# NAVAL POSTGRADUATE SCHOOL

**MONTEREY, CALIFORNIA**

# DISSERTATION

**MISSION-BASED ARCHITECTURE FOR SWARM COMPOSABILITY**

by

Kathleen Giles

March 2018

Dissertation Supervisor: Kristin Giammarco

**Approved for public release. Distribution is unlimited.**

THIS PAGE INTENTIONALLY LEFT BLANK

| REPORT DOCUMENTATION PAGE | | | Form Approved OMB No. 0704–0188 |
|---|---|---|---|
| *Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.* | | | |
| **1. AGENCY USE ONLY** *(Leave Blank)* | **2. REPORT DATE** March 2018 | **3. REPORT TYPE AND DATES COVERED** Dissertation 07-01-2014 to 03-30-2018 | |
| **4. TITLE AND SUBTITLE** MISSION-BASED ARCHITECTURE FOR SWARM COMPOSABILITY | | | **5. FUNDING NUMBERS** |
| **6. AUTHOR(S)** Kathleen Giles | | | |
| **7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Naval Postgraduate School, Monterey, CA 93943 | | | **8. PERFORMING ORGANIZATION REPORT NUMBER** |
| **9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)** N/A | | | **10. SPONSORING / MONITORING AGENCY REPORT NUMBER** |
| **11. SUPPLEMENTARY NOTES** The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: NPS.2017.0052-IR-EP7-A | | | |
| **12a. DISTRIBUTION / AVAILABILITY STATEMENT** Approved for public release. Distribution is unlimited. | | | **12b. DISTRIBUTION CODE** |

**13. ABSTRACT** *(maximum 200 words)*

This research introduces the Mission-based Architecture for Swarm Composability (MASC) and methodology. This dissertation applies a mission engineering approach with model-based systems engineering foundations to formalize a swarm architecture, which is an example of a complex adaptive system. This architectural framework and methodology extend current swarm system design methods, which are primarily bottom-up approaches focused on the behavior of individual agents. MASC introduces a top-down, hierarchical approach with an overarching mission decomposed into phases, tactics, plays, and algorithms. MASC is applied to three unmanned aerial vehicle swarm case studies and assessed for incorporating mission doctrine, enhancing architecture reusability, and improving user accessibility. The assessment of these three factors indicates that MASC improves the state-of-the art methods in complex adaptive system architecture design.

| **14. SUBJECT TERMS** | | | **15. NUMBER OF PAGES** 299 |
|---|---|---|---|
| swarm systems, complex adaptive systems, MBSE | | | **16. PRICE CODE** |
| **17. SECURITY CLASSIFICATION OF REPORT** | **18. SECURITY CLASSIFICATION OF THIS PAGE** | **19. SECURITY CLASSIFICATION OF ABSTRACT** | **20. LIMITATION OF ABSTRACT** |
| Unclassified | Unclassified | Unclassified | UU |

NSN 7540-01-280-5500 | Standard Form 298 (Rev. 2–89) Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

**Approved for public release. Distribution is unlimited.**

**MISSION-BASED ARCHITECTURE FOR SWARM COMPOSABILITY**

Kathleen Giles
Commander, United States Navy
B.S., U.S. Naval Academy, 1998
M.S., Johns Hopkins University, 2006

Submitted in partial fulfillment of the
requirements for the degree of

**DOCTOR OF PHILOSOPHY IN SYSTEMS ENGINEERING**
from the
**NAVAL POSTGRADUATE SCHOOL**
**March 2018**

Approved by:

Kristin Giammarco
Associate Professor of Systems Engineering
Dissertation Supervisor

Ronald Giachetti
Professor of Systems Engineering
Dissertation Committee Chair

Eugene Paulo
Associate Professor of Systems Engineering

Timothy H. Chung
Program Manager, Defense Advanced Research Projects Agency

Raymond R. Buettner
Associate Professor of Information Science

Approved by: Ronald Giachetti
Chair, Department of Systems Engineering

Approved by: Douglas Moses
Vice Provost for Academic Affairs

THIS PAGE INTENTIONALLY LEFT BLANK

# ABSTRACT

This research introduces the Mission-based Architecture for Swarm Composability (MASC) and methodology. This dissertation applies a mission engineering approach with model-based systems engineering foundations to formalize a swarm architecture, which is an example of a complex adaptive system. This architectural framework and methodology extend current swarm system design methods, which are primarily bottom-up approaches focused on the behavior of individual agents. MASC introduces a top-down, hierarchical approach with an overarching mission decomposed into phases, tactics, plays, and algorithms. MASC is applied to three unmanned aerial vehicle swarm case studies and assessed for incorporating mission doctrine, enhancing architecture reusability, and improving user accessibility. The assessment of these three factors indicates that MASC improves the state-of-the art methods in complex adaptive system architecture design.

THIS PAGE INTENTIONALLY LEFT BLANK

# Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

# List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

# List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

# List of Acronyms and Abbreviations

| | |
|---|---|
| **A-A** | air-to-air |
| **ACAT** | acquisition category |
| **AFOTEC** | Air Force Operational Test and Evaluation Center |
| **A-G** | air-to-ground |
| **AGL** | above ground level |
| **ALFUS** | Autonomy Levels For Unmanned Systems |
| **ALOC** | air lines of communication |
| **AOR** | area of responsibility |
| **ARSENL** | Advanced Robotic Systems Engineering Laboratory |
| **ASW** | Anti-Submarine Warfare |
| **ATO** | air tasking order |
| **BDA** | Battle Damage Assessment |
| **CARUS** | Cooperative Autonomous Reconfigurable UAV Swarm |
| **CAS** | complex adaptive systems |
| **C2** | command and control |
| **C4ISR** | command, control, communications, computers, intelligence, surveillance, and reconnaissance |
| **COI** | contact of interest |
| **CONOPS** | concept of operations |
| **COTS** | commercial-off-the-shelf |

| | |
|---|---|
| **CRUSER** | Consortium for Robotics and Unmanned Systems Education and Research |
| **DARPA** | Defense Advanced Research Projects Agency |
| **DOD** | Department of Defense |
| **DODAF** | Department of Defense Acquisition Framework |
| **DOS** | Department of State |
| **DOTE** | Office of the Director, Operational Test and Evaluation |
| **DRM** | design reference mission |
| **DTE** | developmental test and evaluation |
| **DTIC** | Defense Technical Information Center |
| **EA** | evolutionary algorithm |
| **EMCON** | emissions control |
| **EO** | electro-optic |
| **FAA** | Federal Aviation Administration |
| **FANET** | flying ad-hoc network |
| **FSM** | finite state machine |
| **GCS** | ground control station |
| **GOH** | Government of Haiti |
| **GSO** | Glowworm Swarm Optimization |
| **GUI** | graphical user interface |
| **HADR** | humanitarian assistance and disaster relief |
| **HN** | host nation |

| | |
|---|---|
| **HSI** | human systems integration |
| **HSR** | human subjects research |
| **HUAS** | heterogeneous unmanned aircraft system |
| **IGOs** | inter-government agencies |
| **INCOSE** | International Council on Systems Engineering |
| **IR** | infrared |
| **ISR** | intelligence, surveillance, reconnaissance |
| **JCA** | Joint Capability Areas |
| **JCIDS** | Joint Capabilities Integration and Development System |
| **JFACC** | joint force air component commander |
| **JHUAPL** | Johns Hopkins University Applied Physics Lab |
| **JUONS** | Joint Urgent Operational Needs Statement |
| **JTF** | joint task force |
| **JTFC2** | joint task force command and control node |
| **LCAC** | landing craft air cushion |
| **LHA** | landing helicopter assault |
| **LHD** | landing helicopter dock |
| **LIDAR** | light detection and ranging |
| **LML** | Lifecycle Modeling Language |
| **MANET** | mobile ad-hoc network |
| **MASC** | Mission-based Architecture for Swarm Composability |
| **MBSE** | model-based systems engineering |

| | |
|---|---|
| **MIO** | maritime interdiction operations |
| **MIOC** | Maritime Interdiction Operations Commander |
| **MOE** | measure of effectiveness |
| **MOPs** | measures of performance |
| **MP** | Monterey Phoenix |
| **MSC** | Military Sealift Command |
| **NASA** | National Aeronautics and Space Administration |
| **NCO** | non-commissioned officer |
| **NGO** | non-governmental organization |
| **NIFC-CA** | Naval Integrated Fire Control-Counter Air |
| **NPS** | Naval Postgraduate School |
| **OFFSET** | OFFensive Swarm-Enabled Tactics |
| **OPORD** | operational order |
| **OPSIT** | operational situation |
| **OPTASK** | operational tasking orders |
| **OPTASK SUPPS** | operational tasking order supplements |
| **OSA** | open systems architecture |
| **OSC** | on-scene commander |
| **OSI** | open systems interconnection |
| **OTE** | operational test and evaluation |
| **PFSM** | probabilistic finite state machine |
| **PMO** | Project Manager Office |

| | |
|---|---|
| **PSO** | Particle Swarm Optimization |
| **RF** | radio frequency |
| **RHIB** | rigid-hull inflatable boat |
| **ROE** | rules of engagement |
| **RTL** | return to rally point |
| **SAR** | search and rescue |
| **SASC** | Service Academies Swarm Challenge |
| **SEAD** | suppression of enemy air defenses |
| **SLOC** | sea lines of communication |
| **SME** | subject matter expert |
| **SoS** | system of systems |
| **SoSE** | system of systems engineering |
| **SPINS** | special instructions |
| **SUWC** | Surface Warfare Commander |
| **SvS** | Swarm versus Swarm |
| **SysML** | Systems Modeling Language |
| **TACRON** | tactical air control squadron |
| **TCP** | Transmission Control Protocol |
| **TOGAF** | The Open Group Architectural Framework |
| **TOI** | target of interest |
| **TTP** | tactics, techniques, and procedures |
| **UAS** | unmanned aerial system |

| | |
|---|---|
| **UAV** | unmanned aerial vehicle |
| **UDP** | User Datagram Protocol |
| **UGS** | unmanned ground system |
| **UGV** | unmanned ground vehicle |
| **UML** | Universal Modeling Language |
| **UN** | United Nations |
| **UNSCR** | United Nations Security Council Resolution |
| **USAID** | United States Agency for International Development |
| **USG** | United States government |
| **USN** | U.S. Navy |
| **USV** | unmanned surface vehicle |
| **UUV** | unmanned underwater vehicle |
| **VADM** | Vice Admiral |
| **VANET** | vehicular ad-hoc network |
| **VBSS** | visit, board, search and seizure |

# Executive Summary

This dissertation presents an architectural framework and methodology for developing operational unmanned aerial vehicle (UAV) swarm systems. Mission-based Architecture for Swarm Composability (MASC) formalizes swarm mission doctrine as a primary design factor in swarm systems engineering to promote architecture reusability and operator usability. This research merges methods from multiple academic disciplines by integrating systems engineering principles into the interdisciplinary field of swarm robotics. As complex adaptive systems that exhibit collective emergent behavior, swarm systems present a significant design challenge to systems engineers. Swarm system research is an emerging field in which the pertinent design factors are not well established. Swarm systems lack a mission doctrine for guiding system architecture development. Consequently, much of the existing work has resulted in point designs that require specialized programming skills not held by fleet operators. This exploratory design research, using a UAV swarm as a case study within specific mission sets, aims to incorporate swarm doctrine into swarm system architecture design, foster greater system architecture reuse, and enable programming at a level of abstraction commensurate with fleet operator capabilities.

Swarm systems are "large numbers of relatively simple physically embodied agents [that] can be designed such that a desired collective behavior emerges from the local interaction among agents and between agents and the environment"[1]. Swarm technology is rooted in several disciplines: robotics, artificial intelligence, and evolutionary biology. The military's attention to swarm systems has grown over the last two decades due to their potential expendability, redundancy, and expanded sensor coverage. Much of this interest can be attributed to the dynamic field of unmanned systems technology, which has been rapidly developing both in government and in the private sector. To advance from current unmanned systems paradigm in which a a single pilot controls a vehicle or a few vehicles at most [2], [3], [4], to remotely supervised swarms, the system should be designed to limit the cognitive burden on the human operator.

Bottom-up system design approaches, such as agent-based modeling, finite state machines (FSMs), and Petri Nets, focus on assembling sub-components of systems to build more complex systems and are frequently employed in swarm system design. Bottom-up models

are advantageous from a modularity and composability perspective, but they often risk failing to meet higher level system requirements. For that reason, the software engineering heuristic of combining bottom-up and top-down methods [5] is applied to a UAV swarm system to augment the typical bottom-up, behavior-based design approach.

The design research approach is used to develop a new methodology for composing swarm mission architectures that leverages common operational patterns. Design research explores the problem space, describes the design method, evaluates the method using iteration, and documents the changes and knowledge gained in building the artifact [6], [7]. Figure 1 shows the proposed swarm operational framework, MASC, which is applied to three mission case studies. Using MASC, a mission is decomposed into phases, which are composed of a library of tactics. Similarly, tactics are decomposed into plays, and plays are composed of algorithms. Within the context of the MASC framework, the swarm algorithms reside at the boundary wherein the operational architecture ends and the solution architecture begins. Swarm robotics research up to this point has focused on the algorithm and play (behavior or primitive in common parlance) levels. This research focuses on the operational part of the architecture—the missions, phases, tactics, and plays.

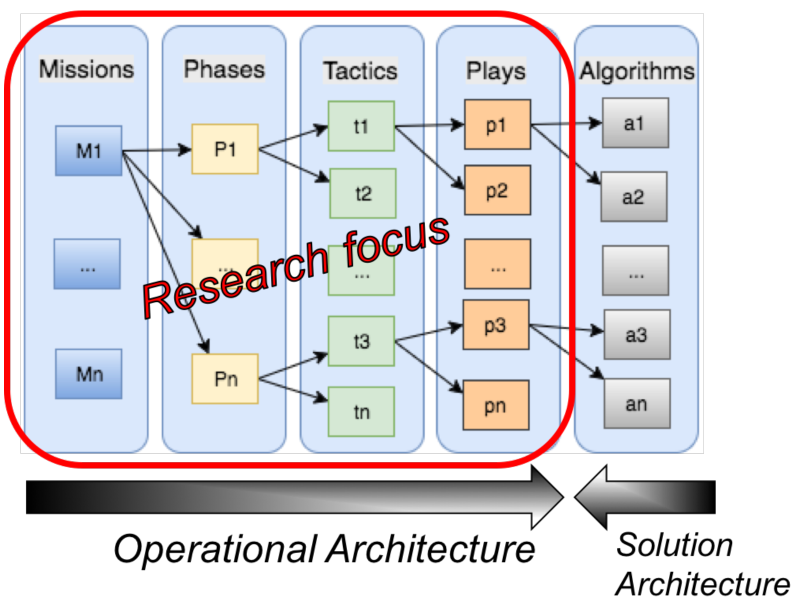



Figure 1. MASC Framework

This dissertation applies a mission engineering approach with model-based systems engineering foundations to formalize a swarm architecture, which is an example of a complex adaptive system. MASC's taxonomy and swarm-specific playbook are key enablers for this model-based systems engineering (MBSE)-founded swarm architecture. The taxonomy provides a formal, mission-focused naming convention for categorizing elements of a relatively new type of complex adaptive system. MASC is applied to three mission case studies to demonstrate incorporation of mission doctrine into the design process, architecture reusability, and accessibility to the operational user community.

## References


[1] E. Sahin, "Swarm robotics: From sources of inspiration to domains of application," in *International Workshop on Swarm Robotics*, no. 631. Santa Monica, CA: Springer Berlin Heidelberg, 2004.

[2] M. Cummings, C. Nehme, and J. Crandall, "Predicting operator capacity for supervisory control of multiple UAVs previous experimental multiple UAV studies," *Innovations in Intelligent Machines-1*, vol. 37, pp. 11–37, 2007.

[3] D. R. J. Olsen and S. B. Wood, "Fan-out: Measuring human control of multiple robots," in *Proceedings of the SIGCHI Conference on Human Factors in Computing System*, no. 1. Vienna, Austria: ACM, 2004, vol. 6, pp. 231–238.

[4] B. Hilburn, P. G. Jorna, E. A. Byrne, and R. Parasuraman,"The effect of adaptive air traffic control (ATC) decision aiding on controller mental workload," *Human-Automation Interaction: Research and Practice*, pp. 84–91, 1997.

[5] D. Buede, *The Engineering Design of Systems: Models and Methods*, 2nd ed. Hoboken, NJ: John Wiley & Sons, Inc., 2009.

[6] R. Giachetti, "Systems engineering thesis methods," Monterey, CA, 2016.

[7] N. Cross and R. Roy, *Engineering Design Methods*, 4th ed. New York: Wiley, 1989.

THIS PAGE INTENTIONALLY LEFT BLANK

# Acknowledgments

I am grateful for the support of my dissertation committee throughout this work. Thank you, Kristin, for your system architecture modeling insight, patience, and mentoring; Tim, for your swarm robotics expertise and for inspiring this research topic; Ron, for your detailed, perceptive feedback; and Gene and Ray, for your big picture view and multi-disciplinary perspective. A special thank you to Mikhail Auguston, who was not officially on my committee but taught me much about Monterey Phoenix and abstract thinking. Finally, thank you, Scott, for supporting me throughout this journey.

THIS PAGE INTENTIONALLY LEFT BLANK

# CHAPTER 1:
# Introduction

*Portions of this chapter previously appeared in the Proceedings of the Complex Adaptive Systems Conference, 2017 [1].*

This dissertation presents an architectural framework and methodology for designing a swarm system. Swarm systems are complex adaptive systems (CAS) that present significant design challenges to systems engineers. The key design factors in this emerging interdisciplinary field are not well established. Swarm systems lack a mission doctrine for guiding system architecture development. Doctrine stipulates a standardized framework for linking military mission strategy, operations, and tactics, that is influenced by technology, organizational structure, the enemy's capabilities, and geography. Current methods for swarm system design are primarily bottom-up methods focused on the behaviors of individual agents and lack the mission-level perspective as a prescribing design factor. Consequently, existing designs are limited in their reusability across missions and require specialized programming skills not compatible with the average fleet operator. This dissertation contributes to the field of systems engineering through the development of Mission-based Architecture for Swarm Composability (MASC), a mission engineering approach that applies model-based systems engineering (MBSE) foundations for developing swarm architectures using a top-down methodology that is iteratively refined via bottom-up feedback methods.

Mission-based Architecture for Swarm Composability (MASC) formalizes swarm mission doctrine as a primary design factor in swarm systems engineering to promote architecture reusability across missions and support mission planning in future operational systems. This research employs model-based design methods to formalize an unmanned aerial vehicle (UAV) swarm system architecture by structuring the doctrinal input with the same formality used for technical specifications, using a mission engineering approach. The MASC framework and methodology aim to break down the swarm system's complexity into more manageable pattern elements and focuses on designing behaviors at the sub-swarm level rather than the individual agent level. MASC is constructed using a top-down, hierarchical approach with an overarching mission decomposed into phases, tactics, plays, and algorithms. MASC is also designed to support operational mission planning by focusing the operator's attention to the swarm tactics level rather than forcing them to program low-level behaviors. This architecture is demonstrated for three different missions. This work was inspired by and builds upon the groundbreaking UAV swarm field experimentation conducted by Advanced Robotic Systems Engineering Laboratory (ARSENL) [2]–[4].

## 1.1 Background

Swarm system research is nascent; the relevant design factors and tradespace are not well understood. Much work has been done in swarm systems (described in Chapter 2), but it has been fragmented, resulting in point designs rather than an overall design framework. Scharre asserts that swarm systems will "pose profound operational and policy challenges," requiring new doctrine, capabilities, organizational structures and experimentation with these novel constructs [5]. In the systems engineering field, Beery [6] developed the Methodology for Employing Architecture in System Analysis (MEASA) to formalize the connection between model-based systems engineering (MBSE) architecture description models and analysis models for traditional systems, while Gillespie [7] extended this work for system of systems (SoS). Swarm systems present additional challenges to systems engineers because they do not have existing doctrine with which to inform a system architecture.

Mission engineering is the "deliberate planning, analyzing, organizing, and integrating of current and emerging operational and system capabilities to achieve desired warfighting mission effects" [8]. Mission engineering leverages system integration methods for composing mission functions to define a mission capability and treats the *mission* as the system [8], [9]. By focusing on the swarm system from a mission perspective, doctrine is instituted as a primary design factor for architecting the system. MBSE provides the formalized application of Lifecycle Modeling Language (LML) and Systems Modeling Language (SysML) models and automated tools to focus on swarm architecture design using the mission engineering approach. Defining mission functions in a modular, composable architecture allows automation to be incorporated into the design.

Automation enables progression from remotely piloted UAVs under the "single robot parenting" [10] paradigm, to remotely supervised swarms. Automation allows tasks previously accomplished by humans to be offloaded to the mechanical vehicle, or agent. The way in which humans supervise, control, and intervene with swarms will be different from how they currently interact with just one or two agents [11]. Swarm operations requires the operator to assume a supervisory role at the macroscopic level and reduce attention to the individual vehicle level. Incorporating automation for basic vehicle guidance and navigation has shown to reduce pilot workload for single and multi-UAV systems, freeing cognitive capacity for mission level management and decision-making [12]–[14].

Limited research has been conducted on human swarm interaction or automation for operating large numbers (50) of vehicles [3], [15], [16]. The high vehicle-to-human ratio associated with swarm systems makes them cognitively demanding for humans to operate. Experimental and theoretical research on multi-UAV control has found five to six UAVs to be the upper bound for a single operator to control directly [3], [14], [17]. For this reason, ARSENL operates UAV swarms by allocation of functional tasks rather than assignment of vehicle control to an operator, with the Swarm Commander commanding the behaviors of the entire swarm and the Health Monitor managing any errant UAVs by separating them from the swarm [3]. While this approach has proved successful in scaling operations up to a 50-UAV swarm, the Swarm Commander (an experienced software programmer) currently commands each sub-swarm with single behaviors which becomes cognitively demanding when the swarm is divided into sub-swarms performing different behaviors. As evidenced by the author's experience as a Mission Commander at the 2017 Defense Advanced Research Projects Agency (DARPA) Service Academies Swarm Challenge (SASC), UAV swarm versus swarm engagements happen quickly. The teams that were able to stitch together multiple behaviors that executed according to adversary-induced triggers were able to gain a tactical advantage.

Combining mission engineering and MBSE facilitates architecture reusability. Many of the missions performed by unmanned systems share common operational patterns. For example, search and rescue (SAR), maritime interdiction operations (MIO), intelligence, surveillance, reconnaissance (ISR), and Anti-Submarine Warfare (ASW) missions all require a system to search and track a target. While differences exist within the environments and target parameters for these missions, the search and track tasks are similar and offer opportunities for reusable patterns. Furthermore, for each of these missions, the system must launch, transit to and from the operating area, and return to base. Common mission elements can be designed for reuse across multiple missions by leveraging shared patterns rather than designing for specific missions, domains, or warfare communities.

*Playbooks* are an efficient mission-focused way to manage common tactical patterns and simplify operator usability for multi-robot systems [18], [19]. Providing a collection of pre-defined operations simplifies user management, allows for variability in system autonomy, synchronizes agent tasking, and eases required communications [20]. A suitable playbook should be constructed at a high enough level of abstraction such that action plans (*tactics* in the MASC framework) are common to several missions and can be tailored for a variety of uses by adjusting tune-able parameters. A benefit to using a playbook architecture for UAVs is the possibility to extend the pool of potential operators beyond just those who are specifically trained UAV operators.

## 1.2 Statement of the Problem

Swarm systems are being engineered without guidance from swarm mission doctrine, system architectures have limited reusability, and the low-level programming required to operate them inhibits swarm system operational use. Development of mission doctrine is dependent upon a comprehensive understanding of the problem space; swarm systems are new and their range of employment is not well understood. The focus on low-level, individual agent behaviors may be sufficient for lab and field experimentation, but is ineffective for operational systems.

## 1.3 Research Objective

This research aims to incorporate mission doctrine into the design process, enable architecture reusability, and improve user accessibility by augmenting traditional bottom-up, behavior-based design with a top-down, mission-based framework. Software design has progressed from low-level machine languages to ever higher levels of abstractions, allowing humans to manage system complexity [21]. This principle of abstraction is borrowed from software engineering and applied to UAV swarm system design so that the architecture can be designed using a top-down framework to augment traditional bottom-up, behavior-based, swarm system design methods. A goal of the research is to formalize an operational framework of versatile common patterns for swarm missions that promotes architecture reusability and improves operator usability.

## 1.4 Research Methodology

The framework is pursued using a design research approach to propose a new methodology for composing swarm mission architectures as depicted in Figure 1.1. Design research explores the problem space, describes the design method, evaluates the method using iteration, and documents the changes and knowledge gained in building the artifact [22], [23].

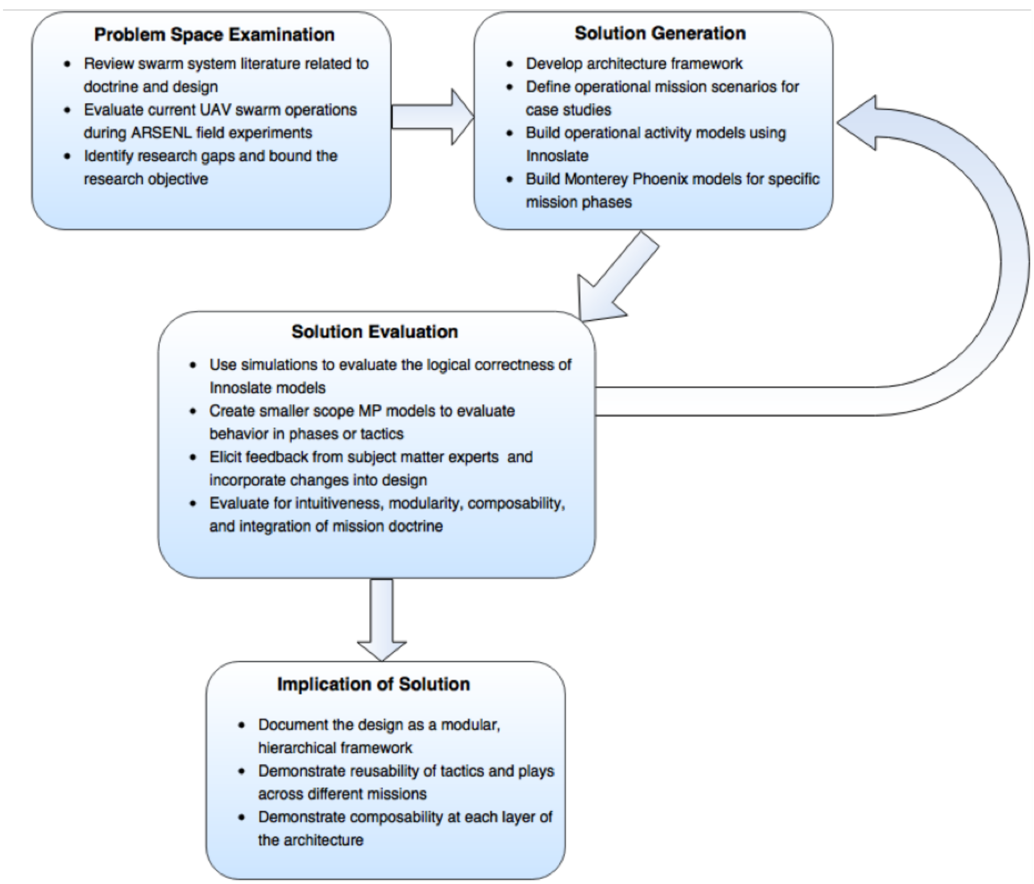



Figure 1.1. The research methodology follows a design research approach. Adapted from [23].

### 1.4.1 Problem Space Examination

First, the problem space is explored by reviewing swarm unmanned system literature from a technical and doctrinal perspective, and identifying gaps in the current research. Applicable literature includes academic fields ranging from engineering and computer science to cognitive psychology. Participating in the quarterly ARSENL UAV swarm field experimentation provides practical and pertinent data regarding how large swarms are currently operated in the field experimentation domain.

### 1.4.2 Solution Generation

Next, the architecture called MASC is developed as a composable model intended to be abstract enough to apply across a range of operational missions. The architecture should incorporate mission doctrine, demonstrate modularity across missions, structured using composable elements, and be intuitive to the human operator. These attributes are important for promoting resuability across missions in a conceptual level architecture. For the purpose of this dissertation, a swarm is defined as a group of 20 or greater individual, self-organized, homogeneous UAVs that perform a mission through local interactions under a decentralized control architecture [24], [25].

#### MASC Taxonomy

MASC describes how a swarm mission is composed of modular, reusable templates of tactics and plays. The following UAV swarm mission taxonomy is proposed to describe the overall mission architecture, and includes the following terms:

- A *swarm mission* describes the overall task and purpose delineating the action assigned to the UAV swarm. Example swarm missions include: ISR, MIO, humanitarian assistance and disaster relief (HADR), SAR, and counter drug operations. Each swarm mission is the parent of five swarm mission phases.
- A *swarm mission phase* describes a distinct time period within the mission. There are five phases in a swarm mission: Preflight, Ingress, OnStation, Egress, and Postflight. The three phases that cover the in-flight portion of the mission—Ingress, OnStation, and Egress—are the focus of this research. A swarm mission phase is composed of one or more swarm tactics.

- A *swarm tactic* is the employment and ordered arrangement of agents in relation to one another for the purpose of performing a specific task [26]. Swarm tactics include `Efficient search`, `Track`, `Evade`, and `Attack`. A swarm tactic can be used in multiple swarm missions and is composed of one or more swarm plays. Swarm tactics are designed to be used in multiple missions.
- A *swarm play* describes the lower-level maneuvers and behaviors of the swarm as a collective of agents [26]. The artificial intelligence and robotics communities use the term "behavior" to describe "a regularity in the interaction dynamics between the agent and the environment" [27]. Swarm plays can be described as behaviors with specific triggers and temporal constraints, and are the building blocks of swarm tactics. Example swarm plays include `Launch`, `Transit to waypoint`, `Split`, `Join`, and `Orbit`. Swarm play parameters are tunable characteristics of a play that can be changed based on the mission or rules of engagement (ROE). A swarm play may be used in multiple swarm tactics and is composed of one or more swarm algorithms.
- *Swarm algorithms* are the step-by-step procedures used by the controlling software to solve a recurrent task such as sorting, path planning or foraging. Swarm algorithms are the building blocks of swarm plays. Swarm algorithms use data from the individual UAVs such as position, heading, velocity, altitude, attitude, health status, and state [26], [28].

Figure 1.2 shows the proposed decomposition of the swarm operational architecture. A mission is decomposed into phases, which are composed of a library of tactics. Similarly, tactics are decomposed into plays, and plays are composed of algorithms. Within the context of the MASC framework, the swarm algorithms reside at the boundary wherein the operational architecture ends and the solution architecture begins. Swarm robotics research up to this point has focused on the algorithm and play (behavior or primitive in common parlance) levels. This research focuses on the operational part of the architecture—the missions, phases, tactics, and plays.

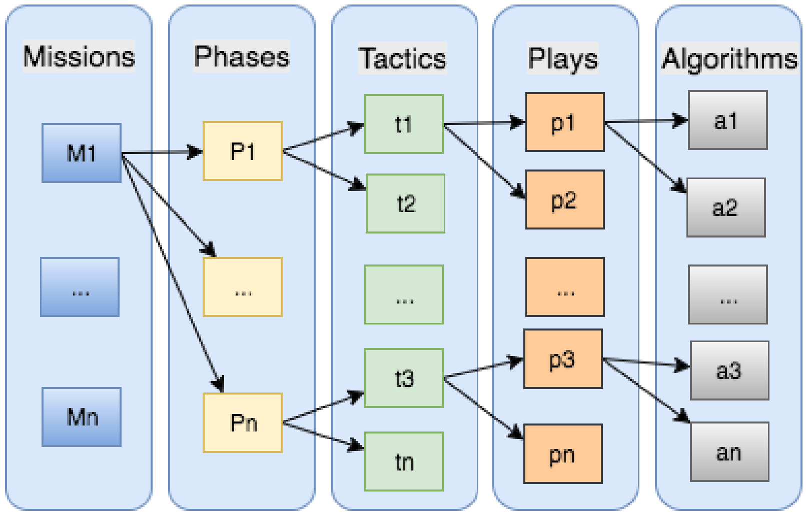



Figure 1.2. MASC is a many-to-many framework of elements starting with missions at the highest level. Each mission is composed of phases, tactics, plays, and algorithms at the lowest level.

Figure 1.3 illustrates a swarm mission at the phase level. Each of the five phases (Preflight, Ingress, OnStation, Egress, and Postflight) are used for a swarm mission. This research is focused on the in-flight phases: Ingress, OnStation, and Egress. The Ingress and Egress phases are similarly composed for each mission modeled, with minor variations due to mission-specific ROE. The major compositional differences occur in the OnStation phase, a result of the variety in mission objectives and requisite level of human involvement. Each phase decomposes into its corresponding tactics at the next lower level in the model, as indicated by the "decomposed" text at the bottom of each box. Figure 1.3 represents the majority of swarm missions at the phase level. Exceptions to this standard mission flow are missions in which the UAVs are deemed expendable and recovery is not planned. For such cases, the Egress and Postflight phases are absent.


Figure 1.3. Each swarm mission is composed of five operational phases (generated using Innoslate)

### 1.4.3 Solution Evaluation

MASC's effectiveness in formalizing an operational framework of common patterns in swarm mission that promote architecture reuse is evaluated in Chapter 5 during its application to three mission case studies (Chapter 4) using the following criteria:

- The architecture should instantiate swarm doctrine to support mission planning and operations for a variety of missions including air-to-air (A-A), maritime, and overland operations.
- The architecture should be modular. With the exception of temporally constrained activities such as ingress or egress, the tactics should be rearrangeable and reusable to support a variety of missions and conditions. The elements should be portable across a variety of missions. There should not be a separate set of tactics developed for each mission.
- The architecture should be composable; it should be capable of "select[ing] and assembl[ing] simulation components in various combinations into valid simulation systems to satisfy specific user requirements" [29].
- The architecture should be intuitive and graspable [30]; a current fleet aviator with operational experience should be able to construct a swarm mission plan.

MASC's modularity and composability are evaluated by applying it to three different case study missions. The first mission, Swarm versus Swarm (SvS), is representative of the field experimentation conducted by the ARSENL team from the Naval Postgraduate School (NPS). ARSENL's research is currently focused on developing A-A tactics for employment against an adversary swarm. In the SvS mission, the swarm functions at a higher level of automation, involving less human intervention than it does in the second and third missions. The HADR and MIO are notional UAV swarm operational navy missions that require less time-critical decision-making, operate at a lower level of automation, and thus need more human interaction. Chapter 3 presents case studies for MASC, with the associated design reference missions described in Appendix B. Innoslate is a web-based MBSE software tool, developed by Spec Innovations, used to support the systems development life cycle including: requirements management, requirements analysis, and Department of Defense Acquisition Framework (DODAF) 2.02 compliant architectures. It enables physical and behavioral system modeling, and provides system design traceability. It is used to document the swarm mission architectures, swarm system hierarchies, and swarm mission activity flows developed to support this research. Operational activity models are developed in Innoslate for each scenario using MASC elements.

To support composability evaluation, the models are evaluated for logical correctness using Innoslate simulations, Monterey Phoenix (MP) models, and subsequent MP event trace generation. Monterey Phoenix is used in conjunction with Innoslate to generate mission use cases (as event traces) based on the swarm mission scenarios. Monterey Phoenix is an NPS-developed formal language and method for modeling system behaviors and operational processes in systems architecting.

MASC's intuitiveness and swarm doctrine incorporation is evaluated by having human subjects research (HSR) participants construct swarm mission plans using MASC elements during a table-top exercise [31] (described in Appendix C). Following the exercise, the participants answer several questions regarding their perception of task workload and provide feedback on the structure of MASC elements within the mission context. The participant's mission plans are analyzed for consistency and appropriateness of MASC element usage, and common patterns of elements. Then the mission plans are compared to the baseline Innoslate model.

### 1.4.4 Solution Implication

Finally, the knowledge gained from the stakeholder inputs is incorporated into the final MASC design demonstrating the reusability of tactics and plays across different mission scenarios. Additions and eliminations of tactics are documented, in addition to gaps in the composability of the model. Figure 1.4 is an example of the MASC framework applied to the MIO mission, partitioned into its respective phases (in yellow), tactics (in green), plays (orange), and algorithms (grey).

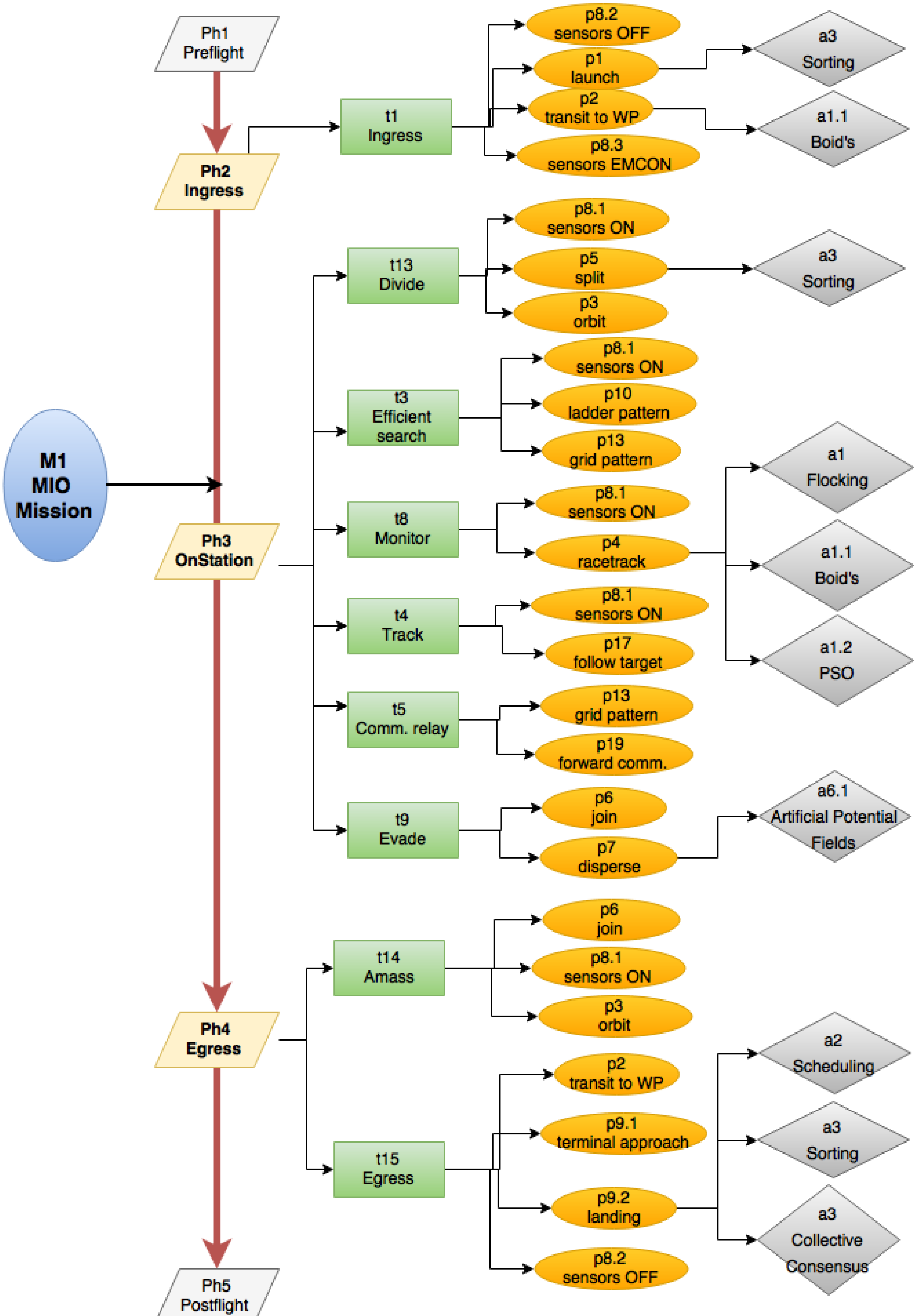





Figure 1.4. Example of an MIO mission architecture composed of phases (yellow), tactics (green), plays (orange), and a subset of the algorithms (grey).

## 1.5 Scope of Study

The scope of this research is bounded to accommodate the multi-dimensional complexity of swarm robotics systems in terms of diversity of agents, degree of inter-agent coordination and communication, and the extent of human interaction. The Defense Science Board [32] characterizes robotic swarms in terms of five attributes, as shown in Table 1.1.

Table 1.1. The Defense Science Board categorizes swarm robotic systems in terms of five attributes. Source: [32].

| **Attribute** | **Harder to implement**<br>*Most DoD "swarm" efforts* | **Easier to implement**<br>*Natural examples* |
|---|---|---|
| Diversity | Heterogeneous<br>*e.g.*, mixed ground and air platforms | Homogeneous<br>*e.g.*, standard platform, perhaps with modular payloads |
| "Intelligence" | High<br>*e.g.*, complex reasoning | Minimal<br>*e.g.*, simple, pre-defined rule sets |
| C2/decision making | Complex<br>*e.g.*, highly interactive decisions | Minimal<br>*e.g.*, implicit C2 |
| Communications bandwidth | High<br>*e.g.*, to provide detailed intra- (or extra-) swarm updates | Low<br>*e.g.*, stigmergy (environmental marking) |
| Complexity of human interaction | High<br>*e.g.*, could require advanced human-machine interface | Minimal<br>*e.g.*, limited to human giving "Go" command |

Chung also characterizes swarm systems in terms of five attributes that characterize system complexity [3]. As Figure 1.5 shows, different parameters can be scaled up to escalate system complexity. As ARSENL learned, increasing a single parameter (i.e., number of agents) has drastic impacts on system logistics, processes, and human interaction [3].

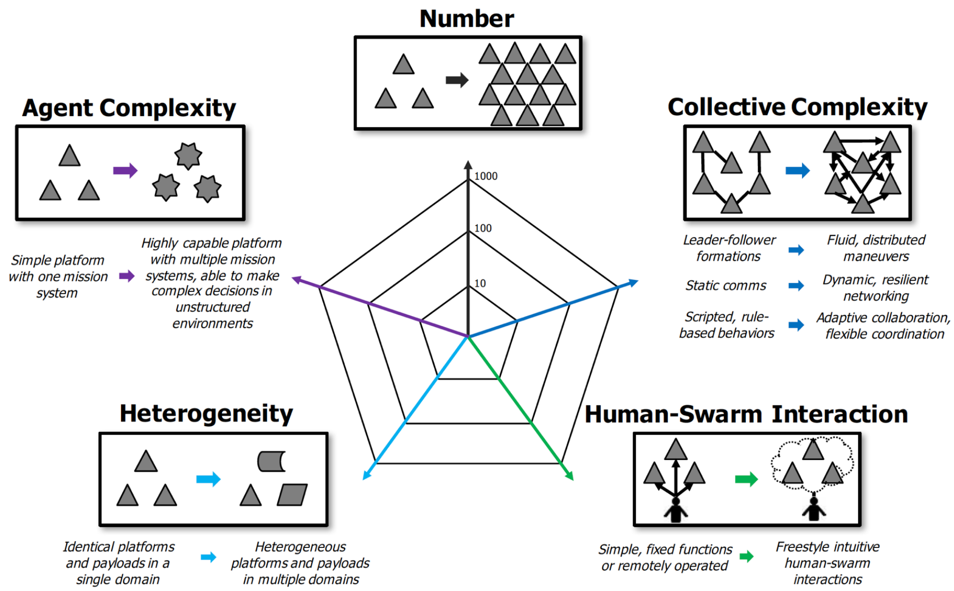



Figure 1.5. Swarm system complexity evaluated in terms of individual agent complexity, number of agents in the swarm, collective complexity of the agents' interactions, extent of interaction with the human operator, and degree of agent similarity. Source: [33].

In relation to the paradigms shown in Table 1.1 and Figure 1.5, this research will focus on homogeneous, UAV swarms of 20 or more, operating under distributed command and control (C2) and moderately compact rule sets, with variable human interaction (depending on the mission). Within those bounds, the research will focus on developing a UAV swarm taxonomy, and a high-level architecture consisting of missions decomposed into tactics and plays.

The following areas are considered out of scope for this research:

- mission decomposition for every possible swarm mission,
- modeling of individual interactions between agents,
- collision avoidance solutions,
- human machine interface design of swarm controlling software,
- detailed logistical implications of UAV swarm operations, and
- ROE and legal ramifications of operating UAV swarms.

## 1.6 Summary

There is a need to formalize mission doctrine as an influencing factor in swarm system architecture design. As swarm systems become more prevalent in Department of Defense (DOD), a common method for categorizing their collective behavior is needed and not fully addressed by current swarm system architectures. This research explores augmenting current heuristic-based, bottom-up swarm robotics design methods with an MBSE-founded, top-down, mission-based approach. A common, reusable library of tactics and plays is designed to support a variety of missions. Three mission scenarios are demonstrated using the MASC architecture and methodology. The next chapter surveys existing research relevant to swarm system design.

# CHAPTER 2:
Literature Review

*Portions of this chapter previously appeared in the Proceedings of the Complex Adaptive Systems Conference, 2017 [1].*

After describing general attributes of swarm systems and challenges in swarm system development, the first part of this chapter discusses methodologies that have been used in swarm system design. Swarm system design methods from various academic fields such as computer science, robotics, cognitive psychology, and ecology are examined. Early swarm system modeling methodologies approached the problem with a bottom-up perspective by focusing on the interactions between agents at the individual agent level. These bottom-up methods such as agent-based modeling, finite state machines (FSMs), and Petri Nets are commonly used in the computer science and robotics communities. More recent methodologies consider the swarm system from a higher level of abstraction, concentrating on the collective behavior of the agents in addition to the lower-level interactions.

The second part of this chapter discusses influential factors in swarm design such as swarm doctrine, communication architecture, and human-swarm interaction. Swarm doctrine is surveyed from a historical view of military swarming strategies throughout various conflicts. Communication architectures are reviewed, as they impact both the doctrinal and technical aspects of UAV swarm architecture. Similarly, the human component must not be ignored as a major contributor to UAV swarm design. The sheer number of agents in a swarm demands a shift from the customary role of a pilot directly controlling the system toward a mission or system manager.

## 2.1 Swarm Systems

Swarm systems have roots in many disciplines, such as artificial intelligence, evolutionary biology, and robotics. Key enabling technologies to swarm system development include improved communication networks, cost effective miniaturization of electronics, and automation. The UAVs in a swarm must communicate for collision avoidance, sensor payload management, and health monitoring. Communication must be timely. Meshed ad-hoc network architectures, where the network nodes self-organize their forward-relay capabilities, have shown promise in minimizing frequency spectrum and bandwidth conflicts and providing reliability and flexibility in swarm communication [2], [34]. Miniaturization of electronics including radio receivers, GPS, video cameras, and autopilot processors has made UAV swarm agents smaller, lighter, and more capable. The dramatic drop in cost and the increase in availability of these components have made swarm systems affordable. These trends are likely to continue.

Military interest in swarm technology has increased over the last two decades due to their redundancy, expanded sensor coverage, and potential to be expendable. Much of this interest can be attributed to the dynamic field of unmanned systems technology, which has been rapidly developing both in government and in the private sector. Unmanned system technology has expanded from physically hazardous, high-altitude, extended-endurance military missions to agriculture, mining, search and rescue, and environmental research civilian and commercial missions [35]. Unmanned systems provide many advantages over manned systems. Unmanned aerial systems are less constrained by human factors, such as crew rest, G-tolerance, environmental conditions, and comfort. Unmanned systems can be expendable and could have lower life-cycle costs than manned systems; however, low system reliability [36], low technology readiness levels, large logistical footprints, and an ironic increased manpower requirement have marginalized cost advantages. Likewise, unmanned systems' test and evaluation struggles [37] and poor track record for meeting operational effectiveness and suitability requirements have historically contributed to higher system life cycle costs.

Swarm application to unmanned systems derives inspiration from biology. Large numbers of individuals such as birds, fish, or insects may collectively work together to accomplish useful tasks that cannot be completed by an individual or any group of non-cooperative individuals. Swarm robotics is the "study of how large numbers of relatively simple, physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment" [38]. Members of the swarm may be unintelligent and inefficient on an individual scale, yet inter-agent interactions produce emergent behavior that enables advantages such as robustness, flexibility, and scalability [38]. Beneficial collective behaviors may be elicited by inter-agent exchanges or interactions between agents and the environment [38]. When a multiple-agent system performs a task that increases the total utility of the system, it has then accomplished cooperative behavior [39]. At first glance, a swarm may be confused with a team; however, there are several key differences between the two. Clough [40] compares general attributes of swarms and teams in Table 2.1. His characterization of swarms is relevant to this research; however, the composition of a swarm should not be limited to only homogeneous configurations. Future naval missions will require multi-domain swarms in the form of UAVs, unmanned surface vehicles (USVs), and unmanned underwater vehicles (UUVs).

Table 2.1. A comparison showing the differences between swarms and teams. Source: [40].

| **Attribute** | **Swarm** | **Team** |
|---|---|---|
| Temporal | reactive | predictive |
| Composition | homogeneous | heterogeneous |
| Interrelationships | simple | complex |
| Predictability | probabilistic | deterministic |
| Individual worth | expendable | critical |
| Efficiency | low | high |

## 2.2 Swarm System Challenges

Swarm systems, like other CAS, present significant challenges to traditional systems engineering processes in which systems are expected to execute predictable tasks within a constrained environment. Complex systems engineering is aimed toward producing systems "capable of adaptation, change and novelty" [41] and therefore necessitates a different approach than the traditional systems engineering top-down methodology [42]. The complex systems engineering approach must focus on the coordination aspect of the system, more specifically "quantifying the coordination above the level of behavior of the individual" [41]. Minai and Braha state that "in order for system to be effective, it must be able to coordinate the right number of components to serve each task, while allowing the independence of other sets of components so that they can perform their respective tasks without binding the action of one such set to another" [41]. This insight is telling for designing

UAV swarm architectures and tactics, as it shapes the organization and task allocation of the system, keeping it flexible and adaptable for a variety of missions.

Swarm systems present a considerable challenge from an operational perspective. In [5], Scharre asserts that "the biggest challenge in adopting multi-vehicle control is not technical, but rather understanding the cognitive demands placed on the human operator and how many vehicles can be effectively controlled." The Swarm Commander will be required to control swarm systems at a higher level of abstraction than allowed for by current designs. To enable the swarm commander to focus on the mission, incorporating autonomy into both multi-vehicle control and tactical decision making is necessary [5]. Another significant operational challenge is the cultural acceptance of these systems. The Navy's cultural and organizational resistance to unmanned systems is well-documented [43]. Swarm systems—especially swarm UAV—compound this reluctance by requiring the operator to control more than one vehicle at a time [5]. A key finding of the Navy's 2017 study on cultural and organization impediments to unmanned systems is particularly applicable to swarm systems: "Experimentation to support military transformation must allow for exploration of new CONOPS and TTPs, and needs feedback mechanisms to enable iterative development of the new capabilities" [43].

## 2.3 Designing a UAV Swarm

Swarm system architectures fall into four general C2 categories: orchestrated, hierarchical, distributed, and emergent swarming control [44]. Taxonomies and nomenclature of swarm systems vary widely, which is not surprising for an emerging, multi-disciplinary technology. Bottom-up design methods, especially behavior-based ones, are the still the most prevalent swarm design methods. However, top-down methods are surfacing to augment the bottom-up models and expedite their progression from labs and simulations to operationally useful systems. Model-based composable architectures, such as playbooks, merge the benefits of top-down and bottom-up approaches and promote reusable designs. Finally, mission doctrine, communication architectures, and human-swarm interaction must be carefully considered as influential factors in designing UAV swarms that are operationally suitable for military missions.

### 2.3.1 Swarm System Command and Control Architectures

The UAV swarm architecture should be designed to support a dispersed and networked system. Selection of an appropriate swarm C2 structure should consider elements such as vulnerability to disruption, predictability, scalability, connectivity reliability, mission risk, response speed, and optimality [5]. In general, there are four main, high-level C2 architectures used in swarm-like systems: orchestrated, hierarchical, distributed, and emergent swarming control [44]. Dekker classifies the first three types as "situationally aware swarming," wherein the network is used to fuse sensor data to form an integrated tactical picture and coordinate actions [44]. Other researchers consider *centralized control* to be another architecture for multi-vehicle systems [39], [45]. It requires a hub-and-spoke communication architecture that presents several disadvantages: it limits the autonomous behavior of the system, it does not enable communication between agents, it limits scalability due to communication bandwidth requirements, and it allows for a single point of failure in the design [2], [46]. Therefore, this structure is not relevant to future swarm systems and is not discussed in this research.

In *orchestrated control*, one agent is selected as a temporary leader based on specified transient factors (e.g., location, state, mission scenario). The leader receives sensor data from the other agents and broadcasts the fused, common, integrated picture. If the leader is disabled, a replacement is selected to continue in that role. This architecture is somewhat robust, but is not scalable to larger swarms or geographically dispersed swarms, and places a significant processing burden on one agent.

A *hierarchical control* architecture resembles a traditional military command and control structure wherein agents are organized in a hierarchy and detailed tactical information is fed up the chain of command. While this hierarchical design simplifies data flow, its lower flexibility makes it less suitable in dealing with dynamic scenarios that require rapid reactions from agents.

A *distributed control* is characterized by the absence of a leader; rather, decisions are made via collective consensus among agents. This type of architecture is robust and scalable and can be used locally when low bandwidth communications preclude swarm-wide communication. The distributed architecture topology shown in Figure 2.1 is a completely connected communication network which is not a requirement for distributed control. Mesh ad-hoc networks allow agents to self-organize as communication relays to forward data to other agents [34], [47]. As with other elements of swarm system design, a hybrid of C2 architectures can be used to take advantage of the strengths of each. The U.S. Navy's Cooperative Engagement Capability anti-air warfare system utilizes a distributed architecture for situational awareness data and an orchestrated architecture for selecting targets [44]. Distributed control architectures using market-based (or auction) methods and implicitly derived single-agent solutions, have been successfully demonstrated in UAV swarms [2].

Finally, while most researchers recognize the first three categories ( [39], [45], [48]), Dekker names a fourth. He uses *emergent swarming* to describe the relationships which occur in ant, termite, and bee colonies in which there is no management [44]. These agents have no leader, have low situational awareness, and follow simple rules based on local information (i.e., sharing pheromone signals) [44]. This type of architecture has the potential to become more relevant as genetic algorithms are further developed [49]–[51]. Figure 2.1 shows the four general types of swarming C2 architectures.

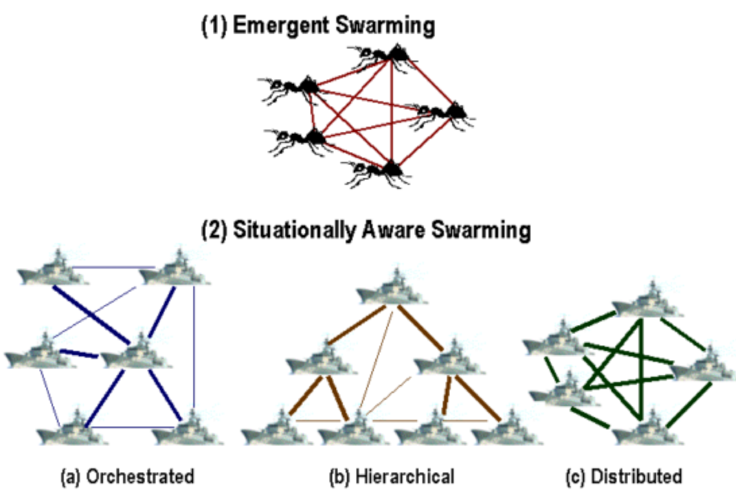



Figure 2.1. The four general categories of swarm C2 architectures. Emergent swarming is found in nature and has no prescribed management scheme. Orchestrated, hierarchical, and distributed are used in man-made systems. Source: [44].

### 2.3.2 Taxonomies of Swarm Robotics Systems

Before discussing current swarm robotics systems design methods, it is prudent to review existing swarm system taxonomies. Taxonomies provide a common framework for naming and classifying characteristics and components of a system. This is particularly useful for swarm systems, which draw researchers who use diverse nomenclature from various academic fields. A UAV swarm taxonomy should describe and classify a swarm system using standardized nomenclature. Some existing swarm taxonomies focus on physical or functional architectures and levels of automation, while others characterize swarm systems based on problems or tasks [52].

Dudek et al. developed a formative taxonomy of swarm robotics based on seven different design variables: swarm size, swarm range, communication topology, communication bandwidth, collective reconfigurability, processing ability, and collective composition [53]. While Dudek's taxonomy provides an organized and useful collection of design parameters for building a physical system, it does not provide the mission-oriented insights necessary for designing a system architecture specific to military swarm operations.

Cao et al. [39] chose to categorize swarm robotics using five research axes related to cooperative behavior: group architecture, resource conflicts, origin of cooperation, geometric problems, and learning. He proposed that cooperation is an underlying fundamental behavior that enables task performing interfaces. Group architecture describes the composition (heterogeneous vs. homogeneous) and communication structure of the swarm. Resource conflicts delineate the rule set or priorities the agents use to communicate with each other to avoid collisions and manipulate targets in the environment. Origins of cooperation characterizes how the cooperative behavior is achieved whether through biologically inspired methods or game theory, and geometric problems describe spatial aspects such as path planning and pattern generation. Finally, learning via adaptability and flexibility is a key to realizing collective behavior.

Brambilla et al. classified and categorized swarm engineering publications into two taxonomies: the methods used to design and analyze swarm systems, and the collective behaviors studied (Figure 2.2).

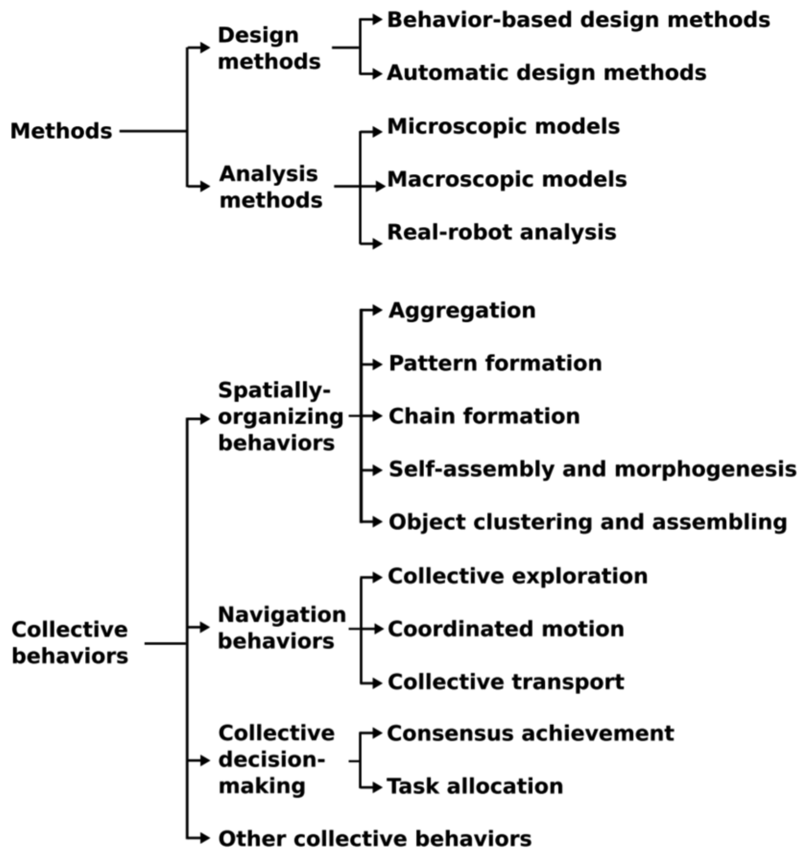



Figure 2.2. Swarm engineering research classified by Brambilla et al. under two major categories: design and analysis methods and collective behaviors studied. Source: [54].

