that vehicles must be able to access within a set timeframe.
- Policy Probe – Messages reported by policy-enforcing agents. Policies such as vehicle proximity to no-fly zones discovered by a policy-checker can potentially influence plan performance.

With the exception of the vehicle and critical area, there is little processing performed by probes. In most cases, probes will simply forward relevant information to interested gauges.

4.1.1 Vehicle Probe Additional Processing

Vehicle state updates are reported several times a second. For the plan monitoring task, updates of this frequency are not necessary for two reasons. One, the turnaround time from a probe report to an effector triggered by information from that report is analyzed through several independent modules in the autonomics framework and generally takes longer than one second. Two, telemetry data is incremental; that is, in most cases there is a small difference between a vehicle state and its next state. To that end, the vehicle probe records the latest vehicle states and forwards only the latest information to gauges every one second. This drastically reduces processing of extra data that likely does not offer much towards effective plan execution monitoring. Furthermore, this probe performs additional processing of vehicle speed in order to better handle telemetry of real vehicles which is further explained in a later section.

4.1.2 Critical Area Probe Additional Processing

Mission requirements may dictate coverage of certain areas of the environment by vehicles; not necessarily live coverage but the ability for a vehicle to be able to reach the area within a certain time. Since the operator can be managing multiple UxVs, their routes, and other tasks simultaneously, maintaining sufficient SA to keep this mission requirement in mind can be a challenge. This is especially true since there are few cues for the operator to recognize that at any point in time, at least one vehicle is able to reach the area quickly. The critical area probe offers to help the operator by handling the tracking of this requirement. This probe is a special case because it is the only active probe used by the plan monitor. As opposed to passively collecting data through subscriptions, the critical area probe actively queries the IMPACT system. This is done by publishing a request to the messaging hub with information on vehicles and an intended target. The route planning IA ingests these requests, calculates a route for each requested vehicle to the target and responds with estimated times for each vehicle to arrive at the destination. The probe then forwards the information to gauges. We set the period of critical area probe requests to one-minute intervals in order to reduce the traffic we generate in the hub and to avoid overburdening the route planner with requests for vehicle timings.

4.2 Gauges and Network Models

The primary function of gauges is to aggregate probe reports into useful data and translate this data into Acme elements to be analyzed by the Architecture Evaluator. Challenges towards this goal are:

- Translating IMPACT entity types into Acme models.
- Deciding the level of model detail necessary for effective monitoring.
- Representing IMPACT plans as networks.

In our previous work [8], we outlined the technical details for automatic Acme element generation through the use of software introspection. Briefly, native system data models represented in a type-introspection capable language, Java in the case of IMPACT, are mapped to abstract Acme element types. At runtime, concrete Acme elements are instantiated per unique data model instance. Data model properties, e.g. runtime Java object field values, are converted to Acme element properties to keep Acme models up-to-date with corresponding data models. The motivation for this model discovery mechanism is to free the system designer from the burden of explicitly declaring detailed model templates at design time; and because it is not always known which properties will be useful at some future time. Most gauges utilize this approach to generate Acme components from data gathered through probe reports. In most cases, components map one-to-one with native data models.

However, the plan monitor uses several Acme software constructs [4] towards the plan monitoring task:

- Components – The primary computation elements.
- Connectors – Interactions between components.
- Groups – Collections of the above.

To represent mission plans as networks, vehicle components are linked, though connectors, to other elements in the plan such as other vehicles, routes, and associated tasks. A vehicle connected to another vehicle matters when considering multi-vehicle plans. A plan can be thought of as a collection of vehicles, routes, and tasks encompassing the plan’s network. Therefore, plans can be represented as Acme Groups of connected vehicle, waypoint, and task components. The Acme element generation discussed above converts independent data models such as a vehicle, a task, or an area but does not automatically convert complex data models such as plans. For that, a special plan gauge has been developed to handle plan related messages.

