SYSTEMS THEORETIC PROCESS ANALYSIS APPLIED TO MANNED-UNMANNED TEAMING

By

Jeremiah Robertson

B.S. Aerospace Engineering, Georgia Institute of Technology (2016)

Submitted to the Department of Aeronautics and Astronautics in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Aeronautics and Astronautics
at the
Massachusetts Institute of Technology

January 2019

Signature of Author _____

Jeremiah Robertson
Department of Aeronautics and Astronautics
January 15, 2019

Certified by _____

Nancy G. Leveson
Professor of Aeronautics and Astronautics and Engineering Systems
Thesis Supervisor

Accepted by _____

Sertac Karaman
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

[Page Intentionally Left Blank]

Systems Theoretic Process Analysis Applied to Manned-Unmanned Teaming

By

Jeremiah R. Robertson

Submitted to the Department of Aeronautics and Astronautics on January 15, 2019 in Partial Fulfillment of the Requirements for the Degree of Master of Science in Aeronautics and Astronautics

ABSTRACT

The Air Force Research Laboratory (AFRL) has identified autonomy as one of three game changing technologies for the future, along with hypersonic vehicles and directed energy weapons. One common application of autonomy that has been explored by numerous laboratories and research centers internationally is an unmanned aerial vehicle (UAV). AFRL is hoping to develop a UAV that will act as wingman in the traditional role of fighter pilots. Autonomous UAVs have several advantages over manned aircraft. First, they can operate in extreme environments with abnormal conditions where traditional fighter aircraft cannot maneuver. Second, autonomous UAVs can operate without human input where boring tasks like searching or monitoring would fall short due to lack of situational awareness. Finally, UAVs eliminate the risk of having Air Force personnel within firing range of an enemy.

However, manned-unmanned teaming (MUM-T) is a relatively new concept that has limited operational use. One of the challenges is designing safety into a system where automation can make decisions. The growth of MUM-T operations is primarily limited due to skeptical concerns about its safety and security. The Air Force maintains large amounts of classified data, and that information is transferred across several networks. If an enemy gained access to imagery or communications, a mission would fail and the enemy could prepare a counterattack. Prior attempts to perform a safety or security analysis of an autonomous UAV have focused on reliability as opposed to safety. FMEA and FTA calculate the probability of a component failure which is different from preventing a hazard. By following STPA, the requirements generated are directly traced back to hazards and losses, and the analysis will include interactions among components as opposed to strictly component failures.

Thesis Supervisor: Nancy Leveson
Title: Professor of Aeronautics and Astronautics and Engineering Systems

[Page Intentionally Left Blank]

ACKNOWLEDGEMENTS

First, I want to thank my wife Sierra. I moved us all the way out to a freezing, Patriot-infested tundra but you still came even though the number of Starbucks was abysmal. You put up with me for the past year and a half, and although it hasn't been easy, none of this would be possible without your constant support to help me push through every challenge. I want to thank my dog Red for waking me up on time every morning by licking my face and for always forcing me to get exercise even when it was -10 degrees outside with the wind chill. You are an incredible dog and thank you for letting me pet you when I got stressed out.

I would also like to thank my advisors during the process. To Professor Nancy Leveson for her constant guidance and teaching. I could not have done this without your leadership. Initially, I had trouble understanding how to use STPA. Now I hope to use all the knowledge and learning from the past two years to bring about a new perspective within several industries. To Dr. John Thomas, you always answered my emails even when I asked silly questions and you were always there to sit down and talk through ideas. Thank you for being patient with me and helping me learn every single day. I want to thank Kerianne Hobbs and the team at AFRL/RQ for their support and expertise. I also appreciate the Systems Engineering Lab (Diogo, Lawrence, Sarah, Michael, and Dylan) for being supportive and helping me every step of the way.

Thank you to my family and friends who have supported me throughout my life. I would not be where I am without you, and I will always remember each and every one of you. To the Graduate Christian Fellowship and Aletheia Church, thank you to everyone that has helped me in my spiritual walk. Your constant grace, love, and kindness will never be forgotten.

[Page Intentionally Left Blank]

Table of Contents

Table of Figures ..... **viii**
Table of Tables ..... **ix**
List of Acronyms ..... **x**
Chapter 1 ..... **1**
Introduction **1**
Motivation *2*
Objective *4*
Thesis Structure *5*
Chapter 2 ..... **6**
Software Design Myths *6*
Literature Review **7**
STAMP *8*
Hazard Analysis Techniques *9*
Fault Tree Analysis 9
Failure Modes and Effects Analysis 12
Hazard and Operability Study 13
Systems Theoretic Process Analysis 14
MUM-T History *16*
AFRL 18
HAVE RAIDER 18
US Army Future Vertical Lift Project 19
UAV Designs *20*
CONOPS *37*
Fighter Pilot and Remote Pilot Mission Process *38*
Mission Specifics *41*
Chapter 3 ..... **45**
MUM-T Mishaps, Hazards, and High-Level Safety Constraints *45*
UxAS Functional Control Diagram *54*
Step 1: Unsafe Control Actions *61*
Step 2: Causal Scenario Generation *64*
Chapter 4 ..... **80**
Summary and Conclusion *80*
References **84**
Appendix A: Unsafe Control Actions **88**
Appendix B: Scenarios and Requirements **102**

Table of Figures

Figure 1. UAV Swarm WSN Example [13] 8
Figure 2. Control Structure for Fault Tree Example [15] 11
Figure 3. Fault Tree Analysis Example [15] 11
Figure 4. FMEA Power Control Structure [16] 13
Figure 5. JHU APL UAV swarming modular architecture [24] 22
Figure 6. MIT/LL Reference Service-Oriented Architecture 24
Figure 7. MIT and Aurora Flight Sciences Open System Architecture 25
Figure 8. High Level System Architecture Example for Wildfires 27
Figure 9. Main Components and Layers of the Aerostack Architecture [28] 31
Figure 10. Mission Planning Communication Architecture [29] 32
Figure 11. The RAND Corporation Recommended UAV POSA 36
Figure 12. ISR Mission 38
Figure 13. Air to Air Combat 38
Figure 14. Generic FMS Display 42
Figure 15. Feedback Control Loop 49
Figure 16. UAV Functional Control Structure [14] 55
Figure 17. UAV Physical Data Structure [37] 56
Figure 18. MUM-T Functional Control Diagram 61
Figure 19. Causal Factors for Scenarios [39] 66
Figure 20. Causal STPA Scenario Generation Model [40] 67
Figure 21. Team Lead Control Structure Focus 68
Figure 22. Ground Station Control Structure Focus 72
Figure 23. Autonomous Controller Focus 74
Figure 24. Autonomous Controller Control Structure Specifics 74

Table of Tables

Table 1. FMEA Table for Power Example [16] 13
Table 2. HAZOP Guide Words 14
Table 3. UAV Communication Technology Details 33
Table 4. Mishaps and Hazards 48
Table 5. Feedback and Responsibilities 56
Table 6. Aircraft Specification 58
Table 7. Autonomous Controller Unsafe Control Actions 62
Table 8. Team Lead Unsafe Control Actions 63
Table 9. Ground Station Unsafe Control Actions 63

List of Acronyms

AFRL	Air Force Research Laboratory
ATC	Air Traffic Control
AuC	Autonomous Controller
AWT	Aircraft, Weapon, or Target
BDA	Battle Damage Assessment
CONOPS	Concept of Operations
COTS	Commercial Off-The-Shelf
DOD	Department of Defense
EII	Enemy Identifying Information
FMEA	Failure Modes and Effects Analysis
FMS	Flight Management System
FTA	Fault Tree Analysis
GPS	Ground Positioning System
GS	Ground Station
IFF	Identification Friend or Foe
ISR	Intelligence, Surveillance, and Reconnaissance
JCUA	Joint Common UAV Architecture
LOS	Line of Sight
MFC	Main Flight Computer
ML	Machine Learning
MP	Mission Planners
MUM-T	Manned-Unmanned Teaming
NASA	National Aeronautics and Space Administration
PI	Positive Identification
PNT	Position, Navigation, and Timing
POSA	Partially Open System Architecture
ROE	Rules of Engagement
STAMP	Systems Theoretic Accident Model and Processes
STPA	System Theoretic Process Analysis
TL	Team Lead
UAV	Unmanned Aerial Vehicle
US	United States
UxAS	AFRL-developed open architecture for UAV swarms

[Page Intentionally Left Blank]

Chapter 1

Introduction

On January 11, 2018, a Russian military base in Syria was attacked by a swarm of approximately 13 unmanned aerial vehicles (UAVs). According to the Russian Ministry of Defense, the forces at Khmeimim Air Base and Tartus Naval Facility successfully protected their assets against a potential terrorist attack because the UAVs may have contained weaponized missiles. Several drones were destroyed by Pantir-S anti-aircraft missiles and others were intercepted by Russian electronic warfare units. The intercepted aircraft were either safely escorted to ground or crashed on the runway. This event is the first publicly known event involving a large-scale, coordinated UAV swarm assault on a fixed installation with intelligence value [1]. Although Defense applications are a primary customer for coordinated UAV tactics, civilian applications are becoming popular as several UAV projects have emerged over the past decade including UAV-NET, COMETS, cDrones, MDRONES, and more.

UAV-NET is a software framework that utilizes a series of quadcopters to set up a wireless internet network such as an IEEE 802.11s wireless mesh network. There are nodes on each quadcopter that are automatically interconnected to establish a relay of wireless mesh nodes. UAV-NET can set up the nodes on each machine, deploy the nodes to extend the network, manage the signal strength and security, and monitor the network for intrusions [2]. This capability is ideal for forest rangers or park service members in areas with little cell phone or data reception coverage that require a high degree of wireless access to upload or download data while moving. As a result, the researchers from the University of Switzerland are negotiating a deal to sell the UAV-NET design and implementation procedures to the Swiss government.

Both examples highlight the most common applications for UAV swarms in the present day. Most applications are either developed for governments defending against or attacking other nations, or for civilian functions such as gathering intelligence about lost citizens. The problem that arises with the government application is that the software and hardware become proprietary because governments have a vested interest in ensuring no other nation can employ the same technology. This makes it difficult to imitate the software or design. However, there are assumptions built into the software and design so even if the technology could be re-used, the likelihood of it working appropriately for other applications is extremely small.

For example, if Amazon wanted to set up a UAV swarm to deliver packages, then using UAV-NET may cause problems for several different reasons. First, UAV-NET is designed to have co-leaders involved in running the swarm. Amazon would not want just two swarm leaders to oversee the entire fleet of UAVs under their jurisdiction because it is computationally impossible to have two aircraft control such a large network. Moreover, UAV-NET is specifically designed to set up a wireless mesh network. Thus, the optimization pattern for the UAVs is to increase the amount of distance between them to extend the network without sacrificing the wireless signal

for users accessing the network in real-time. However, Amazon may want to minimize the amount of time a UAV takes to deliver a package and return to base to conserve battery life, preventing unnecessary charging throughout the day. As a result, there is an increasing need for a UAV swarm architecture that allows for flexibility in mission objectives, optimization needs, and number of users or aircraft in the system.

The Air Force Research Laboratory (AFRL) developed an open source UAV swarm architecture in March 2017 called *OpenUxAS*. This architecture was created to solve the problem of limited mission flexibility. OpenUxAS, or UxAS for short, works for diverse sets of missions, and can be modified to fit the optimization needs of the assignments for each UAV. UxAS is a collection of services that use a message passing architecture to send packets of information between each UAV like a Robot Operating System (ROS). All software classes can create, change, or serialize the messages which are based on the Light-weight Message Control Protocol (LMCP) format [3]. This allows the user to determine the appropriate input and output for a service by reading the messages coming into and leaving the system. There are several different services available including, but not limited to, automated route planning, diagramming message traffic, optimizing task orders, validating mission requests, etc. Perhaps the most valuable service of UxAS is the optimal task allocations, which takes in an assortment of tasks and determines which UAV should complete a specific task, the order the tasks need to be accomplished, and the amount of time available to complete a task.

Although UxAS has already been released to the public, it was designed with specific assumptions in mind. For example, UxAS engages only if a UAV is in autopilot and steady level flight. Therefore, UxAS only partially provides autonomous capabilities by providing autonomy during a small portion of flight. Once the UAV begins aiming at a target or becomes engaged in combat, UxAS may no longer be useful. A method is needed to analyze what safety and security requirements would be necessary for an autonomous controller to properly handle a UAV during different phases of a mission, and for a broader set of possible missions. Additionally, how would a manned aircraft and other members of a mission interact with the Autonomous Controller and UAV(s). Specifically, this thesis will analyze the requirements needed for a generic manned-unmanned teaming (MUM-T) mission using Systems Theoretic Process Analysis (STPA).

Motivation

Most methods used to perform hazard analysis are 50-70 years old and only work on hardware. For example, Failure Mode and Effects Analysis (FMEA) was developed in the late 1940s to determine the probability of a design failing or operating successfully and for what percent of the time. As design changes were made or single points of failure were identified, the analysis could be re-done to understand if the changes were beneficial or caused further problems. However, FMEA does not work on system components like software or humans that do not “fail” randomly. FMEA can handle hardware failures such as pipes breaking, an O-ring falling off, a screw becoming loose, a fuselage cracking, etc. Conversely, if the mechanics did not inspect the bolts or the pressure increases because the software logic opens the wrong pipe at the wrong time, then replacing the hardware or providing redundant systems will not permanently prevent

the hazard. Additionally, the probabilities are subjective. Just because a probability of failure seems small does not mean it cannot lead to a severe accident when coupled with other failures in the system.

Other techniques have similar problems. Fault tree analysis (FTA) was developed in the early 1960s by Bell Laboratories as a top-down approach to complement FMEA. FTA uses Boolean logic to identify how the interactions between component failure can combine to cause an accident. Because the goal is to reduce the probability of system failure, redundancy is the usual way that the design is changed to reduce the failure rate. Redundancy, however, is only one way to deal with component failures and tends to be expensive and error-prone itself. Like FMEA, FTA was developed for the systems being developed in the1960s, which were primarily composed of hardware components. Humans were omitted from the analyses (or simplistic probabilities used for "human failure"). Software was not used for direct control until much later.

Modern systems have increasing complexity in terms of technology, interactions between components, and changes in the roles of various components. Because of this, accidents no longer result primarily from hardware component failures. For example, the Mars Polar Lander incident had no component failures although the Lander crashed into Mars surface. This occurred due to a software requirements error. The Lander was supposed to deploy a parachute and descend with reverse thrust engines through the atmosphere. Once the Lander touched down on the surface, the software would be informed by sensors on the landing legs that the spacecraft touched down so the descent engines could be cut off. As the spacecraft approached, the sensors received sensor feedback when the landing legs deployed while the spacecraft was still 40 meters above the surface. As a result, the spacecraft thought it landed and shut down the engines, causing the Lander to crash into Mars surface (cite Nancy). This example illustrates a scenario where nothing failed, but rather, complex interactions led to an accident.

Flawed software requirements are the cause of a significant number of accidents. Engineers make incomplete or wrong assumptions about the operation of controlled systems and this leads to unhandled system states which when combined with environmental conditions, leads to an accident. Furthermore, trying to get the software correct or make it reliable will not necessarily make it safe. A system can be reliable and still be unsafe as previously described. The Lander's software worked correctly (as defined by its requirements) as well as the sensors, but the requirements were wrong.

Software allows for unlimited system complexity so it becomes difficult to plan, understand, anticipate, and guard against undesired system behavior [4]. There is also no way to exhaustively test and get out all the design errors. Thus, analysis methods are needed that can handle hardware, software, human factors, component interactions, system design errors, management, regulation, policy, and environmental factors.

Traditionally, people coped with complexity using either analytic reduction or statistics. Analytic reduction, or decomposition, posits that dividing the system into distinct components and analyzing each part separately is a plausible approach. After the analysis, the parts can be combined to evaluate the interactions between the components and understand the behavior of the whole system. This assumes that each component operates independently, and that the

analysis results are not distorted when the components are considered separately. This assumption in turn implies that the components or events are not subject to feedback loops and other nonlinear interactions and that the behavior of the components is the same when examined singly as when they are playing their part in the whole. Unfortunately, this is rarely ever the case in modern systems. From a simplified view, an aircraft flight mode computer interacts with GPS receivers, an inertial navigation system, the Control Display Unit, a keyboard and touchscreen, the Electronic Flight Instrument System, a navigation display, and a cross talk bus. Separating out these components and assuming a singular failure will miss accidents that result from the interactions between them when they do not fail.

Additionally, statistics can be used for systems that are sufficiently random. This could be something like a gas mixture chamber that involves billions of different molecules where the final outcome can only be understood using probabilities and statistical mechanics taken over a certain number of iterations. However, most systems do not exhibit this degree of randomness.

Systems theory was developed in the 1960s to analyze systems that are too complex for analytic reduction and too organized for statistics [5]. In systems theory, emergent properties, such as safety and security, are properties only of the system and not individual subsystems. STPA is based on systems theory. It has been used on systems in most safety-critical industries including aircraft, spacecraft, ATC, automobiles, medical devices, chemical plants, petrochemicals, and more.

In terms of its effectiveness, there are now at least a hundred comparisons between STPA and other hazard analysis methods. In each of these comparisons, STPA found not only the same hazard causes as the other methods, but additional causes that other methods did not discover. Moreover, the cost was orders of magnitude less than traditional methods. For example, an evaluation and comparison of the Traffic Collision Avoidance System (TCAS) validated that STPA was more comprehensive and complete than a fault tree provided by The Mitre Corporation for the Federal Aviation Administration [6]. STPA found scenarios involving improper independent testing and software logic contradictions that the fault tree did not. As another example, STPA performed on a US Navy software-intensive positioning system identified a scenario that the Navy decided not to correct. The scenario occurred within two months of putting the system into operation and led to a collision with a nuclear submarine [7]. As a final example, the Air Force Flight Test Center found twice as many recommendations to eliminate hazards using STPA for flight testing while only requiring four-man-months to isolate the corrective actions compared to years of evaluation with FMEA [8].

Objective

The main objective of this thesis is to apply Systems Theoretic Process Analysis (STPA) to a generic MUM-T Loyal Wingman CONOPS. This will generate safety and security requirements necessary for a potential Intelligence, Surveillance, or Reconnaissance (ISR) mission. Additionally, the analysis can be applied to air to air combat scenarios where the UAVs are acting as a sensor truck or weapons truck.

Thesis Structure

Chapter 2 covers some of the background and development of MUM-T along with various UAV swarm architectures, and why STPA is critical to ensuring safe and secure systems. Chapter 3 provides an overview of the STPA analysis applied to MUM-T. Finally, Chapter 4 will summarize the results and their applicability to MUM-T systems.

Chapter 2

Software Design Myths

There are several different techniques that can be used to analyze the overall safety of software. However, before diving into the different techniques that could be used, it is important to understand two of the most common misconceptions when it comes to software reliability. People tend to believe that highly reliable software equates to highly safe software [4]. This problem is described by Frola and Miller in their 1984 text on System Safety for the Department of Defense (DOD). They describe a series of problems that occurred on the software of various weapon systems. Examples include a missile that failed to separate from a launcher due to sequencing timing, a fly-by-wire aircraft had a mechanical malfunction the software was not designed to handle, and a B-52 bomb bay door closed unexpectedly when the computer's controller circuit had a glitch and the power began to fluctuate. These problems not only involve the software of the main flight computer, but also aircraft hardware, computer hardware, and the pilots [9]. As highlighted in each of these cases, the software worked as it was intended. There were no software errors like an incorrect syntax or a misappropriation of error handling causing the software to err.

As a result, software needs to be analyzed from a systems perspective to make it safer. Software must be analyzed within the context of an entire system, including who uses it, the hardware connected to it, and what other actuators or feedback influence the software process model. Leveson notes that computers are so powerful that there are no longer physical constraints to limit designs [4]. However, this results in extremely complex software that can go beyond the capacity of our ability to understand it. With so many components interacting and software in between those interactions, the result is software accidents that scale with increasingly complex designs. Altogether, software reliability does not ensure a safe system or even safe software.

Another software myth that is important to discuss is the idea that software is easy to change. Although software itself can be augmented with ease, the results of those changes can be costly in terms of testing, validation and verification (V&V). Boehm noticed in the early 1980s that the cost of software changes significantly increases as the software time of life increases. Large software projects tended to have 100 times as many changes from the starting software to 5 years after the start, whereas smaller projects were around four times greater [10]. Moreover, system changes in general become more expensive later in the design lifecycle, so these software changes can exacerbate the cost [11]. Although computer science has greatly advanced since the 1980s, the concept still rings true as researchers have confirmed the finding through similar research. Additionally, not only does recertification and revalidation bring immense expenses, but the software also becomes more fragile. It is difficult to change software without introducing what may be potential errors because the changes also depend on the hardware [12].

These software fallacies are important to understand within the context of this research problem because these issues will arise when comparing STPA with outputs from similar hazard analysis techniques.

Literature Review

UAVs are utilized in numerous mission spaces such as border patrol, emergency assistance, search and rescue missions, and natural disaster monitoring. However, rarely does one UAV suffice for these types of missions; typically, a minimum of five UAVs will be used depending on the area of coverage, sensor or payload requirements and length of mission. As a result, a demand exists for the ability to have each UAV within a swarm communicate with each other from takeoff to landing.

To provide an example of the potential complexity of these missions, consider the recent wildfire devastation in Southern California. There are four main stages during the lifecycle of a disaster that involve UAV swarms: pre-disaster preparedness, disaster assessment, disaster response, and recovery. Pre-disaster preparedness includes monitoring the region for potential storms and forecasting the disaster using a wireless sensor network (WSN) as an Early Warning System. UAVs will sense a temperature, pressure, chemical or environmental change within their surroundings and send that information to a relay network for data transmission. The data from each UAV within the WSN network is sent to a few common relay UAVs that downlink this information to a server for human analysis to investigate if a disaster is occurring.

Subsequently, this leads to the disaster assessment phase where humans continue to evaluate the data and determine what actions need to be taken in real time. Additionally, if any damages occur during data transmission, then standalone systems are deployed to assist in transmitting data packets when relay networks are having trouble receiving or sending signals. Finally, once the overall damage of the situation is analyzed, UAVs will assist in search and rescue missions if necessary to identify where humans may be trapped or injured during the disaster and if there are any imminent threats to their survival. Some UAVs even have medical applications where they can calculate an individual's oxygen levels, heart rate, and blood loss from afar using visual inspection and healthcare data from the patient's history. An example of UAV teams functioning together using a WSN network can be seen in Figure 1 [13].

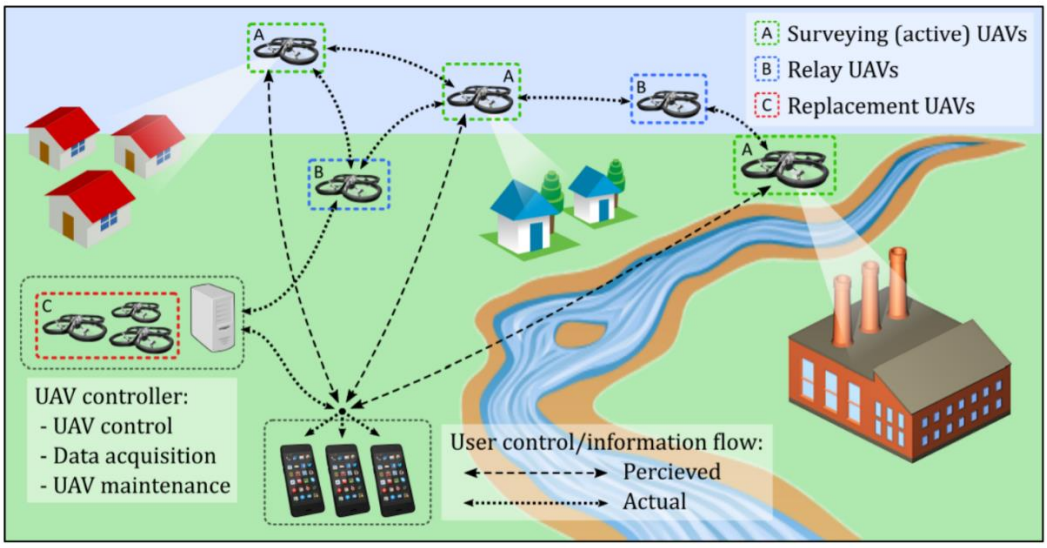

Figure 1. UAV Swarm WSN Example [13]

This is a good example of a CONOPS diagram where each UAV performs a specific task, and those functions fit together to accomplish the overall mission of surveying the region. All missions requiring UAV swarms will follow a similar structure. There will always be a complex flow of information between UAVs, data acquisition systems, human monitors, and more. This information lifecycle is complex considering the magnitude of data being sent and the number of players involved. As a complex system, it is important to utilize a systems-based design process to properly evaluate the design, conduct a technology review, construct an operations lifecycle plan, and ultimately retire the technology at end of life. There are several examples of systems engineering processes. Each one will be explored in detail to identify how STPA impacts the processes at each segment. There are safety processes for each organization, and those formal methods will be compared with STPA to show the difference in how safety is currently done and how STPA will implement safety as a dynamic control problem and not exogenous to the system.

STAMP

Systems Theoretic Accident Model and Processes, or STAMP, is a new model of accident causation based on systems theory. Accidents are a result of complex and dynamic processes, so treating safety like a dynamic control problem as opposed to a one-time physical failure problem is a more accurate manner of incorporating safety. This requires enforcing a set of constraints on a system such that interactions between components do not violate the constraints. Safety then becomes a control problem, and not a reliability or redundancy problem. Accidents are a result of inadequate control due to a lack of enforcing the respective constraints on the system. Thus, by controlling the behavior instead of trying to improve probabilities, teams can improve safety instead of trying to minimize the potential for any accidents [4].

A simple way to illustrate this framework is thinking about stretching before exercising. Stretching is an example of risk reduction that people use to avoid pulling their muscles or causing major damage to a specific muscle group. However, when thinking about how someone exercises, there are several different ways this could take place. A person might lift a weight, jump up and down on the floor, pull themselves up, push themselves down, throw a ball to someone else, etc. Within each of these actions is the potential for something to go wrong. Stretching may prevent someone from pulling a muscle, but if someone drops a weight on their foot while trying to stretch their arms, an accident has still taken place. This is the difference between risk reduction and analyzing safety from the perspective of systems theory and control actions. In this case, knowing that someone could drop a weight on their foot, a constraint could be developed that someone needs to be able to stretch in a safe location along with several other requirements or constraints for the person stretching.

In addition, Major Summers identified that STPA is also compliant with the Air Force airworthiness process standards as defined in sections 4 and 14 of MIL STD 516C. Through the handbook, the verification for compliance is identified as an inspection. STPA provides a step by step process for inspection analysis as opposed to previous experience or engineering judgment which may be incomplete or inadequate to judge a newer system that is significantly different from its legacy counterpart. Another important aspect of airworthiness is the ability to maintain it throughout the lifecycle of a system. STPA also provides a construct to evaluate changes. If safety concerns are introduced in the modifications, STPA will provide constraints for the design that can be integrated with the baseline system. STPA will also cover the support structure such that hazards are not introduced from outside the system design which could include Congressional funding, Air Force management, and more [14].

There are three parts to STAMP: safety constraints, a hierarchical control structure, and process models. In STAMP, safety constraints are developed by beginning with a top-down approach of a system instead of looking at individual components. A control structure will enforce the safety constraints identified, and as the system evolves, STAMP can be re-iterated with ease to ensure continued effectiveness and consistency over time. This allows flexibility in the development process because now a team can change the design relatively easily without worrying about how to re-develop requirements and redo a risk matrix for the system. Safety is no longer tied to individual component failures but the system behavior as a whole. Therefore, changing one component may not require an entire analysis from the beginning, and if it does, at least the analysis is concise while still maintaining high accuracy. Some organizations have tried to perform similar analyses using other techniques, but there are limitations as noted in the next section.

Hazard Analysis Techniques

In this section, several different hazard analysis techniques are presented. Each subsection will explore why the methods are insufficient when considering modern day accidents or systems.

Fault Tree Analysis

Fault Tree Analysis (FTA) was developed by Bell Labs in 1962 as part of the hazard analysis for the U.S. Air Force Minuteman Intercontinental Ballistic Missile program. Subsequently, the technique was published and covered at the System Safety conference in 1965 in Seattle. The FAA began using FTA in airworthiness regulations in 1970, and in 1988, published Order 8040.4 which established FTA as a risk management tool to use in a range of certified aircraft activities such as Air Traffic Control. This technique has expanded to other fields like nuclear power and pharmaceuticals as an example of starting from a hazard and tracing down what component could cause an accident.

FTA can be organized into 5 major steps:

1. The user defines what type of undesired event to analyze. This step is very important considering the rest of the analysis hedges on the definition of the incident. No two accidents will produce the same FTA so it is important to define the accident explicitly and be as detailed as possible.

2. The system itself must already be known and constructed with specific components. As the components are analyzed, probabilities are assigned to identify how likely a certain component will fail based on its subcomponent failing.

3. Now one can construct the fault tree and apply logical AND/OR gates to show how a component failure affects the system at each level up to the accident.

4. Once completed, the team can analyze the FTA to understand the best ways to improve how to manage the risk involved.

5. The final step is to mitigate any of the potential hazards by inserting redundant systems, isolating a component, etc.

An example of a logical diagram, along with its FTA, can be seen in Figure 2 and Figure 3 [15]. This example shows a controller that is opening and closing a supply valve using a pressure transducer. The controller receives feedback from a pneumatic actuator. The top of the FTA shows the undesired event as the Gaseous Oxygen line fails to flow through to the supply valve. Subsequently, the entire tree is developed with the bottom layer showing how command failures, improper contacts, power shut downs, etc. lead to the hazard using a series of AND/OR gates between the events.

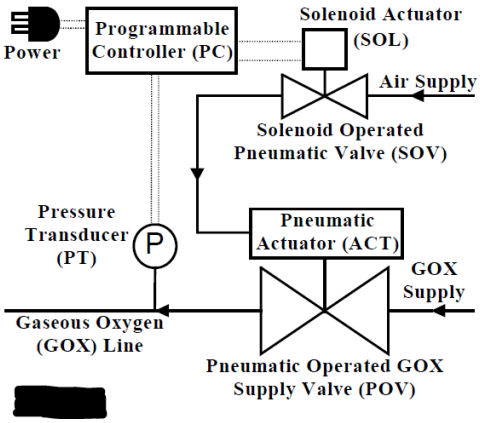

Figure 2. Control Structure for Fault Tree Example [15]

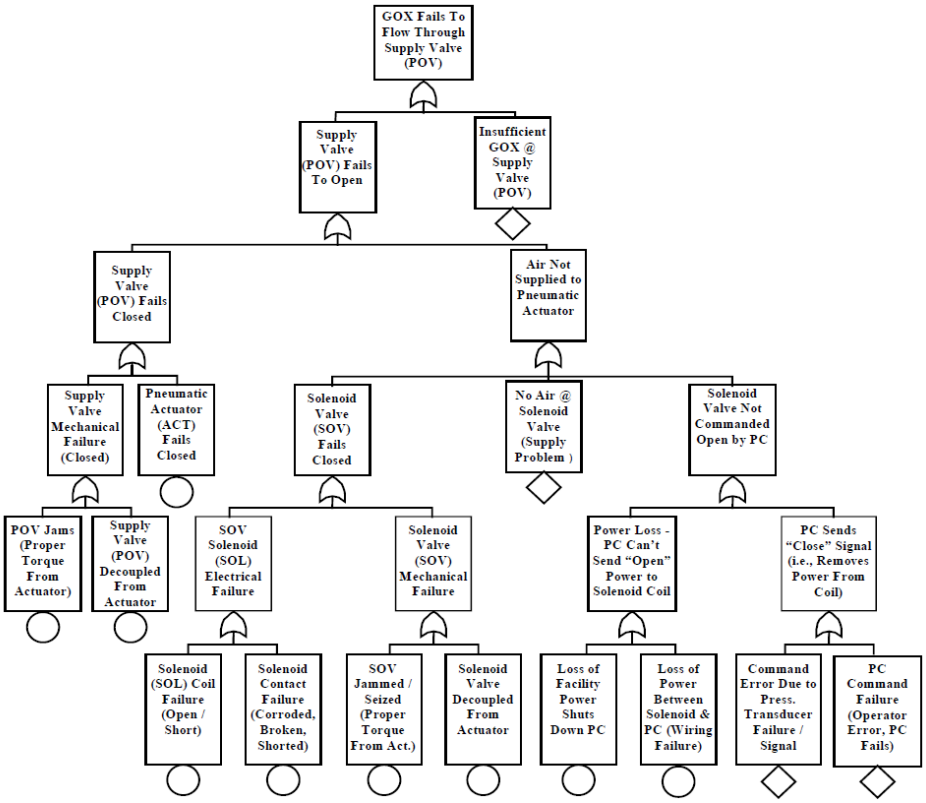

Figure 3. Fault Tree Analysis Example [15]

One of the strengths of FTA is that the component failures are traceable to the hazard. There is a "chain-like" way of demonstrating how a bottom layer component directly affects each of the higher-level components to eventually result in the hazard. FTA can also highlight specific weak points in the subcomponents where redundancy may need to be built in.

One of the limitations of FTA is that it assumes independence between each set of components and tree lines. In the previous example, the actuator may send faulty readings to the controller. This event by itself may not lead to a hazard, but if the readings are too high, then the controller

will think the valve needs to be closed so that the valve does not over-pressurize. However, the valve needs to be open so that GOL can enter the system. Thus, it is the interactions between the components that can also lead to a hazard which FTA cannot account for.

Moreover, there is no way to capture non-discrete variables. Only one state can be examined at a time. Although some groups have attempted to create dynamic fault trees, this is burdensome and requires thousands of pages to investigate such that the utilization of FTA is nullified. Finally, FTA only examines simple failure cause and effects. If a microchip fails, then usually the processor will fail but this type of simple statement about the hardware will not always be the case. Sometimes the processor might move to a backup chip, or sometimes it might stall and wait for another chip, or sometimes it might take 4 or 5 chips to fail before the processor decides to stop functioning. However, these scenarios cannot be captured in FTA because the user is making assumptions about the functionality of the system.

Failure Modes and Effects Analysis

Failure Modes and Effects Analysis (FMEA) an alternative to FTA. FMEA is a bottom up approach that analyzes individual component failures to understand how they will affect the system. Often this analysis is split into 2 parts: the first is understanding what the component failures do, and the second is a criticality analysis to understand how critical the failure is. This type of analysis can be done at varying levels of a system. For a laptop, someone could look at high-level components like the screen, keyboard, and hard drive. One could also dig into sub-components like mini LED bulbs, resistors, and small discs associated with higher-level components.

Ideally, the FMEA is a living document that is updated as system changes are made so that engineers can understand the criticality of certain subsystems which may warrant further design changes depending on the results. Therefore, a FMEA is useful primarily in the early stages of development if it can be completed before the product is developed. Typically, several FMEAs are completed during the design process, starting with a high-level analysis once a preliminary design is constructed. Then a lower-level FMEA is completed after specific trade studies identify what components will be in the system.

When building out a FMEA, there are specific rules to consider. First, only one failure mode can be considered at a time. The goal is to see how one specific component affects the whole system, so analyzing two component failures simultaneously will not highlight which component is causing the accident. All inputs to the system are assumed to be normal such that each subsystem is acquiring appropriate inputs in order to create ideal outputs. Another assumption is there are enough quantities of any resources exogenous to the system. For example, a solar panel requires sunlight to develop energy. Therefore, part of the analysis will not examine a lack of sunlight since this source is from the Sun and not the panel; the accident cannot occur because there is not enough of an input to the system. The power necessary for a system to operate will always assumed to be sufficient for FMEA. A simple power example is included in Figure 4 and Table 1 [16].

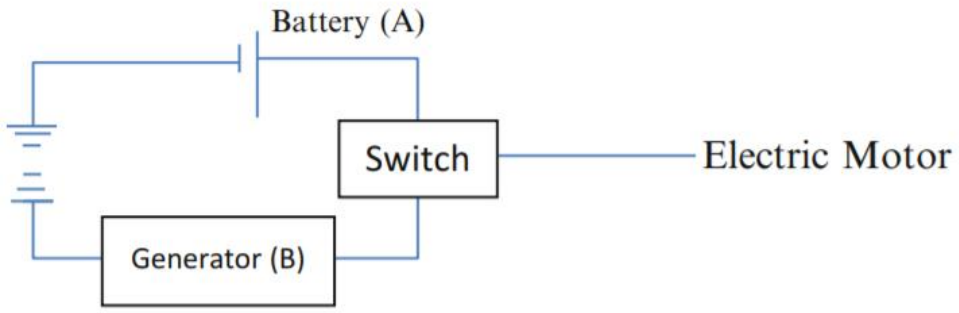

Figure 4. FMEA Power Control Structure [16]

Table 1. FMEA Table for Power Example [16]

Component	Failure Mode	Probability	Effects on Motor
Battery	Low voltage	1e-3	Major
	Short circuit	1e-6	Critical
	Fluctuating voltage	1e-2	Minor
Diesel Generator	Engine failure	1e-4	Severe
	Alternator failure	1e-8	Severe
	Fluctuating voltage	1e-2	Minor
Switch	Stuck in generator	1e-6	Major
	Contacts broken	5e-3	Critical

One of the problems with FMEA is that it encourages redundancy as a solution similar to FTA. If a lower probability of a given failure mode is desired, then increasing the quantity of a specific component and assuming independence among components will decrease the probability. The probability could be cut by a fraction and a failure's effect on the system decreases. However, this result assumes independence between the components, and redundancy should not be the only solution when building a safer design. Finally, the FMEA process can become very complicated and tedious. Every design change requires a complete re-evaluation of the effects and probabilities which can result in thousands of pages of worksheets. The failure modes must already be known as well. Failures cannot be considered if they are unknown. Additionally, the probabilities are subjective to the individual completing the FMEA. The numbers are completely based on an individual's expertise and unrelated to the safety. A system could have a 10^{-99} chance of occurring but once it happens, then the probability is irrelevant.