In their review of design methods, Brambilla et al. recognized that there is not a standardized method for designing individual agent behavior to create the desired swarm collective behavior; rather, the design is influenced by the discernment of the designer [54].

### 2.3.3 Bottom-up Swarm System Design

Bottom-up modeling approaches focus on assembling sub-components of systems to build more complex systems. Agent-based modeling, FSMs, and Petri Nets are bottom-up modeling methods frequently used in swarm system design. Bottom-up models are advantageous from a modularity and composability perspective, but they often risk failing to meet higher level system requirements if design begins before a higher level system architecture is established. For that reason, combining bottom-up and top-down models is a software development heuristic [55].

### Agent-based Models

Modeling agent-level interactions can provide valuable information regarding the emergent behavior inherent to CAS. Agent-based modeling is a commonly used approach for modeling a CAS as a group of autonomous agents who make decisions individually based on their assessment of the environment and in accordance with a rule set [56], [57]. McCune et al. [46] used agent-based modeling to investigate C2 of UAV swarms, Bonabeau simulated human systems using agent-based modeling methods, and Munoz [58] studied defensive UAV swarm employment using agent-based modeling. Agent-based modeling is considered a bottom-up approach due to its focus on agent-level interactions. These agent-level interactions can provide valuable information regarding the emergent behavior inherent to CAS. Maier describes emergent behavior as functions that are performed that "do not reside in any component system" [59]. Emergent behavior is also called "collective intelligent" behavior by researchers in the robotics community [53]. The emergent behavior of a swarm system is a critical attribute to consider when designing a swarm system, developing swarm tactics, or devising an assessment methodology for a swarm system. While agent-based modeling can provide useful information regarding interactions within a system, the lack of standardization in approach and variety in agent definitions make model verification difficult [57].

### Finite State Machines

Finite state machines (or finite state automata) have been used to model multi-vehicle autonomous, unmanned system architectures [60]–[63]. Within a FSM architecture, each agent operates within one of several defined states at a given time. The trigger events that cause the agent to transition between states are precipitated by environmental conditions it senses or events it encounters. This type of structure is applicable in developing military swarm systems as the states and triggers can be defined deterministically, which is necessary for high-risk mission events such as target strikes. Weiskopf et al. demonstrated a control architecture based on a library of basic tasks to support the search and track reference mission [63]. The FSM for the reference mission, which was loaded into all of the UAVs, contained

16 states and 28 possible state transitions. Their Mission Control Software demonstrated a sensor architecture allowing small UAVs to operate cooperatively and autonomously [63]. Figure 2.3 is an example from part of an actual FSM for a UAV conducting a search task. The system starts in Global Search and operates in one of four states according to six different conditions that trigger the transitions.

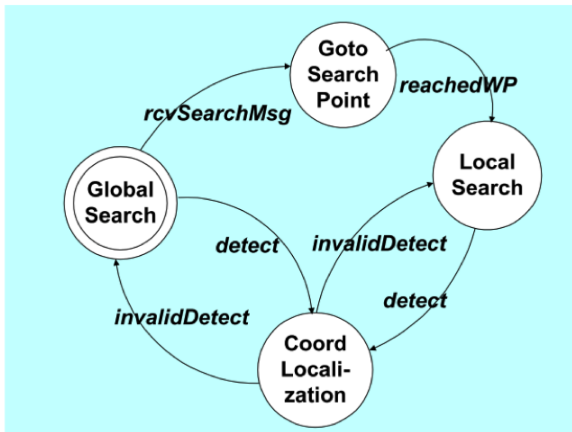



Figure 2.3. Example of a UAV's control architecture depicted as an FSM. Source: [63].

Conversely, there may be other mission events, such as covert searching, in which some bounded degree of unpredictability is desired. In those cases, a probabilistic finite state machine (PFSM), or probabilistic finite state automata, can be used to allow for different behaviors within a state or by allowing multiple transitions between states [62]. Using this architecture, the transition probabilities can remain fixed or change over time [54]. Task allocation and aggregation behaviors have also been accomplished using PFSM approaches [60].

**Petri Nets**

Petri Nets are a well-established process modeling method that have been used to model and analyze system and business processes. A Petri Net is a bi-partite graph consisting of places and transitions as the two types of nodes [64], [65]. An advantage of Petri Nets is their formal semantics; a Petri Net is represented mathematically by a tuple consisting of finite sets of: places, transitions, arcs from places to transitions, arcs from transitions to places, and an integer vector marking the current position [65]. Petri Nets are useful for modeling system interactions that are well known. Palamara et al. [66] demonstrated a Petri Net

Plans framework to model high-level multi-agent behavior. Hamadi and Benatallah [67] used Petri Nets to model internet web services. Rao et al. [68] modeled a complex network centric system, and Levis and Wagenhals [69] developed a methodology for command, control, communications, computers, intelligence, surveillance, and reconnaissance (C4ISR) architecture design using Colored Petri Nets. Zhu and Brooks [65] compared Petri Nets and FSMs for developing a discrete event controller for a distributed surveillance network. They found both approaches to be roughly equivalent, with each having an advantage. The FSMs performed well for smaller scale systems but proved cost prohibitive (computationally) for complex systems [65]. The Petri Nets provided a more compact state space and were more suitable for modeling systems with repeated structure [65]. Traditional Petri Nets lose efficiency as the number of states increases; however, this limitation can be overcome using Colored Petri Nets, which enable additional routing behavior and more compact models [70].

**Behavior-based design**

Behavior-based design, in which the individual behavior of each agent is developed iteratively until the desired swarm behavior is acquired, is a typical swarm system design method. This bottom-up development method is counter to traditional top-down systems engineering design. The term *behavior* is commonly used in robotics literature to describe the actions being performed by robots or agents. Behaviors apply to individuals and environments as well as to groups, often called *collective behaviors* [54]. Behaviors may also be categorized as higher-level abstract behaviors and lower-level primitive behaviors, or simply *primitives* [71]. The term “primitives” is borrowed from the computer science discipline and functions similarly in robotics literature; they act as building blocks for programming higher level functions. This rationale is directly applicable to the proposed UAV swarm architecture model wherein tactics are composed of plays.

A seminal behavior-based design is Brooks’ subsumption architecture, which uses a layering approach for controlling systems, and incorporates augmented FSM processors for managing inputs and outputs [72]. A key contribution of Brooks’ work was that he decomposed the robot control problem into behaviors rather than into functional modules. Furthermore, his hierarchy of layered behaviors allows higher levels of behaviors to subsume lower layers of less complex behavior. Parker’s ALLIANCE is a distributed control, fault-tolerant behavior-based architecture that allows agents to perform multi-level behavior sets according to their current status [73]. Nicolescu and Matarić presented a behavior-based hierarchical architecture for robots in which reusable primitive behaviors support a library of abstract behaviors [71]. Brambilla et al. decomposed collective behaviors into four categories: 1) spatially-organizing, 2) navigation, 3) collective decision-making, and 4) other collective behaviors such as fault detection and human-swarm interaction [54]. Spatially organizing and distributing agents within their environment is a critical function for avoiding collisions and enabling efficient use of sensors. The most fundamental swarm organizing behavior is frequently called aggregation in swarm robotics literature. From aggregation, more complex patterns and formations such as flocking can be assembled. *Consensus* is a typical method for achieving a collective decision amongst the swarm [4], [74], [75]. The proposed MASC framework presents an approach similar to Nicolescu and Matarić's, but inspired from a naval doctrinal perspective, that may provide applications across the gamut of prospective military swarm operations to enhance the mission suitability of future swarm systems.

Virtual physics-based design, in which each agent is deemed a virtual particle that applies "forces" on other agents, includes the artificial potential fields model which is based on the theory of the electromagnetic potential field. When designing a behavior with artificial potential fields, the attractive forces are placed around the goal or objective and the repulsive forces are positioned at obstacles [54], [76]. Some undesirable effects of artificial potential fields include stagnation at local minima and unstable agent oscillations. Virtual physics-based design is discussed further in swarm algorithms (Section 2.3.8).

Automatic design methods focus on design at the collective level rather than at the agent level. The most prevalent types of automatic design methods are reinforcement learning and evolutionary robotics [60], [75]. Challenges with these methods include difficult property verification, and computational limitations with scalability as the number of possible system states grows exponentially with the number of agents [54]. Evolutionary algorithms are further discussed in swarm algorithms (Section 2.3.8).

### 2.3.4 Top-Down Swarm System Design

As previously mentioned, most swarm systems have been developed using bottom-up development methods in which an individual agent's behavior is iteratively fine-tuned until the desired collective behavior is achieved, commonly called "code and fix." DeLoach et al. [77] developed the Multi-agent Systems Engineering methodology for analyzing, designing, and producing heterogeneous multi-agent systems. Their method uses graphically-based models to define the agents and interfaces. The seven-step process is shown in Figure 2.4.

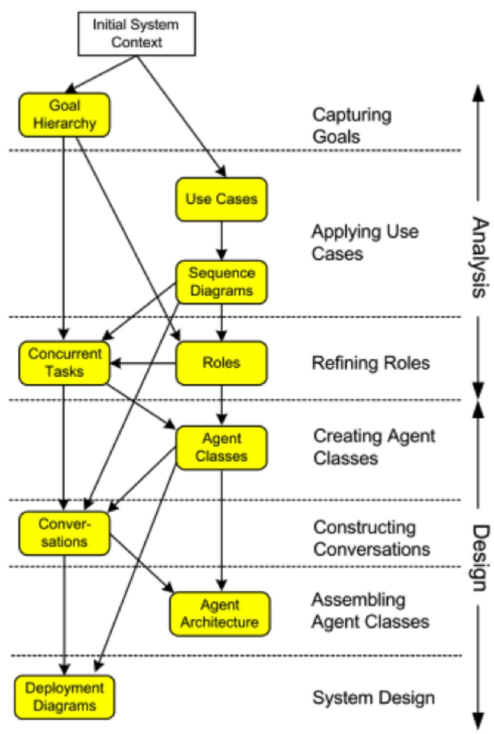



Figure 2.4. The Multi-agent Systems Engineering (MASE) methodology for analyzing, designing, and producing heterogeneous multi-agent systems. Source: [77].

Brambilla et al. [75] proposed a property-driven, top-down design method that formally describes the features of the system the designer wants to realize. Their method has four phases:

1. Phase one: formally state system requirements by specifying the intended properties;
2. Phase two: create an abstract macroscopic model and model checker to verify the properties;

3. Phase three: use the macroscopic model as a guide for implementing the system (macroscopic to microscopic transition); and
4. Phase four: test the system using real robots.

The top-down design method proposed by Brambilla et al. [75] compels further exploration of current swarm system design research with respect to requirements development, modeling methods, and field experimentation.

### 2.3.5 Swarm System Requirements Development

Developing design requirements for new technology is an incremental, iterative, and drawn out process in traditional systems engineering methodology. First, a conceptual design is created from a concept of operations (CONOPS) by analyzing user needs and requirements. The CONOPS should describe how the system will be used, provide a statement of mission requirements, and include a collection of scenarios describing how the user will operate the system [55]. Following the conceptual design, a preliminary design assigns physical architectures to different alternatives which may result in the construction of prototypes. Eventually, after many iterations of refinements, a full-scale, detailed design is developed. Day et al. [78] identified agile software methods, automatic software testing, and continuous integration techniques as best practices for accomplishing accelerated development of multi-UAV systems. By using agile methods to reduce the scope of requirements analysis into smaller components to keep abreast with rapidly evolving requirements, developers are able to launch software revisions at shorter cycle times and reduce overall program risk [78], [79]. In general, the UAV swarm must be able to avoid collisions (both inter-agent and with the environment), aggregate, communicate, and operate within its flight envelope. Once those basic functions are met, mission tasks such as navigation, mission planning, surveillance, decision-making, sensor management, and coordination must be accomplished.

### 2.3.6 Modeling Swarm Systems

Swarm robotics systems are typically modeled at the microscopic (individual agent) level and the macroscopic (collective) level. Microscopic models account for the interactions between agents and with the environment, while macroscopic models assess the entire swarm as a system [54]. Rate equations have been particularly useful for modeling swarm behavior because they can enable translation of microscopic models into macroscopic models [61], [75]. Other researchers may model swarms with differential equations or classical control and stability theory such as Lyapunov stability theory, to analyze swarm formation properties [54], [80]. Throughout the research surveyed, swarm models are developed at varying levels of abstraction depending on the focus of the analysis. Boskovic et al. presented a four-layer autonomous, hierarchical, intelligent control architecture for UAVs in which the layers are connected by the Achievable Dynamic Performance (ADP) calculation module, allowing the system to continue to perform despite failures, albeit in degraded modes [81]. A notable contribution of this architecture is the level four decision-making layer which uses the ADP and sensor information to make trade-off decisions between mission success and the survivability of the vehicles [81]. Kolling et al. conducted a thorough review of swarm models in [82]. The focus of the MASC research is to model the swarm at the macroscopic level, emphasizing the tactics level of abstraction.

### 2.3.7 Swarm System Verification and Validation

Lightweight formal methods have been used in software development for the past few decades as a means for early error detection [83], [84]. The methodical, mathematical techniques and abstraction, which are innate to formal modeling, enable complexity to be reduced and facilitate instantiation of precise system specifications. For these reasons, formal methods were first applied to requirements engineering in which specifications were dominated by natural language. Eastbrook et al. describe three case studies in which lightweight formal methods were applied to requirements modeling for fault protection software requirements on National Aeronautics and Space Administration (NASA) systems [83]. In these case studies, lightweight formal methods were applied selectively to the most critical requirements, resulting in the discovery of errors not detected using traceability analysis or inspection. The deficiencies were then corrected during the development phase, when changes are more easily managed and less costly [83]. Other examples of lightweight formal methods in system requirements development include IBM's Customer Information Control System in the 1980s, a new display information system for the UK Civil Aviation Authority's air traffic management system, and a requirements specification for the Traffic Collision Avoidance System developed by the Safety-Critical Systems Research Group, both in the early 1990s [85]. Lightweight formal methods offer advantages for CAS modeling by specifying certain behavior sets for agents and providing a means for determining all possible state combinations within a system, also called "state space methods" [86].

Lightweight formal methods' role in software architecture design permits the architect to focus on specific, critical components of the system by using abstraction [87]. Abstraction allows designers to develop a conceptual model that can be used to capture interactions between components before the component details have been specified. This application is pertinent to UAV swarm modeling, in which the operational processes can be modeled at the tactics level, allowing the architect to focus on what tactics should be allowed under a prescribed set of conditions. Once the sequence of allowable tactics has been established, the architect can proceed to sorting out the plays at the next lower level. For example, Revill identified an unexpected output while modeling UAV swarm failure modes when a UAV performing a search and track mission terminated a return-to-base mode to start tracking a new target [88], [89]. While this behavior may be desirable in certain operational scenarios in which the agent is considered expendable, it requires an additional step for the operator to approve tracking targets found while in the return-to-base condition.

Monterey Phoenix (MP) supports step-wise system architecture design by using formal methods to define behavior models. MP is an NPS-developed formal language and method for modeling system behaviors and operational processes in systems architecting [90]. MP enables behavior specification for each agent or system component, separate from the interactions among the agents. This separation between the agents and their interactions enables re-use of the models for different scenarios. Events are characterized in terms of inclusion or precedence relationships, while behaviors may be depicted as alternative ("or" statements), concurrent, iterative (loops), or optional actions. A benefit of MP over other modeling language is that it generates an exhaustive set of use cases that are visually depicted [91]; this is particularly useful for CAS modeling and dealing with the state explosion problem. However, even a relatively simple system's state space exceeds a human's ability to extrapolate every possible combination of outcomes. Jackson's small scope hypothesis proposes that most errors in models can be discovered in a comparatively small number of counter examples [21]. By generating a complete set of use cases, MP enables a viable method for identifying unwanted behavior and validating system behavior models for CAS [31]. It also has the potential to augment and support other types of top-down behavior models such as activity diagrams and state machine diagrams.

**Swarm System Field Experimentation**

It is not possible to model and simulate all facets of reality, and thus the modeling and simulation data should be validated with real robots. The ARSENL at NPS has demonstrated a large-scale, fixed-wing, outdoor, UAV swarm capability [2], [3], [78]. Other notable fixed-wing, multi-UAV field experimentation programs include:

- DARPA's recent Service Academies Swarm Challenge [16] demonstrated innovation in defensive and offensive swarm tactics using a mix of fixed-wing and quad-rotor UAVs in a capture-the-flag type scenario [16], [92], [93]. The event culminated in a 30 vs. 30 UAV swarm [16]. The author participated in this event as a swarm mission commander.
- The University of Colorado, Boulder's work with heterogeneous aerial robotic networks that used the mothership/daughtership concept with swarms of micro UAVs [94]. Their five-level heterogeneous unmanned aircraft system (HUAS) is a bottom-up layered approach consisting of (from highest to lowest level): 1) cooperative algorithms, 2) application layer communication protocols, 3) sensor, communication and control fusion, 4) data routing and network configuration, and 5) transport. This approach is similar to Brooks' subsumption architecture [72] and focuses on the network aspect of a swarm system.
- The University of California, Berkeley developed a software architecture to support multiple mission sets for a heterogeneous UAV swarm [95]. They identified the need for a software architecture built to the level of abstraction necessary to support rapid growth of modular software. Their hierarchical approach separates low-level control from high-level planning processes onto different hardware, improving system fault tolerance [95]. This approach is similar to ARSENL's in which the Pixhawk autopilot governs low-level navigation, control, and guidance while the Hardkernel ODroid U3 autonomy payload controls high-level coordination tasks [3].
- The Apollo system at the University of Porto used three layers in its heterogeneous vehicle control architecture - mission supervision, vehicle supervision, and maneuver control - to support multi-UAV coordinated missions [96]. Like many other multi-UAV projects [3], [34], [95], [97], [98], it uses a "publish-subscribe" communication framework in which nodes coordinate based on message topic commonality between the publisher (message issuer) and the subscriber (message receiver). This framework

allows for a more dynamic network topology and enables scalability, both important design factors in swarm systems.

- Finally, Stanford University's DragonFly project was an early and influential multi-UAV test bed in terms of identifying the importance of integrating the control algorithms and the software architecture into an integrated modular architecture [99]. Unlike more recent test beds, DragonFly used a server-client communication architecture that led them to use shared memory to coordinate message scheduling. Their effective integration of commercial-off-the-shelf (COTS) products paved the way for later test beds.

Many of the challenges discovered such as human-swarm interaction issues and logistical constraints discovered during field experimentation would not have been fully realized in simulation alone. Almeida et al. articulated this insight in [96]: "The major challenges come from the distributed nature of these systems and from the human factors. This is why we need to couple the development of scientific frameworks with field tests with human operators."

### 2.3.8 Toward a Mission-based Composable Architecture

A key advantage of swarm technology is that the swarm is designed to be composed of simple, modular, identical components [38]. These homogeneous agents are not programmed for a specific role and do not operate under a centralized coordinating agent. Accordingly, the loss of an individual does not cause a significant decrease in system performance because another agent or agents can assume its duties. Thus, a homogeneous swarm can be adaptable, expendable, robust, and scalable [25], [38]. Bachrach et al. [100] eloquently underscored the need for a composable swarm architecture:

> The dream is that using a high-level language to program a multi-robot application, a programmer would be able to succinctly implement robust group behavior primitives and to quickly compose new programs out of existing primitives and simpler programs. In current practice, however, a programmer typically specifies the behavior of individual robots and attempts to show that their interactions will produce the desired aggregate behavior.

**Playbooks**

The idea of using a *playbook*, or collection of pre-defined tactics or action plans for multi-robot unmanned systems is not new, and has been used in both UAVs [19] and unmanned ground vehicles (UGVs) [18]. A playbook can be described as a "library of plans of action that are available for the operator to instantiate at various levels of detail, allowing various levels of autonomy for the agents" [20]. Rather than controlling each sequence of actions, the catalog of pre-defined behaviors simplifies user control, reduces necessary communications, and synchronizes agent tasking [20]. Sheard et al. emphasize that "[complex system] structure cannot be described at a single level or with a single view; multi-scale description are needed to understand complex systems" [101]. Playbooks offer a means to organize complex systems into more tractable components.

Squire et al. studied the effects of various multi-vehicle control architectures on human performance [10]. Experiments by Parasuraman et al. used a simplified version of the "Playbook" interface in the RoboFlag multi-vehicle simulation environment [102]. The simplified RoboFlag Playbook used a hierarchical task model to task robots to perform various functions under temporal and conditional constraints. The architectures were categorized in three automation configurations: manual, automated, or a flexible configuration in which operators used "manual control and automated plays" [10]. The authors evaluated the effects of these three architectures on human performance under defensive, offensive, and mixed conditions using subjective measures to rate mental workload and situational awareness, in addition to performance metrics to capture mission completion time and mission success rate. The results of this study suggest that a playbook type architecture provides operators with the requisite flexibility of balancing manual control and cognitive workload when managing unforeseeable conditions [10]. Furthermore, the playbook system architecture looks to be promising for overcoming the "single robot parenting" paradigm.

The "STP" (skills, tactics, and plays) multi-robot architecture developed by Browning et al. is particularly applicable to this research because it was designed to control a team of autonomous robots in an *adversarial* environment: RoboCup robot soccer [103]. Though MASC's architecture terms sound similar to [103], they are different in terms of hierarchy and function. STP's elements are categorized temporally. Browning et al. define *skills* as "encoded low-level single-robot control algorithms for executing a complex behavior to achieve a short-term objective" (300ms–5sec time period), *tactics* delineate which skills the individual robot should execute to achieve a specific goal (1–30sec), while *plays* determine "how the team of robots should coordinate their execution of tactics in order to achieve the team's overall goals" (5–30sec) [103]. Tactics, which include a set of acceptable parameters which are dependent upon the play, determine which state machine (composed of skills) a robot will execute. In this architecture, tactics and skills can be detached from plays to support a hierarchical control structure for operating individual *heterogeneous* robots that perform different roles.

McLurkin developed a library of behaviors for swarm robots, that are designed to be used as building blocks for more complex tasks [104]. The approach focuses on behaviors at the group level to make programming distributed systems easier by developing scalable, reusable behaviors that generate predictable outcomes. His architecture, influenced by Brooks' subsumption architecture [72], consists of a hierarchy of behaviors ranging from highest to lowest level: demos, group, pair, and primitive behaviors [104]. While McLurkin's work focuses on ground robots and behaviors related to navigation, clustering, and dispersion, the philosophy is applicable to MASC.

Goldman et al. [19] addressed the problem of over-constrained missions in which the system cannot meet the requests of the operator, due to temporal, geographic or other constraints. They improved the Smart Information Flow Technology (SIFT) Playbook-enhanced Variable Autonomy Control System (PVACS) project by adding a "best effort" planning mode, using cost-based optimization, to relax constraints and provide viable alternative play options [19]. The capability to automate modifications to plays extends the potential for multi-vehicle unmanned systems to be operated by users who may not be specifically trained for the UAV's capabilities.

In 2017, DARPA announced a program called OFFensive Swarm-Enabled Tactics (OFFSET) to advance swarm technology by focusing on human-swarm teaming and swarm autonomy within a realistic gaming environment [33]. Their method takes a hierarchical approach to the swarm framework which is composed of a mission, tactics, primitives, and supporting algorithms [33]. This program focuses exclusively on the urban operational environment, with a goal of building a playbook of tactics to support uncooperative urban missions. DARPA's research is particularly relevant as it aims to bridge the gap between the operational level of control and the programming solution level.

### Swarm Algorithms

Swarm algorithms are the "under the hood" applications of the swarm behaviors (*plays* in MASC). There is a considerable body of research in robotics algorithms, especially with regard to search and tracking functions. Typically, these optimization problems are treated as local, small-scale sub-spaces of the larger, global (NP-complete) problem to satisfy practical applications [76]. The following section highlights a few algorithms which may be applicable to supporting UAV swarm plays. In general, swarm algorithms need to scale well with the number of agents and adversaries, rely on local rather than global information, and not require loads of memory or continuous, direct communication [76]. Many algorithms applicable for UAV swarms can be characterized as biologically-inspired or evolutionary [105].

Biologically-inspired algorithms include: Reynold's "Boids," ant colony, bee colony, Particle Swarm Optimization (PSO), and pheromone-based algorithms [76]. Flocking algorithms simulate the behavior of a flock of birds in flight and compel each agent to steer itself based on three simple rules [106]:

- separation: avoid other agents,
- alignment: align heading with other agents, and
- cohesion (steer toward center of agents).

By the firefly algorithm, attractiveness is proportional to firefly brightness, so the dimmer agent will move towards the brighter one. If there is no brighter one, the agent moves randomly. Attractiveness variation $\beta$ varies according to distance $r$, and is given by:

$$\beta = \beta_0 e^{-\gamma r^2} \tag{2.1}$$

where $\beta_0$ is the attractiveness when $r = 0$ and $\gamma$ is the absorption coefficient of the light through the medium [50]. Because the attraction decreases with range, the swarm can automatically divide into sub-swarms [76]. Yang further improved the firefly algorithm by combining it with the attributes of Levy flight, forming the Levy-Flight Firefly Algorithm which has shown promise in simulation tests for applications in solving NP-hard problems [50]. This algorithm is applicable to swarm subdivision maneuvers such as the split and join plays.

Similarly, the Glowworm Swarm Optimization (GSO) is modeled after glowworms' use of luminescence for stigmergic communication (much like pheromone-based communication). GSO has been shown effective for tracking multiple mobile targets [76].

The artificial bee colony optimization is based on the foraging behavior of a honeybee colony. Unlike some of the other swarm algorithms mentioned, food sources represent potential solutions, and bees are the agents responsible for finding new food sources [76], [105]. The probability, $p_i$, of a food source being selected is given by:

$$p_i = \frac{f_i}{\sum_{n=1}^{N} f_n} \tag{2.2}$$

where $f_i$ is the fitness value of the food source $i$ and $N$ is the number of food sources [105]. Bee colony optimization is useful for path planning and search functions for UAV swarms.

The PSO algorithm was developed by Eberhart and Kennedy to graphically simulate bird flocking [107]. The model assigns particles with randomly assigned velocities that are flown through the problem space. Particles update according to their own best positions, which may be local or global [76], [107]. The original PSO algorithm was intended for solving global optimization problems, but has been adapted for multi-agent systems by using sub-swarms [76]. A weakness of PSO, getting trapped in local optima, has been minimized with improvements to the algorithm [108], [109]. PSO is useful for solving the same types of problems as evolutionary algorithms (EAs), searching for targets, but it has certain advantages over EAs; most notably, it has memory and retains knowledge of good solutions [107].

Brownian motion and Levy flight are two types of "random walk" algorithms based on biologically inspired optimal foraging theory, in which animals base their foraging strategy on maximizing their nutrient collection based on their exerted effort. Brownian motion step lengths have a scale (usually defined by the mean and variance) while Levy flight step lengths have no characteristic scale and exhibit a heavy tailed probability distribution [110]. In practice, agents following Brownian motion take small random walks within certain area while those demonstrating Levy flight will take longer walks on occasion. Levy flight is particularly useful in improving search algorithms because the random longer jumps may prevent a local optimum from occurring [76]. These algorithms may be applicable to plays that support random searches (wherein efficiency is not the priority) and in cases where an evasive search is warranted to minimize exposure to a potential threat [111].

Consensus-based algorithms are commonly used in decentralized swarm system control to enable convergence to a common solution [74], [112], [113]. “Consensus algorithms are designed to be distributed, assuming only neighbor to neighbor interaction between vehicles” [112]. Distributed sorting algorithms, a type of consensus algorithm in which agents “self-organize according to criteria derived from individual UAV state characteristics,” can be used to assign behaviors to the swarm [4]. For example, the swarm can be sorted by health status in order to land the agents with the lowest fuel status or battery status first. Alternatively, agents can be sorted for target tasking based on proximity to a search area. Davis et al. have conducted promising UAV swarm field experimentation on two consensus-based algorithms designed to tolerate the unreliable communication present in outdoor operations [4].

Physicomimetic or potential field algorithms model agents and obstacles as carrying the same “charge” with targets carrying the opposite charge, creating an artificial potential field. Collisions are avoided by repulsive forces between agents or between agents and obstacles, while attractive forces between opposite charges draw the agent toward the target [76], [114]. This approach has performed well in scenarios in which agents are outnumbered by targets [115] and it is often used in collective behaviors that require a pattern formation [54].

Evolutionary algorithms are used to approximate solutions to difficult optimization challenges, which relates to path planning and target searches for UAV swarms. The term EA covers genetic programming, genetic algorithms, evolution strategy, and evolutionary programming [105]. Of these, genetic algorithms are the most widely used in swarm intelligence algorithms [51], [105], [116]. Multi-objective EAs have been used effectively in multi-robot systems to provide a decision maker with potential solutions based on different optimization levels of specified parameters such as distance traveled, cost, or hazard level [117]. [118] provides a methodology to compare the performance of different EAs.

Choosing algorithms for multi-robot system task allocation and coordination typically involves compromise between efficiency and optimality of solutions. While consensus algorithms may consistently converge on a solution, they may require substantial time and heavy data transmission. The auction algorithm [119], in which agents place “bids” on tasks and the highest bid wins the task, has been shown to produce efficient sub-optimal solutions. Robotics researchers have modified traditional centralized auction algorithms by developing decentralized auction and market-based algorithms more suitable for the distributed architectures of multi-robot systems [2], [52], [74], [120]. The market-based algorithms shift to a different economic paradigm from the self-interest bidder to the merchant who adjusts prices to execute tasks faster. Choi et al. combined these two approaches into decentralized task-allocation algorithms that produce conflict-free solutions despite variances in local situational awareness data [74].

Early swarm robotic system design has been dominated by bottom-up and ad-hoc modeling strategies such as agent based modeling and FSMs. The foundational swarm algorithms such as Boid's, ant colony, and PSO, along with the iterative improvements to these algorithms is critical for swarm system architecture design from the technical perspective; they are the building blocks for higher-level behaviors. These techniques have merit in accomplishing small-scale control and sensor collection goals but impede the transition of swarm technology from labs and field experiments to an operational military environment. Combining these bottom-up methods with top-down strategies such as lightweight formal methods and traditional MBSE models provides a means for extending swarm systems into militarily useful scenarios.

## 2.4 Factors Influencing Swarm Design

There has been significant work done in the technical aspects of swarm design, particularly in biologically-inspired behavior-based modeling. Equally as important as the technical aspects of UAV swarm design are influential factors such as swarm doctrine, communication architecture, and the human-swarm interaction. Swarm doctrine is surveyed from a historical view of military swarming strategies throughout various conflicts. Communication architectures are reviewed, as they impact both the doctrinal and technical aspects of UAV swarm architecture. Similarly, the human component must not be ignored as a major contributor to swarm design. The sheer number of agents in a swarm demands a shift from the customary role of a pilot directly controlling the system toward a mission or system manager.

### 2.4.1 Doctrine, Tactics, and Strategy

Before discussing swarming doctrine, it is important to define doctrine, tactics, and strategy, as these related but distinct terms are often misused. Military doctrine is a guide for how the military should conduct major campaigns and operations, and it applies at both the strategic and tactical levels. It provides a standardized conceptual framework for connecting strategy, operations, and tactics, and is influenced by technology, the enemy's capabilities, organizational structure, and geography. Doctrine shapes how missions should be accomplished in terms of roles, functions, and tasks. It describes how the military trains, organizes, plans, and what it buys [121]. NATO defines doctrine as "fundamental principles by which the military forces guide their actions in support of objectives. It is authoritative but requires judgment in application" [122]. Hughes defines doctrine from a naval perspective, that is primarily directed towards the operational warfare level, as "policies and procedures followed by forces to assist in collective action, either strategic or tactical" and "includes battle plans and practices for the immediate application of force" [123]. Doctrine development is guided by past experience, current CONOPS, and experimentation using modeling and simulation, war-gaming, and field exercises. Doctrine is an important element of *mission engineering*, "the deliberate planning, analyzing, organizing, and integrating of current and emerging operation and system capabilities to achieve desired warfighting mission effects" [8], [124].

Strategy outlines the broader objectives of the military operation, and comprises the plans and policies that govern military actions. The purpose of strategy is to "affect the outcomes of wars, or campaigns, of tactics, the outcomes of battles or engagements" [123]. Strategy and tactics must work in concert, as Sun Tzu famously said, "Strategy without tactics is the slowest route to victory. Tactics without strategy is the noise before defeat" [125].

Tactics are the detailed means used to achieve a military objective. They describe the "handling of forces in combat" and the "maneuver and application of combat power" [123]. VADM Arthur Cebrowski, USN, described tactics as "the sum of the art and science of the actual application of combat power" [123]. In *Fleet Tactics*, Hughes asserts that naval tactics are built on five propositions: 1) delivering firepower successfully, 2) reconnoitering (he uses the term "scouting") to locate the enemy, 3) exercising C2 for transforming reconnaissance and firepower into force implementation, 4) attacking effectively first, and 5) using properly timed maneuver as a tactical process [123]. These five precepts are pertinent to developing a swarming doctrine that will capitalize on the inherent strengths swarm systems can bring to the battlefield.

### 2.4.2 Swarming Doctrine

Military historians have used the term “swarming” to describe one of the four general engagement patterns for military land, sea, and air operations: melees, massing, maneuver, and swarming [126]. Disordered melees were characterized by individuals fighting on their own, massing involved mainly fixed, controlled, inflexible formations, and eventually maneuver patterns offered the most flexibility by increasing mobility and combining it with force massing [5]. Swarming combines the advantages of decentralization in melees with the more mobile maneuver to enable large numbers of individual agents to battle collectively [5]. Swarming has also been categorized as one of three major sources of non-linear warfare, with guerrilla and maneuver warfare the other two forms [127]. Edwards characterizes swarming as “several units conduct[ing] a convergent attack on a target from multiple axes” [127]. From a network-centric warfare perspective, swarming doctrine has been described as “an offensive action generated in pulses by highly dispersed forces that do not employ traditional hierarchical command and control structures” [128]. This progression of engagement patterns was enabled by the extent and efficiency of information processing, and each engagement pattern has been built upon foundations of the earlier patterns [129].

Looking through this lens, many conflicts throughout history could be characterized as swarming warfare: the British versus the Spanish Armada in 1588, the British against the swarming German U-boat wolf packs in the North Atlantic, the British Fighter Command exercising defensive swarming against the German Luftwaffe, the Japanese kamikaze attacks against the U.S. Navy, the U.S. military in the Battle of Mogadishu, typical operations of non-governmental organizations, and Al Qaeda’s strikes on multiple U.S. targets on September 11, 2001 [128], [129]. Edwards provides a comprehensive study of swarming in military battles from ancient (the Battle of Alexandria Eschate in 329 BC) to recent times (Operation Iraqi Freedom in 2003) [127]. In his analysis of 23 case studies, he identified “encirclement, elusiveness, superior situational awareness, standoff, and simultaneity” as the five leading factors that most influenced swarming’s outcome in a conflict [127]. Edwards generalized historical swarming examples into two types: massed or cloud swarms, and dispersed or vapor swarms [127]. The cloud swarm reaches the battle en masse and divides to execute a double envelopment, while the vapor swarm arrives as dispersed sub-units that converge upon the adversary from all directions as shown in Figure 2.5 [127]. While the examples cited involved manned assets in each case, the tactical patterns of dispersion and convergence are applicable to robotic swarms.

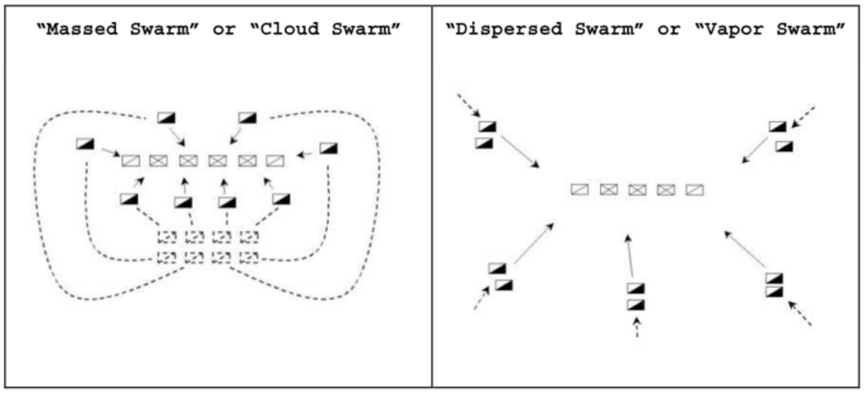



Figure 2.5. The two types of swarming in historical military operations: massed or cloud swarm and dispersed or vapor swarm. Source: [127].

This historical characterization of swarming as pulsed attacks from traditional units under a decentralized command structure is likely different from what the future UAV swarm doctrine will look like. In the historical cases, while each individual unit (whether a submarine or an Al Qaeda operative) operated somewhat autonomously using a decentralized command and control structure, each was commanded by an individual human. They did not exhibit true local communication and sensing capabilities; they were not exhibiting cooperative behavior, and the individual units exhibited too much variation in behavior to be considered homogeneous. The future modern swarming doctrine should expansively cover operations using agents to perform missions with much less human supervision than previously seen in historical military swarming examples.

What will the modern swarming doctrine look like? Swarming differs greatly from the current hierarchical, sequential, and reductionist structure used by most military organizations in which top-down management directs collective behavior [130]. Future swarming doctrine may include a centralized strategy but focus on more widely-distributed, smaller units executing pulse-like tactics in which units repeatedly converge and disperse to employ mass as a decisive factor in battle [5]. As a result, the organizational structure will be flatter than a traditional military organization's hierarchy, and there will be a transition from "few and large" forces to "many and small" units [131]. As highlighted in [129], militaries looking to use swarming capabilities will need to consider close-in strategies after decades of primarily using standoff strategies shaped by precision-guided munitions. Otto Heilbrunn's concept of "concentric dispersion," in which small groups of forces are amassed together to make quick strikes before dispersing is applicable to swarm tactics which will involve continuous changes in unit size over the course of a mission [129]. Potential acquisition benefits of this "disaggregation of combat power into a larger number of less exquisite systems" are a more diverse and resilient arsenal, lower technology risk and reduced life cycle costs [5].

Many of our current military challenges such as the Chinese activity in the South China Sea and Russian operation in the Ukraine are not traditional wars, but rather "gray space" operations [132], [133]. These types of "in between" operations do not play to the traditional strengths of the U.S. military and lie somewhere in phase I to phase II region of the notional operation plan phases, Figure 2.6.

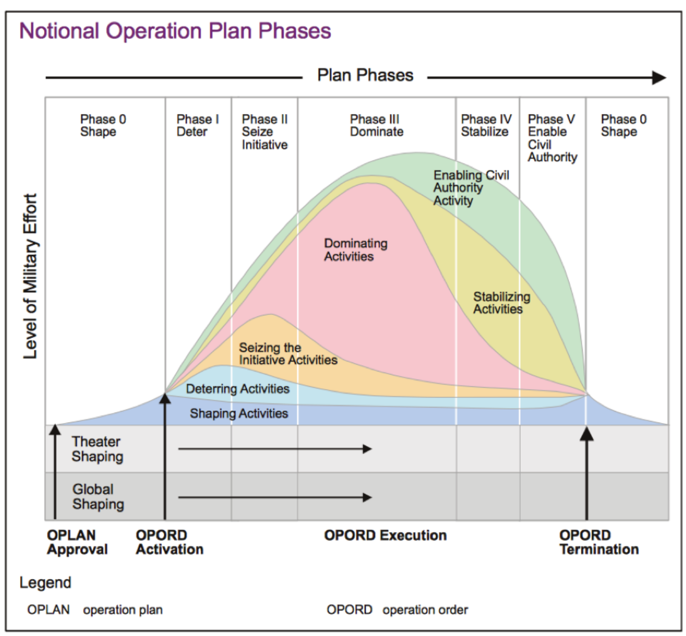



Figure 2.6. Progression of joint operations activities from beginning to end. Source: [134].

In [133], Chairman of the Joint Chiefs, General Joseph Dunford, highlights the need to develop more effective methods for dealing with scenarios "with a military dimension short of a Phase 3 or traditional conflict." UAV swarms should be considered key players in these gray spaces for their expendability, scalability, and capability to reduce risk to humans. Furthermore, their robustness and non-deterministic behavior make them compatible with missions that involve wide-area search and surveillance (especially when there is minimal cueing data), widely distributed attacks, diversion tactics, and suppression of enemy attacks [40].

Future wars are expected to be characterized by "astute use of communications, cyberspace, and technology, such that their impact extends regionally and globally" [135]. The Navy's highly connected platforms and network-centric warfare strategy has made the fleet reliant on continuous communication. The Chief of Naval Operations calls the global information system one of the three major influential forces in the maritime environment [136]. "Astute use of communication" may include the use of emissions control (EMCON) as a tactic to the electromagnetic spectrum. Stefanus contends that future maritime warfare will include conflicts in which the U.S. Navy is denied use of the entire electromagnetic spectrum, and in turn must develop ship tactics for this "Dark Battle" [137]. This electromagnetic spectrum-denied environment scenario also necessitates a reevaluation of current unmanned system doctrine and supporting communication architectures which are heavily dependent on reliable networked communication.

### 2.4.3 Communication Architecture

The UAV swarm network presents important communication advantages in terms of survivability and reliability due to redundancy, and is a key consideration in developing a system architecture, as it enables the effective and efficient collaboration and cooperation of the UAVs. While the UAV swarm communication architecture is not a focus of this research, it warrants discussion because of its impact from both a doctrinal and technical perspective on the overall system architecture. Furthermore, there is a considerable amount of UAV swarm architecture research focused on the communication network perspective. With the emphasis on robust, reliable communication for high performing swarm systems comes a trade-off between broad connectivity and discontinuities in the operational arena causing loss of system control [129]. Challenges in managing information flow, and protecting the information flow against enemy attacks, have technical and doctrinal implications. The flatter and more decentralized command structure of swarm warfare will require more tactical decisions to be made at the junior officer and non-commissioned officer (NCO) levels to take full advantage of the convening and dispersing nature of swarm tactics. Similarly, the end-state application or mission needs to be considered when designing the network. The following section surveys UAV swarm design challenges and current swarm communication network test beds.

Designing a multi-UAV communication network is challenging due to the fluid nature of swarm network topology, in which the number of nodes and links change along with their relative positions. UAV swarm communication networks must be designed for scalability, latency, and bandwidth constraints, along with the weight and space constraints of the individual UAVs. Ad-hoc networks have shown promise for mobile and vehicular networks in which nodes frequently join and fall out from the network. However, research in applying mobile ad-hoc networks (MANETs) and vehicular ad-hoc networks (VANETs) to the single UAV problem have resulted in limited application due to the unique requirements of a UAV swarm network [47]. While MANETs and VANETs are designed to support the slower movement of personnel and ground vehicles, a multi-UAV network must be able to sustain higher mobility in three dimensions, with consideration for UAV energy constraints. The network must support coordination and control between UAVs for effective task planning and be able to adjust when nodes (UAVs) transition in and out of the network [47].

Current research trends indicate flying ad-hoc networks (FANETs) to be a promising lightweight solution for payload constrained unmanned aerial system (UAS). The term FANET has been used to describe a type of MANET in which the UAVs are the nodes, allowing the system to maintain connectivity through neighbor UAVs rather than relying solely on UAV-to-infrastructure links [98]. Due to the higher speeds of multi-UAV systems, FANETs require accurate localization data with less latency than MANETs. Figure 2.7 shows the basic differences in architectures for MANETs, VANETs, and FANETs.

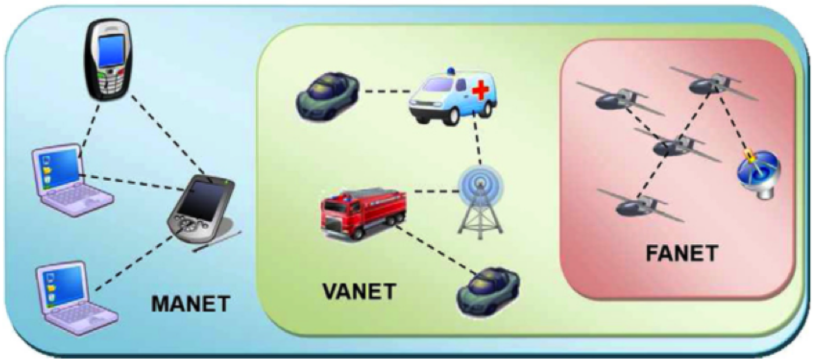



Figure 2.7. Three different types of ad-hoc communication architectures: mobile ad-hoc networks (MANET), vehicular ad-hoc networks (VANET), and flying ad-hoc (FANET). Source: [98].

Within multi-UAV system FANETs, there are several different communication architectures being used. Common FANET topologies are star, multi-star, mesh, and hierarchical mesh [47]. In star topologies, UAVs rely on direct communication with a ground node which results in high latency because the downlink is longer than the inter-UAV distances [47]. Mesh networks are characterized by nodes that act as relays to forward data, and communication between a UAV and a ground station may take place over multiple hops via intermediate nodes [34]. In comparison to star networks, mesh networks provide more reliable and flexible communication because they can transmit data packets across multiple links. They are also desirable for their self-reorganization capabilities which enable them a degree of resilience when nodes fail [47]. Figure 2.8 shows the star, multi-star, flat mesh, and hierarchical mesh topologies.

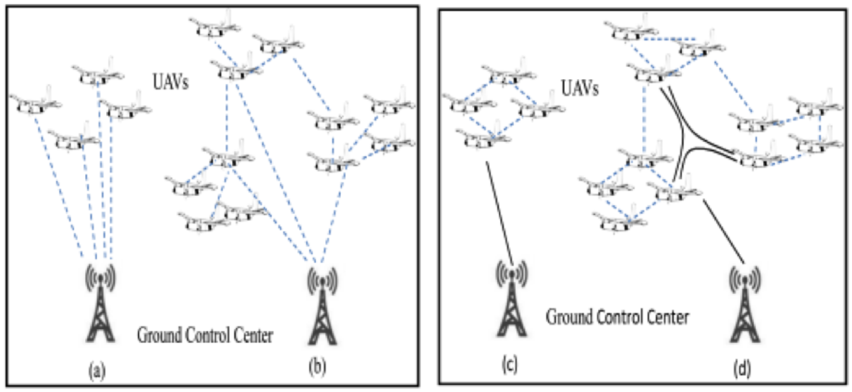



Figure 2.8. Four types of mesh networks: a) Star, b) Multi-star, c) Flat Mesh, and d) Hierarchical. Source: [47].

Routing protocols suitable for UAV swarm networks require energy and location awareness, and must adapt to topology changes. Conventional routing protocols that rely on end-to-end paths will not work in a swarm. Likewise, MANET and FANET protocols such as static, proactive, reactive, and hybrid routing have demonstrated limited results for FANETs [47]. Geographic three-dimensional protocols which utilize "greedy forwarding," wherein each node forwards data to the destination based on local information, have demonstrated adequate simulation performance in terms of throughput and latency, but do not guarantee message delivery [47]. Other promising protocols are store-carry-forward (SCF) and direct delivery [47].

Antenna structure is an important design factor in UAV swarm communication networks. Omni-directional and directional antennae are both used and have different strengths and weaknesses. Omni-directional antennae are naturally suited to an environment in which node locations change frequently, but their transmission ranges are lower, they are more prone to jamming, and latency may be higher [98]. Directional antennae bring longer ranges, higher data transmission capacities, and some protection from jamming, but they require node orientation [98].

Universities and government research labs are performing key field experimentation research using multi-UAV network test beds. A University of Colorado team developed a wireless network test bed, called HUAS, using 802.11b radio equipment on small UAVs, with the dynamic source routing protocol [138]. HUAS was innovative in that it used a layered network composed of off-the-shelf technologies and system-specific algorithms [34]. The Berkeley Aerobot Team uses fixed and rotary wing UAVs in its 802.11 scalable multi-agent UAV network [139]. The Georgia Tech UAV Research Facility conducts multi-UAV research focused on guidance, navigation, and control [140]. In [141], the Cooperative Autonomous Reconfigurable UAV Swarm (CARUS) project flew a swarm of five rotary-wing micro-UAVs to survey ground points of interest by using a distributed broadcast-based approach that enabled swarm decisions to be made locally, while airborne.

Perhaps the most relevant applied research in UAV swarm communication networks has been conducted by ARSENL's aerial combat swarms inspired research [2]–[4]. ARSENL's network is composed of the primary 802.11n (2.4GHz) wireless command and control network that connects the aircraft and ground stations, and two backup communication systems for troubleshooting: a radio controlled receiver for manually controlled flight and a serial telemetry radio that enables two-way communication with a single aircraft's autopilot onboard computer [3]. ARSENL's unique protocol sends command messages as a single broadcast to the entire swarm rather than to individual aircraft, allowing the individuals to sort out specific tasking using consensus algorithms [3], [4].

Most of the research on UAV swarm networks, including the test beds mentioned above, used proprietary unencrypted wireless networks. [142] proposed a CONOPS to leverage public wireless communication for homeland security missions. While these configurations may be suitable for commercial data gathering operations, they are not suitable for DOD missions in adversarial environments. Research in encrypted wireless sensor networks is limited. In [143], Courtney developed a network design for IPv6 over Low Power Wireless Personal Area Networks (6LoWPAN) designed to work with the IEEE 802.15.14 standard. While it has not been tested with mobile nodes or outside of the simulation environment, the framework shows promise for future tactical wireless sensor networks.

In summary, the most promising UAV swarm communication network is currently the meshed ad-hoc network. Meshing provides redundancy in communication, the shorter inter-agent (vs. agent-to-ground) ranges allows bandwidth to be used more efficiently, and meshing extends ranges beyond that of a direct link. The key advantage of meshed networks is that they enable agent self-organization, which allows for greater autonomy and a wider range of missions. Continued research in mesh networks to improve the quality of data transfer and improve energy consumption will likely involve cross-layer (open systems interconnection (OSI) model layers: physical, data link, network, transport) design [47]. Furthermore, FANETs should be encrypted before they are deployed for DOD missions.

### 2.4.4 The Human Component

Even with the advances in autonomy and automation technology, UAVs still require some degree of human guidance. Prior research [14], [17], [144] has shown that the number of UAVs an operator can effectively control depends upon the operational workload, task complexity, and the level of automated decision support tools. Studies of air traffic controllers have shown they can control up to 16–17 aircraft; however, controllers are only providing navigation guidance and not operating sensor or flight controls [17], [145]. Similarly, experimental studies of Tactical Tomahawk Land Attack Missile operators have shown the capability to control up to 12 missiles, operating under substantial missile autonomy [14]. Multi-UAV control experimental and theoretical research have found five to six UAVs to be the upper limit of responsibility for a single operator [3], [14], [17]. Many current single UAV systems, including RQ-4 Globalhawk, MQ-9 Predator, and RQ-7 Shadow operate with a many-to-one ratio of operators to vehicles. Furthermore, these current UAS operate with a minimal level of autonomy. A new model for humans managing machines needs to be considered with autonomy as a partner or aid, rather than as a replacement, to reduce operator workload and to increase performance in dynamic operating environments.

Human-swarm interaction presents a significant challenge in designing a system architecture for large numbers of agents. The traditional pilot paradigm, in which an operator directly controls all aspects of the assigned vehicle(s), does not scale well for large numbers of agents (greater than 10) [3]. In fact, multiple operators are currently needed to control single UAVs such as Predator and Globalhawk [146]. In [146], Lewis defines human-swarm interaction in terms of computational complexity. Within this context, Kolling et al. emphasize that the complexity associated with controlling a swarm of homogeneous agents falls between $O(1)$ for cases in which the number of operator actions are independent of swarm size, and $O(n)$ for cases in which required operator actions scale with the swarm size [82]. By "treat[ing] the swarm as a single entity much of the time, multiple robots can added or removed without impacting the cognitive burden of the human operator" [82]. An application of this concept is demonstrated in Chung et al.'s unique organizational approach for controlling a large swarm. Instead of assigning operators to air vehicles in the traditional sense, they parsed the functions into health monitoring and swarm behavior execution [3]. By splitting the roles by function rather than by vehicle, they were able to fly 50 UAVs simultaneously, with the swarm monitor managing the health of the swarm and the swarm operator concentrating on executing swarm behaviors in support of the mission [3].

Incorporating the autonomous aspects of CAS into the DOD inventory presents challenges in the realms of doctrine, system design methods, and system verification and validation. These domains are not disjoint. From a doctrinal perspective, a flatter more decentralized command structure should be used to take advantage of lower cost, distributed systems. Rules of engagement will not only bound how, when, and where these systems will be used, but also influence system design, such as allowing for mission-specific variability in human input. For situations needing quick reaction, such as defensive countermeasures, removing the human from the decision loop makes sense and is already practiced [5]. Incorporating formal methods into the system design method, particularly in requirements development and system verification and validation has the potential to improve risk characterization and increase trust, further influencing doctrine.

*Autonomy* should not be confused with *automation*, which describes a system that functions with little to no human involvement and is limited to specific rule-based responses in a controlled environment [147], [148]. The most widely used method for automation measuring is Sheridan's 10-level scale [149] in which level one describes a fully manual human-controlled system and level 10 is a completely autonomous system in which human inputs are ignored. Levels 2-4 focus on who makes the decision (the computer or the human), and levels 5-9 focus on how the decision is executed. Proud et al. expanded Sheridan's work by [150] proposing a similar eight-level scale and mapped to the four stages of the Boyd's OODA (observe, orient, decide, and act) Loop. Studies of human interaction with various levels and types of automation within the air traffic control community [145], [151], [152] provide insight into the question "which systems should be automated and to what extent" [151]? Many of these insights can be applied to swarm systems; however, the incorporation of autonomy into the system design increases the complexity of these systems.

Autonomy is a capability that permits a system function to operate within programmed boundaries and in some cases, enables self-governing behavior [153]. An autonomous system operates under a set of intelligent-based capabilities that allow it to react to unanticipated scenarios [147]. Autonomy is built upon the foundation of artificial intelligence, the capability of a computer-based system to execute human-like tasks such as perceiving and making decisions [32]. Swarm intelligence, a type of artificial intelligence, is a "behavior in response to a change in the environment external to the robot, [that] is neither random nor predictable from physical measurements of the environment" [24]. To exhibit this type of behavior, the collective of intelligent agents operates using internal algorithms that are not readily modifiable by humans, in combination with interactions to the external environment. Unpredictability (i.e., not producing ordered patterns [24]) in behavior is another characteristic of swarm intelligence and ties in with the notion of emergent behavior.

**Bounding Autonomy**

In general, systems can be characterized as fully autonomous or semi-autonomous according to the amount of human intervention that is permitted. A completely autonomous system is likely to produce outcomes that are unpredictable [154], and it can be thought of as "having free will" [40]. In 2012, the Defense Science Board asserted that defining levels of autonomy was "not useful" [153]. They argued that competing autonomy definitions are confusing, overly simplistic, and do not address the holistic capability when integrated with the human. Instead, they argued that the DOD should adopt a three-faceted autonomous systems framework composed of cognitive echelon, mission timelines, and human-machine system tradespaces [153]. Previously, the Autonomy Levels For Unmanned Systems (ALFUS) group, composed of research labs from across DOD, developed a framework for characterizing autonomy levels based on a three-axis model including environmental difficulty, mission complexity, and human independence as factors [155]. Yet another approach to classifying autonomy is a certification or licensing process, similar to the way in which biological autonomous systems (human pilots) are certified by the Federal Aviation Administration (FAA) to fly aircraft [156]. Following this approach, different classes of licenses would be granted with consideration that the user understands the capabilities and limitations of the system, in the context of the mission and required tasks [156]. This concept is directly applicable to MASC, which permits variable (by mission) autonomy.

**The Role of Trust**

Trust, "the system status in the mind of the human" [147], presents a considerable barrier to fielding autonomous systems. Trust impacts human performance variables such as workload, training, and stress as well as system measures like reliability, response accuracy, and dependability [36]. Research indicates that properly calibrated trust is a key aspect of safely operating autonomous systems [147]. The Defense Science Board's most recent autonomy study [32] highlights the significant of developing, building, and maintaining trust throughout the life cycle of an autonomous system, Figure 2.9.

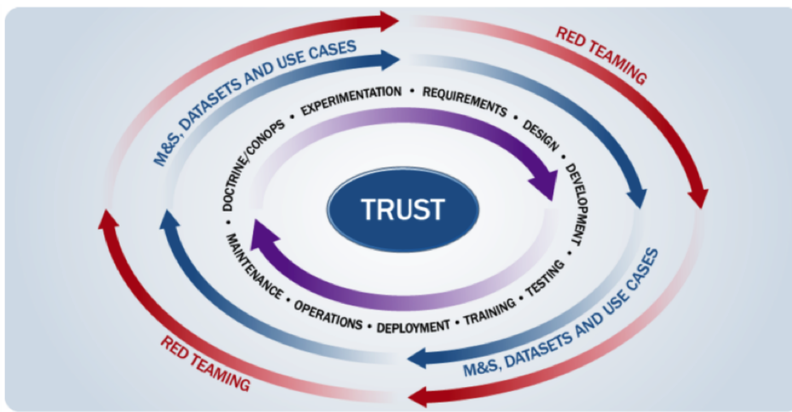



Figure 2.9. Addressing trust in autonomous systems continues throughout the life cycle. Source: [32].

Trust may also be considered "a psychological state comprising the intention to accept vulnerability based upon positive expectations of the intentions or behavior of another" [157]. Cognitive psychologists have found that a decrease in reliability leads to an erosion of trust, and a subsequent increase in operator workload [36]. Research in UAV autonomy has demonstrated that reliability is a critical element in determining appropriate human-to-vehicle ratios [12], [36], [158]. As we move from automated to autonomous systems, the initial need for human interaction increases to account for unpredictability and gaps in trust.

### 2.4.5 Influencing Design Factors Summary

The communication network is a key enabler for supporting a flatter hierarchy, greater elusiveness, and increased situational awareness in swarm systems. Swarms can thrive on distributed communication and sensor networks for coordination, task allocation, and information sharing. An appropriately engineered communication network can enable the swarm to operate using decentralized control. Swarm communications challenges are not only technical; communication is also a critical element from the doctrinal perspective. The flatter and more decentralized command structure of swarm warfare will require more tactical decisions to be made at the junior officer and NCO levels to take full advantage of the converging and dispersing nature of swarm tactics. Automation engineering will enable effective task allocation between agents and between agents and humans. Militaries have traditionally been hierarchical and organized in large groups such as air wings, divisions, expeditionary units, and aircraft carriers, while swarms may benefit from a much flatter organizational structure. Resistance to organizational change will be a non-trivial challenge in developing military swarm doctrine.

## 2.5 Assessment of Previous Work

The literature review describes the vast amount of research across multiple academic disciplines in swarm systems. Promising advances in electronics miniaturization, distributed communication and control networks, biologically-inspired behavior algorithms, and 3-D printing have spurred swarm system development. As with other types of complex adaptive systems, swarm systems present considerable challenges to traditional systems engineering. The swarm's emergent behavior arising from agent-to-agent interfaces along with interactions with the environment generates patterns that are difficult to predict [101]. The majority of the current design approaches are based on using bottom-up design methodologies such as agent-based modeling, FSMs, and Petri Nets to create certain behaviors or perform specific localized tasks. Behavior-based design will continue to play an important role in swarm system design; these approaches have produced successful lab and field experimentation for UAV swarms up to 50 [3], but they are controlled by robotics engineers and computer programmers. The next challenge is advancing swarm systems so that they can be operated by a military user in an operational environment. To accomplish this next step and meet higher level system requirements, the design approach should be reconsidered.