4.2.1 Plan Gauge

The plan gauge ingests plan-related probe reports with information on new plans, or modifications to currently executing plans. Probe reports on new plans include vehicle, route, and task information. Route waypoint metadata includes the expected vehicle speed and any tasks the vehicle performs while on the waypoint. Upon receipt of a new plan, the plan gauge first calculates plan quality baseline parameters up front, effectively caching the expected performance of vehicles to be compared with actual values during plan execution. Waypoint metadata is augmented with these calculated values:

- Distance to next waypoint – Great circle distance of a waypoint’s location to the next waypoint’s location.
- Expected time in transit to next waypoint – Distance to next waypoint divided by the waypoint’s expected vehicle speed.
- Distance to task – Aggregate distance to next waypoint from the first waypoint through waypoints without associated tasks.
- Expected time in transit to task – Aggregate expected time in transit to next waypoint from the first waypoint through waypoints without associated tasks.

Assumptions are that a route leg is a straight line and any obstacles have been considered at plan instantiation, and in the latter two cases that a waypoint without associated tasks is in transit. Next, the plan gauge instantiates an Acme group element with placeholder properties for plan quality to be modified during plan execution. Associated vehicle, route and task Acme components are added to this Acme group. Finally, Acme connectors are created to link vehicle components with vehicle, task and route components in the group. At this point, the plan is essentially modeled as a network in Rainbow with components representing nodes in the network, and connectors representing edges. Grouping plan elements in such a network allows for a simple query of connected elements while performing plan quality evaluation, as opposed to fetching the desired elements from the entire model through a look-up or filter. This technique has the potential to help scale the plan monitoring task as we move towards larger forces of autonomous vehicles.

Probe reports on currently executing plans direct the plan gauge to modify plan models in several ways. An IMPACT operator is able to pause, resume, or cancel plays while the IMPACT system is able to activate or mark plans as completed. The plan gauge responds to these reports as follows:

- Pause – Marks a plan as inactive in the model. Halts processing of plan quality.
- Resume – Marks a plan as active in the model. Resumes processing of plan quality.
- Activate – Marks a plan as active in the model. Initiates processing of plan quality. In this case, a plan has just been created with the default setting of inactive until this message is received.
- Cancel and Complete –
  - Marks plan as inactive to immediately remove the plan from evaluation logic.
  - Deletes Acme connectors generated through the creation of this plan.
  - Deletes task and route Acme components. Vehicle components are not deleted.
  - Deletes the Acme group representing the plan.

We model the environment of the IMPACT system at runtime by maintaining network models of plans and modifying them as needed. This enables us to evaluate the quality of all plans as the scenario and tactical situation evolve over time.

4.3 Plan Quality

When an IMPACT play is called, the operator is essentially directing a vehicle to perform a task at a location to achieve some goal. The plan monitor in this case needs to consider what is being monitored. We can think of monitoring a plan as having two phases: route monitoring and task monitoring. Route monitoring considers a vehicle's real-time performance while traveling towards its destination. E.g. Will the vehicle arrive on time? Is it moving slower than expected? Task monitoring, on the other hand, considers how well the vehicle is performing its task once it has arrived at its destination. When performing plan monitoring, all plans will experience a transition from route monitoring to task monitoring. In the case of multi-vehicle plans, each performing vehicle will transition at a different point along the execution of the plan.

A consideration for plan monitoring, specifically through autonomics, is how and when to communicate plan quality to the operator. The autonomic control loop responds with adaptations when design time constraints are violated. Since a primary goal of the plan monitor is to evaluate the quality of all plans and communicate quality to the operator, we need to consider the case when a plan is in a good state. It would be a misuse of the autonomics control loop to trigger adaptation strategies to communicate a good plan state; since a good plan state does not violate constraints and the system is not in need of adaptation. However, a plan quality report with nominal quality is still useful to the operator's SA so we do not want to restrict plan quality reports to only those in bad states. To that end, the plan monitor delegates plan quality evaluation and communication to a helper module that exists outside the autonomic control loop. This helper module is a delegate of the plan gauge and as delegate has access to gauge functionality such as the ability to access and modify model properties. Normally, a gauge will act on information from a probe report but in the case of plan quality, we process all plans continuously rather than waiting for probe reports. If a plan is in a bad state, the delegate will modify the properties that will be noticed by the Architecture Evaluator to later trigger adaptation for the plan. During plan execution, the delegate queries the Rainbow model every second to processes the quality of all active plans. Quality parameters we communicate to the operator depends on the stage in the plan's execution; that is, whether we perform route monitoring or task monitoring.

Figure 3: Plan quality for two mission plans communicated to the IMPACT operator.