Hazard and Operability Study

Hazard and Operability Study (HAZOP) provides a systematic and structured method for analyzing processes or operations to identify problems specifically with personnel or equipment. This technique breaks down a process into different nodes to individually review them so that the system is less complex to analyze for the user. This technique is purely qualitative and uses guide words to help users evaluate a node. HAZOP was also developed in the 1960s for chemical processing, but has since been extended to mining, nuclear power, and other fields.

Similar to FTA and FMEA, HAZOP requires a fully defined process with all of the working components and people. HAZOP begins by defining the scope and goals for the analysis. A team may only want to analyze one segment of a process instead of the whole process. Subsequently, a study is done to collect data about the process: this could include personnel hours, equipment fatigue, hours of operation, number of work injuries, etc. Next, based on the data, analysis is done on segments. This analysis uses the guide words to help identify accident causes and whether significant problems could occur [17].

An example of applying guide words for Figure 4 could be referencing the flow of electricity between the battery and the switch. A list of guide words used in HAZOP, along with their application to the battery, is in Table 2. For example, the word "More" indicates that there is too much of something than is necessary. For this case, the battery might send a higher electrical flow than the switch can handle. The words "As well as" indicate that the electrical flow could be dangerous if it occurs with improper properties (wrong voltage, worn circuit, etc.).

Table 2. HAZOP Guide Words

Guide Word	Battery Application
More	High electrical flow
Less	Low electrical flow
None	No electrical flow
Reverse	Reverse electrical flow
As well as	Electrical flow at wrong voltage
Part of	Electrical flow is contaminated

After identifying different scenarios of where accidents might occur, a report is generated with the findings. Sometimes, part of the analysis may include identifying safeguards to prevent the causes along with actions that are required to implement the safeguards and who must perform the actions. Once again, HAZOP experiences the same limitations as FTA and FMEA. By segmenting a process into nodes, HAZOP assumes independence between the nodes. One node might send inputs and receive outputs from other nodes and segmenting them eliminates the ability to find causes as a result of improper coordination between nodes. Additionally, HAZOP is tome consuming and expensive. An extremely complex process could have 1,000 nodes. Even with a team of 20 people, this becomes extremely onerous.

Systems Theoretic Process Analysis

Systems Theoretic Process Analysis, or STPA, is a solution to some of the limitations involved with other hazard analysis techniques. STPA is based on STAMP and includes causal factors that the older techniques cannot handle. This includes aspects like software flaws, component interactions, decision making models, organizational and management influence, and human machine interfaces. Moreover, STPA provides a step-by-step method to the user walking through the process. STPA works based on a functional control diagram instead of physical components, and the results from the analysis are safety and security requirements that system owners can implement to ensure the design acts safely and securely.

Another advantage of STPA is the user does not need a design prior to performing the analysis. STPA can guide the design process and provide input on which designs are safer than others early in the product development lifecycle which is cost-effective for engineers since design changes late in the lifecycle are orders of magnitude more expensive. STPA has similar goals to the other techniques, but STPA provides a framework for analyzing interactions and other system components like software or humans. There are two main steps involved in STPA, each with a series of processes that assist the user.

The first step is to identify a set of unsafe control actions that trace back to the accidents and hazards. Each UCA follows a similar structure, and there are four different types of UCAs that are possible.

1. A control action required for safety is not provided or not followed
2. An unsafe control action is provided
3. A safe control action is applied too early or too late
4. A safe control action is applied for too long or stopped too soon

An example of a UCA for the Gaseous Oxygen Line example is below. Notice that the structure of the UCA follows the format outlined in Equation 1.

The Programmable Controller does not provide a release command to the actuator when gaseous oxygen needs to enter the line, resulting in under-pressurization of the valve.

[Controller] [Type] [Control Action] [Context] [Hazard]

Equation 1. UCA Content

After developing a set of UCAs, the second step is examining the control loop and demonstrating how a UCA could occur by identifying scenarios resulting in the UCA. Each scenario will highlight specific design constraints that are necessary to produce a safe system and eliminate a hazard. The comparison of different hazard analysis techniques highlights how STAMP and STPA differ from all others. STAMP develops a functional model as opposed to a physical model. STAMP analyzes interactions and failures as opposed to simply failures. STPA comes up with requirements instead of probabilities. Therefore, the results from STPA will help design safety into systems instead of trying to add on safety features. The next section will analyze the brief history of MUM-T and provide overviews of different UAV swarm architectures to understand how MUM-T might function given a specific architecture selection. The results from STPA can be compared with these architectures to establish if any current architecture would succeed if used in for MUM-T or in what ways the STPA requirements are unique compared with current architectures.

MUM-T History

Before diving into the history of MUM-T, it is important to define MUM-T to differentiate between systems. There is no exact definition, so the author will define MUM-T as it relates to most modern systems that are defined as MUM-T systems. Manned-Unmanned Teaming is a system where a controller (human, software, etc.) works together with unmanned systems. Two key takeaways from this definition are that: 1.) a controller is specified instead of a human. This definition does not require a human although the word manned is in MUM-T. 2.) The main objective of MUM-T is not specified in this definition. Every organization will utilize MUM-T systems for different purposes or goals so this definition works for any system regardless of what it is used for.

One of the earliest examples of MUM-T is during World War II. The Royal Air Force of England took off with a B-17 and three chase aircraft heading towards Germany. The B-17 carried ten tons of explosives and was designed to be flown into a submarine bunker. Upon reaching cruise altitude at 2,000 feet, the pilots would arm the explosives and then parachute into the English Channel while the chase aircraft would guide the B-17 into the bunker remotely. Although the mission failed, the concept was essentially a MUM-T operation. The chase aircraft tried to control the B-17 remotely from their aircraft. The benefits of MUM-T were clear, and now MUM-T has popularly resurged while U.S. services were operating in Afghanistan [18].

The Air Force and Army had trouble evaluating critical targets without putting soldiers in dangerous situations. The Predator XP UAV provided a way to analyze the targets without putting people in danger. The Predator was also less expensive to operate on a cost per flight hour basis than manned fighter aircraft. Additionally, the Predator had longer flight times and less visible acoustic signatures, so it could hide from radars. Manned platforms have endurance limitations that limit their ability to provide Positive Identification (PI) of targets and complete a Battle Damage Assessment (BDA) [19]. This was the primary motivation for a platform that utilized the advantages of manned and unmanned platforms so that missions could be completed by unmanned systems while the manned system could provide guidance and oversight to ensure the unmanned systems were as accurate and successful as possible.

One of the most recent MUM-T demonstrations involved an H-145M helicopter in conjunction with Schiebel company's S-100 UAV. The H-145M cockpit was able to control the S-100 and fly jointly, performing a pretend ISR mission while also completing takeoff and landing. These flight tests were conducted with the Austrian military and Airbus. The next step for this specific group is to analyze the workload data and optimize the human machine interface to keep workload at a minimum for pilots [20].

L3 Technologies is currently working under a 97-million-dollar contract with the U.S. Army to bolster teaming between the AH-64 Apache helicopter and the RQ-7 Shadow drones. This would allow the Army to complete armed scout missions since the Kiowa Warrior helicopter is retired from service. The Army Science Board is conducting studies to understand what capabilities are necessary for MUM-T, what would the CONOPS look like, and what are the challenges in linking manned and unmanned aircraft. The Defense Advanced Research Projects Agency (DARPA) also has a program to evaluate whether a tablet can be used to control an S-76B while

being on board a Black Hawk helicopter. The program is set for demonstration and testing in 2019 [21].

One of the challenges in determining how manned and unmanned aircraft should work together is examining what type of UAV architecture allows for the optimal experience. For example, the Airbus H-145M cockpit utilizes a large dual FMS screen centrally located between pilots which allows for either individual to communicate with the UAV while the pilot's main screens remain relatively the same in terms of controlling the H-145M. However, the most recent development from L-3 shows that the Apache may be getting dual screens so that each pilot can select different drones to control the payload of a given aircraft. Currently, all AH-64D/E aircraft can receive video footage from a Shadow or MQ-1C Grey Eagle using direct data links [22].

Additionally, 30 of the AH-64E aircraft can control the sensors and flight path using a Tactical Common Data Link Assembly. However, this capability was deemed "prohibitively expensive" and the Army abandoned the system in favor of more data links to the helicopter that would increase video streaming resolution from the UAV. Now, all Apache rotorcraft have been retrofitted or designed for MUM-T capabilities that allow for controlling fire control radars, sensors, and flight paths. The Army has completed Operating Test and Evaluation, and by next year, Apache pilots are scheduled to be able to fire weapons from the Grey Eagle. Boeing is currently investigating whether the Apache's three screen display should be decreased down to one to limit pilot workload like in the F-35 [23].

Altogether, each aircraft is different. The Apache can connect with the Grey Eagle and the H-145M can connect with the S-100 but this is limited. Thus far, the only MUM-T operational capabilities are between specific aircraft and no firing capabilities have been examined. Additionally, there have already been dozens of mishaps for the Army drones. The Shadow and Grey Eagle have together experienced 40 Class A mishaps from 2011 to 2017 which is equivalent to almost 100 million dollars in damages. This does not include the other 250 mishaps which are not class A; however, this increases the monetary value of damages to almost 300 million dollars [24].

The army has traditionally used root cause analysis to understand the causes of the accidents and prevent those causes from re-occurring. However, one of the problems with root cause analysis is when to select a starting cause. Additionally, there is usually more than one cause for an accident. It is a complex series of interactions with lots of moving parts so fixing one part of the system may not prevent the accident. An example of this is the materiel failures such as engine issues, turbo-charger combustion, etc. When these problems occur, there may be a loose bolt or a leak in a valve, but this does not highlight where in a Technical Order the Army may need to change the wording or highlight to mechanics how to check the valves to ensure their integrity. Therefore, Causal Analysis using Systems Theory (CAST) should be applied to accidents to identify all the factors leading up to accident to truly prevent further mishaps.

Another issue with MUM-T is how to develop a UAV architecture that will allow for a human to control the unmanned system. Many current MUM-T demonstrations, including the Army and Airbus, utilize prior UAV architectures and retrofit them to allow a manned aircraft to control the UAV instead of a ground operator. Although this may work for simple missions and specific

aircraft, this retrofitting may not work for every scenario. However, no analysis has been completed to understand what safety and security requirements are necessary within a UAV architecture to allow for MUM-T to be done successfully with multiple controllers.

AFRL

AFRL developed an STPA for a MUM-T system with one manned aircraft and multiple UAVs. The CONOPS focused on different mission phases beginning with loading an aircraft on the runway and ending with offloading the vehicle following landing [25]. One issue with the AFRL analysis is that the separation by phase might miss accidents that occur within multiple phases. Sometimes an issue in the takeoff phase can cause an accident in the formation phase. However, these interactions can be missed by trying to segment a mission.

Additionally, one of the goals of the AFRL analysis was to utilize the control structure as a starting point for an Architecture Analysis and Design Language (AADL) model. AADL is used to model the software and hardware architecture of an embedded, real-time system. This limited the extent of the STPA analysis. AFRL developed a set of unsafe control actions but the rest of the analysis was not complete. No scenarios were developed, and there were no requirements for a MUM-T system. As a result, a full STPA analysis is needed that includes scenarios and requirements for a control structure that is not divided into phases but defines the appropriate control actions and feedback through all mission phases.

HAVE RAIDER

Dan Montes performed extensive research into MUM-T applications, along with background into the Air Force's strategies for autonomy. In 2010, the Chief Scientist of the Air Force published a report claiming that utilizing autonomy would be "disproportionately valuable". Again in 2012, the Defense Science Board developed a report recommending that the Air Force focus on developing software for MUM-T applications independent of any physical platform. This idea aligns with one of the goals at the time which was to be "inoperable" so that software could run on multiple UAVs or manned vehicles to provide interconnectivity between them. The road maps for UAVs extend out to 2040 and identify dynamic environment utilization and connectivity with satellites, ground stations, tanks, and more as two primary objectives.

In an effort to include greater autonomy in military systems, the Air Force teamed with Lockheed martin to develop HAVE RAIDER. The requirements for the MUM-T system were to:

- Automatically plan a ground-attack mission
- Execute tasks based on priorities given by the operator
- Dynamically re-plan the mission to minimize exposure to the threat
- Demonstrate autonomous formation flying, route following and rejoining

HAVE RAIDER was a demonstration of technologies required for an unmanned vehicle to fly as a wingman with a manned aircraft in battlespace. The experiment took place using an experimental F-16 where the pilot provides mission-level commands to the wingman and the

wingman were able to fly in formation and react to changing threats. Although HAVE RAIDER is only a demonstration of capabilities, the experiment highlights why STPA is so crucial for MUM-T systems.

Montes notes that many participants in HAVE RAIDER had trouble understanding what safety plan to implement when problems occurred. An aircraft may not fly exactly in formation as intended, and when that happens, many pilots were confused on the procedure for how to correct its attitude or eliminate the aircraft from the formation so he or she did not have to worry about them. Additionally, several pilots noted a distinction between mission vs. control levels. If a manned aircraft is in a situation that requires manual steering, throttle setting, or sensor pointing (control-level inputs), then this essentially negates the benefits of MUM-T. Pilots preferred providing mission-level inputs such as what formation to hold, when to fire, what object to search for, etc.

Montes also notes the distinct differences between an STPA-based project plan vs. using traditional methods. STPA identifies hazards inherent for the entire system and not just subsystems. STPA explicitly defines the system along with its boundaries and tends to be more straightforward in following the process. Users found that traditional methods required experienced reviewers to catch any missing pieces of the analysis due to confusion in what is included in a technical plan versus a safety plan. The data from this project indicates that implementing STPA for MUM-T is advantageous in defining safety and security requirements although the control analysis and details for the system definition are intricate and must be developed with careful consideration for potential mission sets [26].

US Army Future Vertical Lift Project

Nancy Leveson, in conjunction with members at the Systems Engineering Research Lab at the Massachusetts Institute of Technology, provided a demonstration of STPA based on the medium class of the Future Vertical Lift (FVL) family of aircraft CONOPS. This includes the Objective Mission Equipment Package (OMEP) as defined by the Joint Multi Role (JMR) Mission Systems Architecture Demonstration effort. The functional flight vehicle structure is modeled and partially analyzed as well as a portion of a weapons control system. Other more general aspects of the entire control structure, such as acquisition, maintenance, the DoD command structure, tasking, mission generation and assurance etc., are partially modeled but not analyzed. Modeling was kept as generic as possible during the process in order to promote commonality and interoperability in the resulting functional architecture. In this demonstration, STPA is used to generate high-level system and functional component safety and cyber security requirements/constraints and related design recommendations [27].

One differentiation with MUM-T from FVL is that the Army preferred a manned aircraft that could control another unmanned aircraft as a stepping stone for MUM-T. As a result, the FVL demonstration focuses on control-level inputs (similar to a remote pilot instance for a drone) instead of the mission-level inputs as noted in the HAVE RAIDER demonstration. An example of a subset of the control actions used in the analysis makes this point clearer. Two of the control actions from the Pilot in Command to the pilot vehicle interface were to “set aircraft attitude” and to “set aircraft power”. Both control actions are similar to the UAV instance described by

Sarah Summers except that the Pilot in Command is on an aircraft and a UAV is typically controlled by a ground operator. Additionally, this analysis was only partially as a demonstration to the Army of STPA's capabilities. A full analysis with extensive safety and security requirements is needed for all future MUM-T systems to understand the advantages of STPA over other methods.

As more groups begin developing MUM-T systems, one issue that will become prevalent is retrofitting software for a MUM-T purpose. The next section explores different UAV architectures to understand how they were designed, and why the STPA requirements for MUM-T will be significantly different than what has already been developed such that trying to re-use software may only exacerbate issues in MUM-T systems.

UAV Designs

Although UAV swarms are becoming more prevalent as uses for the technology evolve, there are still several challenges with the technology such as:

- UAV Location Guidance: UAVs have a limited flight time due to battery life, speed limitations, payload space requirements, and maneuverability considerations. Therefore, researchers have worked to develop machine learning algorithms that would allow a UAV to optimize its workload utilizing self-learning techniques but unfortunately, current work appears to be sub-optimal. Humans need to be involved in the information loop and provide feedback through the relay network to prevent any loss of data or the actual UAV. Cellular Network Location Databases can provide last recorded information to UAVs in an area to allow the system to estimate what people are affected and how the individuals are distributed, but this requires a complex set of signaling protocols that have yet to be developed. Algorithms need to be able to maneuver UAVs based on mobile pings, but this would also open concerns for hacking in terms of diverging UAVs from a disaster in a malicious scenario.

- Relay Network Maintenance: There are 2 main parts to designing and creating a relay network for data transmission. First, the UAVs need to identify optimized, central locations called *anchors* that connect the disaster region to a nearby radio network. Although having backup UAVs step in if a part of the relay malfunctions would be ideal, this simple replacement is complicated for the same reason that reserving backup channels on a cellular network does not solve the problem. The energy variations will fluctuate differently, the handoff process between the two vehicles will consume resources, and there will be great functional losses considering the UAV was doing something else before stepping into the role. One would need accurate channel models to optimize the number of hops a data packet might take to be executed, along with a control algorithm at each layer of the disaster management system.

- Data Fusion and Delivery: Currently, data is collected by the UAVs such as videos and images and transmitted to a control center. However, people on the ground can also send text messages or social media posts via the network and all feeds are sent to the same

control center. The bandwidth is appropriate for all of the information; nonetheless, problems occur because currently algorithms assume a fixed topography. The UAV relay network attempts to stay near a focus area, but the targets are dynamic, and people are moving and changing locations constantly while it is happening. Thus, the channels need to change according to how a situation evolves and expand or contract appropriately. The handover process is also difficult considering there may be smoke resulting in power losses of the signals, and the UAV's stability may require more power leaving little effort for the data transmission if winds or weather begin to affect the situation.

- Charging: At best, UAVs have 2 and a half hours of flight time without charging. Efforts to extend that amount of time or provide quick charging are improving but still substandard. Some have used magnetic induction charging stations on the ground for a UAV to land and charge quickly without manual intervention. Some have solar panels that can provide small amounts of energy considering it takes almost as much just to convert the sunlight into power the UAV can use. However, algorithms to optimize charging points in the peripheral area of a target location would be good to ensure no materials are lost during the process, especially considering the costs associated with each UAV could be up to two thousand dollars for small UAVs with limited capabilities up to millions of dollars for larger aircraft.

These are a few areas of improvement for UAV swarms. Subsequently, a need arises for a UAV swarm architecture that can complement multiple mission types and account for the varying obstructions that exist based on current architectures. Additionally, based on the discussion in Chapter 1, a large gap exists in terms of open source software UAV architectures. There are several open source architectures that exist, but each one is designed using a general requirement approach like the Waterfall or V method. The subsequent investigation looks at several different architectures and the significant discrepancies between STPA and the methods used by each group.

One key distinction is the difference between a service-oriented architecture versus a distributed architecture, and in modern times what is known as cloud computing. Distributed computing was introduced in the 1960s as a method for a client to access a server and provide a command. The server could then access other computers located on the network to accomplish the task and return the results to the end user. The distinguishing feature of distributed computing is that each computer is connected (either physically or wirelessly) to other computers on the network. For service-oriented architectures (SOA), the main differences are that: 1.) most of the servers on a network are now connected wirelessly, 2.) applications are built from the services already defined, 3.) an interface exists so the end user can treat the system as a black box, and 4.) a communication protocol is used over the network so that each service can talk to another. Cloud computing is essentially a services-based architecture except that the services can be layered such that certain services can only communicate with other services in their stack, and users can pay only for the services they are utilizing which makes it cheaper than traditional SOA.

Johns Hopkins University Applied Physics Laboratory (JHU APL)

JHU APL developed small autonomous UAVs that would demonstrate cooperative behavior for simple surveillance missions. Bamberger et al. developed an architecture that would allow the UAVs to take off and land, as well as select targets autonomously. The UAVs were organized in teaming arrangements so that there are multiple teams of UAVs working together on various tasks. Although the algorithms for searching and selecting targets are based on optimization tools, the locations were not pre-programmed into the machine. As a result, machine learning was required to assist the UAV in determining where to fly to search for targets. This architecture also utilizes a multi-UAV Command and Control (C2) interface using an 802.11 wireless mobile network, creating the cornerstone of their modular system architecture. There are two layers of control, each with separate modules that are categorized primarily into "low-level" and "high-level" processes as shown in Figure 5.

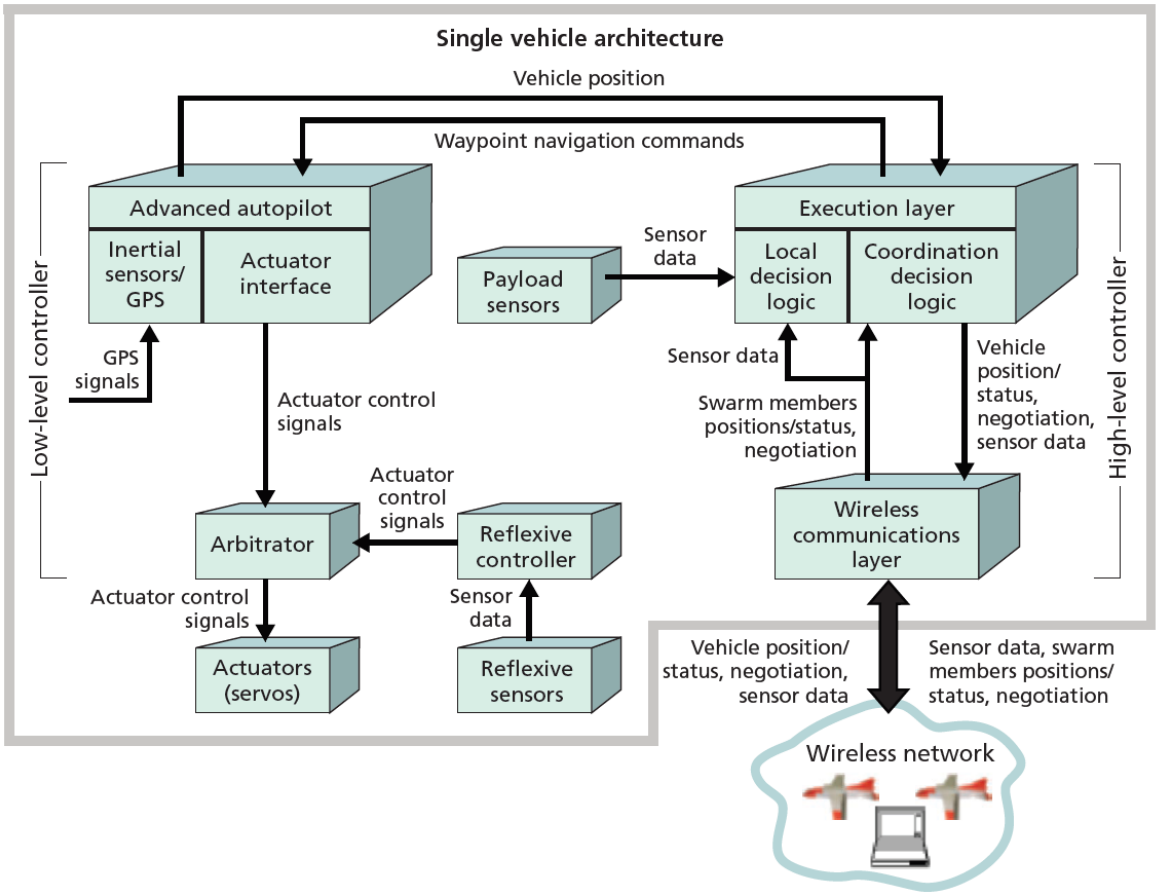

Figure 5. JHU APL UAV swarming modular architecture [28]

Wireless communication between the UAVs, the Ground Control Segments (GCS), and the human pilots are shown in the bottom right. One key note for this architecture is that it was primarily developed for smaller UAVs with a total volume less than 50 cubic feet [28]. Larger UAVs such as the MQ-1B Predator would require more power and thus the communication losses would be beyond what this network could handle. Additionally, this architecture was developed in the early 2000s as the first of its kind. Currently, most UAVs only carry a single communications payload and maybe two sensors at most, while downlinking to a GCS directly via line of sight (LOS). Moreover, UAVs are multifaceted with multiple radios and sensors, and

they can use satellite communication links to download data to a GCS. Therefore, an architecture that can support larger UAVs must be service-based to account for all the different missions and functions occurring simultaneously. APL did not require such an architecture because the UAVs involved were simple and there were not hundreds of them. However, as is evident in the following designs, larger UAVs require more messaging buses and flight control dispatches. It is imperative for larger UAVs to have services defined to prevent disruption or lagged processing due to large amounts of data traffic.

MIT Lincoln Labs

MIT Lincoln Laboratories (MIT/LL) developed software and data services for a UAV swarm. However, the goal for LL was to incorporate UAV information sharing for a variety of environments such oceanic flight, international airspace, tactical operations, etc. LL specifically focused on non-autonomous vehicles that were already pre-existing, so the architecture was only partially supplemental to APL's design. Around the same time, the Federal Aviation Administration (FAA) undertook initiatives to safely integrate UAVs into national airspace equipped with systems that would protect them and other aircraft. For example, the FAA wanted UAVs to carry a system called Traffic Collision Avoidance System, or TCAS, which is used on commercial transport aircraft. The plan was to incorporate sensors on UAVs to provide a similar Sense And Avoid (SAA) capability so that aircraft could avoid crashing into each other in mid-air. As a result, LL developed a service-oriented architecture with open standards for integral interfaces like SAA such that ground-based surveillance assets could be leveraged until a permanent solution was fostered. Ideally, SAA sensors will be located on every UAV to avoid collisions in the future.

This architecture accommodates both on-board and off-board SAA information. However, this makes the architecture significantly more complicated because the aircraft is communicating with different ground-based and airborne assets in attempting to perform SAA maneuvers. Therefore, the architecture must be service-based. It must be real-time and interoperate with ground assets. Data from the sensors, whether on-board or off-board, could be published to two buses, identical to the "low-level" and "high-level" design by APL. Some services like SAA would subscribe to sensor data and output its own product, while also requiring real-time throughput. Thus, this architecture divides services across the buses to ensure high quality for real-time services as shown in Figure 6 [29].

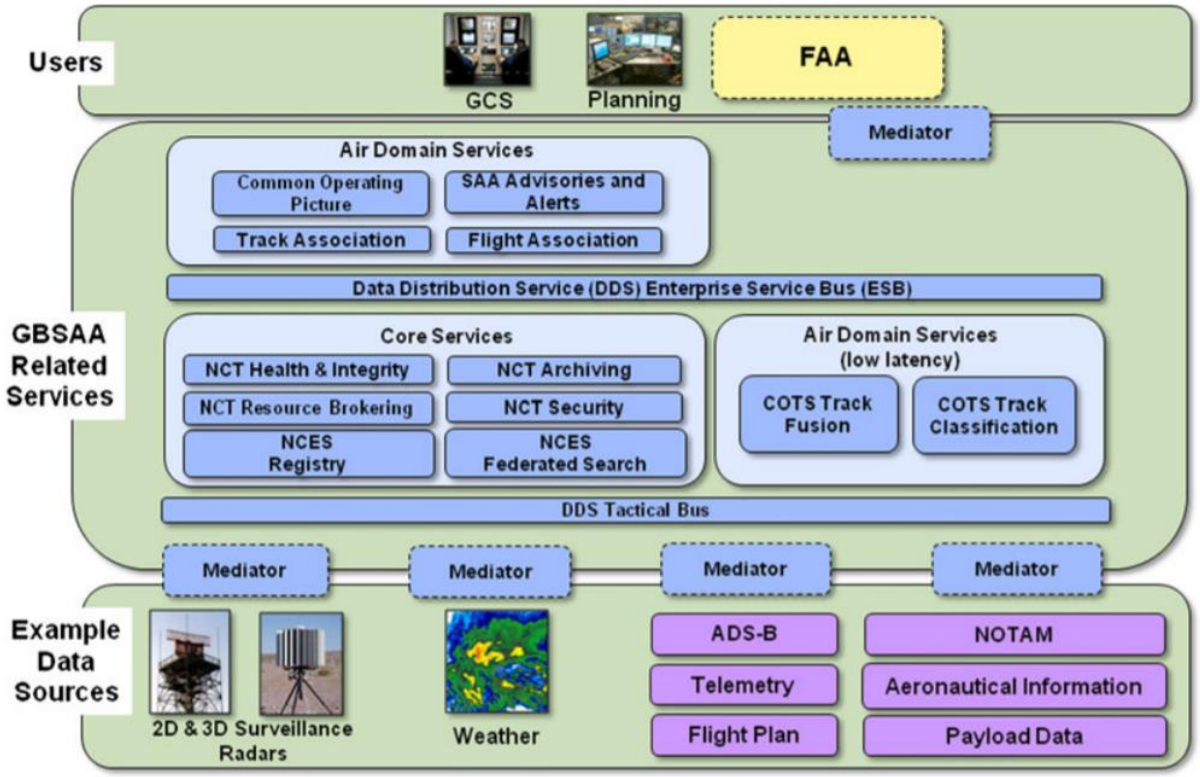

Figure 6. MIT/LL Reference Service-Oriented Architecture

MIT and Aurora Flight Science

Several professors, postdoctoral fellows, and graduate students at MIT teamed up in conjunction with Aurora Flight Sciences to develop a UAV swarm that can accomplish a Cooperative Search, Acquisition, and Track (CSAT) mission. Although the primary goal was to develop algorithms that would improve tracking and cooperative search between UAVs, the team also developed a modular open system architecture for multiple UAVs. This architecture is shown in Figure 7. Each UAV has a control module for mission planning and vehicle maneuvering. The UAVs may or may not have sensors depending on their size, but the control module can handle sensors based on the three different modules in it: the onboard vision module (OVM), onboard planning module (OPM), and auto pilot module (APM). The OPM generates environmental information for the UAV about current weather, pollution, wind patterns, etc. It also includes target detection and tracking data which is done by analyzing sensor information produced from the OVM. Without sensors, the OVM solely relies on the camera.

The OPM data from one UAV is transferred to another UAVs OPM and APM, which uses the data to generate navigation commands. The OPM data transfer can occur between any set of UAVs; it depends on an individual UAV's targets and waypoint selection. Target estimates are sent from the OVM to the OPM, and the generated waypoints are sent from the OPM to the APM. Vehicle state information is sent in a feedback loop from the APM to the OVM and OPM. Overall, the cooperative behavior between UAVs is accomplished through the sharing of targeting data and environmental parameters [30].

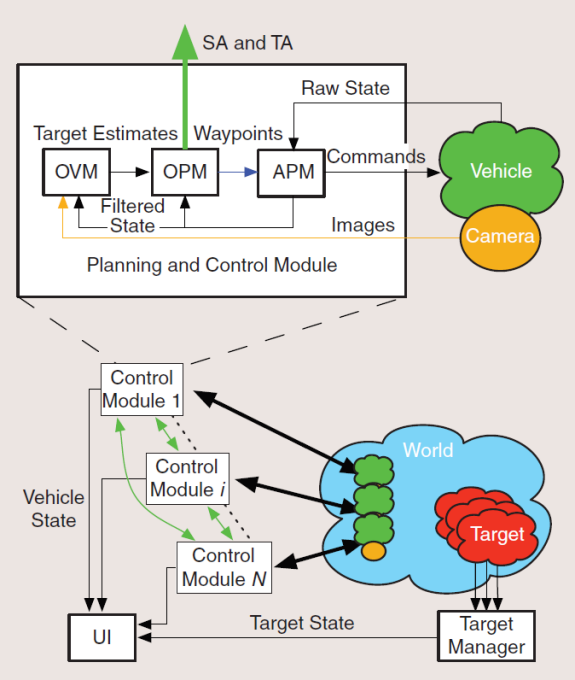

Figure 7. MIT and Aurora Flight Sciences Open System Architecture

Technical University of Catalonia, Barcelona, Spain

The Computer Architecture Department developed a service-oriented open system architecture that includes middleware logic focused on three primary services: Flight Services, Mission/Payload, and Awareness. The middleware also takes in information from communications links, sensors, actuators, and autopilot similar to the other architectures previously described. Each middleware element also consists of the hardware behind the logic. For example, Flight Services is responsible for the basic UAV flight such as autopilot management, flight monitoring for any end-users, flight contingency management, terrain avoidance, etc. Thus, this service would directly connect to TCAS or attitude hardware (elevators, ailerons, etc.). Mission/Payload is responsible for constructing the UAV mission, regulating the payload and surveillance area along the ground, processing Earth observation data and providing data to the end user. Payload services are low level services such as device drivers; this includes any object that facilitates the input, output, or transfer of communications or data.

From the overall mission perspective, there are five main components outlined in their swarming process and each one is described through a forest fire example.

1. UAV sensor platforms and control selection: First, a vehicle or a fleet of vehicles must be chosen depending on the necessary sensors and the cost objectives associated with the mission. In this case, a high definition visual camera along with a thermal camera are necessary to confirm hot spots in the region. The hardware will need storage capabilities to accumulate pictures and process them for determining the extent to which fire is spreading.

2. Ground teams will be able to control the UAVs in real time using LOS communication and have information sent to them. The UAV statuses, state of the operation, and surveillance data will be gathered, along with other information designated by the ground team. Hot spot management and fire detection data would happen here as well to complete the overall picture of what is happening.

3. Portable devices like iPads will be integrated into the wireless network to send and receive pictures, messages, videos, alarms, maps, commands, etc. They will act as information terminals that can channel data when and where necessary.

4. The ground clients are designed to control one UAV or maybe a few at a time. In the case that multiple UAV teams are operating on dispersed fires, then a Data Processing and Storage Center should coordinate their operation by taking in all mission data for high decision making and supervision. They would have more computer processing power and tools available to work quickly and store high volumes of data without wearing down the teams actively working the fires.

5. A reliable communication network between the UAVs, the ground stations, and information terminals is critical. The middleware will also buffer the incoming data for the UAVs to make sure high priority information is treated first, while also considering the bandwidth of the data and the relative cost of any given connection its making **[31]**.

An example of this high-level model is shown in Figure 8.

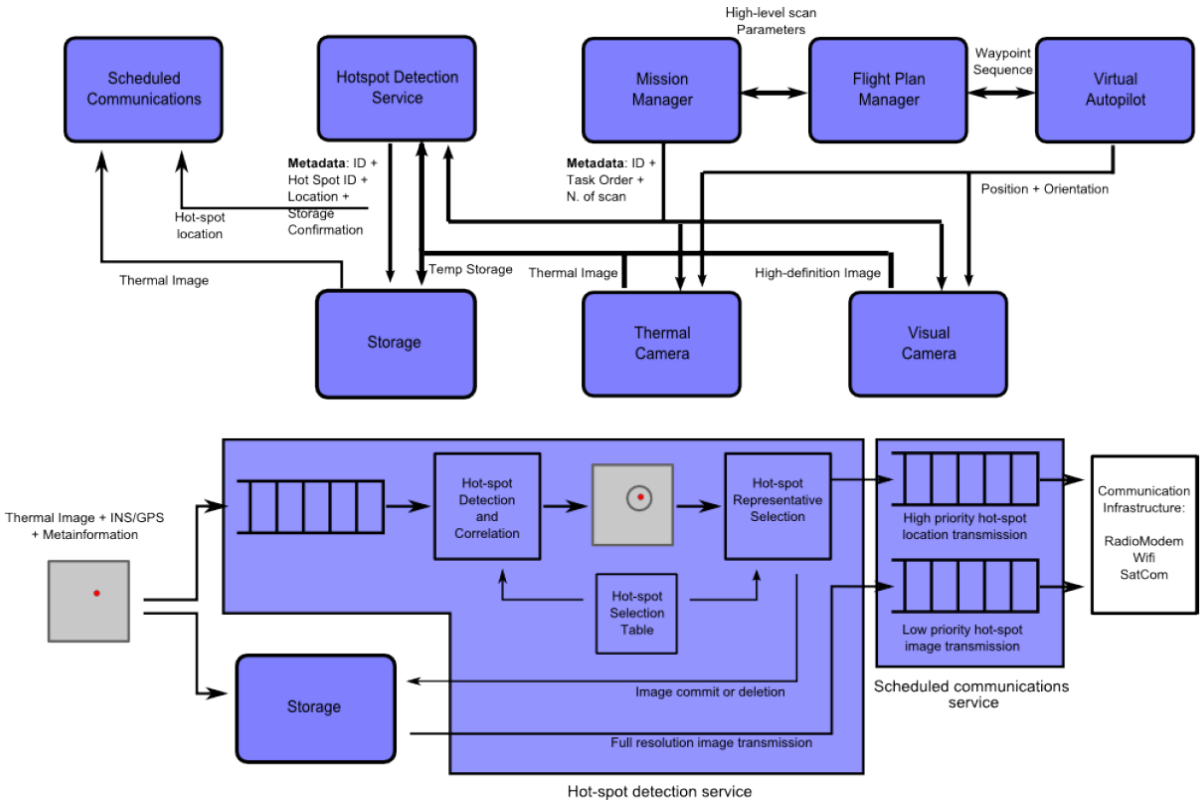

Figure 8. High Level System Architecture Example for Wildfires [31]

Each of these players interact within the services provided by the middleware. There are four main services, each with various subsystems.