There has been less research focused on top-down, mission-driven architectures that support practical CAS applications to DOD operations. Brambilla et al.'s four-step method provides a general guideline for a top-down approach and rightly emphasizes the need for experimentation with real robots. Perhaps most relevant to this research is the work done with playbooks, particularly DARPA's OFFSET and Browning et al.'s STP. A key feature of STP is the separation of single robot behavior from team behavior so that the team's behavior is a function of the coordinated sequence of behaviors for each individual robot [103]. The MASC framework is less complex in that it creates an overarching, modular playbook of behaviors for a *homogeneous* swarm that can be assembled and sequenced to support a variety of missions. The atomic unit in MASC for behavior control is the sub-swarm.

Playbooks promote using a "high-level language to program a multi-robot application" as underscored by Bachrach et al. [100]. Playbooks can provide a means to develop a standard taxonomy, catalog reusable patterns, and reduce the complexity required to operate the system. The playbook structure can also permit automation to be integrated at varying levels according to ROE and tactics, techniques, and procedures (TTP). Above all, playbooks may provide the critical link between the system designer and the fleet operator.

It is important to consider the doctrinal aspect of swarm systems during the design process. Hughes' five propositions of naval tactics, although written for the surface navy, are relevant to the maritime UAV swarm. The implementations for a distributed, expendable system are different, but the foundations are pertinent. Arquilla and Edwards' historical study of military swarming characterizes patterns that are relevant to modern swarm tactics, particularly, dispersing and amassing to support pulse-like tactics. Incorporating these "many and small" [129] distributed systems into a force that has historically been dominated by carrier strike groups, Marine expeditionary units and army divisions will likely prompt organizational changes.

The complex emergent behavior of swarms including self-organization, adaptability, and collective behavior cannot be completely understood using solely traditional systems engineering decompositional methodologies. Swarm system design cannot be developed in a vacuum, rather it must be done with mission doctrine as a design element. The proposed research pursues the expansion of the work previously accomplished in playbooks and bottom-up methods by adding a top-down methodology for designing UAV swarms by decomposing missions into smaller components. Key swarm patterns such as dividing, dispersing, and amassing are incorporated into MASC. This research approach merges a playbook approach with MBSE foundations and automated tools to develop a top-down methodology that is driven by mission doctrine, and iteratively refined via bottom-up feedback methods to support development of operationally suitable swarm systems. This dissertation contributes to the systems engineering body of knowledge by defining a methodology for integrating mission engineering and systems engineering into a framework that supports both system architecture design and operational analysis models.

# CHAPTER 3: The MASC Framework

Terminology for swarm system functions varies depending on academic field, as discussed in Chapter 2. A standard taxonomy and framework for swarm system functions that is relatable across academic disciplines would facilitate architecture design across communities. This chapter describes the assumed Swarm Operational Team, each component of the Mission-based Architecture for Swarm Composability (MASC) architecture, the methodology for applying MASC to a mission scenario, and the research limitations and assumptions. The MASC components and methodology are applied to three missions case studies in Chapter 4.

## 3.1 Swarm Operational Team

The swarm operational team is based on the UAV swarm field experimentation conducted by ARSENL. ARSENL's approach allocates the functions of swarm management in terms of operating the swarm tactically and monitoring swarm health [3]. This contrasts with more traditional unmanned system management styles wherein a single operator manages all aspects of a vehicle. The team envisioned to operate future swarms is composed of a Swarm Commander, Swarm Health Monitor, and Ground Crew. The Swarm Commander builds the swarm strategy, selects the swarm tactics, and is responsible for the overall execution of the mission. The Swarm Health Monitor oversees the health and function of the swarm, and separates errant individual UAVs from the swarm [3]. The Ground Crew is responsible for swarm system preflight, launch, and postflight duties. The ARSENL team also employs a Mission Commander to oversee the mission from a safety perspective. This research assumes that improvements to the ground control station (GCS) graphical user interface (GUI) and advancements in technology readiness levels of a notional swarm system will allow the Swarm Commander to assume the Mission Commander role. Figure 3.1 shows the Swarm Team within the physical context of a swarm operational mission.

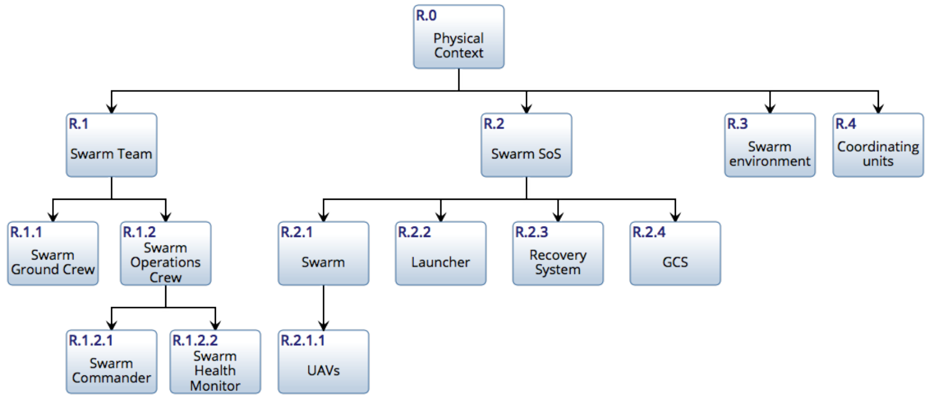



Figure 3.1. The physical context of the swarm system is composed of the swarm human team, the swarm SoS, the environment, and the other units with which the swarm coordinates.

## 3.2 MASC Description

The MASC framework uses a modular, composable framework for defining UAV swarm missions. Each mission includes five sequential phases, each phase is composed of one or more tactics, each tactic is composed of one or more plays, and likewise, each play is composed of one or more algorithms. Modifiable parameters such as altitude, velocity, offset distances, execution times, target behavior triggers, and geographic constraints enable the plays to be tailored to mission specific conditions. These elements are designed to be reusable for different missions. Figure 3.2 shows the overall picture of the MASC architecture. This research focuses on the operational part of the architecture—the missions, phases, tactics, and plays—and treats the UAV swarm as a singular unit to simplify operational modeling.

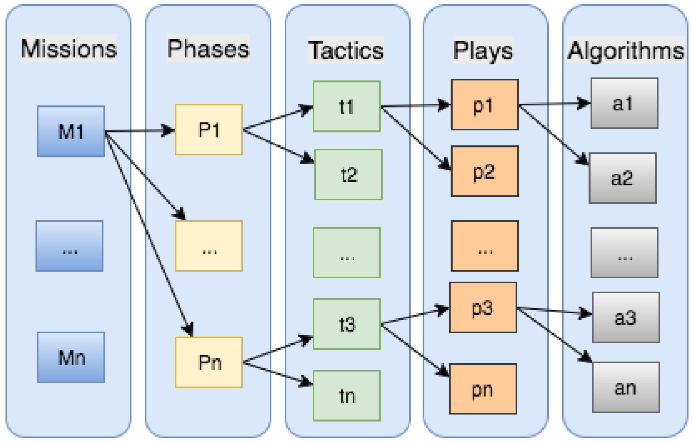


Figure 3.2. MASC is a many-to-many framework of elements, starting with missions at the highest level. Each mission is composed of phases, tactics, plays, and algorithms at the lowest level.

### 3.2.1 Missions

The swarm mission is the highest level element of the architecture, and describes the overall task or objective assigned to the swarm. The MASC framework was applied to three missions: SvS, MIO, and HADR. Other potential operational swarm missions include ISR, SAR, suppression of enemy air defenses (SEAD), close air support, and ASW. Swarms are assumed to operate alongside other unmanned and manned operational assets.

### 3.2.2 Phases

The swarm mission phases are a means of categorizing the tactics temporally within a mission. There are seven swarm mission phases: Staging, Mission Planning, Preflight, Ingress, OnStation, Egress, and Postflight. The first two phases are planning focused, while the next five are considered operational phases. Each mission includes the same pattern of five operational phases—Preflight, Ingress, OnStation, Egress, and Postflight—in addition to common patterns used within the phases. The Preflight, Ingress, Egress, and Postflight phases tend to be similar among the different mission scenarios, with the greatest variety between missions occurring in the OnStation phase. The inflight phases of the mission (Ingress, OnStation, and Egress) are the focus of the MASC framework.

The Staging phase begins once the UAV swarm is in its travel configuration and arrives at the designated deployment site (ground or shipboard). The Mission Planning phase begins when the Swarm Commander receives a tasking order to be prepared to execute a mission. The Preflight phase begins when the swarm is powered on and concludes when the swarm is in a flight ready status. The Ingress phase commences when the first UAV has been loaded onto the launcher and concludes when the UAV swarm has arrived at the OnStation waypoint. The OnStation phase begins when the entire swarm reaches the assigned OnStation area and ends when the first UAV in the swarm reaches bingo fuel (the minimum fuel required for a safe return to base) or the OnStation tasking ends. The Egress phase starts when the UAV swarm is on a flight path to return to base (or ship) and concludes when the entire swarm has landed. The Postflight phase begins when the swarm has landed and ends when the mission has been debriefed.

### 3.2.3 Tactics

A swarm tactic commands the ordered formation and employment of individual agents to perform a specific task as a cooperative group. They are designed to function at a level of abstraction sufficient for use across multiple missions. Swarm tactics are composed of one or more swarm plays. Each tactic is described below in terms of the plays from which it is composed. Activity diagrams of each tactic are shown in Appendix D. The MASC includes the following swarm tactics:

- **`Ingress` ($t_1$)** includes swarm launching, transit to the OnStation waypoint, and sensors activation. It is composed of the following plays: `Launch` ($p_1$), `Transit to waypoint` ($p_2$), and `Sensors ON` ($p_{8.1}$). The `Ingress` tactic is the first inflight tactic of each mission and is shown in Figure D.1.
- **`Evasive search` ($t_2$)** uses a random search pattern to confuse the adversary (Figure D.2). It is composed of the following plays: `Random pattern` ($p_{14}$), and `Sensors ON` ($p_{8.1}$) or `Sensors EMCON` (emissions control) ($p_{8.3}$).
- **`Efficient search` ($t_3$)** is used for methodical searching, wherein efficiency is prioritized over the risk of predictability to the adversary (Figure D.3). It is composed of the following plays: `Sensors ON` ($p_{8.1}$), and a choice of `Ladder pattern` ($p_{10}$), or `Expanding square pattern` ($p_{11}$), or `Constricting square pattern` ($p_{12}$), or `Grid pattern` ($p_{13}$).
- **`Track` ($t_4$)** is used for maintaining sensor focus on a target and continuing to follow

the target (Figure D.4). It is composed of the following plays: `Sensors ON` ($p_{8.1}$) and `Follow target` ($p_{18}$).

- **`Communication relay` ($t_5$)** is used to relay communication between other participating units (Figure D.5). If communication relay is the primary task, then the swarm is scattered in a grid pattern at a specified altitude to optimize relay of communication to other units. In the grid pattern, an individual UAV orbits within assigned box and spacing is based on the desired coverage probability, area to be covered, and number of agents in swarm. If other maneuver patterns are desired, then communication will be relayed (as a secondary task) while performing the maneuver. Ideally, a sub-swarm is assigned this tactic to relay communication while other sub-swarms perform different tactics. Communication relay is composed of the following plays: `Sensors ON` ($p_{8.1}$), `Transmit video` ($p_{8.4}$), `Forward communication` ($p_{19}$), and a choice of `Ladder pattern` ($p_{10}$), `Expanding square pattern` ($p_{11}$), `Constricting square pattern` ($p_{12}$), or Grid pattern ($p_{13}$).
- **`Attack` ($t_6$)** is an offensive tactic reserved for missions operating under ROE that allow weapons to be fired (Figure D.6). Authorization from a human operator can be specified as a required input for enabling this tactic which includes: `Weapon armed` ($p_{15}$), `Smart greedy shooter` ($p_{20}$), or `Patrol box shooter` ($p_{21}$), or `Wingman shooter` ($p_{22}$), and `Weapon fire` ($p_{17}$).
- **`Battle Damage Assessment (BDA)` ($t_7$)** is used to assess the damage to a target that has been attacked (Figure D.7). `BDA` may be followed by another attack. It is composed of `Sensors ON` ($p_{8.1}$) and `Expanding square pattern` ($p_{11}$).
- **`Monitor` ($t_8$)** is used for monitoring an area for radio frequency (RF), radar returns, and voice traffic using all sensors or just passive sensors such as electronic surveillance measures receivers, infrared (IR), and electro-optic (EO) (Figure D.8). `Monitor` is composed of: `Orbit` ($p_3$) or `Racetrack` ($p_4$) and `Sensors ON` ($p_{8.1}$) or `Sensors EMCON` ($p_{8.3}$)
- **`Evade` ($t_9$)** is a defensive tactic used when the swarm senses it is being threatened (Figure D.9). It disperses the swarm to multiple directions and altitudes to avoid an attack, and includes the plays: `Sensors EMCON` ($p_{8.3}$), `Disperse` ($p_7$), `Join` ($p_6$), and `Jam` ($p_{19}$).
- **`Harass` ($t_{10}$)** is an offensive tactic used to disrupt an enemy by maintaining a presence and confusing the enemy with active sensors (Figure D.10). After reaching the

designated area, the swarm conducts electronic jamming on the enemy's radar while performing one of several maneuver options. `Harass` consists of the plays: `Transit to waypoint` ($p_2$), `Sensors ON` ($p_{8.1}$), `Jam` ($p_{20}$), and three maneuver options: `Orbit` ($p_3$), `Tail following` ($p_{21}$), `Disperse` ($p_7$), and `Join` ($p_6$).

- **`Defend` ($t_{11}$)** is a defensive tactic used to guard a high value asset or home base, in which the swarm transits to the designated area and operates in a racetrack pattern with sensors on and weapon armed (Figure D.11). `Defend` is composed of the plays `Transit to waypoint` ($p_2$), `Racetrack` ($p_4$), `Sensors ON` ($p_{8.1}$), and `Weapon armed` ($p_{15}$).
- **`Deter` ($t_{12}$)** is a defensive tactic used to dissuade the enemy from operating in a particular area (Figure D.12). It is a less escalated version of `Defend`, as it does not arm a weapon. This tactic is useful when operating under ROE which do not permit armed UAVs. `Deter` is composed of the plays `Transit to waypoint` ($p_2$), `Racetrack` ($p_4$), or `Orbit` ($p_3$), and `Sensors EMCON` ($p_{8.3}$) or `Sensors ON` ($p_{8.1}$).
- **`Divide` ($t_{13}$)** is used to logically divide a swarm into sub-swarms, enabling multiple mission tactics to be performed concurrently (Figure D.13). `Divide` is an important tactic that is likely be used continuously throughout swarm missions. Dividing and re-assembling the swarm to execute pulsing attacks, as discussed in Section 2.4.2, embodies the unique tactical advantage of a swarm system. `Divide` is composed of the plays `Split` ($p_5$), `Sensors EMCON` ($p_{8.3}$) or `Sensors ON` ($p_{8.1}$), and `Orbit` ($p_3$).
- **`Amass` ($t_{14}$)** is used to re-assemble sub-swarms into a larger swarm after they have been divided (Figure D.14). `Amass` is designed to be used along with the `Divide` tactic to support pulsing attacks by surprising the enemy with variable force sizes. `Amass` is composed of the plays `Join` ($p_6$), `Sensors EMCON` ($p_{8.3}$) or `Sensors ON` ($p_{8.1}$), and `Orbit` ($p_3$).
- **`Egress` ($t_{15}$)** is the last tactic executed for each mission, and occurs after the OnStation phase has finished (Figure D.15). It returns the swarm to the operating base or ship, as applicable. `Egress` is composed of `Transit to waypoint` ($p_2$, `Terminal approach` ($p_{9.1}$), `Sensors OFF` ($p_{8.2}$), and `Landing` ($p_{9.2}$).
- **`Air combat maneuvers (ACM)` ($t_{16}$)** is an offensive tactic used for A-A engagements with airborne targets (Figure D.16). `ACM` is designed to be used in missions in which the degree of swarm autonomy is high and the decisions are made autonomously by the system due to the kinetic nature of this mission type. It is composed of `Sensors`

ON ($p_{8.1}$), Weapon armed ($p_{15}$), and aerobatic maneuvers such as Tail following ($p_{21}$) and future growth ACM plays.

### 3.2.4 Plays

The swarm plays describe the behaviors and maneuvers of the swarm as a cooperative group. Plays are the building blocks for swarm tactics and may be used in combination to create multiple tactics. Each play is defined and mapped to potential algorithms. No original algorithms were developed during this research, however, existing algorithms are mapped to plays for context within the MASC framework. The MASC framework includes the following swarm plays:

- **Launch ($p_1$)** is the play used for transitioning the UAVs from the airfield or ship into the airborne environment. The operator modifiable parameters are: number of UAVs to be launched, number of launchers, and the time interval in seconds between UAV launches. The time interval can be set to a minimum threshold to perform a rapid launch. Launch is part of the Ingress tactic.
- **Transit to waypoint ($p_2$)** can be used as a single play or in combination to perform various flight paths and patterns. It is used in the Ingress, Egress, Deter, Harass, and Defend tactics. Operator modifiable parameters include: transit altitude or altitude block (for evasive transits), and waypoint latitude/longitude.
- **Orbit ($p_3$)** is a play that enables the swarm to encircle a fixed geographic position or target. It is used in the Monitor, Harass, and Deter tactics. The operator modifiable parameters are: altitude or altitude block (to stack the UAVs at multiple altitudes), clockwise or counterclockwise flow, designated center waypoint latitude/longitude, and radius of orbit (in m or nm).
- **Racetrack ($p_4$)** is a play that enables the swarm to encircle a fixed geographic position or target in a racetrack pattern. It is used in the Monitor, Defend, and Deter tactics. The operator modifiable parameters are: search sensor coverage parameters, altitude or altitude block (to stack the UAVs at multiple altitudes), clockwise or counterclockwise flow, designated center waypoint latitude/longitude, width, and length (in m or nm).
- **Split (logic-based) ($p_5$)** splits the swarm into sub-swarms based on parameters: health status, battery life remaining, number of sub-swarms, location, or proximity to

a reference waypoint. This play supports the `Divide` tactic.

- **`Join` ($p_6$)** is used to return UAVs to their original swarm or sub-swarm after being split or dispersed. It is used in the `Evade, Harass`, and `Amass` tactics. The operator modifiable parameters are designated join time, specified joining waypoint latitude/longitude, and altitude block.
- **`Disperse` ($p_7$)** is a defensive play to scatter and separate the UAVs within a cylinder, to support the `Evade, Harass`, and `Divide` tactics. The operator modifiable parameters are dispersion radius and altitude block.
- **`Sensors ON` ($p_{8.1}$)** is the default sensor setting that turns on all available non-navigational sensors. Sensors are turned on after the UAV has been launched and has executed a positive rate of climb. Several OnStation phase tactics, including both search tactics, `Track`, and `Monitor`, begin with `Sensors ON`. The operator modifiable parameters are the individual ON or OFF settings for each individual sensor (i.e., radar, EO, IR, light detection and ranging (LIDAR), electronic surveillance measures).
- **`Sensors OFF` ($p_{8.2}$)** is the default sensor setting that turns off all non-navigational sensors. Sensors (i.e., radar, EO, IR, LIDAR, electronic surveillance measures) are turned off just prior to landing, as part of the `Egress` tactic to prevent damage to sensors and injury to personnel.
- **`Sensors EMCON` ($p_{8.3}$)**, or emissions control, is the sensor setting used for evasive tactics. It turns the active sensors, such as radar and LIDAR, off and leaves the passive sensors (i.e., EO, IR) on. `Sensors EMCON` is part of the `Monitor` and `Evade` tactics.
- **`Transmit video` ($p_{8.4}$)** is the play that enables swarm video to be transmitted to receivers other than the GCS, and it is part of the `Communication relay` tactic. The operator modifiable parameters are the receiving nodes' addresses and sensor data to be transmitted.
- **`Terminal approach` ($p_{9.1}$)** sequences the swarm for landing according to the individual UAV's battery life and proximity to the landing waypoint. Operator modifiable parameters allow for a single terminal approach path or multiple terminal approach paths, normal glideslope or combat descent rate, and inbound headings. `Terminal approach` is part of the `Egress` tactic and it is used at the end of each mission.
- **`Landing` ($p_{9.2}$)** defines the landing area for the swarm. User modifiable parameters are available for specifying the dimensions of landing area: radius of a circle or *m* x

*n* dimensions of a rectangle, or a specified separation distance between UAVs other than the default setting. `Terminal approach` is part of the `Egress` tactic and it is used at the end of each mission.

- **`Ladder pattern` ($\mathbf{p_{10}}$)** is a pattern play used in the `Efficient search` and `Communication relay` tactics. The pattern commences at a known datum and executes a long track leg, followed by a 90° turn, a short leg, another 90° turn in the same direction, a long track leg of the same length as the previous long leg, a 90° turn in the opposite direction from the previous turn, a short leg of the same length, etc. The track leg lengths are dependent on the swarm size, sensor range, and the operator modifiable parameters: desired probability of detection, search area dimensions, search sector, and altitude block.
- **`Expanding square pattern` ($\mathbf{p_{11}}$)** is commonly used in SAR missions. The pattern starts at a known datum, then transits to the first waypoint for a distance $d$, makes a 90° turn (all turns are in same direction), transits to the next waypoint at a distance $d$, makes a 90° turn for a distance $2d$, makes another 90° turn, transits $2d$ to the next waypoint, performs a 90° turn, then transits $3d$ to next waypoint, and so forth. The distance $d$ is dependent on timeliness of the datum and the best known speed of target, and consideration of environmental conditions such as wind and current. The `Efficient search, Communication relay`, and `BDA` tactics all have the `Expanding square pattern` play option. The operator modifiable parameters are altitude block, search area dimensions, search sector, datum latitude/longitude, and estimated speed of the target.
- **`Constricting square pattern` ($\mathbf{p_{12}}$)** supports the `Efficient search` and `Communication relay` tactics. Beginning from an offset distance $d$, it is performed in an inverse fashion of the `Expanding square pattern`, moving inward toward the last known position. $d$ is dependent on timeliness of the datum and the best known speed of target, and consideration of environmental conditions such as wind and current. The operator modifiable parameters are altitude block, search area dimensions, search sector, datum latitude/longitude, and estimated speed of the target.
- **`Grid pattern` ($\mathbf{p_{13}}$)** supports the `Efficient search` and `Communication relay` tactics. `Grid pattern` defines an operating area in terms of a large cube broken into smaller boxes. A lead UAV is collectively selected and acts as master UAV by allocating tasks and grid assignments to the other UAVs. Each UAV searches its

assigned box and reports back to the assigned lead, using communication relay via the other UAVs as necessary. The operator modifiable parameters are search area dimensions, desired area coverage, and altitude block.

- **`Random pattern` ($p_{14}$)** maneuvers the swarm together in a random pattern by transiting to randomly generated waypoints within an operating area. `Random pattern` supports the `Evasive search` tactic and its operator modifiable parameters are altitude block, search area dimensions, and search sector.
- **`Weapon armed` ($p_{15}$)** enables weapon arming when the swarm has reached a swarm ready state and if the `Attack` tactic is enabled for the mission. Availability of this play is ROE dependent and loaded during mission planning.
- **`Weapon fire` ($p_{16}$)** enables the available weapon(s) to be fired. The operator modifiable parameters are weapon selection and manual or automatic mode. Automatic mode is designed for specific scenarios such as close-in ship support or a SEAD mission, and is ROE dependent. Manual mode is the normal mode of operation and requires the operator to manually initiate the weapon after a target has been identified. `Weapon fire` is part of the `Attack` tactic.
- **`Follow target` ($p_{17}$)** is used to follow and track a ground target and is part of the `Track` tactic. Operator modifiable parameters are offset slant range and altitude block.
- **`Forward communication` ($p_{18}$)** is part of the `Communication relay` tactic and it forwards voice or data link communication received from another unit to another unit.
- **`Jam` ($p_{19}$)** uses various forms of electronic jamming and other electronic deception techniques to interfere with an enemy's radar and other C2 systems [159]. It is part of the `Evade` and `Harass` tactics. The operator modifiable parameters are frequency bands and temporal schedule.
- **`Smart Greedy Shooter` ($p_{20}$)** was developed by ARSENL and improves on the original ARSENL greedy shooter by coordinating target allocation to ensure each adversary is engaged by no more than one friendly UAV. Each UAV engages the adversary determined to be optimal via the shortest distance to an adversary [160]. This play supports the `Attack` tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.
- **`Patrol Box Shooter` ($p_{21}$)** was developed as an ARSENL behavior and provides

area-defense that incorporates smart shooter selection semantics in randomized patrol of defended area. UAVs individually determine patrol patterns within a defended area without coordination or notification [160]. This play supports the `Attack` tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.

- **`Wingman Shooter` ($p_{22}$)** was developed as an ARSENL behavior and coordinates target allocation to ensure each adversary is engaged by two friendly UAVs. The pair will engage the adversary determined to be optimal via shortest distance to adversary [160]. This play supports the `Attack` tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.
- **`Tail following` ($p_{23}$)** used to follow other aircraft, friend or foe. The operator modifiable parameters are altitude block, offset range (m or nm), and knock-it-off criteria (i.e., range from home or geographic boundary, or surrounded by x bandits).

### 3.2.5 Algorithms

Swarm algorithms are the self-contained sequence of procedures to be performed by the UAV swarm control software for executing a specific play. They derive data such as position, heading, velocity, altitude, health status, and state from individual UAVs, and act as building blocks for the swarm plays. A swarm play is composed of one or more swarm algorithms. Within the context of the MASC framework, the swarm algorithms show the boundary between the operational concepts and the implementation programming levels. The MASC includes the following swarm algorithms (described in detail in Section 2.3.8) as examples for play-to-algorithm mapping:

- Flocking ($a_1$)
- Boid's ($a_{1.1}$)
- PSO ($a_{1.2}$)
- Levy Flight ($a_{1.3}$)
- Scheduling ($a_2$)
- Sorting ($a_3$)
- Collective consensus ($a_4$)
- Artificial potential fields ($a_5$)
- Brownian motion ($a_{8.7}$)

- Leader-follower ($a_{11}$)

## 3.3 MASC Application Methodology

The MASC elements described in Section 3.2 are applied to three mission scenarios in Chapter 4 using the following procedure. Figure 3.3 shows the overall methodology flow.

1. Develop a high-level mission scenario. Consider the swarm's role in the mission, including ROE, potential interactions with key players, and appropriate level of swarm automation. Write a detailed narrative describing what key players are doing during each swarm phase from the temporal perspective of swarm.
2. Create an FSM showing the high-level interactions between swarm tactics.
3. Create an activity model simulation for the mission, beginning at the phase level. Then decompose each phase into tactics from the swarm's perspective using the model heuristics (4.1.1). Next, select the applicable plays from the options available, and modify the play parameters as required.
4. Run the activity model simulation to check for logic errors while using MP modeling to focus on specific interactions and to identify any potential undesired behaviors.
5. Review the implementation with the stakeholder.
6. Modify playbook elements and activity model as necessary.
7. Incorporate the lessons learned into swarm doctrine and swarm system requirements.

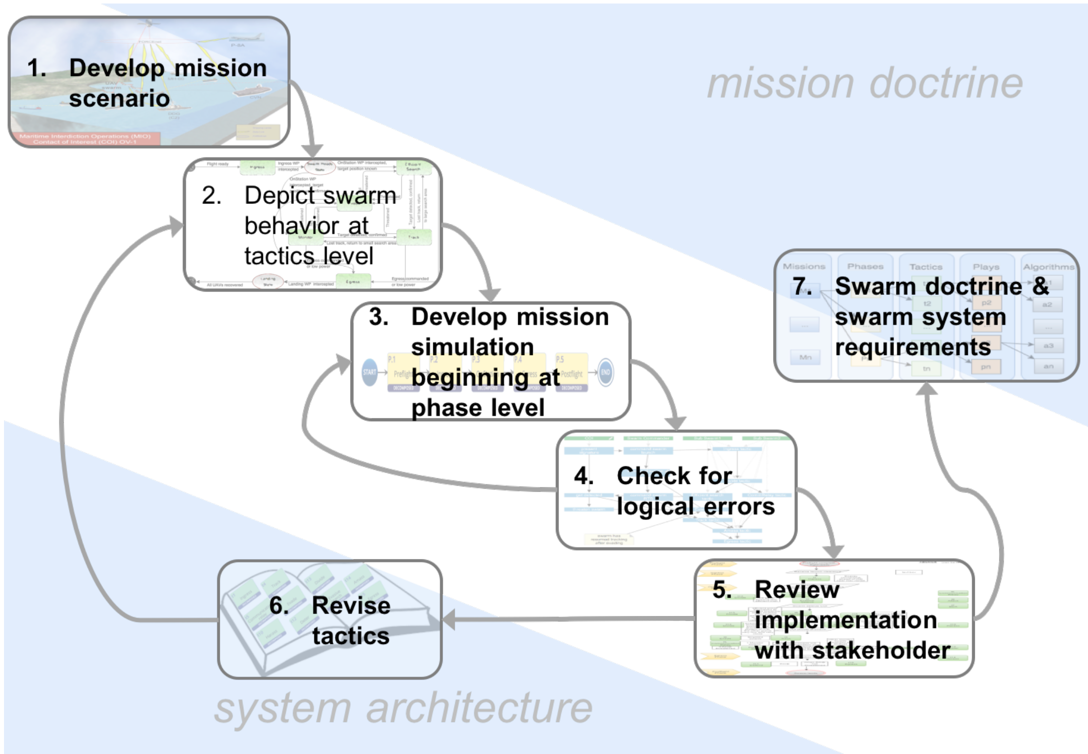



Figure 3.3. MASC is applied to a mission using a seven-step iterative process.

## 3.4 Research Assumptions and Limitations

The MASC architecture assumes a UAV swarm system that is mature enough to operate with three operator roles: Swarm Commander, Swarm Monitor, and Ground Crew. The Ground Crew role will likely be performed by more than one person, but for model conciseness, it is described in terms of one role. The notional swarm system considered for this research is a homogeneous swarm of low-cost DOD category 1–2 fixed-wing UAVs. Two preparation phases (Staging, Mission Planning) are assumed to occur before each mission but are not included as part of the architecture because they mainly involve logistical activities. Tactics are performed by the entire swarm, with the sub-swarm acting as the smallest unit available for tactics assignment. A sub-swarm is capable of performing one tactic at a time; to enact different tactics concurrently, the swarm must be split into sub-swarms and assigned different tactics accordingly. The swarm system is assumed to “know” how many UAVs are airborne and uses this data in swarm play execution. Failures, errors, and failsafe modes are not included as part of the models in this research. In the event of an individual UAV failure, the vehicle is assumed to enter a failure mode and execute the corresponding failsafe logic as in reference [161].

## 3.5 Summary

This chapter began by describing the swarm team envisioned to operate a future UAV swarm. Each MASC element—mission, phase, tactic, play, and algorithm–was defined as a composable element of the architecture and the methodology for applying MASC to an operational scenario was explained. Finally, the assumptions and limitation associated with the research were described. Chapter 4 takes the MASC elements and methodology and applies it to three mission scenarios.

# CHAPTER 4: MASC Application to Three Missions

This chapter applies the components of the MASC architecture and application method, described in Chapter 3, to a UAV swarm operating within three mission case studies—SvS, MIO, and HADR. MASC is applied to three scenarios to demonstrate architecture reusability in terms of modularity, composability, and mission doctrine integration. Each mission scenario is detailed in Appendix B. The SvS scenario was selected as the first case study because it is closely aligned with the ARSENL field experimentation work and provided a baseline for swarm algorithms and plays, and a tangible resource for human-swarm interaction considerations. The MIO mission was developed next as the focus for the HSR because it provided a scenario familiar naval aviators and offered variability in tactic and play selection due to the contested environment. HADR was used for the third case study because it is one of the core capabilities of the U.S. Navy, and it is representative of currents operations in uncontested environments. The three case studies show the logical composition of MASC in an operational construct. The Preflight, Ingress, Egress, and Postflight phases are similar for each mission, while the OnStation phases vary greatly between missions in terms of tactics composition and are the focus in this chapter. The Preflight phase is the only one that does not contain any tactics, as it covers the swarm activities just prior to flight.

## 4.1 Overall Approach

After the general concept for the MASC framework was established using activity modeling in Innoslate to categorize the elements, it was applied to the SvS scenario. This case study was explored first due to its familiarity to the author as the quarterly ARSENL field experimentation research. As the catalog of tactics and plays expanded, the MIO and HADR scenarios and simulations were added. A difference in swarm operational team composition prompted a project split between the experimentation-based SvS scenario and the two notional operational missions. In the end, the same five mission phases represent the top-level of the Innoslate activity model simulation for all three case studies (Figures E.1, E.9, E.19). The makeup of phases between the three case studies is quite similar (Appendix

E). The greatest variation between patterns can be seen in the elements of the OnStation phases due to the difference in missions, participating assets, interfaces, and triggering inputs.

### 4.1.1 Modeling Heuristics

The following five heuristics from Rodano and Giammarco are applied to the Innoslate models [162] to simplify and standardize interfaces, and promote architecture modularity and composability. In the model construct, performers perform activities which result in a subsequent activity or resource output. $A$ denotes the set of activities, $P$ denotes the set of performers, and $R$ denotes the set of resources [162]. The first three heuristics were also applicable to the MASC framework. The application of these heuristics to the models and framework are assessed for completeness in Section 4.5.

1. Every activity not designated a context activity should have at least one parent.
   $(\forall a_i \in A)[\neg context(a_i) \rightarrow (\exists a_j \in A) decomposes(a_i, a_j)]$
2. No activity shall have exactly one child.
   $(\forall a_i \in A)(\forall a_j \in A)[decomposedby(a_i, a_j) \rightarrow (\exists a_k \in A)(decomposedby(a_i, a_k) \wedge (a_j \neq a_k))]$
3. No activity shall be decomposed by itself.
   $(\forall a \in A)\neg[decomposedby(a, a) \in A)]$
4. No performer shall have more than seven children.
   $(\forall p_i \in P)[(\forall p_j \in P)|decomposedby(p_i, p_j)| \leq 7]$
5. Every activity shall have at least one input or trigger.
   $(\forall a \in A)(\exists r \in R)[input(r, a) \vee trigger(r, a)]$

### 4.1.2 Innoslate Model Development

Early Innoslate models attempted to capture too many mission elements at one level, not allowing for modular elements—such as the swarm's preflight activities—to be shared between missions. Eventually, mission *phases* were added to the framework to facilitate modularity and composability across the missions. Many of the activity patterns within each phase—particularly those found in Preflight, Ingress, Egress, and Postflight—can be reused across the missions. The phases are designed from the temporal perspective of the UAV swarm and include not only the swarm's activities, but also those of the coordinating units during those periods, as shown in Figure 4.1. Each performer's activities are represented on a separate branch of the diagram. The green parallelograms represent interaction constraints between activities. The interactions that only show an input or an output in Figure 4.1 (COI position, target signature, and nothing) coordinate with different sections of the simulation not depicted in this diagram. The HADR Preflight phase (Figure E.20) is simpler, with fewer coordinating units, but contains the same "Preflight swarm" activity. The SvS mission's Preflight contains just the activities of the swarm and swarm team (Figure E.2).

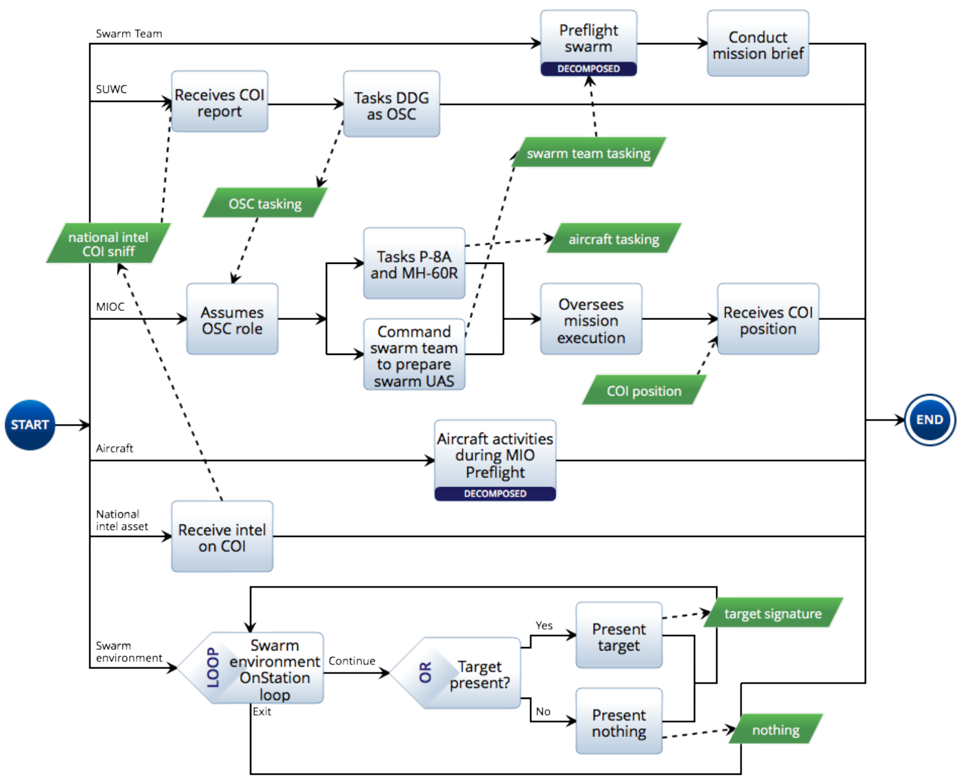
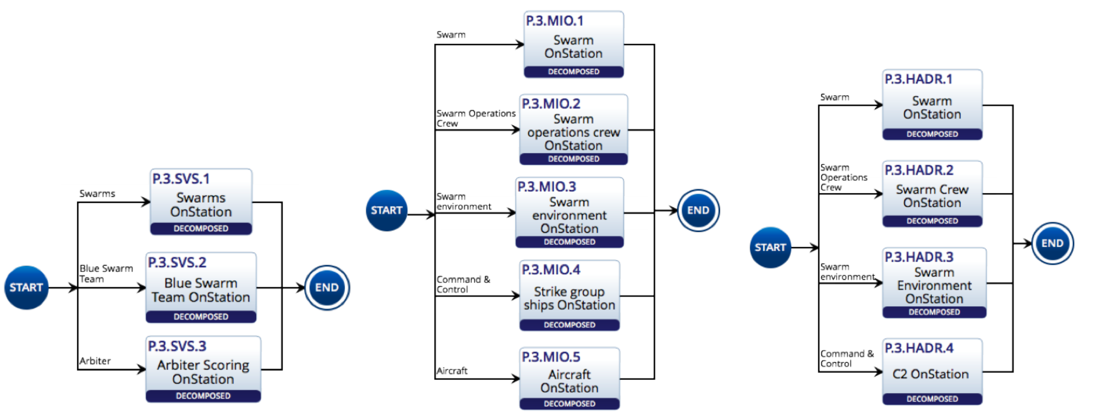



Figure 4.1. The MIO mission preflight phase showing the activities taking place during the swarm's preflight. Each parallel branch contains a performer's activities. Green parallelograms indicate interactions between events.

The Ingress, Egress, and Postflight phases for the three missions are similar, allowing them to be reused among the missions. The Ingress phase contains the activities of the swarm team, scenario-related coordinating units, and the `Ingress` tactic as show in Figures E.3, E.11, E.21. The MIO and HADR Egress phases contain the activities of the swarm team and the Amass and Egress tactics (Figures E.17, E.26). The SvS Egress phase does not include the `Amass` tactic because the swarm operates as one swarm and does not divide into sub-swarms during OnStation (Figure E.7). The MIO and HADR Postflight phases are identical, while the SvS version only varies by crew composition (includes Mission Commander) as shown in Figure E.8.

Of the five operational phases, OnStation contains the largest number of activities. To adhere to heuristic #5 and instill modularity (and minimize diagram clutter), an intermediate diagram was created to group the activities by performers as shown in Figure 4.2.




(a) SvS (b) MIO (c) HADR

Figure 4.2. The activities for each performer during the OnStation phase for the SvS, MIO, and HADR Innoslate mission simulations.

The three resulting Innoslate models are executable as simulations, in discrete event or Monte Carlo modes. The discrete event mode is useful for verifying the logical correctness of the model and process bottlenecks while the Monte Carlo mode can be used to analyze variance of event sequences and timing across multiple simulation runs. Furthermore, time approximations are assigned to each action in the model so that a mission time can be calculated for a simulated mission. Sections 4.2, 4.3, and 4.4 detail the swarm's activities during the OnStation phase for each of the three missions.

### 4.1.3 Swarm Missions Modeled as FSMs

Finite state machines, or finite automata, are commonly used to depict individual UAV behavior, as described in Section 2.3.3. The FSM is also a concise way to express a UAV sub-swarm's activities at the tactics level. A finite automaton $M$ representing a mission is defined by a 5-tuple $(\Sigma, S, s_0, F, \delta)$ [163] in which:

- $\Sigma$ denotes the set of inputs to $M$
- $S$ denotes the set of states, including tactics, of $M$
- $s_0 \in S$ denotes the initial state of $M$
- $F \subseteq S$ denotes the set of final states of $M$
- $\delta : Sx\Sigma \Rightarrow S$ denotes the transition function

The Innoslate tool does provide a state machine diagram, but it does not interface with the simulation function to enable logic checks. For this reason, the FSMs were used as a planning tool for developing the Innoslate action diagrams, specifically the triggers between tactics. However, the FSMs could be translated into MP models to generate use cases.

### 4.1.4 MP Model Development

Innoslate provides a means to catalog the MASC elements, create the overall mission scenarios, and develop working simulations, while MP is used for analyzing different combinations of tactics or plays within a smaller scope of the model, such as the OnStation phase. From the exhaustive set of use cases that MP generates, unwanted behaviors and interactions can be identified to promote model improvement. When MP models are built using the tactics as composite events composed of plays, the number of possible use cases can be compared between tactics-level management and play-level management by commenting out the plays. This parameter could be used to compare computational complexity for different configurations.

Another advantage of the MP tool is that it allows for conversion of FSMs to MP models using Kripke structures [90], [164]. The FSM approach is a workaround for overcoming MP's stipulation that events must have unique occurrences [90], which prevents a swarm modeled as a root from cycling between tactics. Details on MP syntax, semantics, and applications are described in [90].

## 4.2 MASC Applied to the SvS Mission

### 4.2.1 SvS Action Diagram and Simulation

The activities of the Blue and Red Swarms during the OnStation phase are depicted as a LML action diagram in Figure 4.3. The action diagram represents a portion of the mission simulation developed in Innoslate [165]. The red circled part of the diagram highlights the activities of the Blue Swarm using the MASC framework. Tactics are portrayed by green rectangles and labeled “t#.”

Once the Blue Swarm arrives at the OnStation waypoint (Figure 4.3), it receives authorization from the Arbiter to begin the search and destroy mission against the Red Swarm, as circled. The `Evasive search` and `Track` tactics are somewhat artificial in this case study as the UAVs do not have actual sensors but receive re-transmitted GPS positions of adversaries from the Arbiter. The Blue Swarm continues to `Evasive search`, `Track`, and `Attack` Red Swarm UAVs until they have been depleted, as notified by the off-station notification from the Arbiter. The two parallel activity branches depict the Blue Swarm UAV depletion by the Red Swarm and the message traffic between Blue Swarm and the Arbiter. The overall mission flow for the SvS mission (described in Section B.1) and diagrams of the phases other than Swarms OnStation are shown in Figures E.1, E.2, E.3, E.4, E.7, and E.8. The composition of each tactic is depicted in Appendix D.

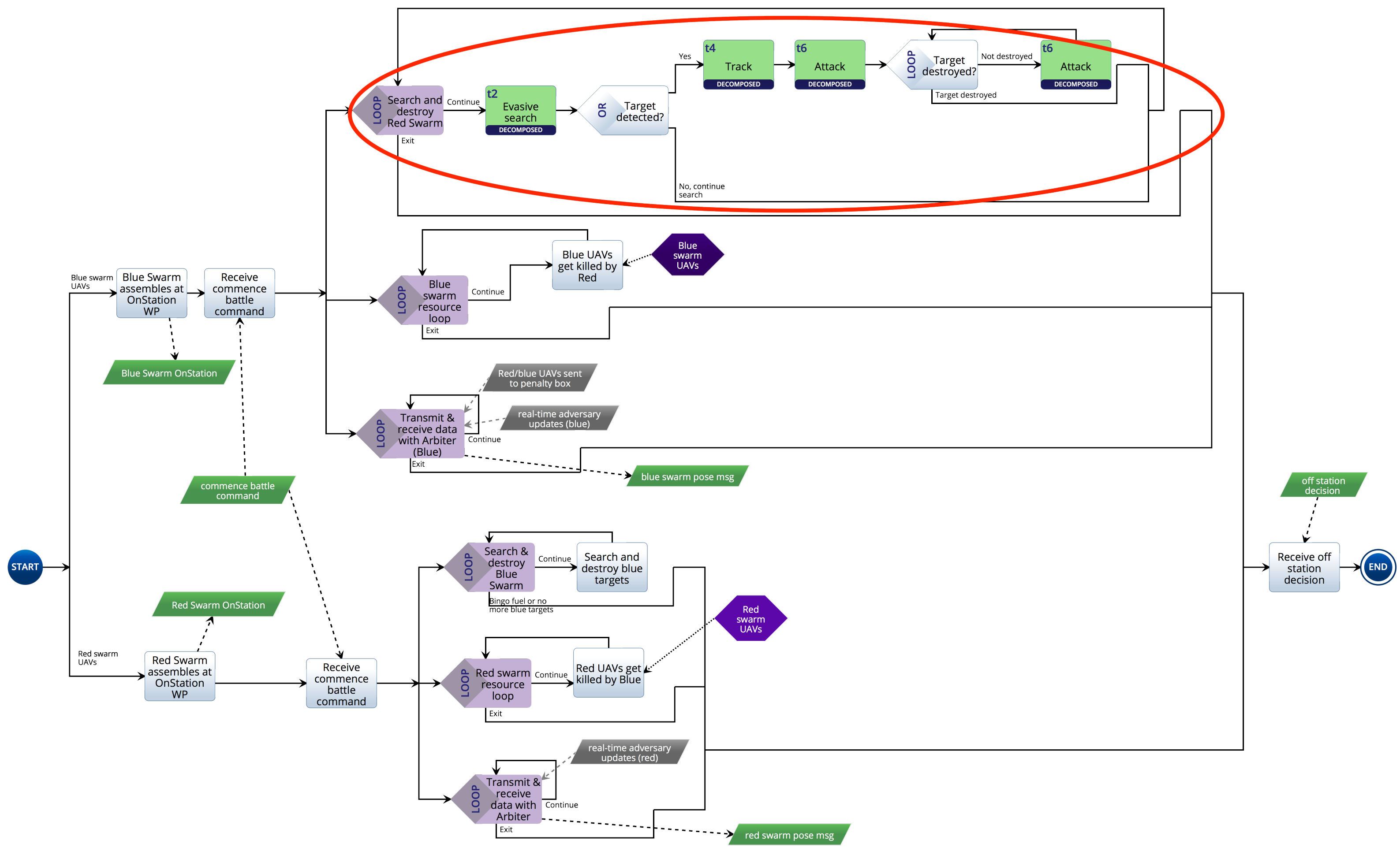



Figure 4.3. Action diagram showing UAV swarms operating in the SvS mission during the OnStation phase (P.3.SVS.1). The red circle highlights the tactics being executed by the swarm (developed in Innoslate).

### 4.2.2 SvS Mission Depicted as FSM

Figure 4.4 is a tactics-level state machine diagram describing the current ARSENL UAV swarm system's operations using the MASC framework within the SvS scenario. The initial state ($s_0$), "on deck and preflighted," is represented by the solid black circle. The final state, ($F$), is just "on deck and recovered" as this model does not include error states. The set of states ($S$) includes the tactics `Ingress, Evasive search, Attack, Track`, and `Egress` (represented by green boxes), and the Swarm Ready and Landing states. The inputs ($\Sigma$) are depicted in natural language describing the transition that occurs between the states. Finally, the transition functions ($\delta : Sx\Sigma \Rightarrow S$) are the mappings of inputs to original states which result in a subsequent state change. For example, *δ(Evasive search, Target detected)* = *Track*. For this research-based case, the Arbiter (described in Appendix B) assesses the battle damage. In this scenario, the swarm acts as an expendable system and does not attempt evasion of the adversary. This type of ROE may be desirable in situations wherein the UAV swarm is providing close-in defense of a high-value asset.

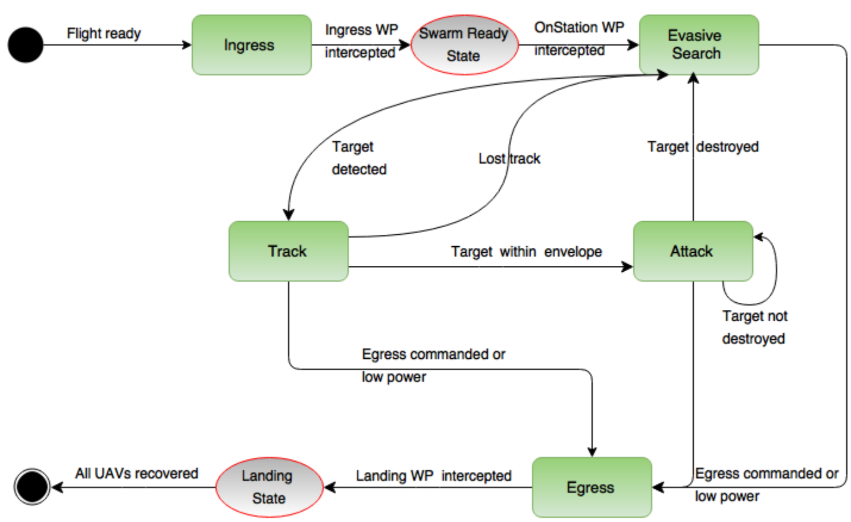



Figure 4.4. State machine diagram for UAV sub-swarm operating in the SvS mission during the OnStation phase. The green boxes are tactics modeled as states. Two additional non-tactics states, Swarm Ready and Landing, are shown as red ovals.

Figure 4.5 is a tactics-level state machine diagram describing a modification to the current ARSENL UAV swarm configuration (Figure 4.4). The `Evade` and `BDA` tactics (depicted as states) are added to improve the survivability and efficiency of the swarm's operation. If the swarm senses it is threatened, it evades until it no longer senses a threat, then returns to the `Evasive search` tactic. The `BDA` tactic directs the swarm to assess the success of the attack to determine if additional attacks are required.

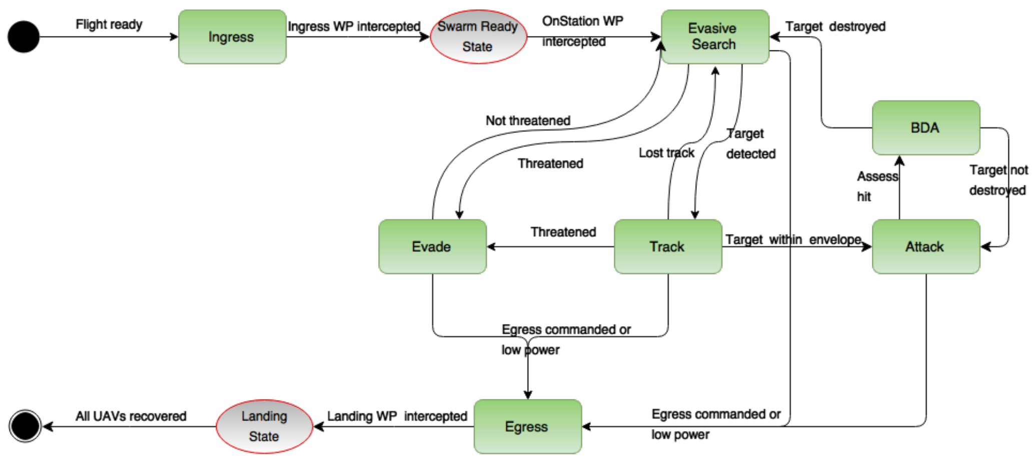



Figure 4.5. State machine diagram for UAV sub-swarm operating in the SvS mission during the OnStation phase with additional tactics: `BDA` and `Evade`.

### 4.2.3 Using MP to Shape the Development of MASC in the SvS Mission

The early MP SvS models included each member of the swarm team, the Arbiter, Range Control, the swarm itself, and the environment as roots (green boxes at the top) composing the respective behaviors (in blue) as shown in by the use case example in Figure 4.6 which is illegible due to the large number of roots and interactions cluttering the diagram. While this approach was useful for showing an overall synopsis of the human-swarm interactions of current ARSENL field operations, it proved to be too large in scope—producing 48 events in Scope 1 and 378 events in Scope 2—and distracted the emphasis away from the desired focus: the swarm tactics. This approach served as a reminder to reduce the number of unnecessary interactions where possible, by taking advantage of autonomy in the swarm architecture.

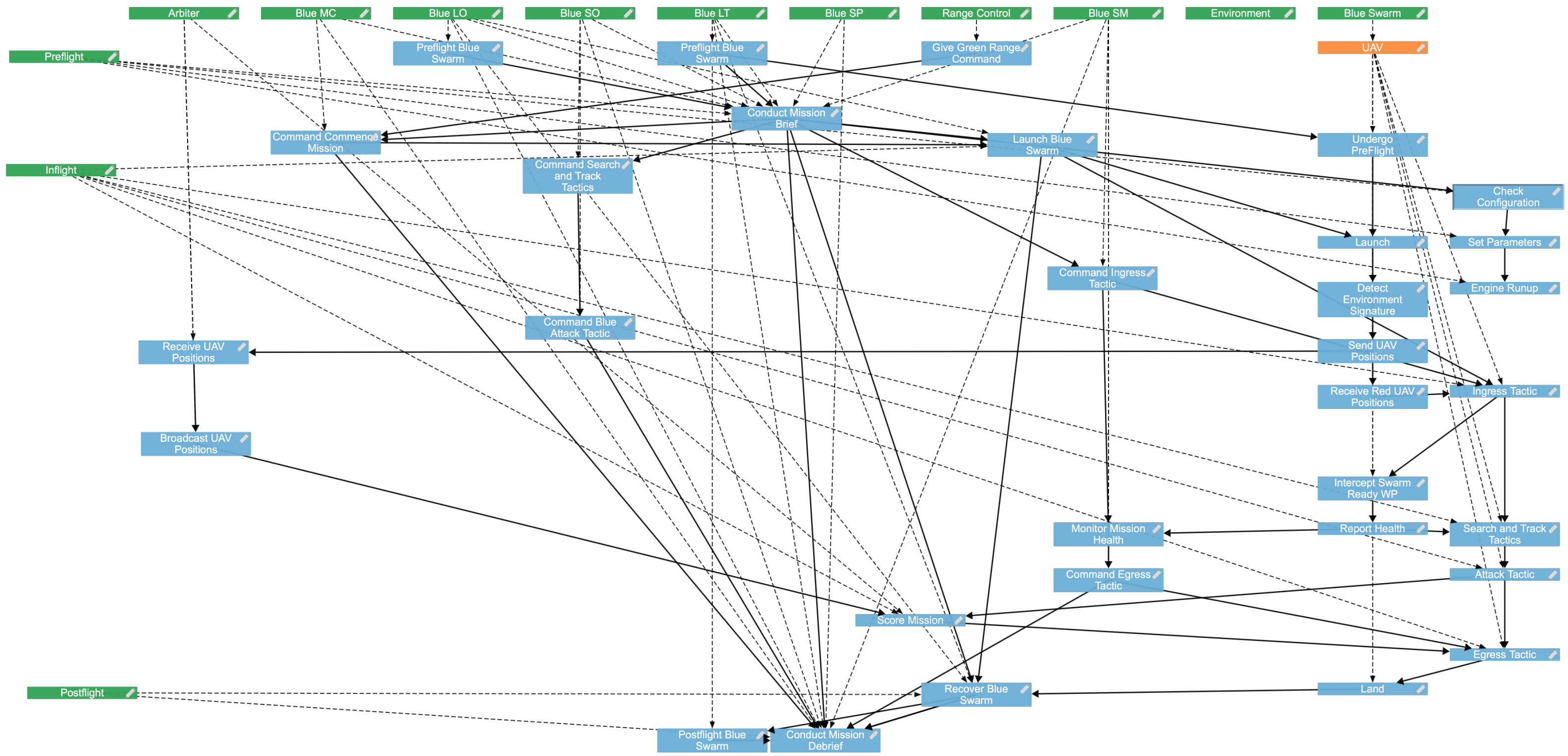



Figure 4.6. This SvS MP model is over-scoped and illegible due to the excessive number of root performers (ten) and phases (three) included in a single model (see code in Figure F.1).

Later models, reduced in scope, focused on the interactions between the opposing swarms, or the combinations of possible tactics and plays. Figure 4.7 shows one of the 650 possible use cases (in Scope 2) for a simple Blue Swarm versus Red Swarm encounter in which each side performs A-A or air-to-ground (A-G) attacks which result in hits or misses; each side earns five points for an A-A hit.

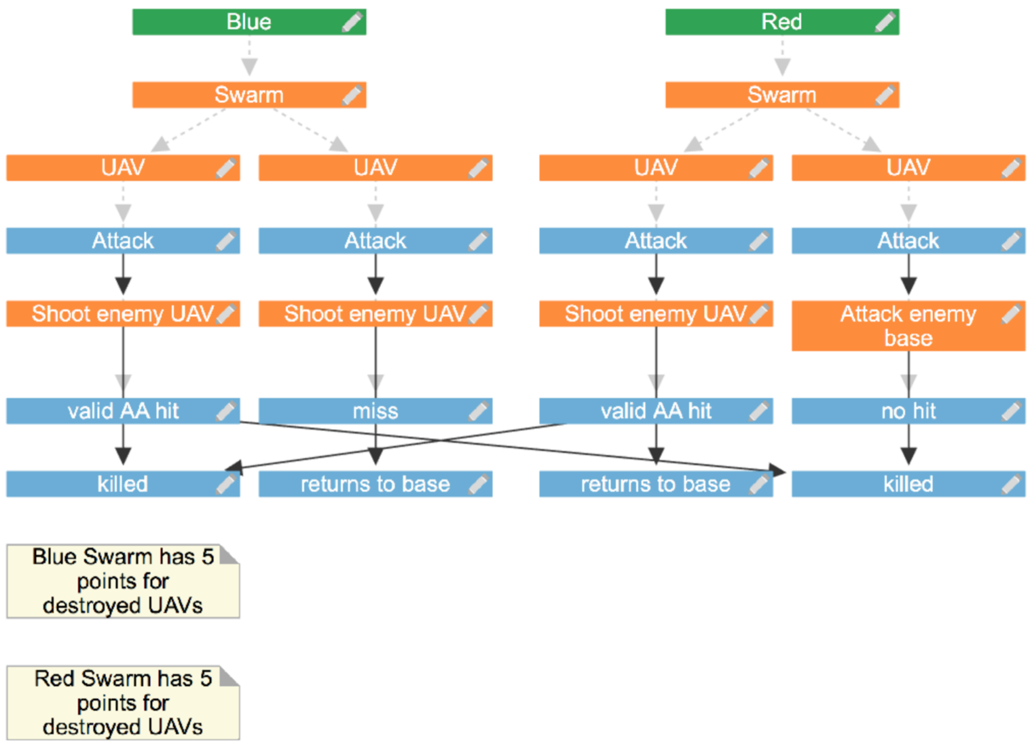



Figure 4.7. An example use case for an MP model that scores hits and misses for a Blue Swarm versus Red Swarm battle (see code in Figure F.2).