Quality parameters always include speed and fuel, as these are universal to all plans and need to be monitored at all times; including when a vehicle is not actively executing a mission plan. Since there may be multiple ongoing plans, and each plan may be in a different stage in its execution, the plan monitor process each plan sequentially through the following:

1. Evaluate speed quality parameter.
2. Evaluate fuel quality parameter.
3. Determine if vehicle is currently in transit or on task.
    a. If vehicle is in transit, evaluate expected time to execution (ETE) quality parameter.
    b. If vehicle is on task, evaluate task quality parameter.

We can determine if a vehicle is in transit by comparing the vehicle’s currentWaypointID property to the connected route component’s metadata. If the waypoint is not associated to a task, the vehicle is currently in transit. Otherwise, the vehicle has arrived and is performing a task. To calculate ETE, the plan monitor compares the expected ETE, which was cached by the Plan Gauge at plan instantiation (section 4.2.1), with actual ETE. If the vehicle is projected to arrive outside of a configurable threshold, then the ETE quality parameter will be of low quality. ETE thresholds can be configured at design time as a constraint in the model. The task quality parameter will depend on the type of task being performed. In general, all tasks will have meta-data that can be used to establish task quality. Some tasks like a point inspect task specify a location for surveillance, while others such as an escort task will have a requirement to maintain a minimum distance [7]. Figure 3 shows how plan quality is represented to the operator. The active play manager features a simple color-coded (Green, Yellow, Red) indicator that provides continuous feedback on the quality of the play. The active play manager tile is always available on the screen, and the operator only needs to glance at the queue of plays to determine if their plans are being performed as expected. This approach enhances SA as plan quality for all plans is summarized into a single metric that can be reviewed at any time. If more detail on the performance of a play is necessary, the operator can click the play in the active play manager to open a plan quality tile (Figure 3). This tile will display Speed, Fuel and ETE or Task quality parameter values. In the case of multi-vehicle plays, the least performing vehicle quality parameters are shown.

4.4 Strategies and Effectors

Automated adaptation towards improving the state of the system is the goal in autonomic computing [9]. As an instantiation of an autonomics framework, the most interesting part of the plan monitor is the adaptation. Strategies are the method by which the plan monitor offers utility as a team member in the human-autonomy team. By sensing changes in the environment through probes and gauges, and establishing constraints in the model of the world at design time, the plan monitor can offer varying levels of automation. Strategies range from simply notifying the operator of an event, to autonomously re-planning a mission if permitted. Effectors in the plan monitor publish to the messaging hub as it is the primary way of communicating within the IMPACT system.

4.4.1 Notification Strategies

An IMPACT operator will receive aural feedback on actions and events while performing C2. The IMPACT system will announce information on play-related events such as starting a new play, or the state of the vehicles performing background behaviors. Similarly, the IMPACT system will display a text message at the bottom of the screen to draw attention from the operator. The plan monitor can communicate information to the operator through these methods. Notifications by the plan monitor are as follows:

- Fuel – The plan monitor will track fuel levels for all vehicles at all times. Thresholds set in the model represent two fuel levels: warning and critical. A message will be published to the hub once per threshold breach announcing fuel level and vehicle call sign. For example: “Fuel State Warning: Devonian12 has 32% fuel remaining.”
- Flightline – The critical area probe (section 4.1.2) tracks vehicle times to arrive at a specified area such as a flightline. Constraints in the model specify thresholds per vehicle type. For example, air vehicle thresholds can be configured to be 120 seconds from flightline, while ground vehicles can be configured to 360 seconds. As long as there is one vehicle able to reach the flightline within the specified time, the constraint is satisfied. If there are no vehicles satisfying the constraint, the following message is displayed to the operator: “Flightline warning: There are no vehicles in range of the flightline.”.
- Restricted Area – IMPACT provides tools to the operator to designate restricted areas. By drawing a restricted area on the map through the user interface, an entity is generated and its details published to the messaging hub. The plan monitor is aware of restricted areas through the geo probe (section 4.1). Restricted areas are checked for vehicle presence every one second. If the presence of a vehicle is detected, the plan monitor will announce that information to the operator. For example: “Restricted Area Warning: Devonian12 violates area No Fly 2.” This vehicle area pair is tracked for the duration of the violation and a notification is published when the vehicle exits the area; e.g. “Restricted Area Exit: Devonian12 no longer violates area No Fly 2.”.