1. Flight Services

    a. There are two main drawbacks to current commercial autopilot services for UAVs. First, the autopilot telemetry is extremely difficult to exploit. This is due to a need for keeping UAVs secure and not allowing hackers to gain unauthorized access to a vehicle. However, it has also made it difficult for UAV telemetry data to be sent to other UAVs within a swarm. Additionally, most flight plans are just a series of waypoints defined by the operator. Thus, the flight plan within this architecture is independent from the mission and payload that the UAV controls. As a result, the main subservices include:

        i. Virtual Autopilot Service: Sends information between the autopilot service from the operator and the mission/payload systems

        ii. Flight Plan Manager: Offers unlimited waypoints, waypoint grouping, emergency alternatives to structures flight plans, mission specific trip legs that encourage prioritization of the right missions. These legs can be modified without re-designing the flight plan.

iii. Contingency Management: Monitors critical parameters (battery life, flight time, fuel, etc.) to take action in case contingencies are needed.

iv. Electrical and Engine Management: Relays electrical and engine statuses to the contingency planner

v. Flight Termination System: Deploys a parachute for contingencies or in failure mode

2. Mission Services

a. Earth is the primary target of observation for UAV flight. The UAV selects a geographic area for navigation and must define the input sensors depending on what information it needs to take in. The navigation pattern over the surveillance area is given before take-off. There is a recognition service that can detect certain objects or movements in the area. There also scenarios where the service can re-define the mission area based on updated intelligence. There are 5 main subservices:

i. Mission Manager: Supervises the flight services and payload services. It performs actions predefined and is capable of modifying the flight plan by redefining parameters or integrating new legs.

ii. Real Time Data Processing provides image processing

iii. Mission Monitor provides current mission states to the end user

iv. Payload Monitor gathers health statuses on the payloads and is connected to the Contingency manager in case of malfunctions

v. Geographical databases are on board to provide data to services

3. Awareness Services

a. These services monitor the surroundings and overtake the management of the flight when there are critical conditions. There are 4 subservices involved:

i. The Awareness Data Function collects data about other vehicles near the UAV as well as terrain and meteorological conditions.

ii. The Tactical/Strategic Conflict Detection service analyzes the information from the Awareness Data Function to detect potential collisions or bad climate conditions.

iii. The Tactical/Strategic Reaction Service implements avoidance procedures depending on the severity of the conflict. It can overtake the Flight Plan Manager to perform avoidance maneuvers and then release control back to

the Flight Plan Manager. Some scenarios may warrant the reaction service to retain the same mission before the avoidance incurred.

iv. Awareness sensors acquire information about nearby aircraft

4. Payload Services

    a. This service is mainly for low level devices raw data acquisitions that need to be processed before use in real time analysis onboard the aircraft. These are created and adapted according to the mission and the end user's preferences. For common devices, there are several pre-defines services already in existence.

Computer Vision Group at the Center for Automation and Robotics

Many of the researchers from the Technical University of Catalonia also work with the Computer Vision Group at the Center for Automation and Robotics under the Consejo Superior de Investigaciones Cientifices in Spain. This team included an additional researcher, Professor Molina, who works in the Department of Artificial Intelligence at the Technical University of Madrid. This team developed a system architecture called *Aerostack*. This design is similar to the previous designs. The primary difference is that this architecture purposefully generates a communication layer that is separate from the sensors, processing, etc. In doing so, if the communication layer becomes corrupt, the other layers can still function independently. There are five layers involved: reactive, executive, deliberative, reflective, and social. The layout of the communication between each layer, as well as how the pilot communicates with the UAV, is shown in Figure 9.

1. The reactive layer happens in the present. It includes motion controllers and feature extractors that read sensor states and deliver vision and pattern recognition using various sensors. The feature extractor finds items like bumpers, extraneous colors, power lines, markers, etc. and integrates them into a sequence if necessary to form a larger picture. This information is sent to the controller to develop a desired position in terms of speed, altitude, yaw, aileron, etc.

2. The reflective layer helps supervise the other layers to see if the UAV is making progress toward the end goal. If problems arise, then this layer provides recovery actions.

3. The social layer represents the communication between each UAV and with the pilot in command. It overlays each layer to ensure that the proper controls and feedback are being sent at the appropriate times.

4. The deliberate layer provides solutions to complex problems generated during the planning phase. Additionally, thus layer will provide directions for other layers in terms of how they should communicate, what sensors will need to be used, what the recovery plan is, etc.

5. Finally, the executive layer generates behavior sequences for the reactive layer and integrates sensor information into a succinct form that can be read by multiple layers. The reactive layer cannot analyze the information it is putting together, so the executive layer provides functions to examine data as it becomes available.

There are three main systems within the middle layers of the diagram that provide a majority of the services. These systems are designed to optimize the autonomous capabilities and interactions with end users. As such, the systems attempt to provide the end users with significant information and ask for clarification when necessary without overburdening the operator with remedial tasks that the automation can handle.

1. The planning system generates goals to accomplish a mission. For example, if the goal is to pick up a ball and deliver it back to a starting location, then the first goal might be to get past a cloud of smoke. The next goal might be to drop altitude. The next goal would be to drop the arm to pick up the ball, and so on and so forth. There is a task-oriented mission planner that increases autonomy by providing trajectory and necessary forcing moments for the UAV, especially when missions are complex and take place in changing environments.

2. The executive system's behavior manager sequences all the prospective actions and translates them from a symbolic description to actual logic that the software can execute. This system performs the translation extremely quickly to minimize the time between when an action comes in and when it is performed by the aircraft.

3. The supervision system consists of failure detection, notification, and recovery. If any systems run into problems, this system will immediately notify the end user in case he or she must take command of the aircraft.

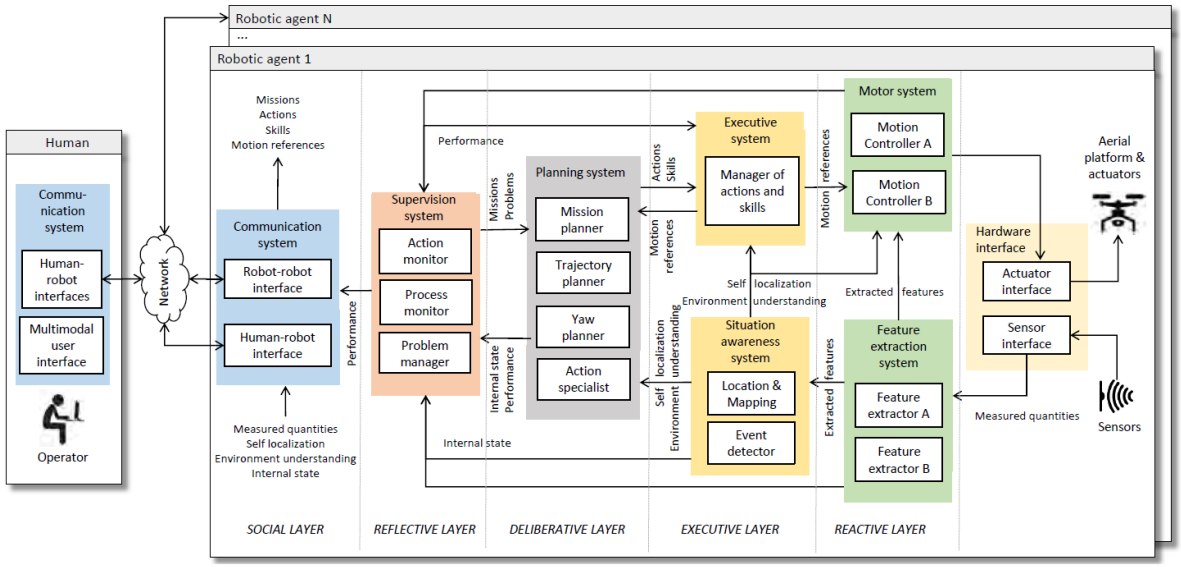

Figure 9. Main Components and Layers of the Aerostack Architecture [32]

Figure 10 outlines the general logical flow of the communication architecture for the global mission planner which is responsible for the entirety of the mission and the agent mission planner which is respective to each UAV in the swarm. The global mission planner provides individual goals to each agent mission planner knowing what the ultimate objective of the mission is. The steps in this flow are summarized below. The first set of steps are done within the global mission planner, and then the following steps are included with each corresponding UAV involved.

Global Mission Planner

1. Sample the mission zone: Divide the mission area into regions and develop check points or central locations within each region that the UAVs can use as waypoints.

2. Generate mission points: These are points that are critical to the mission. UAVs will visit or explore these points to physically accomplish the mission.

3. Distribute the points: Each of the points are distributed to specific UAVs in the swarm depending on which UAV is going to work on which task. This step is done per UAV such that no two UAVs will have an idea of where all the mission points are located; they will only know where their specific points are and not the other UAVs.

4. Task generation: Next a UAV is given a task depending on the points that are set for it. If the UAV is set for a certain distance or only has so much fuel, then that limits it potential tasks. Examples of tasks can be: take-off, hover, move to point, yaw, land, etc.

5. Mission generation: Now the tasks are arranged in such a way that the mission can be completed.

6. Send missions to UAVs: Each UAV is now given its mission, including the concerned waypoints and tasks.

7. Re-planning: The whole process will be re-run at least once, if not multiple times, to ensure that the process is optimized according to the UAVs and the mission. If any UAV responds to its mission parameters with an initial fault or other alarms, then re-planning will notify the human operator and come up with an alternate plan.

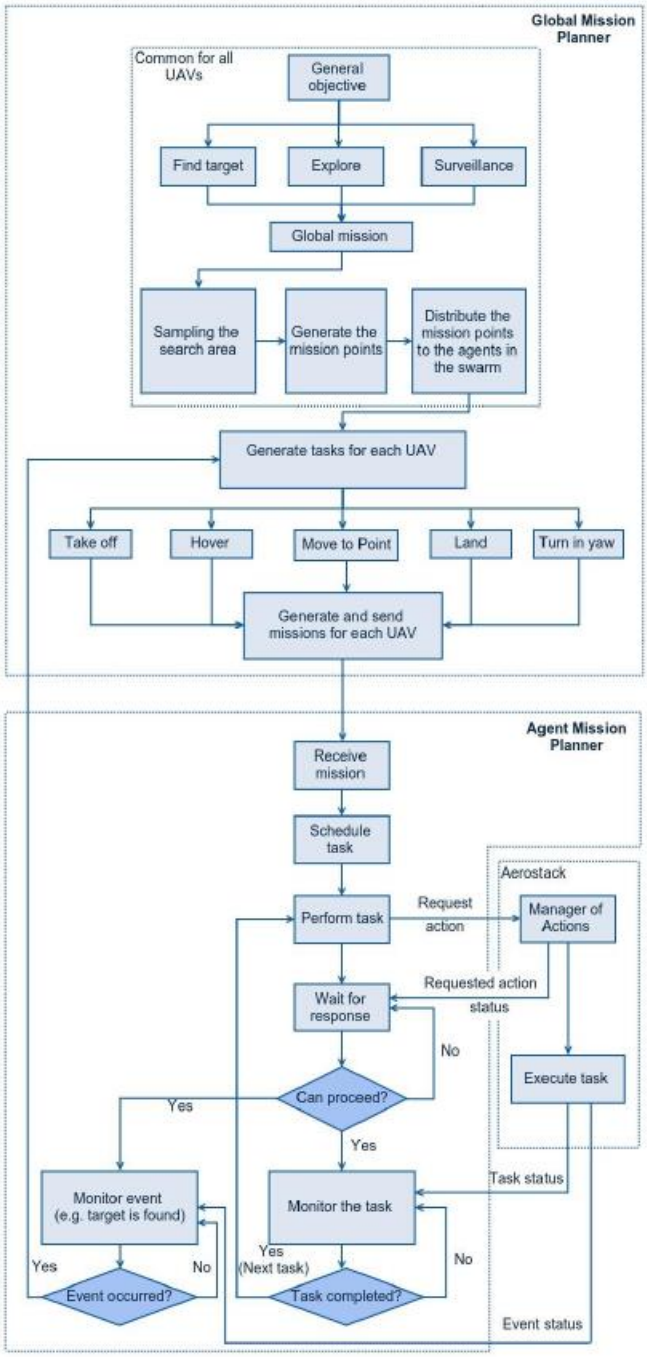

Figure 10. Mission Planning Communication Architecture [33]

Agent Mission Planner

1. Receive the mission: Each UAV receives the mission from the GMP.
2. Schedule the tasks: Each task in scheduled in order.
3. Perform the tasks: A task is performed.
4. Monitoring: There is monitoring for each task such that once a task is completed, then next one is performed.

Both mission planners are iterative. As the global mission planner updates, so does each individual agent mission planner.

The Institute of Networked and Embedded Systems

A team of researchers composed of government and industry officials from the Institute of Networked and Embedded Systems in Klagenfurt Austria as well as Lakeside Labs worked to create a set of requirements specifically for a UAV network. This paper set out to identify the ideal network based on a specific UAV swarm and the mission set(s) given to it. The comprehensive analysis evaluates technologies and their limitations, as well as recommendations for which technologies should be used in certain scenarios. There were several different technologies considered, as well as dozens of parameters like standard, spectrum type, communication range, and latency to name a few. With the numerous options available and the innumerable amount of missions that may exist, the authors conclude that the combination of technologies to use depends entirely on a large set of parameters that are difficult to calculate. Each technology has specific limitations, so deciding to use Commercial Off-The-Shelf (COTS) technology is not necessarily the best option depending on the needs of the mission [34]. A summary highlighting some of the major differences between the technologies can be seen in Table 3.

Table 3. UAV Communication Technology Details

Technology	Standard	Spectrum	Mobile	Range (m)	Latency (ms)	Max # nodes	Reference
Bluetooth	802.15.1	Unlicensed	Yes	150	3	Not defined	[35]
Zigbee	802.15.4	Unlicensed	Yes	10-100	15	65000	[36]
WAVE	802.11p	Licensed	Yes	1000	100	Not defined	[37]
GPRS	GPRS	Licensed	Yes	5000	500	Not defined	[38]

The authors also analyzed a variety of potential mission sets for civilian UAV applications and surveyed missions from 2000-2015 to understand what ranges previous mission sets used to assist international airspace regulators in defining permanent communication ranges for certain UAVs or missions. The authors primarily evaluated two factors: device autonomy (should the

UAV control be centralized or decentralized?) and mission autonomy (Is there a base station that acts as the primary decision authority?). For each potential mission type, they list the constraints, assumptions, and mission requirements to understand why certain network bands are necessary for certain missions. Although safety is not the primary concern, it is included as a factor in determining the network bands. However, this analysis made assumptions about certain mission types in order to derive communications-specific bands. These assumptions are solely based on prior mission sets, and since these missions are primarily civil, this will most likely not apply to the military [39]. Military environments include several other factors such as jamming, Electromagnetic Pulse interference, radiation, and other factors that would cause communications as well as on-board payloads to become damaged throughout the mission.

The Rand Corporation

The RAND National Defense Research Institute, a federally funded research and development center, developed guidelines for an open source architecture for the Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics in 2014. This study was in response to a request from the Defense Science Board to address dilemmas regarding unmanned systems (UxS) command and control (C2) structures. For this research project, UxS includes unmanned ground vehicles (UGVs), unmanned maritime vehicles (UMVs), and UAVs. The increasing number of unmanned systems puts a large burden on communication networks to handle the data traffic. Additionally, these systems can be deployed in extreme environments where they would require greater autonomy to operate more effectively. After discovering that the UGVs and UMVs were largely proprietary and that their navigation problems differed significantly from UAVs, the RAND Corporation decided to focus exclusively on UAVs for the project. Thus, the main objective was to develop an architecture that 1.) improves UAV to UAV interoperability, 2.) provides greater autonomy to UAVs, and 3.) allows for cooperative UAV behaviors so UAVs can work in teams to successfully complete complex missions in extreme environments.

The RAND Corporation believed the best answer to accomplishing these mission goals is by developing an open architecture system. Some of the notable challenges of this study are:

- **Currently, the Department of Defense Architecture Framework (DoDAF) 1.0 products do not contain information needed for interoperability or to predict the occurrence of interoperability problems.**

- **There needs to be a separate architecture developed depending on the vehicles involved considering each vehicle (UAV, UGV, and UMV) possess major differences in the design, capability, and function.**

- **A UAV Task Force Interagency Team called the Horizontal Integration Working Group developed an initial version of a Joint Common UAV Architecture (JCUA) that was useful in identifying UAV gaps relating to controlling UAVs but did not contain ground information about transmitting data or how to communicate with aircraft. A separate working group developed an algorithm for the ground control**

station communications to improve UAV-GCS interoperability but did not address common UAV problems. These two architectures were developed separately and did not correspond such that combining them would create a better architecture.

- **The syntax used by different groups varies widely and creates complications in communicating what a group has done and what they are working on**

In response to the first and third points above, The RAND Corporation recommended developing a partially open system architecture (POSA) that would support a conglomerate of potential UAV manned-unmanned systems. This would not only improve interoperability and component reuse, but the autonomy of the vehicles would increase significantly. Additionally, if open interfaces were generated then the autonomous capabilities could be used for any set of UAVs interacting together. One of the great trials in early software architectures for autonomous and semi-autonomous unmanned systems is that there are a variety of modular schemes for architectures, in addition to varying software foundations. For example, one architecture may use a central database to send and receive messages whereas others may use a multitude of servers that exchange temporary memory depending on a UAVs location. In other words, if the UAV moves closer to another server, then that server can request greater storage from other servers to be able to handle any data the aircraft needs to downlink. The advantage of centralized databases is that they can provide a history of all messages efficiently without needing to query multiple servers for information. However, decentralized servers can provide real time performance which is important since UAVs move quickly and updates on their position, velocity, attitude, etc. are critical to operations.

The RAND Corporation analyzed the architectures developed by MIT LL, the MIT CSAT staff, and JHU APL to develop a set of essential modular components that would support multiple levels of autonomy and that are consistent with the principles of a POSA. These components were identified as common elements of each architecture and they support two levels of system control in what is termed a Technical Reference Model (TRM). The TRM achieves highly responsive vehicle control using an Auto Pilot Module which is connected to the flight control computer and sensors through a high input bus. An additional second bus supports services that do not require real time objectives such as regular air domain services (processing air traffic information, SAA advisories, etc.) as shown in Figure 11.

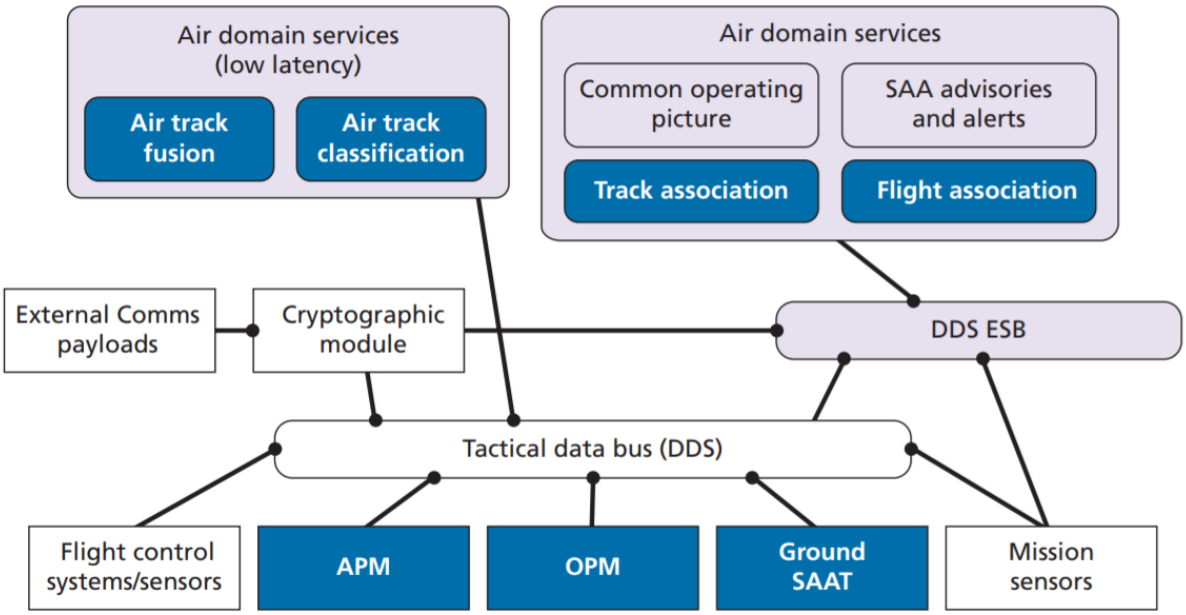

Figure 11. The RAND Corporation Recommended UAV POSA

The blue modules allow for autonomous capability and are open interfaces so other people can alter the software to fit their specific system. The main benefit of this schematic is that the autonomous capabilities are separated from the vehicle platform. This reduces the UAV communication demands and eliminates the need for real time communications in extreme environments by removing remote pilot control [40]. Another significant note is that the purple boxes are services that may not be needed. These boxes are required for large, high-value UAVs flying within FAA-controlled airspace that necessitate defensive countermeasures against aircraft or ground threats. If a UAV is small, flying at lower altitudes, not in FAA-designated airspace, or does not have the indicated sensors and warning systems, then these services may be superfluous.

Three important notes based on this design with respect to this thesis are:

- There are no requirements generated from the study. The author solely made recommendations for a design based on their expertise and understanding of the previous designs developed by other groups. However, the requirements used to acquire the design are unknown or may not exist.

- There is no systems theory applied to the recommendations. The authors were given a problem by the DOD and used their expertise to give a solution. They did not analyze the problem to understand if it truly covers the scope of UAV systems, nor did they apply systems theory to arrive at their recommendations.

- This architecture applies to a limited number of UAVs with a specific mission scope. Since the DOD was the primary customer, the authors had in mind missions for the DOD

that are usually targeting or search and rescue missions. However, as noted in the background, UAVs are being explored for use in hundreds of civil applications. This may have affected the extent to which the authors provided an architecture suggestion for all applications.

To conclude, each of these designs has specific limitations. First, safety is not designed into the systems. Each system was developed under a specific set of assumptions (UAV in autopilot, UAVs fly at varying altitudes, etc.) and older hazard analysis techniques were applied. However, this does not account for software design flaws or organizational issues that may affect the design. By applying STPA to a generic MUM-T architecture, this analysis will identify how to properly design a swarm architecture that is safe when coordinating with a manned aircraft. Moreover, this analysis provides requirements for any future UAV architecture. Most engineering systems have a specific goal or set of goals that are specified ahead of time for the system to accomplish. However, there have not been any systems that might have an unclear mission set. By using STPA, MUM-T can integrate the requirements to be safe and secure based on the mission selected for analysis. The STPA can be re-run for different mission sets to understand architecture differences that would be needed depending on the mission.

Secondly, most of these architectures rely on decomposition as a method for constructing the system by breaking it into different pieces, constructing the pieces individually, and then putting them all together to make the system. This idea can be traced back to the Ancient Greeks who used to build ships with teams constructing separate parts and then combining them at the end. This method of design works well for assembly line type operations where teams are developing the same parts and there is only hardware involved. However, when humans are interacting with automation and software, designing each subsection assumes that these different groups act independently which is not the case.

As a result, STPA is applied to MUM-T to identify safety and security requirements for an entire system including the software, manned aircraft, Ground Station, and more. This will also highlight safety and security concerns within the interactions between UAV(s). Before diving into the STPA, it is important to understand what happens during air to air combat, air to ground operations, Intelligence, Surveillance, and Reconnaissance (ISR) missions. This will set the groundwork for the STPA.

CONOPS

Although not necessary for the sake of the STPA analysis, Figure 12 and Figure 13 show CONOPS diagrams as examples for the loyal wingman case. Figure 12 displays various UAVs that are searching for intelligence or a target within a specific region as the Team Lead provides commands from behind the UAVs. This could also be an air to ground mission where the UAV(s) or Team Lead can fire at ground targets upon finding the appropriate target. Figure 13 displays an air to air combat situation where the UAVs provide detection of the enemy and allow for targeting before the enemy comes within firing range. The CONOPS are important because they will identify what environments the STPA should focus on, what other players may be involved in the control structure, how other aircraft may become involved in the mission, etc.

Figure 12. ISR Mission

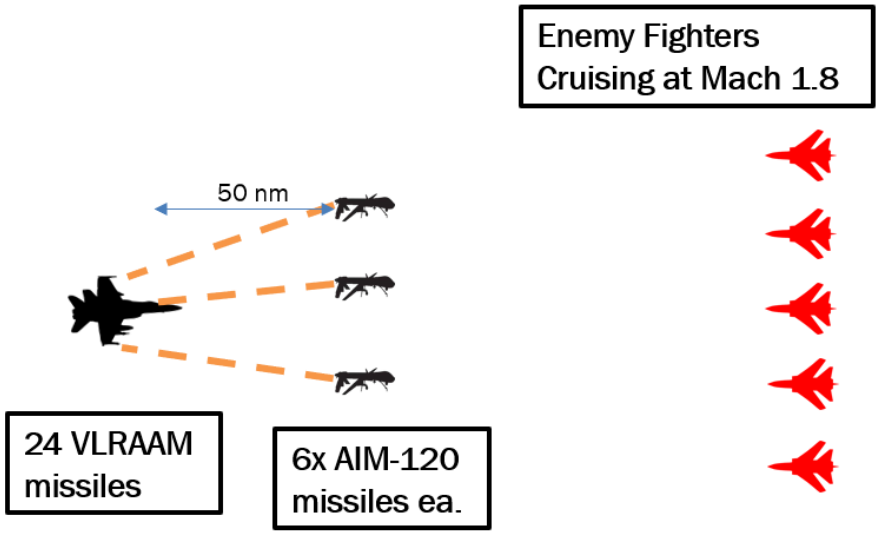

Figure 13. Air to Air Combat

Fighter Pilot and Remote Pilot Mission Process

Fighter pilots and remote pilots have similar mission processes in terms of preparation, execution, and debriefs. This section will describe the process and any differentiations that exist.

Preparation starts a day before the mission. Pilots all eat together, sleep together, work out together, and even go to the bathroom at the same times. They act as a unit and make sure that each person is taken care of and prepared for a mission at any time. The squadron's scheduling shop will usually post a schedule of the week including the training rides, ground events, sorties, etc. Therefore, pilots know with reasonable accuracy what they will be doing during any given week. Upon arriving at the squadron, it is time for mission planning.

The mission type and position within the mission (two-ship, four-ship) will determine the type of mission planning. Air to air combat vs. an air to ground practice ride require different planning

procedures because the main objectives of each flight are different. Mission planning can take up to 5 hours depending on the complexity of the mission. Typically, a mission is planned on a computer with specific airspace boundaries, flight routes, communication protocol, and weapons loadout information on a data transfer cartridge (DTC). The DTC is loaded into the jet during ground operations and transfers all of the data into the aircraft's computer system.

Pilots will also receive paper maps and lineup cards. A lineup card contains critical information necessary to complete the mission such as call signs, takeoff and landing weights, mission time, code words, weapons configurations, radio frequencies, etc. Especially if tanker aircraft or hot ranges are needed to time everything perfectly, then even more planning is required. After mission planning comes the flight briefing. At the flight briefing, all team members come together and lay out the details of what will happen and how it will get done. The brief is led by the flight lead, and usually takes place two hours prior to takeoff.

The flight lead will begin by briefing the sorties with no interruptions; all questions are held until the end. There is an administrative and a tactical portion to the brief. The administrative piece addresses the basics of ground operations, radio frequency plans, recovery plans, formation positions, etc. This is a simple overview to remind pilots to do certain things in certain events and always remember "Be on X frequency during takeoff" or "Be in formation until an enemy does X". The latter statement involves the contracts for each pilot. Each pilot has a contract that describes what they are responsible for and must accomplish no matter what. A popular contract is announcing any limitations in weapon capabilities such as "one missile left" or "weapons storage bay jammed".

For missions with unmanned aircraft, this portion of the brief focuses less on exact sorties and instead on the exact targets that should be hit. For air to air combat, formation is critical. Unmanned systems cannot operate in formation due to limited technology, so remote pilots are more worries about aiming and firing at the correct target than trying to hold a formation with a flight lead.

The tactical portion describes the details of the in-flight execution. It could be as simple as how to release a weapon to more complicated maneuvers that the team has been practicing and when that maneuver could be executed in flight. Once the brief is complete, the flight lead wraps up with the final objectives and pilots ask questions. After the brief, there is about an hour before takeoff and it is time to suit up. The team grabs their mission materials, dresses in their G-suit, prepared their helmet, and heads to the Operations Desk. This is where a senior member of the squadron will work with maintainers to make sure the right aircraft are in the right configuration, any weather changes are accounted for, schedule changes are addresses, pilot illnesses are not precluding, and more. This is essential for ensuring the daily flow is smooth and all missions can be executed in a timely manner.

Finally, after the step brief, pilots arrive at their jets. They perform a walk around with the Crew Chief and the maintainers, following a set of checklist procedures to ensure the flight controls, hydraulic lines, tanks, weapons settings, etc. are ready for flight. Then it is time to climb in, taxi out, and takeoff. After takeoff, jets will need to rejoin the formation and proceed to the working airspace. Along the way, a "fence check" is done where the aircraft is made ready for flight. This

could include setting the radar or targeting pods, testing flares, turning volumes up or down for different alerts, ensuring the oxygen is flowing, etc. There is a camera recording system that films the pilot along with all displays for later debrief. UAVs also have recording systems that film different sections of the UAV as well as its environment. Missiles and bombs can also be warmed up depending on the targets and missile technology.

Next, the real mission begins. Upon entering "fragged" airspace, the pilots may have air to air engagements with adversaries. This could include performing high G turns, dropping bombs on targets, targeting different objects, and more. Some sorties will have multiple different actions in them, but most of the time, pilots will focus on one segment of training at a time. For example, all air to air intercepts will happen in one sortie while basic air to ground weapon deliveries happen in another. The flight lead carefully watches and listens to everything that happens during the mission for the debrief later. Any imperfections must be corrected to ensure each mission is executed as flawlessly as possible.

In the case of air to ground missions, the remote pilots are also at the whim of any flight lead or manned aircraft in the vicinity. If a flight lead requires a UAV to move to perform their mission, then this requires the remote pilot to move the UAV to a different location. These missions also tend to require less G turns than air to air combat unless a significant counter attack occurs using surface to air missiles (SAM). For ISR missions, pilots tend to focus more on holding formation to prevent adversaries from knowing where they are. Pilots will focus on maneuvering the aircraft to receive the best angle for the sensors or radars to pick up signals and visual images from ground units.

Once the formation is low on gas and "bingo" fuel is reached, it is time to return to base (RTB). Bingo fuel means that the planned fuel state that allows for a normal recovery back to the airfield with enough reserve fuel has been reached. The formation will join back up, fence out, and head home. On the way, any armed weapon systems are "safed" up, meaning the master arm switch is off so that weapons cannot be fired. A battle damage (BD) check is done where each aircraft will fly behind and under another to look for any holes, stuck weapons, leaking hydraulics, hung flared, etc. If everything looks good, the formation can land, taxi back to the maintenance crew, and shutdown the aircraft so it can be prepared for the next team.

After landing, the pilots will fill out paperwork, get out of the flight hear, stow equipment, and eat before the debrief which is scheduled an hour after landing. The debrief will highlight any mistakes that were made and how to correct them for next time. Typically, pilots will playback all critical events that occurred and take mental notes. Critical items include weapon release timings, airspace violations, training rule mishaps, etc. Instructor pilots will review the tapes for each pilot and prepare the room for debrief.

The Instructor pilot (IP) or flight lead will go one by one for each flight member and examine their actions. Each member will watch it on screen and the IP will identify exactly what happened. The why and the how of each action is the main purpose of the debrief so no one makes the same mistake every again. Some debriefs can take three to four hours depending on the number of mistakes [41].

Remote pilots debrief from their remote locations with other manned aircraft, but the conversation looks very different. Remote pilots are seen as being of the same rank as fighter pilots, so the conversation tends to focus on where everyone can improve instead of focusing on just the wingmen.

These processes are important to understand because unmanned aircraft will significantly change not only the actual mission process, but the pre and post mission procedures as well. One cannot brief or debrief unmanned aircraft. They can be provided training sets (in the event of machine learning) and the algorithms can be run and re-run to get desired results, but one cannot reason or improve an unmanned aircraft immediately before or after a mission unless it is an easy software bug fix which is rarely the case. The STPA will highlight a plethora of the issues that occur during the mission, but additional studies can focus on efforts outside the actual mission to understand how mission planning and other activities are affected by using unmanned aircraft as autonomous wingmen.

Mission Specifics

This section describes several significant characteristics of combat and ISR missions for both manned and unmanned aircraft. This list does not identify all significant pieces of information and it is not all inclusive, but it does provide a succinct overview of the important pieces of information [42] [43] [44].

- Shoot List
    - Each pilot manages a shoot list in their cockpit. The shoot list identifies which airplanes a pilot wants to target. However, a pilot will also put airplanes on the shoot list to track airplanes and receive information on their velocity, position, altitude, and more. Typically, a pilot will put hostile planes on their shoot list as long as there is evidence indicating the plane is hostile and not friendly, the pilot has the best kinematic position to take out the hostile, the hostile is on a weapons quality sensor track, and the hostile meets the rules of engagement for firing. Current tactics state that pilots must put an object on the shoot list when it is TBD miles away.
- Flight Management System (FMS)
    - An example of a generic FMS is located in Figure 14. The Yellow targets represent hostiles that returned a hostile IFF transmission. The white targets are neutral because the IFF transmission was either not returned, returned a civilian response, returned no response, or returned a response that was neither hostile nor friendly (i.e., an unknown response). The blue circle targets indicate that another team member has a target designation and will fire at it if all conditions are met. If the pilot provides a target designation, then a target will turn white with a white circle around it. When the pilot fires at the target, it will turn red. Targets that have not been designated will flash on the screen until the pilot designates them.

Figure 14. Generic FMS Display

- Rules of Engagement (ROE)
    - Each pilot has a rules of engagement (ROE) that they follow. The ROE describes in what circumstances a pilot can designate and fire at targets. However, there are circumstances where a pilot is unsure of what action to take and must ask for assistance from the general to fire at certain targets.
- Identification Friend or Foe (IFF)
    - There are two parts to IFF: technological and procedural. The technological aspect also has two parts: cooperative and noncooperative. The technological part is when the pilot's IFF transponder sends out a signal to other transponders. Each friendly aircraft will return a signal with appropriate crypto and frequency to be identified as a friendly. Hostile targets are only classified as such when the IFF transmission is similar to a frequency and crypto located in the pilot's FMS database for hostile IFF transmissions. Neutral targets like civilian airliners return known frequencies without crypto. All other targets with frequencies or crypto that is unknown will be classified as unknown.
        - The cooperative aspect is similar to modes 4 and 5 on the transponder where an aircraft is communicating over the friendly data link using the appropriate crypto.
        - The noncooperative aspect is when the pilot's radar is analyzing electronic emissions from another aircraft to see what type of transmissions they are giving off.
    - The procedural aspect involves visually assessing a target to confirm its identity. This could involve a pilot flying behind a potential target or having UAV(s) fly with specific heat radars or doppler radar to measure the velocity or engine components of an aircraft to identify it.

- Data Link (Link 16)
    - Friendlies share data across Link 16. This include targets, aircraft, ground objects, terrain, and more. This allows each pilot's FMS to update based on the most current information available. Aircraft, ground units, and ships can all have a real time tactical picture. Link 16 also supports the exchange of text messages, imagery data and provides two channels of digital voice. Link 16 used to be solely used for line of sight communication, but with TCP/IP advances, satellites can pass data over long distance protocols.
- Weapon track
    - Each object that appears on the FMS can be on the shoot list to provide situational awareness for the pilot. Each target produces a weapon track which is a trace of where the object was, where it currently is, and in some cases, where the object is heading. This allows the missile to more accurately geolocate the target. The weapon track updates based on radar signals that are received. Therefore, the track might transfer in the middle of firing a missile because it takes time for the pilot's command to fire to reach the missile. However, the missile can still track the target to its new location. Advanced Medium Range Air to Air Missiles (AMRAAM) can use radar while in flight and be data linked to the fighter aircraft to tell it where to go. Once the missile is fired, it sends an initial message and a few seconds later, it begins data linking. Therefore, once the track transfers, the aircraft can provide an updated location for the target. If there is an extreme location shift, the missile may not be able to handle the shift. Other friendly aircraft besides the one that fired the missile can provide track updates to the missile.
- UAV launch
    - Currently, when an RQ-1 (Predator) or MQ-9 (Reaper) needs to take off, ground personnel walk the aircraft to a point on the runway. A trailer near the airport with line of sight communication provides the takeoff commands. Then, a handoff takes place once the aircraft reaches steady level flight where mission control takes over using beyond line of sight via satellite links. Once the mission is over, the aircraft is sent to a pre-programmed point and the controller near the landing point will take over and land the plane. If the satellite link is lost during the mission, the aircraft performs a lost link profile where it is automatically armed safe, so it cannot fire, and it flies to a pre-designates safe point where it will circle and wait for a local team to connect using line of sight for launch recovery. The pre-programmed course uses altitudes and holding points and announced to everyone what it is doing (i.e., which track it is flying).
- Global Positioning System (GPS)
    - The RQ-4 (Global Hawk) does takeoff and landing using differential GPS (DGPS). This means the maintainers walk the aircraft to a point on the runway and mission control provides an input that allows the aircraft to takeoff or land autonomously. There are no sticks or throttles to control the aircraft. DGPS uses a known fixed point to adjust real time positions. The MQ-9 and RQ-1 use regular

GPS. The aircraft can use an inertial navigation system (INS) as a backup if GPS is lost.