Due to its capability to generate an exhaustive set of use cases, MP proved to be useful for identifying undesired behaviors or logical flaws in transitions between tactics by separating the root behaviors from their interactions. Figure 4.8 shows an event trace from a model containing Blue and Red Swarms and the Arbiter. The Blue Swarm is designed to behave in an automated mode according to its simple tactics. This model characterizes the swarm as a root event and its tactics as composite events that include the plays. For this iteration, the model was run at the tactics level with the plays commented out in the code (Figure F.3) to reduce the computing time. Scope 2 generated 162 use cases; however, 152 of the cases were marked with undesired behavior—the Blue Swarm initiated the `Egress` tactic without completely destroying all of the Red UAVs. Instead of a "hit" triggering the `Egress` tactic, the code was be modified to ensure that the number of Red UAVs equals the number of hits before the Blue Swarm is allowed to initiate the `Egress` tactic (Figure F.4). Table 4.1 is a summary of the number of MP use cases generated and associated computation times for the revised SvS mission for tactics only and with plays enacted. By running both the plays and tactics, additional use cases were generated by the three targeting option plays included in the `Attack` tactic.

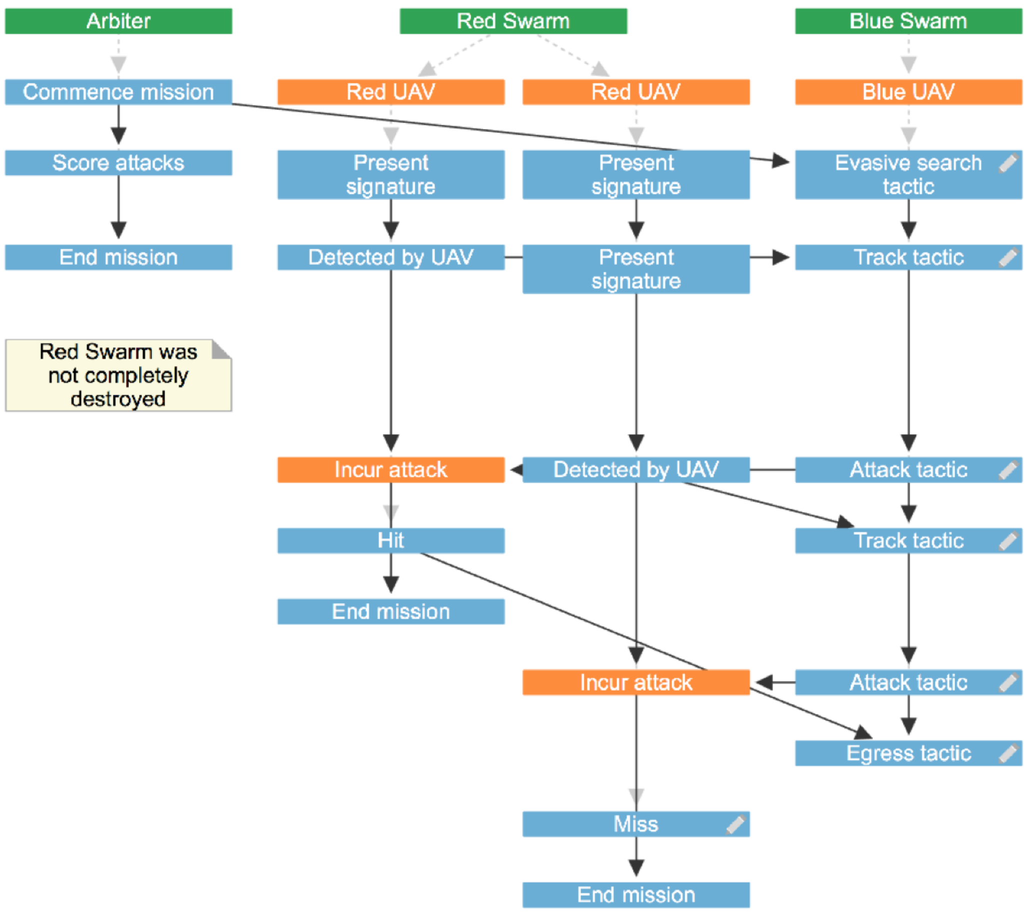



Figure 4.8. SvS OnStation MP use case with root performers: Arbiter, Red Swarm, and Blue Swarm (see code in Figure F.3).

Table 4.1. Table summarizing the number of MP generated use cases and associated computation times for the SvS mission.

| **Configuration** | **Scope 1 Use cases** | **Time (sec)** | **Scope 2 Use cases** | **Time (sec)** |
|---|---|---|---|---|
| Tactics only | 1 | 0.01 | 28 | 3.92 |
| Tactics & plays | 3 | 0.02 | 216 | 263 |

**SvS Mission Modeled as FSM in MP**

The SvS mission scenario requires the swarm to cycle between tactics during operations, which presented a modeling structure challenge because MP does not permit implicit or explicit recursion in event grammar rules; it is designed to display each possible scenario on a time line [90]. Each event must have unique occurrences; therefore, the conventional MP modeling approach with the swarm as a root event does not allow for cycling between tactics. For this reason, the finite state transition diagram was used to depict the swarm's behaviors as transitions between states in MP using the technique described in [90].

Using this alternate approach, an FSM was modeled in MP (Figure 4.10) from Figure 4.5 that focused on the states (tactics) between Swarm Ready and `Egress` (Swarm Ready is not a tactic, rather a swarm state used for model simplification as an initial state following `Ingress`). In the model, each state's behavior is captured by a root event (in green) and transitions between states are modeled as composite events (in orange). Within each root event, all of the valid paths into the state (tactic) are denoted, followed by the state itself, which is then followed by all of the valid paths out of the state. The MP code is shown in Figure 4.9.

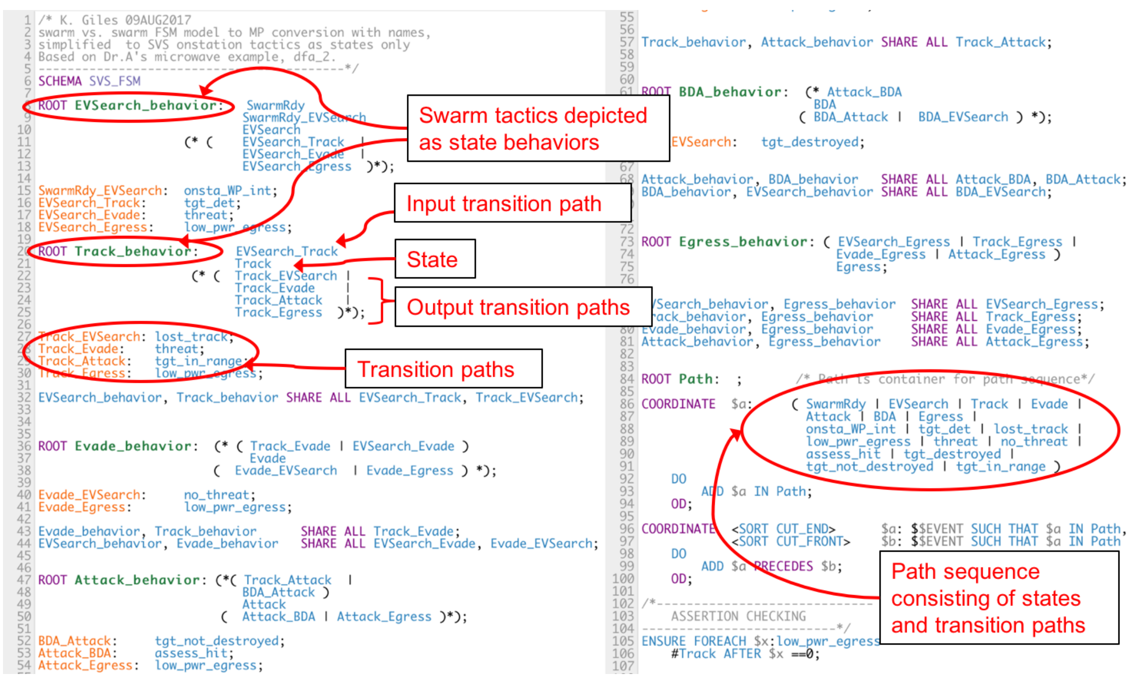



Figure 4.9. MP code for the SvS mission modeled as an FSM. Each tactic is modeled as a root.

Within the MP generated use case (Figure 4.10), the blue boxes represent the sequence of tactics executed and the transitions between them. Figure 4.10 depicts one of two use cases discovered in Scope 2 with a potential undesired behavior pattern—the swarm is able to execute the `Track` tactic after a "low power egress" transition instead of proceeding straight to `Egress`. Revill made a similar discovery using the more traditional activity modeling approach [88]. For a scenario involving the protection of a high value unit, ROE may dictate the swarm be used as an expendable asset, permitting it to continue its mission on low power. Another scenario may prioritize the survivability of the swarm and avoidance of this pattern. For that case, an ENSURE command was added to the MP model to prioritize the survivability of the swarm, reducing the number of use cases in Scope 2 from eight to six (Figure 4.9). Figure 4.11 illustrates an acceptable use case generated in Scope 2 with the constraint in place. The FSM method in MP was useful for generating the full range of state transitions and identifying potential undesired behaviors.

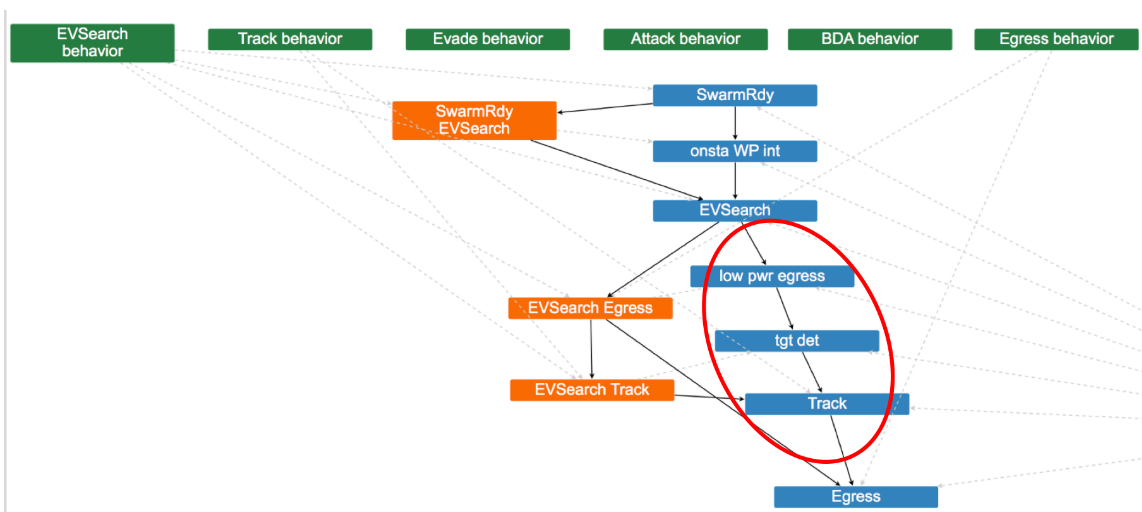


Figure 4.10. MP use case showing a potential undesired behavior in during the SvS mission OnStation phase (modeled as an FSM). The operator may not want the swarm to execute `Track` after the "low power egress" trigger is initiated (code in Figure 4.9).

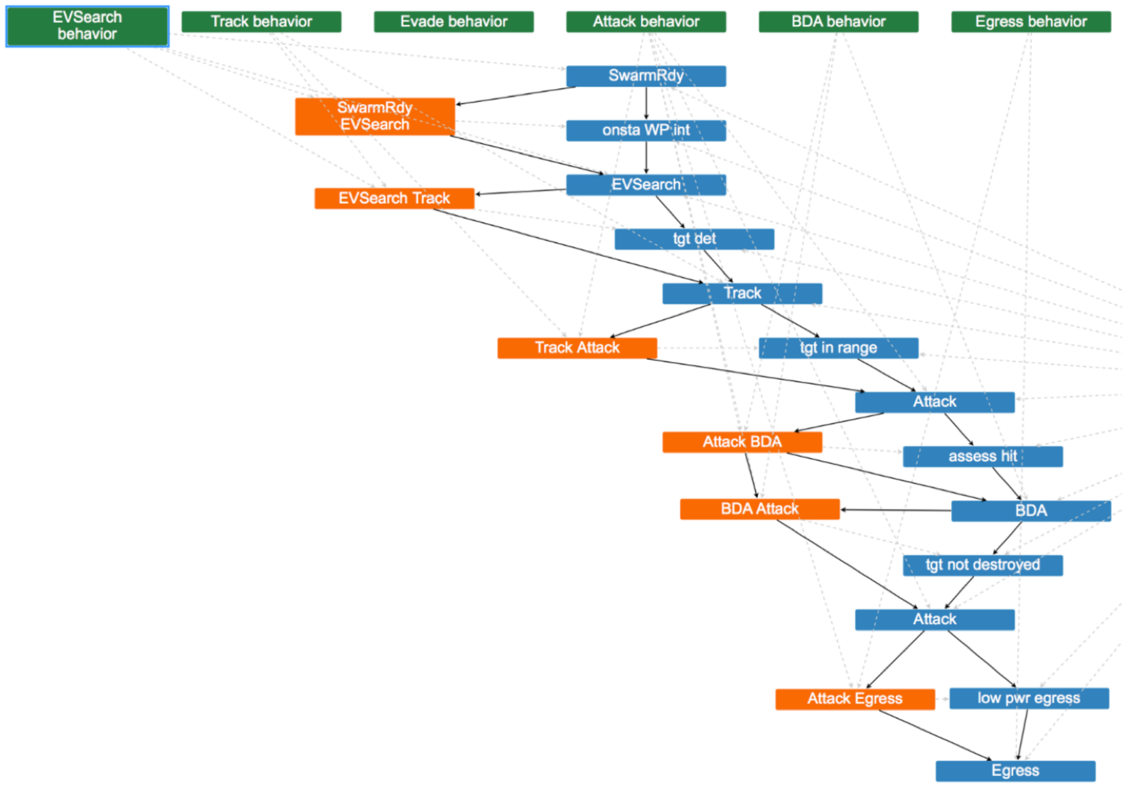



Figure 4.11. An acceptable MP use case for the SvS mission modeled as an FSM (code in Figure 4.9).

Table 4.2 shows the summary composition of the SvS mission using the MASC framework. The mission is composed of seven available tactics, which are generated using 15 different plays. Several of the plays (`Sensors ON` and `Transit to waypoint`) are used multiple times for different tactics.

Table 4.2. Table showing the composition of the SvS mission in terms of its available tactics, plays, and algorithms.

| Tactic ID | Tactic Name | Play ID | Play Name | Algorithm ID | Algorithm Name |
|---|---|---|---|---|---|
| t1 | Ingress | p8.2 | Sensors OFF | a2 | Scheduling |
| | | p1 | Launch | a3 | Sorting |
| | | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p2 | Transit to waypoint | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>Particle Swarm Optimization (PSO) |
| t2 | Evasive search | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p14 | Random pattern | a1.3<br>a8.7 | Levy flight<br>Brownian motion |
| t4 | Track | p8.1<br><br><br>p23 | Sensors ON<br><br><br>Tail following | a2<br>a3<br>a4<br>a11 | Scheduling<br>Sorting<br>Collective consensus<br>Leader-follower |
| t6 | Attack | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p15 | Weapon armed | a12 | Finite state machine (FSM) |
| | | p20 | Smart greedy shooter | a12 | FSM |
| | | p16 | Weapon fire | a12 | FSM |
| t7 | BDA | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p11 | Expanding square pattern | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>PSO |
| t9 | Evade | p19 | Jam | a4<br>a5 | Collective consensus<br>Artificial potential fields |
| | | p6 | Join | a3<br>a5 | Sorting<br>Artificial potential fields |
| | | p7 | Disperse | a5 | Artificial potential fields |
| t15 | Egress | p2 | Transit to waypoint | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>PSO |
| | | p9.1 | Terminal approach | a3 | Sorting |
| | | p9.2 | Landing | a3 | Sorting |
| | | p8.2 | Sensors OFF | a2 | Scheduling |

## 4.3 MASC Applied to the MIO Mission

### 4.3.1 MIO Action Diagram and Simulation

Lessons learned from applying MASC to the SvS mission were employed in a fictional MIO mission (described in Section B.3). The activities of the swarm during the OnStation phase are depicted as an LML action diagram in Figure 4.12. Two main parallel branches show the swarm's employment of tactics on the top branch and the swarm's consumption of power in the bottom branch (for Innoslate simulation synchronization). Within the swarm's tactics branch, two parallel activity branches depict the swarm dividing into two sub-swarms and executing their respective tactics. Sub-swarm 1 performs the search and track functions while Sub-swarm 2 maintains contact with the other participating units via the `Communication relay` tactic. Figure 4.12 is a simple example of how to employ a UAV swarm in a MIO mission using swarm tactics. The key pattern of swarm doctrine is the repeated use of the `Divide` tactic to expand the capabilities and range of the swarm, and the `Amass` tactic to re-assemble the swarm prior to the next tactics sequence. This pattern is indicative of the pulse-like tactics in which units repeatedly converge and disperse as described by Edwards and Arquilla [129], [166]. The overall flow for the MIO mission and diagrams of phases other than the swarm OnStation phase are shown in Figures E.9, E.10, E.11, E.12, E.17, and E.18. Action diagrams showing the composition of the tactics are depicted in Appendix D.

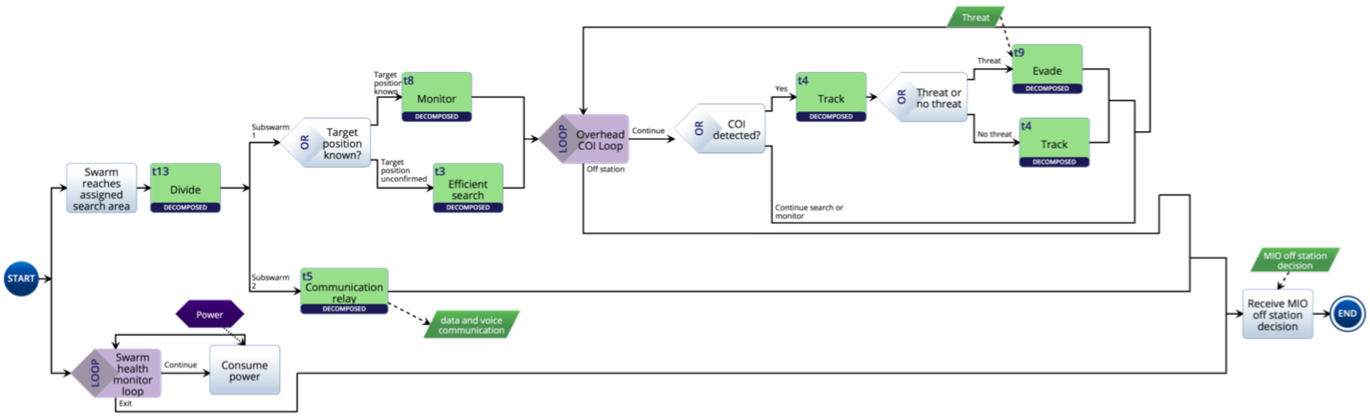



Figure 4.12. Action diagram for a UAV swarm operating in an MIO mission during the OnStation phase (P.3.MIO.1). The upper branch shows the tactics in green and the lower branch captures power consumption by the system (developed in Innoslate).

### 4.3.2 MIO Mission Depicted as FSM

The FSM approach was applied to the MIO mission to describe Sub-swarm 1's operations within the scenario at the tactics level (Figure 4.13). As in the SvS scenario, the initial state ($s_0$) is "on deck and preflighted" and the final state ($F$), is "on deck and recovered." The set of states ($S$) includes the tactics `Ingress`, `Efficient search`, `Evade`, `Track`, `Monitor`, and `Egress` (represented by green boxes), and the Swarm Ready and Landing states. The inputs ($\Sigma$) are depicted in natural language describing the transitions that occur between the states. Finally, the transition functions ($\delta : Sx\Sigma \Rightarrow S$) are the mappings of inputs to original states which result in a subsequent state change. For example, *δ(Track, Threatened)* = *Evade*. In this scenario, the swarm operates in a less autonomous mode than the SvS profile; a target must be confirmed by the Swarm Commander as a valid target for it to be tracked. The ROE in this scenario does not allow for the Attack tactic. In an actual system implementation, the unavailable tactics would be "greyed out" in the GUI.

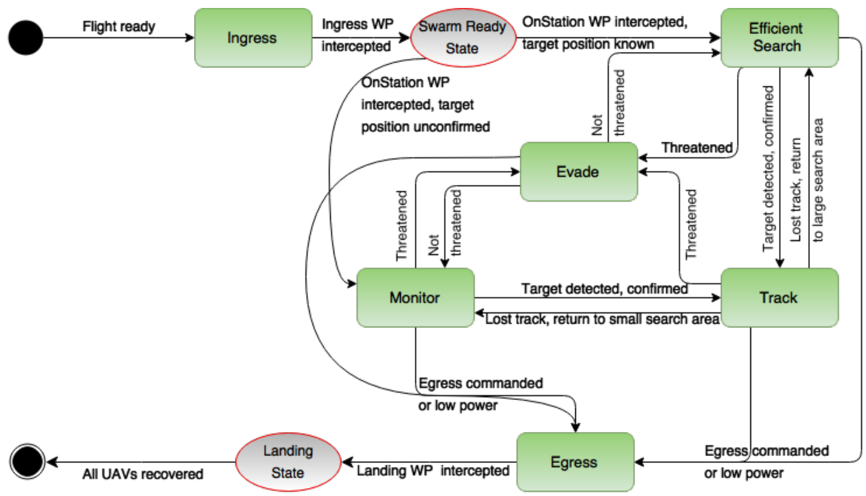



Figure 4.13. State machine diagram depicting tactics used by UAV Sub-swarm 1 operating in the MIO mission.

### 4.3.3 Using MP to Shape the Development of MASC in the MIO Mission

Once again, MP proved to be useful for identifying potential undesired behaviors or logical flaws in transitions between tactics. Figure 4.14 shows an event trace from a model containing the roots: contact of interest (COI), a Swarm Commander, and a UAV swarm depicted as Sub-Swarm 1 and Sub-Swarm 2. This model characterizes the sub-swarms as root events that behave according to their assigned tactics. The tactics are composite events which include the plays. For this iteration, the model was run at the tactics level with the plays commented out in the code (see Figure F.5) to reduce the computing time and number of scenario variants. Scope 2 generated 11 use cases; however, there were two potential cases of undesired behavior in which Sub-Swarm 1 initiated `Track` after `Evade` while the COI was still threatening the swarm. The code could be modified such that the Sub-Swarm 1 only initiates the `Track` tactic (in cases following `Evade`) if it is not threatened by the COI. On the other hand, over-constraining the interaction rules between the tactics or between the roots resulted in desired uses cases not being generated. MP provided a quick and effective means to experiment with different options for interactions that could be fed back into the Innoslate simulation model.

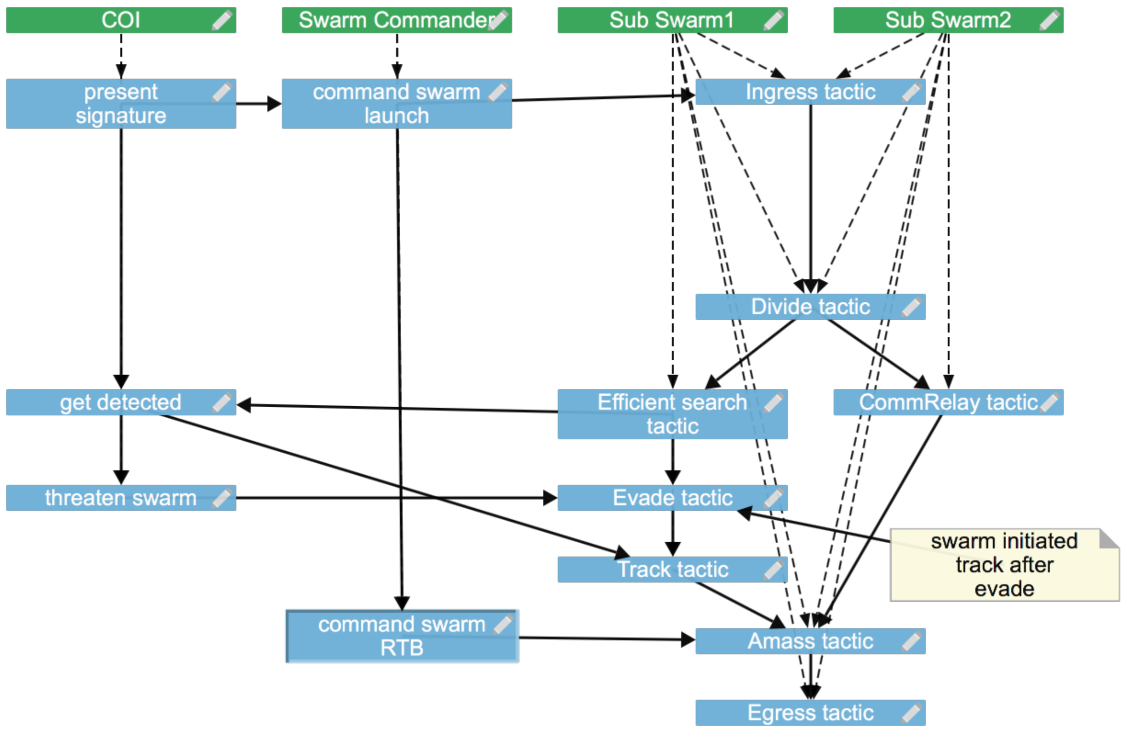



Figure 4.14. MP use case depicting interactions between contact of interest, Swarm Commander, Sub-swarm 1, and Sub-swarm 2 during the MIO mission. Assertion checking found a possible case of undesired behavior in which Sub-swarm 1 transitions from `Evade` to `Track` without confirming the threat is gone (code in Figure F.5).

The MP model was also useful for comparing the state space between operating at the tactics level versus the play level. Executing the MIO MP model with the plays active produced 384 use cases in Scope 1 compared with only 2 in Scope 1 without enacting the plays as shown in Table 4.3). Running the model with plays active above Scope 1 resulted in lengthy computation times due to the number of play options for tactics such as `Efficient search` and `Communication relay`. Considering the number of play options to offer to the user is an important design attribute as it impacts the computation complexity of the swarm system. These results support the assertion that managing a swarm at a higher level of abstraction (swarm tactics) reduces the operational complexity for the Swarm Commander.

Table 4.3. Table summarizing the number of MP generated use cases and associated computation times for the MIO mission.

| Configuration | Scope 1 Use cases | Time (sec) | Scope 2 Use cases | Time (sec) |
|---|---|---|---|---|
| Tactics only | 2 | 0.02 | 11 | 0.26 |
| Tactics & plays | 384 | 261 | 3072 | 6254 |

Table 4.4 shows the summary composition of the MIO mission using the MASC framework. This specific MIO scenario is composed of nine tactics, which are generated using 18 different plays. Several of the plays (`Sensors ON`, `Transit to waypoint`, and `Join`) are used multiple times for different tactics. There are other possible tactics and plays which could have been selected for this mission. For example, in tactics selection, Evasive search could have been used in place of `Efficient search` and `Deter` may have been used instead of `Monitor`. Regarding plays, `Sensors EMCON` is an alternative to `Sensors ON` for the `Ingress` tactic and `Ladder pattern` is one of four available play options under the `Efficient search` tactic (see Figure D.3. Subtle differences in mission ROE or a Swarm Commander's interpretation of the operational scenario warrants a variety of valid mission plans.

Table 4.4. Table showing the composition of the MIO mission in terms of its available tactics, plays, and algorithms.

| Tactic ID | Tactic Name | Play ID | Play Name | Algorithm ID | Algorithm Name |
|---|---|---|---|---|---|
| t1 | Ingress | p8.2 | Sensors OFF | a2 | Scheduling |
| | | p1 | Launch | a3 | Sorting |
| | | p8.1 | Sensors ON | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p2 | Transit to waypoint | a1 | Flocking |
| | | | | a1.1 | Boid's |
| | | | | a1.2 | Particle Swarm Optimization (PSO) |
| t13 | Divide | p5 | Split (logic-based) | a3 | Sorting |
| | | p8.1 | Sensors ON | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p3 | Orbit | a1 | Flocking |
| | | | | a1.1 | Boid's |
| | | | | a1.2 | PSO |
| t3 | Efficient search | p8.1 | Sensors ON | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p10 | Ladder pattern | a1 | Flocking |
| | | | | a1.1 | Boid's |
| | | | | a1.2 | PSO |
| t8 | Monitor | p8.1 | Sensors ON | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p4 | Racetrack | a1 | Flocking |
| | | | | a1.1 | Boid's |
| | | | | a1.2 | PSO |
| t5 | Communication relay | p8.1 | Sensors ON | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p8.4 | Transmit video | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p13 | Grid pattern | a1.3 | Levy flight |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | | | a8.7 | Brownian motion |
| | | p18 | Forward communication | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| t4 | Track | p8.1 | Sensors ON | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p17 | Follow target | a6 | Firefly algorithm |
| | | | | a8.3 | Ant colony optimization |
| | | | | a5 | Artificial potential fields |
| t9 | Evade | p8.3 | EMCON | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p6 | Join | a3 | Sorting |
| | | | | a5 | Artificial potential fields |
| | | p7 | Disperse | a5 | Artificial potential fields |
| t14 | Amass | p6 | Join | a4 | Collective consensus |
| | | | | a5 | Artificial potential fields |
| | | p8.1 | Sensors ON | a2 | Scheduling |
| | | | | a3 | Sorting |
| | | | | a4 | Collective consensus |
| | | p3 | Orbit | a1 | Flocking |
| | | | | a1.1 | Boid's |
| | | | | a1.2 | PSO |
| t15 | Egress | p2 | Transit to waypoint | a1 | Flocking |
| | | | | a1.1 | Boid's |
| | | | | a1.2 | PSO |
| | | p9.1 | Terminal approach | a3 | Sorting |
| | | p9.2 | Landing | a3 | Sorting |
| | | p8.2 | Sensors OFF | a2 | Scheduling |

## 4.4 MASC Applied to the HADR Mission

### 4.4.1 HADR Action Diagram and Simulation

Lessons learned from applying MASC to the SvS and MIO missions were employed in a fictional HADR mission (detailed in Section B.2). The activities of the swarm during the OnStation phase are depicted as an LML action diagram in Figure 4.15. Two main parallel branches show the swarm's employment of tactics in the top branch and the swarm's consumption of power in the bottom branch (for Innoslate simulation synchronization). Within the swarm's tactics branch, two parallel activity branches depict the swarm dividing into two sub-swarms and executing their respective tactics. Sub-swarm 1 performs the searching, tracking, and monitoring functions while sub-swarm 2 maintains contact with the other participating units via the `Communication relay` tactic. Figure 4.12 is a simple example of how to employ a UAV swarm in a HADR mission using swarm tactics. A natural extension of this pattern to support swarm doctrine for wide-area search is to use multiple sub-swarms to conduct reconnaissance over different areas and at varying altitudes. This becomes especially advantageous in regions with varying terrain (such as Haiti) where widely dispersed sub-swarms assigned to relay communications supports improved connectivity between coordinating units. The overall mission flow for the HADR mission and diagrams of the phases other than the OnStation phase are shown in Figures E.19, E.20, E.21, E.22, E.26, and E.27. Action diagrams showing the composition of the tactics are depicted in Appendix D.

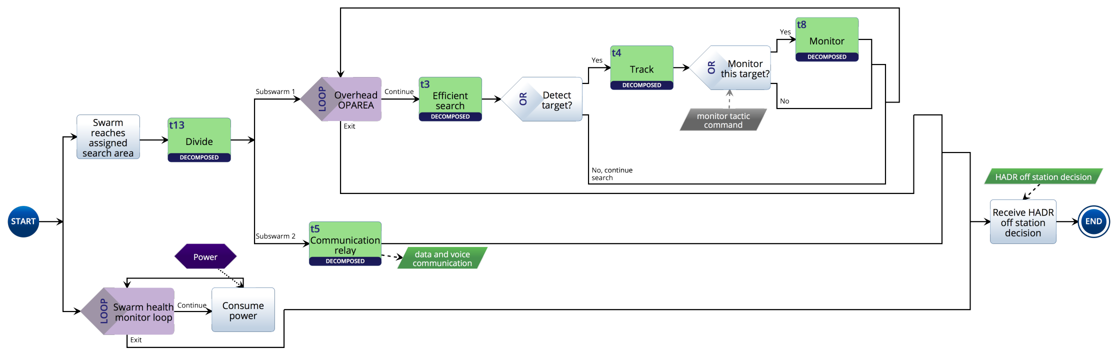



Figure 4.15. Action diagram for a UAV swarm operating in an HADR mission during the OnStation phase (P.3.HADR.1). The upper branch shows the tactics in green and the lower branch captures power consumption by the system (developed in Innoslate [165]).

### 4.4.2 HADR Mission Depicted as FSM

The HADR mission can be described using the FSM approach. Figure 4.16 is a tactics-level state machine diagram describing the Sub-swarm 1's operations within the HADR scenario using the MASC framework. As in the SvS scenario, the initial state ($s_0$) is "on deck and preflighted" and the final state ($F$), is "on deck and recovered." The set of states ($S$) includes the tactics `Ingress, Efficient search, Track, Monitor`, and `Egress` (represented by green boxes), and the Swarm Ready and Landing states. The inputs ($\Sigma$) are depicted in natural language describing the transition that occurs between the states. Finally, the transition functions ($\delta : Sx\Sigma \Rightarrow S$) are the mappings of inputs to original states which result in a subsequent state change. For example, *δ(Efficient Search, Target detected) = Track* describes the automatic transition from `Efficient search` to `Track` tactic if a potential target of interest (TOI) is detected. Similar to the MIO scenario, the swarm operates in a less autonomous mode than the SvS profile in that a target must be confirmed by the Swarm Commander as "of interest" for it to be monitored. The ROE in this scenario does not enable the `Evade` tactic as the criticality of locating survivors supersedes the risk of an adversary attacking the swarm.

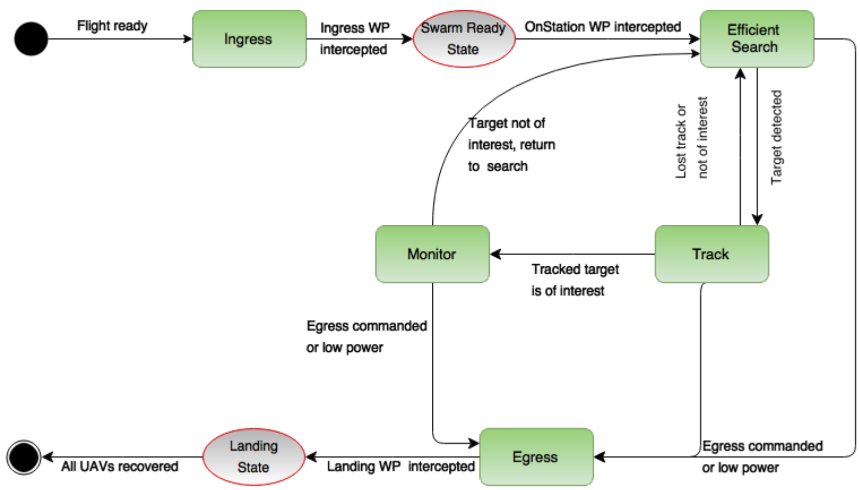



Figure 4.16. State machine diagram depicting tactics used by UAV Sub-swarm 1 operating in the HADR mission.

### 4.4.3 Using MP to Shape the Development of MASC in the HADR Mission

The MP model facilitated the simplification of tactics selected for the HADR mission. Figure 4.17 shows an event trace from a model containing the roots: TOI, a Swarm Commander, and a UAV swarm depicted as Sub-Swarm 1 and Sub-Swarm 2. This scenario assumes the Swarm Commander is in direct communication with Sub-Swarm 1 conducting reconnaissance on the TOI while Sub-Swarm 2 is being used to relay communications during the entire mission to beyond line-of-sight units. This model characterizes the sub-swarms as root events that behave according to their assigned tactics. To overcome the recursion restriction inherent to MP, a different approach was used from the MIO MP model. The `Efficient search` tactic was coded as "Resume Efficient Search" and "Continue Efficient Search" for subsequent instances as a workaround to depict multiple instances of cycling between tactics. The tactics—except for the resume and continue modes—are composite events which include the plays. For this iteration, the model was run at the tactics level with the plays commented out in the code (Figure F.6) because we are interested in operating the system at the tactics level. With the plays included, Scope 2 generates 96 use cases (Table 4.5).

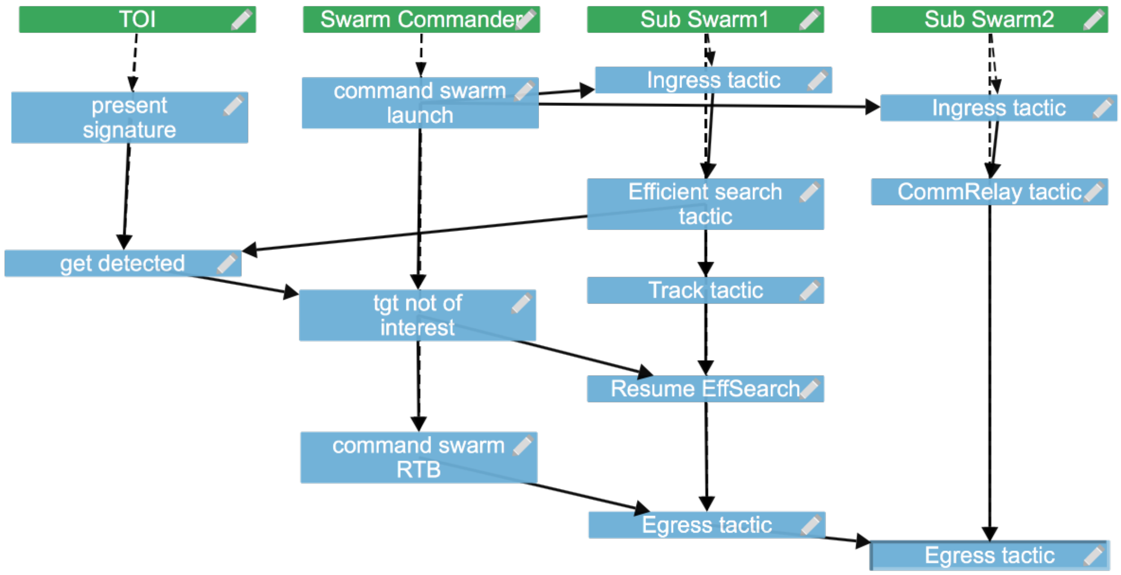



Figure 4.17. MP use case for the HADR mission showing interactions between target of interest, Swarm Commander, Sub-swarm 1, and Sub-swarm 2 (code in Figure F.6).

Table 4.5. Table summarizing the number of MP generated use cases and associated computation times for the HADR mission.

| **Configuration** | **Scope 1 Use cases** | **Time (sec)** | **Scope 2 Use cases** | **Time (sec)** |
|---|---|---|---|---|
| Tactics only | 3 | 0.01 | 3 | 0.01 |
| Tactics & plays | 96 | 5.4 | 96 | 5.4 |

Scope 2 generates just three use cases at the tactics level due to the considerable list of coordinate statements which by nature, restrict the output of use cases. User inspection of the use cases could prompt the system architect to restructure the interactions between roots to expand this set of generated use cases. Monterey Phoenix provided an effective way to consider different degrees of automation based on the number of required interactions between roots. Balancing the number of play options to offer the user with system computation complexity is an important design consideration. The MP model inspired revisions that were fed back into the overall HADR model in Innoslate. For example, MP raised some usability related questions into handling the transition of swarm tactics when the swarm detects a target: should the swarm automatically begin tracking a target and prompt the Swarm Commander to determine if the target is "of interest"? Should the swarm automatically return to `Efficient search` if the target is not of interest? Should `Track` be a play instead of a tactic? This question was debated several times by the author, and eventually, `Track` was categorized as a tactic for usability reasons and because it supported modularity among the missions. In the future, `Track` may return to being a play and a component of several tactics.

Table 4.6 shows the summary composition of the HADR mission using the MASC framework. The mission is composed of eight tactics, which are generated using 15 different plays. Several of the plays (`Sensors ON, Transit to waypoint`, and `Orbit`) are used multiple times for different tactics.

Table 4.6. Table showing the composition of the HADR mission in terms of its available tactics, plays, and algorithms.

| Tactic ID | Tactic Name | Play ID | Play Name | Algorithm ID | Algorithm Name |
|---|---|---|---|---|---|
| t1 | Ingress | p8.2 | Sensors OFF | a2 | Scheduling |
| | | p1 | Launch | a3 | Sorting |
| | | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p2 | Transit to waypoint | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>Particle Swarm Optimization (PSO) |
| t13 | Divide | p5 | Split (logic-based) | a3 | Sorting |
| | | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p3 | Orbit | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>PSO |
| t3 | Efficient search | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p10 | Ladder pattern | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>PSO |
| t4 | Track | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p23 | Tail following | a11 | Leader-follower |
| t8 | Monitor | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p4 | Racetrack | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>PSO |
| t5 | Communication relay | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p8.4 | Transmit video | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p13 | Grid pattern | a1.3<br>a3<br>a4<br>a8.7 | Levy flight<br>Sorting<br>Collective consensus<br>Brownian motion |
| | | p18 | Forward communication | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| t14 | Amass | p6 | Join | a4<br>a5 | Collective consensus<br>Artificial potential fields |
| | | p8.1 | Sensors ON | a2<br>a3<br>a4 | Scheduling<br>Sorting<br>Collective consensus |
| | | p3 | Orbit | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>PSO |
| t15 | Egress | p2 | Transit to waypoint | a1<br>a1.1<br>a1.2 | Flocking<br>Boid's<br>PSO |
| | | p9.1 | Terminal approach | a3 | Sorting |
| | | p9.2 | Landing | a3 | Sorting |
| | | p8.2 | Sensors OFF | a2 | Scheduling |

## 4.5 Modeling Heuristics Application Assessment

After applying MASC to the three missions, the Innoslate models and MASC framework (as applicable) were verified against the heuristics in Section 4.1.

1. Within MASC, each element below mission (phase, tactic, play, algorithm) has at least one parent. The Innoslate models are constructed so that each element has at least one parent.
2. The framework violates heuristic #2 in that the Ingress phase contains only one tactic, the Ingress tactic. Similarly, the Preflight phase does not contain any tactics. However, this was deemed permissible because the Preflight phase is not one of the inflight phases. Furthermore, the Innoslate models comply, as they also contain the activities of other players in the scenario.
3. Both the MASC framework and the models comply with heuristic #3; however, one might argue that the Ingress and Egress phase names should be changed as they each contain a tactic of the same name.
4. The Innoslate models were assembled so that each branch contained the activities of a "performer" (i.e., swarm, ground crew, environment). If more than seven performers were needed for a particular phase, the activities were abstracted to a higher level and decomposed on another diagram. This technique was useful for improving diagram readability.
5. Each activity, except for the initiating activity, within the Innoslate models was prompted by at least one input or trigger.

## 4.6 Summary

This chapter applied the MASC components (defined in Chapter 3) to three mission case studies to demonstrate their capability to cover a variety of missions. From the design reference missions (DRMs) described in Appendix B, FSMs depicted swarm behavior at the tactics level, mission simulations were developed and checked for logical errors using Innoslate and MP. Chapter 5 continues the process with an evaluation of MASC's suitability for formalizing a swarm system operational framework.

# CHAPTER 5: MASC Evaluation

After applying the MASC framework to three notional UAV swarm missions in Chapter 4, MASC is evaluated for its effectiveness in formalizing an operational framework of common patterns in swarm missions that promote architecture reusability. MASC's intuitiveness, modularity, composability, and mission doctrine integration (described in Section 1.4.3) are assessed in this chapter to support the evaluation. In addition to evaluating MASC within the confines of the models and simulations, HSR augments the evaluation by gathering prospective user feedback on the swarm tactics and plays. The purpose of the HSR exercise is to collect feedback from subject matter experts (SMEs) on MASC and to determine if there is a difference in operator perceived workload between mission planning at the tactics level versus at the play level.

The 15 volunteer SME participants, targeted for their maritime aviation experience, were naval aviators and naval flight officers of military rank O-3 to O-5 (Figure 5.1). The participants were randomly assigned to two groups: Group 1 (seven participants) developed a UAV swarm mission plan for a notional MIO scenario using the 16 available swarm tactics, while Group 2 (eight participants) developed a mission plan using only the 27 available swarm plays and without knowledge of the swarm tactics. The null hypothesis ($H_o$) was: there is no difference in workload or cognitive effort between Group 1 and Group 2. The alternate hypothesis ($H_A$) was that it would be easier for the participants to develop a mission plan at the tactics level (Group 1) than at the play level (Group 2). The significance level, $\alpha$, used was 0.05 to reject the null hypothesis in favor of the alternate hypothesis. Both groups were given the opportunity to create an "option" tactic or play to develop a swarm action not covered by MASC. All of the SMEs read the same scenario and completed the exercise in less than one hour. Clarifying questions were answered, but no training was conducted in order to promote creative responses from the participants. The scenario information, instructions, and survey given to participants are in Appendix C.

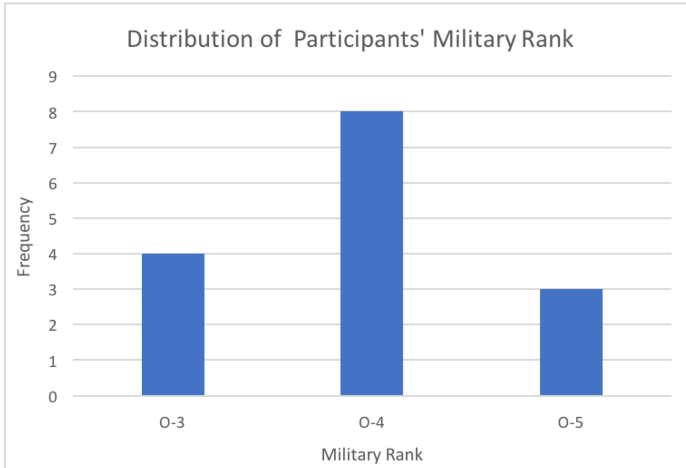


Figure 5.1. All subject matter expert participants were of rank O-3, O-4, or O-5.

## 5.1 Intuitiveness

To assess intuitiveness beyond just the SME participants' task completion time, we reviewed the selection of tactics or plays, their placement within the phases of the mission, and their relation to the mission triggers. Following the swarm mission plan construction, each SME responded to a series of questions designed to support measuring MASC intuitiveness. Quantitative data were collected using the NASA Task Load Index method [167] to assess participant perception of their task workload on a 1-5 scale (Figure 5.2), along with the time required to complete the task (Table 5.1). Participant responses to questions about the phases structure, and type of tactics and plays offered were also reviewed to assess intuitiveness.

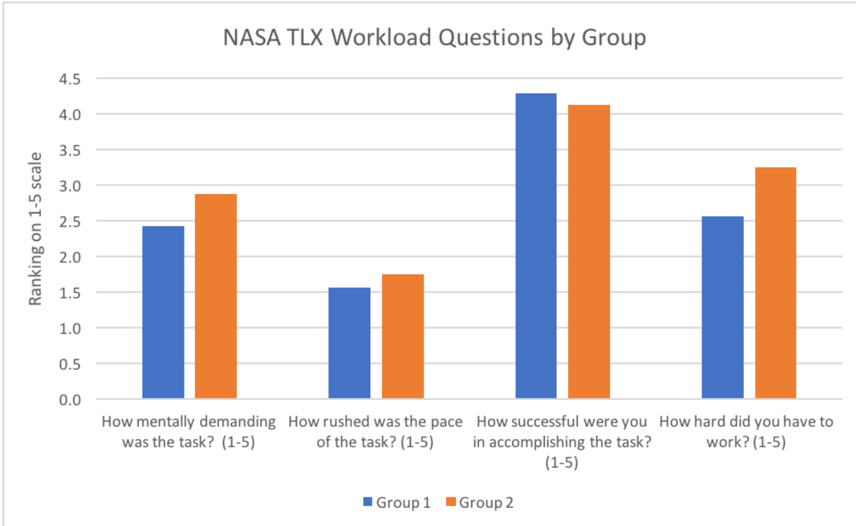



Figure 5.2. Participants answered four workload related questions from the NASA TLX workload scale.

There was no statistically significant difference between the two groups regarding task completion time and perceived workload (Figure 5.2), as evidenced by the p-values in Table 5.1. The participants completed the mission plans in just over 36 and 39 minutes for Group 1 and 2, respectively. Only the perceived level of effort (how hard did you have to work?) indicated a potential difference between the two groups. Both groups reported a mean perceived success of over 4 out of 5 (Table 5.1). A larger sample size may have resulted in stronger evidence to support the alternate hypothesis in terms of completion time and cognitive demand (how mentally demanding was the task?). Future HSR should be conducted using a more interactive prototype system.

Table 5.1. Summary of data collected during human subjects research. Results indicate that there is not a statistically significant difference between the two groups.

| Time to complete mission plan (min) | How mentally demanding was the task? (1-5) | How rushed was the pace of the task? (1-5) | How successful were you in accomplishing the task? (1-5) | How hard did have to you work? (1-5) | Statistical Parameters |
|---|---|---|---|---|---|
| 36.1 | 2.4 | 1.6 | 4.3 | 2.6 | Group 1 sample mean |
| 134.0 | 0.6 | 0.3 | 0.2 | 0.6 | Group 1 sample variance |
| 39.3 | 2.9 | 1.8 | 4.1 | 3.3 | Group 2 sample mean |
| 93.7 | 0.4 | 0.2 | 0.4 | 0.5 | Group 2 sample variance |
| | | | | | |
| -0.6 | -1.2 | -0.7 | 0.6 | -1.7 | $t$-statistic |
| 12.0 | 12.0 | 12.0 | 13.0 | 12.0 | $v$ (df) |
| **0.29** | **0.13** | **0.25** | **0.70** | **0.05** | ***p*-value** (one-sided) |
| | | | | 7 | Sample size Group 1 (m) |
| | | | | 8 | Sample size Group 2 (n) |

### 5.1.1 Group 1 Mission Plans

The Group 1 participant mission plans were evaluated for consistency and appropriateness of tactics selection. Table 5.2 shows the distribution of tactics used in the three missions. The “B” denotes a tactic selected for the mission case study baseline Innoslate model while the integer represents the number of Group 1 HSR participants who selected the tactic for their MIO mission plan. All seven of the participants selected `Ingress, Track, Communication relay, Evade`, and `Divide` and used them at appropriate times in the scenario. The participants were split between choosing `Evasive` or `Efficient search` and selected tactics such as `Harass, Defend, Deter`, and `BDA` that were not used in the baseline model. `Harass, Deter`, and `BDA` were appropriately used; however, `Defend` was not a valid tactic for this scenario based on the ROE. All but one of the Group 1 participants used the `Egress` tactic appropriately, as expected. Similarly, all but one preceded `Egress` with the `Amass` tactic. Each Group 1 SME chose `Evade` following the COI shots and no SME selected `Attack`, which was available for selection but not authorized per the scenario ROE. Apart from the two exceptions noted, the Group 1 participants selected tactics appropriate for the scenario.

Table 5.2. Table showing the distribution of tactics used across the three mission case studies. The "B" indicates a tactic used in the Innoslate baseline model. The numbers in the fourth column indicate the number of Group 1 SME participants who selected the tactic during the HSR exercise.

| Tactic | SvS | MIO | MIO HSR | HADR |
|---|---|---|---|---|
| Ingress | B | B | 7 | B |
| Evasive search | B | | 4 | |
| Efficient search | | B | 5 | B |
| Track | B | B | 7 | B |
| Communication relay | | B | 7 | B |
| Attack | B | | | |
| BDA | B | | 3 | |
| Monitor | | B | 7 | B |
| Evade | B | B | 7 | |
| Harass | | | 3 | |
| Defend | | | 1 | |
| Deter | | | 2 | |
| Divide | | B | 7 | B |
| Amass | | B | 6 | B |
| Egress | B | B | 6 | B |
| ACM | | | | |
| Option | | | 1 | |

B = selected for baseline mission case study (Innoslate model)
\# = number of HSR participants who selected tactic

Participant responses to the survey indicate that MASC was understandable and supported the MIO scenario. Responses to survey question 7 indicated that for the most part, the SMEs thought the phases supported mission execution. One SME recommended the OnStation phase be broken into additional mission phases that would "allow for better planning and be more adaptable." Several SMEs provided feedback on the tactics themselves (question 8). Two of the suggested tactics, "Disguise RHIB" and "Disrupt Fire," closely resemble the existing `Deter` tactic. However, using the swarm as a visual shield for camouflage was not considered in the existing playbook. The concept for a "Suicide Attack" tactic allowing for a swarm attack in self-defense for cases in which ROE preclude armed UAVs should be considered as an addition to the MASC playbook. One participant desired user-specified boundaries to be built into the tactics. Those attributes currently exist as play parameters and were not emphasized in the brief description of the HSR exercise. Additional training in future HSR could enable better understanding of the MASC elements. Similarly, HSR using an interactive prototype would enable more focus on swarm system GUI.

### 5.1.2 Group 2 Mission Plans

The Group 2 participant mission plans were evaluated for consistency and appropriateness of play selection. Table 5.3 shows the distribution of plays used in the three missions. The "B" denotes a play selected for the mission case study baseline Innoslate model (as an element of its parent tactic) while the integer represents the number of Group 2 HSR participants who selected the play for their MIO mission plan. All eight Group 2 participants used the plays `Launch, Transit to waypoint`, and `Landing` appropriately as expected, while one omitted `Terminal approach`. There were differences in how the `Sensors ON` and `Sensors OFF` plays were used. Most participants launched the swarm assuming that sensors were off, and turned them on after executing the `Launch` play (`Sensors OFF` is part of the `Ingress` tactic); however, in two cases, sensors were not activated until after reaching OnStation. All but one used `Sensors OFF` just before `Landing`. One SME deliberately decided to break the ROE by enacting the `Weapon fire` play for self-defense of the rigid-hull inflatable boat (RHIB). Only one Group 2 participant selected `Jam` which is a standard play option within the `Evade` tactic (selected by all Group 1 SMEs). Finally, another participant opted to use the `Smart greedy shooter` play—knowing the system was unarmed—to "harass the COI" (similar to the `Harass` tactic).

Table 5.3. Table showing the distribution of plays used across the three mission case studies (as elements of parent tactics). The "B" indicates a play used in the Innoslate baseline model. The numbers in the fourth column indicate the number of Group 2 SME participants who selected the play during the HSR exercise.

| Play | SvS | MIO | MIO HSR | HADR |
|---|---|---|---|---|
| Launch | B | B | 8 | B |
| Transit to WP | B | B | 8 | B |
| Orbit | | B | 7 | B |
| Racetrack | | B | 4 | B |
| Split (logic based) | | B | 7 | B |
| Join | B | B | 8 | B |
| Disperse | B | B | 8 | |
| Sensors ON | B | B | 8 | B |
| Sensors OFF | B | B | 7 | B |
| Sensors EMCON | | B | 2 | |
| Transmit video | | B | 8 | B |
| Terminal approach | B | B | 7 | B |
| Landing | B | B | 8 | B |
| Ladder pattern | | B | 2 | B |
| Expanding square pattern | B | B | 2 | B |
| Constricting square pattern | | B | 2 | B |
| Grid pattern | | B | 2 | B |
| Random pattern | B | | 3 | |
| Weapon armed | B | | | |
| Weapon fire | B | | 1 | |
| Follow target | | B | 5 | B |
| Forward communication | | B | 4 | B |
| Jam | | | 1 | |
| Smart greedy shooter | B | | 1 | |
| Patrol box shooter | B | | | |
| Wingman shooter | B | | | |
| Tail following | B | | | |
| Option | | | 5 | |

B = selected for baseline mission case study (Innoslate model)
\# = number of HSR participants who selected play

Participant responses to the survey indicated that MASC was in general, understandable and supported the MIO scenario. Several participants provided feedback on the plays (questions 8, 9). One SME recommended the OnStation phase be broken into additional phases such as Search-Detection and Detection-Collection. This suggestion is worthy of consideration in future swarm system software GUI design. One sought a play to "follow and orbit around a moving target," essentially the Monitor tactic. Another suggested a single drop-down menu with all of the maneuver patterns in one place instead of separate plays for each. This is valid for future GUI design—the plays were parsed out separately to support the MASC Innoslate simulation mechanics. A participant advocated for a "System/communication Check" play for the swarm to execute after arriving OnStation, a likely carryover from typical manned aircraft TTPs. A SME recommended a "Sector" pattern to support a search plan based on operator-modifiable relative headings from a reference position. This suggestion was incorporated into the parameters for swarm pattern plays. Another SME used Sensors EMCON instead of ON after launch to conduct a covert transit to OnStation. The original `Ingress` tactic composition did not support this and was subsequently modified. When specifically asked if pre-defined combinations of plays would make the task easier, seven of eight responded in the affirmative, and two identified `Ingress` and `Egress` combinations specifically. Another SME grouped `Follow target` and `Smart greedy shooter` for a "Harassment" effect much like the `Harass` tactic. Interestingly, one participant thought combinations of plays would make the task easier after TTPs and best practices were identified, while another thought the pre-defined play combinations would make applying the TTPs easier.

## 5.2 Modularity

The MASC framework demonstrated modularity at several levels. The five operational phases—Preflight, Ingress, OnStation, Egress, and Postflight—cover each of the three missions. Tactics such as `Divide`, `Communication relay`, `Efficient search`, and `Track` can be used in more than one mission. To evaluate modularity, we looked to see how the tactics or plays were reused across the three missions and within the mission.

Table 5.2 shows the distribution of tactics used in the three missions. The MIO and HADR missions needed several of the same reconnaissance-type patterns, and therefore shared many of the same tactics. The SvS mission, being the only A-A case study and the only scenario for which weapons were authorized, used the tactics specific for those conditions. The participants were split between choosing `Evasive search` or `Efficient search` and also selected tactics that were not used in the baseline model (`Harass`, `Defend`, `Deter`, and `BDA`). All of the participants chose `Ingress`, `Track`, `Communication relay`, `Evade`, and `Divide` while six of seven selected `Amass` and

`Egress`. Several of the Group 1 participants reused tactics including `Monitor, Track, Deter, Communication Relay`, and `Divide` within their mission plans. All participants divided the swarm into sub-swarms to assign a variety of tactics or to cover different geographic areas.

Table 5.3 shows the distribution of plays used in the three missions. The MIO and HADR missions required common patterns such as searching, tracking, and relaying communication, and therefore shared many of the same plays. As the only A-A offensive mission, SvS used the plays applicable to that operational environment. Within the MIO scenario, the Group 2 participants reused many of the plays including `Transit to waypoint, Sensors ON, Sensors OFF, Orbit, Racetrack, Split`, and `Forward Communication` in their mission plans. Seven of eight participants split the swarm into sub-swarms to assign a variety of plays or to cover different geographic areas; however, only four of eight split the swarm to provide communication forwarding services. Within Group 1, all seven included the `Communication relay` tactic throughout the mission.