In a complex IMPACT environment where the operator experiences a heavy workload, notifications can help to maintain SA. Monitoring fuel, flightline and restricted area violations manually through visual checks would require a cognitive load that may not be available depending on the operational situation.

4.4.2 Actionable Strategies

By communicating with other IAs in IMPACT, the plan monitor can execute strategies that cause changes in the environment. The policy probe (section 4.1) monitors messages issued by the Configurable Operating Model Policy Automation for Control of Tasks (COMPACT) agent that offers compliance checking of policies to IMPACT [10]. COMPACT provides monitoring of vehicle communication ranges and is able to issue a policy summary when a vehicle is projected to lose communications. COMPACT has knowledge of restricted areas that may not exist in the IMPACT environment, and will communicate policy summaries when vehicles are projected to enter these restricted areas. The plan monitor can then introduce information on new restricted areas from COMPACT into the IMPACT system. Through the task manager [11], the plan monitor can initiate the calling of a new play, or re-routing of an existing play to address policy violations. The task manager, if configured to allow human-on-the-loop principles through working agreements [12], will execute the changes autonomously.

- Call Play – Upon detection of a communications policy violation, a strategy will initiate a request to call a new Communications Relay play. Meta-data in the policy violation specifies the affected vehicle's callsign. The strategy will look-up the required play calling data in the model, such as the vehicle's ID to compose a message published to the hub. The route-planner and resource-allocation IAs will select a vehicle to act as relay, if one is available, and the play will be executed without human intervention.
- Introduce restricted areas – A strategy to address a restricted area policy summary will initiate a dialogue with the COMPACT system. The initial policy summary will describe the vehicle and name of the restricted area but will omit details such as the coordinates, shape and size of the area. The strategy will request area details from COMPACT to compose a message that will generate the area and introduce this new knowledge into the IMPACT system. IAs listening on the hub become aware of this new area and will take it into consideration when performing route-planning or resource-allocation.
- Re-route play – Upon receipt of a restricted area policy summary, and generating the area in the IMPACT system through the strategy above. The next step is to re-route the affected vehicle's plan. This is done by a strategy through a simple "ReroutePlay" message with information on the vehicle and play ID. The route-planning IA will adjust the vehicle's route as necessary to avoid the area.

Human-in-the-loop principles can also be applied to these three strategies through working agreement configuration. Instead of the effect happening autonomously, the task manager will add a supervisory task in the task list for the human operator to perform manually.

5.0 REAL WORLD TELEMETRY

The Maritime Autonomous PLatform Exploitation framework (MAPLE) is one of the agents within the IMPACT system [10]. MAPLE is a Ground Control Station (GCS) that allows IMPACT operators control of real unmanned systems. Real world telemetry of actual vehicles is injected into an IMPACT instantiation to practice supervisory control of a combination of real and simulated vehicles. The use of real data introduces unique issues not found while working solely with simulated vehicles. The nature of the physical systems used to report telemetry can involve a level of signal noise. In particular, vehicle speed is a fundamental quality parameter for the plan monitor because it affects other quality parameters such as ETE. To that end, we have implemented an exponential moving average (EMA) algorithm to smooth out vehicle speeds. The vehicle probe (section 4.1.1) tracks the EMA of vehicle speeds by providing more weighting to more recent vehicle speeds captured from the hub. Plan quality evaluation then considers this average speed when reporting speed quality, and when calculating ETE.

6.0 FUTURE WORK

Related to the signal noise in vehicle speeds discovered while working with real vehicles through MAPLE, we intend to investigate the use of machine learning algorithms to learn trends in vehicle types. We expect to find that real vehicles will behave in certain ways under certain conditions. By learning trends in vehicle types, we may be able to adjust the thresholds towards triggering adaptation. Currently, these thresholds are configured at design time but it is technically possible to update thresholds at runtime. This could ultimately improve the plan monitor's ability to determine when adaptation should be used.