- Common Line
    - During a mission, aircraft act along a common line. Each friendly aircraft will ensure they are flying within a certain distance of the line so that they are in front of and behind the correct aircraft. This keeps weapons deconflicted and keeps friendlies from flying in front of each other. For example, if hostiles are located due north, then the common line will run from east to west.
- Main Flight Computer (MFC)
    - The MFC takes in control inputs or commands from the pilot, and implements the command using aircraft control surfaces, sensors, etc. One important note for the MFC is that it prefers to skip an action as opposed to delaying the implementation of the action. In other words, the MFC prefers to skip actions that cause delays instead of trying to force the command before others. The MFC will never process too slow – it would rather not process. The benefit of this is that the computer will continue functioning and updating at the right speed. The disadvantage is that the MFC may be operating with outdated information (2 or 3 seconds behind) so the weapon tracks could be completely wrong. The MFC can be reset; however, it takes 4 minutes to perform. Unfortunately, pilots reset the MFC on a regular basis. The MFC ignores redundant commands. If the pilot provides a target designation but it is already provided to a target, then the MFC ignores the command.

Chapter 3

MUM-T Mishaps, Hazards, and High-Level Safety Constraints

Before diving into the STPA, there are several assumptions that must be defined about this system. First, this MUM-T system will use a manned aircraft as the Team Lead (or flight lead as described in Chapter 2) as opposed to software or other controllers. Second, this system definition will include a Crew Chief (CC), Air Traffic Control (ATC), Mission Planners (MP), an Autonomous Controller (AuC), and a Ground Station (GS). The Crew Chief acts as the head of tactical aircraft maintainers, coordinates the aircraft's care, and calls in specialists (like avionics or propulsion technicians) when a problem is found. Air Traffic Control will provide coordination between aircraft to prevent air and ground collisions. The Mission Planners provide mission updates and prepare pilots during the brief. Finally, the Ground Station provides tactical coordination for changing targets or locations and may also control unmanned systems when necessary using remote pilots.

For the purposes of this system, the Ground Station is assumed to include remote pilots that can perform LOS or BLOS with an aircraft. There is no distinction between remote pilots at different locations because functionally they are performing the same control actions. Finally, the Autonomous Controller provides hardware actuation on the UAV(s) based on commands from the Team Lead or the Ground Station. For this analysis, the exact details of the autonomous controller are irrelevant. The exact location or decision-making capabilities do not change the analysis. Whether it uses machine learning or not, whether it is on multiple aircraft, on one aircraft, or just on the ground does not make a difference. The key is that functionally, the AuC is implementing commands from the GS and TL by actuating hardware on the UAV(s).

There are other controllers that could be included in the system such as maintenance, Generals, etc. However, this system did not include those controllers because the focus is primarily on interactions between the Ground Station and the Team Lead. There are numerous coordination issues between just these two controllers during the mission, and no other hazard analysis technique can analyze these interactions like STPA. Other methods can evaluate issues with maintenance or communication breakdowns between a general and the Team Lead. However, only STPA can truly dissect the problems that might occur when a Team Lead and remote pilots at a Ground Station are trying to ensure UAVs are doing the right thing at the right time.

The STPA will be done in several steps. The first step is to identify mishaps and hazards for the system. The second step is to develop a functional control structure. The third step is to develop unsafe control actions (UCAs). The final step is to develop causal scenarios and requirements for the system. Some STPAs will include an additional step which involves coming up with recommendations to eliminate UCAs. However, this analysis does not require this step.

Identifying mishaps and hazards is important to set the goals of the analysis. To start, the stakeholders must identify the mishaps of importance to them. They may prioritize them if they want. Potential mishaps for a MUM-T mission include (but are not limited to):

1. M1: Death or injury of a person (includes ground system personnel, Team Lead, civilians, friendly forces, etc.)
2. M2: Destruction or damage to aircraft
3. M3: Non-achievement of mission
4. M4: Ground property damage (either U.S. Air Force or civilian)

M1 includes any loss of life of that is not the target. It could even include specific enemies that are not being targeted for the mission. M2 includes damage to any friendly aircraft, whether it is in the air or on the ground. M3 can include several different losses, but generalized, is the non-achievement of the mission. Examples include not firing at a specified target, dropping a captured component, losing intelligence on a region, etc. Additionally, there can be several sub-missions during a mission, so a sub-mission loss is included. Assume the primary mission is to find an enemy's location and a sub-mission is to not be seen by the enemy. The friendly forces can find the location but maybe the enemy steals intelligence from the UAV in the process on their whereabouts. This could be considered non-achievement of mission because the goal was to find the location without being seen. M4 can be damage to Air Force or civilian property. This may or may not include enemy territory depending on the mission. Sometimes firing at enemy territory is authorized, and sometimes enemy territory must be avoided to prevent a greater war.

There could be additional mishaps. For example, another potential mishap could be loss of mission critical data, including intelligence data, weapon codes, etc. This could be included in M4 if the purpose of a mission is to complete a set of tasks without losing data during the process, whether it is inadvertent or an enemy gains access to that data. However, defining it as a separate mishap would highlight specific instances where this mishap occurs and how it can be prevented. Another potential loss is the release of pollutants. This could occur if weapons are involved with the mission, or if the target of interest for the swarm is a nuclear reactor. This mission set could also be covered under M2, damage to ground property. If the property happens to include nuclear materiel, then this loss will be highlighted in M2. However, these mishaps are not included in the analysis because these are considered less severe or less significant than the previously defined mishaps. Missions rarely involve nuclear materiel and loss of data is not as severe as losing an aircraft or an entire Ground Station so this is the reasoning for the four mishaps.

An additional note is the mishaps are in line with how the DOD defines a mishap as shown below:

From MIL-STD-882E: A mishap is an event or series of events resulting in unintentional death, injury, occupational illness, damage to or loss of equipment or property, or damage to the

environment. For the purposes of this Standard, the term “mishap” includes negative environmental impacts from planned events.

From Engineering a Safer World: An undesired or unplanned event that results in a loss (including loss of human life or injury, property damage, environmental pollution, and so on).

The hazards associated with the mishaps are:

1. H1: Aircraft violates minimum separation from other aircraft or terrain [M1, M2, M3, M4]
2. H2: Aircraft loss of control (includes departure from stable flight) [M1, M2, M3]
3. H3: Aircraft does not execute planned operations [M3]
4. H4: Aircraft departs approved airspace (where approved airspace is defined by mission planning and ATC) [M2, M3]
5. H5: UAV fires at friendly forces [M1, M2, M4]
6. H6: Team Lead fires at friendly forces [M1, M2, M4]

Once again, the hazards are defined similarly when comparing ESW and DOD definitions.

From MIL-STD-882E: A hazard is a real or potential condition that could lead to an unplanned event or series of events (i.e. mishap) resulting in death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment.

From Engineering A Safer World: A system state or set of conditions that, together with a particular set of worst-case environment conditions, will lead to an accident (loss).

H1 specifies that the aircraft should not come with a certain distance of other objects. This can include trees, mountains, waterfalls, buildings, other aircraft, etc. H2 includes any form of loss of control such that the aircraft becomes unstable. An example is if the aircraft exceeds the structural or functional limits of an aircraft. If a wing falls off or the fuselage experiences structural damage, then this would lead to an inability to control the aircraft.

H3 states that the aircraft does not execute the planned operations. This can include several different scenarios, but ultimately, the individual conducting the STPA analysis must decide what constitutes a loss of mission. The missions for this analysis will primarily be examples of how mission planners and crew chiefs in the US Air Force define their missions. H4 includes any point in time where a friendly aircraft enters unapproved airspace or unapproved ground space. Air space and ground space are typically defined by mission planners during pre-flight mission planning. H5 is if the UAV fires at friendly forces, and H6 is if the Team Lead fires at friendly forces. The reason H5 and H6 are separate is so the analysis can highlight differences between constraints for the Team Lead versus the Autonomous Controller for the UAVs.

The hazards are traceable back to the mishaps as shown in the brackets following the hazards with M1, M2, etc. A succinct diagram demonstrating the traceability is shown in Table 4. There are also mishaps that were not included but could be. For example, a hazard could be the Team Lead or UAV is fired at or targeted. This hazard was not included because it completely depends on an adversary and not just on this system. Each adversary has different capabilities, and an in-depth analysis is needed to determine the control actions and feedback necessary to avoid being targeted by an enemy.

Table 4. Mishaps and Hazards

X	X	X	X	H1	Aircraft violates minimum separation
X	X	X		H2	Aircraft Instability
		X		H3	Aircraft doesn't complete mission
	X	X		H4	Aircraft departs approved airspace
X	X		X	H5	UAV fires at friendly forces or civilians
X	X		X	H6	Team Lead fires at friendly forces or civilians
M1	M2	M3	M4		
Loss of life	Loss of aircraft	Loss of mission	Ground property damage		

To prevent the hazards from occurring, there must be safety constraints on the system. Therefore, the following list identifies examples of safety constraints that must be enforced. According to the STPA handbook, a system-level constraint specifies system conditions or behaviors that need to be satisfied to prevent hazards (and ultimate prevent losses). By inverting the condition for the hazards, a safety constraint is formed. Constraints can also define how the system must minimize losses in the event a hazard occurs. Finally, hazards can also be refined into sub-hazards. Sub-hazards assist with complex applications where modelling the control structure may be difficult. One method of deriving sub-hazards is to ask what controls are necessary to prevent the hazard. If the controls are not controlled adequately, then a hazard could occur.

- SC-1.1: Aircraft must satisfy minimum separation requirements from other aircraft and objects [H1]
- SC-1.2: If aircraft violates minimum separation, then the violation must be detected, and measures taken to prevent collision [H1]
- SC-2: Aircraft airframe integrity must be maintained under worst-case conditions [H2]
- SC-3: Aircraft must satisfy mission parameters [H5]
- SC-4.1: Aircraft should not depart approved airspace [H6]

- SC-4.2: If aircraft violates approved airspace constraint, then the violation must be detected and measures taken to prevent encounters with enemy or law enforcement [H6]
- SC-5.1: UAV must not fire at friendly assets or forces [H3]
- SC-5.2: If a UAV violates the friendly firing constraint, then the violation must be detected and measures taken to prevent impact [H3]
- SC-6: The Team Lead must not fire at friendly assets or forces [H4]

The next step is to develop a functional control structure. A control structure is a system model composed of feedback control loops. An effective control structure will enforce constraints on the behavior of the system. A controller will provide control actions to control a process and enforce constraints on the behavior of the process. A control algorithm represents how the controller makes decisions in determining what control actions to implement and when. Controllers have process models that represent the controller's instantiation of a process, or mental models in the case of humans. The process model identifies how a process should occur as opposed to the actual process which is what actually occurs. Process models or mental models are updated using feedback from the process. An example of generic control loop is in Figure 15.

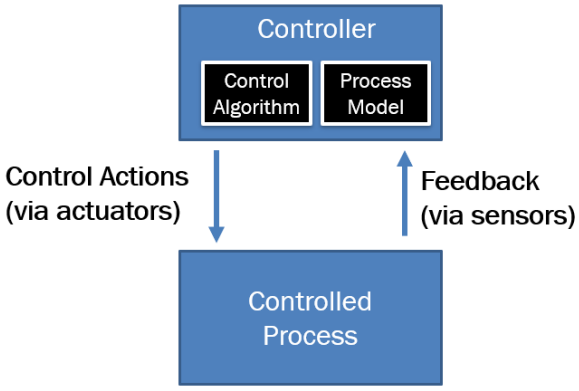

Figure 15. Feedback Control Loop

In general, a control structure will contain five elements: controllers, control actions, controlled processes, feedback, and inputs or outputs between components in the structure. Inputs and outputs are neither control actions nor feedback, but rather communication between controllers that is necessary to enforce safety constraints on the system. All downward arrows represent control actions or commands, and the upward arrows represent feedback to the controllers. The control structure is hierarchical so each controller that is higher in the diagram has authority over other controllers or processes below it.

One strategy for developing the control structure is to develop states and responsibilities for each controller in order to determine what control actions and feedback are required. A controller will need to have awareness of the states of specific controllers to function safely in addition to completing tasks while preventing hazards. Therefore, the controllers need to have cognizance of

the states of specific individuals within the system. Additionally, knowing the necessary states of a controller will highlight where inputs and outputs will come from within the system. Finally, the responsibilities of the controllers have to be understood in order to understand what the controller will do. If the controllers have vague functional responsibilities, then it is difficult to define what they need to control or get feedback from. Therefore, the following list highlights the states needed of each controller and the responsibilities for them.

ATC

States needed:

- UAV(s)
- Team Lead
- Air traffic

Responsibilities:

- R-1: Provide coordination to ensure minimum separation is maintained between all aircraft [SC-1.1]
- R-2: Issue landing and takeoff instructions for each aircraft [SC-1.1]

ATC will need to know the state of the UAVs, the Team Lead, and the traffic within its region to keep the aircraft from colliding with one another. ATC has two primary responsibilities. The first is monitoring and directing the movement of aircraft on the ground and in the air. The second is to issue takeoff and landing instructions. There are other responsibilities that air traffic control has like managing communications by transferring departures between different control centers as well as accepting control of arriving flights. They provide information to pilots including weather updates and runway closures. They also alert response staff in the case an emergency occurs. However, these may or may not apply for every mission set, so the two primary responsibilities have been listed since they apply for every mission.

Mission Planners

States needed:

- UAV(s)
- Team Lead
- Ground Station
- Model of mission

Responsibilities:

- R-3: Provide operational support during the mission [SC-5]
- R-4: Develop a mission plan [SC-3]
- R-5: Provide new missions coming from the Air Operations Center to the Team Lead and Ground Station [SC-3]
- R-6: Provide updates and changes to the mission [SC-3]

The mission planners need a state of the Team Lead, the Ground Station, and the UAVs. They need to know what the Team Lead is doing, if the Ground Station has control of the UAV, and what the UAV is doing to know if they need to change or update the mission in any way. They also will have a model of the mission that needs to be communicated to other groups. They have four main responsibilities: developing a mission plan, providing the plan to the team, providing changes/updates to the team, and providing support during the mission in case of emergency or other events.

Crew Chief

States needed:

- UAV(s)
- Team Lead aircraft
- Ground Station

Responsibilities:

- R-7: Ensure the base has the proper resources to complete the mission [SC-3]
- R-8: Inspect aircraft and ensure it is fit for flight [SC-2]

The Crew Chief needs a state of the Team Lead aircraft, the Ground Station (specifically the Ground Station resources), and the UAVs. The Crew Chief must be able to maintain the aircraft so they are mission ready and ensure the Ground Station has the capabilities to watch over the mission and take over when necessary.

Team Lead

States needed:

- Autonomous Controller
- UAV(s)
- Ground Station

- Air Traffic Control
- Mission Planners/Crew Chief
- Model of mission
- Environment

Responsibilities

- R-6: Decide when and what to fire at, and which aircraft will fire [SC-5.1, SC-5.2, SC-6]
- R-7: Provide formations and search targets for the UAV(s) [SC-3]
- R-8: Monitor location to ensure proper waypoint adherence [SC-4.1, SC-4.2]
- R-9: Perform preflight walkaround with Crew Chief [SC-2]
- R-10: Fly aircraft and avoid collisions [SC-1]

The Team Lead will need to know the states of many different controllers involved in the mission. The Team Lead must know if the Autonomous Controller is receiving commands, if the UAVs are performing the correct actions, if the Ground Station is ready to override the UAVs, if ATC is aiding in aircraft avoidance within the region, if the Mission Planners or Crew Chief are able to provide the mission updates, etc. The mission model involves how the Team Lead thinks the mission should proceed and what they expect or assume along the way in terms of support, communication, attitude, etc. Finally, the Team Lead has several responsibilities. The first is to decide when to fire at a specific target. This is ultimately up to the Team Lead to fire his/her weapon or to authorize the UAV to do so. The Team Lead must maneuver their aircraft and avoid collisions. They must monitor their location as well as the UAVs to ensure formations and positioning are correct. Finally, they must monitor the UAV activity and provide override in case of non-normative behavior.

Ground Station

States needed:

- Autonomous Controller
- UAV(s)
- Team Lead
- Mission planners/crew chief
- State of airport
- Environment

Responsibilities:

- R-10: Takeoff and land the UAVs (assuming modern day UAVs) [SC-3]
- R-11: Provide mission updates/changes to the TL and UAVs [SC-4]
- R-12: Monitor UAV activity and provide override commands in case of malfunction [SC-1.1, SC-3, SC-4.1, SC-4.2, SC-5.1, SC-5.2]
- R-13: Coordinate handoff to and from Team Lead [SC-1]

The Ground Station must also know the states of the several controllers in the system for similar reasons to the Team Lead. The Ground Station must have a state of the Autonomous Controller to know if intervention is necessary. They must have a state of the UAV and its position, velocity, etc. They need to know where the Team Lead is located and their progress. They must know if the Mission Planners or Crew Chief have given authority to take off or land. Additionally, they need a state of the airport for landing and takeoff of the UAVs as well as of the environment to know if abnormal behavior may occur due to extreme circumstances. The main responsibilities of the Ground Station are to take off and land the UAVs, provide mission updates and changes for the UAVs, to monitor UAV activity to provide override if necessary, and to coordinate the handoff to the Team Lead after takeoff and before landing.

Autonomous Controller

States needed:

- UAV(s)
- Environment

Responsibilities:

- R-13: Actuate aircraft hardware when the Ground Station or Team Lead provide a command (target, search area, fire, formation command, etc.) [SC-1, SC-2, SC-3, SC-4, SC-5, SC-6]
- R-13.1: Perform avoidance maneuver(s) when the condition for avoidance is present [SC-1]
- R-13.2: Perform correction maneuver(s) when the aircraft is about to enter restricted airspace or is approaching a condition for instability [SC-2, SC-6]
- R-13.3: Land the UAV when requested by the Ground Station

The Autonomous Controller needs to have a state of the UAV(s) and a state of the environment that the controller is operating in. In terms of responsibilities, the controller will actuate the hardware when a command is sent to it. This can include performing an avoidance maneuver, getting in position to image a target, avoiding restricted air space, etc. However, the specific actuation is the responsibility of the Autonomous Controller. The Autonomous Controller has a

process model and control algorithm as well that define how it will provide commands to the UAV hardware.

UAV

States need:

- Autonomous Controller
- Team Lead
- Ground Station
- Environment

Responsibilities:

- R-14: Perform actions provided by the Autonomous Controller, Team Lead, or Ground Station [SC-1, SC-2, SC-3, SC-4, SC-5, SC-6]

Finally, the UAV needs a state of the Autonomous Controller, the Team Lead, and the Ground Station. If the Ground Station cannot provide override commands, then the UAV hardware can adjust to use a stronger signal receiving from the Team Lead or vice versa. The UAV needs to know their states in order to adjust signals or other configurations based on who is available to send commands and who is not. It also needs a model of the environment to understand how the hardware may need to be modified for specific conditions. The responsibility of the UAV is to perform commands given to it by the Autonomous Controller, the Team Lead, or the Ground Station.

The next step in STPA is to identify a functional control structure that itemizes the control actions for each controller.

UxAS Functional Control Diagram

The next step is to develop a functional control diagram identifying the different controllers and processes necessary for a MUM-T mission. There is a significant difference between a functional structure and physical structure. Physical structures include direct linkages between components whether wired or wireless. This could be a GPS connection, radio frequency signal, Bluetooth, etc. These are not the connections of interest. Functional structures provide the specific tasks between various controllers and processes to describe how the system operates. For example, a keyboard might be physically connected to a central processing unit via USB. However, functionally, the keyboard provides character inputs to the unit. As a comparison, a functional control structure like those used in STPA is shown in Figure 16 [14].

The ground station provides functional inputs, or control actions, to the aircraft such as Start Engine, Stop Engine, Attitude commands, and more. The aircraft then also provides feedback to the ground station such as the GPS position of the aircraft, its fuel level, the engine status, etc. The cyclic nature of this structure where control actions are provided to the aircraft and feedback is sent back to the ground station is known as a control loop. This example did not include actuators or sensors, but typically, there will be actuators that perform the control actions and sensors will provide the feedback back to the controller. Since systems theory frames safety as a control problem where interactions between components are not adequately controller, this type of structure will provide insight into how those interactions might break down. Figure 17 shows a physical data link example in contrast.

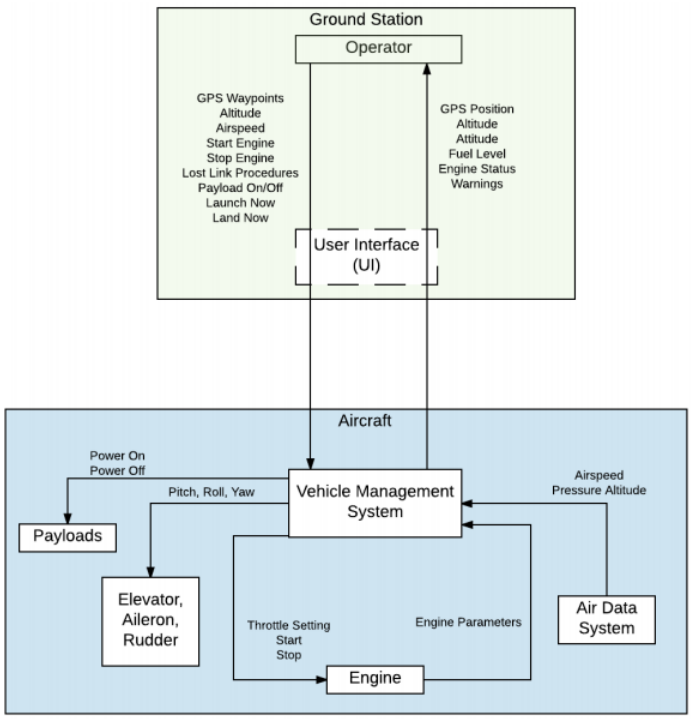

Figure 16. UAV Functional Control Structure [14]

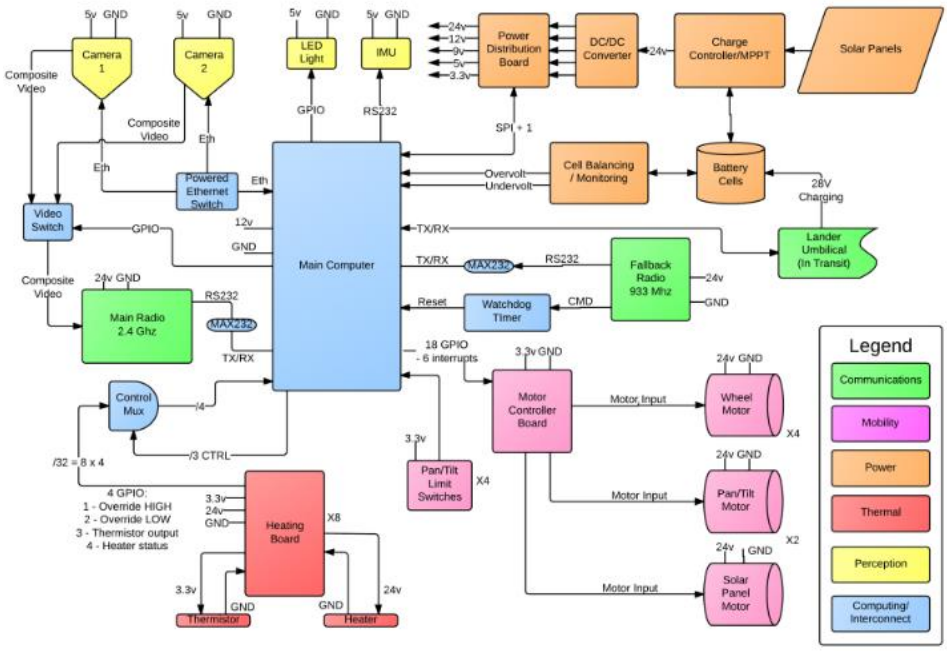

Figure 17. UAV Physical Data Structure [45]

The control actions are commands that the controllers must provide to processes. To identify what feedback must be provided to the controllers, the responsibilities provide information that will highlight the necessary feedback. An example of this process is in Table 5.

Table 5. Feedback and Responsibilities

Autonomous Controller Responsibility	Process Model	Feedback
Land the UAV(s) when the Ground Station requests it [SC-1]	Ground Station request to land	Current task execution
Lower the throttle and pitch down to descend when a landing command comes from the Ground Station [SC-2]	Ground Station request to land	Throttle setting Attitude

One of the responsibilities for the Autonomous Controller is to land the UAV(s) when the Ground Station requests it. Therefore, in order for the Autonomous Controller to land the UAV, its process model must know when the Ground Station requests a landing. For the Ground Station to know that the Autonomous Controller is executing the command, they must receive feedback specifying the UAV is landing. It is possible to be more specific in this case. Another responsibility for the Autonomous Controller could be to lower the throttle and pitch down to descend when a landing command comes from the Ground Station. The process model is still the same, but now the feedback includes throttle settings and aircraft attitude.

Air Traffic Control (ATC) is the ultimate authority during any mission. If ATC asks an aircraft to perform an action, then that request must be acknowledged and satisfied. One assumption for this analysis is that ATC has a radar providing positions for each aircraft within the region. Thus, they can grant clearance requests from the Team Lead and Ground Station as applicable. Subsequently, the Mission Planners and Crew Chief are in the next blocks below. Mission Planners provide updated mission plans and data to the Team Lead and Ground Station and receive current mission statuses from these groups in order to update the plans. The Crew Chief works with the Team Lead and Ground Station to provide maintenance to the aircraft and inspect them prior to takeoff. The Ground Station or Team Lead are in the next level as these teams provide control actions to the UAV(s) and provide coordination with other nearby traffic.

There are several assumptions for the Ground and Team Lead. The Ground Station can implement the same control actions as the Team Lead in case the Team Lead loses communication with the UAV(s). However, for this system definition, the Team Lead is limited in control actions to avoid high workload and situational awareness issues. It is possible to provide the Team Lead with more control actions, but studies would need to confirm how many control actions a pilot can handle without becoming oversaturated with tasks. In this case, the Ground Station primarily provides takeoff or landing capabilities. Additionally, if the Team Lead cannot focus on providing formations to the UAV(s) and the Autonomous Controller is acting inappropriately, then the Ground Station may use line of sight or beyond line of sight to alter the aircraft's physical parameters like altitude, position, etc. and take back control over the Autonomous Controller. The Team Lead provides functions that involve formation, targeting, and firing. For this architecture, the Team Lead will specify targets for the Autonomous Controller, provide authority to fire or engage targets, and provide a general formation command. Additionally, the Team Lead must control his own aircraft which involves setting attitude and airspeed, firing his weapons, setting waypoints, etc.

There are other controllers that could be considered within this diagram. For example, there could be maintenance which provides feedback to the Team Lead and Ground Station about aircraft performance, equipment degradation, etc. There could be a missile battery or ground troops that provide feedback to the Team Lead about the locations of targets. The diagram could become quite detailed. However, this simple setup allows for focus on the primary controllers tackling the mission. Other controllers mostly provide feedback without implementing or providing control actions for ISR missions or air to air combat within the Air Force.

One significant assumption involving the Team Lead is whether the Team Lead selects an aircraft to perform certain functions or if the Team Lead implements a function without specifying an aircraft. Each control action is different. Table 6 identifies which control actions require specification from the Team Lead and which ones do not (i.e., the Autonomous Controller will select the appropriate aircraft based on the Team Lead's control action). This impacts under what context a control action becomes unsafe for each controller, and what control actions they have.

Table 6. Aircraft Specification

Control Action	Team Lead Specification	Autonomous Controller specification
Fix on target	X	
Search for target/object		X
Identify target		X
Track target		X
Enter formation		X
Fire at target	X	

The Team Lead provides a specific aircraft for a fix on target to prevent multiple aircraft from fixing on the same target. The Autonomous Controller might have multiple aircraft fix on the same target which could result in more damage than is necessary. Additionally, the Team Lead provides an aircraft for firing at the target for the same reason. Only specific aircraft with certain weapons must be selected to fire, and the Team Lead must select those options without leaving room for the Autonomous Controller to decide. Actions like searching for a generic object can be specified by the Autonomous Controller because they do not involve specific weapons or targets. However, once a weapon is required, the Team Lead must provide the aircraft and weapon specification. The Autonomous Controller will still provide options to the Team Lead, but the TL must decide whether a specific aircraft will perform the command, with what weapons, and against which target.

For a "Search for target" command, the TL can provide a generic object to look for instead of specifying a UAV or provide an exact target, but he does not need to select which UAV(s) will perform the action. He can manually select a UAV to do so if he chooses, but the Autonomous Controller can distribute UAV(s) to search for something. There may be situations where the Autonomous Controller inappropriately provides too many or not enough aircraft on a "search for target" command. However, this risk is minimal compared to a friendly fire or excessive damage on a target. Other control structures can rearrange the control actions based on the specifications. However, Table 6 is what will be used for the rest of the analysis.

There is communication that occurs between the Team Lead and the Ground Station. However, this is assumed to be informational for coordination rather than a control action or feedback because neither controller is controlling the other by enforcing safety constraints on behavior. It is possible for the Team Lead and the Ground Station to not communicate about mission status and still have a safe system because they both receive feedback from the Autonomous Controller on current tasks performed by the UAV(s). Therefore, it is helpful for the team to communicate, but in terms of commanding a process, this communication is not required for controlling the system to enforce safe behavior.

Moreover, there is an Autonomous Controller (AuC) that interacts with the Team Lead, Ground Station, and UAVs. The AuC can take many forms in this architecture. There is no one specific type of computer that is involved, but rather a general machine that performs certain functions. The AuC has numerous control actions that involve the UAV physical components including attitude adjustment, throttle setting, weapons release, and more. It is possible to have an additional controller between the AuC and the TL/GS that actuates the hardware while another

provides planning, task allocation, way points, etc. but for this analysis, there are not two separate controllers. This is because most modern aircraft computers can perform the intended functions and using two separate controllers could cause delays in transmissions. Additionally, adding an additional controller creates more unsafe control actions because there can be more problems with implementing control actions or receiving feedback between the two autonomous controllers. There could also be an autonomous controller shown between the Team Lead and his aircraft, but this controller is not the highlight of the safety and security problems with MUM-T. This type of analysis had been completed by several STPA users already. [14]

The AuC also receives feedback from certain components on the aircraft like latitude and longitude from the GPS unit, airspeed from the pitot tube, altitude from the altimeter, etc. There is one significant assumption involved with the AuC to understand why the AuC has different control actions from the Team Lead and the Ground Station.

As noted in Sheridan's paper on different levels of automation, level two involves the computer computing a plethora of options available to the human as well as alternatives to ideal options [46]. This is the level of autonomy assumed for the interactions between the Ground Station/ Team Lead and the Autonomous Controller. The AuC will provide options to the Ground Station or Team Lead without narrowing them down, and the Ground Station/Team Lead can choose one of the options or provide their own parameters that are not any of the automated options. Therefore, the AuC cannot decide to fire at enemies, target an object, search a region, etc. Only a command from an outside operator would allow the AuC to take that action. There may be unauthorized users that provide the command, but a command is still required. Other architectures may include an AuC that is able to decide to take these actions at higher levels of automation, but this analysis will only examine level two because most modern MUM-T systems match this level of automation. This allows for easier comparisons with safety requirements of other systems already in existence.

Another important assumption involves the algorithms. The Autonomous Controller can utilize machine learning algorithms to identify targets, predict their future locations, select a weapon to fire at them, select a UAV to provide a target designation, etc. However, regardless of whether the algorithms are machine learning oriented or use rules engines, the algorithms can be examined within the context of STPA. For this analysis, machine learning is analyzed because ML allows a machine to adjust its output using data. ML is dynamic and does not require human input to make changes. Due to its popularity, it is important to understand how ML could assist or impair systems from completing a mission. However, ML will not change the results of the analysis. ML or non-ML software can result in the same hazards and UCAs. The only distinction is in the refined scenarios; however, both algorithms can have requirements flaws.

This analysis will assume the use of ML for specific control level functions. For example, ML can be used to determine which set of UAVs should search for a target and which set should track another target. ML might determine which UAV has a weapon quality track and a kinematic advantage in order to fire at a hostile. As noted above, ML will not be involved in mission-level decisions such as prioritizing weapon tracks, deciding when to fire, etc. ML will be used to execute the control-level tasks from the Autonomous Controller to the UAV. The decision to use ML for only control level tasks is due to feedback from various Air Force pilots

on what the use of autonomy should be. As ML is trusted for performing the control-level tasks, then moving to mission-level tasks can be implemented later as trust is established.

There are other assumptions that can be made about the Autonomous Controller and its relationship with the other controllers; however, these assumptions do not affect the resulting UCAs and scenarios. This could include whether the Autonomous Controller is centralized or decentralized. As discussed in the UAV architectures in Chapter 2, there are advantages and disadvantages to both. However, this does not affect the scenarios or requirements. It is simply an architectural design preference that does not change the functionality of the Autonomous Controller.

For the final part of the architecture, the AuC receives feedback from the UAV(s) hardware including positions data, engine data, payload data, etc. The AuC provides position, orientation, and formation commands to the UAV(s) so the aircraft are aligned appropriately to complete the mission. The feedback mechanisms in the control structure are subject to change depending on the results from performing the STPA analysis. Currently, they are based on the responsibility analysis for each controller.

There is also more general communication that takes place in this architecture that is neither a control action nor a feedback mechanism. The UAVs will provide data to each other to assist in functions, but this is not considered a control action in this architecture because the UAVs are not commanding another UAV. In a sense, it seems that a UAV is indirectly controlling another UAV because it can be determined what actions a UAV would perform depending on whether another UAV sends the respective data or not. However, this is not the case. Autonomy may not perform the exact same operation given the same set of UAVs, intelligence, radar information, etc. The probability may be high that the UAV would execute an action given a set of circumstances, but the AuC commands the UAV to perform the action, not another UAV.

Moreover, assuming this sense of intelligence with the autonomy, then the communication that occurs is not a command and therefore, not a control action. This communication could involve ISR data, enemy targets, other UAV positions, and more. The UAVs will transfer data, but this communication is not controlling another part of the system. It is simply providing additional information without providing a direct command. However, the communication or lack thereof can cause scenarios which will be seen in Step 2 of the STPA. The pink boxes in Figure 18 represent the controllers and controlled processes. The blue boxes are control actions, and the green boxes are feedback to the controllers. There is general communication between UAVs, and between the Ground Station and the Team Lead shown as smaller text near those controllers.