## 5.3 Composability

Composability, "the capability to select and assemble simulation components in various combinations into valid simulation systems to satisfy specific user requirements" [29] is an important architecture design element for promoting reusability. Semantic composability, whether the models within the simulations can be "meaningfully composed" [29] into a semantically logical simulation, is present in the MASC Innoslate simulations. Figure 5.3 is an example of a swarm mission composed of its MASC architecture elements. Beginning in the upper left, an MIO mission is composed of five phases, one of which (OnStation) is depicted. The OnStation phase is further broken down into the activities of the primary participants using the heuristics (Section 4.1.1) as guidelines. The tactics level is next and shows the swarm's activities in terms of the tactics which are represented by the green rectangles. The `Evade` tactic is circled and decomposed into its play elements (orange rectangles), and finally the `Sensors EMCON` play is shown in terms of the algorithms available to support its implementation.

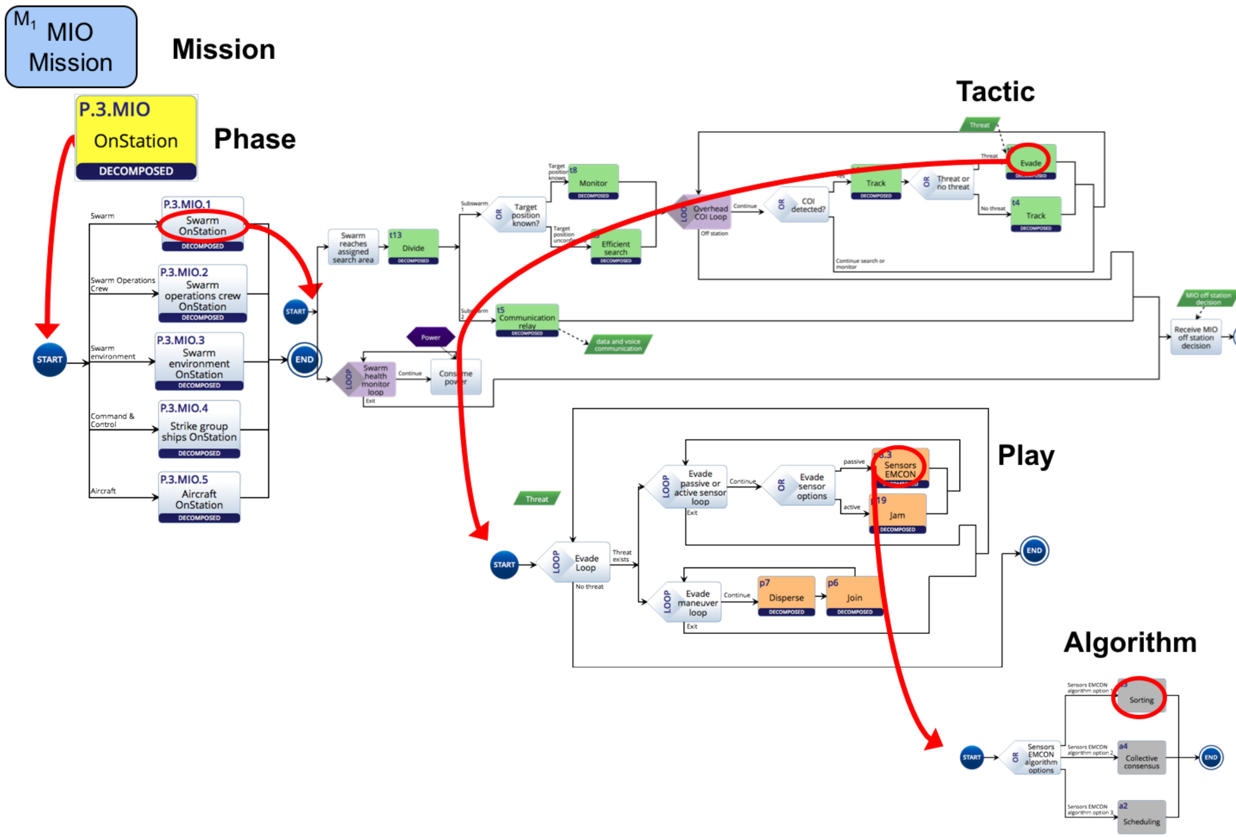



Figure 5.3. The composability of a mission is shown using the MIO mission as an example. The mission is composed of phases (OnStation is used as the example and the activities of actors are grouped separately), which are composed of tactics, plays, and algorithms. The mission runs as a discrete event or Monte Carlo simulation in Innoslate.

## 5.4 Mission Doctrine Integration

The three baseline models and HSR mission plans were reviewed to assess MASC's role in integrating swarm doctrine into the operational architecture. MASC was designed to support common mission patterns on multiple levels. For example, the three mission case studies are describable using the same flow of mission phases. Within the OnStation phase, a common pattern is dividing the swarm to distribute its capabilities between intelligence collection and relaying communication. The tactics are designed such that they can support a variety of missions by varying the play parameters and setting up the tactics available to support mission-specific ROE and TTPs. From the human subjects research, we looked for any common mission patterns used by the participants. The mission plans were also compared with the MIO Innoslate model (Figure 4.12) and the MIO state machine diagram (Figure 4.13) for similarities between triggering events.

There were several common mission patterns used by the Group 1 participants. One example was dividing the swarm into multiple sub-swarms and assigning a sub-swarm to perform `Communication relay` at a higher altitude while allocating the other sub-swarms to distributed search patterns or monitoring duties. Another example included a close-in monitoring sub-swarm accompanied by another sub-swarm operating at a safe stand-off distance after the COI adopted a threatening posture. Three SMEs divided the swarm to perform both `Evasive search` and `Efficient search` simultaneously. When compared to the Innoslate (Figure 4.12) and FSM diagrams (Figure 4.13), many of the same tactics and patterns were used by the participants. The SMEs selected several tactics not used by the author for this scenario including `Evasive Search, Harass`, and `Deter`. They also sub-divided the swarm to a greater extent than the simple baseline models.

The common mission patterns used by the Group 2 participants were similar to those enacted by Group 1. Separating the sub-swarm by altitude to have one act as a communication relay and another to collect intelligence was used by several SMEs. One participant split his swarm after the COI threat such that close-in sub-swarm surveilled using only passive sensors while another sub-swarm enacted active sensors at a longer range. This pattern should be considered for developing a new tactic.

## 5.5 Summary

In this chapter, MASC's intuitiveness, modularity, composability, and doctrine integration were evaluated as supporting attributes for architecture reusability. While the HSR exercise was unsophisticated and did not result in findings of differences between Group 1 and Group 2, the participant SMEs provided relevant, valuable feedback on the intuitiveness of MASC in a notional operational scenario. Seven of the participants were H-60 variant pilots and had performed some type of MIO mission operationally. The SMEs recognized the expendable nature of the swarm and used `Divide` appropriately to distribute the swarm's capabilities. They identified mission-suitable tactics beyond those identified in the research models and envisioned a couple of potential future tactics. Overall, system management at both the tactics and play levels seemed to provide the user with a modular, composable, intuitive format. Although the general flow of the mission plans was similar between the two groups, several of the plays such as the sensor settings or Transit to waypoint were omitted. Combining common patterns of plays into tactics reduces the selections required by the operator and chances for omissions. Larger test populations along with a higher fidelity test interface (such as a gaming environment) could provide more conclusive evidence to support the tactics paradigm.

# CHAPTER 6:
# Conclusion

## 6.1 Conclusions

To design swarm systems, we must be able to define the missions in which they will operate and describe system elements in a formal architecture. MASC provides this definition and improves our ability to develop complex adaptive systems by incorporating mission engineering and MBSE foundations in the CAS architecture design. This dissertation identified three main research objectives for formalizing swarm architectures: incorporating mission doctrine into swarm system design, enhancing architecture reusability, and improving user accessibility by augmenting bottom-up, behavior-based design with a top-down, mission-based framework.

Mission doctrine integration was demonstrated in the common mission patterns that were applied across missions in the simulations and by the different participants in the HSR. The three mission case studies were construed using the same mission phases and the tactics were designed to support a diverse mission set by varying the play parameters according to mission-specific ROE and TTPs. A common pattern within the OnStation phase was dividing the swarm to distribute its capabilities between intelligence collection and relaying communication using the `Divide` and `Amass` tactics. In fact there were several common tactics patterns used that could support another level of MASC elements *above* the tactics level. For example, a pre-defined combination of tactics (established during mission planning) dividing the swarm into a close-in monitoring sub-swarm, a sub-swarm monitoring from a safe standoff distance from a threat, and a third sub-swarm performing communication relay could be called "MIO Surveillance 1." However, this type of doctrinal development needs an interactive simulation environment in which experimentation with tactics combinations can be conducted.

MASC's modularity and composability were evaluated as supporting attributes for architecture reusability. Within the scope of application, MASC demonstrated modularity at multiple levels in the phases, tactics, and plays that were used across different missions and within each mission. MASC's modular design facilitated its composability, demonstrated in the semantically logical LML action diagram simulations. The MASC framework was modified multiple times during the mission case study model development. As previously mentioned, phases were not part of the original framework, but added later to facilitate tactic and play reuse across missions. The `Deter` tactic was added later in development to support mission scenarios in which rules of engagement forbid UAV weapon carriage. The ACM tactic incorporates minimal capability in its current state, but contains a placeholder for future A-A plays. The framework was not evaluated directly to determine if it is platform agnostic. This research was developed under the assumption of a swarm composed of small (DOD category 1-2), fixed-wing UAVs to support the endurance requirements of the MIO and HADR scenarios. However, the same type of framework could be applied to smaller fixed-wing and rotary aircraft. Adjustments for platform performance and capability differences can be accounted for in the play parameters.

The HSR exercise provided relevant, valuable feedback on the user accessibility of MASC in a notional operational scenario. All participants assembled a swarm mission plan with few errors in less than one hour and without training. Results from the workload survey and qualitative comments indicate the intuitiveness of the framework. Several comments collected during the HSR should be considered for future versions of MASC. Two SME participants recommended improvements to the OnStation phase that included breaking it down into smaller modules while another suggested a "Suicide attack" in which the unarmed swarm is capable of conducting an inertial attack against an enemy. Results of the human subjects research indicate that the playbook is relatively intuitive under the conditions in which it was assessed.

MASC extends current systems engineering practices by defining a framework and methodology for swarm system architecture design using mission engineering and MBSE foundations. This research was applied to a homogeneous swarm system but could be extended to CAS composed of autonomous agents operating under decentralized control which results in some advantageous collective behavior. This approach supports the perspectives of both the systems engineer and the operational user. Promoting reusable CAS architectures shortens development cycles, enables rapid technology upgrades, facilitates leveraging a common framework to other domains (sea surface and undersea), and enables more efficient use of high-demand software programming expertise.

## 6.2 Recommendations and Future Work

There are several areas of future work in this research domain. These include employing MASC into a prototype virtual environment, incorporating failure modes into the application of the methodology, developing swarm system measures of performance, and continuing research in swarm algorithms.

### 6.2.1 Prototype System

Employing MASC into an interactive gaming environment is crucial to progressing swarm doctrine and expanding mission sets. Establishing a taxonomy, conceptual framework, and methodology is an important initial step, but a virtual environment is needed for wargaming, tactics experimentation, and human-swarm teaming research to advance swarm doctrine development. The simulation environment could be used to develop GUIs suitable for swarm missions and continue to gather stakeholder response throughout system development. Developing a swarm system GUI that leverages the strengths of humans and machines as a team is an important research area for swarm systems.

### 6.2.2 Failure Modes

Early models for the three scenarios included off-ramps for premature mission completion at each phase level to account for operational circumstances or system failures. Operational triggers include changes in weather or mission tasking while system failures could occur in support equipment, GCS, or individual UAVs. The assumed UAV was designed to execute pre-programmed failsafe procedures in the event of lost communication link, GPS failure, autopilot malfunction, or breached geographic boundary [168], [169]. These alternate use cases are important for operational swarm system design but were scoped out of the current research to focus on the composition of the MASC playbook. Future work in this area is needed to understand how to effectively operate swarms in degraded modes.

### 6.2.3 Swarm System Measures of Performance

Swarm system measures of performance (MOPs)—measures that characterize functional attributes of the system—are needed to support acquisition. These measures could be developed based on Joint Capability Areas (JCA) attributes such as timeliness, latency, survivability, connectivity, stealth, endurance, strike, expeditionary, and interoperability to assess that algorithms are meeting play objectives, plays are meeting tactics objective, etc. The set of MOPs for each mission could then combine into measures of effectiveness (MOEs) for the mission. Figure 6.1 illustrates how these factors could relate to potential mission areas and in turn, drive algorithm and technology investment strategies.

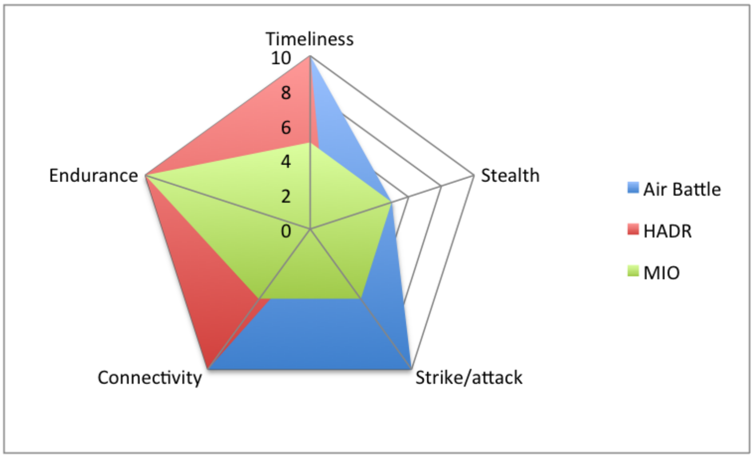



Figure 6.1. An example showing five potential performance factors for measuring swarm mission capabilities and their mappings to three mission types.

### 6.2.4 Continued Research in Swarm Algorithms

Finally, this research focused on the operational architecture—the missions, phases, tactics, and plays—rather than the solution architecture. The algorithms discussed in this dissertation exist in ARSENL behaviors or other robotics applications. The swarm algorithms reside at the boundary at which the operational architecture and the solution architecture meet. There is much work to be done in assigning the right algorithm to each play.

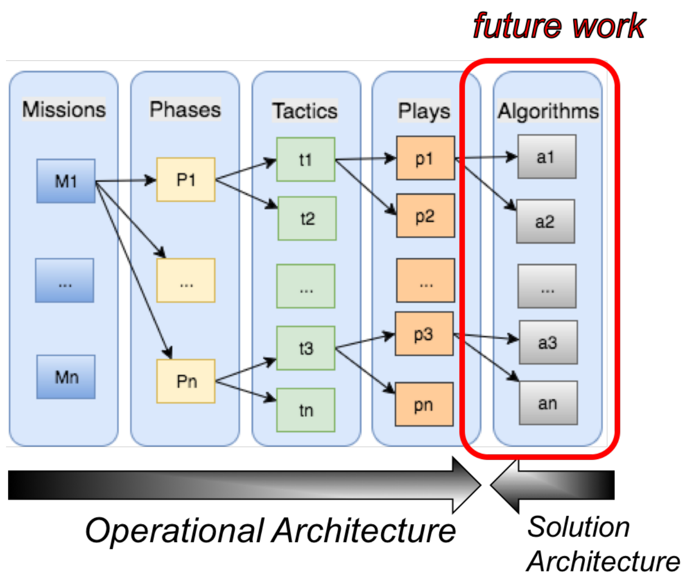



Figure 6.2. This dissertation focused on the operational part of MASC while future work will continue on algorithm development and play integration.

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX A: Background Research

This appendix covers additional general background material related to swarm systems and their relevance to SoS and CAS. Established system architecture frameworks and common methodologies for modeling CAS behavior including traditional MBSE methods, systems dynamics, and lightweight formal methods are briefly described.

## A.1 The UAV Swarm: SoS or CAS?

A SoS is an "arrangement of systems that results when independent and useful systems are integrated into a larger system that delivers unique capabilities" [170]. Maier describes SoS as having managerial and operational independence and exhibiting evolutionary development and emergent behavior [59]. Emergent behavior is behavior "which cannot be predicted through analysis at any level simpler than that of the system as a whole. Emergent behavior, by definition, is what is [sic] left after everything else has been explained" [171]. The DOD categorizes four types of SoS, listed in order of least to most centrally controlled: virtual, collaborative, acknowledged, and directed. UAV swarms are best represented by the directed category in which the system is built for a specific purpose, centrally managed during long-term operations and its normal operational mode (from the perspective of an individual agent) is subordinate to the overall purpose of the system, in this case the swarm [170]. Yet the UAV swarm is more tightly interrelated than the typical directed SoS described by DOD publications such as [170] or [172]. Furthermore, a simple, individual agent from a swarm cannot be considered "useful" in the same capacity as an individual platform from a SoS such as the Naval Integrated Fire Control-Counter Air (NIFC-CA).

Complex systems are those that exemplify self-organization, interdependence, and emergence [173], [174]. International Council on Systems Engineering (INCOSE) considers complexity as a "measure of how well the knowledge of a system's component parts explains the system's behavior and by the number of mutually interacting and interwoven parts, entities or agents" [175]. A type of complex system, CAS, are natural systems such as the brain, the human immune system, weather, and insect colonies, as well as artificial ones including economies, artificial neural networks, and social networks. CAS are composed of individual agents that interact with each other according to rules [175]. Formal definitions for CAS vary according to academic field. Dooley concisely describes them as “a group of semi-autonomous agents who interact in interdependent ways to produce system-wide patterns, such that those patterns then influence behavior of the agents” [176]. Several key attributes of CAS distinguish them from other types of systems or SoS: decentralized control, inter-connectivity, sensitivity to initial conditions, and emergent behavior.

Complex adaptive systems are composed of a large number of components, characterized by self-organization and a lack of central control, and demonstrate behavior that cannot be predicted by studying the individual sub-components in isolation. They are characterized by scalability, adaptability, flexibility, resilience, evolvability, and robustness [41]. Mitchell describes three common properties of complex systems: complex collective behavior, signaling and information processing, and adaptation [116]. Under the principle of self-organization, local interactions between agents over time combine together to create structure on a large scale [41]. From a complex system perspective, equilibrium is one of several different possible states, rather than the natural state of the system [177]. Swarm robotics systems operate at both the collective (or macroscopic) level and at the individual (or microscopic) level, with their collective behavior resulting from the interactions between agents and with the environment [75]. By these descriptions and characterizations, a swarm of UAVs can be considered a CAS.

## A.2 System Architecture Frameworks

System architecture frameworks, or enterprise architecture frameworks, guide system design by providing a modeling foundation via tools, techniques and methods. As system complexity increases, the architecture methods required become more abstracted and heuristic based, and integrated modeling becomes necessary [42]. Frameworks provide syntax, semantics, and context that facilitate modeling standardization, promote understanding among stakeholders and ultimately assist decision makers [178]. Common system architecture frameworks are DODAF [179]–[181], Zachman’s Framework [179], [182], [183], and The Open Group Architectural Framework (TOGAF) [184].

The purpose of the system architecture framework is to simplify the complex system by focusing on the essential items of interest. An architecture model is "an abstraction used to test and assess proposed concepts prior to their implementation" [89]. Clements says "to be architectural is to be the most abstract depiction of the system that enables reasoning about critical requirements and constrains all subsequent refinement" [30]. The UAV swarm architecture is developed to analyze, investigate, and refine interactions, behaviors, modularity, and cohesion [89].

Maier and Rectin [42] highlight six system architecting roles that models fill:

- communication with client, users, and builders
- maintenance of system integrity through coordination of design activities
- design assistance by providing templates, and organizing and recording decision
- exploration and manipulation of solution parameters and characteristics; guiding and recording of aggregation and decomposition of system function, components, and objects
- performance prediction; identification of critical system elements
- provision of acceptance criteria for certification for use

The models used in this research for developing the proposed framework are designed around specific proposed UAV swarm missions and support the six roles.

## A.3 Methods for Modeling CAS Behavior

As DOD systems become increasing complex, the use of modeling and simulation throughout the system life cycle, from conception to disposal, has become more prevalent. Due to the diverse origins of complex systems, a variety of methods have been used to model them. Model-based systems engineering has been widely adopted by the private sector and DOD as a systems engineering best practice. Behavior models are commonly used in system architectures to capture system functions and interactions between components. A system's behavior includes responses to requests for services, input-to-output transformations, and what it must do to meet requirements [185]. In systems engineering, behavior models are typically implemented from a top-down functional approach. Bottom-up approaches, in which complex systems are "built up" from individual components are common in software engineering, for example, object-oriented programming language such as Java, C++, and Python. System dynamics models—using differential equations to model large systems such as ecosystems or economies—characterize complex systems but many exclude the non-linear aspects and miss capturing the emergent behavior [177]. Other researchers have successfully used cellular automata to model swarm robotics systems, particularly controllers, and capture collective behaviors [58], [177], [186].

### A.3.1 Traditional MBSE Models

Top-down design models, in which the high-level functional elements are initially specified before decomposition of lower-level functions, are common in the systems engineering field. Activity, sequence, state machine, and use case diagrams are traditional systems engineering behavior modeling diagrams. While each of those diagrams has strengths, each one is challenged to completely capture CAS behavior. Activity diagrams, the primary SysML diagrams for system behavior modeling, describe the functional flow of actions in the system [187]. They are structurally similar to Functional Flow Block Diagrams and Enhanced Functional Flow Block Diagrams [188]. While they are useful for describing activity flows in a mission, the syntax and semantics inherent to activity diagrams can make them challenging to use in simulations due to the detail that is required to accurately capture system behavior [189]. Furthermore, continuous actions and concurrent behaviors for multiple agents can be challenging to display graphically on a single model. Establishing strict model syntax and semantics, simplifying entity types, and using multiple levels of abstraction can make activity diagrams more efficient for modeling CAS. State machine diagrams describe system behavior as distinct states in which the system can operate, with triggers enabling transitions between the states [187]. State machine diagrams offer simplicity and enable a large portion of system behavior to be displayed in one diagram; however, it may be difficult for the modeler to catalog all possible combinations of events generated by a state machine diagram in more complex systems. This conundrum is known as the state explosion problem [86]. Use case diagrams, in which actors interact with the system to perform a task, are useful for describing system usage from a high-level perspective, but they are less effective for connecting with other parts of a model or building a system architecture [189].

SysML supports MBSE methodologies including the design, analysis, and verification of complex systems. SysML is a commonly used graphical modeling language for systems engineering applications, including behavior models. SysML is derived from Universal Modeling Language (UML), the software engineering industry standard, and the two languages share diagrams and principles [187]. The capability to relate system requirements to the system model is a major strength of SysML that facilitates model verification and requirements traceability. Furthermore, automated software tools such as Innoslate and Vitech's CORE enable some automatic diagram generation based on already existing diagrams, saving the systems architect from manually building diagrams containing the same data. The different SysML diagrams allow the architect to parse out model views and requirements, based on stakeholder concerns. Of the nine diagrams supported by SysML, shown in Figure A.1, there are four that focus on behavior modeling: activity, sequence, state machine, and use case diagrams. Estefan [190] and Beery [6] survey current MBSE methods used by major organizations and companies; many of which rely on SysML products.

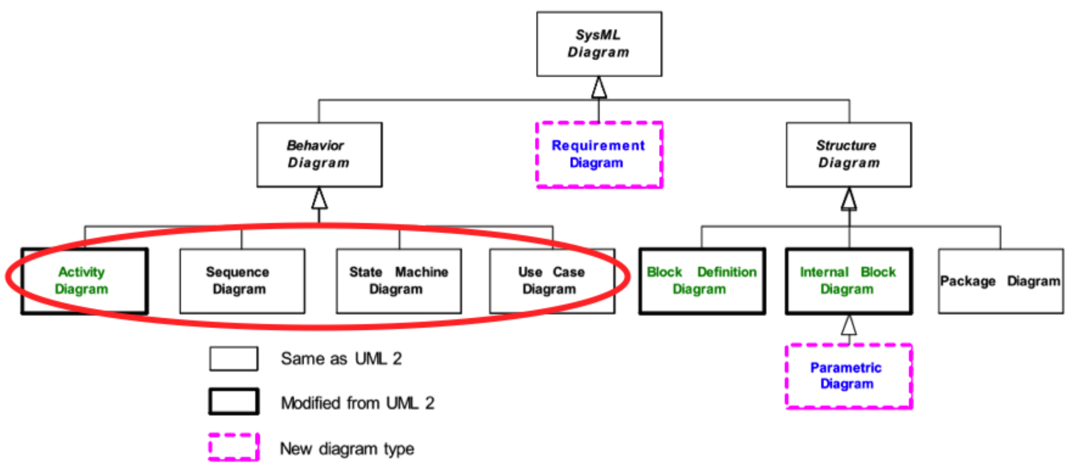



Figure A.1. SysML Diagram Taxonomy. Source: [187].

### A.3.2 System Dynamics Methodology

The system dynamics methodology represents system behavior using flows, feedback loops, and levels (also called stocks), which are supported by mathematical equations. A key aspect of this methodology is the notion that most systems are composed of many nested feedback loops that produce future actions or problems [191]. System dynamics models are used to evaluate the behavior of complex, non-linear systems over time using flows, feedback loops, stocks, and piles [192]. Forrester called system dynamics “the study of information-feedback characteristics of industrial activity to show how organizational structure, amplification (in policies), and time delays (in decisions and actions) interact to influence the success of the enterprise” [193]. Figure A.2 shows the importance of iteration and feedback loops in Forrester’s systems dynamics model, which was designed for modeling non-linear physical and social systems.

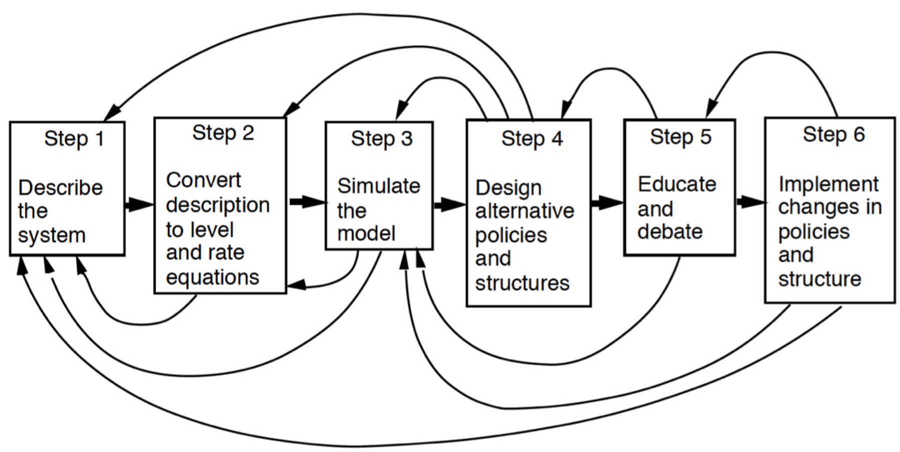



Figure A.2. System Dynamics Process Model. Source: [194].

While the system dynamics methodology is designed to manage non-linear, complex system behavior, it does have drawbacks relevant to CAS. This methodology requires “causally-closed” systems - those in which the causes of the behavior of interest reside within the system [194]. The interrelation of causal factors within a system dynamics model makes it difficult to partition the agents and incrementally improve the model. In effect, these limitations preclude the development of composable, modular CAS models.

### A.3.3 The Formal Methods Approach

Formal methods have been used by the software engineering community to reduce the ambiguity of natural language specifications and improve the quality of system requirements in complex, software-intensive systems [195]. In computer science and software engineering, formal methods are mathematically grounded techniques including logic, semantics, and formal languages [85]. Formal methods have been used by the software engineering community to produce "formally defined functional and nonfunctional properties" by defining system behavior as sets of permitted and non-permitted operational sequences, that can be constrained by timing or other parameters [42].

The degree of rigor on the formality spectrum varies between direct, logical interpretations using proofs and theorems, to less rigorous methods that employ discrete mathematics notations to develop specifications [195]. Rushby classifies formal methods within four levels [195]: Level 0: no use of formal methods; Level 1: replacing some natural language in requirements with discrete mathematics concepts and notations; Level 2: using formalized specification language along with mechanized support tools; and Level 3: using completely formal specification language with widespread theorem proving and proof checking support tools. Formal methods can also be categorized in terms of breadth of application, from widespread across all stages of the life cycle to certain components or phases of systems development [195]. Others use the term *lightweight* to characterize an approach used to analyze part of the specification or requirements document without re-baselining the entire specification, and the term *heavyweight* to describe a deeper, more complete application of the methodology [83], [87]. A distinct advantage of formal methods is that they can be used to validate certain system characteristics that can never be tested completely. Theorems can be formulated based on formal model, proving that specific event chains cannot occur [42].

Well-established formal methods tools such as the Vienna Development Method [196], Larch [197], and the Z specification notation [198] specify sequential system behavior in terms of relations, sets, and functions. Likewise, Communicating Sequential Processes [199], I/O Automata [200], and Temporal Logic [201] specify behavior of concurrent systems [85]. The FORMAN (FORMal ANnotation) approach uses the concepts of event grammar and event hierarchies to build system behavior models to formalize universal assertions for defining debugging rules [202]. As of 2013, MP provides a framework for business processes and software system architecture design based on behavior models [203]–[205].

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX B: Design Reference Missions

This appendix describes the DRMs used as case studies for MASC in Chapters 3 and 4. The first DRM, SvS, is closely linked to the field experimentation conducted quarterly by the ARSENL team from NPS. ARSENL's research is currently focused on developing A-A tactics for employment against an adversary swarm. The SvS mission architecture operates at a higher level automation than the second and third scenarios, which require more human interaction. For this reason, the swarm vs. swarm mission architecture could be considered as a foundation for a future ship defense scenario. The second and third DRMs—HADR and MIO—are notional UAV swarm operational missions. These missions require less time-critical decision-making, and consequently operate at a lower level of automation and with more human interaction. Furthermore, these two operational missions assume a more mature system with simpler user interface than the ARSENL field experimentation, resulting in the merge of the Mission Commander role and Swarm Commander role into one Swarm Commander.

## B.1 SvS Design Reference Mission

This DRM outlines an operational context for a notional autonomous UAV swarm vs. swarm mission. Specific swarm unmanned system scenarios may be developed from this DRM, for designing operational architectures solutions. The purpose of this mission is to seek innovations in offensive and defensive tactics, autonomy algorithms, and CONOPS, and support ongoing research in system design. The analysis questions motivating the development of this DRM are the following:

- How could a UAV swarm be employed to engage in saturation attacks on an opponent, and
- How could swarms be used to protect a high-value home base?

This DRM contains a specific example of a basic swarm vs. swarm scenario to demonstrate aerial combat swarming capabilities by deploying two teams of homogeneous UAVs to each protect their own home base, and attack their opponent's home base and defenders.

From this basic scenario, offensive and defensive tactics can be designed, while different communication and coordination architectures can be explored.

### B.1.1 Mission Background

The swarm vs. swarm mission derives its inspiration from the DARPA Tactical Technology Office SASC, which was designed to provide a test bed for developing defensive and offensive tactics for an autonomous UAV swarm, in support of DOD, Navy, and Office of Naval Research strategic goals. The SASC format used a mix of fixed-wing and multi-rotor aircraft, however this DRM will only consider a homogeneous swarm of fixed-wing UAVs which is similar to the regular ARSENL format. The competition setup provides a common infrastructure, in terms of standards and open software, as a means for rapid technological innovation for large numbers of autonomous, cooperative UAVs. The operational relevance of this scenario applies to the SEAD mission in which manned military aircraft are used to suppress land-based air defense systems such as anti-aircraft artillery (AAA), surface-to-air missiles (SAMs), command, control, and communication (C3) capabilities and early warning radars. To perform these missions, aircraft must be equipped with anti-radiation missiles, precision guided air-to-ground weapons and electronic warfare systems.

### B.1.2 Operational Concept

A swarm vs. swarm operational concept is featured in the OV-1 diagram in B.1. The physical environment is an overland area of responsibility (AOR). The Blue Swarm is represented by the blue airfoils, and the Red Swarm is represented by the red airfoils. Each swarm is depicted defending its respective home base. The virtual arbiter is represented by the referee icon and it will score the mission in accordance with the rules delineated in B.1.9.

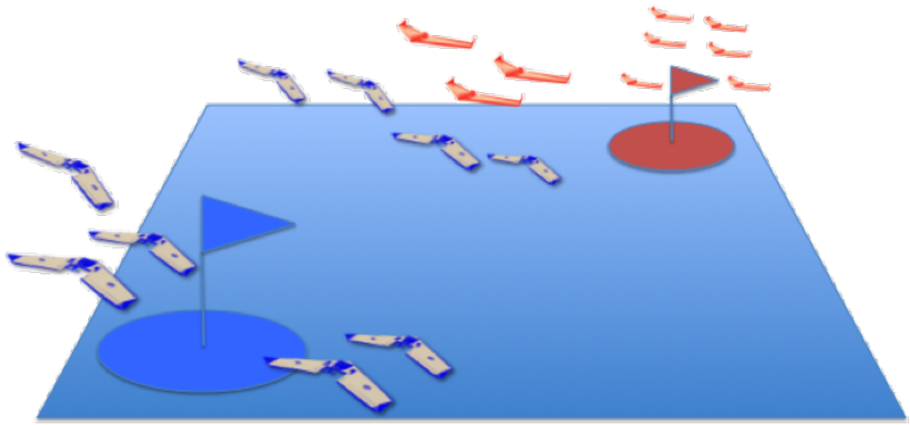


Figure B.1. OV-1 Operational Concept for an Autonomous SvS. Source: [206].

### B.1.3 Projected Operating Environment

The projected operating environment is the environment in which the autonomous UAV swarms are expected to operate. This section provides details that describe the environmental conditions, types of locations, and threats to which the system will be subject and establishes a context within which interactions and interfaces of the system may be modeled to produce measurable outcomes to enable future physical architecture design trade offs [207]. Detailed environmental restrictions can be found in the Zephyr II UAS CAT 3 Interim Flight Clearance for NPS Multi-Air Vehicle Operations with Restricted or Warning Areas [208].

#### Environmental Conditions

The swarm systems are expected to operate in:

- Day visual meteorological conditions only
- Temperature: 40-115°F
- No flight through visible moisture or moderate or greater turbulence
- Maximum winds allowed:
- Takeoff and landing headwind: 21 kts
- Takeoff and landing tailwind: 4 kts
- Takeoff and landing crosswind: 13 kts

- Winds aloft: 22 kts

The DRM will use a 500m x 500m x 500m above ground level (AGL) "Battle Cube," elevated 78 m above ground level at Camp Roberts near Paso Robles, CA, depicted in Figure B.2, as the operational area. The ARSENL team has used this location for the past four years for various UAV flight tests.

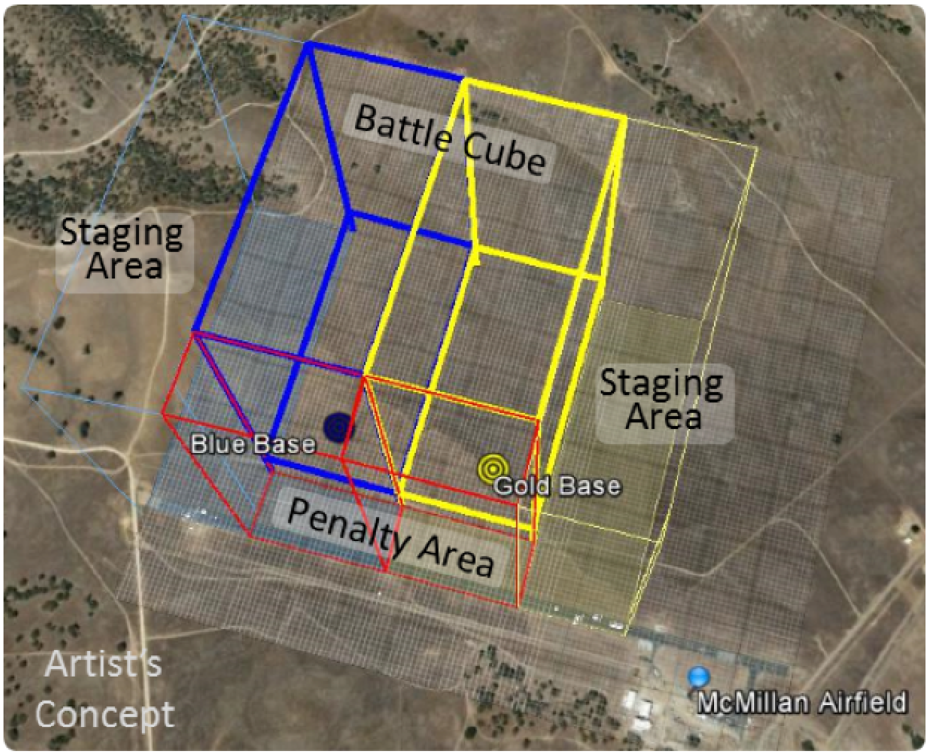



Figure B.2. Camp Roberts Airspace, near Paso Robles, CA. Source: [33].

The AOR, sequence of waypoints for each swarm to reach its initial operating box (east and west), geofence (annotated in green), and landing points (indicated in yellow) are shown in Figure B.3.

Figure B.3. SvS AOR. Source: [206].

### B.1.4 Assumed Threat Environment

Threats to the success of the SvS mission are primarily environmental in nature. The primary weather related threat is excessive wind speeds. A mobile weather station will be used to monitor wind conditions. As part of the mission, each swarm will be threatened by its adversary swarm.

### B.1.5 Mission Success Requirements

For the mission to be successful, one swarm must "win" via scoring the most points by: 1) executing A-G kills against the opponent's home base or 2) achieving A-A kills against the opponent's aircraft. The virtual A-A weapon has a firing rate of 1 Hz with unlimited rounds and scores a hit within the envelope shown in Table B.1 and Figure B.4. The scoring system rules are delineated in Section B.1.9.

Table B.1. Parameters for Virtual Weapon Envelope. Adapted from [92].

| Parameter | Value |
|---|---|
| Range (r) | 100 m |
| Azimuth angle ($\alpha$) | 15° |
| Altitude angle ($\beta$) | 180° |

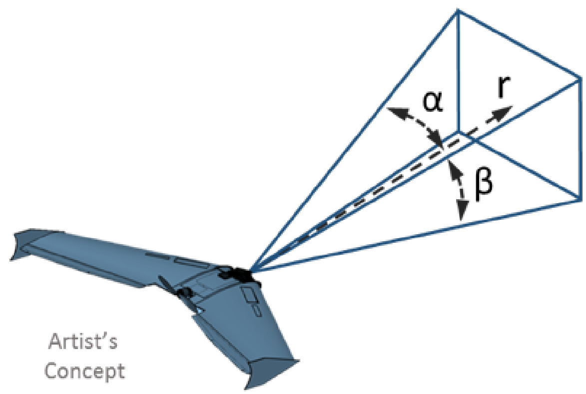



Figure B.4. Virtual Weapon Envelope. Source: [92].

### B.1.6 Mission System and Safety Requirements

All of the following high-level system and safety requirements must be met for any swarm vs. swarm mission [2], [209]:

- Swarm aircraft must act independently, exhibit autonomous behavior, and require only local information,
- Swarm aircraft must maintain constant communication with the arbiter,
- Swarm aircraft must not exceed 26 m/s, or 5 kg,
- Only swarm aircraft launched within the time limit will be allowed to participate,
- The swarm system must comply with prescribed batteries and fuels,
- The swarm system will not conduct electronic warfare,
- Intentional mid-air collisions are not permitted,
- Collision avoidance will be mitigated by the altitude de-confliction plan:
  - The aircraft within each swarm will be vertically separated by 30m, and the offset between each swarm will be 15m.

- For example: If Blue Swarm aircraft are stacked at 100, 130, 160m AGL, then Red Swarm aircraft will be stacked at 115, 145, 175m AGL.

- The swarm aircraft will not carry weapon stores, intentionally launch any objects from the aircraft in flight, or utilize any type of tether,
- The swarm system must comply with ARSENL failsafe methods as described in [161], [168], [169].

### B.1.7 Mission Definition

The main reference mission provides a level of detail necessary for collecting measures to assess mission success requirements. As technology and tactics develop, additional mission variations will be added to allow for increased aircraft, incorporation of sensors and weapons, different tactics and plays, other unmanned or manned assets, and different environmental conditions. The capability needs statement for the main SvS mission is:

> The DOD is vulnerable to saturation attacks where an adversary can exploit large numbers of inexpensive unmanned systems to overwhelm defense capabilities. To develop counter swarm tactics to combat this threat, an autonomous swarm vs. swarm mission needs to be developed to design and explore offensive and defensive tactics for UAV swarm systems.

The following operational situation (OPSIT) pertains to a Blue 25 versus Red 25 homogeneous UAV swarm scenario. This differs from the DARPA challenge which allows for a 25 versus 25 heterogeneous swarm composed of a variable mixture of fixed wing and quad rotor UAVs. The assumptions being made about the mission's operational environment are stated, followed by the mission narrative.

**Capture the Flag OPSIT**

Two swarm teams of 25 aircraft each, called Blue Swarm and Red Swarm will be situated on deck in their respective home bases. The autonomous swarm vs. swarm mission commences with a preparation phase. The mission flow timeline consists of the following general phases [2]:

- preparation including pre-flight and briefing (2 hrs)

- launching the aircraft (8 min)
- Aerial Combat Swarm battle (30 min)
- landing the aircraft (7 min)
- post-flight, and de-brief (60 min).

The following roles (modeled as assets) are used in the mission and reflect the operational team model for ARSENL and the DARPA SASC:

1. Arbiter—acts as a virtual sensor network and referee. It receives GPS positions of all aircraft and re-broadcasts the opposing team's aircraft as bandits. The arbiter enforces mission rules, and scores the mission.
2. Swarms (Blue Swarm and Red Swarm)—each composed of 25 homogeneous UAVs.
3. Mission Commander for each swarm (Blue MC and Red MC)—responsible for conducting mission brief and de-brief, exercises command decision authority for swarm.
4. Swarm Monitor for each swarm (Blue SM and Red SM)—responsible for health monitoring, battery condition, and altitude blocks of individual UAVs.
5. Swarm Commander for each swarm (Blue SC and Red SC)—responsible for execution of swarm tactics. Assumes control of swarm from SM when swarm reaches "swarm ready" state.
6. Ground Crew is composed of Launcher Operator and Launch Technician:
   - Launcher Operator for each swarm (Blue LO and Red LO)—responsible for aircraft launch, and recovery of aircraft.
   - Launch Technician (Blue LT and Red LT) - responsible for pre-flight and post-flight of aircraft.
7. Safety Pilot (Blue SP and Red SP) for each swarm—manually takes control of errant aircraft using a 900 Mhz serial radio controller.
8. Air Boss (AB)—Monitors airspace, coordinates with tower, provides clearance to swarm to operate.

### B.1.8 Mission Execution

The following mission narrative describes the reference mission "SvS Autonomous Aerial Combat." General rules describe any events that are recurring, or that can occur at any point during the mission. Mission start time (T+00) occurs at the beginning of the launch phase.

**SvS Autonomous Aerial Combat Mission Narrative**

- Blue Swarm Crew and Red Swarm Crew conduct system preflight as commanded by respective MC
- Blue MC and Red MC conduct mission brief with respective SC and Ground Crew
- Blue MC and Red MC receive Green Range call from AB
- Blue MC and Red MC authorize Ground Crew to begin launching aircraft
- Blue and Red Ground Crews launch aircraft
- Blue Swarm and Red Swarm execute Ingress tactic
- Blue Swarm and Red Swarm reach OnStation waypoint
- Blue MC and Red MC receive battle start command from Arbiter
- Blue SC and Red SC execute search tactics
- Blue Swarm executes A-A attacks on Red Swarm, Red Swarm executes A-A attacks Blue Swarm
- Arbiter scores A-A attacks.
- Blue Swarm executes A-G attacks on Red Swarm, Red Swarm executes A-G attacks on Blue Swarm (by landing at the opposing team's home base)
- Arbiter scores A-G attacks
- Blue SM and Red SM execute Egress tactic for remaining Blue UAVs and Red UAVs
- Blue and Red Ground Crews recover aircraft
- Blue MC and Red MC conduct mission de-brief with respective crews

**General Rules**

1. Throughout the mission, both Blue SM and Red SM constantly monitor battery conditions. As the aircraft approach low battery conditions (20% or 10.6V), the SM will command the swarm or subswarms to egress.
2. A virtual penalty box will be used by the Arbiter to identify aircraft that have violated a game rule. These aircraft will be prevented from scoring point following their infractions. Penalties for the mission include:
   (a) Airspace violations—breaching the boundaries of the operating area.
   (b) Airspeed violations—exceeding the maximum allowed airspeed.

    (c) Intentional mid-air collisions.

3. Failsafe modes [161]:
   (a) Low battery (20% battery, 10.6V)—autopilot executes autoland at failsafe landing waypoint
   (b) Ground Control Station link lost- autopilot executes return to rally point (RTL) after 2 min of no link
   (c) Loss of GPS signal—autopilot enters LOITER mode for 5 sec, cuts throttle if GPS has not restored, and executes a contained crash.
   (d) Geofence breach—autopilot executes RTL. After 20 sec, if not returned to the other side of the fence, throttle is cut.
   (e) Loss of link to payload (payload heartbeat lost)—autopilot executes RTL, autoland after 2 min if this has also resulted in loss of GCS link
   (f) Motor failure—autopilot executes RTL, manually or automatically land if possible, otherwise a contained crash is executed
   (g) Autopilot failure—autopilot executes contained crash if manual mode not accessible
   (h) Throttle stuck high—results in fence breach failsafe
   (i) Loss of electrical power—autopilot executes contained crash
   (j) Control surfaces failure—autopilot executes contained crash.

### B.1.9 Measures

The competition is scored based on criteria for A-A attacks, A-G attacks (landing at the opponent's base) and an endurance factor for keeping one's aircraft airborne over time. The coefficients weight the A-A higher than the others to encourage A-A tactics development to support the counter-swarm mission. The team accruing the higher score wins. To provide quantitative results and recommendations, the following scoring system is used to assess the technical and tactical performance of the swarm [92]:

$$Score = \alpha A_{\mathrm{a}} + A_{\mathrm{g}} + \chi E \tag{B.1}$$

where:

- $A_{\mathrm{a}}$ is the number of A-A tags

- $A_g$ is the number of A-G tags
- $\alpha$ = 3.2 (A-A factor)
- $\chi$ = 3.8 (logistics factor)
- $E$ is the measure for swarm endurance:

$$E = \sum_{n=1}^{max} \frac{\tau_n}{T} \tag{B.2}$$

  - $\tau_n$ is the airframe flight time
  - $T$ is the total combat time
  - *max* is 25 airframes

## B.2 Swarm HADR Mission

This section outlines a DRM for a HADR scenario in which assistance and relief efforts are supported by a UAV swarm. This DRM is used to explore methods for improving response efficiency and effectiveness during humanitarian disasters using a UAV swarm and to support CONOPS development and ongoing research in system design. Analysis questions motivating the development of this DRM include:

- Could a UAV swarm be employed during a HADR mission scenario in a permissive environment? If so, how?
- If employed, would the use of a UAV swarm improve the responsiveness of humanitarian aid to those in need? If so, how?
- Could a UAV swarm be used improve communication and information dissemination among the stakeholders? If so, how?

This DRM contains a scenario with a UAV swarm operating in response to a HADR effort generated by an earthquake in a foreign country. From this basic scenario, communication and coordination tactics can be explored to influence the UAV swarm architecture. Additional scenario variations will provide basis for follow-on analysis from the use cases generated.

### B.2.1 Mission Background

HADR is one of the six core capabilities (forward presence, deterrence, sea control, power projection, and maritime security are the other five) of the US naval forces that directly support the national military strategy [210]. Humanitarian assistance encompasses the "aid and action designed to save lives, alleviate suffering, and maintain and protect human dignity during and in the aftermath of emergencies" [211]. HADR missions involve acts of nature such as drought, flood, fire, hurricane, earthquake, and volcanic eruption; or human-generated disasters such as civil war, riot, and epidemic [212]. In recent years, the Navy has assisted in the relief efforts of the Indian Ocean earthquake and tsunami near Indonesia (2004), Hurricane Katrina (2005), Cyclone Sidr in Bangladesh (2007), and the earthquake in Haiti (2010). U.S. Navy (USN) platforms bring a unique and valuable capability to HADR in that they provide organic medical support, operate without relying on ports and land-based airfields, and manage extensive communication systems and airlift capabilities [210]. Remote sensing data, such as the aerial imagery collected and disseminated by a UAS, were effectively used to assess the degree of landslides, the extent of blocked roadways, infrastructure damage assessment, and guiding SAR teams [213]. The collected imagery can be compared to existing satellite imagery (such as Google Earth). LIDAR technology is particularly useful for generating three-dimensional models of landslides covered by vegetation. The Navy plays a more prominent role in disasters occurring in close proximity to shorelines, typically hurricanes and tsunamis (often cause by earthquakes). By using surface ships, the Navy brings significant cargo transfer capability and can distribute essential relief supplies to those in need within weeks. With the rising world population and high percentage of people living near the coast, the Navy expects its role in HADR missions to remain substantial.

### B.2.2 Operational Concept

In a unilateral response, the United States Agency for International Development (USAID) or Department of State (DOS) normally lead HADR activities and are by supported by military forces [212], [214]. The U.S. ambassador typically manages all aspects of the U.S. relationship with the host nation (HN) [214]. The DOS requests military support from DOD, which are then approved and delegated to the regional combatant commander and joint task force commander [214]. In the likely case of a multinational response, the United Nations acts as the prime organizing and coordinating agency.

There are many different operational models for HADR missions. Civilian agencies typically simplify HADR into three phases: preparation, immediate response and reconstruction [215] while the military uses campaign and operation level joint doctrine phasing models such as those found in JP 3-29 Foreign Humanitarian Assistance, JP 5-0 Joint Operation Planning and TACMEMO 3-07.6-06 Foreign Humanitarian / Disaster Relief Operations Planning.

As with other military operations, the planning phase is a critical component to a successful mission. HADR is unique in that the time between notification and deployment is much shorter than most military operations. Furthermore, the specific environment and circumstances are not known in advance and therefore much of the doctrinal guidance is general. During the planning phase, the HN, USAID, other US government agencies, non-governmental organizations (NGOs), and inter-government agencies (IGOs) conduct needs assessments to determine the capability of each participating agency. The relief system and mission statement are developed based on the required support needed, within the context of the operational environment. Once the relief system has been established, the joint force structure is developed to provide coordination and communication processes between agencies [212].

The execution and assessment phase begins with deployment, which includes joint reception, staging, onward movement and integration [212]. Once the forces are in place, C2 and sustainment operations bring communication, transportation, and logistics support to the HN. Intelligence and information gathering and dissemination help the appropriate relief agency prioritize requests for aid and deliver supplies. Finally, assessment is conducted throughout the operation and is a key component for a smooth transition of control back to the HN and NGOs at the appropriate time. The expected duration of a HADR mission can vary widely. The relief efforts for the Indian Ocean tsunami, Hurricane Katrina and the Haitian earthquake lasted 81, 42 and 72 days, respectively; however, the USN part of the mission was completed in 41, 38, and 41 days, respectively [216]. In those three examples, NGOs and the HN were able to take over the relief effort after the immediate response and stabilization efforts made by USN assets in the initial 5–6 weeks.

This DRM focuses on the immediate response phase. The operational concept (OV-1) is shown in Figure B.5 and the following assets may be used in the scenario:

- USN ships:
  - landing helicopter dock (LHD) amphibious assault ship – with medical support, CH-53 and MH-60 variants for transport, lift, and SAR; and landing craft air cushion (LCAC) for ship-to-shore supply delivery
  - landing helicopter assault (LHA) amphibious assault ship - with medical support, CH-53, MH-60 variants, and MV-22 for transport, lift.
- joint task force command and control node (JTFC2)—tactical air control squadron (TACRON), joint force air component commander (JFACC), or other joint task force (JTF) asset who will be providing air traffic control. Responsible for coordination between military and NGO assets.
- Helicopters—MH-60 variants and CH-53, for SAR and ship-to-shore personnel and

supply transport; and C-2 for personnel and supply transport from the LHD.

- UAV swarm consists of a collection of identical UAVs launched from the LHD, a GCS, launch, and recovery systems, capable of providing:
  - streaming IR, video for detecting, classifying and identifying targets in the IR spectrum, during wide-area, day or night search
  - EO video for detecting, classifying and identifying targets in the visible light spectrum during wide-area, day-time search in clear atmosphere
  - synthetic aperture radar for all-weather detection and classification of stationary objects, and for determining the status of infrastructure such as roads, bridges, and buildings. IR and EO sensors can be cross-cued to and initial synthetic aperture radar target detection.
  - simultaneous voice relay and data-link communication over VHF, UHF, and military and commercial satellite

Military Sealift Command (MSC) cargo and hospital ships are useful in HADR missions for carrying large quantities of cargo and functioning as floating hospitals; however, they are not included as assets for this immediate response phase scenario. MSC cargo ships are manned with small crews, and may not have embarked helicopters, limiting their SAR and other immediate response mission utility. Hospital (T-AH) ships are not kept in a ready status (medical personnel are pulled from Navy active duty hospital staff or from the Navy's Reserve), which delays their arrival.

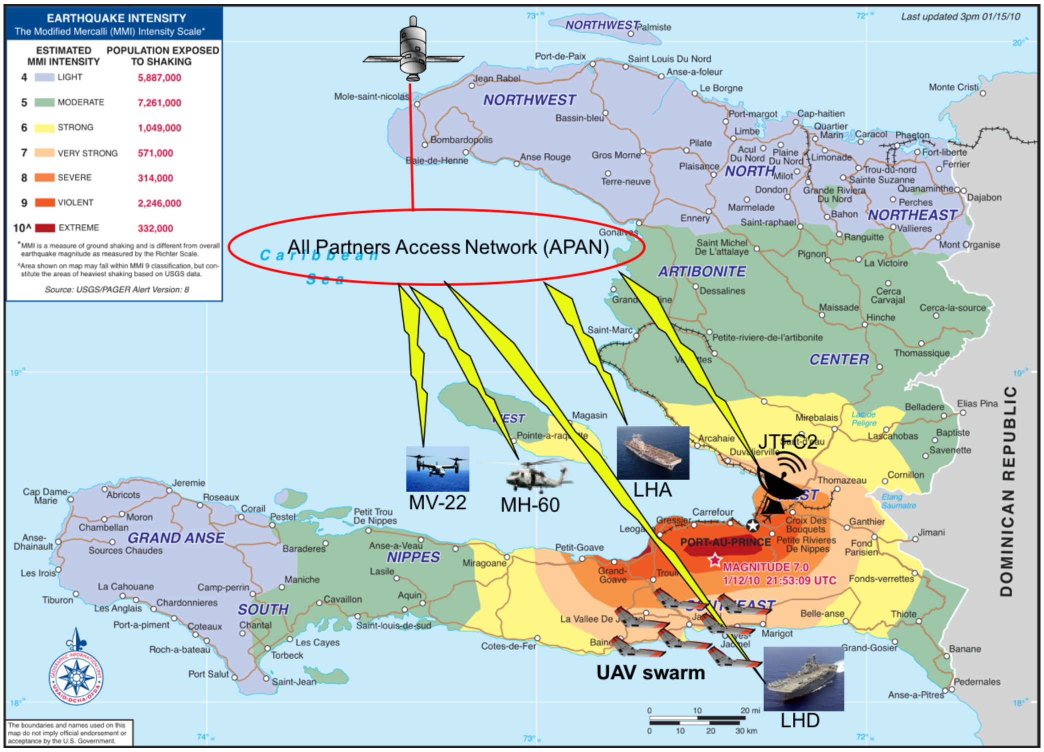



Figure B.5. OV-1 Operational Concept for HADR. Adapted from [217].

### B.2.3 Projected Operating Environment

The Projected Operating Environment is the environment in which the swarm is expected to operate. This section provides details that describe the environmental conditions, types of locations, and threats to which the system will be subject.

#### Environmental Conditions

The HADR mission domain is the nation of Haiti, specifically the capital city of Port-au-Prince and the surrounding area. The swarm will operate in the Caribbean maritime environment under the following expected conditions:

- daytime, visual meteorological conditions
- mountainous terrain - highest peak: Pic la Selle (8,793 ft)
- Caribbean tropical weather, with temperature: 70 – 90°F
- maximum operating altitude: <18K' MSL

- light precipitation
- wind gusts less than 20 kts
- multiple electromagnetic emissions within a wide radio frequency range

**Assumed Threat Environment**

Threats to the success of the HADR UAV swarm mission are expected to come from convective weather and mountainous terrain. A mobile weather station on board the LHD will be used to monitor weather conditions. The threat environment is considered permissive—threats from human actors, groups, or governments are not considered.

### B.2.4 Mission Success Requirements

For the mission to be successful, the following high-level requirements must be met for the swarm system under design:

- embark on and operate from LPD-19, LHD–5, LHA–6, or LHA-8 class ships,
- collect and disseminate imagery data to military and civilian units to improve timeliness of humanitarian need prioritization and decrease response time to deliver relief supplies, and
- provide communication relay to other military and civilian units to improve information dissemination among participating units and decrease response time to deliver relief supplies.

### B.2.5 Mission Definition

The main reference mission provides a level of detail necessary for collecting measures to assess mission success requirements. As technology and tactics develop, additional mission variations will be added to allow for increased aircraft, incorporation of sensors and weapons, different tactics, other unmanned or manned assets, and different environmental conditions. The capability needs statement for the main UAV swarm HADR mission is: the US Navy needs a cost-effective means to rapidly conduct reconnaissance, and support network-centric communication to support immediate response to HADR missions, freeing crews to conduct other necessary missions. The main mission's OPSIT describes the assumptions being made about the mission's operational environment that have implications for logistics, deployment, and time required to complete the mission. The following OPSIT pertains to a US Navy HADR task force supporting the Government of Haiti (GOH):

**OPSIT**

This OPSIT is a fictional scenario, based on the cataclysmic, magnitude 7.0 earthquake which occurred on 12 January 2010. In this scenario, Haiti sustains a magnitude 6.8 earthquake on 15 January 2018, at 1603 eastern standard time. Over the following 5 days, more than 10 aftershocks greater than magnitude 3.5 are recorded. To increase the effectiveness of this mission, the GOH and USAID have requested the assistance of the US Navy to provide relief in the form of medical support, temporary communication infrastructure, airborne reconnaissance, SAR, relief supply delivery, and berthing capacity for an expected 40-day period. The primary mission for the swarm is to provide remote sensing data (EO, IR, synthetic aperture radar) to assist infrastructure damage assessment and to guide SAR operations. The swarm launches from the LHD and establishes communication with the JTFC2 node. The swarm proceeds with the briefed tasking and potential targets, but may receive in-flight re-tasking based on the dynamics of the relief effort. USN ships and helicopters receive the swarm imagery and target positions via common data link and Link-16. A secondary mission is for the UAV swarm to act as an interim airborne communications relay node over the area of operation until more permanent communications can be established. The primary requirement will be relaying UHF and VHF voice and data communications between geographically separated ground elements that cannot establish direct line-of-sight communications. Once the swarm reaches bingo fuel or mission conclusion is commanded, it egresses to the LHD where it is recovered.