Currently, the plan monitor features adaptation strategies to re-route active plays when influenced by a restricted area. This is done by publishing a "ReroutePlay" message to the messaging hub which will invoke re-planning by the route-planning IA. We intend to experiment with strategies that can perform re-planning rather than delegating re-planning to another agent. This would effectively make the plan monitor a planner, and would allow greater control of the environment through strategies.

7.0 SUMMARY

In this paper, we described our approach towards plan execution monitoring of unmanned vehicle mission plans using an autonomics framework. Our approach uses the autonomic framework's probes to sense the C2 environment, and gauges to build and update a model that is maintained at runtime. By modeling mission plans as networks, we are able to provide continuous plan quality evaluation to the C2 operator and discuss how modeling plans as networks can allow for plan monitoring of large numbers of unmanned vehicles. Quality parameters such as vehicle fuel, speed, ETE and task quality are displayed to the operator in a simple and intuitive way that makes it easy to evaluate plan quality at any time. Strategies to notify the operator of important events as they occur can help towards maintaining SA. Finally, strategies to make changes in plans and the environment if permissible through working agreements complete the plan monitor's range of contributions as a team member in the human-autonomy team.

8.0 REFERENCES

[1] Gutzwiller, Robert S., Douglas S. Lange, John Reeder, Rob L. Morris, and Olinda Rodas. "Human-computer collaboration in adaptive supervisory control and function allocation of autonomous system teams." In *International Conference on Virtual, Augmented and Mixed Reality*, pp. 447-456. Springer, Cham, 2015.

[2] Behymer, Kyle, Clayton Rothwell, Heath Ruff, Michael Patzek, Gloria Calhoun, Mark Draper, Scott Douglass, Derek Kingston, and Doug Lange. *Initial evaluation of the intelligent multi-uxv planner with adaptive collaborative/control technologies (IMPACT)*. Infoscitex Corp. Beavercreek, 2017.

[3] Cummings, Mary L., Sylvain Bruni, S. Mercier, and P. J. Mitchell. *Automation architecture for single operator, multiple UAV command and control*. Massachusetts Inst Of Tech Cambridge, 2007.

[4] Garlan, David, Robert T. Monroe, and David Wile. "Acme: Architectural description of component-based systems." *Foundations of component-based systems* 68 (2000): 47-68.

[5] Garlan, David, S-W. Cheng, A-C. Huang, Bradley Schmerl, and Peter Steenkiste. "Rainbow: Architecture-

based self-adaptation with reusable infrastructure." *Computer* 37, no. 10 (2004): 46-54.

[6] Cheng, Shang-Wen, and David Garlan. "Stitch: A language for architecture-based self-adaptation." *Journal of Systems and Software* 85, no. 12 (2012): 2860-2875.

[7] Dzieciuch, Iryna, John Reeder, Robert Gutzwiller, Eric Gustafson, Braulio Coronado, Luis Martinez, Bryan Croft, and Douglas S. Lange. "Amplifying human ability through autonomics and machine learning in IMPACT." In *Micro-and Nanotechnology Sensors, Systems, and Applications IX*, vol. 10194, p. 101941Y. International Society for Optics and Photonics, 2017.

[8] Coronado, Braulio, Eric Gustafson, John Reeder, and Douglas S. Lange. "Mixing Formal Methods, Machine Learning, and Human Interaction Through an Autonomics Framework." In *2016 AAAI Fall Symposium Series*. 2016.

[9] Kephart, Jeffrey O., and David M. Chess. "The vision of autonomic computing." *Computer* 1 (2003): 41-50.

[10] Lucero, Crisrael, Braulio Coronado, Eric Gustafson, and Douglas S. Lange. "Tracking Provenance in Decision Making Between the Human and Autonomy." In *International Conference on Engineering Psychology and Cognitive Ergonomics*, pp. 171-180. Springer, Cham, 2018.

[11] Lange, Douglas S., Robert S. Gutzwiller, Phillip Verbancsics, and Terence Sin. "Task models for human-computer collaboration in supervisory control of teams of autonomous systems." In *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2014 IEEE International Inter-Disciplinary Conference on*, pp. 97-102. IEEE, 2014.

[12] Lange, Douglas S., and Robert S. Gutzwiller. "Human-autonomy teaming patterns in the command and control of teams of autonomous systems." In *International Conference on Engineering Psychology and Cognitive Ergonomics*, pp. 179-188. Springer, Cham, 2016.