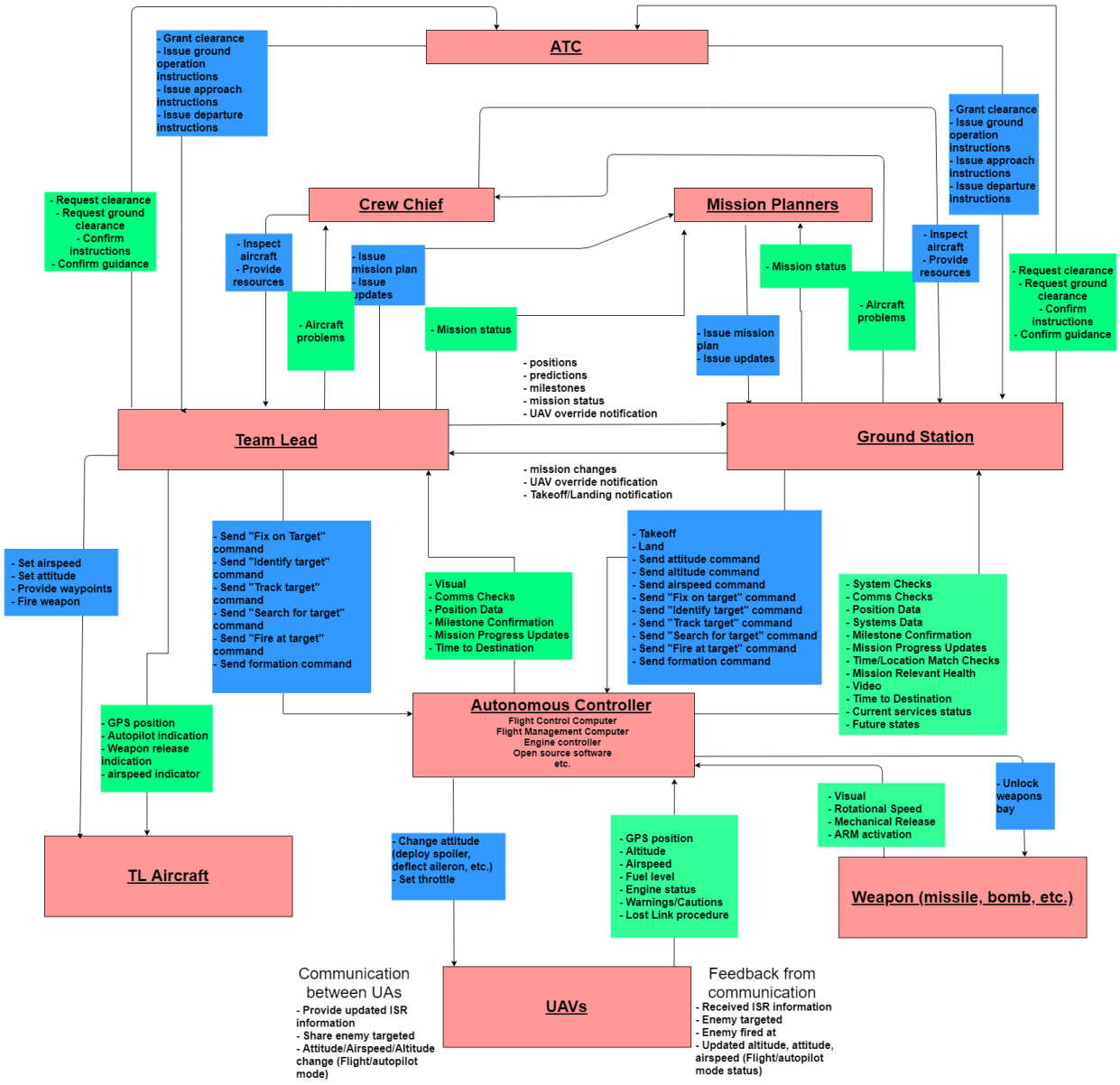

Figure 18. MUM-T Functional Control Diagram

Step 1: Unsafe Control Actions

The next step is to identify unsafe control actions (UCAs) by analyzing the functional control structure. A control action is any functional command that a controller must provide to a process for the system to operate successfully as defined by its mission goals or objectives. Control actions are hazardous if:

1. A control action required for safety is not provided or not followed.

2. An unsafe control action is provided.
3. A potentially safe control action is provided too early or too late, at the wrong time or in the wrong sequence.
4. A control action required for safety is stopped too soon or for too long.

Control actions become unsafe specifically because of the context in which they are executed. The Ground Station not providing a land command to the Autonomous Controller is not unsafe by itself; it becomes unsafe if the UAV needs to land because it is out of fuel and the Ground Station does not provide the command. Therefore, each UCA will specify a context under which it is unsafe. Additionally, the UCAs are traceable to the hazards to keep track of which hazards result from a UCA. By implementing constraints and eliminating UCAs, hazards can also be eliminated.

A full list of the UCAs are listed in Appendix A. For brevity, this section will cover a specific set of UCAs that involve the Autonomous Controller, Team Lead, and Ground Station. These controllers are highlighted because there are interesting scenarios in the next step that are derived from these UCAs. The four types of unsafe control actions are represented in each column and are in no particular order. For the Autonomous Controller, the control actions are focused on providing attitude commands to the UAV. For the Team Lead, the control actions are focused on providing "fire at target" commands to the Autonomous Controller. Finally, for the Ground Station, the control actions are focused on providing a land command to the Autonomous Controller.

Table 7. Autonomous Controller Unsafe Control Actions

Control Action	**Not Providing**	**Providing**	**Too Early/Late**	**Stopped Too Soon/ Applied Too Long**
Release weapon (bomb, missile, etc.)	Autonomous Controller does not release a weapon when the Team Lead commands it (H3)	Autonomous Controller releases a weapon for the wrong target (friendly, civilian, wrong hostile, etc.) (H2)	Autonomous Controller releases weapon before receiving a command from the Team Lead (H3)	N/A
			Autonomous Controller releases a weapon when the target is out of range or can no longer be hit with TBD certainty (H5)	

Table 8. Team Lead Unsafe Control Actions

Control Action	Not Providing	Providing	Too Early/Late	Stopped Too Soon / Applied Too Long
Send "Fire at target" command	Team Lead does not provide "fire at target" command to UAV(s) when the mission requires it (H3)	Team Lead provides a "fire at target" command for the wrong target (H3)	Team Lead provides a "fire at target" command after the target is out of range of fire (H5)	N/A
		Team Lead provides a "fire at target" command to the wrong UAV (H3, H5)	Team Lead provides "fire at target" command before UAV(s) have fixed on the appropriate object (H5, H7)	
		Team Lead provides a "fire at target" command when friendlies or civilians are near the target (H5)	Team Lead provides "fire at target" command before receiving authorization from the general (H5)	

Table 9. Ground Station Unsafe Control Actions

Control Action	Not providing	Providing	Too early/Late	Applied too long/stopped too short
Land	Ground Station does not provide land command when the UAV is in a pattern and at minimum fuel (H1, H2)	Ground Station provides land command when the runway is not clear (H1)	Ground Station provides land command before the UAV completes the airfield arrival procedure (H1)	N/A
	Ground Station does not provide land command when the UAV is at the airfield and other aircraft are trying to enter a pattern (H1)	Ground Station provides land command when the UAV is above unstable terrain (H1)	Ground Station provides land command after the UAV has already entered a restricted air space (H6)	

	Ground Station does not provide land command when the UAV is ready to land (H1, H2)	Ground Station provides land command when the UAV is in a restricted landing area (H6)	Ground Station provides land command after the UAV is already out of fuel or is no longer over a safe landing area (H1, H2, H6)	
			Ground Station provides land command before the UAV(s) complete the mission (H5)	

Examples of safety constraints from some of these UCAs would be:

- The Ground Station must provide a land command when the UAV is in a pattern at minimum fuel
- The Ground Station must provide a land command before a UAV enters restricted airspace
- The Team Lead must provide "fire at target" command for the correct target
- The Autonomous Controller must provide attitude correction if the UAV is off course

Step 2: Causal Scenario Generation

With the unsafe control actions identified, the next step is to analyze each individually and create causal scenarios to determine how each unsafe control action can occur. These causal scenarios can be used to create more specific safety requirements that can be imposed on the system to promote safe operation. One way to identify scenarios is to analyze four different parts of the control structure as outlined by John Thomas in a presentation at the 2017 STAMP conference in Cambridge, MA [47]. The four areas include:

1. Command not followed or followed inadequately
2. Inappropriate decision
3. Inadequate feedback or other inputs
4. Inadequate process behavior

Examples of how the four areas affect the control loop are shown in Figure 19. This example involves an operator, autonomous controller (also known as the Vehicle Management System [VMS]), and a UAV as developed in Sarah Summers' thesis. The VMS is providing attitude and throttle commands to the UAV while receiving feedback. The control action for this specific case is an attitude command. The actuators include different components such as the hydraulics

system and the Main Flight Computer (MFC). The feedback comes through the MFC. These four areas represent basic types of scenario, and refinement can be done to explore how the basic scenarios occur. The advantage of using the basic scenarios is that the full control loop is examined so there are no missing scenarios. The basic scenarios provide ideas for how a scenario could occur whereas a checklist makes it difficult to creatively come up with a scenario that one is not familiar with.

These 4 basic scenarios are also described in the STPA handbook from pages 43-51. John Thomas describes the high-level scenarios for each type.

Type 1:

- Control action not executed
    - Control action is sent by controller but not received by actuators
    - Control action is received by actuators, but actuators do not respond
    - Actuators respond but the control action is not applied to the controller process
- Control action improperly executed
    - Control action is sent by controller but received improperly by actuators
    - Control action is received correctly by actuators, but actuators respond inadequately
    - Actuators respond adequately, but the control action is applied improperly at the controller process
    - Control action is not sent by controller, but actuators respond as if it had been sent

Type 2:

- Failures involving the controller (for physical controllers)
- Inadequate control algorithm
- Unsafe control input (from another controller)
- Inadequate process model

Type 3:

- Feedback not received
    - Feedback is sent by sensors but not received by controller
    - Feedback is not sent by sensors but is received to sensors
    - Feedback is not received or applied to sensors
    - Feedback does not exist or sensors do not exist
- Inadequate feedback is received
    - Sensors respond adequately but controller received inaccurate feedback
    - Sensors respond inadequately to feedback that is received
    - Sensors are not capable to provide necessary feedback

Type 4:

- Control action not executed
    - Control action is applied by the controlled process, but the process does not respond
- Control action improperly executed
    - Control action is applied by the controlled process, but the process responds improperly

- Control action is not applied by the controlled process but the process responds as if the control action had been applied

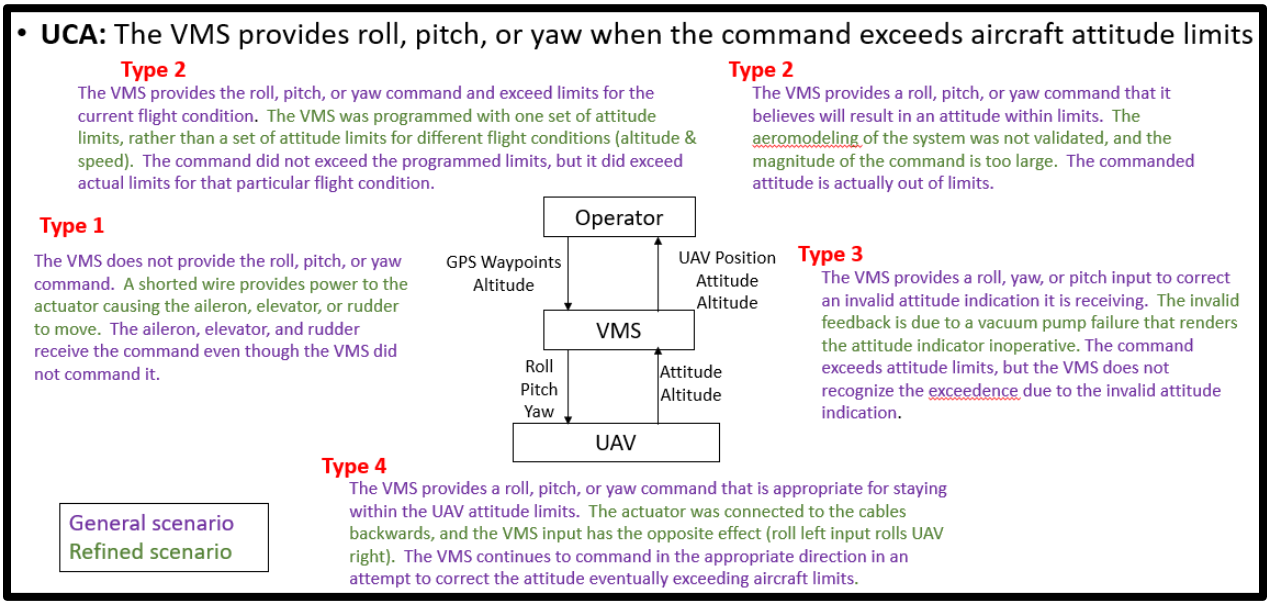

Figure 19. Causal Factors for Scenarios [47]

These scenario types can be analyzed within the context of the new general causal control model displayed in Figure 20. This is an excellent representation of a potential UAV with an Autonomous Controller and a human controller like the Ground Station or Team Lead. The human has a process model and assumptions about the autonomy, and there is a built-in process model for the autonomy, along with modes of what other controllers are in. Other controllers and the environment interact with the AuC, GS, and TL, any of which could result in a scenario.

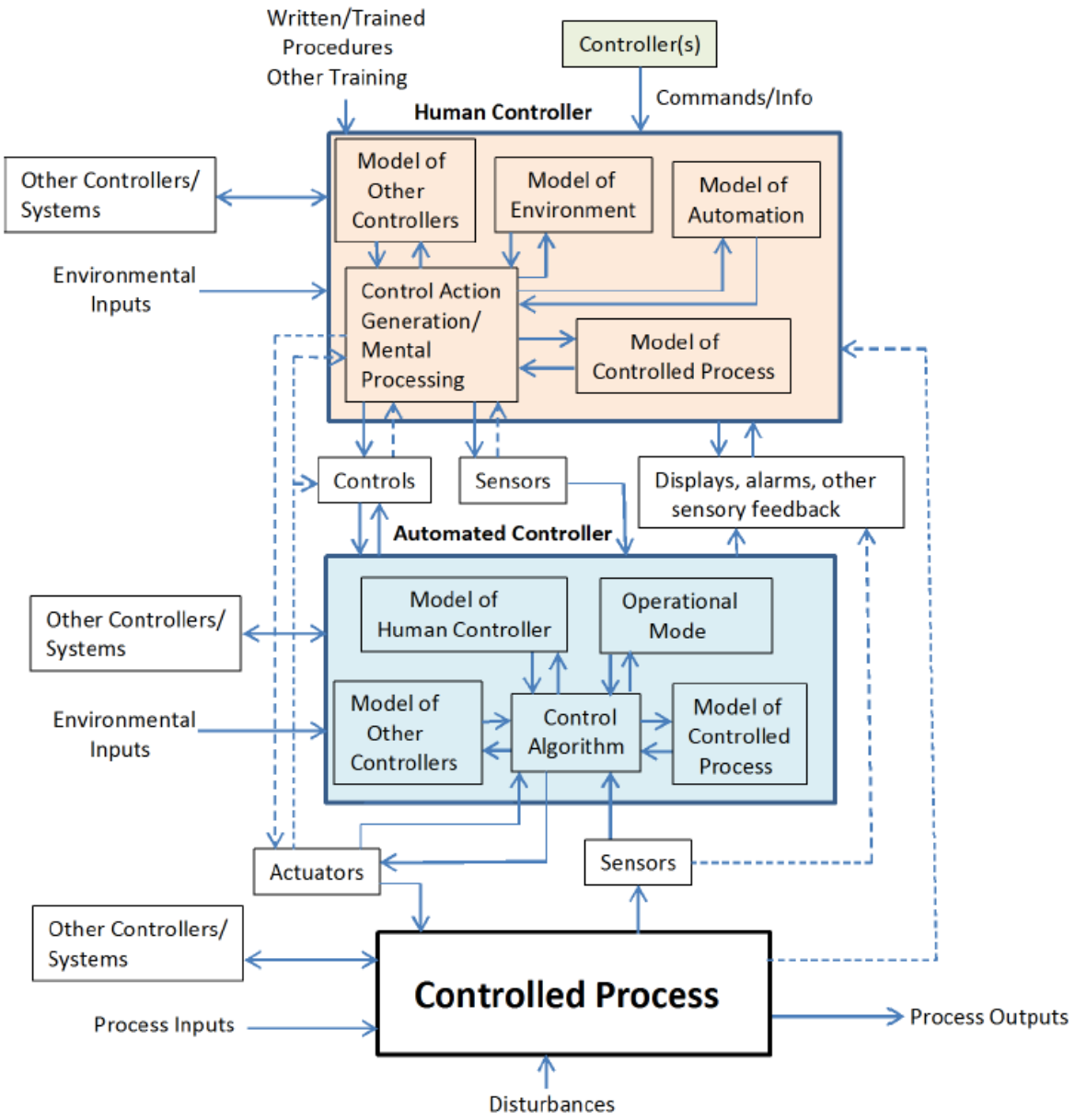

Figure 20. Causal STPA Scenario Generation Model [48]

For example, consider the previously identified UCA between the Team Lead and Autonomous Controller:

UCA: Team Lead provides "fire at target" command to the wrong UAV(s).

A diagram showing how this UCA fits into the control structure developed from Step 2 is shown in Figure 21. The Team Lead is providing a fire at target command to the Autonomous Controller, and in this particular UCA, it is unsafe because the Team Lead is providing the command to the incorrect UAV(s).

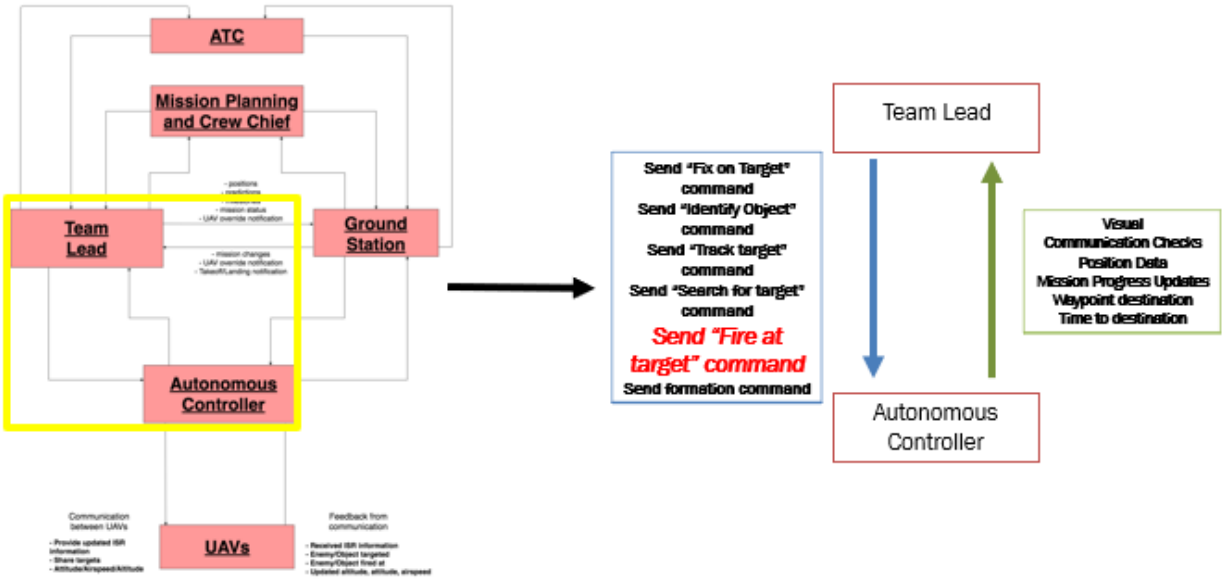

Figure 21. Team Lead Control Structure Focus

There are several scenarios that result from this UCA. An example of a type 2 scenario that involves inappropriate decisions by the Team Lead are shown below.

1. Scenario 1

    a. The Team Lead provides the fire at target command to a UAV that is not part of the mission. The Team Lead may incorrectly believe the UAV is part of the current mission because the Flight Management System interprets the signal as a valid friendly. This could occur if:

        i. A friendly UAV for another mission is within range of the Team Lead and broadcasts a valid friendly transponder signal to the Team Lead and to its originally assigned mission commander.

        ii. An adversarial UAV comes within range of the Team Lead and intercepts the transponder signals from the friendly UAV. As a result, the Team Lead sends the fire command to the adversarial UAV rather than the originally assigned UAV.

        iii. An adversarial UAV provides Link 16 response data to the Team Lead including imagery or communication information from nearby ground stations. If the Flight Management System previously characterized the UAV as neutral during the IFF test, the system will re-classify the UAV as friendly if this communication occurs in conjunction with the friendly transponder signal.

iv. The Team Lead's transponder malfunctions. Because of the protocol used in the Team Lead's FMS, in the event of a malfunction, the FMS will display all transponder signals as neutral. The Team Lead may then incorrectly select the wrong UAV to send the fire command to.

Note that security is covered in the scenarios, for example, number iii above. Next, requirements are necessary to enforce constraints on the system so the scenario does not occur. Examples of potential requirements are:

v. Friendly UAV(s) must broadcast different IFF transponder signals than mission UAV(s)

vi. The Team Lead and mission UAV(s) IFF transponders must differentiate between copied and authentic IFF responses

vii. Friendly aircraft must differentiate between Link 16 data from friendlies and neutral or enemy aircraft

viii. A procedure must be developed in case the Team Lead loses IFF transponder capabilities

This scenario highlights some interesting requirements for the Team Lead and the interface for providing a firing command. For example, the Team Lead IFF transponder must be able to distinguish between friendly responses and copied friendly responses. How should this differentiation occur? Does an IFF transponder software reboot reset the differentiated values? Or a software crash? How does the Team Lead receive notification that the IFF transponder reset its response parameters? Should the Team Lead be notified of copied friendly responses? There are several design implications to consider for just this one requirement, but it is necessary to ensure safe system operation.

There is no record of IFF transponder failures and their impact on aircraft because this information is classified. This makes it difficult to compare these scenarios with real life situations that might have already happened. However, the scenario can be understood in the context of how an IFF transponder works. An IFF transponder receives interrogation signals at one frequency and provides frequencies at a separate frequency with specific signal spacing. There are five modes of operation, two of which are specifically for military use. A transponder can receive signals of a certain frequency and will classify them as friendly as long as the signal uses the appropriate encryption and friendly signifiers in the signal (phase, amplitude, etc.). Therefore, an enemy could receive a signal by being on the appropriate frequency, decrypt the signal, and send a similar signal to the Team Lead. This process would only take a matter of minutes, especially if the signal is sent to an enemy Ground Station for processing and supplied back to the aircraft upon successful decryption.

Another scenario involving a similar UCA as the previous example highlights scenarios that occur regularly and must be addressed to successfully include unmanned aircraft in ISR or air to air missions.

UCA: The Team Lead provides a “fire at target” command for the wrong target

2. Scenario 2: The Team Lead provides a fire command for a friendly or civilian target. The Team Lead may incorrectly believe the target he selected is an enemy target. This could occur if:

    a. The Team Lead is notified by the battery that the target has a missile track and there is no indication that a friendly fired the missile. The supposed missile is actually a friendly aircraft.

    b. The Team Lead misidentifies the target because there are heat signatures coming from troops that were not planned to be in a certain area in unclear conditions (dark, cloud cover, etc.)

    c. The Team Lead misidentifies the target because the target does not have a signifier (specific color roof, flag, etc.) to signal it is an ally.

    d. The Team Lead misidentified the target because the target looks similar to an enemy stronghold, base, equipment, etc.

    e. The Team Lead misidentified the target because the video quality of the target from the UAV lacks detail to differentiate the target from an enemy.

    f. The aircraft does not notify the Team Lead that they are friendly. Additionally, the aircraft does not respond to an IFF or auditory transmission.

These scenarios have all occurred and requirements must account for them to avoid greater damages using unmanned aircraft to fire at targets. On April 2, 2003, two Navy F/A-18s were near Karbala in central Iraq and heading back to their ship, the USS Kitty Hawk. A Patriot missile battery mistakenly identified one of the planes as an Iraqi missile and notified the headquarters for air defense known as the Information Coordination Center. The Center mistakenly designated the flight path of the Navy jet as a missile track. Seconds later, a second battery located closer to the front line of fighting also detected the plane and also mistakenly determined it was a missile. The second battery decided that it and the military unit it was defending were the missile's target.

The effect of the corroborating reports highlighted that the operators at the two batteries and at the command center were "increasingly confident that they were all detecting the same hostile missile, that their detection was accurate, and that this missile was a direct threat to U.S. forces," said the summary of the report. The command center then ordered that two missiles be launched to eliminate the threat. The operators were fooled because the aircraft path looked like a missile path heading towards a friendly location [49]. It is clear that requirements are necessary to assist ground units and aircraft operators in identifying friendlies versus enemies.

A similar incident in Afghanistan involved an American helicopter gunship and a mud-walled compound built by British soldiers. Troops from the 3rd Battalion "The Rifles" were in a firefight with insurgents in December 2009. Unknown to them, senior British officers were watching a drone's grainy images from the safety of Camp Bastion, about 30 miles from the battle at Patrol Base Almas. The officers, however, had mistaken the soldiers' mud-walled compound for the enemy's lair. The US helicopter gunship killed a British soldier after being ordered to attack by officers who had misinterpreted the grainy images from the drone.

Lance Corporal Christopher Roney, 23, a married father of one, suffered fatal head injuries when the US Apache gunship swooped in and opened fire with its 30-millimeter chain guns, firing about 200 rounds of ammunition. The soldier was flown to Camp Bastion but died the next day. Eleven of his comrades were also wounded. The coroner reported that: "The deployment and use by friendly forces of attack helicopters was done in circumstances that ought to have been assessed by them to conclude sooner than they did that their target was not an enemy force and that the attack should be aborted." He also blamed the fact that Patrol Base Almas was not marked on military maps and that British commanders insisted there were no friendly forces in the vicinity [50].

If there are issues with grainy drone video footage now, then implementing a sensor truck where operators are expected to fire weapons based solely on flawed telemetry data is egregious. There are numerous scenarios where aircraft fired at friendlies or civilians due to the inability to properly identify other objects. This small set of requirements highlights a significant need for MUM-T systems.

Another causal scenario highlighting interaction between the Ground Station and the Team Lead is located below. The area of focus for the control structure is highlighted in Figure 22 to provide context.

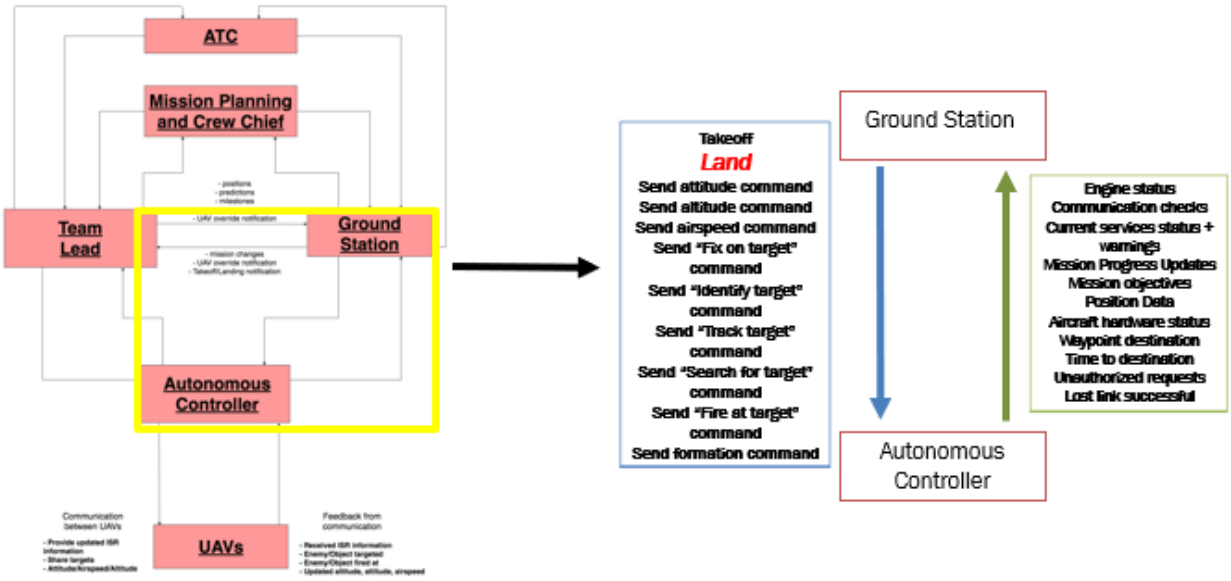

Figure 22. Ground Station Control Structure Focus

UCA: The Ground Station provides a land command before the UAV(s) complete the mission

3. Scenario 3:

    a. The Ground Station provides a land command while the UAV(s) are performing the mission because they believe the mission is already complete and the UAV(s) require immediate landing to prevent damage. This could occur if:

        i. An attacker provides false communication to the Ground Station notifying them that the mission is complete

        ii. The Autonomous Controller states that all tasks within its queue have been completed, but there are still tasks to complete

        iii. The UAV(s) begin implementing a post-mission procedure including downlinking all available footage, implementing "Return to Base" formation, etc. which confuses the Ground Station into thinking the mission is completed

        iv. The Ground Station receives false feedback from the Autonomous Controller that it completed all mission tasks.

        v. The Ground Station misinterprets the Team Lead's request for a mission extension as a call to end the mission early

Possible requirements for this scenario include:

vi. The Ground Station must be able to verify the origin of a communication

vii. The Ground Station must verify the end of mission with the Team Lead

viii. The UAV(s) must not implement a post-mission procedure until provided authorization from the Team Lead or Ground Station

ix. The Team Lead must provide verification to the Ground Station that the UAV(s) completed the mission

x. The Team Lead must provide redundant feedback to the Ground Station to verify mission completion

Requirement vi, vii, and viii were identified by team members at the Air Force Lifecycle Management Center as a critical requirement after receiving feedback from pilots. This shows the analysis provides requirements that are useful for system developers when thinking about how they will design their system. Although a miscommunication of mission completion sounds farfetched, similar incidents have occurred.

For example, in 2010, a team of soldiers infiltrating an enemy base in Mosul radioed back to the Ground Station that the mission was "successful". The Ground Station assumed that the troops had captured an insurgent that was providing orders for Iraqi citizens on where to place car bombs. However, the troops only entered the building and found no one inside [51]. This led to the distinction between "completed" missions and "accomplished" missions. Completed means that the team was able to safely finish the mission without being damaged or harmed. Accomplished means that all or some mission objectives were achieved. A mission can be incomplete but accomplished, complete but unaccomplished, or neither. However, the terminology is standard throughout services or even with squadrons. This type of scenario must be addressed to prevent conflicting commands along with the other requirements.

Finally, the Autonomous Controller is analyzed. Figure 23 highlights the control action focus and the focus from a control structure standpoint. This scenario will follow a format that differs from the previous three examples. This scenario was developed using John Thomas' method as described in his 2017 STAMP presentation: "A Process for STPA: STAMP Accident Model of HITOMI and Expansion to Future Safety Culture". In other words, it follow the 4 basic types described in Figure 19. The scenario is formatted as follows:

Basic scenario: Identifies what type of basic scenario is being evaluated for the UCA

Refined scenario: Indicates how the basic scenario occurred

The result: Identifies the outcome from the scenario.

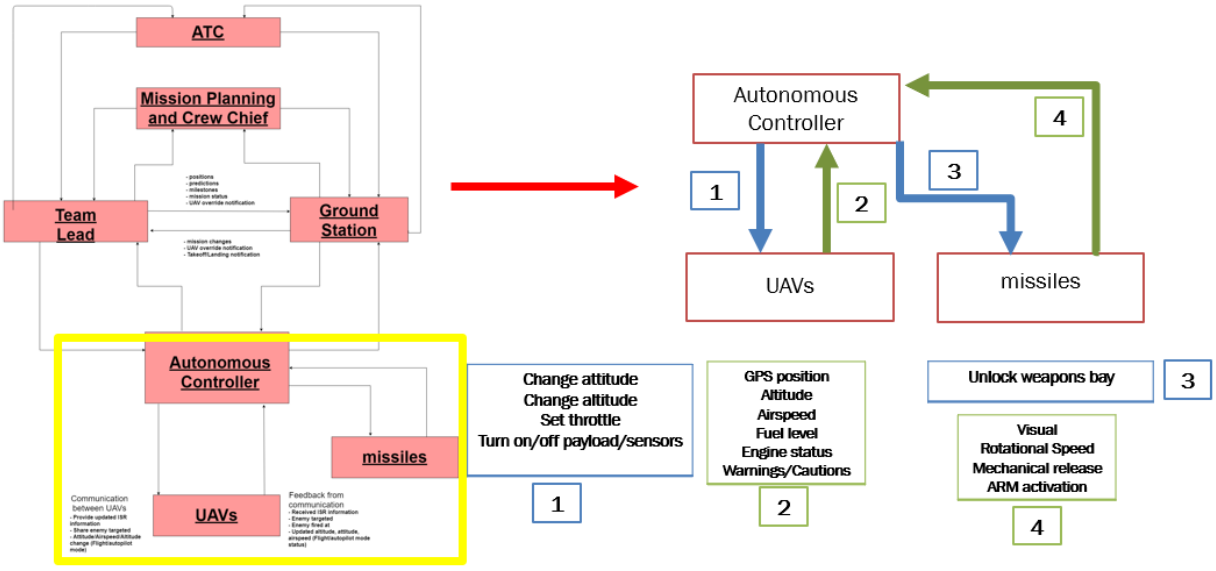

Figure 23. Autonomous Controller Focus

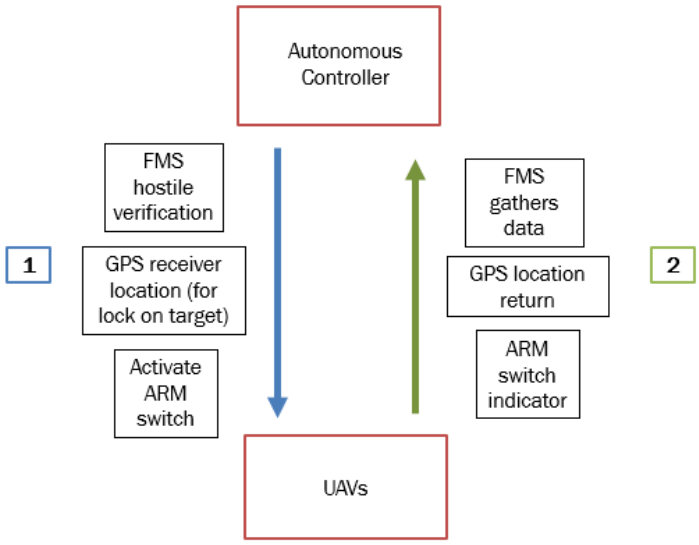

Figure 24. Autonomous Controller Control Structure Specifics

UCA: The Autonomous Controller releases a missile intended for a civilian or friendly target

- Scenario 4:
- Basic scenario: The mechanical release unlocks the missile from the weapons bay with the appropriate target.
- Refined scenarios:

- The missile receives a conflicting command from another controller while heading for the appropriate target. The command causes the missile to head towards an object using the missile's infrared homing system instead of the guidance system.

- The missile is unlocked from the weapons bay before confirming a target or its location. The missile immediately heads for the first target it can locate using the last known data from the FMS.

- The missile is launched in boresight mode, but the pilot is immediately bombarded by a missile heading towards him. The priority is to safely avoid the missile. This causes the pilot to forget to select the target for the missile, and the missile heads for the last ground target available from the FMS.

- The machine learning algorithm within the missile guidance system is unable to classify an aircraft and waits for Team Lead or Ground Station input to classify it while in flight. The GS/TL are occupied and do not provide an input within TBD seconds, resulting in the missile executing a bogey trajectory. The missile cannot complete the maneuver and crashes near a civilian site.

- The AuC is unable to use visual recognition, radar, and/or IFF responses to classify a friendly aircraft. The friendly aircraft fires at an enemy, but another friendly aircraft is near the enemy. The ML algorithm classifies the firing friendly aircraft as an enemy because its ammo hits the friendly aircraft near the enemy.

- An adversary applies physical changes to a friendly object to make it look like a significant enemy target. The machine learning algorithm classifies the target as a high priority enemy.

- The aircraft goes beyond visual range for the missile to properly locate the target (using BVR missiles) as it is released from the weapons bay. The missile can no longer properly locate the target. This results in the missile trying to adjust attitude towards the target.

- The pilot loses his lock on target before the missile reaches the target. This causes the missile to attempt using visual guidance. The missile misses the target and impacts the ground near civilians.

- The result: The missile(s) heads toward a friendly or civilian, or a target that is not supposed to be fired at.

Possible requirements include:

- The missile's guidance and infrared homing systems shall be protected from receiving unauthorized commands during its travel

- A missile with an unconfirmed location or target shall have a series of locations that it can detonate safely while avoiding friendly or civilian objects

- The Ground Station must be notified and be able to select targets for missiles launched in boresight mode

- A friendly and enemy machine learning training set must include all possible friendly, enemy, and civilian aircraft

- A pilot must provide target designation within TBD seconds of launching a missile in boresight mode

- The control algorithm must not fire at targets without receiving full EII and FOL for targets from the FMS

- A human shall provide additional guidance to confirm non-lethal targets, time-permitting

- The missile visual guidance system must notify the FMS if the missile is no longer within visual range immediately (TBD milliseconds) before ejection from the weapons bay

- Missiles requiring lock until impact must ask for further instruction from the Autonomous Controller or the pilot if the lock is lost

Any requirement with "TBD" means that experimentation, models, and testing must be done to specify exactly what the number should be. Thus, "TBD" is a variable placeholder until the system designers decide what the number should be for their system.

Some of these scenarios have already occurred for autonomous controllers within aircraft. For example, in July 2017, an SF260 aircraft dropped bombs on friendly locations in the Philippines. The aircraft successfully hit its first three assigned targets but missed its fourth, hitting the ground troops instead. Military reports from Marawi indicate that the missile was 250 meters off target, causing the collapse of nearby structures. "Large debris from heavily reinforced buildings accidentally hit two of our personnel who succumbed to death in the process" while 11 others sustained minor shrapnel wounds [52].

The reason for the miss can be found in the aircraft's origins for design and production. The SF260 derives from the FA-50 and the T-50 Golden Eagle which were designed to serve as advanced jet training aircraft. The T-50 was intended to serve as a two-seat Lead-In Flight Trainer to prepare pilots for flying actual combat aircraft. The T-50 trainers were deemed so successful and easy-handling that Korean Aerospace Industries (KAI) decided to produce an upgraded version, the TA-50, that could also operate as a light attack plane using precision-guided weapons and a more powerful radar. KAI then pushed the design one step further with

the FA-50 which served as a supersonic fighter with fourth-generation avionics and stealth-like technologies.