### B.2.6 Mission Execution

The following mission narrative describes a UAV swarm HADR reference mission. Mission start time (T+00) occurs at the beginning of the staging phase. The mission will be composed of the following phases: Staging, Mission Planning, Preflight, Ingress, OnStation, Egress, and Postflight. An optional failsafe mode is enacted if a system error (such as loss of telemetry link or loss of GPS signal) occurs. The swarm will be operated by two personnel: 1) the Swarm Commander, in charge of operating the GCS, controlling the UAV swarm while airborne and coordinating with external units; and 2) the Ground Crew, responsible for UAV swarm pre-flight, launch and recovery. Both personnel are involved with the staging and mission planning phases.

**UAV Swarm HADR Mission Narrative for Aerial Reconnaissance Mission**

- **Staging Phase** (completed once)
  - Staging phase begins once the UAV swarm in its travel configuration and arrives at the designated deployment site (ground or shipboard)
  - Swarm Commander, Swarm Monitor, and Ground Crew unpack and assemble the swarm system from its travel configuration
  - Swarm Commander, Swarm Monitor, and Ground Crew configure system in preparation for executing a flight mission upon receipt of orders
  - Ensure GCS has up-to-date digital charts with:
    - AOR
    - airfields and landing zones
    - natural and man-made hazards (powerlines, towers),
    - restricted airspace
    - no-fly zones depicted
    - cities and villages
    - hospitals
    - reservoirs
    - environmental waste/pollution sites
  - Swarm Commander establishes communications with JTFC2
  - Staging phase ends when the UAV swarm has been assembled, components tested, communications established, and the crew is ready to receive and plan missions.
- **Mission Planning Phase**
  - The Mission Planning Phase begins when the Swarm Commander receives a tasking order to be prepared to execute an aerial reconnaissance mission
  - Update communication capabilities of other players
  - Swarm Commander reviews air tasking order (ATO), special instructions (SPINS), and any other relevant operational tasking orders (OPTASK) from the joint task force commander. These documents contain aircraft callsigns, mission types, coordination frequencies, airspace boundaries, ingress/egress

corridors, failsafe rally waypoints, etc.
  - Update GCS digital charts to reflect any changes from previous configuration
  - Plan ingress and egress routes to and from the search area, OnStation waypoint, recovery waypoint, and failsafe rally waypoint
  - Plan search pattern to cover assigned tasking order
  - Review data collection plan
  - Check weather
  - Conduct mission brief
  - Mission Planning Phase ends following completion of the mission brief. The crew is ready to execute a mission.
- **Preflight Phase**
  - Preflight phase begins when UAV swarm is powered on for launch
  - Swarm Monitor and Ground Crew preflight swarm system
  - UAVs reports flight ready status system status indications
  - Swarm Commander verifies UAV swarm is in flight ready status
  - Swarm Commander establishes communication with JTFC2
  - Preflight phase ends when the swarm is in flight ready status
- **Ingress Phase**
  - Ingress phase begins when the Swarm Commander receives launch clearance from the JTFC2
  - Swarm Commander receives launch command from JTFC2
  - Ground Crew loads UAVs on launcher
  - Swarm Commander commands Ground Crew to launch swarm
  - UAVs transmits status and pose messages (position and orientation data) to GCS (on-going throughout each phase of mission)
  - Launch phase ends when the last UAV has left the launcher
  - UAV swarm maneuvers to clear obstacles then levels off at OnStation altitude
  - Swarm Monitor monitors system health (on-going throughout each phase of mission)
  - Swarm Commander supervises flight path to OnStation area
  - The Ingress phase ends when the swarm arrives at OnStation waypoint and assigned altitude
- **OnStation Phase**

- The OnStation phase begins when the swarm reaches the assigned OnStation area
- Swarm Monitor monitors system health
- Swarm follows search pattern
- **Conduct Aerial Reconnaissance**
  * Collect data using IR, EO, synthetic aperture radar and LIDAR
  * Locate and image roads, bridges, airfields, ports, utility systems, and other infrastructure
  * Provide aerial reconnaissance for urban SAR missions
  * Locate and image affected members of the HN population
  * Transmit real-time video to JTFC2 and navy ships
  * Establish communication between military assets and applicable HN NGOs
  * Perform communication relay for JTF to reach other assets
- The OnStation phase ends when swarm reaches bingo fuel or the JTFC2 has commanded the swarm to return to base

- **Egress Phase**
  - The egress phase begins when egress criteria have been met and the swarm is on a flight path to return to base or ship
  - Swarm Monitor monitors system health
  - Swarm Commander supervises flight path
  - Swarm arrives at recovery way point
  - Swarm Commander executes auto-land
  - The Egress phase ends when the swarm has landed (land-based) or has been captured in a recovery system (shipboard)
- **Postflight Phase**
  - The postflight phase begins once the swarm has landed or been captured
  - Swarm Commander, Swarm Monitor, and Ground Crew perform post-flight procedures and inspections on UAVs
  - Swarm Commander and Swarm Monitor perform post-flight procedures and inspections on GCS
  - Ground Crew perform post-flight procedures and inspections on launcher and recovery system (as applicable)
  - Swarm Commander, Swarm Monitor, and Ground Crew debrief mission

- Swarm Commander, Swarm Monitor, and Ground Crew generate after action report
- The Postflight phase ends once the mission after action report has been completed

## B.3 MIO Design Reference Mission

This DRM outlines an operational context for a notional UAV swarm MIO mission. Specific swarm unmanned system scenarios may be developed from this DRM, for designing operational architectures solutions. The purpose of this mission is to seek innovations in offensive and defensive tactics, autonomy algorithms, concepts of operation, and to support ongoing research in system architecture design. The analysis question motivating the development of this DRM is: How could a UAV swarm be employed to enhance mission effectiveness for MIO missions?

This DRM contains a specific example of a basic UAV swarm MIO scenario to demonstrate aerial swarming capabilities by deploying a team of homogeneous UAVs to perform MIO operations. From this basic scenario, offensive and defensive tactics can be designed, while different communication and coordination architectures can be explored. Additional variations on this scenario will provide context for follow-on analysis from the use cases generated.

### B.3.1 Mission Background

A maritime interdiction (also called interception) operation is a U.S. Navy mission, typically executed with maritime air support that involves surveillance and interception of private or commercial vessels, boarding and searching of suspect vessels, and detaining, diverting or seizing vessels found in violation of United Nations (UN) sanctions or other international laws [218]. The MIO mission falls under sea control and maritime security, two of the five core capabilities of the US Navy, with the other four being: forward presence, deterrence, and maritime power projection. Command and Control for Joint Maritime Operations (Joint Publication 3-32) is one of several foundational sources of US Navy doctrine for MIO, and defines it as “efforts to monitor, query, and board merchant vessels in international waters to enforce sanctions against other nations such as those in support of United Nations Security Council resolutions and/or prevent the transport of restricted good” [219]. This mission supports the safe passage of maritime vessels, and protection sea lines of communication (SLOC) and air lines of communication (ALOC), by intercepting contraband, preventing drug smuggling, and combating piracy.

The historical roots of MIO in maritime warfare can be traced to embargoes, blockades, anti-piracy and commerce raiding. MIO's legal standing is based upon international law (such as United Nations Security Council Resolution (UNSCR)), national authorities, or regional authorities. Recent examples of MIO missions include: supporting UN sanctions against Haiti, the Balkans, and Iraq; and protecting freedom of navigation and disruption of terrorist supply lines (under UNSCR 1373) around the Arabian peninsula as part of Operation Enduring Freedom [220].

The handling of MIO missions is dependent upon well-timed intelligence and the associated risk assessment of an intervention, or boarding. Boardings are classified as compliant, non-compliant, and opposed [219]. Compliant boardings describe a situation in which a suspect vessel obeys the commands of the on-scene commander (OSC), non-compliant boardings occur when the suspect vessel fails to comply with OSC instructions or attempts to impede or delay the boarding team, and opposed boardings are characterized by active or passive means to resist the boarding or hostile actions [218]. Standard navy boarding teams do not have opposed boarding or airborne insertion capabilities; however, U.S. Navy ships may be tasked to support other embarked forces with such capabilities [218]. This DRM scope focuses on compliant and non-compliant boardings that can be conducted by a standard U.S. Navy boarding team.

The visit, board, search and seizure (VBSS) part of the MIO mission has typically been conducted by eight-man teams of Sailors, Coastguardsmen, Marines, and law enforcement personnel using RHIBs for intercepting the targets of interest [219]. A typical VBSS operation for a boarding consists of the following main phases [218]:

- deployment—encompasses the boarding team's trip (on a RHIB) from the mother ship to the suspect vessel
- embarkation/insertion—covers the boarding process
- objective—the longest phase; describes the searching, inspecting, rescuing, seizing, or other operations
- extraction—involves the boarding team's exit from the suspect vessel and concludes when they have returned to the mother ship.

### B.3.2 Operational Concept

Following the sanctioning of the MIO (via UNSCR or other means), the responsible combatant commander issues an operational order (OPORD) that complies with the resolution. Then the fleet commanders issue operational tasking order supplements (OPTASK SUPPS) to outline: rules of engagement, descriptions of suspect vessels, classification criteria, reporting procedures, questions to ask the suspect vessel, and materials to be identified and seized [218]. A UAV swarm MIO operational concept is featured in the OV-1 diagram in Figure B.6. The following assets are used in the scenario:

- CVN—includes the Surface Warfare Commander (SUWC), and provides overall C2 of strike group
- DDG—includes Maritime Interdiction Operations Commander (MIOC), boarding team, RHIB, and UAV swarm
- UAV swarm – launched from the DDG, responsible for providing close-in, real-time audio and video data from the COI
- RHIB—includes the embarked navy boarding team
- P-8A—a maritime patrol aircraft that provides long-range, airborne ISR capabilities
- MH-60R—provides organic, airborne ISR capabilities.

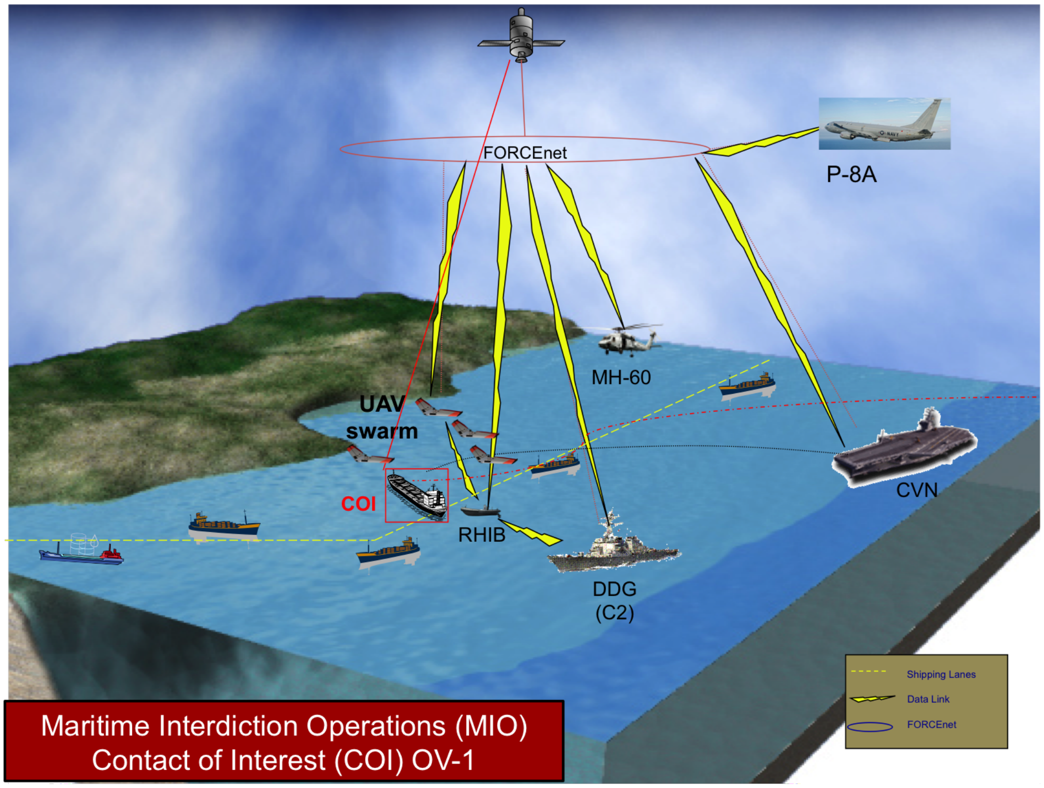



Figure B.6. OV-1 Operational Concept for MIO. Adapted from [221].

### B.3.3 Projected Operating Environment

**Environmental Conditions**

The MIO mission domain is composed of seas, oceans, bays, estuaries, islands, coastal regions, and the airspace overhead the aforementioned areas [219]. The UAV swarm is expected to operate in:

- daytime
- temperature: 60 – 100°F
- maximum operating altitude: 3000 ft AGL
- light precipitation
- no icing
- wind gusts less than 20 kts
- multiple electromagnetic emissions within a wide radio frequency range

This DRM uses the Indonesian islands between Sumatra and Java, Figure B.7, as the operational area.


Figure B.7. Counter-smuggling MIO in the Sunda Strait. Source: [222].

**Assumed Threat Environment**

Threats to the success of the swarm MIO mission are expected to come from potential adversary small arms fire, and from the environment in the form of rainstorms and geographically constrained operating areas. A mobile weather station on-board the DDG will be used to monitor weather conditions.

### B.3.4 Mission Success Requirements

For the mission to be successful, the swarm must:

- collect information regarding suspect vessel position and activities,
- distribute collected information to OSC and boarding team, and
- provide advanced warning of potential threats from suspect vessel to boarding team.

### B.3.5 Mission System Requirements

System requirements for MIO mission include those which support: maritime domain awareness, flexible force packages, sensors and data management systems, networks, and VBSS capabilities [223].

### B.3.6 Mission Definition

The main reference mission provides a level of detail necessary for collecting measures to assess mission success requirements. As technology and tactics develop, additional mission variations will be added to allow for increased aircraft, incorporation of sensors and weapons, different tactics and plays, other unmanned or manned assets, and different environmental conditions. The capability needs statement for the main UAV swarm MIO mission is: the U.S. Navy needs a cost-effective means to investigate suspect vessels potential threats prior to embarking boarding team personnel on the vessel.

#### OPSIT

The OPSIT describes the assumptions being made about the mission's operational environment that have implications for logistics, deployment, and time required to complete the mission. The following OPSIT pertains to a U.S. Navy MIO task force supporting the Indonesian Navy in conducting compliant or non-compliant boardings to support MIO for suspected smugglers in the Sunda Strait.

The Indonesian Navy has increased patrols of the waterways between Sumatra and Java to counter drugs, human trafficking, and other forms of smuggling. This unified effort involves cooperation with the multiple national law enforcement agencies, including the national police anti-trafficking force. Hundreds of small vessels operate in these waters daily, and the Sunda Strait is used as a major shipping link heading into and out of the Java Sea. To increase the effectiveness of this mission, the Indonesian Navy has requested the assistance of the U.S. Navy. The U.S. Navy's mission is to identify, board (if necessary), and stop any illegal shipping in this region, focusing particularly on the inter-island traffic. If smugglers are found, the U.S. Navy will detain the crew and their vessel until they can be turned over to other law enforcement agencies [222].

### B.3.7 Mission Execution

The following mission narrative describes two general types of UAV swarm MIO reference missions, compliant and non-compliant boardings. "If-then" logic is incorporated into the narrative to show possible alternate paths that could occur during mission. General rules describe any events that are recurring, or that can occur at any point during the mission, are designated separately from the "if-then" event sequence. Mission start time (T+00) occurs at the beginning of the preparation phase. The mission narrative describes the mission in general, and makes the distinction between the two different scenarios in the OnStation phase. The Staging, Mission Planning, Preflight, Ingress, Egress, and Postflight phases are similar in both scenarios and are thus described once. The first scenario's OnStation phase describes a notional compliant boarding while the second describes a non-compliant boarding. The second scenario is used to support the human subject research model validation (Appendix C).

**UAV Swarm MIO Mission Narrative**

**Staging Phase** (completed once)

- Staging phase begins once the UAV swarm is in its travel configuration and arrives onboard the ship
- Swarm Commander, Swarm Monitor, and Ground Crew unpack and assemble swarm system from its travel configuration
- Swarm Commander, Swarm Monitor, and Ground Crew configure system in preparation for executing a flight mission upon receipt of orders
- Ensure GCS has up-to-date digital charts with
  - AOR
  - airfields and landing zones
  - natural and man-made hazards (small islands, oil rigs),
  - restricted airspace
  - no-fly zones depicted

- Determine existing communication capabilities of the other players
- Staging phase ends when the swarm system has been assembled, components tested, communications established, and the crew is ready to receive and plan a mission.

**Mission Planning Phase**

- The Mission Planning Phase begins when the Swarm Commander receives a tasking order to be prepared to execute a MIO mission
- Update communication capabilities of other players
- Review ATO, SPINS, and any other relevant OPTASK from the joint task force commander. These documents contain aircraft callsigns, mission types, coordination frequencies, airspace boundaries, ingress/egress corridors, failsafe rally waypoints, etc.
- Update GCS digital charts to reflect any changes from previous configuration
- Plan ingress and egress routes to and from the search area, OnStation waypoint, recovery waypoint, and failsafe rally waypoint
- Plan search pattern to cover area(s)
- Review data collection plan
- Check weather
- Conduct mission brief
- Mission Planning phase ends following completion of the mission brief. The crew is ready to execute a mission.

**Preflight Phase**

- Preflight Phase begins when national intelligence asset reports the existence of a COI to DDG
- The SUWC, who is embarked on CVN-75 USS Harry S. Truman, tasks the MIOC on-board DDG-108 USS Wayne E. Meyer, to act as OSC and prosecute the COI
- MIOC tasks P-8A and MH-60R with search areas, and commands swarm team to preflight the swarm
- Swarm Monitor and Ground Crew preflight swarm system
- P-8A and MH-60R conduct surface search, and provide maritime domain awareness throughout the mission
- P-8A reports position of COI to MIOC

- Preflight Phase ends when the swarm system is in flight ready status

**Ingress Phase**

- Ingress phase begins when the MIOC orders the swarm to be launched
- Swarm Commander receives launch clearance from MIOC and initiates the launch process
- MIOC launches boarding team on RHIB
- UAVs transmits status and pose messages (position and orientation data) to GCS (assumed to be on-going throughout each in-flight phase of mission and not modeled for simplicity)
- UAV swarm is launched and transits to OnStation waypoint
- Swarm Monitor monitors system health (on-going throughout each phase of mission)
- Swarm Commander supervises flight path to OnStation area
- The Ingress phase ends when the UAV swarm arrives at OnStation waypoint

**OnStation Phase for Compliant Boarding**

- The OnStation phase begins when the UAV swarm reaches the assigned OnStation area
- Swarm Commander monitors OnStation activities l
- Swarm Monitor monitors swarm fuel status (ongoing)
- Swarm follows search pattern
- Swarm mission tasks:
  - surveil COI for RF transmissions, small arms threat, possible smuggling activity,
  - collect data on COI infrastructure and count number of personnel on board using IR, EO, synthetic aperture radar and LIDAR
  - transmit real-time video, images, and electronic surveillance data of the COI to MIOC and boarding team RHIB
  - Perform communication relay between boarding team and MIOC and other players
  - UAV swarm uses tactics from the MASC playbook such as: Search, Monitor, Track, and Communication Relay to support the mission
- P-8A and MH-60R continue providing maritime domain awareness by providing Link-16 tracks and identifying them with on-board sensors (asynchronous).

- National intelligence assets provide intelligence updates to DDG (asynchronous).
- Boarding team boards COI, COI obeys boarding team instructions.
- Boarding team conducts search, then relays smuggler or non-smuggler information to MIOC. If smuggler, MIOC notifies Indonesian law enforcement. If non-smuggler, boarding team disembarks and returns to DDG.
- The OnStation phase ends when swarm reaches bingo fuel, or the OnStation time period has concluded, or the MIOC has commanded the swarm to return to base

**OnStation Phase for Non-Compliant Boarding**

- The OnStation phase begins when the UAV swarm reaches the assigned OnStation area
- Swarm Commander monitors OnStation activities
- Swarm Monitor monitors swarm fuel status (ongoing)
- UAV swarm mission tasks:
  - based on COI location data from the P-8A, surveil area for COI
  - collect RF transmissions, evidence of small arms threat, smuggling evidence
  - collect data on COI infrastructure and count number of personnel on board using IR, EO, synthetic aperture radar and LIDAR
  - transmit real-time video, images, and electronic surveillance data of the COI to FORCENET and boarding team RHIB
  - Perform communication relay between boarding team and MIOC and FORCENET
- Swarm uses tactics from the MASC playbook to support the mission
- Swarm Commander intercepts radio communication and sees personnel movement on video that indicates a non-compliant posture from COI
- Personnel on COI begin shooting at swarm *(does this prompt a tactics change?)*
- P-8A and MH-60R continue providing maritime domain awareness by providing Link-16 tracks and identifying them with on-board sensors.
- National intelligence assets provide intelligence updates to MIOC
- Boarding team boards COI and COI does not obey boarding team instructions
- Boarding team conducts search, finds contraband, and reports findings to MIOC
- MIOC notifies Indonesian law enforcement.
- Indonesian law enforcement arrives to detain COI, boarding team disembarks and transits back to DDG, MIOC commands swarm to return to ship

- The OnStation Phase ends when the MIOC has commanded the swarm to return to ship.

**Egress Phase**

- The Egress Phase begins when the OnStation support requirements have been met and the swarm is ready to return to the ship
- Swarm transits to the terminal way point and executes landing sequence
- Swarm Monitor monitors system health
- The Egress Phase ends when the swarm has landed or has been captured in a recovery system onboard the DDG

**Postflight Phase**

- The Postflight Phase begins once the swarm has landed or been captured
- Swarm Commander, Swarm Monitor, and Ground Crew perform post-flight procedures and inspections on UAVs
- Swarm Commander and Swarm Monitor perform post-flight procedures and inspections on GCS
- Ground Crew perform post-flight procedures and inspections on launcher and recovery system (as applicable)
- Swarm Commander, Swarm Monitor, and Ground Crew debrief mission
- Swarm Commander, Swarm Monitor, and Ground Crew generate after action report
- The Postflight Phase ends once the mission after action report has been completed

### B.3.8 Measures

This DRM is designed to provide the necessary context for a system under development to assess current system capabilities and tactics. Notional key system characteristics include: interoperability, reliability, and maintainability. The following measures from the Universal Naval Task List [224], may be useful for evaluating the effectiveness of the system for meeting mission requirements:

**NTA 2.4.4.4 Evaluate the Threat**

To evaluate and assess threat (or potential threat) forces, military and non-military capabilities, limitations, centers of gravity, and critical vulnerabilities. To assess the enemy in terms of mobilization potential, order of battle (ground, air, maritime, electronic), tactical organization (including allied forces) and dispositions, doctrine, military capabilities, command and control, personalities including history of key leaders' performance, communications and information systems, current activities and operating patterns, and decision making processes. **(JP 2-0 Series, MCDP 2, MCWP 2-1, NDP 2, NWP 2-01)**

| | | |
|---|---|---|
| M1 | Percent | Of enemy branches and sequels were correctly identified during planning. |
| M2 | Percent | Of new processed intelligence data integrated within targeting cycle. |
| M3 | Percent | Of forecasted significant enemy actions were false alarms. |

**NTA 5.1 Acquire, Process, Communicate Information, and Maintain Status**

To obtain information on the mission, enemy forces, neutral/non-combatants, friendly forces, terrain, and weather. To translate that information into usable form and to retain and disseminate it. This task includes disseminating any type information. **(JP 1, 2-0 Series, 3-0, 6 Series, NDP 6, NWP 5-01 Rev A, MCDP 6, MCWP 6-22)**

| | | |
|---|---|---|
| M1 | Percent | Of units are in communication with commander throughout planning and execution. |
| M2 | Hours | To process status information and disseminate to subordinate units. |
| M3 | Percent | Of available information examined and considered in latest status report. |

**5.1.1.1.2.2 Relay Communications**

To pass information which cannot reach its targeted audience directly. This includes the use of aircraft for tactical relay. **(JP 3-0, 6-0, 6-02, NDP 6)**

| | | |
|---|---|---|
| M1 | Number | Messages relayed. |
| M2 | Minutes | To relay required messages. |
| M3 | Percent | Correct messages received. |

**MCT 4.2.2.4 Conduct Repair**

To conduct repair operations on equipment. Repair is the return of an item to serviceable condition through correction of a specific failure or unserviceable condition. The repair cycle starts when the maintenance activity removes an unserviceable part or reparable component. It ends when the maintenance activity reinstalls the replacement part or reparable component, and places the equipment back in service. **(MCWP 4-11.4)**

| | | |
|---|---|---|
| M1 | Time | Average repair cycle. |
| M2 | Man-hours | Of repair activity conducted per day. |
| M3 | TBD | |

The current Innoslate and Monterey Phoenix models for the MIO UAV swarm scenario are not capable of collecting data to support the measures listed above, but can be considered in future work.

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX C: Human Subjects Research Exercise

This appendix covers the scenario information, instructions, and survey given to participants in the human subjects research exercise for limited-scope model usability validation. The purpose of the exercise is to gather prospective user feedback on MASC and determine if there is a difference in perceived workload between mission planning at the tactics level versus the play level. The selected participants were divided into two groups, presented with an abbreviated version of the MIO non-compliant boarding scenario and asked to assemble a UAV swarm mission plan using the MASC playbook. The first group was tasked to assemble the mission plan using the tactics while the second group was assigned to build a mission plan using only the plays and without knowing the MASC tactic-play composition. Following the assembly of the mission plan, participants responded to a survey. Section C.1 applies to both groups, section C.2 applies only to group 1, and section C.3 applies only to group 2.

## C.1 Mission Background

This DRM outlines an operational context for you to use a swarm of homogeneous UAVs to support a notional MIO mission. The capability needs statement for the main UAV swarm MIO mission is: the U.S. Navy needs a cost-effective means to investigate suspect vessels potential threats prior to embarking boarding team personnel on the vessel. The analysis question motivating the development of this DRM is: How could UAV swarm be employed to enhance mission effectiveness for MIO missions?

The MIO mission is characterized by intelligence gathering and a risk assessment of an intervention, or boarding. Boardings are classified as compliant, non-compliant, and opposed [219]. Compliant boardings describe a situation in which a suspect vessel obeys the commands of the OSC, non-compliant boardings occur when the suspect vessel fails to comply with OSC instructions or attempts to impede or delay the boarding team, and opposed boardings are characterized by active or passive means to resist the boarding including hostile actions [218]. Standard navy boarding teams do not have opposed boarding or airborne insertion capabilities, however U.S. Navy ships may be tasked to support other forces with such capabilities [218]. This DRM scope focuses on a non-compliant boarding that can be conducted by a standard U.S. Navy boarding team.

The VBSS part of the mission is normally conducted by eight-man teams of Sailors, Coastguardsmen, Marines, and law enforcement personnel using RHIBs for intercepting the targets of interest [219]. A typical VBSS operation for a boarding consists of the following main phases [218]:

- deployment—encompasses the boarding team's trip (on their RHIB) from the mother ship to the suspect vessel
- embarkation/insertion – covers the boarding process
- objective—the longest phase; describes the searching, inspecting, rescuing, seizing, or other operations
- extraction—involves the boarding team's exit from the suspect vessel and concludes when they have returned to the mother ship.

## C.1.1 Operational Concept

Following the sanctioning of the MIO, the responsible combatant commander issues an OPORD. Then the fleet commanders issue OPTASK SUPPS to outline: rules of engagement, descriptions of suspect vessels, classification criteria, reporting procedures, questions to ask the suspect vessel, and materials to be identified and seized [218]. A UAV swarm MIO operational concept is featured in the OV-1 diagram in Figure C.1. The following assets are used in the scenario:

- CVN—includes the SUWC, and provides overall C2 of strike group
- DDG—includes MIOC, boarding team, RHIB, and UAV swarm
- UAV swarm – launched from the DDG, responsible for providing close-in, real-time audio and video data of the COI
- RHIB—includes the embarked navy boarding team
- P-8A—a maritime patrol aircraft that provides long-range, airborne ISR capabilities
- MH-60R—provides organic, airborne ISR capabilities.

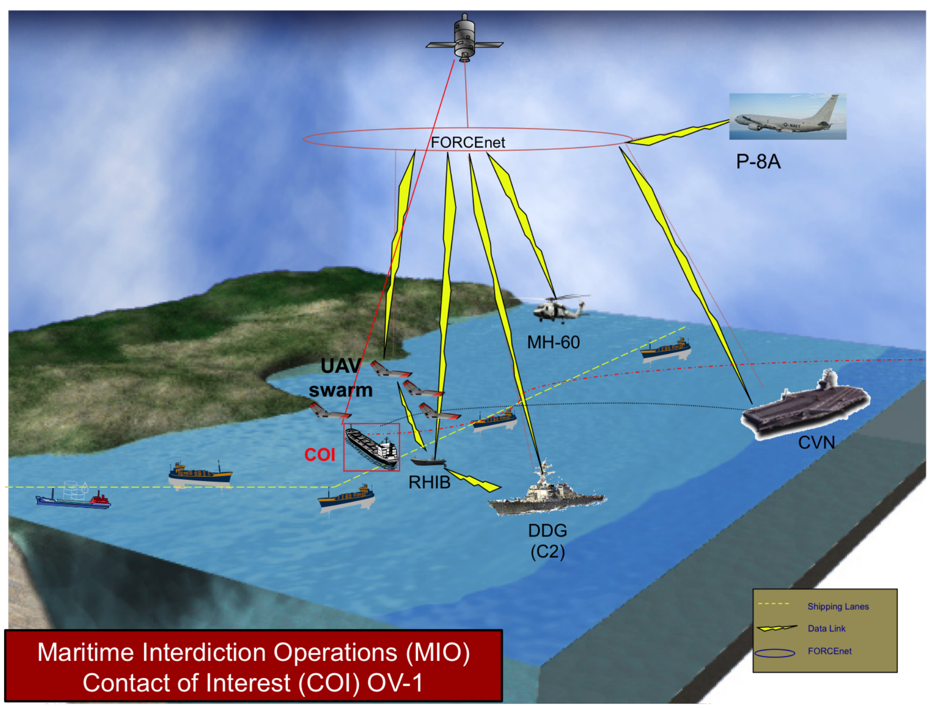



Figure C.1. OV-1 Operational Concept for MIO. Adapted from [221]

### C.1.2 Projected Operating Environment

The MIO mission domain is composed of seas, oceans, bays, estuaries, islands, coastal regions, and the airspace overhead the aforementioned areas [219]. The UAV swarm is expected to operate in:

- daytime
- temperature: 60 – 100°F
- maximum operating altitude: 3000 ft AGL
- light precipitation
- no icing
- wind gusts less than 20 kts
- multiple electromagnetic emissions within a wide radio frequency range

This DRM uses the Indonesian islands between Sumatra and Java, Figure C.2, as the operational area.


Figure C.2. Counter-smuggling MIO in the Sunda Strait. Source: [222].

**Assumed Threat Environment**

Threats to the success of the UAV swarm MIO mission are expected to come from potential adversary small arms fire, and from the environment in the form of rainstorms and geographically constrained operating areas. A mobile weather station on-board the DDG will be used to monitor weather conditions.

### C.1.3 Mission Success Requirements

For the mission to be successful, the surveillance assets must: collect information regarding suspect vessel position and activities, distribute collected information to OSC and boarding team, and provide advanced warning of potential threats from any vessel to the boarding team.

### C.1.4 Operational Situation

The OPSIT describes the assumptions being made about the mission's operational environment that have implications for logistics, deployment, and time required to complete the mission. The following OPSIT pertains to a U.S. Navy MIO task force supporting the Indonesian Navy in conducting non-compliant boardings to support MIO for suspected smugglers in the Sunda Strait.

The Indonesian Navy has increased patrols of the waterways between Sumatra and Java to counter drugs, human trafficking, and other forms of smuggling. This unified effort involves cooperation with the multiple national law enforcement agencies, including the national police anti-trafficking force. Hundreds of small vessels operate in these waters daily, and the Sunda Strait is used as a major shipping link heading into and out of the Java Sea. To increase effectiveness, the Indonesian Navy has requested the assistance of the U.S. Navy. The U.S. Navy's role is to identify suspect smugglers, board (if directed), and halt illegal shipping, focusing particularly on the inter-island traffic. If smuggling is confirmed, the U.S. Navy will detain the crew and their vessel until they can be turned over to local law enforcement agencies [222]. The ROE for this mission do **not** permit UAVs to use weapons.

## C.2 Mission Execution for Group 1

The following mission narrative describes a notional non-compliant boarding scenario. The Staging, Mission Planning, Preflight, and Postflight phases are for background information and not covered by the UAV swarm playbook. The playbook focuses on the inflight phases: Ingress, OnStation and Egress. For this scenario, the UAV swarm consists of 20 homogeneous UAVs in terms of capabilities and behaviors. If one UAV fails, the remaining UAVs collectively adjust UAV tasking to assume the duties of the lost UAV. Each UAV carries the following intelligence collection sensors: radar, EO, IR, chemical, LIDAR, and electronic surveillance (for signal collection). The swarm system includes the UAV swarm, GCS, and launch and recovery systems.

The UAV swarm team is composed of a Swarm Commander, Swarm Monitor, and Ground Crew. The Swarm Commander selects the swarm tactics and is responsible for the overall execution of the mission. The Swarm Monitor oversees the health and function of the swarm, and separates errant individual UAVs from the swarm [3]. The Ground Crew is responsible for preflight, launch, recovery, and postflight duties.

The individual UAVs fall within the DOD Group 1-2 UAV classes. The swarm is designed to operate at a low autonomy level for the MIO mission, requiring the Swarm Commander to respond to changes in the scenario by initiating the swarm tactics. The swarm may be divided into sub-swarms, which are the smallest unit available for tasking with a swarm tactic. A sub-swarm consists of five or more agents. The agents can be assumed to operate throughout the notional 90-minute mission without pausing to re-fuel. The following narrative guides you through the template (Figure C.4) you will use to assemble your mission plan.

### C.2.1 UAV Swarm MIO Mission Narrative for Group 1

**Staging Phase** (completed once)

- Staging phase begins once the swarm system is in its travel configuration and arrives onboard the ship
- Swarm Commander, Swarm Monitor, and Ground Crew unpack and assemble swarm system from its travel configuration
- Swarm Commander, Swarm Monitor, and Ground Crew configure system in preparation for executing a mission upon receipt of orders

- Ensure GCS has up-to-date digital charts with
  - AOR
  - natural and man-made hazards (small islands, oil rigs),
  - restricted airspace
  - no-fly zones depicted
  - airfields and landing zones
- Determine existing communication capabilities of the other players
- Staging phase ends when the UAV swarm has been assembled, components tested, and communications established. The crew is ready to receive and plan a mission.

**Mission Planning Phase**

- The Mission Planning Phase begins when the Swarm Commander receives a tasking order to execute a MIO mission
- Update communication capabilities of other players
- Review ATO, SPINS, and any other relevant OPTASK from the joint task force commander. These documents contain aircraft callsigns, mission types, coordination frequencies, airspace boundaries, ingress/egress corridors, failsafe rally waypoints, etc.
- Update GCS digital charts to reflect any changes from previous configuration
- Plan ingress and egress routes to and from the search area, OnStation waypoint, recovery waypoint, and failsafe rally waypoint
- Plan search pattern to cover area(s)
- Review data collection plan
- Check weather
- Conduct mission brief
- Mission Planning Phase ends following completion of the mission brief. The crew is ready to execute a mission.

**Preflight Phase**

- Preflight Phase begins when national intelligence asset reports the existence of a COI to DDG
- The SUWC, who is embarked on CVN-75 USS Harry S. Truman, tasks the MIOC on-board DDG-108 USS Wayne E. Meyer, to act as OSC and prosecute the COI

- MIOC tasks P-8A and MH-60R with search areas, and commands swarm team to preflight the swarm
- Swarm Monitor and Ground Crew preflight swarm system
- P-8A and MH-60R conduct surface search, and provide maritime domain awareness throughout the mission
- P-8A reports position of COI to MIOC
- Preflight Phase ends when the swarm system is in flight ready status

**Ingress Phase** (*playbook starts here*)

- Ingress Phase begins when the MIOC orders the swarm to be launched
- Swarm Commander receives launch clearance from MIOC and initiates the launch process *(select tactic)*.
- MIOC launches boarding team on RHIB
- UAVs transmits status and pose messages (position and orientation data) to GCS (assumed to be on-going throughout each in-flight phase of mission)
- Swarm is launched and transits to OnStation waypoint
- Swarm Monitor monitors swarm health (on-going throughout each phase of mission)
- Swarm Commander supervises flight path to OnStation area
- Swarm flies ahead of RHIB to collect intelligence
- The Ingress Phase ends when the swarm arrives at OnStation waypoint

**OnStation Phase for Non-Compliant Boarding**

- The OnStation Phase begins when the swarm reaches the assigned OnStation area
- Swarm Commander monitors OnStation activities
- Swarm Monitor monitors swarm fuel status (ongoing)
- UAV swarm mission tasks:
  - based on COI location data from the P-8A, surveil area for COI
  - collect RF transmissions, evidence of small arms threat, smuggling evidence
  - collect data on COI infrastructure and count number of personnel on board using IR, EO, synthetic aperture radar and LIDAR
  - transmit real-time video, images, and electronic surveillance data of the COI to FORCENET and boarding team RHIB
  - Perform communication relay between boarding team and MIOC and FORCENET

- Swarm uses tactics from the MASC playbook to support the mission *(select tactics)*
- Swarm Commander intercepts radio communication and sees personnel movement on video that indicates a non-compliant posture from COI
- Personnel on COI begin shooting at swarm *(does this prompt a tactics change?)*
- P-8A and MH-60R continue providing maritime domain awareness by providing Link-16 tracks and identifying them with on-board sensors.
- National intelligence assets provide intelligence updates to MIOC
- Boarding team boards COI and COI does not obey boarding team instructions
- Boarding team conducts search, finds contraband, and reports findings to MIOC
- MIOC notifies Indonesian law enforcement.
- Indonesian law enforcement arrives to detain COI, boarding team disembarks and transits back to DDG, MIOC commands swarm to return to ship
- The OnStation Phase ends when the MIOC has commanded the swarm to return to ship.

**Egress Phase**

- The Egress Phase begins when the OnStation support requirements have been met and the swarm is ready to return to the ship *(select tactic)*
- Swarm transits to the terminal way point and executes landing sequence
- Swarm Monitor monitors system health
- The Egress Phase ends when the swarm has landed or has been captured in a recovery system onboard the DDG

**Postflight Phase**

- The Postflight Phase begins once the swarm has landed or been captured
- Swarm Commander, Swarm Monitor, and Ground Crew perform post-flight procedures and inspections on UAVs
- Swarm Commander and Swarm Monitor perform post-flight procedures and inspections on GCS
- Ground Crew perform post-flight procedures and inspections on launcher and recovery system (as applicable)
- Swarm Commander, Swarm Monitor, and Ground Crew debrief mission
- Swarm Commander, Swarm Monitor, and Ground Crew generate after action report

- The Postflight Phase ends once the mission after action report has been completed

### C.2.2 Description of the MASC

The MASC uses a modular, composable framework for defining UAV swarm missions. Each mission is composed of five sequential phases, each phase is composed of one or more tactics, each tactic is composed of one or more plays, and each play is composed of one or more algorithms. Modifiable parameters such as altitude, velocity, and geographic waypoints enable the plays to be tailored to mission specific conditions. These elements are designed to be reusable for different missions. The architecture is designed so that the operator can build a mission at the tactics level, and the composite plays are listed for background information. Figure C.3 shows the overall picture of the MASC architecture.

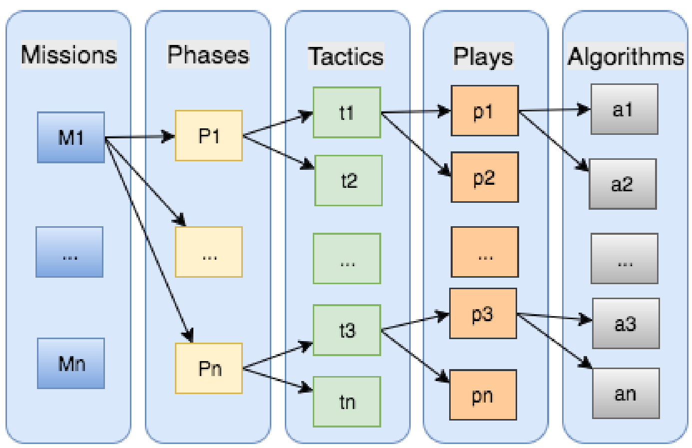


Figure C.3. MASC Architecture

Given the MIO scenario you have just read, build a swarm mission plan to support the mission using the tactics in the MASC playbook and the template provided. Table C.1 describes the tactics, along with the plays that comprise it. Your mission plan should consist of a sequence of tactics in an activity model.

Table C.1. MASC Tactics

| Tactic ID | Tactic Name | Description | Play ID | Play Name |
|---|---|---|---|---|
| t1 | Ingress | First inflight tactic of each mission. Includes launching, transiting to OnStation waypoint, and activating sensors. | p8.2 | Sensors OFF |
| | | | p1 | Launch |
| | | | p2 | Transit to waypoint |
| | | | p8.1 | Sensors ON |
| t2 | Evasive search | Uses a random pattern to confuse adversary. Keeps adversary guessing using randomization of maneuver and sensors at the expense of efficiency. | p8.1 | Sensors ON |
| | | | p8.3 | Sensors EMCON |
| | | | p14 | Random pattern |
| t3 | Efficient search | Used for methodical searching, wherein efficiency is prioritized over the risk of being predictable to the adversary. Multiple search pattern options. | p8.1 | Sensors ON |
| | | | p10 | Ladder pattern |
| | | | p11 | Expanding square pattern |

Continued on next page

**Table C.1 – continued from previous page**

| Tactic ID | Tactic Name | Description | Play ID | Play Name |
|---|---|---|---|---|
| | | | p12 | Constricting square pattern |
| | | | p13 | Grid pattern |
| t4 | Track | Used for maintaining sensor focus on a target and continuing to follow the target. Air-to-ground targets enact the Follow target play and air-to-air targets use the Tail following play. | p8.1 | Sensors ON |
| | | | p17 | Follow target |
| | | | p20 | Tail following |
| t5 | Communication relay | If communication relay is primary task, then swarm is scattered in grid pattern at specified altitude to optimize relay of communication to other units. In grid pattern, individual UAVs orbit within an assigned box and spacing is based on desired coverage probability, area to be covered and number of agents in swarm. If other maneuver patterns are used, then communication will be relayed while performing that maneuver as a secondary task. | p8.1 | Sensors ON |

Continued on next page

**Table C.1 – continued from previous page**

| Tactic ID | Tactic Name | Description | Play ID | Play Name |
|---|---|---|---|---|
| | | | p8.4 | Transmit video |
| | | | p10 | Ladder pattern |
| | | | p11 | Expanding square pattern |
| | | | p12 | Constricting square pattern |
| | | | p13 | Grid pattern |
| | | | p18 | Forward communication |
| t6 | Attack | Offensive tactic reserved for missions operating under ROE that allow weapons to be fired. Must have authorization decision before attack tactic can be enabled. | p8.1 | Sensors ON |
| | | | p15 | Weapon armed |
| | | | p20 | Smart greedy shooter |
| | | | p21 | Patrol box shooter |
| | | | p22 | Wingman shooter |

Continued on next page

**Table C.1 – continued from previous page**

| Tactic ID | Tactic Name | Description | Play ID | Play Name |
|---|---|---|---|---|
| | | | p16 | Weapon fire |
| t7 | BDA | Battle damage assessment. Used to assess damage to target that has been attacked, and determine if additional attacks are required. | p8.1 | Sensors ON |
| | | | p11 | Expanding square pattern |
| t8 | Monitor | Used for monitoring an area for RF, radar returns, visual, voice traffic using all sensors or EMCON (just passive sensors such as ESM receivers, IR, EO). There are 2 maneuver pattern options. | p8.1 | Sensors ON |
| | | | p8.3 | Sensors EMCON |
| | | | p3 | Orbit |
| | | | p4 | Racetrack |
| t9 | Evade | A defensive tactic used when the swarm is threatened. Sensors can go passive (EMCON) or they can jam incoming signals. The swarm will alternate disperse and join maneuvers to confuse the threat. Tactic executes until threat no longer exists or Swarm Commander stops it. | p8.3 | EMCON |
| | | | p19 | Jam |
| | | | p6 | Join |

Continued on next page

**Table C.1 – continued from previous page**

| Tactic ID | Tactic Name | Description | Play ID | Play Name |
|---|---|---|---|---|
| | | | p7 | Disperse |
| t10 | Harass | An offensive air-to-air tactic used to disrupt the enemy by maintaining presence, and confusing the enemy. After reaching the designated area, the swarm performs electronic jamming on the enemy's radar while performing one of several different maneuver options. | p8.1 | Sensors ON |
| | | | p2 | Transit to waypoint |
| | | | p19 | Jam |
| | | | p6 | Join |
| | | | p7 | Disperse |
| | | | p3 | Orbit |
| | | | p20 | Tail following |
| t11 | Defend | A defensive tactic used to guard a high value asset or home base, in which the swarm transits to the designated area and operates in a racetrack pattern with sensors on and weapon armed. | p8.1 | Sensors ON |
| | | | p2 | Transit to waypoint |
| | | | p4 | Racetrack |
| | | | p15 | Weapon armed |

Continued on next page

**Table C.1 – continued from previous page**

| Tactic ID | Tactic Name | Description | Play ID | Play Name |
|---|---|---|---|---|
| t12 | Deter | A defensive tactic used to dissuade the enemy from operating in a particular area. It is a less escalated version of Defend, as it does not arm weapons. This tactic is useful when operating under ROE which do not permit armed UAVs. | p8.1 | Sensors ON |
| | | | p2 | Transit to waypoint |
| | | | p3 | Orbit |
| | | | p4 | Racetrack |
| t13 | Divide | Used to logically divide swarm into subswarms to concurrently perform multiple swarm tactics. Subswarm sensor setting options are ON or EMCON. Subswarms maneuver in Orbit pattern after division. | p8.1 | Sensors ON |
| | | | p8.3 | EMCON |
| | | | p5 | Split |
| | | | p3 | Orbit |
| t14 | Amass | Used to re-assemble subswarms into larger swarm after they have been divided. Subswarm sensor setting options are ON or EMCON. Subswarms maneuver in Orbit pattern after joining. | p8.1 | Sensors ON |
| | | | p8.3 | EMCON |
| | | | p6 | Join |

Continued on next page

**Table C.1 – continued from previous page**

| Tactic ID | Tactic Name | Description | Play ID | Play Name |
|---|---|---|---|---|
| | | | p3 | Orbit |
| t15 | Egress | Last tactic executed for each mission. Occurs after OnStation phase has finished and returns swarm to ship or base. | p2 | Transit to waypoint |
| | | | p9.1 | Terminal approach |
| | | | p8.2 | Sensors OFF |
| | | | p9.2 | Landing |

### C.2.3 Description of Plays

For your background information, the MASC swarm plays referenced in Table C.1 are described below. Plays are used in different combinations to form the tactics. Each tactic is composed of one or more sensor plays and one or more maneuver plays.

- **Launch ($p_1$)** is the play used for transitioning the UAVs from the airfield or ship into the airborne environment. The operator modifiable parameters are: number of UAVs to be launched, number of launchers, and the time interval in seconds between UAV launches. The time interval can be set to a minimum threshold to perform a rapid launch. Launch is part of the Ingress tactic.
- **Transit to waypoint ($p_2$)** can be used as a single play or in combination to perform various flight paths and patterns. It is used in the Ingress, Egress, Deter, Harass, and Defend tactics. Operator modifiable parameters include: transit altitude or altitude block (for evasive transits), and waypoint latitude/longitude.
- **Orbit ($p_3$)** is a play that enables the swarm to circle around a fixed geographic position or target. It is used in the Monitor, Harass, and Deter tactics. The operator modifiable parameters are: altitude or altitude block (to stack the UAVs at multiple altitudes), clockwise or counterclockwise flow, designated center waypoint latitude/longitude, and radius of orbit (in m or nm).
- **Racetrack ($p_4$)** is a play that enables the swarm to encircle a fixed geographic area or target in a racetrack pattern. It is used in the Monitor, Defend, and Deter tactics. The operator modifiable parameters are: search sensor coverage parameters, or altitude or altitude block (to stack the UAVs at multiple altitudes), clockwise or counterclockwise flow, designated center waypoint latitude/longitude, width, and length (in m or nm).
- **Split (logic-based) ($p_5$)** splits the swarm into sub-swarms based on parameters: health status, battery life remaining, number of sub-swarms, location, or proximity to a reference waypoint. This play supports the Divide tactic.
- **Join ($p_6$)** is used to return UAVs to their original swarm or sub-swarm after being split or dispersed. It is used in the Evade, Harass, and Amass tactics. The operator modifiable parameters are designated join time, specified joining waypoint latitude/longitude, and altitude block.
- **Disperse ($p_7$)** is a defensive play to scatter and separate the UAVs within a cylinder, to support the Evade, Harass, and Amass tactics. The operator modifiable parameters

are dispersion radius and altitude block.

- **Sensors ON ($\mathbf{p_{8.1}}$)** is the default sensor setting that turns on all available non-navigational sensors. Sensors are turned on after the UAV has been launched and has executed a positive rate of climb. Several OnStation tactics including both search tactics, Track, and Monitor begin with Sensors ON. The operator modifiable parameters are the individual ON or OFF settings for each individual sensor (i.e., radar, EO, IR, LIDAR, electronic surveillance measures).
- **Sensors OFF ($\mathbf{p_{8.2}}$)** is the default sensor setting that turns off all non-navigational sensors. Sensors (i.e., radar, EO, IR, LIDAR, electronic surveillance measures) are turned off just prior to landing, as part of the Egress tactic to prevent damage to sensors and injury to personnel.
- **Sensors EMCON ($\mathbf{p_{8.3}}$)** is the sensor setting used for evasive tactics. It turns the active sensors, such as radar and LIDAR, off and leaves the passive sensors (i.e., EO, IR) on. Sensors EMCON is part of the Monitor and Evade tactics.
- **Transmit video ($\mathbf{p_{8.4}}$)** is the play that enables swarm video to be transmitted to the receivers other than GCS, and it is part of the Communication relay tactic. The operator modifiable parameters are the receiving nodes' addresses and sensor data to be transmitted.
- **Terminal approach ($\mathbf{p_{9.1}}$)** sequences the swarm for landing according to the individual UAV's battery life and proximity to the landing waypoint. Operator modifiable parameters allow for a single terminal approach path or multiple terminal approach paths, normal glideslope or combat descent rate, and inbound headings. Terminal approach is part of the Egress tactic and it is used at the end of each mission.
- **Landing ($\mathbf{p_{9.2}}$)** defines the landing area for the swarm. User modifiable parameters are available for specifying the dimensions of landing area: radius of a circle or m x n dimensions of a rectangle, or a specified separation distance between UAVs other than the default. Terminal approach is part of the Egress tactic and it is used at the end of each mission.
- **Ladder pattern ($\mathbf{p_{10}}$)** is a pattern play used in the Search and Communication relay tactics. The pattern commences at a known datum and executes a long track leg, followed by 90° turn, a short leg, another 90° turn in the same direction, a long track leg of the same length as the previous long leg, a 90° turn in the opposite direction from previous turn, a short leg of the same length, etc. The track leg lengths are

dependent on the swarm size, sensor range, and the operator modifiable parameters: desired probability of detection, search area size, and altitude block.

- **Expanding square pattern ($\mathbf{p_{11}}$)** is commonly used in SAR missions. The pattern starts at a known datum, then transits to the first waypoint for a distance *d*, makes a 90° turn (all turns are in same direction), transits to next waypoint at a distance *d*, makes a 90° turn for a distance *2d*, makes another 90° turn, transits *2d* to next waypoint, performs a 90° turn, then transits *3d* to next waypoint, etc. The distance *d* is dependent on timeliness of the datum and the best known speed of target, and consideration of environmental conditions such as wind and current. The Efficient search, Communication relay, and BDA tactics all have the Expanding square pattern play option. The operator modifiable parameters are altitude block, datum latitude/longitude, and estimated speed of the target.
- **Constricting square pattern ($\mathbf{p_{12}}$)** supports the Efficient search and Communication relay tactics. Beginning from an offset distance *d*, it is performed in an inverse fashion of the Expanding square pattern, moving inward toward the last known position. *d* is dependent on timeliness of the datum and the best known speed of target, and consideration of environmental conditions such as wind and current. The operator modifiable parameters are altitude block, datum latitude/longitude, and estimated speed of the target.
- **Grid pattern ($\mathbf{p_{13}}$)** supports the Efficient search and Communication relay tactics. Grid pattern defines an operating area in terms of large rectangle broken into smaller rectangles. A lead UAV is collectively selected and acts as master UAV by allocating tasks and grid assignments to the other UAVs. Each UAV searches its assigned rectangle and reports back to the assigned lead, using communication relay via the other UAVs as necessary. The operator modifiable parameters are search area dimensions, desired area coverage, and altitude block.
- **Random pattern ($\mathbf{p_{14}}$)** maneuvers the swarm together in a random pattern by transiting to randomly generated waypoints within an operating area. Random pattern supports the Evasive transit tactic and its operator modifiable parameters are altitude block and search area dimensions.
- **Weapon armed ($\mathbf{p_{15}}$)** enables weapon arming when the swarm has reached a swarm ready state and if the Attack tactic is enabled for the mission. Availability of this play is ROE dependent and loaded during mission planning.

- **Weapon fire ($\mathbf{p_{16}}$)** enables the available weapon(s) to be fired. The operator modifiable parameters are weapon selection and manual or automatic mode. Automatic mode is designed for specific scenarios such as close-in ship support or a SEAD mission, and is ROE dependent. Manual mode is the normal mode of operation and requires the operator to manually initiate the weapon after a target has been identified. Weapon fire is part of the attack tactic.
- **Follow target ($\mathbf{p_{17}}$)** is used to follow and track a ground target and is part of the Track tactic. Operator modifiable parameters are offset slant range and altitude block.
- **Forward communication ($\mathbf{p_{18}}$)** is part of the Communication Relay tactic and it forwards voice or data link communication received from another unit to another unit.
- **Jam ($\mathbf{p_{19}}$)** uses various forms of electronic jamming and other electronic deception techniques to interfere with an enemy's radar and other C2 systems [159]. It is part of the Evade and Harass tactics. The operator modifiable parameters are frequency bands and temporal schedule.
- **Smart Greedy Shooter ($\mathbf{p_{20}}$)** was developed by ARSENL and improves on the original ARSENL greedy shooter by coordinating target allocation to ensure each adversary is engaged by no more than one friendly UAV. Each UAV will engage the adversary determined to be optimal via shortest distance to adversary [160]. This play supports the Attack tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.
- **Patrol Box Shooter ($\mathbf{p_{21}}$)** was developed as an ARSENL behavior and provides area-defense that incorporates smart shooter selection semantics in randomized patrol of defended area. UAVs individually determine patrol patterns within defended area without coordination or notification [160]. This play supports the Attack tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.
- **Wingman Shooter ($\mathbf{p_{22}}$)** was developed as an ARSENL behavior and coordinates target allocation to ensure each adversary is engaged by two friendly UAVs. The pair will engage the adversary determined to be optimal via shortest distance to adversary [160]. This play supports the Attack tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.
- **Tail following ($\mathbf{p_{23}}$)** used to follow other aircraft, friend or foe. The operator modi-

fiable parameters are altitude block, offset range (m), knock-it-off criteria (i.e., range from home or geographic boundary, or surrounded by x bandits).

### C.2.4 Group 1 Participant Assignment

Figure C.4 is a picture of the template you will use online in Draw.io to assemble your MIO mission plan using a combination of the tactics discussed above. The template shows the general mission flow by phase on the left and event triggers from the mission narrative you just read in the middle. Your task is to place the green tactics in the middle section in response to the event triggers. Place the tactics in temporal sequential order from left to right, and stack concurrent tactics (for sub-swarms) on top of each other. If there is a tactic you would like to use that is not included in the playbook, use the "option" tactic and describe its purpose and function in the survey. Begin assembling the mission plan when you are able to complete the task in one sitting and record the start and finish times for the task.

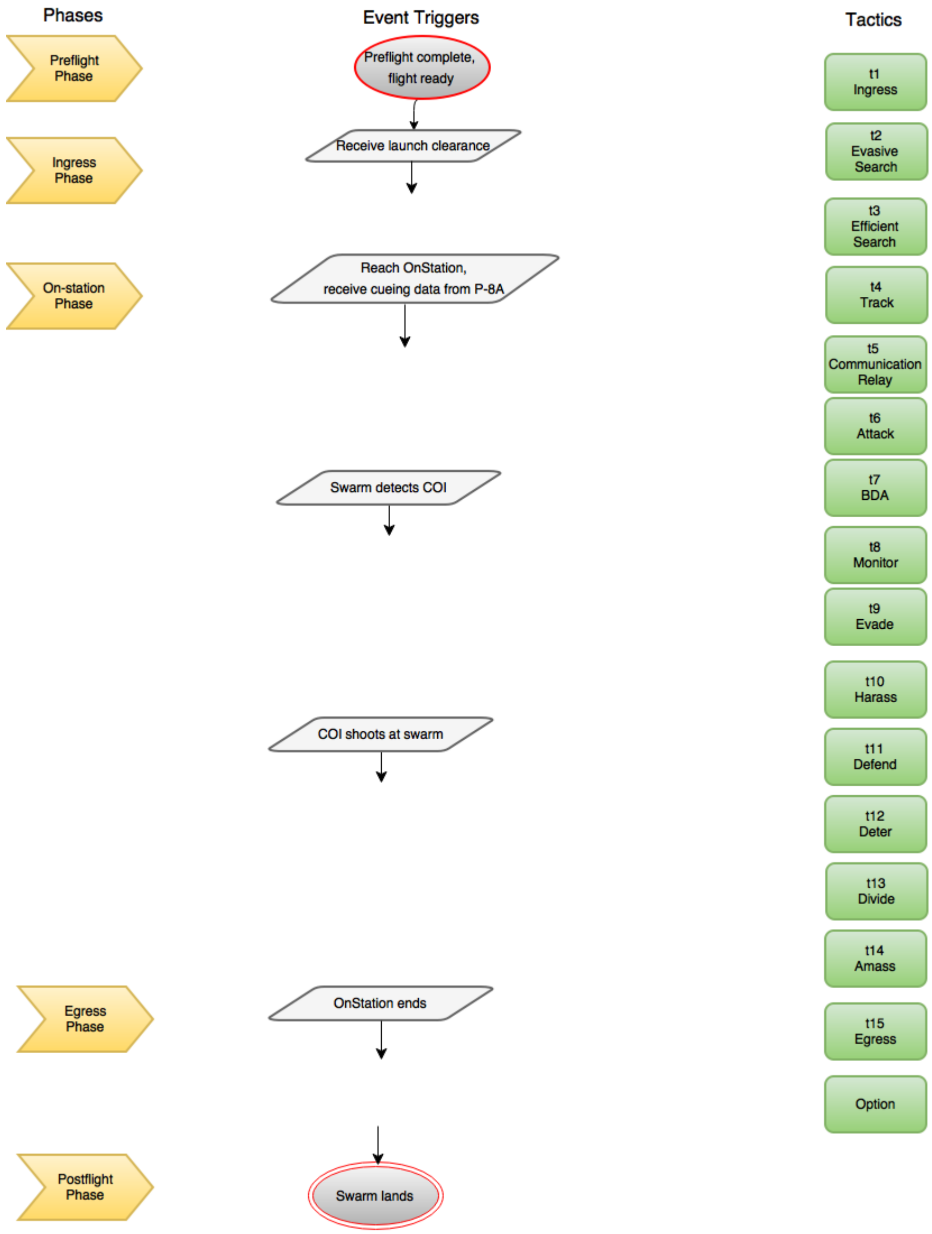



Figure C.4. Scenario Template

### C.2.5 Stakeholder Feedback Survey

1. How long did it take you to complete the swarm mission plan?

2. What is your military rank?

3. What is your military occupational specialty?

4. How many years of military operational experience do you have?

5. What is your experience with commanding or participating in UAV swarm missions?

6. What is your experience with commanding or participating in single UAV missions?

7. Does the structure of the mission phases support the execution of this mission? If not, how could it be improved?

8. How could this playbook of tactics be improved to better support this mission scenario?

The following questions are adopted from the NASA Task Load Index method [167]. Please answer the remaining questions based on the following rating scale:

1 = very low     2 = low     3 = medium     4 = high     5 = very high

9. How mentally demanding was the task?

10. How hurried or rushed was the pace of the task?

11. How successful were you in accomplishing what you were asked to do?

12. How hard did you have to work to accomplish your level of performance?

## C.3 Mission Execution for Group 2

The following mission narrative describes a notional non-compliant boarding scenario. The Staging, Mission Planning, Preflight, and Postflight phases are for background information and not covered by the UAV swarm playbook. The playbook focuses on the inflight phases: Ingress, OnStation and Egress. For this scenario, the UAV swarm consists of 20 homogeneous UAVs in terms of capabilities and behaviors. If one UAV fails, the remaining UAVs collectively adjust UAV tasking to assume the duties of the lost UAV. Each UAV carries the following intelligence collection sensors: radar, EO, IR, chemical, LIDAR, and electronic surveillance (for signal collection).