The FA-50, which made its first flight in 2011, adds greater fuel capacity and key avionics upgrades, including a radar-warning receiver to alert the pilot if he or she is being targeted by hostile radar. There is also a night-vision system to identify aircraft and ground objects in the dark, and a specific data link to integrate the airplane with friendly sensor and weapon platforms that also carry the data link. Most importantly, the FA-50 carries a pulse Doppler radar that can detect fighter aircraft within 100 kilometers of its location. The radar can be used to lock on to air, ground and sea targets. The radar has a shorter range and fewer capabilities than the US radars, but Samsung is looking to develop a similar radar for use in the Golden Eagle.

The FA-50, though highly maneuverable, is not in the same league as fourth-generation fighters. However, a factory fresh FA-50 costs around 30 to 35 million dollars, whereas the F-35 and its counterparts presently cost anywhere from 94 to 120 million [53]. Thus, the FA-50 is at an acceptable price point for less wealthy countries. However, in air-to-air combat, the FA-50 still lacks the critical ability to fire beyond-visual range (BVR) missiles. It currently relies on short-range AIM-9M Sidewinders. KAI is reportedly working on integrating use of extra long-range AIM-120 Scorpions as well as superior short-range AIM-9X missiles.

In 2015 the Philippine Air Force announced it would upgrade three or four FA-50s to carry radar-guided AIM-7 Sparrow missiles with ranges of 70 kilometers. The Sparrows would be a major improvement—but the upgrades would average 17 to 22 million dollars per plane, more than half the cost of the FA-50 itself. Given that for the current short-range missiles the pilot must keep his radar locked on the enemy fighter until it impacts, the price seems exorbitant. Possible explanations may include steep fixed costs, and/or the provision of Sparrow missile to go with the upgrade. In any case, until the FA-50 is certified for carrying medium or long-range missiles, it cannot successfully serve as an air-superiority fighter [54].

There are several important notes in the history of the SF260. First, the aircraft derives from a trainer aircraft. This aircraft was not designed to carry weapons or track targets, so trying to retrofit the aircraft to do so is costlier than designing a proper aircraft. Secondly, the costs of the upgrades were too large to ensure the aircraft could lock on targets with medium range missiles or even carry medium range missiles. Due to the cost constraints, the Philippines settled for short range missiles. The missile for the recent 2017 mission required at least a medium range Joint Direct Attack Munition (JDAM) anti-tank missile. Additionally, although the avionics and radars were upgraded, they were not tested against the targets like the ones for this mission.

Finally, although machine learning algorithms have not been operationally used in UAV(s) to fire weapons, there is a potential for this to occur. As a result, this analysis highlights a subset of the requirements necessary to ensure safe operation of machine learning within this context. The algorithm must use a training set with all potential friendly and enemy aircraft. The process model must account for changes or updates in aircraft over time (fuselage modifications, new markings, etc.). The algorithm must receive feedback from certain aircraft sensors and components before initiating certain targeting or firing commands. This is not to say that using machine learning algorithms is unsafe in the context of UAV(s), but rather, for UAVs to be

successful in performing air to air combat or ISR missions, these requirements are critical to enduring safe operations.

The Army Science Board is currently performing a study to identify capability gaps for achieving MUM-T capability needs. One of the primary gaps to be filled is developing algorithms and artificial intelligence that can handle obstacles and countermeasures that are experienced while driving in extreme environments and terrain. Army vehicles do much more than just driving down a road, and if targeting is also involved, the vehicles must recognize and identify objects. Automated cars work well in identifying garbage cans in the road while it is sunny, but Army ground vehicles might experience weird pieces of metal, dangerous terrain, and severe temperatures which requires further algorithm development to be able to handle these various scenarios.

The board also identified communication and sensors as critical capability gaps. Ground vehicles operate in dense electromagnetic environments where waveforms require protection from power overloads and signal losses. Sensors on vehicles not only provide extended situational awareness for the human and the vehicle, but the sensors feed into the algorithms to decide which way to turn, where to lock on a target, etc. The previous requirements touch on all of these major capability gaps. Safety penetrates every component and subsystem. The algorithms, sensors, communication, human-machine interface, modelling and simulation, testing, and more must all have safety requirements defined so that the system operates safely and securely when needed.

In summary, the requirements are essential to prevent losses. There will always be cost and other performance requirements, but the safety requirements are always necessary to prevent accidents from occurring. Some of the scenarios and requirements may not apply for a system. A team of experts involved with the system needs to determine which scenarios apply and which scenarios are missing or need to be included. The scenarios cover a large range of potential causes. Some causal scenarios apply to a plethora of the UCAs but were only applied to one just to show how it could be applied. For example, a broken wire could be the case for all of the UCAs, but it was only applied to a few to identify more issues with flawed requirements that only STPA would find.

A list of the full scenarios and requirements is located in Appendix B. Some scenarios and requirements are similar for different UCAs. Those cases are highlighted in the appendix. For example, jamming can be used for multiple scenarios so each scenario with jamming provides a link to an original scenario for reference. Additionally, most of the scenarios follow the format developed by John Thomas. This format provides a simple method for beginners to develop strong scenarios without trying to spend time thinking about how something could occur. By following the basic scenarios and refining them, one can analyze what makes a scenario the strongest and it helps prevent too many repeat scenarios.

Some of the scenarios will display an additional control structure not shown in the generic MUM-T example. These additional structures provide context describing how the scenarios are developed from in-depth control loops that were not included in the generic structure for sake of space. However, these in-depth structures with actuators and feedback mechanisms are necessary to develop strong scenarios.

Finally, although not required specifically for this analysis, the final step of STPA is to include recommendations for the system design based on the scenarios. The reason this was not needed for the AFRL analysis is because the actual system design requires tradeoffs between safety requirements and others like performance, cost, and more. Since our laboratory is not familiar with the necessary tradeoffs, the analysis stopped with scenarios and constraints from scenarios. However, this leads to the final step in the STPA process which is identifying what specific safety design considerations are necessary for a safe system and evaluating whether implementing these constraints prevents the UCA's from occurring. Examples of potential recommendations based on the prior scenarios could include:

- The Team Lead receives an audible and visual alert when a UAV begins landing (can be disabled during landing mission phase)

- The Team Lead and Ground Station can select modes of waypoint travel for specific UAV(s)

- The Team Lead receives a message notification to alert the Ground Station when the mission is over

- The Team Lead and UAV(s) utilize updated cryptography keys every 5 seconds

- Visually guided missiles utilize sensor information from UAV(s) to narrow down target location

Chapter 4

Summary and Conclusion

This analysis provided an overview of how STPA can be applied to MUM-T. One major contribution is identifying what requirements are necessary for air to air combat or ISR missions involving manned and unmanned aircraft. The case study in this thesis included an in-depth hazard analysis. The functional control structure that was modeled is generic enough that it can be modified to fit other applications as well. Furthermore, the full results of the STPA case study presented in Appendices A and B identify unsafe control actions and causal scenarios that may be applicable not only for ISR missions or combat missions, but also similar missions used in additional applications. It is the hope of the author that the results obtained through the use of STPA on this case study can be used to make design changes as well as procedural changes that will increase the safety of UAVs and autonomous systems used in MUM-T missions.

There were several different UAV architectures presented in Chapter 2. Each was designed for various payloads, various mission sets, and various communication capabilities. Therefore, trying to retrofit a UAV architecture for a MUM-T mission can result in exponentially greater costs and schedule delays than designing and building a MUM-T architecture from scratch. A perfect example of this concept is described by Susan Gomez, a GPS flight test engineering at NASA Johnson Space Center. NASA tried to use a common navigation sensor for the Shuttle, International Space Station (ISS), and crew return vehicle to save money. However, the shuttle's receiver required a different software interface because the old navigation system on it had requirements to maintain. The ISS and CRV were similar in hardware and the software was intended to be used on both, but in the end only hardware was the same thing between them.

There were three main requirements for the ISS, two of which could be met by a Honeywell GPS, but one could not be met which was for the attitude accuracy due to filtering on the ISS. Time outputs were incorrect because the algorithm for the GPS was for a particular implementation and integer resolution. There were spikes in times, flat lines in data, and the flight controller clock readings reported different positions at the same time. This was all due to coding deficiencies in the Honeywell System Processor code. Sometimes the ISS navigation would even stop processing after getting a "Not a number" error and it would try to reset itself when it could not perform a reset. The integer search method was intended for aircraft which have completely different specifications.

For navigation the velocity and position data did not match up. There was no way to match up the robot arm with where it was supposed to be grabbing supplies and bringing them into the ISS from the docking station. It was hard to boost the ISS to a new location. All in all, the major lessons learned were that the current software quality process was inadequate. Additionally, extensive testing did not overcome the horrible design. The timeline for the project was too rapid to properly test everything. The contract type for the work did not allow for variability in the schedule. Finally, the manufacturer must be an expert in what they are developing.

Gomez makes a plethora of excellent points, especially as it relates to government-contractor projects. If something is designed poorly, it does not matter how much testing is done; it will still be bad. The only difference is you will not know it is bad depending on the test inputs and environmental conditions you give it. Testing cannot be used to verify or validate a design. It is a new way of thinking for engineers because a lot of thought and hard work goes into the system, so telling someone that the design has flaws is hard to hear. However, this is not the purpose of testing. Validation only comes from people using it and having the system operate successfully. Firm fixed contracts were only used in the Air Force when operators knew exactly what they wanted, and the contractor was going to build exactly that system. If there is any chance for changes or offsets, these contracts were not the best solution. The situation becomes worse when the contract is already in place for a system, and it cannot be undone until after the system is already built [55].

Another NASA engineer, John Goodman, made similar points about the navigation units for the Space Shuttle. NASA assumed that using a common GPS and EGI for the shuttle program would save costs, but the problem is these units only fit the requirements of the original customers. It was assumed that using units with prior shuttle experience and performance characteristics would be better in terms of reducing cost than adapting and testing new units. However, trouble ensued and Goodman notes several of the same lessons as Gomez. Fixed price contracts should not be used for development work. Technical issues on a project need to be addressed early, and resources and schedule need to be accommodating to include testing. Interim software versions should be flown and tested along the way to the finished product. The vendor and the customer need to keep a close relationship to overcome work culture issues and make sure they are working as a team instead of working as adversaries with competing demands or goals.

The vendor should understand what the customer plans on using the product for and should talk to other groups who have used the product before. Test as much as possible; there should never be limits on how much testing is done. Independent verification and validation are important, and tests done before critical decisions are even more valuable. Issues with proprietary documentation should be resolved in the beginning to prevent issues with later. The Interface Control Document is extremely important because it identifies the inputs and outputs between each subsystem. The theory behind algorithm requirements need to be documented in addition to the requirements. There are also lessons from other programs including ignoring issues and attributing them to a new way of doing things. Not defining roles and responsibilities. Lack of management oversight or involvement.

Goodman points out a lot of notes that are quite valuable and great to keep in mind for projects. In the end, an aviation navigation unit repurposed for space applications should be a development project and not a plug and play project with a fixed price. What is particularly interesting is that he points out how projects that go over budget and over schedule have a big impact on the engineers involved. It leads to a distrust of the technology and potential unrealistic expectations on future projects. This is something leaders of projects do not emphasize enough. A project going over-schedule and over-budget is not bad just for the project or the product. It is bad for everyone involved. Trust issues arise. People no longer want to work together or cooperate. Sometimes it takes decades to remake the severed relationships. It can lead to people quitting jobs and turning over their lives to do something else. No one thinks about all of that

when starting a project, but it is important to remember people are people and not just man hours on a spreadsheet [56].

In the end, there are several issues that arise when trying to repurpose technology or systems for another mission. The cost and schedule savings are tempting, but the safety and organizational issues can abound. Every system and situation is different, but when examining the differences between the UAV architectures that currently exist and the MUM-T requirements generated, a new system would mostly likely be safer, more secure, cheaper, and take less time than trying to match a current architecture for MUM-T missions.

Other mission sets can be considered, as well as more complex functional control structures. There could be tank commanders that have to provide control actions to tanks and communicate with the Team Lead. There could be satellites controlled by separate Ground Stations and ships providing radar information to the UAV Ground Station. There can even be multiple autonomous controllers for the UAV(s) that control different functions. One could imagine an extremely complex diagram. STPA is necessary for future work in this area. There are numerous different control structures that are possible with varying control actions and feedback. This means that numerous STPAs must be done to identify the safety and security differences between the options. However, this is the advantage of STPA. Having to perform multiple fault trees takes significant amounts of time and effort. STPA allows a team to alter the analysis with ease to understand whether design changes create or remove unsafe control actions.

STPA has become critical now more than ever. In an era where systems are made of hundreds of subcomponents that come from a wide variety of manufacturers, it is absolutely essential to include system safety early in the design process. UAVs are developed by combining sensors, mapping software, connectivity programs, goal seek algorithms, special microprocessors, and developer tools to edit source code as needed. There are hundreds of companies building within each of those fields, and the Original Equipment Manufacturers are pulling from all of them to put together these machines. Companies believe that if the components are tested and safe individually, then they will work within the context of the whole system which simply is not the case. As outlined by STAMP, accidents happen due to interactions of components even everything works as designed without fault.

By appropriately defining requirements and performing specific forms of verification and validation during the design and development process, the Air Force can utilize a phased deployment for MUM-T systems that will build trust with their operators while ensuring safety is maintained. While traditional validation worked well for systems with mostly hardware, modern automation systems are complex and require firm requirements for safety since validation can no longer provide safety assurances beyond individual components. By exposing systems to a plethora of environments including simulations, closed air courses, and public air space, robust testing can be done to ensure that the requirements defined in STPA are appropriate for ensuring the safety of the system. Only in testing the boundaries and limits of a system will missing safety or security requirements become evident.

To conclude, STPA is a structured method to generate safety and security requirements early in the lifecycle design process. It allows developers to overcome the layers of complexity that arise

as systems include interactions between people, software, hardware, sensors, actuators, the environment, other controllers, etc. Process models and mental models must be taken into account, and STPA is the only hazard analysis technique that can handle this complexity. As a result, STPA provides system developers with information to make informed decisions about their system. As developers evaluate risk and different design options, STPA will provide evidence to allow developers to make cases supporting their designs for safety and security.

References

[1] P. Dockrill, "First-Ever Drone Swarm Attack Has Struck Russian Military Bases, Sources Claim," *Science Alert*, p. 2, 11 January 2018.

[2] S. Morgenthaler, T. Braun, Z. Zhao, T. Staub and M. Anwander, "UAVNet: A mobile wireless mesh network using Unmanned Aerial Vehicles," in *Globecom Workshops*, Anaheim, CA, USA, 2012.

[3] D. Kingston, "OpenUxAS Introduction," Air Force Research Laboratory, 10 September 2017. [Online]. Available: https://github.com/afrl-rq/OpenUxAS. [Accessed 21 January 2018].

[4] N. G. Leveson, Engineering a Safer World: Systems Thinking Applied to Safety, Cambridge, MA, USA: The MIT Press, 2011.

[5] G. Weinberg, An Introduction to General Systems Thinking, New York: John Wiley & Sons, 1975.

[6] N. G. Leveson, "Model-Based Analysis of Socio-Technical Risk," Massachusetts Institute of Technology Engineering Systems Division, Cambridge, MA, USA, 2002.

[7] B. Abrecht and N. G. Leveson, "Systems Theoretic Process Analysis of an Offshore Supply Vessel Dynamic Positioning System," MIT Lincoln Laboratory, Cambridge, MA, USA, 2016.

[8] S. Folse, "Systems Theoretic Process Analysis of Small Unmanned Aerial System Use at Edwards Air Force Base," Massachusetts Institute of Technology, Cambridge, MA, USA, 2017.

[9] F. Frola and C. Miller, "System Safety in aircraft acquisition," in *Logistics Management Institute*, Washington D.C., January 1984.

[10] B. W. Boehm, Software Engineering Economics, Redondo Beach, CA, USA: Prentice Hall, 1981.

[11] A. Strafaci, "What does BIM mean for Civil Engineers?," CE News, Transportation, 2008.

[12] N. G. Leveson, Safeware: System Safety and Computers, Boston, MA, USA: Addison-Wesley Publishing Company, 1995.

[13] M. Erdelj and E. Natalizio, "UAV-Assisted Disaster Management Applications and Open Issues," in *International Conference on Computing, Networking, and Communications*, Kauai, HI, USA, 2016.

[14] S. Summers, "Systems Theoretic Process Analysis Applied to Air Force Acquisition Technical Requirements Development," Massachusetts Institute of Technology, Cambridge, MA, USA, 2017.

[15] A. Long, "Beauty & The Beast - Use and Abuse of Fault Tree as a Tool," University of Alabama, Huntsville, AL, USA, 2009.

[16] N. Balakrishnan, "Dependability in Medicine and Neurology," in *An Overview of System Safety Assessment*, Switzerland, Springer International Publishing, 2015.

[17] Product Quality Research Institute, "Hazard & Operability Analysis," Manufacturing Technology Committee - Risk Management Working Group, pqri.org/HAZOP_Training_Guide.pdf, 2015.

[18] C. T. Eschenbach, "Unmanned Aircraft Systems and Manned-Unmanned Teaming," Army Aviation Magazine, 13 November 2017. [Online]. Available: http://armyaviationmagazine.com/index.php/archive/not-so-current/589-unmanned-aircraft-systems-manned-unmanned-teaming.

[19] "GA-ASI Predator XP with MUM-T: A Leap Forward in Situational Awareness for Combat Operations," General Atomics, 10 October 2016. [Online]. Available: https://www.janes.com/images/assets/051/80051/GA-ASI-White-Paper-Janes360-102517.pdf.

[20] D. Parsons, "Army Looks to Industry for Innovation in Manned-Unmanned Teaming," Avionics International, 25 May 2018. [Online]. Available: https://www.aviationtoday.com/2018/05/25/army-looks-industry-innovation-manned-unmanned-teaming/.

[21] "US Army Tests DARPA Autonomous Flight System, Pursuing Integration with Black Hawk," Defense Advanced Research Projects Agency, 29 October 2018. [Online]. Available: https://www.darpa.mil/news-events/2018-10-29.

[22] J. Judson, "These two drones are leaders in accident rates. How is the US Army responding?," Defense News, 25 April 2018. [Online]. Available: https://www.defensenews.com/digital-show-dailies/aaaa/2018/04/25/these-two-drones-are-leaders-in-accident-rates-how-is-the-us-army-responding/.

[23] V. Insinna, "Army's helicopter-drone teams to get capability boost in 2019," Defense News, 8 October 2017. [Online]. Available: https://www.defensenews.com/digital-show-dailies/ausa/2017/10/09/armys-helicopter-drone-teams-to-get-capability-boost-in-2019/.

[24] C. Whitlock, "More Air Force drones are crashing than ever as mysterious new problems emerge," The Washington Post, 20 January 2016. [Online]. Available: https://www.washingtonpost.com/news/checkpoint/wp/2016/01/19/more-u-s-military-drones-are-crashing-than-ever-as-new-problems-emerge/?utm_term=.be2608be145c.

[25] K. L. Hobbs, C. Cargal, E. Feron and R. S. Burns, "Early Safety Anslysis of Manned-Unmanned Team System," in *The American Institute of Aeronautics and Astronautics*, 2018.

[26] D. R. Montes, Using STPA to Inform Developmental Product Testing, Cambridge, MA, USA: Massachusetts Institute of Technology, 2016.

[27] N. Leveson, D. Horney, R. Porada and P. Ward, "Safety and Security in the Future Vertical Lift Program," Massachusetts Institute of Technology, Cambridge, MA, USA, 2016.

[28] R. J. B. Jr., D. P. Watson, D. H. Scheidt and K. L. Moore, "Flight Demonstration of Unmanned Aerial Vehicle Swarming Concepts," *Johns Hopkins APL Technical Digest*, vol. 27, no. 1, 2006.

[29] C. Hesiey, A. Hendrickson, B. Chludzinski, R. Cole, M. Ford, L. Herbek, M. Ljungberg, Z. Magdum, D. Marquis, A. Mezhirov, J. Pennell, T. Roe and A. Weinert, "A Reference Software Architecture to Support Unmanned Aircraft Integration in the National Airspace System," *Journal of Intelligent Robotic Systems*, pp. DOI 10.1007/s10846-012-9691-8, 2012.

[30] J. How, C. Frasher, K. Culling, L. Bertuccelli, O. Toupet, L. Brunet, A. Bachrach and N. Roy, "Increasing Autonomy of UAVs," in *Institute of Electrical and Electronics Engineers*, IEEE 16.2, 2009.

[31] E. Pastor, C. Barrado, P. Royo, J. Lopez, E. Santamaria and X. Prats, "An Architecture for the Seamless Integration of UAS Remote Sensing Missions," in *American Institute of Aeronautics and Astronautics Infotech at Aerospace Conference*, Seattle, WA, USA, 2009.

[32] J. L. Sanchez-Lopez, R. A. S. Fernandez, H. Bavle, C. Sampedro, M. Molina, J. Pestana and P. Campoy, "AEROSTACK: An Architecture and Open-Source Software Framework for Aerial Robotics," in *Institute of Electrical and Electronics Engineers*, Arlington, VA, USA, 2016.

[33] C. Sampedro, H. Bavle, J. L. Sanchez-Lopez and R. A. Suarez Fernandez, "A Flexible and Dynamic Mission Planning Architecture For UAV," Madrid, Spain, 2015.

[34] S. Hayat, E. Yanmaz and R. Muzaffar, "Survey of Unmanned Aerial Vehicle Networks for Civil Applications: A Communications Standpoint," in *Institute for Electronics and Electrical Engineers*, Kuala Lumpur, Malaysia, 2016.

[35] J. Decuir, "Introducing bluetooth smart Part 1: A look at both classic and new technologies"," *IEEE Consumer Electronics Magazine*, vol. 3, no. 1, pp. 12-18, 2014.

[36] J. Lee, Y. Su and C. Shen, "A comparative study of wireless protocols: Bluetooth, UWB, Zigbee, and Wi-Fi," in *IEEE Annual Conference of Industrial Electronics Society (IECON)*, Taipei, Taiwan, 2007.

[37] L. Miao, K. Djouani, B. V. Wyck and Y. Hamman, "Evaluation and Enhacement of IEEE 802.11 p standard: A survey," *Mobile Computing*, vol. 1, no. 1, 2012.

[38] P. Jose and A. Gutierrez, "Packet Scheduling and Quality of Service in HSDPA," Department of Communication Technology, Institute of Electronic Systems, Aalborg University, PhD Thesis, 2003.

[39] E. Yanmaz, M. Quaritsch, S. Yahyanejad, B. Rinner, H. Hellwagner and C. Bettstetter, "Communication and Coordination for Drone Networks," in *Ad Hoc Networks: 8th International Conference, ADHOCNETS*, Ottawa, Canada, Springer Nature, 2016, pp. 79-91.

[40] D. Gonzales and S. Harting, "Designing Unmanned Systems with Greater Autonomy," The RAND Corporation, Santa Monica, CA, USA, 2014.

[41] N. S. Jaros, "A Day in the Life of a Fighter Pilot," Fighter Sweep, 7 February 2017. [Online]. Available: https://fightersweep.com/7013/day-life-fighter-pilot-part/.

[42] B. Burfeind, Interviewee, *Major.* [Interview]. 12-13 December 2018.

[43] S. Richardson, Interviewee, *Major.* [Interview]. 15 July 2018.

[44] R. Gordon, Interviewee, *Lt. Col.*. [Interview]. 22 June 2018.

[45] A. Nandaviri, "Mobile Robotics Avionics Block Diagram," 17 February 2014. [Online].

[46] R. Parasuraman, T. B. Sheridan and C. D. Wickens, "A Model for Types and Levels of Human Interaction with Automation," *IEEE Transactions on System, Man, and Cybernetics*, vol. 30, no. 3, pp. 286-297, May May 2000.

[47] J. Thomas, N. Leveson, N. Ishimama, M. Katahira, N. Hoshino and K. Kakimoto, "A Process for STPA: STAMP Accident Model of HITOMI and Expansion to Future Safety Culture," in *STAMP conference*, Cambridge, MA, USA, 2017.

[48] N. G. Leveson and J. P. Thomas, "STPA Handbook," Cambridge, MA, USA, March 2018.

[49] J. Holguin, "Patriot Eyed in Downing of Navy Jet," CBS, 14 April 2003. [Online]. Available: https://www.cbsnews.com/news/patriot-eyed-in-downing-of-navy-jet/.

[50] I. Drury, "Officers' errors 'killed soldier in friendly fire gunship attack': Coroner condems 'unprofessional' use of grainy images from drone aircraft," DailyMail.com, 7 September 2012. [Online]. Available: https://www.dailymail.co.uk/news/article-2199706/British-soldier-killed-US-Apache-helicopter-friendly-Afghanistan-died-result-mistaken-beliefs-cumulative-failures.html.

[51] T. Arango, "War in Iraq Defies U.S. Timetable for End of Combat," The New York Times, 2 July 2010. [Online]. Available: https://www.nytimes.com/2010/07/03/world/middleeast/03iraq.html.

[52] C. Arguillas, ""Friendly fire" in Marawi: 12 killed in 51 days; probe results out after end of combat operations," Minda News, 13 July 2017. [Online]. Available: ♣ http://www.mindanews.com/top-stories/2017/07/friendly-fire-in-marawi-12-killed-in-51-days-probe-results-out-after-end-of-combat-operations/.

[53] A. Macias, "America's most expensive weapons system just got a little cheaper," CNBC, 28 September 2018. [Online]. Available: https://www.cnbc.com/2018/09/28/f-35-fighter-jets-americas-most-expensive-weapons-system-just-got-a-little-cheaper.html.

[54] S. Roblin, "FA-50 Golden Eagle: The Low-Cost Fighter that Might See Some Serious Combat," The National Interest, 11 September 2016. [Online]. Available: https://nationalinterest.org/blog/fa-50-golden-eagle-the-low-cost-fighter-might-see-some-17649.

[55] S. F. Gomez and M. L. Lammers, "Lessons Learned from Two Years of On-Orbit Global Positioning System Experience on International Spacae Station," pp. 1-9, 2004.

[56] J. L. Goodman, "Lessons Learned from Flights of "Off the Shelf" Aviation Navigation Units on the Space Shuttle," pp. 3-10, 2003.

Appendix A: Unsafe Control Actions

Team Lead

Control Action	Not providing	Providing	Too early/Late	Applied too long/stopped too short
Send "Fix on target" command	Team Lead does not send "fix on target" command when enemies are within range (H3)	Team lead sends "fix on target" command to the wrong UAV (H3)	Team Lead sends "fix on target" command before receiving confirmation from crew chief (H3)	N/A
		Team Lead provides "fix on target" command for the wrong target (H3)	Team Lead sends "fix on target" command after the target is out of range (H3)	
		Team Lead sends "fix on target" command for a civilian/friendly target (H3, H6)	Team Lead sends "fix on target" command before AuC has an identification of the target (H3)	
Send "Identify object" command	Team Lead does not send "identify object" command when the mission requires a determination of the objects in a region (H3)	Team Lead sends "identify object" command incorrectly when there are targets that need to be identified (H3)	Team Lead sends "identify object" command after the objects are already out of range (H3)	N/A

Send "Track target" command	Team Lead does not send "track target" command when the mission requires it (H3)	Team Lead sends "track target" command for the wrong target (H3)	Team Lead sends "track target" command too early and exposes the operation to the enemy (H3)	N/A
		Team Lead provides "track target" command incorrectly (H3)	Team Lead sends "track target" command too late after the mission window closed or the target leaves (H3)	
Send "Search for target" command	Team Lead does not send "search for target" command within the mission time (H3)	Team Lead sends "search for target" command for an old target when the mission is updated (H3)	Team Lead sends "search for target" command for the target too late after the target is no longer within detection range (H3)	N/A
Provide Formation	Team Lead does not provide formation command to UAVs when enemies are within range (H3)	Team Lead provides formation command that cause the UAVs to turn the wrong direction (H3)	Team Lead provides formation command before the UAVs reconnect lost LOS (H3)	N/A
		Team Lead provides formation command that put the UAVs in restricted airspace (H4)	Team Lead provides formation command too late after the enemies have fled (H3)	
Provide "Fire at target" command	Team Lead does not provide "fire at target" command when the mission	Team Lead provides "fire at target" command to the wrong UAV (H5)	Team Lead authorizes the fire command too late after the	N/A

	requires it or they're being fired at (H3)		UAVs are being fired at (H3)	
		Team Lead provides "fire at target" command for the wrong target (H5)	Team Lead authorizes the fire command too early before the UAVs have targeted the right object or enemy (H4, H6)	
		Team Lead provides "fire at target" command when civilians or allies are near targets (H6)	Team Lead authorizes the fire command too early before he/she is given authorization by the crew chief (H4)	
Fire at target	Team Lead does not fire at a target when the mission requires it (H3)	Team Lead fires at a friendly or civilian target (H6)	Team Lead fires at a target before receiving authorization from a general (H6)	N/A
		Team Lead fires at the wrong target (H6)	Team Lead fires at a target after the target is out of range (H3)	
Provide waypoints	Team Lead does not provide waypoints when there are no more in his FMS (H3)	Team Lead provides the wrong waypoint to the FMS (H3)	Team Lead provides a waypoint before confirming the location with the GS or MP (H3)	N/A
			Team Lead provides a waypoint after the location is no longer needed to	

			complete the mission (H3)	
Set airspeed	Team Lead does not set the airspeed (H1, H2)	Team Lead sets the airspeed to the wrong setting (H1, H2)	Team Lead sets the airspeed after a target is already out of range (H3)	N/A
Set attitude	Team Lead does not set the attitude when the autopilot is disengaged (H1, H2)	Team Lead sets the attitude to the wrong setting (H1, H2)	Team Lead sets the attitude before the autopilot disengages (H1, H2)	N/A

Ground Station

Control Action	Not providing	Providing	Too early/Late	Applied too long/stopped too short
Send Override Command	Ground Station does not send an override command when a UAV is performing an improper action (firing at friendly, engaging wrong target, maneuvering inappropriately, etc.) (H1, H2, H3, H4, H5, H6)	Ground Station overrides the UAV when the UAV is appropriately accomplishing the mission and it loses a target (H4)	N/A	N/A
Land Now	Ground Station does not provide land now when the aircraft is in a pattern and at minimum fuel (H1, H2)	Ground Station provides land now when the runway is not clear (H1)	Ground Station provides land now too early before the aircraft completes the airfield arrival procedure (H1)	N/A

	Ground Station does not provide land now when the aircraft is at the airfield and other aircraft are trying to enter a pattern (H1)	Ground Station provides land now when the aircraft is above unstable terrain (H1)	Ground Station provides land now too late after the UAV has already entered a restricted air space (H5)	N/A
	Ground Station does not provide land now when the aircraft is ready to land, resulting in the aircraft reaching bingo fuel and experiencing loss of control (H1, H2)	Ground Station provides land now when the aircraft is in a restricted landing area (H5)	Ground Station provides land now too late after the UAV is already out of fuel or is no longer over a safe landing area (H1, H2, H5)	N/A
Take off	Ground Station does not provide takeoff command when the mission starts (H4)	Ground Station provides takeoff command when the runway is not clear (H1)	Ground Station provides takeoff command too early before ground personnel clear the area (H1)	N/A
		Ground Station provides takeoff command when the aircraft is not on the runway (H1)		N/A
Send Altitude command	Ground Station does not provide altitude when waypoints are updated (H1, H2, H5)	Ground Station provides altitude that is below the minimum obstacle clearance (H1)	Ground Station provides altitude too late after LOS is lost and before BLOS is established (H2)	Ground Station provides altitude assignments that exceed the number of waypoints (H4) [too long]

		Ground Station provides altitude that conflicts with another aircraft's altitude (H1)		Ground Station provides fewer altitude assignments than waypoint assignments (H4) [too short]
		Ground Station provides altitude that is above the icing level and the UA flies through clouds (H2)		
Send Airspeed command	Ground Station does not provide airspeed during a flight change or bad environmental conditions (H2, H3, H5)	Ground Station provides airspeed that is below the stall speed (H2)	Ground Station provides airspeed after UAV stalls due to slowed flight (H2)	Ground Station provides an airspeed assignments that exceeds the waypoints (H4)
		Ground Station provides airspeed that is above VNE (H2)	Ground Station provides airspeed after structural damage from flying above VNE (H2)	Ground Station provides airspeed assignments fewer than the number of waypoints (H4)
		Ground Station provides airspeed that is based on auto flight planning fuel duration, but a higher speed is set (H2, H4)		

		Ground Station provides an airspeed that conflicts with another aircraft (H1)		
Start engine	Ground Station does not provide engine start during prelaunch engine run-up (H4)	Ground Station provides engine start when ground personnel are near the propellers (H1)	Ground Station provides engine start too late after the engine fails in flight, but after the aircraft is committed to landing (H1)	N/A
	Ground Station does not provide engine start during before takeoff procedure (H4)			
	Ground Station does not provide engine start when the engine fails in flight and needs a restart (H2)			
Stop engine	Ground Station does not stop engine when the fuel disconnects from the main valve (H2)	Ground Station stops the engine when the aircraft is performing the mission (H4)	Ground Station stops the engine too early before the mission is complete (H4)	N/A
	Ground Station does not stop engine when a foreign object gets stuck in it (H2)		Ground Station stops the engine too late after an object goes into the cavity (H2)	

Begin Lost Link Procedure	Ground Station does not provide lost link when a connection is lost with another aircraft (H4)	Ground Station provides lost link procedures when the waypoints now conliftc with another aircraft (H1)	Ground Station provides lost link procedure before the conflicting traffic, terrain, or weather neccesitate a need to update (H1)	Ground Station provides lost link when the waypoints exceed the storage capacity of the autopilot (H1)
		Ground Station provides lost link procedures that are not above minimum obstruction clearance altitude (MOCA) (H1)		
Power on payload	Ground Station does not power on payload when the aircraft is over the target area (H4)	Ground Station powers on payload when the alternator fails (H4)	Ground Station powers on the payload too early before it is needed, leading to the UA running out of battery and stalling (H2)	N/A
			Ground Station powers on the payload too late after the mission window has already passed (H4)	
Power off payload	Ground Station does not power off when the alternator fails (H4)	Ground Station powers off payload when the aircraft is over the target area (H4)	Ground Station powers off the payload too early before it is finished with the mission (H4)	N/A

			Ground Station powers off the payload too late to conserve energy leading to instability (H2)	

Autonomous Controller

Control Action	Not providing	Providing	Too early/Late	Applied too long/stopped too short
Change attitude	Autonomous Controller does not change attitude when the UAV is off course (H1, H2, H3, H4)	Autonomous Controller changes attitude and the command exceeds physical limits (H2)	Autonomous Controller changes attitude too late when the throttle is reduced to descend, but the pitch down is delayed (H2)	Autonomous Controller changes attitude command for too long and the actuator displacement is not back to neutral when the aircraft reaches the target attitude (H1, H2, H3, H4)
		Autonomous Controller changes attitude that steers the aircraft off course (H3, H4)	Autonomous Controller changes attitude too early when the throttle is increased to climb, but the pitch up is delayed (H2)	Autonomous Controller changes attitude for too short and the actuator displacement is back to neutral before the

				aircraft reaches the target attitude (H1, H2, H3, H4)
Set/Adjust Throttle	Autonomous Controller does not adjust throttle when environmental conditions change (H2, H3)	Autonomous Controller provides throttle but it is too low to maintain an airspeed above stall speed (H2)	Autonomous Controller provides throttle too late after the aircraft flares for landing (H1, H2)	Autonomous Controller provides throttle for too long when the aircraft is at target speed, but it's not reduced before reaching VNE (H2)
	Autonomous Controller does not throttle in a situation where lift is reduced (sustained turn) (H2)	Autonomous Controller provides throttle that accelerates the aircraft above VNE (H2)		Autonomous Controller provides throttle for too short when the aircraft decelerates to target speed, but not increased before reaching stall speed (H2)
Release weapon	Autonomous Controller does not release weapon when the Team Lead commands it (H3)	Autonomous Controller releases a missile for the wrong target (H5)	Autonomous Controller releases a missile before receiving a command from the Team Lead (H5)	N/A

			Autonomous Controller releases a missile after the target is out of range (H3, H5)	
Implement task	Autonomous Controller does not implement a task from the Team Lead (H3)	Autonomous Controller implements a task using the wrong UAV (H3)	Autonomous Controller implements a task before completing an algorithm to determine the optimal task completion (H3)	N/A
		Autonomous Controller implements a task when there is no command from an authorized command provider (H3)	Autonomous Controller implements a task after the target is no longer necessary (H3)	
			Autonomous Controller implements a task after a UAV cannot complete the task (H3)	
Power on payload	Autonomous Controller does not power on payload when the payload is needed to complete the mission (H3)	Autonomous Controller powers on payload when power needs to be conserved (H3)	Autonomous Controller powers on payload before ensuring the UAV can provide power to it (H3)	N/A
			Autonomous Controller powers on payload after the target is already out of range (H3)	

Mission Planners

Control Action	Not providing	Providing	Too early/Late	Applied too long/stopped too short
Issue mission plan	Mission planner does not issue the mission plan when a plan comes in from the AOC (H3)	Mission planners issue the mission plan when it has flaws in it (improper weapon target pairings, etc.) (H3)	Mission planners issue plan too late when the TL and UAVs won't have time to accomplish it (H3)	N/A
Update mission	Mission planner does not provide updates when the mission gets updated (H3)	Mission planners update the mission by providing an improper plan that is impossible to accomplish or is fatal, confusing the TL and UAVs (H3)	Mission planners update the mission too late after the TL or UAV already performed an improper action (H3)	N/A
			Mission planners update the mission too early before the approval to update was given by a General (H3)	

Crew Chief

Control Action	Not providing	Providing	Too early/Late	Applied too long/stopped too short

Inspect aircraft	Crew Chief does not inspect aircraft prior to takeoff (H2)	Crew Chief inspects the aircraft poorly (H2)	Crew Chief inspects the aircraft prior to maintenance being completed (H2)	Crew Chief inspects the aircraft partially (H2)
Provide resources	Crew Chief does not provide resources to the Ground Station to maintain the aircraft (H2)	Crew Chief provides the wrong resources to the maintenance crew (H2)	Crew Chief provides resources after the mission window has passed (H3)	N/A

Air Traffic Control

Control Action	Not providing	Providing	Too early/Late	Applied too long/stopped too short
Provide coordination	ATC does not coordinate the aircraft when they are within range (H1)	ATC provides inappropriate coordination when the aircraft are within range (H1)	ATC provides coordination too late after the aircraft are beyond the minimum separation requirement (H1)	N/A
Grant Clearance	ATC does not grant clearance to the aircraft to take off (H3)	ATC grants clearance when the airspace isn't clear or there's an object on the runway (H1)		N/A

Issue ground instructions	ATC does not issue ground instructions to the UAVs or TL when the aircraft need to take off (H3)	ATC issues ground instructions incorrectly (on the wrong channel, etc.) (H3)	ATC issues ground instructions too early before the ground is clear for taxi (H1)	N/A
Issue approach instructions	ATC does not issue approach instructions when the aircraft need to land (H3)	ATC issues approach instructions for the wrong runway (H1, H2)	ATC issues approach instructions too early before the runway is clear (H1)	N/A
Issue departure instructions	ATC does not issue departure instructions when the aircraft need to take off (H3)	ATC issues departure instructions incorrectly (when they're trying to land, missing heading, etc.) (H3)	ATC issues departure instructions too early before the runway is clear (H1)	N/A

Appendix B: Scenarios and Requirements

Some scenarios will have links associated with them to identify if the scenario has occurred in the past. The link will identify a website, news article, etc. that provides a similar event.