The UAV swarm team is composed of a Swarm Commander, Swarm Monitor, and Ground Crew. The Swarm Commander selects the swarm tactics and is responsible for the overall execution of the mission. The Swarm Monitor oversees the health and function of the swarm, and separates errant individual UAVs from the swarm [3]. The Ground Crew is responsible for preflight, launch, recovery, and postflight duties.

The individual UAVs fall within the DOD Group 1-2 UAV classes. The swarm is designed to operate at a low autonomy level for the MIO mission, requiring the Swarm Commander to respond to changes in the scenario by initiating the swarm plays. The swarm may be divided into sub-swarms, which are the smallest unit available for tasking with a swarm play. A sub-swarm consists of five or more agents. The agents can be assumed to operate throughout the notional 90-minute mission without pausing to re-fuel. The following narrative guides you through the template (Figure C.5) you will use to assemble your mission plan.

### C.3.1 UAV Swarm MIO Mission Narrative for Group 2

**Staging Phase** (completed once)

- Staging phase begins once the swarm system is in its travel configuration and arrives onboard the ship
- Swarm Commander, Swarm Monitor, and Ground Crew unpack and assemble swarm system from its travel configuration
- Swarm Commander, Swarm Monitor, and Ground Crew configure system in preparation for executing a mission upon receipt of orders
- Ensure GCS has up-to-date digital charts with

  - AOR
  - natural and man-made hazards (small islands, oil rigs),
  - restricted airspace
  - no-fly zones depicted
  - airfields and landing zones
- Determine existing communication capabilities of the other players
- Staging phase ends when the swarm system has been assembled, components tested, and communications established. The crew is ready to receive and plan a mission.

**Mission Planning Phase**

- The Mission Planning Phase begins when the Swarm Commander receives a tasking order to execute a MIO mission
- Update communication capabilities of other players
- Review air tasking order (ATO), special instructions (SPINS), and any other relevant operational tasking orders (OPTASK) from the joint task force commander. These documents contain aircraft callsigns, mission types, coordination frequencies, airspace boundaries, ingress/egress corridors, failsafe rally waypoints, etc.
- Update GCS digital charts to reflect any changes from previous configuration
- Plan ingress and egress routes to and from the search area, OnStation waypoint, recovery waypoint, and failsafe rally waypoint
- Plan search pattern to cover area(s)
- Review data collection plan
- Check weather
- Conduct mission brief
- Mission Planning Phase ends following completion of the mission brief. The crew is ready to execute a mission.

**Preflight Phase**

- Preflight Phase begins when national intelligence asset reports the existence of a COI to DDG
- The SUWC, who is embarked on CVN-75 USS Harry S. Truman, tasks the MIOC on-board DDG-108 USS Wayne E. Meyer, to act as OSC and prosecute the COI
- MIOC tasks P-8A and MH-60R with search areas, and commands swarm team to

preflight the swarm

- Swarm Monitor and Ground Crew preflight swarm system
- P-8A and MH-60R conduct surface search, and provide maritime domain awareness throughout the mission
- P-8A reports position of COI to MIOC
- Preflight Phase ends when the swarm is in flight ready status

**Ingress Phase** (*playbook starts here*)

- Ingress Phase begins when the MIOC orders the swarm to be launched
- Swarm Commander receives launch clearance from MIOC and initiates the launch process *(select plays)*.
- MIOC launches boarding team on RHIB
- UAVs transmits status and pose messages (position and orientation data) to GCS (assumed to be on-going throughout each in-flight phase of mission)
- UAV swarm is launched and transits to OnStation waypoint
- Swarm Monitor monitors swarm health (on-going throughout each phase of mission)
- Swarm Commander supervises flight path to OnStation area
- UAV swarm flies ahead of RHIB to collect intelligence
- The Ingress Phase ends when the swarm arrives at OnStation waypoint

**OnStation Phase for Non-Compliant Boarding**

- The OnStation Phase begins when the swarm reaches the assigned OnStation area
- Swarm Commander monitors OnStation activities
- Swarm Monitor monitors swarm fuel status (ongoing)
- Swarm mission tasks:
  - based on COI location data from the P-8A, surveil area for COI
  - collect RF transmissions, evidence of small arms threat, smuggling evidence
  - collect data on COI infrastructure and count number of personnel on board using IR, EO, synthetic aperture radar and LIDAR
  - transmit real-time video, images, and electronic surveillance data of the COI to FORCENET and boarding team RHIB
  - Perform communication relay between boarding team and MIOC and FORCENET
- Swarm uses plays from the MASC playbook to support the mission *(select plays)*

- Swarm Commander intercepts radio communication and sees personnel movement on video that indicates a non-compliant posture from COI
- Personnel on COI begin shooting at swarm *(does this prompt a change in plays?)*
- P-8A and MH-60R continue providing maritime domain awareness by providing Link-16 tracks and identifying them with on-board sensors.
- National intelligence assets provide intelligence updates to MIOC
- Boarding team boards COI and COI does not obey boarding team instructions
- Boarding team conducts search, finds contraband, and reports findings to MIOC
- MIOC notifies Indonesian law enforcement.
- Indonesian law enforcement arrives to detain COI, boarding team disembarks and transits back to DDG, MIOC commands swarm to return to ship
- The OnStation Phase ends when the MIOC has commanded the swarm to return to ship.

**Egress Phase**

- The Egress Phase begins when the OnStation support requirements have been met and the swarm is ready to return to the ship *(select plays)*
- Swarm transits to the terminal way point and executes landing sequence
- Swarm Monitor monitors system health
- The Egress Phase ends when the swarm has landed or has been captured in a recovery system onboard the DDG

**Postflight Phase**

- The Postflight Phase begins once the swarm has landed or been captured
- Swarm Commander, Swarm Monitor, and Ground Crew perform post-flight procedures and inspections on UAVs
- Swarm Commander and Swarm Monitor perform post-flight procedures and inspections on GCS
- Ground Crew perform post-flight procedures and inspections on launcher and recovery system (as applicable)
- Swarm Commander, Swarm Monitor, and Ground Crew debrief mission
- Swarm Commander, Swarm Monitor, and Ground Crew generate after action report
- The Postflight Phase ends once the mission after action report has been completed

### C.3.2 Description of Plays

Given the MIO scenario you have just read, build a swarm mission plan to support the mission using the plays described below and the template provided (C.5). The plays control the behavior of the swarm in terms of sensor operation, maneuver, and weapon control. Your mission plan should consist of a sequence of plays in an activity model.

- **Launch ($p_1$)** is the play used for transitioning the UAVs from the airfield or ship into the airborne environment. The operator modifiable parameters are: number of UAVs to be launched, number of launchers, and the time interval in seconds between UAV launches. The time interval can be set to a minimum threshold to perform a rapid launch. Launch is part of the Ingress tactic.
- **Transit to waypoint ($p_2$)** can be used as a single play or in combination to perform various flight paths and patterns. It is used in the Ingress, Egress, Deter, Harass, and Defend tactics. Operator modifiable parameters include: transit altitude or altitude block (for evasive transits), and waypoint latitude/longitude.
- **Orbit ($p_3$)** is a play that enables the swarm to circle around a fixed geographic position or target. It is used in the Monitor, Harass, and Deter tactics. The operator modifiable parameters are: altitude or altitude block (to stack the UAVs at multiple altitudes), clockwise or counterclockwise flow, designated center waypoint latitude/longitude, and radius of orbit (in m or nm).
- **Racetrack ($p_4$)** is a play that enables the swarm to encircle a fixed geographic area or target in a racetrack pattern. It is used in the Monitor, Defend, and Deter tactics. The operator modifiable parameters are: search sensor coverage parameters, or altitude or altitude block (to stack the UAVs at multiple altitudes), clockwise or counterclockwise flow, designated center waypoint latitude/longitude, width, and length (in m or nm).
- **Split (logic-based) ($p_5$)** splits the swarm into sub-swarms based on parameters: health status, battery life remaining, number of sub-swarms, location, or proximity to a reference waypoint. This play supports the Divide tactic.
- **Join ($p_6$)** is used to return UAVs to their original swarm or sub-swarm after being split or dispersed. It is used in the Evade, Harass, and Amass tactics. The operator modifiable parameters are designated join time, specified joining waypoint latitude/longitude, and altitude block.
- **Disperse ($p_7$)** is a defensive play to scatter and separate the UAVs within a cylinder,

to support the Evade, Harass, and Amass tactics. The operator modifiable parameters are dispersion radius and altitude block.

- **Sensors ON ($\mathbf{p_{8.1}}$)** is the default sensor setting that turns on all available non-navigational sensors. Sensors are turned on after the UAV has been launched and has executed a positive rate of climb. Several OnStation tactics including both search tactics, Track, and Monitor begin with Sensors ON. The operator modifiable parameters are the individual ON or OFF settings for each individual sensor (i.e., radar, EO, IR, LIDAR, electronic surveillance measures).
- **Sensors OFF ($\mathbf{p_{8.2}}$)** is the default sensor setting that turns off all non-navigational sensors. Sensors (i.e., radar, EO, IR, LIDAR, electronic surveillance measures) are turned off just prior to landing, as part of the Egress tactic to prevent damage to sensors and injury to personnel.
- **Sensors EMCON ($\mathbf{p_{8.3}}$)** is the sensor setting used for evasive tactics. It turns the active sensors, such as radar and LIDAR, off and leaves the passive sensors (i.e., EO, IR) on. Sensors EMCON is part of the Monitor and Evade tactics.
- **Transmit video ($\mathbf{p_{8.4}}$)** is the play that enables swarm video to be transmitted to the receivers other than GCS, and it is part of the Communication relay tactic. The operator modifiable parameters are the receiving nodes' addresses and sensor data to be transmitted.
- **Terminal approach ($\mathbf{p_{9.1}}$)** sequences the swarm for landing according to the individual UAV's battery life and proximity to the landing waypoint. Operator modifiable parameters allow for a single terminal approach path or multiple terminal approach paths, normal glideslope or combat descent rate, and inbound headings. Terminal approach is part of the Egress tactic and it is used at the end of each mission.
- **Landing ($\mathbf{p_{9.2}}$)** defines the landing area for the swarm. User modifiable parameters are available for specifying the dimensions of landing area: radius of a circle or m x n dimensions of a rectangle, or a specified separation distance between UAVs other than the default. Terminal approach is part of the Egress tactic and it is used at the end of each mission.
- **Ladder pattern ($\mathbf{p_{10}}$)** is a pattern play used in the Search and Communication relay tactics. The pattern commences at a known datum and executes a long track leg, followed by 90° turn, a short leg, another 90° turn in the same direction, a long track leg of the same length as the previous long leg, a 90° turn in the opposite direction

from previous turn, a short leg of the same length, etc. The track leg lengths are dependent on the swarm size, sensor range, and the operator modifiable parameters: desired probability of detection, search area size, and altitude block.

- **Expanding square pattern ($\mathbf{p_{11}}$)** is commonly used in SAR missions. The pattern starts at a known datum, then transits to the first waypoint for a distance *d*, makes a 90° turn (all turns are in same direction), transits to next waypoint at a distance *d*, makes a 90° turn for a distance *2d*, makes another 90° turn, transits *2d* to next waypoint, performs a 90° turn, then transits *3d* to next waypoint, etc. The distance *d* is dependent on timeliness of the datum and the best known speed of target, and consideration of environmental conditions such as wind and current. The Efficient search, Communication relay, and BDA tactics all have the Expanding square pattern play option. The operator modifiable parameters are altitude block, datum latitude/longitude, and estimated speed of the target.
- **Constricting square pattern ($\mathbf{p_{12}}$)** supports the Efficient search and Communication relay tactics. Beginning from an offset distance *d*, it is performed in an inverse fashion of the Expanding square pattern, moving inward toward the last known position. *d* is dependent on timeliness of the datum and the best known speed of target, and consideration of environmental conditions such as wind and current. The operator modifiable parameters are altitude block, datum latitude/longitude, and estimated speed of the target.
- **Grid pattern ($\mathbf{p_{13}}$)** supports the Efficient search and Communication relay tactics. Grid pattern defines an operating area in terms of large rectangle broken into smaller rectangles. A lead UAV is collectively selected and acts as master UAV by allocating tasks and grid assignments to the other UAVs. Each UAV searches its assigned rectangle and reports back to the assigned lead, using communication relay via the other UAVs as necessary. The operator modifiable parameters are search area dimensions, desired area coverage, and altitude block.
- **Random pattern ($\mathbf{p_{14}}$)** maneuvers the swarm together in a random pattern by transiting to randomly generated waypoints within an operating area. Random pattern supports the Evasive transit tactic and its operator modifiable parameters are altitude block and search area dimensions.
- **Weapon armed ($\mathbf{p_{15}}$)** enables weapon arming when the swarm has reached a swarm ready state and if the Attack tactic is enabled for the mission. Availability of this play

is ROE dependent and loaded during mission planning.

- **Weapon fire ($p_{16}$)** enables the available weapon(s) to be fired. The operator modifiable parameters are weapon selection and manual or automatic mode. Automatic mode is designed for specific scenarios such as close-in ship support or a SEAD mission, and is ROE dependent. Manual mode is the normal mode of operation and requires the operator to manually initiate the weapon after a target has been identified. Weapon fire is part of the attack tactic.
- **Follow target ($p_{17}$)** is used to follow and track a ground target and is part of the Track tactic. Operator modifiable parameters are offset slant range and altitude block.
- **Forward communication ($p_{18}$)** is part of the Communication Relay tactic and it forwards voice or data link communication received from another unit to another unit.
- **Jam ($p_{19}$)** uses various forms of electronic jamming and other electronic deception techniques to interfere with an enemy's radar and other C2 systems [159]. It is part of the Evade and Harass tactics. The operator modifiable parameters are frequency bands and temporal schedule.
- **Smart Greedy Shooter ($p_{20}$)** was developed by ARSENL and improves on the original ARSENL greedy shooter by coordinating target allocation to ensure each adversary is engaged by no more than one friendly UAV. Each UAV will engage the adversary determined to be optimal via shortest distance to adversary [160]. This play supports the Attack tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.
- **Patrol Box Shooter ($p_{21}$)** was developed as an ARSENL behavior and provides area-defense that incorporates smart shooter selection semantics in randomized patrol of defended area. UAVs individually determine patrol patterns within defended area without coordination or notification [160]. This play supports the Attack tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.
- **Wingman Shooter ($p_{22}$)** was developed as an ARSENL behavior and coordinates target allocation to ensure each adversary is engaged by two friendly UAVs. The pair will engage the adversary determined to be optimal via shortest distance to adversary [160]. This play supports the Attack tactic engagement modes. Operator modifiable parameters include geographic boundaries and altitude blocks.

- **Tail following ($\mathbf{p_{23}}$)** used to follow other aircraft, friend or foe. The operator modifiable parameters are altitude block, offset range (m), knock-it-off criteria (i.e., range from home or geographic boundary, or surrounded by x bandits).

### C.3.3 Group 2 Participant Assignment

Figure C.5 is a picture of the template you will use online in Draw.io to assemble your MIO mission plan using a combination of the plays discussed above. The template shows the general mission flow by phase on the left and event triggers from the mission narrative you just read in the middle. Your task is to place the orange plays in the middle section in response to the event triggers. Place the plays in temporal sequential order from left to right, and stack concurrent plays (for sub-swarms) on top of each other. If there is a play you would like to use that is not included in the playbook, use the "option" play and describe its purpose and function in the survey. Begin assembling the mission plan when you are able to complete the task in one sitting and record the start and finish times for the task.

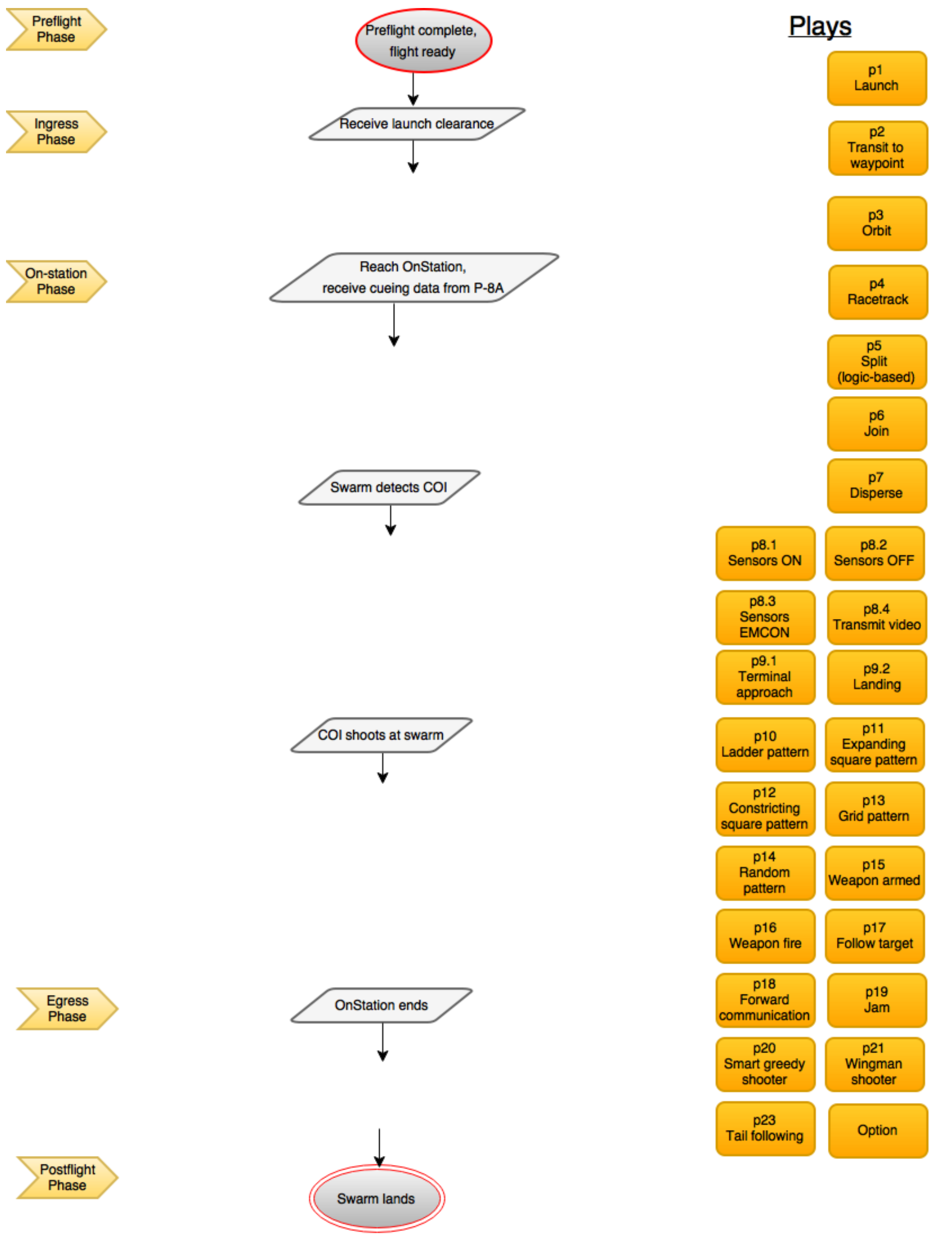



Figure C.5. Scenario Template for Group 2

### C.3.4 Stakeholder Feedback Survey

1. How long did it take you to complete the swarm mission plan?

2. What is your military rank?

3. What is your military occupational specialty?

4. How many years of military operational experience do you have?

5. What is your experience with commanding or participating in UAV swarm missions?

6. What is your experience with commanding or participating in single UAV missions?

7. Does the structure of the mission phases support the execution of this mission? If not, how could it be improved?

8. How could this playbook of plays be improved to better support this mission scenario?

9. Would pre-defined combinations of plays make the task easier?

The following questions are adopted from the NASA Task Load Index method [167]. Please answer the remaining questions based on the following rating scale:

1 = very low     2 = low     3 = medium     4 = high     5 = very high

10. How mentally demanding was the task?

11. How hurried or rushed was the pace of the task?

12. How successful were you in accomplishing what you were asked to do?

13. How hard did you have to work to accomplish your level of performance?

# APPENDIX D:
# MASC Tactics Diagrams

This appendix covers the activity diagrams of the tactics as composed of plays in Innoslate. These tactics were used in the three case study mission simulations built using Innoslate. These diagrams are referenced and described in Chapters 3 and 4.

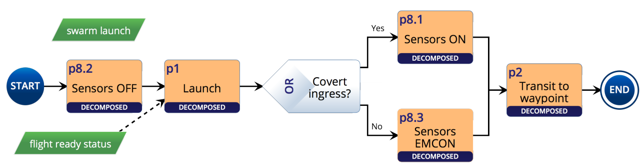



Figure D.1. Ingress Tactic ($t_1$)

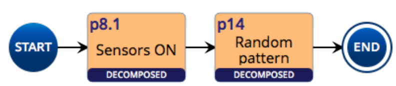



Figure D.2. Evasive Search Tactic ($t_2$)

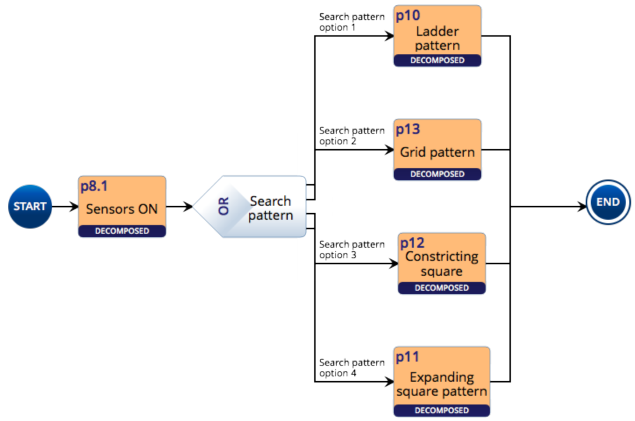



Figure D.3. Efficient Search Tactic ($t_3$)

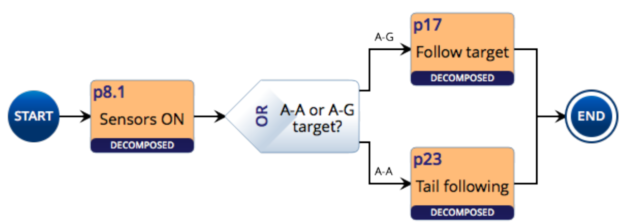



Figure D.4. Track Tactic ($t_4$)

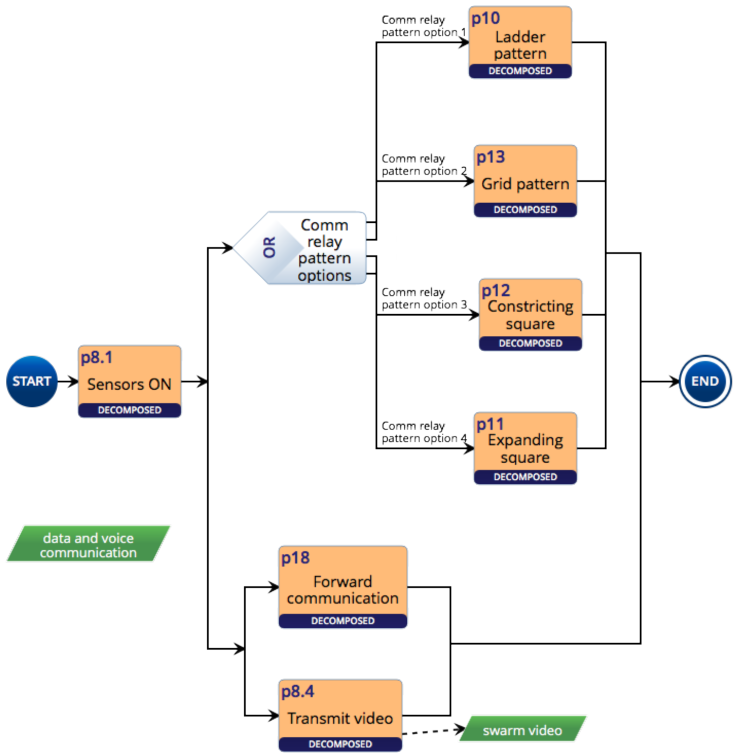



Figure D.5. Communication Relay Tactic ($t_5$)

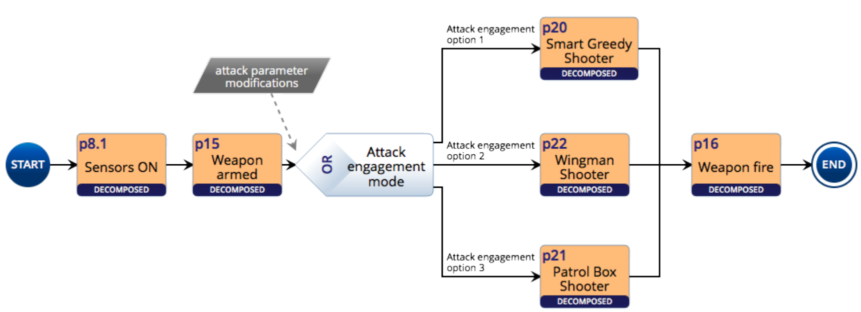



Figure D.6. Attack Tactic ($t_6$)

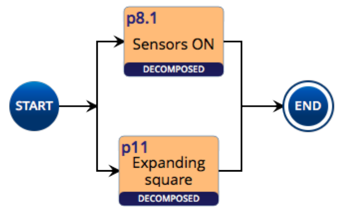



Figure D.7. Battle Damage Assessment Tactic ($t_7$)

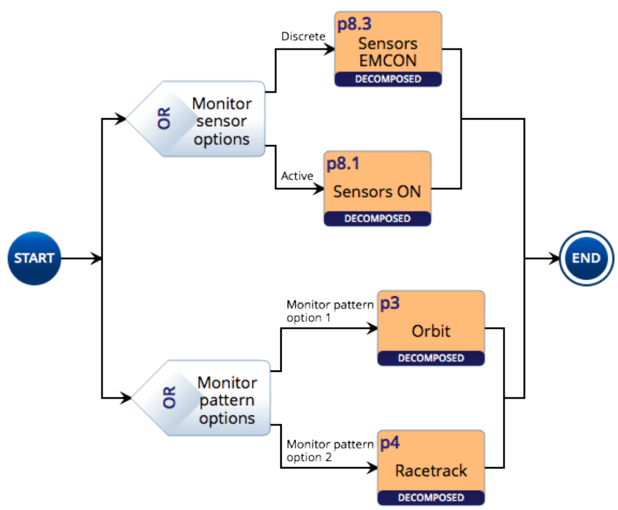



Figure D.8. Monitor Tactic ($t_8$)

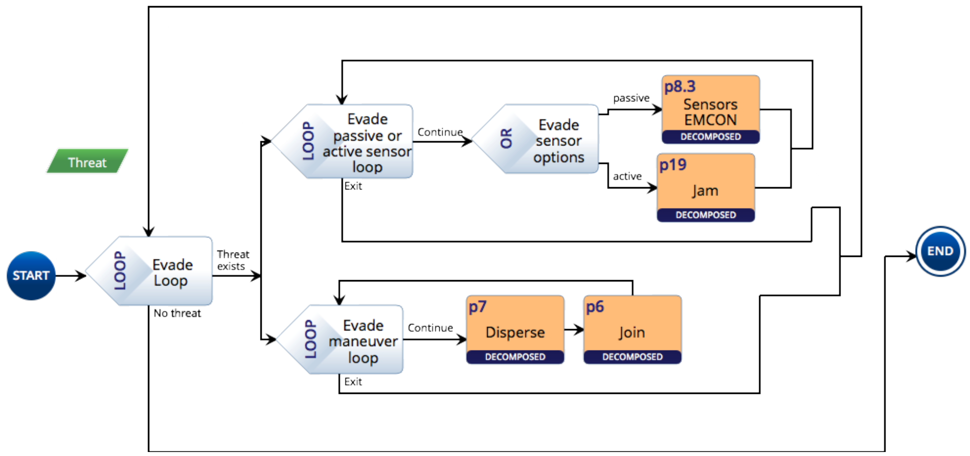



Figure D.9. Evade Tactic ($t_9$)

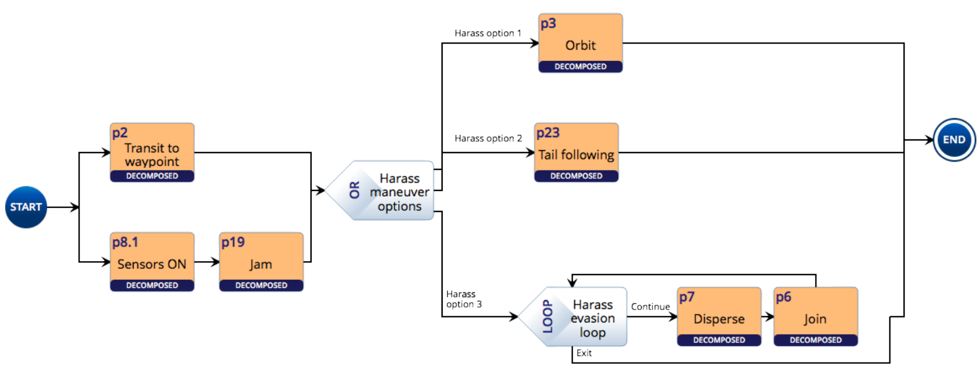



Figure D.10. Harass Tactic ($t_{10}$)

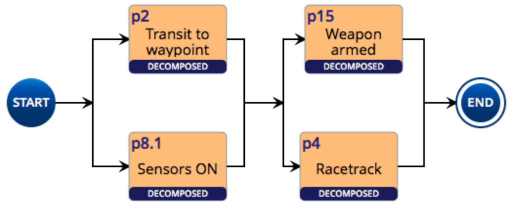



Figure D.11. Defend Tactic ($t_{12}$)

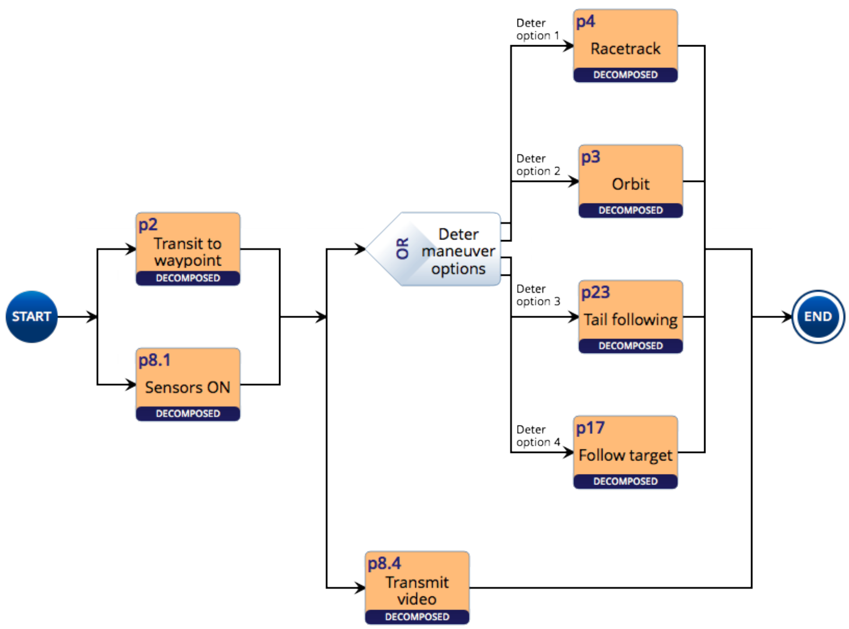



Figure D.12. Deter Tactic ($t_{12}$)

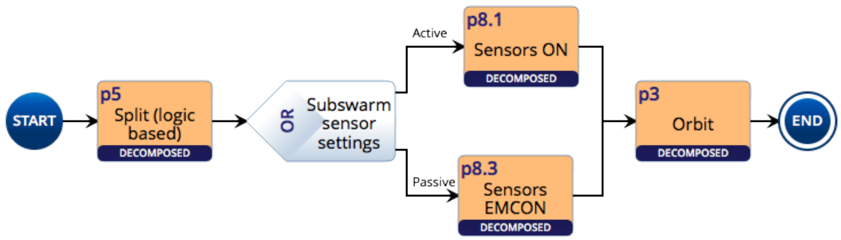



Figure D.13. Divide Tactic ($t_{13}$)

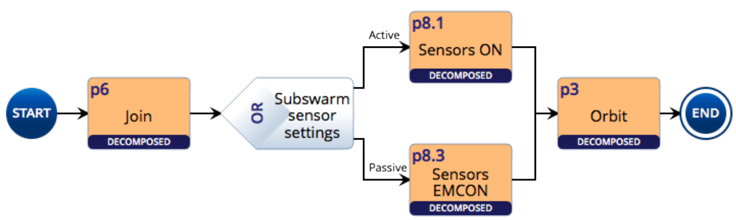



Figure D.14. Amass Tactic ($t_{14}$)

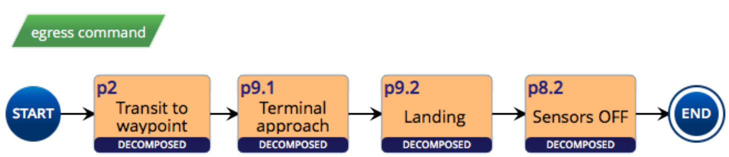



Figure D.15. Egress Tactic ($t_{15}$)

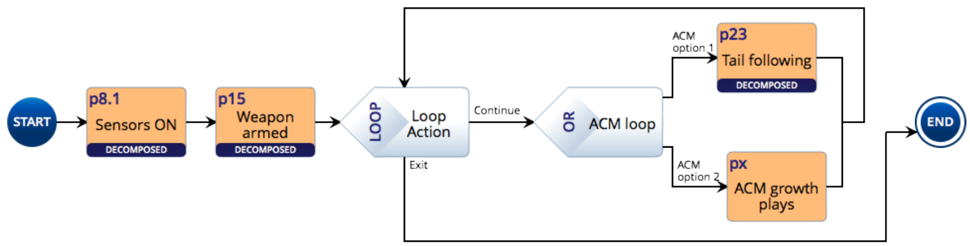



Figure D.16. Air Combat Maneuvering Tactic ($t_{16}$)

# APPENDIX E:
# MASC Mission Diagrams

This appendix covers the mission diagrams for each case study, developed using Innoslate. These diagrams were used in the three mission simulations built using Innoslate, and are referenced and described in Chapter 4.


Figure E.1. SvS Mission Overall Phases

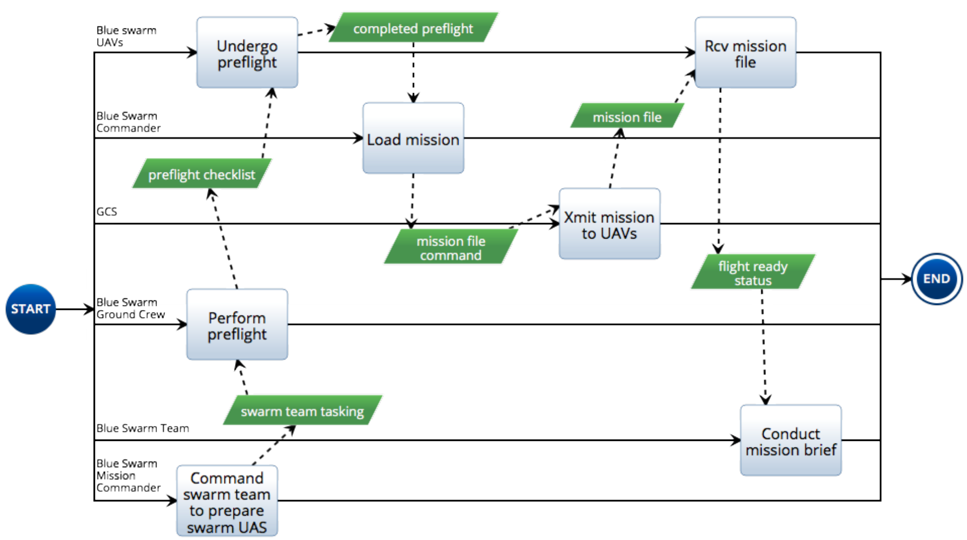



Figure E.2. SvS Mission Preflight Phase (P.1.SVS)

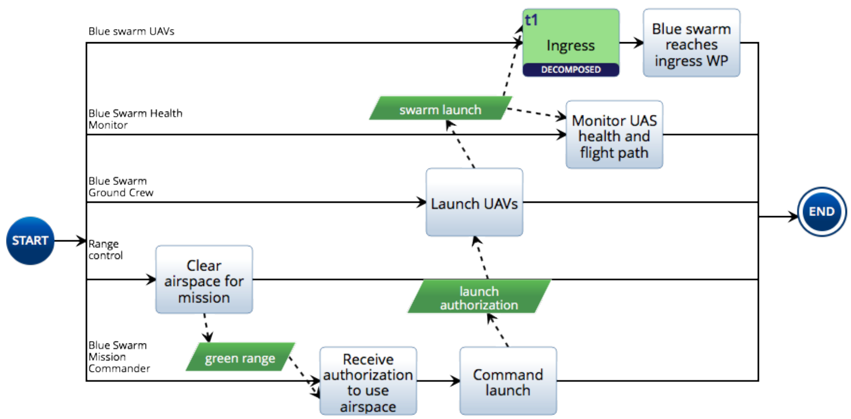



Figure E.3. SvS Mission Ingress Phase (P.2.SVS)

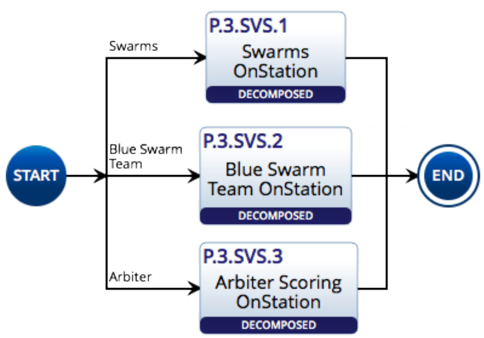



Figure E.4. SvS Mission OnStation Phase (P.3.SVS)

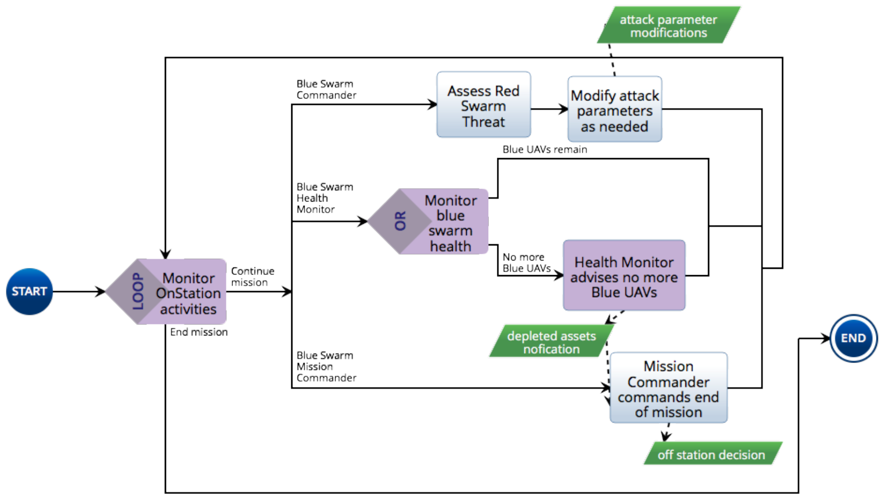



Figure E.5. SvS Mission OnStation Phase Blue Swarm Team Activities (P.3.SVS.2)

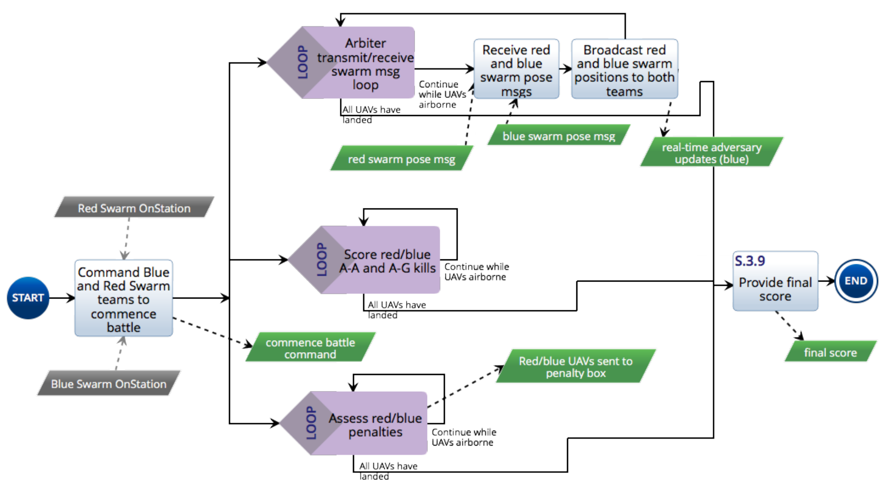



Figure E.6. SvS Mission OnStation Phase Arbiter Activities (P.3.SVS.3)

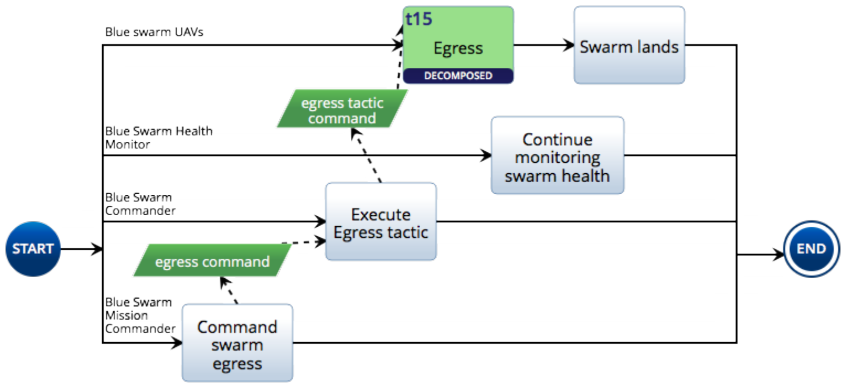



Figure E.7. SvS Mission Egress Phase (P.4.SVS)

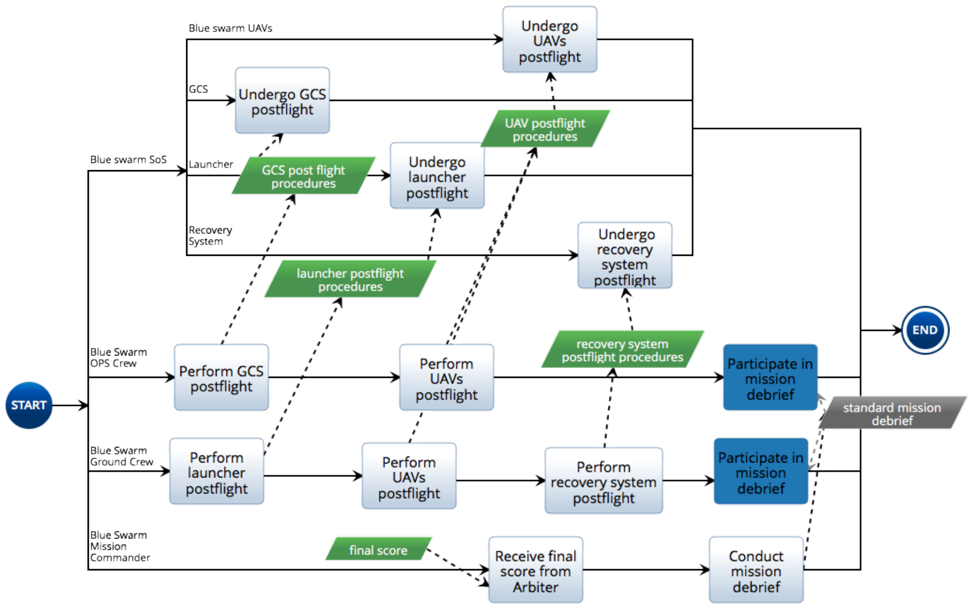



Figure E.8. SvS Mission Postflight Phase (P.5.SVS)


Figure E.9. MIO Mission Overall Phases

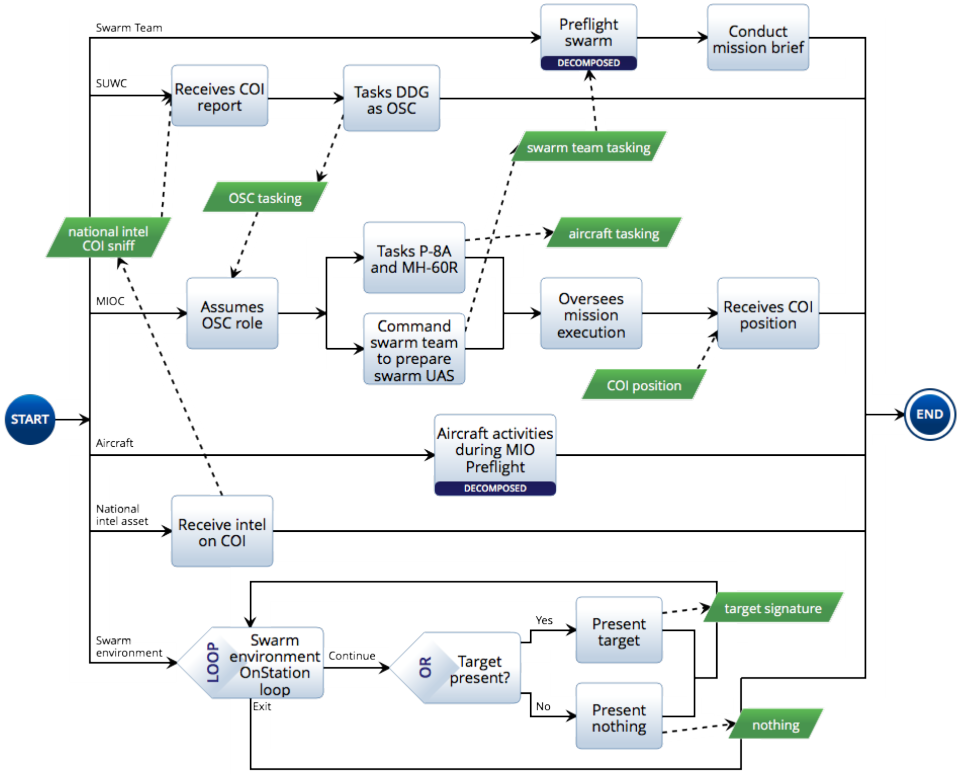



Figure E.10. MIO Mission Preflight Phase (P.1.MIO)

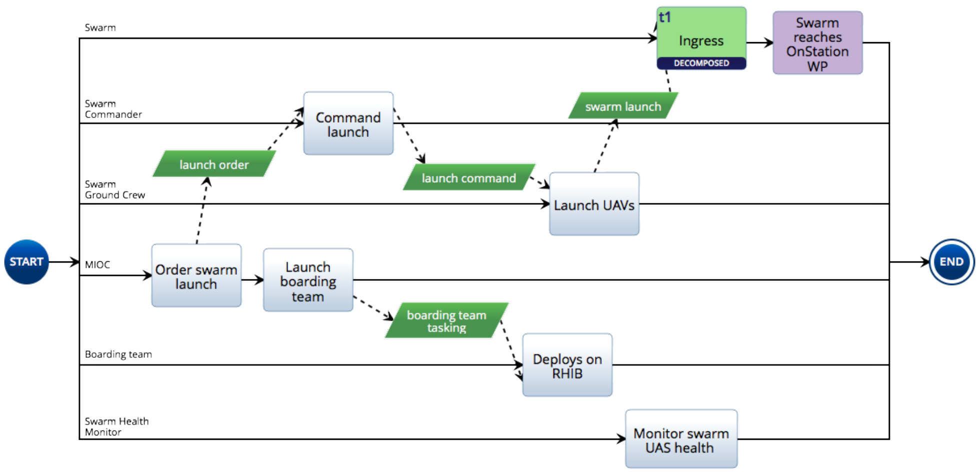



Figure E.11. MIO Mission Ingress Phase (P.2.MIO)

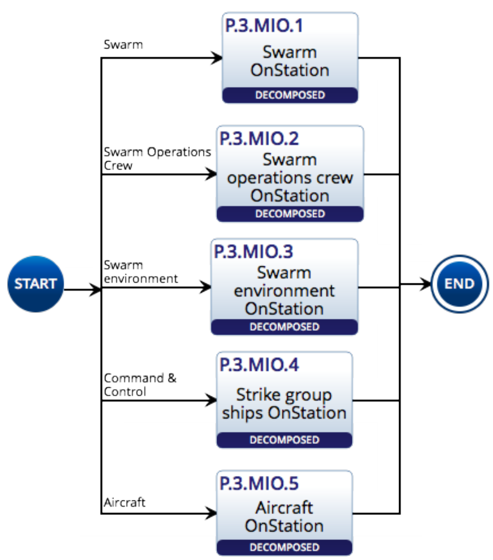



Figure E.12. MIO OnStation Phase (P.3.MIO)

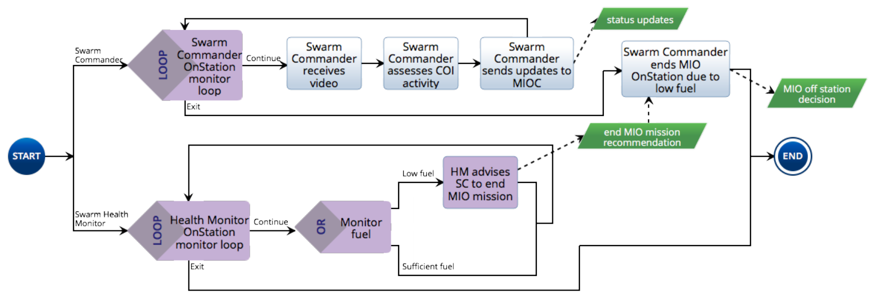



Figure E.13. MIO OnStation Phase Swarm Operations Crew Activities (P.3.MIO.2)

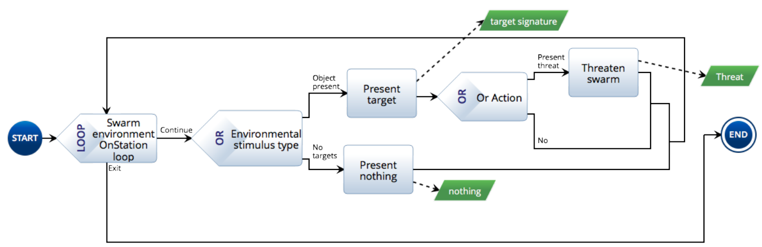



Figure E.14. MIO OnStation Phase Environment Activities (P.3.MIO.3)

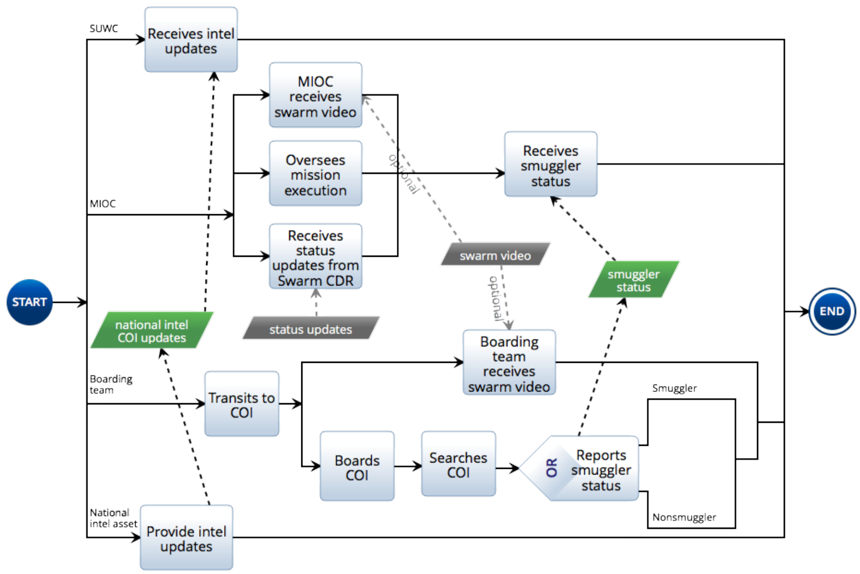



Figure E.15. MIO OnStation Phase Strike Group and Ship Activities (P.3.MIO.4)

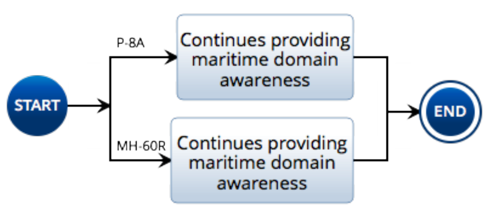



Figure E.16. MIO OnStation Phase Supporting Aircraft Activities (P.3.MIO.5)

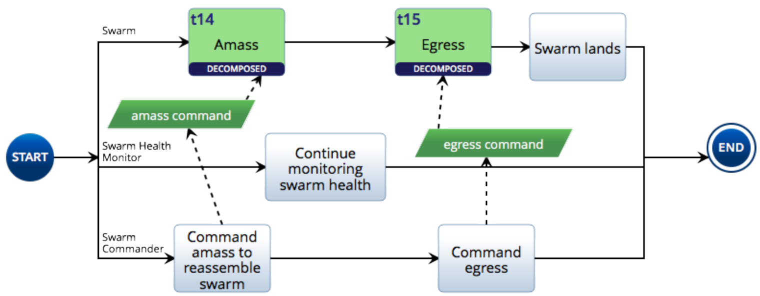



Figure E.17. MIO Egress Phase (P.4.MIO)

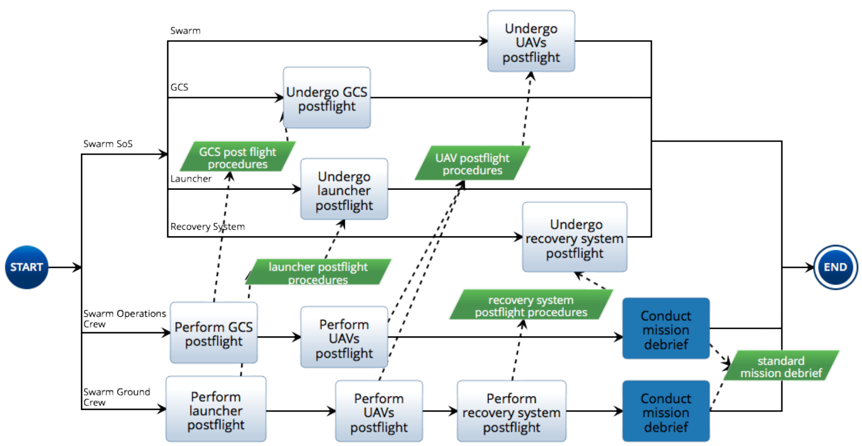



Figure E.18. MIO Postflight Phase (P.5.MIO)


Figure E.19. HADR Mission Overall Phases

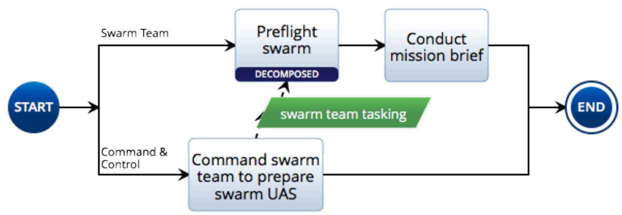



Figure E.20. HADR Mission Preflight Phase (P.1.HADR)

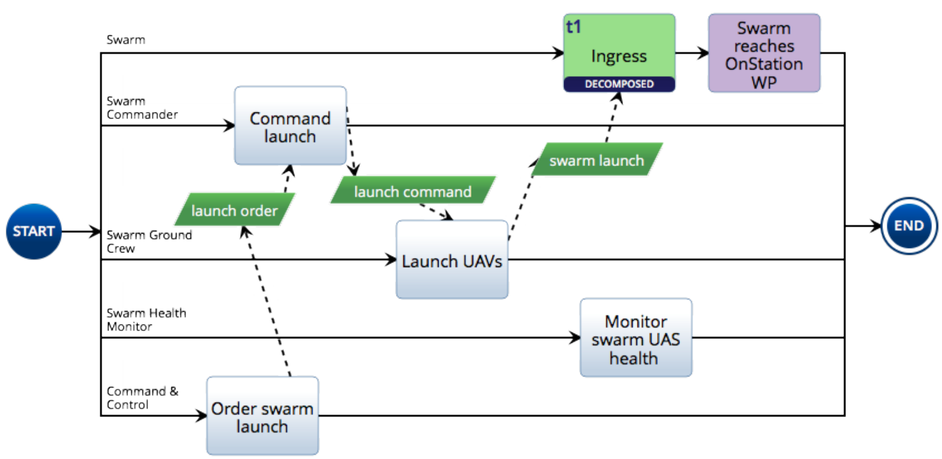



Figure E.21. HADR Mission Ingress Phase (P.2.HADR)

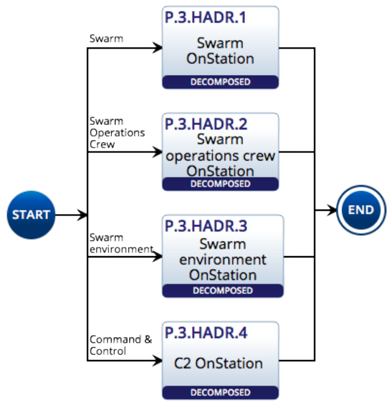



Figure E.22. HADR OnStation Phase (P.3.HADR)