Team Lead Scenarios/Requirements

The "fix on target" command assumes that the Master Arm switch is on.

UCA 1: Team Lead does not provide "fix on target" command to UAV(s) when targets/enemies are within range (H3)

Scenario:

- The Team Lead provides a "fix on target" command
    - Broken wire
    - The Autonomous Controller (AuC) is out of range from the Team Lead
    - DoS attack on AuC prevents command from reaching it
    - Jamming
    - Not enough power to transfer command
    - Command corrupted in transmission by fault injection or SQL injection
- A "fix on target" command is not sent to the Autonomous Controller

Possible Requirements are:

- The communication path from the Team Lead to the FMS must be inspected prior to takeoff
- The Team Lead must be alerted if the Autonomous Controller is out of range for implementing a command
- The AuC must be able to handle TBD commands
- The AuC must receive TBD power from the main battery
- The Team Lead must be able to provide alternate "fix on target" commands if several are corrupted

Scenario:

- The Team Lead receives accurate feedback on the target selection
    - However, the Team Lead may incorrectly believe the target must no longer be fired at. This could occur if the Team Lead receives a false communication from an attacker that the target is no longer of interest

    - The Team Lead may believe that the target can be fired at without a fix on target
    - An attacker causes a mode change to a friendly flight mode that eliminates the ability to provide "fix on target" commands
    - An attacker provides a command that disables the "fix on target" command
    - The Team Lead receives conflicting feedback from ATC that the selected aircraft is a civilian aircraft and not an enemy missile battery
    - The Team Lead believes the AuC will automatically fix on target
- The Team Lead does not provide the "fix on target" command

Possible Requirements are:

- The Team Lead must be able to distinguish false communication from reliable team member comms
- The Team Lead must be notified if a weapon-target pairing requires a fix on target
- The Team Lead must be able to provide "fix on target" commands in all flight modes
- The "fix on target" command must not be disabled unless the authorization comes from an authorized source
- The Team Lead must be able to have an alternate controller (AuC, missile battery, Ground Station, etc.) identify a potential target
- The Team Lead must be notified if the AuC will apply a "fix on target" command after selecting a target

Scenario:

- The Team Lead receives inaccurate feedback about the target selection
    - The FMS is experiencing delayed processing and is unable to update target selections
    - New target selections are being deleted by a malicious software bug
        - Link: https://www.reuters.com/article/us-airbus-a400m/airbus-knew-of-software-vulnerability-before-a400m-crash-idUSKBN1D819P
    - Target selections are being mistaken for friendly aircraft by the FMS algorithm
- The wrong target or no target is selected

Possible Requirements are:

- The Team Lead must receive a calculated delay based on the current processing capabilities and signal power losses
- The FMS must test "fix on target" commands following updates
- The software must be protected against changes while in-flight
- The Team Lead must see the metadata results from the IFF test to verify the interpretation of the FMS in unsure situations

Scenario:

- The "fix on target" command is sent to the AuC
    - The AuC receives a conflicting command to not fix on targets during the mission.

    - The AuC has an incorrect process model that the target is already fixed on.
    - The AuC interprets the "fix on target" command as a search for target command. The machine learning algorithm has a false positive because the command occurred at an unusual time in the mission.
- The AuC does not fix on target

Possible Requirements are:

- The AuC control algorithm logic must account for conflicting control actions
- The AuC must ask for clarification from the Team Lead if a "fix on target" is already applied
- The AuC must ask for clarification if the command interpretation has a false PI of greater than TBD percent

UCA: Team Lead provides "fix on target" command to the wrong UAV (H3)

Scenario:

- The Team Lead provides the "fix on target" command for the right UAV
    - An attacker applies a false injection that causes the FMS to select alternate UAV(s) for "fix on target" commands
- A "fix on target" command for the wrong UAV is sent to the AuC

Possible Requirements are:

- The FMS must alert the Team Lead if a separate UAV is selected to fix on target other than the one selected by the Team Lead

Scenario:

- The Team Lead receives accurate feedback about the UAV that is available to fix on the target
    - The Team Lead incorrectly believes that another UAV can fix on the target if he selects for the UAV to perform the command (incorrect assumption about the FMS control algorithm)
    - The Team Lead chooses a different UAV because he is confused by the user interface and selects the wrong UAV
        - Link: https://trace.tennessee.edu/cgi/viewcontent.cgi?referer=https://www.google.com/&httpsredir=1&article=1321&context=utk_gradthes
    -
- The Team Lead provides the fix on target command to the wrong UAV

Possible Requirements are:

- The Team Lead must be notified if a selected aircraft cannot fix on the target or is not the optimal aircraft to fix on the target

- The interface must ask for confirmation from the Team Lead on inappropriate target fix selections

Scenario:

- The Team Lead receives inaccurate feedback about the UAV that can fix on the target
  - The Team Lead FMS confuses the positions of two UAVs. The FMS calculates the UAV's position incorrectly based on the coordinate data from the GPS satellite.
  - An attacker changes the location of the target on the FMS by constructing a similar target in a separate location.
    - Link:
  - The target changes location faster than the FMS can provide its updated location.
- The wrong UAV is shown fixed on the target

Possible Requirements are:

- The FMS must be able to distinguish UAV(s) TBD feet apart with TBD accuracy
- The UAV(s) and Team Lead must be able to distinguish between true and false targets
- The FMS must update the target's location every TBD seconds and display when the location is updated to the Team Lead

Scenario:

- The "fix on target" command is sent to the AuC for the correct UAV
  - The AuC control algorithm changes the UAV to select the most optimal firing position
  - The AuC implements a delayed command from the Team Lead when a different UAV was selected for the target
  - The AuC cannot identify the selected UAV and chooses an alternate UAV based on optimal firing position and weapon type
- The AuC selects the wrong UAV to fix on target

Possible Requirements are:

- Changes of the UAV selected for "fix on target" must be confirmed by the Team Lead prior to implementation
- Commands that take longer than TBD seconds to implement must notify the Team Lead to cancel the action or wait for completion

UCA: Team Lead provides "fix on target" command for the wrong target (but still an enemy target) (H5)

Scenario:

- The Team Lead provides the "fix on target" command for the right target
  - The FMS cannot identify the target and decides to focus on a different target

    - The target is no longer within range of being fixed by the TL or UAV(s) so the FMS selects a different target
    - There is a delay from an earlier fix on target command that the FMS acts on instead
    - The command only contains an approximate location of the target (within a specific geographical range). There are several potential targets within the range, and the FMS selects the wrong one
- The command is fixed on a different target

Possible Requirements are:

- The FMS must notify the Team Lead if a target cannot be identified or is no longer within range
- The FMS must display all commands in the queue and where in the control loop the command is being implemented
- The FMS must ask for clarification if the target command does not specify a target

Scenario:

- The Team Lead receives accurate feedback about the target to be fixed on
    - However, the Team Lead may incorrectly believe that the wrong target is selected. This could occur if:
        - The Team Lead expects the target in a different location
        - The FMS displayed several targets in the wrong location earlier in the mission
        - The target visually looks like a friendly or civilian
- The Team Lead provides a fix on target for a different target

Possible Requirements are:

- The Team Lead must verify with other mission members (GS, MP, UAV(s), etc.) the target's location and appearance
- The GS must provide target locations if the Team Lead FMS is unable to accurately display targets

Scenario:

- The Team Lead receives inaccurate feedback about the available targets
    - The wrong target has a similar radar and heat signature as the right target
    - The target data is corrupted by an FMS software reset so only some of the targets are displayed
    - The right target is delayed because the FMS is waiting for an IFF response
    - The IFF transponder is degraded and does not receive an IFF response
- The wrong target is selected

Possible Requirements are:

- Targets with similar signatures must provide an additional differing quality for the Team Lead to select the appropriate target
- The target data must be re-loaded if a software reset occurs
- The target data must show the total number of potential targets, IFF responses, and link the responses to targets on the FMS
- The IFF transponder must be able to receive responses from TBD distance

Scenario:

- The "fix on target" command is sent for the right target
    - The AuC confuses the targets with another target. The targets share similar radar signatures, visual recognition, EM frequencies, etc.
    - The AuC is unable to cache the command due to limited memory available because it is processing other data. This causes the AuC to fix on the closest target to avoid delaying the Team Lead.
    - An attacker updates the process model by having the target produce friendly IFF signatures. The AuC algorithm becomes conflicted and decides to fix on a different enemy target.
    - The AuC algorithm accounts for environmental conditions and decides that the target is too far away to properly hit. The AuC decides to fix on a closer target.
- The AuC fixes on the wrong target

Possible Requirements are:

- The AuC must ask for clarification if it is unable to distinguish between targets with similar EM frequencies, visual characteristics, etc.
- The AuC must carry TBD memory to cache commands as they enter the queue
- The AuC must ask for clarification if the prior target is friendly or civilian, or is too far away to properly attack

UCA: Team Lead provides "fix on target" command when a friendly or civilian is the target (H5)

Scenario:

- The Team Lead provides the "fix on target" command for the right target
    - An attacker corrupts the command
    - The FMS transponder responses are switched so friendlies and enemies are switched
- The command targets a friendly or civilian target

Possible Requirements are:

- The FMS must alert the Team Lead if a fix on target is applied to a friendly aircraft
- The IFF transponder wiring must be inspected prior to takeoff

Scenario:

- The Team Lead receives accurate feedback about the friendly and enemy targets
    - A friendly aircraft flies in front of the Team Lead as he applies a sight lock.
    - The Team Lead is an enemy or enemies take over the Team Lead aircraft (unlikely)
- The Team Lead fixes on a friendly or civilian

Possible Requirements are:

- The Team Lead must not be able to provide a target designation without first locking onto the appropriate target
- The missile must calculate trajectories prior to departure from the aircraft
- The Team Lead must be vetted for appropriate character and personality to engage in combat

Scenario:

- The Team Lead receives inaccurate feedback about the targets
    - A radar or visual sensor is damaged. This causes the FMS to classify based solely on IFF responses.
    - The IFF responder received a neutral frequency response
    - The FMS algorithm classified the target as an enemy using limited Enemy Identifying Information (EII)
    - The mission team must be able to disable the Team Lead's capabilities
- The targets are friendlies or civilians

Possible Requirements are:

- The Team Lead must be notified when sensors or other EII equipment are damaged
- The FMS must receive TBD EII when receiving a neutral IFF response
- The FMS must receive full EII (TBD what full EII is)

Scenario:

- The correct target is provided to the AuC
    - An attacker provides malicious code that only allows the AuC to target friendly IFF respondent aircraft
- The AuC targets a friendly or civilian

Possible Requirements are:

- The AuC must ask for clarification if a friendly aircraft is fixed on

UCA: Team Lead provides "fix on target" command before receiving authorization from the Crew Chief (H3)

Scenario:

- The Team Lead is waiting to provide the fix on target command
    - An attacker provides a false fix on target command to the FMS
    - The FMS interprets an identification command as a fix on target command

- A command is sent to the FMS

Possible Requirements are:

- The FMS must alert the Team Lead if a command is provided from an outside source
- The FMS must send the command implemented to the Team Lead

Scenario:

- The Team Lead does not receive authorization
  - The Team Lead is impatient and decides that the target must be taken out immediately
- The Team Lead provides the command

Possible Requirements are:

- The Crew Chief must be able to provide authorization within TBD seconds

Scenario:

- The Team Lead receives inaccurate feedback about the authorization
  - An enemy provides a false authorization
  - There is a delay in transmission such that an earlier approved fix on target is sent to the Team Lead late
  - The Crew Chief's dis-authorization comes across as an authorization due to inaccuracies in the radio (unlikely)
- No authorization was provided

Possible Requirements are:

- The Team Lead must be able to distinguish between true and false Crew Chief communications
- The Team Lead must have a timestamp for when communications are provided
- The Team Lead must receive additional clarification on inaccurate authorizations

Scenario:

- The FMS has a fix on target command with authorization
  - There is a delay in another command from the Team Lead to delete the command
- The AuC receive a fix on target command without authorization

Possible Requirements are:

- Commands to delete a prior command must take precedence over other commands

UCA: Team Lead provides "fix on target" command too late after the target is out of range (H3)

Scenario:

- The Team Lead provides the fix on target while the target is within range
    - The transmission is delayed due to jamming
    - There is a severed wire. The command does not reach the FMS until the Team Lead applies it on a third or fourth try.
    - The FMS is degraded and is delayed in responding to the command
- The FMS receives the command while the target is not within range

Possible Requirements are:

- Communication paths must be protected with anti-jamming technologies
- The FMS must notify the Team Lead on the average transmission time
- The wiring must be inspected prior to takeoff
- The FMS must be able to process fix on target commands within TBD seconds

Scenario:

- The Team Lead receives accurate feedback about the target's locations
    - The Team Lead is waiting for Crew Chief authorization
    - The Team Lead has an inaccurate mental model of how long it takes the AuC to implement a command upon receiving it
- The Team Lead provides the command when the target is out of range

Possible Requirements are:

- The Team Lead must communicate target locations when targets are TBD miles from being out of range to provide time for an authorization
- The Team Lead must receive an estimated time for command implementation by the AuC including sending time

Scenario:

- The Team Lead receives inaccurate feedback about the target's location or the targets themselves
    - The UAV GPS receivers are degraded
    - The FMS is working on updating the locations but cannot provide the locations until the targets are out of range
    - The GPS receivers are inaccurate and provide a flawed location due to improper initial setup for INS
    - The targets show as civilians or friendlies until the FMS can update based on Link 16 and other EII data
- The targets are not within range

Possible Requirements are:

- The GPS receivers must be tested for accuracy prior to takeoff
- The FMS must be able to provide locations within TBD seconds of receiving a command
- The INS setup must ensure an accuracy to within TBD feet

- The estimated processing time for neutral targets must be displayed to the Team Lead

Scenario:
- The FMS sends the command while targets are still within range
    - The AuC is double checking the targets for accuracy, weapon type, etc. and takes too long before fixing on them
- The AuC does not implement the command while targets are within range

Possible Requirements are:
- The Team Lead must be provided an average verification processing time for targets

UCA: Team Lead provides "fix on target" command before AuC has an identification of the target (H3)

Scenario:

- The Team Lead receives accurate feedback that the targets have not been identified
    - The Team Lead believes the AuC will not implement the command until after a delay of TBD seconds, but it implements it earlier
- The Team Lead provides a fix on target command

Possible Requirements are:

- The Team Lead must receive an accurate estimate for command implementation

Scenario:

- The Team Lead receives inaccurate feedback
    - An attacker provides false IFF responses
- The targets have not been identified

Possible Requirements are:

- The Team Lead's FMS must receive the originating location and device IP for IFF responses

UCA: Team Lead does not provide identify target command when the mission requires a determination of the objects in a region (H3)

Scenario:
- The Team Lead provides an identification command
    - Transmission error
    - Delayed
    - Communication Link failure
    - The command lacks authorization for the FMS

    - The command is received when there are no objects to identify
- The FMS does not receive the command

Possible Requirements are:

- The command must be protected from being altered in transit
- The Team Lead must receive the average delay time for sending
- The Team Lead communication path to the FMS must be protected
- The FMS must notify the Team Lead if authorization is needed
- The FMS must notify the Team Lead if there are no targets to be identified

Scenario:

- The Team Lead receives accurate feedback about possible targets
    - The Team Lead believes he already knows that the targets are friendly
    - The Team Lead misinterprets the feedback and believes the FMS already identified the targets as neutral.
    - The Team Lead believes the feedback is inaccurate because the FMS previously showed false targets.
- The Team Lead does not identify them

Possible Requirements are:

- The Team Lead shall check verify targets with the FMS
- The FMS must notify the Team Lead (auditory, haptic, etc.) on a target
- A process is necessary if the FMS is providing consistent false positive targets

Scenario:

- The Team Lead receives inaccurate feedback about the objects
    - A machine learning algorithm in the AuC cannot notice the objects due to an insufficient training set
    - An enemy provides a software bug that causes the algorithm to misread objects within its periphery
- The objects do not show on the Team Lead FMS or are already identified incorrectly

Possible Requirements are:

- The machine learning algorithm must be trained with the appropriate objects
- The machine learning algorithm must be reset if objects are being misread

Scenario:

- The FMS sends the identification command for the correct object
    - The transmission becomes corrupted
    - The AuC defaults to another object that looks similar

    - An attacker provides an identification command that the AuC is still processing
- The AuC moves sensors to identify the wrong object

Possible Requirements are:

- The communication pathway from the Team Lead to the AuC must be protected against signal corruption
- The AuC must match TBD parameters (visual, position, heat signature, size, etc.) to ensure the same target identification as the Team Lead
- The Team Lead must receive command indicators to know what command the AuC is responding to

UCA: Team Lead provides the identify target command incorrectly when there are targets that need to be identified (H3)

Scenario:

- The Team Lead provides the identify target command for the right target
    - Malware deletes the command upon reaching the FMS
    - A software error occurs that does not allow the Team Lead to select a target
    - The FMS is missing a signal authenticator that can accept transmissions from the Team Lead
- The FMS receives a command for a different target

Possible Requirements are:

- The Team Lead must have a redundant method of providing identify command
- The software must be able to reset a prior working version that allows the Team Lead to select targets
- The maintenance team must inspect the aircraft prior to takeoff for all appropriate parts

Scenario:

- The Team Lead receives accurate feedback about the targets to identify
    - The Team Lead identifies other targets based on only his visual because the AuC provided false targets earlier in the mission
    - The Team Lead is identifying old targets
    - The Team Lead implements the default values for identification
    - The Team Lead does not check the environmental conditions for how the AuC is selecting identification for targets
    - The Team Lead does not trust the cost algorithm the AuC uses and decides to choose his own target
- The Team Lead identifies the wrong targets

Possible Requirements are:

- The Ground Station must work with the Team Lead to provide a verification check after altering the AuC's ability to identify targets
- The Team Lead must receive a timestamp and position for targets
- The Team Lead must check the default parameters for target identification options

- The Team Lead must be aware of the environmental assumptions and their effect on the sensors
- The Team Lead must be able to verify targets with the AuC

Scenario:

- The Team Lead receives inaccurate feedback
    - An attacker places false targets within the region
    - The FMS is displaying future locations based on current velocities and attitudes instead of current positions
- The targets are in different locations or are not targets at all

Possible Requirements are:

- The Team Lead and UAV(s) must be able to distinguish between true and false targets
- The user interface must specify the timestamp for information

Scenario:

- The FMS provides the identify command for the right object
    - The AuC already identified the target and defaults to another target
- The AuC identifies the wrong object

Possible Requirements are:

- The AuC must notify the Team Lead if a target is already identified and what the analysis results are

UCA: Team Lead provides identify object command after the object is out of range (H3)

Scenario:

- The Team Lead receives accurate feedback about the objects
    - The Team Lead is pre-occupied with another task
    - The Team Lead already believes he knows what the target is and that he can accomplish future tasks without identifying the object
- The Team Lead provides an identification command late

Possible Requirements are:

- The Team Lead must receive feedback every TBD seconds until providing an identification command for objects with TBD region
- The Team Lead must always provide an identification command unless TBD scenario (emergency, ACAS maneuver, etc.) preclude him from doing so

Scenario:

- The Team Lead receives inaccurate feedback about the objects
    - The objects fluctuate between moving inside and outside of the boundary (unlikely)
    - An attacker is spoofing pretend objects onto the FMS
    - An attacker places fake objects within the region

    - The radar accepts frequencies outside the bounds of what should be considered significant objects (trees, rocks, etc.)
- There are no objects within the region or the objects are not enemies, friendlies, or civilians but rather fake obstacles

Possible Requirements are:

- The Team Lead radar must accuracy to within TBD feet and update at a rate of TBD Hz
- Objects added to the FMS must fulfill TBD conditions (frequency, crypto, visual, etc.)
- The Team Lead's FMS must be able to identify true versus false targets

UCA: Team Lead does not send "track target" command when the mission requires it (H3)

Scenario:

- The Team Lead receives accurate feedback about the target(s)
    - The Team Lead notices a set of UAV(s) begin following the target so he believes they are already tracking the target
    - The Team Lead does not believe the target needs to be tracked.
    - The Team Lead confuses the target with another object (i.e., there are five hostiles but the Team Lead is looking to track one of them and picks the wrong one)
    - The Team Lead believes the AuC can track targets without providing an explicit command
- The Team Lead does not provide a track target command

Possible Requirements are:

- The Team Lead must receive an indicator when a UAV is tracking a target that shows which UAV(s) are tracking it
- The indicator must update at a rate of TBD Hz
- The Team Lead must confirm hostile targets with MP within TBD seconds of locating them if unsure about tracking them
- The Team Lead must be able to properly identify different targets
- The Team Lead must understand the mode changes of tracking a target

Scenario:

- The FMS receives a command to track the right target
    - The algorithm determines that the task is not a priority
    - There is a corruption error and the AuC cannot implement track commands
    - The AuC process model is corrupt. It believes the environment is not set to track objects because of low visuals.
- The Autonomous Controller does not implement a track target command

Possible Requirements are:

- The Team Lead must be able to prioritize tasks

- The software must be able to reset within TBD seconds
- The Team Lead must have a plan if tracking targets is not working
- The TL/GS must be able to adjust the AuC settings for the appropriate environment

UCA: Team Lead sends track target command for the wrong target (H5)

Scenario:

- The Team Lead provides a track target for the right target
    - There is a mode setting that optimizes target tracking. The algorithm defaults to another target that is considered closer and higher priority to the UAV(s)
- The FMS receives the command for the wrong target

Possible Requirements are:

- The Team Lead must have an option to not allow changes to his command(s) by the FMS

Scenario:

- The Team Lead receives accurate feedback about the targets
    - The Team Lead does not notice a newer target that comes in after already applying a track target command.
    - The Team Lead is waiting for the UAV(s) to enter a specific position before applying the command so that certain UAV(s) will track the target
- The Team Lead provides a track target command for the wrong target

Possible Requirements are:

- New targets with TBD requirements must be alarmed to the Team Lead
- The Team Lead must understand what requirements are necessary to implement a track target command with specific aircraft

UCA: Team Lead provides the track target command incorrectly (H3)

Scenario:

- The FMS sends the appropriate command
    - The AuC does not believe the FMS has the appropriate credentials to implement the command
    - The AuC cannot interpret the command due to a recent firmware update
- The AuC does not receive the command or receives a different command

Possible Requirements are:

- The AuC must contain the appropriate credentials to communicate with the Team Lead FMS
- Any firmware updates to the AuC must ensure FMS compatability

UCA: Team Lead provides the track target command when the targets are already out of range or in a restricted area (H3, H4)

Scenario:

- The Team Lead receives the appropriate feedback about the target's location(s)
    - The Team Lead does not notice an alert from the FMS stating they are near a drop out zone that the UAV(s) can no longer reach them
- The Team Lead provides the command at the wrong time

Scenario:

- The Team Lead receives the wrong feedback about the target's locations or the UAV(s) locations
    - The differential GPS is providing an adjusted location based on prior locations because it has a weak signal
- The FMS has the wrong location for the UAV(s) or the target(s)

Possible Requirements are:

- The Team Lead must receive a UAV and target's location within TBD feet every TBD seconds
- The Team lead must be notified if the GPS is providing a corrected location instead of an exact location

UCA: Team Lead provides the track target command early and exposes the mission (H3)

Scenario:

- The Team Lead receives accurate timing and locations for the UAV(s) and target(s)
    - The Team Lead is afraid that waiting will cause another enemy to enter the field and he wants to ensure the mission is done earlier rather than later
- The Team Lead begins the mission early

Possible Requirements are:

- Create timing and location allowances for when specific commands can be implemented (only between XX and XX PM and within 2 miles, etc.)

UCA: Team Lead provides the track target command after the mission window closes (H3)

Scenario:

- The Team Lead receives accurate feedback about the timing for the mission and the locations of the targets
    - The Team Lead selects a mission option from the AuC that uses the incorrect mission start time
- The Team Lead provides the command for the wrong time

Possible Requirements are:

- The Team Lead must check the AuC mission action, time, target, etc.

UCA: Team Lead provides formation command that causes the UAV(s) to turn the wrong direction (H3)

Scenario:

- The Team Lead receives an inaccurate location for the UAV(s). Reasons the locations are inaccurate include:
    - They are operating in a GPS denied environment, so the Team Lead is unable to access their locations.
    - The Autonomous Controller calculates the wrong location. It is using accelerometers within the inertial navigation system, but there are enemy aircraft creating a magnetic field nearby the UAV(s). This causes the Autonomous Controller to calculate inaccurate position data.
- The Team Lead provides an inappropriate formation command

Possible Requirements are:

- The FMS must be able to update the Team Lead and AuC if GPS is denied and what the alternative system is doing
- The Team Lead must work with Mission Planners to develop a plan in case the UAV(s) are lost or location is unknown

Scenario:

- The AuC has an inaccurate location of the Team Lead
    - A nearby aircraft is providing a signal pretending to the Team Lead.
    - The UAV(s) are being provided with pretend Team Lead signal data
- The FMS receives the wrong location from the AuC

Possible Requirements are:

- The Team Lead shall utilize VOR or INS to verify locations in uncertain environments
- The Autonomous Controller shall verify UAV location using alternative methods (pitot tube, etc.)
- The Autonomous Controller shall ask for clarification from the Ground Station if multiple sources are provided for the Team Lead position

UCA: Team Lead provides formation command that places the UAV(s) in restricted airspace (H4)

Scenario:

- The Team Lead provides a formation command that places them in restricted airspace because the Team Lead is unaware of the airspace restriction. The Team Lead map does not identify the area as a restriction. The area could be recently changed, and local intelligence was unable to notify the team in time due to signal losses.

Possible Requirements are:

- The Team Lead and UAV(s) shall maintain a distance of TBD miles from a restricted zone [S45]
- The Mission Planners shall ask for updated restricted zones prior to takeoff [S45]

UCA: The Team Lead provides formation command before the UAV(s) re-connect lost line of sight (LOS)

Scenario:

- The Team Lead provides a formation command early because he believes line of sight connection has already been re-established. Reasons he might believe this include:
    - His interface shows a green signal strength to the UAV meaning strong. However, the interface is frozen from a previous time as it tries to update to the most recent information. The processing lag is due to the limited memory capabilities.
    - He was able to provide commands prior to this one and the UAV was able to implement them. However, the UAV receiving antenna experiences a malfunction when the Team Lead tried to send this command.

Possible Requirements are:

- The Team Lead shall see the timestamp of last known data and current information on his interface
- The UAV equipment shall be inspected prior to takeoff for any malfunctions

UCA: Team Lead provides formation command after enemies are within TBD range (H3)

Scenario:

- The Team Lead provides the command late because he thought the UAV(s) were already in the formation. Reasons for this could include:
    - His user interface showed a completed section for the formation command. He did not verify the positions of each UAV because there are hundreds of them.
    - He receives a confirmation from each UAV saying the command was implemented. However, some of the UAVs have malfunctions, decide to land, and do not notify the Team Lead that they are no longer in formation.

Possible Requirements are:

- The Team Lead shall have an algorithm on the FMS to verify UAV locations based on a completed formation command
- The UAV(s) shall notify the Team Lead if they are no longer completing a command
- The Team Lead shall have an interface detailing the position of each UAV

UCA: Team Lead does not provide fire command when the mission requires it (H3)

Scenario:

- The Team Lead does not provide the fire command because he believes the UAV(s) can automatically fire upon identifying targets. Reasons he believes this include:
    - The Team Lead was told that a new upgrade to the UAV(s) allows automatic firing upon target confirmation by the Team Lead.
    - The Team Lead assumes this system works similar to other MUM-T systems where the UAV(s) have automatic firing capabilities
- The Team Lead does not provide the command

Scenario: The Team Lead does not provide the fire command because he believes the UAV(s) do not need to fire for the mission. The UAV(s) are acting as sensor trucks to identify enemy targets because only a few enemies are expected. The Team Lead is planning on providing all firing power. However, more enemies are involved than expected.

Scenario: The Team Lead does not provide the fire command because he does not trust the UAV(s) to accurately hit targets. He believes this because he provided an early fire command test before the mission on a random target and the UAV(s) did not perform well

Possible Requirements are:

- The Team Lead shall verify and test firing and any system upgrades prior to takeoff
- The Team Lead shall receive specific training on firing capabilities for new systems
- The Team Lead shall prepare for UAV firing in any air to air combat scenario

UCA: Team Lead provides firing command for the wrong target (H5)

Scenario:

- The Team Lead provides the fire command for the wrong target because he believed the right target was selected. Reasons he might believe this include:
    - The targeting system highlighted the correct target. However, the system switched targets while the Team Lead was maneuvering, and the Team Lead did not notice the switch.
    - The Team Lead confused the targets. He lost track of the correct target while trying to maneuver and tracked onto a different nearby target.
    - The Team Lead begins feeling light headed or ill and provides the firing command.

Scenario: The Team Lead provides the fire command for the wrong target because he accidentally applied the firing command. The Team Lead was trying to lock on a target and instead authorized the UAV to fire at the target that it was tracking.

Scenario:

- The Team Lead provides the fire command for the wrong target because the Enemy Identifying Information and Friendly data are inaccurate. Reasons for this could be:

    - The UAV(s) provide inaccurate sensor data about the locations of enemies. There is an electromagnetic field around the enemies that obscures their observed locations using accelerometers.

Possible Requirements are:

- The Team Lead shall verify target selection immediately prior to providing firing command and lock on target prior to firing
- The Team Lead shall provide a lock on target once the target can be locked on
- The Team shall not provide a firing command upon feeling ill
- The Team Lead shall understand all methods that are possible for providing a firing command
- The UAV sensors shall cross check visual observations with accelerometer data

UCA: Team Lead provides firing command for the wrong UAV (incorrect weapon target pairing) (H3)

Scenario 53: The Team Lead provides the firing command for the wrong UAV because he has an incorrect understanding of the weapons on a specific UAV. Reasons for this could include:

- Scenario: There is a discrepancy between what the AuC identifies as weapons on board the UAV and what was loaded by the ground crew prior to takeoff
- Scenario: The Flight Management System shows an inaccurate number of remaining missiles for a UAV
- Scenario: The Autonomous Controller incorrectly calculates the number of remaining missiles on the UAV. Reasons for this could be:
    - There is a weight sensor that was improperly calibrated to detect if missiles were in place
    - The AuC was provided an incorrect starting value by the Ground Station

Possible Requirements are:

- The Team Lead shall verify the weapons on the UAV(s) prior to takeoff [S53.1]
- The Team Lead FMS shall utilize redundant methods for verifying weapons on a UAV [S53.2]
- The Autonomous Controller shall verify estimates for remaining weapons using multiple algorithms and check with the Ground Station
- Multiple Ground Station attendants shall verify the starting missile values

UCA: Team Lead provides fire command after the TL and/or UAV(s) are being fired at (H2)

Scenario:

- The Team Lead provides the fire command late because he does not believe the UAV(s) are being fired at. Reasons he could believe this are:
    - The UAV(s) are out of range and the Team Lead cannot determine if there are enemies near them

-

- The Autonomous Controller does not notify the Team Lead of the enemy fire. Reasons for this could include:
    - The software algorithm is busy processing requests from the Ground Station
    - The Autonomous Controller believes the UAV(s) can handle the enemy target and does not want to burden the Team Lead with extra information

Possible Requirements are:

- The UAV(s) must be within TBD range of the Team Lead at all times
- The Autonomous Controller shall stop processing other requests and immediately notify the Team Lead of enemy fire when it occurs
- The Autonomous Controller shall always notify the Team Lead of enemy fire

UCA: Team Lead provides fire command after the target is out of range (H5)

Scenario: The Team Lead provides the command late because he does not detect the targets. Reasons for this could include:

- The Team Lead radar is receiving low power and having trouble detecting targets.
- The Flight Management System is not notifying the Team Lead of possible targets. There is a software glitch that turned off the alerts.

Possible Requirements are:

- The Team Lead radar and battery shall be inspected prior to takeoff for power sufficiency
- The Ground Station shall provide redundant target alerts to the Team Lead when firing is involved

UCA: Team Lead provides fire command before the right target has been targeted (H5)

Scenario: The Team Lead provides the command early because he believes the right target is selected. Reasons for this could include:

- The FMS is showing that the target is selected by providing the red circle. However, the circle is for another target that's near the target.
- The AuC shows that it implemented the target designation command

Possible Requirements are:

- The FMS must differentiate between close targets by TBD miles on the FMS
- The AuC must provide alternate indicators for a target designation

UCA: Team Lead provides the command early before receiving authorization from the Crew Chief

- The ROE changed during the mission but wasn't communicated by the Ground Station or Mission Planners because of a drop in signal

Possible Requirements are:

- ROE changes during the mission must be communicated to the Team Lead using multiple devices (radio, transponder, etc.)

UCA: Team Lead provides fire command after a target is out of range (H3)

Scenario: The Team Lead believes the target is within range because:

- The AuC shows the target location within range even though the Team Lead receives a notification that it is out of range
- The FMS calculates an appropriate weapon target pairing

Possible Requirements are:

- The Team Lead must receive a warning within TBD miles and TBD seconds if targets are exiting the weapon range
- The FMS must update the weapon target pairing every TBD seconds

UCA: The Team Lead does not provide waypoints for his FMS (H2)

Scenario: The Team Lead is so focused on the UAV(s) locations that he forgets to update his own position

Scenario: The Team Lead misses the warning from the FMS that there are no more waypoints to follow when on autopilot

Possible Requirements are:

- The Team Lead aircraft must have a specific mode such that the Team Lead can focus on the UAV(s) when he cannot focus on his own aircraft. The aircraft could enter autopilot and stay within TBD boundaries
- The Team Lead must receive a haptic warning prior to running low on waypoints

The rest of the Team Lead UCAs have been examined in other STPAs and the scenarios are similar.

Ground Station Scenarios/Requirements

UCA: Ground Station does not provide land command when the aircraft is in a pattern and at minimum fuel (H1, H2)

Scenario: The Ground Station does not provide a land command because:

- They believe the UAV(s) are still executing the mission
- The Team Lead asked them not to
- The AuC shows they are executing a land command already
- The UAV is unable to land appropriately

Possible Requirements are:

- The Ground Station and Team Lead must share a mission time and windows in case communication breaks down
- The Ground Station must authenticate Team Lead communications
- The Team Lead must receive warning when the UAV(s) are at a minimum fuel level and need to RTB
- The FMS must update every TBD Hz so that a land command is not shown when it is not being executed
- The Ground Station must have an alternative landing method if a wheel is damaged

UCA: Ground Station does not provide land command when the aircraft is at the airfield and other aircraft are trying to enter a pattern (H1)

Scenario: The Ground Station does not provide the land command because they do not believe that other aircraft are trying to enter a pattern.

Scenario: The Ground Station does not provide the land command because they do not believe they are at the airfield.

Possible Requirements are:

- The Ground Station must receive feedback on aircraft locations within TBD miles
- The Ground Station must receive UAV locations to within TBD feet
- The Ground Station must be notified when the UAV(s) are near the runway

UCA: Ground Station does not provide land command when the aircraft is unstable (H2)

Scenario: The Ground Station does not provide the land command because they believe the aircraft is stable and does not require an emergency landing.

- The Ground Station does not receive a warning when the UAV is about to stall
- The Ground Station provides the command but the UAV interprets the command to land later instead of immediately since it detects an unclear area beneath it
- The AuC detects unsafe terrain and decides not to implement the command

Possible Requirements are:

- The machine learning algorithm can warn the Ground Station of unsafe terrain but must implement the land command
- The Ground Station stall warning must occur within TBD seconds of UAV stall
- The FMS must differentiate between prospective and immediate commands from the Ground Station (implement now vs. Later)

UCA: Ground Station provides land command when the runway is not clear (H1)

Scenario: The Ground Station provides the land command because they believe the runway is clear. Reasons for this could include:

- ATC provided clearance that the runway is clear and there is dense fog so the pilots do not have a visual of the runway
- The Ground Station does not have time to check that the runway is clear and makes an assumption based on best judgment

Possible requirements are:

- The Ground Station must verify runway clearance before landing

UCA: Ground Station provides land command when the aircraft is above unstable terrain (H1)

Scenario: The Ground Station provides the land command because they believe the aircraft is above stable terrain. Reasons for this could include:

- The geographic terrain data shows the area is at level elevation, but a recent meteorological development changed the terrain.

Possible requirements are:
- The geographic terrain must be updated every TBD days
- The mission planners must provide alternative landing areas along each UAV(s) route

UCA: Ground Station provides land command when the aircraft is above a restricted area (H4)

Scenario: The Ground Station provides the land command because they believe the aircraft is above an unrestricted area. Reasons for this could include:

- The terrain data inserted on the FMS is inaccurate
- The FMS calculates the land area incorrectly based on the inputs provided by the Mission Planners

Possible Requirements are:
- The Ground Station must use a visual to verify landing zones
- The AuC must re-calculate restrictions based on updates throughout the mission

UCA: Ground Station provides land command before the UAV completes the airfield arrival procedure (H1)

Scenario: The Ground Station provides the land command because they believe the UAV already completed the airfield arrival procedure. Reasons for this include:

- The AuC provided feedback stating that it was complete
- A similar aircraft looks to have completed the procedure

Possible Requirements are:

- The AuC must provide statuses for task implementation:
    - Started, implementing, complete, paused, etc.
- The aircraft must be distinguishable from other similar aircraft

UCA: Ground Station provides land command after the UAV is already in restricted airspace (H4)

Scenario: The Ground Station provides the command late because they believe the UAV is not near restricted airspace. Reasons for this could include:

- The Ground Station has an improper location for restricted areas
- The restricted area locations changed during the mission (unlikely)
- The AuC cannot identify the restricted locations provided from the Mission Planners

Possible Requirements are:

- The Ground Station must verify restricted areas with the joint force commander
- Restricted area changes must be provided to all team members within TBD seconds
- The AuC must be able to accept coordinate locations for restricted zones from the Ground Station

UCA: Ground Station provides land command after the UAV is below the minimum fuel level to safely land (H1)

Scenario: Ground Station provides the land command late because they do not believe the aircraft is almost out of fuel. Reasons for this could include:

- The capacitance probe for the fuel gauge is not grounded prior to taking the measurement. This results in a higher fuel reading than what is truly in the tank.
- The Ground Station has a fuel reading for a different UAV. They mix up which UAV they are tracking and are using the fuel reading for a UAV on the ground instead of the one in the air.

Possible Requirements are:

- The fuel capacitance probe shall be checked against the dip check to verify fuel levels prior to takeoff
- The Ground Station shall verify fuel levels of the UAV against what the maintenance crew measure from the capacitance probe and the dip stick

UCA: Ground Station does not provide takeoff command when the mission starts (H3)

Scenario: The Ground Station does not provide the takeoff command because they do not believe the mission is supposed to start yet. Reasons for this could include:

- The Ground Station did not receive the go ahead from the Crew Chief to begin.
    - There is signal jamming occurring near the runway.
    - The Crew Chief is on the wrong radio frequency.
    - The Ground Station has a broken wire in their radio.
    - The Ground Station is flooded with communications and cannot differentiate the General's go ahead from the other comms

Possible Requirements are:

- The radio signal path from the Crew Chief to the Ground Station shall be protected from signal jammers
- The Crew Chief and Ground Station shall verify their communication frequencies in person prior to using the radios []
- The Maintainers shall verify the conductance values for the radio and visually inspect the radios prior during the mission pre-check []
- The Ground Station shall be able to handle TBD communications in event of a DoS attack
- The Ground Station shall communicate using a separate radio if

UCA: Ground Station provides takeoff command when the runway and/or airspace is not clear (H1)

Scenario: The Ground Station provides the takeoff command because they believe the runway and airspace are clear. Reasons for this could include:

- The Ground Station received a clearance from ATC. There is no visual due to dense fog.
- The Ground Station does not have any aircraft on their local radar.

Possible Requirements are:

- The Ground Station shall verify the runway clearance using vehicles to drive up and down the runway and verify it is clear []
- The Ground Station shall verify the airspace clearance using local radar []

UCA: Ground Station provides takeoff command when the UAV is not on the runway

Scenario: The Ground Station provides the takeoff command because they believe the UAV is on the runway. Reasons for this could include:

- The Ground Station receives an inaccurate GPS location of the UAV.
- Someone pretending to be the Runway Manager provides false data that the UAV is on the runway
- The Ground Station is analyzing the wrong UAV's location.

Possible Requirements are:

- The Ground Station must verify UAV location using other UAVs' coordinates
- The Runway Manager must be differentiated from false communications
- The Ground Station must differentiate between each UAV

UCA: Ground Station provides takeoff command before the UAV completes the pre-flight checklist (H1, H2)

Scenario: The Ground Station provides takeoff command because they believe the UAV completed the pre-flight checklist. Reasons for this could include:

- The UAV is at the final step of the checklist
- The Team Lead notifies all members that the mission is ready to start

Possible Requirements are:

- The Team Lead must not start the mission without prior commitment from the ground crew that the UAV is ready
- The Ground Station must not takeoff until the ground crew and AuC confirm takeoff can proceed

UCA: Ground Station provides takeoff command before the UAV is disconnected from all ground equipment (H1, H2)

Scenario: The Ground Station provides the takeoff command because they believe the UAV is disconnected from any ground equipment. Reasons for this could include:

- The ground crew did not see any equipment attached
- The UAV showed that all equipment has been disconnected

Possible requirements are:

- The ground crew must use electrical readers to verify all wires and pumps have been removed from the aircraft

UCA: Ground Station provides takeoff command before ground personnel clear the area (H1)

Scenario: The Ground Station provides the takeoff command early because they believe there are not ground personnel near the UAV.

UCA: Ground Station does not provide altitude command when the waypoints are updated (H1, H2, H6)

Scenario: The Ground Station does not provide an altitude command because they believe the waypoints have not been updated. Reasons for this could include:

- The AuC shows the same waypoints that it heading for
- The FMS shows the same route
- The UAV has the same attitude even when the waypoints are updated

Possible Requirements are:

- The Ground Station must verify the waypoint update using FMS and attitude data
- The Ground Station must be able to directly update the AuC data if the FMS cannot provide updated waypoints

UCA: Ground Station provides an altitude command that is below the minimum obstacle clearance (H1)

Scenario: The Ground Station provides an altitude command below clearance because they believe the UAV is above the clearance. Reasons for this could include:

- The FMS shows that the UAV is decreasing in altitude
- The UAV's locations is switched with another UAV on the Ground Station data

Possible Requirements are:

- The Ground Station location data must be verified for each aircraft prior to takeoff
- The AuC must show task implementation throughout an entire maneuver

UCA: Ground Station provides an altitude command that conflicts with another aircraft's altitude (H1)

Scenario: The Ground Station does not believe another aircraft is at that altitude because:

- The TCAS does not provide any alerts
- There are no other aircraft on their radar
- The Ground Station sent out a signal and did not receive any responses
- The ground radar did not pick up any aircraft

Possible requirements are:

- The Ground Station must verify all potential aircraft collision sources

Autonomous Controller Scenarios/Requirements

Control Action: Power on payload

UCA: Autonomous Controller does not power on the payload when the TL/GS provides a command that requires the payload (H5)

Scenario:

- The AuC provides a command to the MFC to power on the payload
    - There is not enough power to distribute to the MFC due to limited battery resources
- The MFC does not provide power to the payload

Possible Requirements are:

- Upon component upgrade and/or modification, develop an updated power distribution need

Scenario:

- The AuC receives accurate feedback about the components that need to be powered on based on the TL/GS command
    - The AuC uses an old algorithm that defaults to a specific payload
    - The AuC machine learning algorithm determines that the camera is insufficient to complete the task and does not power it on as a result
    - The AuC powers on a payload used for a different command
- The AuC does not provide a command that powers on the appropriate payload

Possible Requirements are:

- If a payload is deemed insufficient, the AuC must notify the TL/GS and encourage them to select the payload manually for power on and utilization
- The TL/GS must be notified of the payload in use by the AuC to confirm the right equipment is utilized
- The TL/GS must provide a power off for any payload that is powered on unnecessarily

Scenario:

- The AuC receives inaccurate feedback about the targets involved in the command
    - The Team Lead provided the wrong target for the search command which requires a separate payload
    - The Team Lead did not provide a target so the AuC defaulted to the nearest target for the command
- The AuC powers on the wrong payload

Possible Requirements are:

- The AuC must notify the TL/GS of the target if no target is given for the command
- The AuC must notify the TL/GS if a payload is unable to successfully complete a command and must provide alternatives to the TL/GS

UCA: Autonomous Controller provides a power on payload command when the Ground Station or Team Lead provide a conserve power command (H5)

Scenario:

- The AuC receives the conserve power command
    - The AuC receives the power on command first and the conserve power command is not implemented until the prior task is complete
- The AuC provides a power on command to the payload

Possible Requirements are:

- The AuC must be configured to analyze and implement TBD number of tasks simultaneously with TBD processing power
- A conserve power command must trump any other commands requiring TBD payload power

Scenario:

- The AuC receives inaccurate feedback about the targets
    - The targets are fuzzy in the sensor readings and look like an object that they are not (e.g., confusing an MH-17 with a C-130). The AuC identifies the targets as higher priority than conserving power
- The AuC powers on the payload when it should be conserving power

Possible Requirements are:

- The TL/GS must be notified of and confirm the target's priority

UCA: Autonomous Controller powers on payload before the conserve power option is no longer implemented

Scenario:

- The AuC receives the time designation for the conserve power command
    - The AuC miscalculates the time by assuming the command is supposed to last for 15 minutes instead of lasting until 15:00 PM.
- The AuC turns on the payload early

Possible Requirements are:

- The AuC must not move on from a conserve power command without authorization from the TL/GS
- The AuC must be configured to accept time or timer values for all commands

Scenario:

- The AuC receives inaccurate feedback about the conserve power command
    - The AuC receives power digression from the payload and determines that the payload is not turned on because there is not enough power going to it.
- The AuC initiates a power on command

Possible Requirements are:

- The AuC must use an additional total power requirement to evaluate any discrepancies in power generation and diversion to and from components

Throttle and attitude commands are evaluated in Sarah Summers thesis between the Vehicle Management System for UAV(s) and the UAV

Control Action: Implement Command

UCA: Autonomous Controller does not implement a task from the Team Lead (H3)

Scenario:

- The Autonomous Controller sends a command to the hydraulics to actuate based on the given task
    - There is a broken wire to the hydraulics
    - The task exceeds a limit for the hydraulic system
- The hydraulics do not actuate

Possible Requirements are:

- Any task implementation must be within the physical limits of the aircraft
- The system connections must be inspected prior to takeoff

Scenario:

- The Autonomous Controller has the appropriate feedback from the UAV regarding its current attitude, altitude, velocity, etc.
    - The Autonomous Controller calculates that the current state is sufficient to accomplish the task
    - The Autonomous Controller algorithm calculates that any deviation from the current state will cause a disruption to the current mission
    - The AuC uses a machine learning algorithm that dynamically changes which UAV should implement the task too often such that it cannot select a UAV to perform the task
- The Autonomous Controller does not implement the task from the Team Lead

Possible Requirements are:

- The AuC must ask for clarification if a command results in the same attitude, velocity, altitude, etc.
- The AuC must warn the TL/GS if a command results in a dangerous maneuver for the UAV
- The AuC must have a time limit on the algorithm to limit interchange between UAV selection for task implementation. Alternatively, once the algorithm reaches the time limit, it must ask for TL/GS input to select an aircraft if necessary with the reasoning behind why it had trouble selecting (e.g., one is better for distance but the other has updated missiles)

Scenario:

- The Autonomous Controller has incorrect feedback about the UAV's attitude, altitude, velocity, etc.
    - The FMS is sending data delayed so the AuC is calculating states behind where the UAV is actually located or its current speed
    - The velocity data provided from the sensor reading is inaccurate due to extreme windspeeds
- The Autonomous Controller does not implement the task from the Team Lead

Possible Requirements are:

- Any delays in sending data from TBD Hz must be communicated to the AuC so the controller can properly adjust its calculations using current data
- The GS must monitor and provide corrected data if the AuC is using inaccurate estimates from the sensors/FMS

UCA: Autonomous Controller implements a task using the wrong UAV (H3)

Scenario:

- The AuC sends the task implementation to the correct UAV
    - The UAV is unable to receive commands so the command is diverted to another UAV
    - The UAV cannot implement the task because it is occupied so it diverts the task to another UAV
    - The AuC calculates that it is more optimal for another UAV to implement the task so it diverts the task to the other UAV
    - The
- The wrong UAV receives the command

Possible Requirements are:

- Notify the TL/GS if a UAV diverts a command to another UAV
- The AuC must not divert tasks to other UAV(s) without notifying the TL/GS and asking for permission to do so for certain tasks (target designation, fire at target, etc.)

Scenario:

- The AuC receives the correct feedback about the current positions, weapons, states of the mission UAV(s)
    - The AuC algorithm miscalculates which UAV is optimal for performing a maneuver. Reasons for this could include:
        - The algorithm does not account for environmental factors impacting the UAV's sensors
        - The machine learning algorithm is unable to identify an object within the UAV's window and decides to avoid using the UAV altogether so that it can focus on avoiding the object
    - The AuC sends the task to the wrong UAV

Possible Requirements are:

- The TL/GS must confirm the task allocations if specific aircraft are required to perform a task
- The algorithm must account for degradations in the environment

UCA: Autonomous Controller implements a task when there is no command from an authorized command provider (H3)

Scenario:

- The Autonomous Controller does not receive a command from an authorized provider
    - The MFC uses machine learning and notices the UAV is following a target. The MFC decides to implement a search for target command although the AuC is trying to track another target
- The MFC begins implementing a task that was not provided

Possible Requirements are:

- The MFC must provide task suggestions to the GS/TL before implementation

Scenario:

- The AuC receives the correct feedback from all authorized command providers and there is no task or command coming from them
    - An outside source provides a command with the proper authorization
    - The AuC asks the TL/GS for input on implementing a command and receives no authorization. However, the algorithm is allowed to provide a root authorization if the original command does not involve firing a weapon or target designations.
    - The AuC determines an additional task is necessary to complete a command the TL provides the authorization without knowing why the task is necessary
- The AuC implements a task

Possible Requirements are:

- The GS must provide immediate assistance on unknown tasks suggested by the AuC
- The AuC must ask for assistance in tasks/commands coming from previously unknown sources
- The AuC must not implement additional commands or tasks without authorization from the GS/TL

Scenario:

- The AuC receives incorrect feedback from the MFC that there is a task that needs to be implemented to complete the mission
    - The MFC miscalculates that additional weapons are needed to destroy a target. The machine learning algorithm is unable to confirm that the target was properly destroyed.

    - The MFC algorithm calculates that the aircraft needs to search for an additional missing target. The target was no longer necessary to complete the mission
- The task is unnecessary to completing the mission

Possible Requirements are:
- The TL/GS must clear out unnecessary targets from the FMS to prevent the UAV(s) from performing unnecessary tasks
- The MFC must ask for clarification if it calculates results different from what is expected in executing a task or command

Scenario:
- The MFC does not provide an actuation to the hydraulics/throttle settings
    - An electrical or physical impulse comes from a nearby source that causes the control surfaces to change (e.g., EMP)
- The hydraulics or throttle change their settings

Possible Requirements are:
- The connections from the MFC to the hydraulics, and from the hydraulics to the control surfaces must be EMP hardened and limit vibrations from outside sources

UCA: Autonomous Controller implements a task before completing an algorithm to determine the optimal task completion (H3)

Scenario:
- The AuC implements an algorithm to determine the task allocation
    - The algorithm took too long to derive a solution and defaulted to a UAV
    - The algorithm stopped running and auto-selected the default UAV
    - The algorithm setup includes a default UAV option without the TL/GS knowledge
    - The algorithm uses machine learning which determines that there are too many targets to complete the algorithm
- The MFC receives the command before the algorithm completes

Possible Requirements are:
- The TL/GS must be prompted to select a default UAV before running the algorithm
- The algorithm must notify the TL/GS if it becomes stopped or when it selects the default and for what reason
- The algorithm must not provide results to the TL/GS without completely running through the whole algorithm

Scenario:
- The AuC receives accurate feedback about the UAV(s) and their locations, and the command from the TL/GS
    - The AuC determines that it has a direct command from the TL/GS instead of requiring the algorithm

- The AuC implements the task without completing the algorithm

Possible Requirements are:
- The TL/GS can select a specific algorithm or option for the AuC to run when providing the task or command (e.g., optimize for TBD, use this type of aircraft, etc.)
- The TL/GS can select whether the command should be directly applied, and if not, which parameters must be optimized (aircraft, fuel, weapon, etc.)

Scenario:
- The AuC receives inaccurate feedback about the task or command, or states of the UAV(s)
    - The UAV(s) are shown to be in the same position so the algorithm is not required for the given task
    - The MFC is unable to provide the AuC with its current task execution so the AuC determines that the UAV is available to receive a task
- It requires the algorithm to calculate certain parameters

Possible Requirements are:
- The AuC must notify the TL/GS if it is unable to retrieve a UAV's current task execution, altitude, attitude, etc so that the UAV can be removed from the available list to execute tasks
- The GS must be able to provide the AuC with corrected data if specific UAV(s) are required to execute the mission

UCA: Autonomous Controller implements a task after the target is no longer necessary or is out of range (H3)

Scenario:
- The AuC sends the task to the MFC
    - The MFC processor is slowed due to high temperatures and consistent operation
- The MFC does not implement the task until after the target is not needed

Possible Requirements are:
- The MFC must maintain an operating speed of TBD Hz in TBD temperatures for TBD time period

Scenario:
- The AuC receives accurate feedback about the target and its current range
    - The machine learning algorithm determines that the target will still be in range for TBD seconds so the AuC can wait to implement the task and focus on other commands. However, the aircraft exceeded the calculated range.
    - The AuC miscalculates the target's priority.
    - The target is no longer needed but the AuC did not remove the task implementation
- The AuC does not implement task until the UAV is out of range or not needed

Possible Requirements are:

- The AuC must account for the maximum speed, fuel range, and distance coverage of an aircraft when using predictive algorithms to determine an aircraft's future location
- Target priorities must be set by the TL/GS prior to executing commands
- When targets are no longer required, the AuC must prioritize the task above all other tasks other than those requiring target designation or firing at targets unless the target is involved with one of those commands

Scenario:

- The AuC receives inaccurate feedback about the UAV(s) or the target(s)
    - The predictive indicator for the target's location calculates a future position based on older data due to a lapse in the update rate
    - The target is confused with another object that it crosses near because they have a similar speed, size, shape, etc.
- The targets are no longer necessary or in different locations

Possible Requirements are:

- A predictive indicator must account for location updates every TBD seconds
- Objects must be distinguishable physically and within the FMS

UCA: Autonomous Controller implements a task after a UAV cannot complete the task (H3)

Scenario:

- The AuC has the appropriate feedback about the task and its completion time within the mission window
    - The AuC algorithm calculates that other tasks within its system have a higher priority
    - The AuC uses a machine learning algorithm that determines the target for the task is not necessary to complete its objective(s). The TL/GS send the command over and over until the AuC accepts the command
- The AuC does not send the task to the MFC until it is too late

Possible Requirements are:

- The AuC must ask for clarification from the TL/GS if it is unable to determine a solution to implementing tasks within the given mission window
- The AuC must not eliminate tasks from its current execution list
- The TL/GS must maintain an accurate record of the tasks sent to the AuC and receive warnings if a task is reaching its upper time limit such that it needs to be executed soon to accomplish the mission
- The TL/GS must receive a status of command implementation from when it is sent to when the task is executed by the AuC

Control Action: Release weapon

UCA: Autonomous Controller does not release weapon when the Team Lead commands it (H3)

Scenario:

- The AuC provides the command to the MFC
    - The MFC cannot understand the command because it does not detect the weapon necessary to complete it on the aircraft
    - The new MFC software recognizes the command but is unable to implement it due to updated authorizations
- The MFC does not receive the command

Possible Requirements are:

- The AuC must confirm task execution with the aircraft
- The AuC most notify the TL/GS if a task cannot be executed
- The AuC software must be tested with the MFC and any weapons the UAv may carry

Scenario:

- The AuC has the right feedback about the Team Lead command
    - The AuC algorithm determines the target the Team Lead is locked on is a friendly or civilian target
    - The AuC uses machine learning to determine that the command's authorization came from a location that the TL/GS is not supposed to be in.
    - The AuC algorithm cannot accept the command without a prerequisite on the aircraft (master arm switch on, specific sensor enabled, etc.)
- The AuC does not implement the command

Possible Requirements are:

- The TL/GS must be able to authorize commands for the AuC using alternative methods if location, code, etc is not working
- The AuC must ask for additional authorization if the TL/GS lock on a friendly or civilian target
- The TL/GS must ensure that all UAV components are enabled prior to takeoff or be able to adjust them during the mission

Scenario:

- The AuC has inaccurate feedback about the Team Lead command
    - There is a code-switching error that causes the AuC to switch the command due to if-then errors
- The Team Lead did not provide a command or it was a different command than what the AuC identifies

Possible Requirements are:

- The GS/TL must be able to provide direct commands to the hardware components if the AuC is unable to properly implement commands

Scenario:

- The MFC has the right command
    - The MFC cannot actuate the hardware because it is damaged
- The UAV does something different that does not implement the task

Possible Requirements are:

- The hardware connections to the MFC must be inspected prior to takeoff

UCA: Autonomous Controller releases a missile for the wrong target (H5)

Scenario:

- The AuC receives the right feedback about the target
    - The AuC algorithm determines another target is of higher value
    - The AuC cannot identify the target designation for the target
    - The AuC interprets the command as a location
- The AuC releases the missile for another target

Scenario:

- The AuC receives inaccurate feedback about the target
    - The MFC received frequency data indicating that the target returned a friendly IFF response
    - The MFC
- The target is actually a hostile with a target designation

UCA: Autonomous Controller releases a missile before receiving a command from the Team Lead (H5)

Scenario:

- The AuC does not send a command to the MFC
    - The MFC algorithm notices the command in the queue and executes the command
- The MFC implements the command

Possible Requirements are:

- The MFC must not execute commands from the AuC until the AuC appropriately applies the command execution

Scenario:

- The AuC receives accurate feedback about the command from the Team Lead
    - The AuC algorithm determines that a target can be fired at without a Team Lead command
    - The AuC receives a target designation that it determines can lead to releasing a missile as long as parameters are satisfied
- The AuC decides to release the weapon without a Team Lead command

Possible Requirements are:

- The AuC must not provide a weapons release without explicit command from the TL/GS (no assumptions or parameters can authorize it)

Scenario:

- The MFC does not open the weapons bay
    - There is an electrical inference that causes the bay to unlock
    - A missile is loose in the bay and pushed through the bay doors
- The weapons bay releases the missile

Possible Requirements are:

- The weapons storage area must be secured prior to takeoff

UCA: Autonomous Controller releases a missile after the target is out of range (H3, H5)

Scenario:

- The AuC receives accurate feedback about the target and its location
    - The AuC calculates using the incorrect missile parameters
    - The AuC algorithm does not account for delays in releasing the weapon or unlocking the weapons bay
- The AuC does not implement the command until it is too late

Possible Requirements are:

- The AuC must be checked prior to takeoff for accurate weapon parameters for each weapon on board
- The AuC algorithm must account for delays in opening the weapons bay, sending the command, etc.

Scenario:

- The AuC receives inaccurate feedback about the target, UAV, or weapon capability
    - The UAV cannot determine the target's exact location due to signal losses in the GPS radar
    - The AuC is notified that a different missile in the release position and switches to a separate weapon
- The target and UAV distance or weapon capability is inaccurate

Possible Requirements are:

- The radar must provide an estimate of the error in the GPS location to allow the TL/GS to make an informed decision on whether to fire
- The TL/GS must notify the AuC if a weapon can be released in case the AuC receives improper feedback about the weapon

Mission Planners Scenarios and Requirements

UCA: Mission Planners do not issue the mission plan when the plan is available from the Air Operations Center (H3)

Scenario: The Mission Planners do not issue the plan because they do not believe that a plan is available yet from the Air Operations Center (AOC). Reasons for this could include:

- The Air Operations Center did not send the plan to the Mission Planners.
    - Maintenance performs a power reset which corrupts the data file
    - The Mission Planners are experiencing a Denial of Service attack. There are numerous false plans that are sent to the MP servers which slows down the servers and prevents the MPs from selecting the appropriate plan.

Possible Requirements are:

- The MP must be able to issue a plan TBD hours in advance in case redundant measures are necessary to send it

Scenario: The MP believe the plan has flaws and decide not to provide it.

UCA: Mission Planners do not provide mission updates when the mission is updated by the Joint Force Commander (H3)

Scenario: Mission Planners do not provide mission updates because they do not believe the mission has been updated. Reasons for this could include:

- There is a shift change between the mission planners. The planners are supposed to follow the T.O. for the shift change and update the new planners on any necessary information. The old planners do not provide the mission updates to the new planners because they do not follow the T.O. to know what information needs to be provided.

Possible Requirements are:

- The Mission Planners shall follow the T.O. procedures during a shift change to provide necessary updates

UCA: Mission Planners issue a mission plan that has flaws in it (improper weapon target pairing, etc.) (H3)

Scenario: Mission Planners issue a plan with an incorrect weapon target pairing because they believe the weapon to target pair is appropriate. Reasons for this could include:

- The Mission Planners have the wrong potential targets in mind.

UCA: Mission Planners issue a plan after the TL and UAV(s) do not have enough time to accomplish it (H3)

Scenario:

- The MP receive accurate feedback about a plan that needs to be executed from the Air Operations Center
    - The MP only issue the plan to the TL, but the GS has assumed control of the UAV(s) due to an issue with the TL communication protocol
- The MP do not issue the plan to the TL/GS

Possible Requirements are:

- The MP must issue plans to both the TL and GS at all times

Scenario:

- The MP receive inaccurate feedback about the plan coming from the AOC
    - There is a miscommunication from the AOC about when the plan needs to be executed
- The plan is supposed to be issues earlier rather than later

Possible Requirements are:

- The MP must confirm mission times with the AOC

UCA: Mission planner does not provide updates when the mission gets updated (H3)

Scenario:

- The MP receive accurate feedback about the need to provide a mission update to the TL/GS
    - The MP do not want to overwhelm the TL because he is near or in the middle of combat
- The MP do not provide the update

Possible Requirements are:

- The ROE must specify when the TL must be available to receive mission updates

Scenario:

- The MP provide the updated targets to the TL FMS
    - The FMS cannot recognize the target's identification from the local radar
- The FMS does not show the new targets on the screen to the TL

Possible Requirements are:

- The MP confirm that the TL and AuC can identify the targets on local radar before providing an updated plan

UCA: Mission planners update the mission by providing an improper plan that is impossible to accomplish or is fatal, confusing the TL and UAVs (H3)

Scenario:

- The MP provide a plan that is reasonable
    - The CC loads an old plan onto the aircraft.
- The FMS receives a target that is another location

Possible Requirements are:

- The FMS and MFC must be able to confirm the targets are within range prior to beginning the mission and notify the maintainers or pilot if there are errors

Scenario:

- The MP receive inaccurate feedback about the plan that is provided to the TL/GS
    - The MP TL says something that is different than what the MP intended so they provide a change to the mission based on the statement
- The MP issue a new mission that is unachievable

Possible Requirements are:

- The MP must be able to confirm the TL/GS statement or re-affirm the statement from the source

UCA: Mission planners update the mission too late after the TL or UAV already performed an improper action (H3)

Scenario:

- The MP believe they sent the mission update to the TL based on the updated feedback on the FMS system
    - The TL FMS is delayed and does not update with the new targets
    - The FMS displays the new targets but does not target. However, the MP see the new targets and believe the TL will target them
- The TL does not receive the mission update

Possible Requirements are:

- The MP must verbally confirm updates to the mission in addition to changing the FMS directly
- The TL must confirm new targets with the MP to ensure the update is coming from an authenticated source

UCA: Mission planners update the mission too early before the approval to update was given by a General (H3)

Scenario:

- The Mission Planners believe the General has given approval to update the mission
    - The approval was provided by a source pretending to the General
- The MP provide an unauthorized mission update

Possible Requirements are:

- The MP must be able to authenticate the source of the General as opposed to using a chain of command and assuming that the chain received the answer from the General

Crew Chief Scenarios/Requirements

UCA: Crew Chief does not inspect aircraft prior to takeoff (H2)

Scenario:

- The Crew Chief believes that he already inspected the aircraft
    - This could occur if the Crew Chief does not designate the aircraft by their tail number and decides that since the aircraft is in the same location, then it must be the same aircraft. However, the aircraft was switched for another aircraft and no one notified the Crew Chief.
- The Crew Chief did not inspect the aircraft

Possible Requirements are:

- Any change in aircraft must be notified to the Crew Chief immediately so he/she can re-inspect the aircraft

UCA: Crew Chief inspects the aircraft poorly (H2)

Scenario:

- The Crew Chief inspected the aircraft prior to an earlier mission and decides to run through the check list quickly.
    - There is a leaky fuel valve that was not tightened after re-fueling the aircraft
- The Crew Chief performs a poor inspection

Possible Requirements are:

- The Crew Chief must perform a full inspection prior to each flight of the aircraft

UCA: Crew Chief inspects the aircraft prior to maintenance being completed (H2)

Scenario:

- The Crew Chief is not notified of maintenance that needs to be performed on the aircraft
- The Crew Chief performs the inspection prior to the maintenance

Possible Requirements are:

- All maintainers must notify the Crew Chief when maintenance is performed
- The Crew Chief must keep a log that maintenance can update when changes are done

UCA: Crew Chief does not provide resources to the Ground Station to maintain the aircraft (H2)

Scenario:

- The maintenance team breaks a tool that is necessary to fix the aircraft
    - The Crew Chief was unaware of the shortage of the tool and did not order more tools
- The maintenance team does not have the proper resources to fix the aircraft

Possible Requirements are:

- The maintenance team must immediately notify the Crew Chief of required parts and number the parts every TBD days in case more orders are required
- The Crew Chief must be aware of the resources the maintenance team has and order new parts every TBD days depending on the part

UCA: Crew Chief provides the wrong resources to the maintenance crew (H2)

Scenario:

- The maintenance team runs out of a particular screw for the jet engine
    - The Crew Chief orders the wrong part for a different type of engine
- The maintenance team receives the wrong resources the next time the engine requires new parts

Possible Requirements are:

- The Crew Chief must check with the maintenance team, pilots, and Combatant Command to ensure he/she is ordering the appropriate parts for the aircraft assuming a certain type of engine, sensors, etc.

UCA: Crew Chief provides resources after the mission window has passed (H3)

Scenario:

- The maintenance team is waiting for a shipment to come in that has new parts
    - The Crew Chief receives the shipment but is unaware that the maintenance team requires the shipment to proceed. The Crew Chief believes they already have the parts because he did not receive any communications from the team.
- The maintenance team does not receive the resources in time

Possible Requirements are:

- The Crew Chief must regularly (every TBD minutes) check in with the maintenance team to ensure the aircraft changes are successful

ATC Scenarios/Requirements

Some ATC UCAs are not considered due to redundancy in the scenario and requirement details

UCA: ATC does not coordinate aircraft when aircraft are within range (H1)

Scenario:

- ATC receives the proper feedback on the locations and communication protocols for the aircraft
    - There is a handoff of ATC controllers and the new controller believes the previous controller provided the proper instructions to the aircraft
    - The controller believes another controller is providing support to the aircraft
- ATC does not provide coordination for the aircraft

Possible Requirements are:

- ATC ensure that the handoff controller confirms the aircraft has the appropriate instructions
- ATC must confirm with other controllers who is providing support to an aircraft

Scenario:

- ATC receives inaccurate feedback about the locations or communication protocols for the aircraft.
    - A false aircraft radar signal pretending to be one of the aircraft alerts ATC with an inaccurate GPS location.
    - False calls are reported to air traffic control about multiple aircraft in the airspace of one of the aircraft
- The aircraft are in different locations or on different radio frequencies

Possible Requirements are:

- ATC must be able to distinguish between true and false aircraft reports based on the origin of the signal, the identity of the person, etc.

UCA: ATC provides inappropriate coordination to the aircraft (H1)

Scenario:

- ATC does not provide instructions when the aircraft are on a collision course. This could occur if:
    - ATC does not receive a warning from their local modelling system
    - ATC's system has the incorrect heading or velocity for the aircraft

Possible Requirements are:

- ATC must confirm an aircraft's heading, velocity, etc. every TBD minutes to ensure the system is working appropriately
- ATC must track the aircraft in case they violate the minimum separation and not assume the system will warn them when to do so

UCA: ATC provides coordination after the aircraft are beyond minimum separation requirements (H1)

Scenario: ATC provides coordination after the aircraft are beyond minimum separation because ATC does not believe the aircraft are near the minimum separation requirement. Reasons for this could include:

- The GPS location of each aircraft on the ATC interface is incorrect.

Possible Requirements are:

- ATC must use a redundant GPS system to verify the location of the aircraft

UCA: ATC does not grant clearance for the UAV to takeoff when the UAV is ready, and the runway and airspace is clear (H5)

Scenario: ATC does not grant clearance because they are waiting to hear back from the Ground Station that the UAV is ready.

Scenario: ATC does not grant clearance because they believe that the runway or airspace is not clear. Reasons for this could include:

- There is a delay between an aircraft's FMS and the control tower. ATC is receiving the positions after an aircraft is passed that location.

Possible Requirements are:

- ATC must receive updated aircraft locations at a rate of TBD Hz
- ATC must use a model to calculate potential delays in communication and the current delay based on power losses

UCA: ATC issues ground instructions to the wrong aircraft (H5)

Scenario:

- ATC communicates with the wrong transponder but does not receive any communication back and does not have another transponder code for the aircraft

Possible Requirements are:

- ATC must be able to communicate with an aircraft without having the appropriate transponder code

UCA: ATC issues ground instructions for the wrong runway or taxi path (H1)

Scenario:
- An incident occurs on the runway just before the aircraft slows down

Possible Requirements are:
- ATC must ensure that all objects on the visual radar are TBD feet away from the runway considering their potential velocity and heading

UCA: ATC does not issue approach instructions when the UAV needs to land (H5)

Scenario:
- The aircraft is low on fuel and asks for immediate landing instructions
    - ATC cannot find a suitable alternative in time
    - ATC is occupied with another close minimum separation and cannot handle the low fuel task

Possible Requirements are:
- ATC must have a pre-approved list of emergency landing sites near the airport
- ATC must have at least TBD controllers considering a minimum number of TBD emergency cases simultaneously

UCA: ATC does not issue departure instructions when the UAV needs to takeoff (H3)

Scenario: ATC does not issue departure instructions because they believe the UAV is not ready for takeoff. This could occur if:

- ATC does not receive notification that the UAV is ready
- ATC cannot find the UAV on their radar

Possible Requirements are:
- ATC must be able to find the UAV on their radar to provide departure instructions
- ATC must receive a communication that the UAV is prepared for takeoff

Scenario: They believe the UAV is not on the runway. This could occur if:
- ATC shows the UAV in a different location

Possible Requirements are:
- ATC must be able to verify the position of the aircraft in any discrepancy

UCA: ATC issues departure instructions incorrectly (H1)

Scenario: ATC issues the instructions incorrectly because they are analyzing the wrong aircraft

This could occur if:

- Another aircraft uses the same tail number and a similar aircraft, and pretends to be the aircraft. This causes ATC to issue different departure instructions for the actual aircraft.

Possible Requirements are:

- ATC must ask for identifying information in case a pilot is pretending to control an aircraft that he is not in control of.

Scenario: ATC issues the instructions incorrectly because they have an inaccurate mental model of the runway.

This could occur if:

- The runway is recently updated but ATC does not have a new map showing the change

Possible Requirements are:

- ATC must receive updated maps any time a change is made to the runway numbers, direction, etc.

UCA: ATC issues departure instructions before the runway is clear (H1)

Scenario:

- ATC provides instructions when the runway is clear
    - ATC provides a land command for the aircraft. However, an object appears on the runway as the aircraft comes in for landing
- ATC does not issue a go around for the aircraft because they do not notice the object

Possible Requirements are:

- ATC must track the aircraft's approach completely until the aircraft enters the hangar