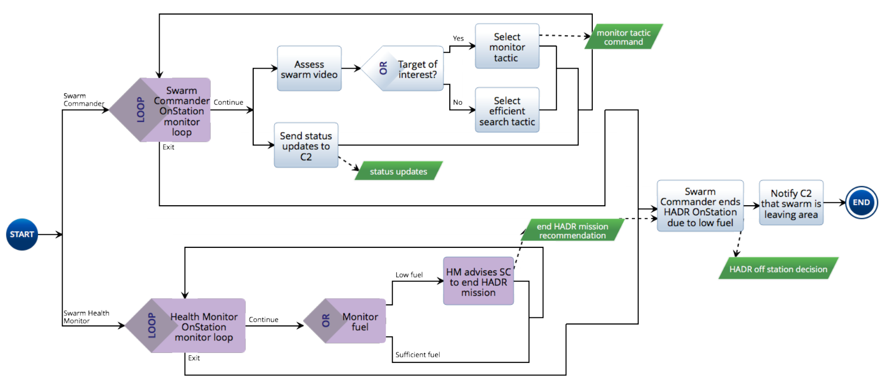



Figure E.23. HADR OnStation Phase Swarm Operations Crew Activities (P.3.HADR.2)

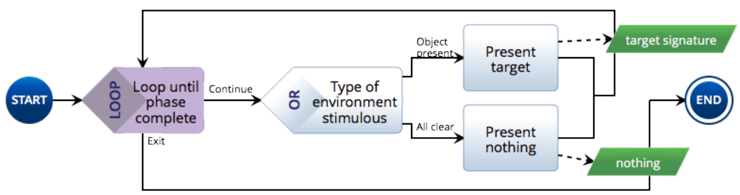



Figure E.24. HADR OnStation Phase Environment Activities (P.3.HADR.3)

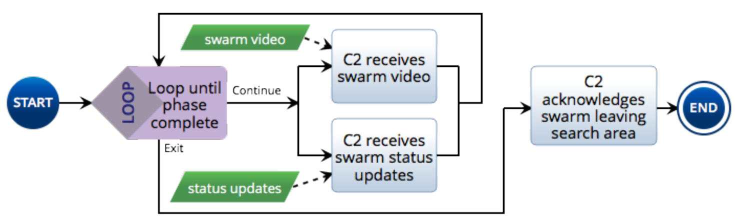



Figure E.25. HADR OnStation Phase Command and Control Activities (P.3.HADR.4)

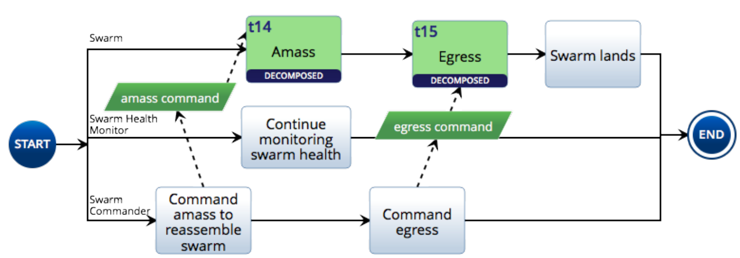



Figure E.26. HADR Egress Phase (P.4.HADR)

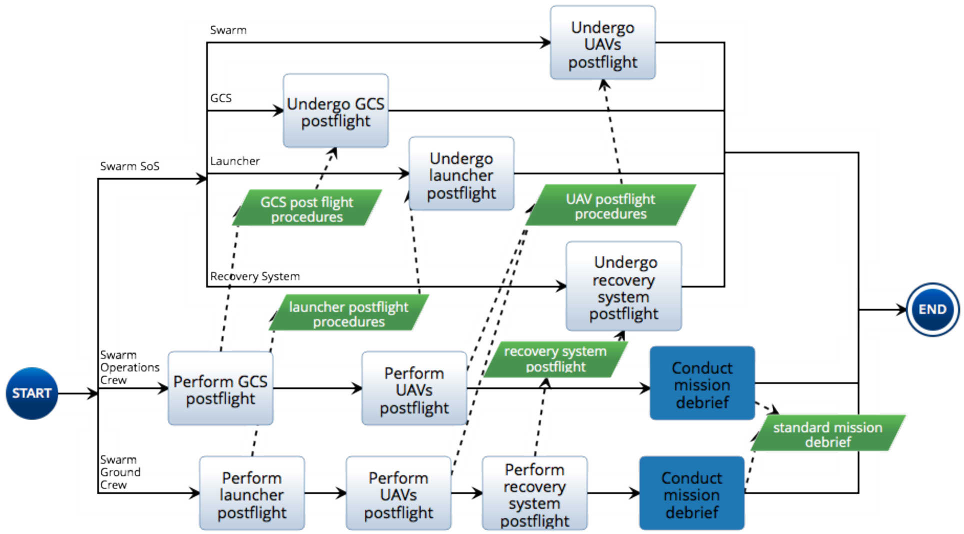



Figure E.27. HADR Postflight Phase (P.5.HADR)

# APPENDIX F: Monterey Phoenix Model Code

This appendix covers the MP code used in the mission diagrams that are referenced and described in Chapter 4.

```
SCHEMA Swarm_V_Swarm_Mission

/* ------------------- MAJOR ACTORS -------------------------------*/

ROOT Arbiter:                /* Receives all UAV positions, re-broadcasts
                                them for common op picture, and scores mission */
    {* Receive_UAV_Positions *}
    Broadcast_UAV_Positions
    Score_Mission;


ROOT Blue_MC:                   /*Blue Swarm Mission Commander  */
    Conduct_Mission_Brief
    Command_Commence_Mission
    (* Monitor_Mission_Tactics *)
    Conduct_Mission_Debrief;

ROOT Blue_LO:                   /*Blue Swarm Launch Operator  */
    Preflight_Blue_Swarm
    Conduct_Mission_Brief
    Launch_Blue_Swarm
    Recover_Blue_Swarm
    Conduct_Mission_Debrief;

ROOT Blue_SO:                   /*Blue Swarm Operator  */
    Conduct_Mission_Brief
    Command_Search_and_Track_Tactics
    Command_Blue_Attack_Tactic
    Command_Egress_Tactic
    Conduct_Mission_Debrief;

ROOT Blue_LT:                   /*Blue Swarm Launch Tech */
    {* Preflight_Blue_Swarm *}
    Conduct_Mission_Brief
    Recover_Blue_Swarm
    Postflight_Blue_Swarm
    Conduct_Mission_Debrief;


ROOT Blue_SP:                   /*Blue Swarm Safety Pilot  */
    Conduct_Mission_Brief
    (* Assess_Flight_Behavior
        ( Behavior_is_Normal |
          Behavior_is_Abnormal Manually_Fly_Errant_UAV) *)
    Conduct_Mission_Debrief;

ROOT Range_Control:
    Give_Green_Range_Command
    (* Receive_UAV_Recovered_Notification *);

ROOT Blue_SM:                   /*Blue Swarm Monitor  */
    Conduct_Mission_Brief
    Command_Ingress_Tactic
    Monitor_Mission_Health
    Command_Egress_Tactic
    Conduct_Mission_Debrief;


ROOT Environment: (* Present_Environment_Signature *);

ROOT Blue_Swarm: /*  Blue Swarm is composed of multiple UAVs  */
    {+ UAV +};

 UAV:   Undergo_PreFlight
        Launch
        Detect_Environment_Signature
        Send_UAV_Positions
        Receive_Red_UAV_Positions
        Ingress_Tactic
        Intercept_Swarm_Ready_WP
        Report_Health
        Search_and_Track_Tactics
        Attack_Tactic
        [Evade_Tactic]
        Egress_Tactic
        Land;

/*-------- MAJOR PHASES OF MISSION --------------*/

ROOT Preflight:
    Conduct_Mission_Brief
    Check_Configuration
    Set_Parameters
    Engine_Runup;

ROOT Inflight:
    Launch_Blue_Swarm
    Ingress_Tactic
    Search_and_Track_Tactics
    Attack_Tactic
    Score_Mission
    Egress_Tactic;

ROOT Postflight:
    Recover_Blue_Swarm
    Conduct_Mission_Debrief;

/*----------OVERLAP OF ACTORS AND MISSION PHASES--------*/

Blue_MC, Blue_LT, Blue_LO, Blue_SO, Blue_SP, Blue_SM, Preflight
        SHARE ALL Conduct_Mission_Brief;

Blue_LO, Inflight  SHARE ALL Launch_Blue_Swarm;

Blue_Swarm, Inflight SHARE ALL Ingress_Tactic,
        Search_and_Track_Tactics, Attack_Tactic, Egress_Tactic;

Arbiter, Inflight SHARE ALL Score_Mission;

Blue_LO, Blue_LT, Postflight SHARE ALL Recover_Blue_Swarm;

Blue_MC, Blue_LT, Blue_LO, Blue_SO, Blue_SP, Blue_SM, Postflight
        SHARE ALL Conduct_Mission_Debrief;

/*-------------COORDINATION STATEMENTS-----------------*/

COORDINATE  <!> $a: Preflight_Blue_Swarm         FROM Blue_LT,
            <!> $b: Undergo_PreFlight            FROM Blue_Swarm
        DO ADD $a PRECEDES $b; OD;


COORDINATE  $a: Give_Green_Range_Command         FROM Range_Control,
            $b: Command_Commence_Mission         FROM Blue_MC
        DO ADD $a PRECEDES $b; OD;

COORDINATE <!>  $a: Send_UAV_Positions           FROM Blue_Swarm,
           <!>  $b: Receive_UAV_Positions        FROM Arbiter
        DO ADD  $a PRECEDES $b; OD;


/* need one-many coordination here */
COORDINATE  $a: Launch_Blue_Swarm                FROM Blue_LO
        DO COORDINATE   <!>  $b: Launch          FROM Blue_Swarm
        DO ADD $a PRECEDES $b; OD;
    OD;

/* need one-many coordination here */

COORDINATE  $a: Command_Ingress_Tactic                   FROM Blue_SM
        DO COORDINATE   <!> $b: Ingress_Tactic           FROM Blue_Swarm
        DO ADD $a PRECEDES $b; OD;
    OD;

COORDINATE  $a: Command_Egress_Tactic                    FROM Blue_SM
            DO COORDINATE   <!> $b: Egress_Tactic        FROM Blue_Swarm
        DO ADD $a PRECEDES $b; OD;
    OD;

COORDINATE  $a: Land                                     FROM Blue_Swarm
            DO COORDINATE $b: Recover_Blue_Swarm         FROM Blue_LT
        DO ADD $a PRECEDES $b; OD;
        OD;

COORDINATE <!>  $a: Report_Health                                FROM Blue_Swarm
                DO COORDINATE  <!>  $b: Monitor_Mission_Health   FROM Blue_SM
        DO ADD $a PRECEDES $b; OD;
        OD;

COORDINATE <!>  $a: Present_Environment_Signature            FROM Environment
        DO COORDINATE  <!>  $b: Detect_Environment_Signature FROM Blue_Swarm
        DO ADD $a PRECEDES $b; OD;
        OD;
```

Figure F.1. SvS Overall Mission MP Code

Run   Scope: 2

```
/*  Swarm vs. Swarm air battle only 7MAY2016
(with help from Mikhail Auguston)
revised 7JUL2017 */

/* run for scope 1 or 2, there still will be two Swarms*/

SCHEMA BlueSwarm_vs_RedSwarm

/* Swarm has at least one UAV initially */
Swarm: {+ UAV +};

/* UAV behavior is constrained for at most one shot
    We assume that if UAV went in action,
    it makes at least one move.
    Mutual UAV destruction is possible.
    UAVs make AA (air-to-air) attacks or
    AG (air-to-ground) attacks.
*/
UAV: Attack ( Shoot_enemy_UAV | Attack_enemy_base)
    (killed | returns_to_base);

Shoot_enemy_UAV: (valid_AA_hit | miss);

Attack_enemy_base: (valid_AG_hit | no_hit);

/* Blue and Red contain a single Swarm,
so make them sets of Swarms
*/
ROOT Blue: Swarm;

ROOT Red: Swarm;

/*  coordinate shooters and targets.
<!> is needed because Swarm is not ordered,
    it is a set */
COORDINATE <!> $h: valid_AA_hit FROM Blue,
<!> $d: killed FROM Red
DO ADD $h PRECEDES $d; OD;

COORDINATE <!> $h: valid_AA_hit FROM Red,
<!> $d: killed FROM Blue
DO ADD $h PRECEDES $d; OD;

/* calculate points and provide statistics
    assuming 5 points for each destroyed
    enemy UAV and 1 point for base hit
*/
  SAY("Blue Swarm has " 5 * #valid_AA_hit FROM Blue
    " points for destroyed UAVs and "
    #valid_AG_hit FROM Blue
    " for enemy base hits");

    /*  MARKing traces with special features */
IF #UAV FROM Blue == #killed FROM Blue THEN
SAY("Blue Swarm was completely destroyed");
MARK;
    FI;

  SAY("Red Swarm has " 5 * #valid_AA_hit FROM Red
    " points for destroyed UAVs and "
    #valid_AG_hit FROM Red
    " for enemy base hits");

    /*  MARKing traces with special features */
IF #UAV FROM Red == #killed FROM Red THEN
SAY("Red Swarm was completely destroyed");
MARK;
    FI;
```

Figure F.2. Blue Swarm versus Red Swarm MP Code

Run   Scope: 2

```
/* K. Giles swarm vs. swarm model with roots blue swarm,
red swarm, and arbiter.
original: 2MAY2017 revised: 17JUL17
-assume no errors, removed swarm commander to
focus on automated tactics
- new approach using tactics, not states
- need to add BUILD ENSURE line to ensure all red UAVs
killed before blue swarm Egress tactic is triggered

/*------------------------------
            Actors
--------------------------------*/
SCHEMA SvSOnstation


ROOT Arbiter:          Commence_mission
                       Score_attacks
                       End_mission ;

ROOT Red_Swarm:    {+ Red_UAV +};

    Red_UAV:           (+ Present_signature +)
                       (+ ( Detected_by_UAV Incur_attack |
                            Not_detected ) +)
                       End_mission;

        Incur_attack: ( Hit | Miss)  ;


ROOT Blue_Swarm:     {+ Blue_UAV +};

    Blue_UAV:              Evasive_search_tactic
                        (*  (  Track_tactic
                               Attack_tactic ) *)
                               Egress_tactic ;

/* ------------------------------------
  Tactics are composed of plays
----------------------------------------*/

/*      Evasive_search_tactic:      Sensors_ON
                                    Random_pattern;

        Track_tactic:   Sensors_ON
                        Follow_target;


        Attack_tactic:   Sensors_ON
                         Weapon_armed
                             (  Smart_greedy_shooter |
                                Wingman_shooter      |
                                Patrol_box_shooter )
                         Weapon_fire;

        Egress_tactic:      Transit_to_WP
                            Terminal_approach
                            Landing
                            Sensors_OFF;
/*------------------------------
            Interactions
--------------------------------*/

COORDINATE  $a: Commence_mission          FROM Arbiter,
            $b: Evasive_search_tactic     FROM Blue_Swarm
    DO ADD $a PRECEDES $b;  OD;


COORDINATE <!> $a: Detected_by_UAV        FROM Red_Swarm,
           <!> $b: Track_tactic           FROM Blue_Swarm
    DO ADD $a PRECEDES $b; OD;


COORDINATE <!> $a: Attack_tactic          FROM Blue_Swarm,
           <!> $b: Incur_attack           FROM Red_Swarm
    DO ADD $a PRECEDES $b; OD;


COORDINATE  $a: Hit                       FROM Red_Swarm,
            $b: Egress_tactic             FROM Blue_Swarm
    DO ADD $a PRECEDES $b; OD;

/*BUILD {
    ENSURE #Red_UAV FROM Red_Swarm == #Hit FROM Red_Swarm;
    BEFORE Egress_tactic FROM Blue_Swarm };

/*-----------------------------------------------
     MARKing traces with special features
-------------------------------------------------*/

IF #Red_UAV FROM Red_Swarm > #Hit FROM Red_Swarm THEN
SAY("Red Swarm was not completely destroyed");
MARK;
    FI;
```

Figure F.3. SvS OnStation MP Code

Run Scope: 2

```
/* K. Giles swarm vs. swarm model with roots blue swarm,
red swarm, and arbiter.
original: 2MAY2017 revised: 17JUL17
-assume no errors, removed swarm commander to
focus on automated tactics
- new approach using tactics, not states
- need to add BUILD ENSURE line to ensure all red UAVs
killed before blue swarm Egress tactic is triggered

v2  revision 21 JUL 2017 John Quartuccio
1. allow coordination of multiple hits with egress using
a DO loop
2. simplified ensure constraint, not quite working yet...

/*-----------------------------
             Actors
-----------------------------*/
SCHEMA SvSOnstationV2


ROOT Arbiter:          Commence_mission
                       Score_attacks
                       End_mission ;

ROOT Red_Swarm:   {+ Red_UAV +};

    Red_UAV:          (+ Present_signature +)
                      (+ ( Detected_by_UAV Incur_attack |
                         Not_detected ) +)
                      End_mission;

        Incur_attack: ( Hit | Miss)  ;


ROOT Blue_Swarm:    {+ Blue_UAV +};

    Blue_UAV:               Evasive_search_tactic
                       (*  (  Track_tactic
                              Attack_tactic ) *)
                              Egress_tactic ;
/* -------------------------------------
  Tactics are composed of plays
----------------------------------------*/

/*      Evasive_search_tactic:      Sensors_ON
                                    Random_pattern;

        Track_tactic:   Sensors_ON
                        Follow_target;

        Attack_tactic:   Sensors_ON
                         Weapon_armed
                             (  Smart_greedy_shooter |
                                Wingman_shooter      |
                                Patrol_box_shooter )
                         Weapon_fire;

        Egress_tactic:      Transit_to_WP
                            Terminal_approach
                            Landing
                            Sensors_OFF;
/*----------------------------
            Interactions
-----------------------------*/
/**/
COORDINATE  $a: Commence_mission            FROM Arbiter,
            $b: Evasive_search_tactic       FROM Blue_Swarm
    DO ADD $a PRECEDES $b;  OD;
/**/
COORDINATE <!> $a: Detected_by_UAV          FROM Red_Swarm,
           <!> $b: Track_tactic             FROM Blue_Swarm
    DO ADD $a PRECEDES $b; OD;

/**/
COORDINATE <!> $a: Attack_tactic            FROM Blue_Swarm,
           <!> $b: Incur_attack             FROM Red_Swarm
    DO ADD $a PRECEDES $b; OD;
/**/
COORDINATE  $a: Hit                         FROM Red_Swarm
    DO COORDINATE
            $b: Egress_tactic               FROM Blue_Swarm
    DO ADD $a PRECEDES $b; OD; OD;

/**/
ENSURE (#Red_UAV - #Hit > 0 -> #Egress_tactic == 0);

/*BUILD {
    ENSURE #Red_UAV FROM Red_Swarm == #Hit FROM Red_Swarm;
    BEFORE Egress_tactic FROM Blue_Swarm };

/*---------------------------------------------
     MARKing traces with special features
----------------------------------------------*/

IF #Red_UAV FROM Red_Swarm > #Hit FROM Red_Swarm THEN
SAY("Red Swarm was not completely destroyed");
MARK;
    FI;
```

Figure F.4. SvS OnStation MP Code Revised

```
/* K. Giles MIO Mission with roots: 2 sub-swarms, SC, and COI. 19SEP2017
- Corrected coordination with SC
- 8SEP17 Added Divide and Amass (shared between 2 swarms)
- Assume no errors or failures
- Incorporated Dr. Auguston's suggestions
- New approach using tactics, not states
- The tactics composition of plays is commented out. If they are used,
the number of use cases increases dramatically. Thus the case for the
Swarm Commander managing the mission at the tactics level.
- Once swarm goes to Egress it should not be able to do anything else
concurrently.
/*-----------------------------
            Actors
-------------------------------*/
SCHEMA MIOMission

ROOT COI:                 present_signature
                         ( get_detected
                            (+ (threaten_swarm | no_threat) +) |
                           not_detected no_threat          );

ROOT Swarm_Commander: command_swarm_launch
                      command_swarm_RTB;

ROOT Sub_Swarm1:           Ingress_tactic
                           Divide_tactic
                          ( Efficient_search_tactic | Monitor_tactic )
                          (+ ( Track_tactic           |
                               Evade_tactic |
                               Efficient_search_tactic ) +)
                           Amass_tactic
                           Egress_tactic;

ROOT Sub_Swarm2:           Ingress_tactic
                           Divide_tactic
                           CommRelay_tactic
                           Amass_tactic
                           Egress_tactic;
/* ------------------------------------
  Tactics are composed of plays
--------------------------------------*/

|       Ingress_tactic: Sensors_OFF
                        Launch
                        Sensors_ON
                        Transit_to_WP;

        Divide_tactic:    Split
                        ( Sensors_ON | Sensors_EMCON )
                          Orbit;

        Efficient_search_tactic:    Sensors_ON
                                (   Ladder_pattern |
                                    Grid_pattern |
                                    Constricting_square_pattern |
                                    Expanding_square_pattern ) ;

        Monitor_tactic:  ( Sensors_EMCON | Sensors_ON)
                          ( Orbit        | Racetrack ) ;

        Track_tactic:   Sensors_ON
                       ( Follow_target | Tail_following );

        Evade_tactic:       ( Sensors_EMCON | Jam )
                            ( Disperse Join )    ;

        CommRelay_tactic:       Sensors_ON
                                Fwd_comm Xmt_video
                              ( Ladder_pattern |
                                Grid_pattern |
                                Constricting_square_pattern |
                                Expanding_square_pattern)  ;

        Amass_tactic:       Join
                            ( Sensors_ON | Sensors_EMCON )
                            Orbit;

        Egress_tactic:      Transit_to_WP
                            Terminal_approach
                            Sensors_OFF
                            Landing;
```

```
/*--------------------------------
        Interactions
----------------------------------*/
COORDINATE  $a: present_signature          FROM COI,
            $b: command_swarm_launch       FROM Swarm_Commander
    DO ADD  $a PRECEDES $b;  OD;

Sub_Swarm1, Sub_Swarm2 SHARE ALL Ingress_tactic;

 COORDINATE $a: command_swarm_launch       FROM Swarm_Commander,
            $b: Ingress_tactic             FROM Sub_Swarm1
    DO ADD $a PRECEDES $b; OD;

 COORDINATE $a: command_swarm_launch       FROM Swarm_Commander,
            $b: Ingress_tactic             FROM Sub_Swarm2
    DO ADD $a PRECEDES $b; OD;

Sub_Swarm1, Sub_Swarm2 SHARE ALL Divide_tactic;

COORDINATE $a: Efficient_search_tactic        FROM Sub_Swarm1,
           $b: ( get_detected | not_detected )  FROM COI
    DO ADD $a PRECEDES $b; OD;

COORDINATE $a: get_detected                FROM COI,
           $b: Track_tactic                FROM Sub_Swarm1
    DO ADD $a PRECEDES $b; OD;

COORDINATE  $a: threaten_swarm             FROM COI,
            $b: Evade_tactic               FROM Sub_Swarm1
    DO ADD  $a PRECEDES $b; OD;

/*COORDINATE    $a: no_threat              FROM COI,
                $b: Track_tactic           FROM Sub_Swarm1
    DO ADD  $a PRECEDES $b; OD; */


Sub_Swarm1, Sub_Swarm2 SHARE ALL Amass_tactic;

Sub_Swarm1, Sub_Swarm2 SHARE ALL Egress_tactic;

COORDINATE  $a: command_swarm_RTB          FROM Swarm_Commander,
            $b: Amass_tactic               FROM Sub_Swarm1
    DO ADD  $a PRECEDES $b; OD;

COORDINATE  $a: command_swarm_RTB          FROM Swarm_Commander,
            $b: Amass_tactic               FROM Sub_Swarm2
    DO ADD  $a PRECEDES $b; OD;

/*------------------------
        Error checking
--------------------------*/
COORDINATE $e: Evade_tactic
        DO IF (#Track_tactic FOLLOWS $e > 0) THEN
        MARK;
        ADD
        SAY("swarm initiated track after evade")
        PRECEDES $e;
        FI;
    OD;
```

Figure F.5. MIO Mission MP Code

```
/* K. Giles HADR Mission with roots:
2 sub-swarms, SC, and potential targets of interest (TOI).
07JUNE2017 revised 24JUL17 to add SC determine TOI
- Assume no errors or failures
- Assume main swarm has executed Divide tactic
- Incorporated Dr. A's suggestions
- New approach using tactics, not states
- The tactics composition of plays is commented out. If they are used,
the number of use cases increases dramatically. Thus the case for the
Swarm Commander managing the mission at the tactics level.
- Once it goes to Egress it should not be able to do anything else concurrently.

/*-----------------------------
            Actors
-----------------------------*/
SCHEMA HADRMission

ROOT TOI:                  present_signature
                            ( get_detected |
                             not_detected )  ;

ROOT Swarm_Commander: command_swarm_launch
                             ( tgt_of_interest    |
                                tgt_not_of_interest |
                                no_tgt                      )
                            command_swarm_RTB;

ROOT Sub_Swarm1:            Ingress_tactic
                            Efficient_search_tactic
                       (    Track_tactic Monitor_tactic     |
                            Track_tactic Resume_EffSearch |
                            Continue_EffSearch )
                            Egress_tactic;

ROOT Sub_Swarm2:            Ingress_tactic
                            CommRelay_tactic
                            Egress_tactic;
/* ------------------------------------
     Tactics are composed of plays
-------------------------------------*/

    /*  Ingress_tactic: Sensors_OFF
                        Launch
                        Sensors_ON
                        Transit_to_WP;

        Efficient_search_tactic:    Sensors_ON
                                 (  Ladder_pattern |
                                    Grid_pattern |
                                    Constricting_square_pattern |
                                    Expanding_square_pattern) ;

        Monitor_tactic:  ( Sensors_EMCON | Sensors_ON )
                         ( Orbit         | Racetrack  ) ;

        Track_tactic:    Sensors_ON
                         Follow_target;

        CommRelay_tactic:   Sensors_ON
                               Fwd_comm Xmt_video
                            (  Ladder_pattern |
                               Grid_pattern |
                               Constricting_square_pattern |
                               Expanding_square_pattern);

        Egress_tactic:      Transit_to_WP
                            Terminal_approach
                            Landing
                            Sensors_OFF;

/*-----------------------------------------------
            Interactions
-----------------------------------------------*/

COORDINATE $a: command_swarm_launch         FROM Swarm_Commander,
           $b: Ingress_tactic               FROM Sub_Swarm1
    DO ADD $a PRECEDES $b; OD;

COORDINATE $a: command_swarm_launch         FROM Swarm_Commander,
           $b: Ingress_tactic               FROM Sub_Swarm2
    DO ADD $a PRECEDES $b; OD;

COORDINATE $a: Efficient_search_tactic          FROM Sub_Swarm1,
           $b: ( get_detected | not_detected )  FROM TOI
    DO ADD $a PRECEDES $b; OD;

COORDINATE $a: get_detected                 FROM TOI,
           $b: ( tgt_of_interest |
                 tgt_not_of_interest )      FROM Swarm_Commander
    DO ADD $a PRECEDES $b; OD;

COORDINATE $a: not_detected                 FROM TOI,
           $b: no_tgt                       FROM Swarm_Commander
    DO ADD $a PRECEDES $b; OD;

COORDINATE $a: tgt_of_interest              FROM Swarm_Commander,
           $b: Monitor_tactic               FROM Sub_Swarm1
    DO ADD $a PRECEDES $b; OD;

COORDINATE $a: tgt_not_of_interest          FROM Swarm_Commander,
           $b: Resume_EffSearch             FROM Sub_Swarm1
    DO ADD $a PRECEDES $b; OD;

COORDINATE $a: no_tgt                       FROM Swarm_Commander,
           $b: Continue_EffSearch           FROM Sub_Swarm1
    DO ADD $a PRECEDES $b; OD;


COORDINATE  $a: command_swarm_RTB           FROM Swarm_Commander,
            $b: Egress_tactic               FROM Sub_Swarm1
    DO ADD  $a PRECEDES $b; OD;

COORDINATE  $a: command_swarm_RTB           FROM Swarm_Commander,
            $b: Egress_tactic               FROM Sub_Swarm1
    DO ADD  $a PRECEDES $b; OD;

COORDINATE  $a: Egress_tactic               FROM Sub_Swarm1,
            $b: Egress_tactic               FROM Sub_Swarm2
    DO ADD  $a PRECEDES $b; OD;
```

Figure F.6. HADR Mission MP Code

# APPENDIX G:
# Monterey Phoenix Model Code Files

This appendix contains the MP code files used in the mission diagrams that are referenced and described in Chapter 4, and pictured in Appendix F.

```
/*
K. Giles 8SEP2017
SvS overall mission with all actors from perspective of blue swarm at
    mission level with phases.
Incorporated Dr. Auguston's suggestions.
Adding to Revill's: "Incorporating Failure Modes and Failsafe Behaviors"
     and "Perform_Mission" event in Swarm2-v8.mp
*/
SCHEMA Swarm_V_Swarm_Mission

/* ----MAJOR ACTORS ----*/

ROOT Arbiter:         /* Receives all UAV positions, re-broadcasts
                   them for common op picture, and scores mission */
   {* Receive_UAV_Positions *}
   Broadcast_UAV_Positions
   Score_Mission;

ROOT Blue_MC:           /*Blue Swarm Mission Commander  */
   Conduct_Mission_Brief
   Command_Commence_Mission
   (* Monitor_Mission_Tactics *)
   Conduct_Mission_Debrief;

ROOT Blue_LO:           /*Blue Swarm Launch Operator  */
   Preflight_Blue_Swarm
   Conduct_Mission_Brief
   Launch_Blue_Swarm
   Recover_Blue_Swarm
   Conduct_Mission_Debrief;

ROOT Blue_SO:           /*Blue Swarm Operator  */
```

```
  Conduct_Mission_Brief
  Command_Search_and_Track_Tactics
  Command_Blue_Attack_Tactic
  Conduct_Mission_Debrief;

ROOT Blue_LT:          /*Blue Swarm Launch Tech */
  {* Preflight_Blue_Swarm *}
  Conduct_Mission_Brief
  Recover_Blue_Swarm
  Postflight_Blue_Swarm
  Conduct_Mission_Debrief;


ROOT Blue_SP:          /*Blue Swarm Safety Pilot  */
  Conduct_Mission_Brief
  (* Assess_Flight_Behavior
        ( Behavior_is_Normal |
          Behavior_is_Abnormal Manually_Fly_Errant_UAV) *)
    Conduct_Mission_Debrief;

ROOT Range_Control:
  Give_Green_Range_Command
    (* Receive_UAV_Recovered_Notification *);

ROOT Blue_SM:          /*Blue Swarm Monitor  */
  Conduct_Mission_Brief
    Command_Ingress_Tactic
    Monitor_Mission_Health
  Command_Egress_Tactic
    Conduct_Mission_Debrief;

ROOT Environment: (* Present_Environment_Signature *);

ROOT Blue_Swarm: /*  Blue Swarm is composed of multiple UAVs  */
    {+ UAV +};

 UAV:    Undergo_PreFlight
    Launch
        Detect_Environment_Signature
    Send_UAV_Positions
```

```
            Receive_Red_UAV_Positions
            Ingress_Tactic
            Intercept_Swarm_Ready_WP
            Report_Health
            Search_and_Track_Tactics
            Attack_Tactic
      [Evade_Tactic]
            Egress_Tactic
            Land;

/*-------- MAJOR PHASES OF MISSION ---------------*/

ROOT Preflight:
   Conduct_Mission_Brief
   Check_Configuration
   Set_Parameters
   Engine_Runup;

ROOT Inflight:
   Launch_Blue_Swarm
   Ingress_Tactic
   Search_and_Track_Tactics
   Attack_Tactic
   Score_Mission
   Egress_Tactic;

ROOT Postflight:
   Recover_Blue_Swarm
   Conduct_Mission_Debrief;

/*---------OVERLAP OF ACTORS AND MISSION PHASES---------*/

Blue_MC, Blue_LT, Blue_LO, Blue_SO, Blue_SP, Blue_SM, Preflight
     SHARE ALL Conduct_Mission_Brief;

Blue_LO, Inflight  SHARE ALL Launch_Blue_Swarm;

Blue_Swarm, Inflight SHARE ALL Ingress_Tactic,
      Search_and_Track_Tactics, Attack_Tactic, Egress_Tactic;
```

```
Arbiter, Inflight SHARE ALL Score_Mission;

Blue_LO, Blue_LT, Postflight SHARE ALL Recover_Blue_Swarm;

Blue_MC, Blue_LT, Blue_LO, Blue_SO, Blue_SP, Blue_SM, Postflight
    SHARE ALL Conduct_Mission_Debrief;

/*----------COORDINATION STATEMENTS---------------*/

COORDINATE  <!> $a: Preflight_Blue_Swarm    FROM Blue_LT,
       <!> $b: Undergo_PreFlight        FROM Blue_Swarm
    DO ADD $a PRECEDES $b; OD;


COORDINATE  $a: Give_Green_Range_Command       FROM Range_Control,
       $b: Command_Commence_Mission     FROM Blue_MC
       DO ADD $a PRECEDES $b; OD;

COORDINATE <!>  $a: Send_UAV_Positions       FROM Blue_Swarm,
           <!>  $b: Receive_UAV_Positions   FROM Arbiter
         DO ADD  $a PRECEDES $b; OD;

COORDINATE  $a: Command_Commence_Mission       FROM Blue_MC,
       $b: Launch_Blue_Swarm          FROM Blue_LO
       DO ADD $a PRECEDES $b; OD;

/* need one-many coordination here */
COORDINATE  $a: Launch_Blue_Swarm          FROM Blue_LO
    DO COORDINATE   <!>  $b: Launch      FROM Blue_Swarm
         DO ADD $a PRECEDES $b; OD;
  OD;

/* need one-many coordination here */

COORDINATE  $a: Command_Ingress_Tactic       FROM Blue_SM
    DO COORDINATE   <!> $b: Ingress_Tactic  FROM Blue_Swarm
    DO ADD $a PRECEDES $b; OD;
  OD;

COORDINATE  $a: Command_Egress_Tactic              FROM Blue_SM
```

```
        DO COORDINATE    <!> $b: Egress_Tactic       FROM Blue_Swarm
        DO ADD $a PRECEDES $b; OD;
  OD;

COORDINATE   $a: Land                         FROM Blue_Swarm
               DO COORDINATE $b: Recover_Blue_Swarm      FROM Blue_LT
          DO ADD $a PRECEDES $b; OD;
          OD;

COORDINATE <!>   $a: Report_Health                      FROM Blue_Swarm
        DO COORDINATE    <!>   $b: Monitor_Mission_Health        FROM Blue_SM
     DO ADD $a PRECEDES $b; OD;
          OD;

COORDINATE <!>   $a: Present_Environment_Signature            FROM
    Environment
     DO COORDINATE   <!>   $b: Detect_Environment_Signature   FROM
    Blue_Swarm
     DO ADD $a PRECEDES $b; OD;
          OD;
```

```
/*  K. Giles 7MAY2016, revised 7JUL2017
Swarm vs. Swarm air battle only
(with help from Dr. Auguston)*/

/* run for scope 1 or 2, there still will be two swarms*/

SCHEMA BlueSwarm_vs_RedSwarm

/* Swarm has at least one UAV initially */
Swarm: {+ UAV +};

/* UAV behavior is constrained for at most one shot
     We assume that if UAV went in action,
      it makes at least one move.
      Mutual UAV destruction is possible.
     UAVs make AA (air-to-air) attacks or
     AG (air-to-ground) attacks.
```

```
*/
UAV: Attack ( Shoot_enemy_UAV | Attack_enemy_base)
     (killed | returns_to_base);

Shoot_enemy_UAV: (valid_AA_hit | miss);

Attack_enemy_base: (valid_AG_hit | no_hit);

/* Blue and Red contain a single Swarm,
so make them sets of Swarms
*/
ROOT Blue: Swarm;

ROOT Red: Swarm;

/*  coordinate shooters and targets.
<!> is needed because Swarm is not ordered,
     it is a set */
COORDINATE <!> $h: valid_AA_hit FROM Blue,
<!> $d: killed FROM Red
DO ADD $h PRECEDES $d; OD;

COORDINATE <!> $h: valid_AA_hit FROM Red,
<!> $d: killed FROM Blue
DO ADD $h PRECEDES $d; OD;

/* calculate points and provide statistics
     assuming 5 points for each destroyed
     enemy UAV and 1 point for base hit
*/
  SAY("Blue Swarm has " 5 * #valid_AA_hit FROM Blue
     " points for destroyed UAVs and "
     #valid_AG_hit FROM Blue
     " for enemy base hits");

     /*  MARKing traces with special features */
IF #UAV FROM Blue == #killed FROM Blue THEN
SAY("Blue Swarm was completely destroyed");
MARK;
     FI;
```

```
  SAY("Red Swarm has " 5 * #valid_AA_hit FROM Red
     " points for destroyed UAVs and "
     #valid_AG_hit FROM Red
     " for enemy base hits");

     /*  MARKing traces with special features */
IF #UAV FROM Red == #killed FROM Red THEN
SAY("Red Swarm was completely destroyed");
MARK;
     FI;
```

```
/* K. Giles original: 2MAY2017 revised: 17JUL17
revised: 21 JUL 2017 with John Quartuccio
1. Allow coordination of multiple hits with egress using
a DO loop
2. Simplified ensure constraint, not quite working yet.
Swarm vs. swarm model with roots blue swarm, red swarm, and arbiter.
Assume no errors, removed swarm commander to focus on automated tactics.
New approach using tactics, not states.
Need to ensure all red UAVs killed before blue swarm Egress tactic is
    triggered.

/*-----------Actors-----------*/
SCHEMA SvSOnstationV2

ROOT Arbiter:        Commence_mission
                Score_attacks
                End_mission ;

ROOT Red_Swarm:    {+ Red_UAV +};

  Red_UAV:      (+ Present_signature +)
             (+ ( Detected_by_UAV Incur_attack |
                        Not_detected ) +)
             End_mission;

     Incur_attack: ( Hit | Miss)   ;
```

```
ROOT Blue_Swarm:  {+ Blue_UAV +};

  Blue_UAV:          Evasive_search_tactic
                      (*  (  Track_tactic
                         Attack_tactic ) *)
                         Egress_tactic ;
/* ------------------------------------
  Tactics are composed of plays
---------------------------------------*/

/*    Evasive_search_tactic:    Sensors_ON
                         Random_pattern;

    Track_tactic:    Sensors_ON
                  Follow_target;

    Attack_tactic:    Sensors_ON
             Weapon_armed
                              (  Smart_greedy_shooter |
                  Wingman_shooter    |
                  Patrol_box_shooter )
             Weapon_fire;

    Egress_tactic:     Transit_to_WP
              Terminal_approach
              Landing
              Sensors_OFF;
/*---------Interactions ------*/
/**/
COORDINATE  $a: Commence_mission      FROM Arbiter,
      $b: Evasive_search_tactic   FROM Blue_Swarm
  DO ADD $a PRECEDES $b;  OD;
/**/
COORDINATE <!> $a: Detected_by_UAV      FROM Red_Swarm,
        <!> $b: Track_tactic        FROM Blue_Swarm
  DO ADD $a PRECEDES $b; OD;

/**/
```

```
COORDINATE <!> $a: Attack_tactic      FROM Blue_Swarm,
        <!> $b: Incur_attack        FROM Red_Swarm
  DO ADD $a PRECEDES $b; OD;
/**/
COORDINATE  $a: Hit                 FROM Red_Swarm
    DO COORDINATE
         $b: Egress_tactic          FROM Blue_Swarm
  DO ADD $a PRECEDES $b; OD; OD;

/**/
ENSURE (#Red_UAV - #Hit > 0 -> #Egress_tactic == 0);

/*BUILD {
  ENSURE #Red_UAV FROM Red_Swarm == #Hit FROM Red_Swarm;
  BEFORE Egress_tactic FROM Blue_Swarm };

/*-------------------------------------------------
     MARKing traces with special features
-------------------------------------------------*/

IF #Red_UAV FROM Red_Swarm > #Hit FROM Red_Swarm THEN
SAY("Red Swarm was not completely destroyed");
MARK;
    FI;
```

```
/* K. Giles 19SEP2017
MIO Mission with roots: 2 sub-swarms, SC, and COI.
Added Divide and Amass (shared between 2 swarms).
Assume no errors or failures.
New approach using tactics, not states.
Incorporated Dr. Auguston's suggestions.
The tactics composition of plays is commented out. If they are used, the
     number of use cases increases dramatically.
Swarm Commander manages the mission at the tactics level.
Once swarm goes to Egress it should not be able to do anything else
    concurrently.

/*--------- Actors ---------*/
```

```
SCHEMA MIOMission

ROOT COI:          present_signature
          ( get_detected
                    (+ (threaten_swarm | no_threat) +) |
            not_detected no_threat          );

ROOT Swarm_Commander: command_swarm_launch
          command_swarm_RTB;

ROOT Sub_Swarm1:          Ingress_tactic
          Divide_tactic
               ( Efficient_search_tactic | Monitor_tactic )
                  (+ ( Track_tactic          |
                     Evade_tactic |
            Efficient_search_tactic ) +)
          Amass_tactic
                  Egress_tactic;

ROOT Sub_Swarm2:     Ingress_tactic
          Divide_tactic
          CommRelay_tactic
          Amass_tactic
          Egress_tactic;

/* -----Tactics are composed of plays----------*/

/*     Ingress_tactic: Sensors_OFF
          Launch
          Sensors_ON
          Transit_to_WP;

       Divide_tactic:     Split
            ( Sensors_ON | Sensors_EMCON )
                     Orbit;

    Efficient_search_tactic:     Sensors_ON
                 ( Ladder_pattern |
                          Grid_pattern |
```

```
                                    Constricting_square_pattern |
                                    Expanding_square_pattern ) ;

     Monitor_tactic:  ( Sensors_EMCON | Sensors_ON)
                ( Orbit       | Racetrack ) ;

     Track_tactic:    Sensors_ON
                     ( Follow_target | Tail_following );

         Evade_tactic:        ( Sensors_EMCON | Jam )
                         ( Disperse Join )   ;

     CommRelay_tactic:      Sensors_ON
                        Fwd_comm Xmt_video
                 (   Ladder_pattern |
                                 Grid_pattern |
                                 Constricting_square_pattern |
                                 Expanding_square_pattern)   ;

         Amass_tactic:    Join
                     ( Sensors_ON | Sensors_EMCON )
                               Orbit;

     Egress_tactic:     Transit_to_WP
                Terminal_approach
                Sensors_OFF
                Landing;

/*------Interactions---------*/

COORDINATE   $a: present_signature        FROM COI,
       $b: command_swarm_launch     FROM Swarm_Commander
  DO ADD   $a PRECEDES $b;  OD;

Sub_Swarm1, Sub_Swarm2 SHARE ALL Ingress_tactic;

 COORDINATE $a: command_swarm_launch     FROM Swarm_Commander,
         $b: Ingress_tactic          FROM Sub_Swarm1
  DO ADD $a PRECEDES $b; OD;
```

```
 COORDINATE $a: command_swarm_launch     FROM Swarm_Commander,
        $b: Ingress_tactic          FROM Sub_Swarm2
  DO ADD $a PRECEDES $b; OD;

Sub_Swarm1, Sub_Swarm2 SHARE ALL Divide_tactic;

COORDINATE $a: Efficient_search_tactic       FROM Sub_Swarm1,
       $b: ( get_detected | not_detected )  FROM COI
  DO ADD $a PRECEDES $b; OD;

COORDINATE $a: get_detected          FROM COI,
       $b: Track_tactic          FROM Sub_Swarm1
  DO ADD $a PRECEDES $b; OD;

COORDINATE  $a: threaten_swarm          FROM COI,
      $b: Evade_tactic          FROM Sub_Swarm1
  DO ADD  $a PRECEDES $b; OD;

/*COORDINATE  $a: no_threat         FROM COI,
        $b: Track_tactic        FROM Sub_Swarm1
  DO ADD  $a PRECEDES $b; OD; */

Sub_Swarm1, Sub_Swarm2 SHARE ALL Amass_tactic;

Sub_Swarm1, Sub_Swarm2 SHARE ALL Egress_tactic;

COORDINATE  $a: command_swarm_RTB      FROM Swarm_Commander,
      $b: Amass_tactic          FROM Sub_Swarm1
  DO ADD  $a PRECEDES $b; OD;

COORDINATE  $a: command_swarm_RTB      FROM Swarm_Commander,
      $b: Amass_tactic          FROM Sub_Swarm2
  DO ADD  $a PRECEDES $b; OD;

/*-----Error checking-----*/

COORDINATE $e: Evade_tactic
    DO IF (#Track_tactic FOLLOWS $e > 0) THEN
    MARK;
    ADD
```

```
    SAY("swarm initiated track after evade")
    PRECEDES $e;
    FI;
  OD;
```

```
/* K. Giles 07JUNE2017
revised 24JUL17 to add SC determines TOI
HADR Mission with roots:
2 sub−swarms, SC, and potential targets of interest (TOI).
Assume no errors or failures.
Assume main swarm has already executed Divide tactic.
Incorporated Dr. Auguston's suggestions.
New approach using tactics, not states.
The tactics composition of plays is commented out. If they are used, the
    number of use cases increases dramatically. Thus the case for the
    SC managing the mission at the tactics level.
Once sub−swarm goes to Egress it should not be able to do anything else
    concurrently.

/*−−−−−−−−−−−−−−Actors−−−−−−−−−−−−−−−−−*/
SCHEMA HADRMission

ROOT TOI:           present_signature
                  ( get_detected |
                    not_detected )  ;

ROOT Swarm_Commander: command_swarm_launch
                ( tgt_of_interest    |
                                tgt_not_of_interest |
                                no_tgt           )
              command_swarm_RTB;

ROOT Sub_Swarm1:        Ingress_tactic
                      Efficient_search_tactic
                      (    Track_tactic Monitor_tactic   |
                           Track_tactic Resume_EffSearch |
                           Continue_EffSearch )
                        Egress_tactic;
```

```
ROOT Sub_Swarm2:     Ingress_tactic
              CommRelay_tactic
              Egress_tactic;

/* ----Tactics are composed of plays---------*/

  /*  Ingress_tactic: Sensors_OFF
              Launch
              Sensors_ON
              Transit_to_WP;

    Efficient_search_tactic:    Sensors_ON
                        (  Ladder_pattern |
                                Grid_pattern |
                                Constricting_square_pattern |
                                Expanding_square_pattern) ;

    Monitor_tactic:  ( Sensors_EMCON | Sensors_ON )
              ( Orbit      | Racetrack   ) ;

    Track_tactic:     Sensors_ON
                    Follow_target;

    CommRelay_tactic:    Sensors_ON
                      Fwd_comm  Xmt_video
                (  Ladder_pattern |
                              Grid_pattern |
                              Constricting_square_pattern |
                              Expanding_square_pattern);

    Egress_tactic:     Transit_to_WP
                Terminal_approach
                Landing
                Sensors_OFF;
/*-----------Interactions-------------*/

COORDINATE $a: command_swarm_launch      FROM Swarm_Commander,
        $b: Ingress_tactic         FROM Sub_Swarm1
  DO ADD $a PRECEDES $b; OD;
```

```
COORDINATE $a: command_swarm_launch      FROM Swarm_Commander,
        $b: Ingress_tactic          FROM Sub_Swarm2
  DO ADD $a PRECEDES $b; OD;

COORDINATE $a: Efficient_search_tactic      FROM Sub_Swarm1,
        $b: ( get_detected | not_detected )  FROM TOI
  DO ADD $a PRECEDES $b; OD;

COORDINATE $a: get_detected          FROM TOI,
        $b: ( tgt_of_interest |
                  tgt_not_of_interest )    FROM Swarm_Commander
  DO ADD $a PRECEDES $b; OD;

COORDINATE $a: not_detected          FROM TOI,
        $b: no_tgt              FROM Swarm_Commander
  DO ADD $a PRECEDES $b; OD;

COORDINATE $a: tgt_of_interest         FROM Swarm_Commander,
        $b: Monitor_tactic          FROM Sub_Swarm1
  DO ADD $a PRECEDES $b; OD;

COORDINATE $a: tgt_not_of_interest       FROM Swarm_Commander,
        $b: Resume_EffSearch         FROM Sub_Swarm1
  DO ADD $a PRECEDES $b; OD;

COORDINATE $a: no_tgt               FROM Swarm_Commander,
        $b: Continue_EffSearch      FROM Sub_Swarm1
  DO ADD $a PRECEDES $b; OD;


COORDINATE  $a: command_swarm_RTB      FROM Swarm_Commander,
       $b: Egress_tactic          FROM Sub_Swarm1
  DO ADD  $a PRECEDES $b; OD;

COORDINATE  $a: command_swarm_RTB      FROM Swarm_Commander,
       $b: Egress_tactic          FROM Sub_Swarm1
  DO ADD  $a PRECEDES $b; OD;

COORDINATE  $a: Egress_tactic         FROM Sub_Swarm1,
```

```
    $b: Egress_tactic        FROM Sub_Swarm2
 DO ADD  $a PRECEDES $b; OD;
```

```
/* K. Giles 09AUG2017
SvS FSM model to MP conversion simplified
to SVS Onstation tactics as states only.
Based on Dr.Auguston's microwave example, dfa_2.
---------------------------------------*/
SCHEMA SVS_FSM

ROOT EVSearch_behavior:      SwarmRdy
                SwarmRdy_EVSearch
                EVSearch
            (* (      EVSearch_Track   |
                              EVSearch_Evade   |
                      EVSearch_Egress   )*);

SwarmRdy_EVSearch:  onsta_WP_int;
EVSearch_Track:    tgt_det;
EVSearch_Evade:    threat;
EVSearch_Egress:  low_pwr_egress;

ROOT Track_behavior:       EVSearch_Track
                 Track
                 (* (   Track_EVSearch |
                              Track_Evade      |
                              Track_Attack     |
                      Track_Egress   )*);

Track_EVSearch: lost_track;
Track_Evade:   threat;
Track_Attack:    tgt_in_range;
Track_Egress:    low_pwr_egress;

EVSearch_behavior, Track_behavior SHARE ALL EVSearch_Track,
    Track_EVSearch;

ROOT Evade_behavior:  (* ( Track_Evade | EVSearch_Evade )
```

```
                 Evade
             (   Evade_EVSearch   | Evade_Egress ) *);

Evade_EVSearch:    no_threat;
Evade_Egress:      low_pwr_egress;

Evade_behavior, Track_behavior     SHARE ALL Track_Evade;
EVSearch_behavior, Evade_behavior    SHARE ALL EVSearch_Evade,
    Evade_EVSearch;

/*COORDINATE   $a: threat,
        $b: Evade
  DO ADD $a PRECEDES $b; OD;*/


ROOT Attack_behavior: (*( Track_Attack   |
                       BDA_Attack )
                       Attack
                (   Attack_BDA | Attack_Egress )*);

BDA_Attack:    tgt_not_destroyed;
Attack_BDA:    assess_hit;
Attack_Egress:  low_pwr_egress;

Track_behavior, Attack_behavior SHARE ALL Track_Attack;

ROOT BDA_behavior:   (* Attack_BDA
              BDA
                ( BDA_Attack |   BDA_EVSearch ) *);

BDA_EVSearch:    tgt_destroyed;

Attack_behavior, BDA_behavior    SHARE ALL Attack_BDA, BDA_Attack;
BDA_behavior, EVSearch_behavior SHARE ALL BDA_EVSearch;

ROOT Egress_behavior: ( EVSearch_Egress | Track_Egress |
               Evade_Egress | Attack_Egress )
               Egress;

EVSearch_behavior, Egress_behavior   SHARE ALL EVSearch_Egress;
```

```
Track_behavior, Egress_behavior   SHARE ALL Track_Egress;
Evade_behavior, Egress_behavior   SHARE ALL Evade_Egress;
Attack_behavior, Egress_behavior  SHARE ALL Attack_Egress;

ROOT Path:  ;   /* Path is container for path sequence*/

COORDINATE  $a:   ( SwarmRdy | EVSearch | Track | Evade |
                  Attack | BDA | Egress |
                    onsta_WP_int | tgt_det | lost_track |
                  low_pwr_egress | threat | no_threat |
                  assess_hit | tgt_destroyed |
                  tgt_not_destroyed | tgt_in_range )
  DO
    ADD $a IN Path;
  OD;

COORDINATE  <SORT CUT_END>     $a: $$EVENT SUCH THAT $a IN Path,
        <SORT CUT_FRONT>  $b: $$EVENT SUCH THAT $a IN Path
  DO
    ADD $a PRECEDES $b;
  OD;
/*-----ASSERTION CHECKING-----*/

ENSURE FOREACH $x:low_pwr_egress
  #Track AFTER $x ==0;
/*
STATES:

S1:    SwarmReady
S2:    EVSearch
S3:    Track
S4:    Evade
S5:    Attack
S6:    BDA
S7:    Egress

TRANSITIONS:

a:     onstation WP intercepted    (occurs once)
b:     target detected          (occurs 1 or more times)
```

```
c:     lost track            (occurs 0 or more times)
d:     low power or egress command (occurs once)
e:     threatened            (occurs 0 or more times)
f:     not threatened          (occurs 0 or more times)
g:     assess hit            (occurs 1 or more times)
h:     target destroyed        (occurs 0 or more times)
i:     target not destroyed      (occurs 0 or more times)
j:     target within range       (occurs 1 or more times)
```

THIS PAGE INTENTIONALLY LEFT BLANK

# List of References


[1] K. Giles and K. Giammarco, "Mission-based architecture for swarm composability (MASC)," in *Proceedings of the Complex Adaptive Systems Conference*, Elsevier, Ed. Chicago, IL: Elsevier B.V., 2017.

[2] T. H. Chung, K. D. Jones, M. A. Day, M. Jones, and M. Clement, "50 vs. 50 by 2015: Swarm vs. swarm UAV live-fly competition at the Naval Postgraduate School," in *AUVSI North America*. Washington, DC: AUVSI, 2013, pp. 1–20.

[3] T. H. Chung, M. R. Clement, M. A. Day, K. D. Jones, D. Davis, and M. Jones, "Live-fly, large-scale field experimentation for large numbers of fixed-wing UAVs," in *Proceedings of the IEEE International Conference on Robotics and Automation*, A. Okamura, Ed. Stockholm, Sweden: IEEE, 2016.

[4] D. T. Davis, T. H. Chung, M. R. Clement, and M. A. Day, "Consensus-based data sharing for large-scale aerial swarm coordination in lossy communications environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Daejeon, Korea: IEEE, 2016, pp. 3801–3808.

[5] P. Scharre, "Robotics on the Battlefield Part II The Coming Swarm," Center for a New American Security, Washington, DC, Tech. Rep., 2014.

[6] P. T. Beery, "A model based systems engineering methodology for employing architecture in system analysis: Developing simulation models using systems modeling language products to link architecture and analysis," Ph.D. dissertation, Dept. of Sys. Eng., NPS, Monterey, CA, 2016.

[7] S. E. Gillespie, "The system of systems architecture assessment model," Ph.D. dissertation, Dept. of Sys. Eng., NPS, Monterey, CA, 2016.

[8] R. Gold, "Mission Engineering," Springfield, VA, pp. 1–15, 2016.

[9] D. F. Beam, "Systems engineering and integration as a foundation for mission engineering," M.S. thesis, NPS, Monterey, CA, 2015.

[10] P. N. Squire, S. M. Galster, and R. Parasuraman, "The effects of levels of automation in the human control of multiple robots in the Roboflag simulation environment," in *Human Performance, Situation Awareness and Automation: Current Research and Trends*, P. A. Vincenzi, Dennis A. , Mouloua, Mustapha, Hancock, Ed., no. 074. Daytona Beach, FL: Embry-Riddle Aeronautical University, 2004, vol. 2, pp. 48–53.

[11] C. E. Harriott, A. E. Seiffert, S. T. Hayes, and J. A. Adams, "Biologically-inspired human-swarm interaction metrics," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 58, no. 1, pp. 1471–1475, 2014.

[12] M. L. Cummings, "Human supervisory control of swarming networks," in *Autonomous Intelligent Networked Systems Conference*. Varna, Bulgaria: IEEE, 2004, pp. 1–9.

[13] M. L. Cummings and P. Mitchell, "Automated scheduling decision support for supervisory control of multiple UAVs," *Journal of Aerospace Computing, Information, and Communication*, vol. 3, no. 6, pp. 294–308, 2006.

[14] M. Cummings, S. Bruni, S. Mercier, and P. Mitchell, "Automation architecture for single operator, multiple UAV command and control," *The International C2 Journal*, vol. 1, pp. 1–24, 2007.

[15] Department of Defense, "Department of defense announces successful micro-drone demonstration," Washington, DC, 2017.

[16] DARPA Public Affairs, "Service academies swarm challenge pushes the boundaries of autonomous swarm capabilities," Arlington, VA, may 2017.

[17] M. Cummings, C. Nehme, and J. Crandall, "Predicting operator capacity for supervisory control of multiple UAVs previous experimental multiple UAV studies," *Innovations in Intelligent Machines-1*, vol. 37, pp. 11–37, 2007.

[18] R. Simmons, D. Apfelbaum, D. Fox, R. P. Goldman, K. Zita Haigh, D. J. Musliner, M. Pelican, and S. Thrun, "Coordinated deployment of multiple, heterogeneous robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Takamatsu, Japan: IEEE, 2000, pp. 127–140.

[19] R. P. Goldman, C. A. Miller, H. B. Funk, and J. Meisner, "Optimizing to satisfice: using optimization to guide users," in *Proceedings of the American Helicopter Society's International Specialists Meeting on Unmanned Aerial Vehicles*, no. January. Phoenix, AZ: AHSI International, Inc., 2005, pp. 18–20.

[20] G. Coppin and F. Legras, "Autonomy spectrum and performance perception issues in swarm supervisory control," in *Proceedings of the IEEE*, no. 3, 2012, vol. 100, pp. 590–603.

[21] D. Jackson, *Software Abstractions: Logic, Language and Analysis*, revised ed. Cambridge, MA: MIT Press, 2006.

[22] R. Giachetti, "Systems engineering thesis methods," Monterey, CA, 2016.

[23] N. Cross and R. Roy, *Engineering Design Methods*, 4th ed. New York: Wiley, 1989.

[24] G. Beni and J. Wang, "Swarm intelligence in cellular robotic systems," in *Robotics and Biological Systems Towards a New Bionics?*, P. Dario, G. Sandini, and P. Aebischer, Eds. Berlin Heidelberg: Springer-Verlag, 1993, pp. 703–712.

[25] G. Beni, "From swarm intelligence to swarm robotics," in *International Workshop on Swarm Robotics*. Santa Monica, CA: Springer Berlin Heidelberg, 2004, pp. 1–9.

[26] T. H. Chung, "Dr. Chung directed study notes 1," Monterey, CA, 2015.

[27] M. J. Matarić, "Issues and approaches in the design of collective autonomous agents," *Robotics and Autonomous Systems*, vol. 16, no. (2-4), pp. 321–331, 1995.

[28] T. H. Chung, "Dr. Chung directed study notes 2," Monterey, CA, 2015.

[29] E. Weisel, M. Petty, and R. Mielke, "Validity of models and classes of models in semantic composability," in *Proceedings of the Fall 2003 Simulation Interoperability Workshop*, Simulation Interoperability Standards Organization, Ed. Orlando, FL: Simulation Interoperability Standards Organization, 2003, vol. 9, p. 68.

[30] P. C. Clements, R. Kazman, and M. Klein, "Evaluating Software Architectures: Methods and Case Studies," in *Evaluating Software Architectures: Methods and Case Studies*. London: Addison-Wesley Professional, 2001, ch. 2, pp. 19–42.

[31] K. Giammarco and K. Giles, "Verification and validation of behavior models using lightweight formal methods," in *15th Annual Conference on Systems Engineering Research*, A. M. Madni, B. Boehm, D. A. Erwin, R. Ghanem, and M. Wheaton, Eds., Redondo Beach, CA, 2017.

[32] R. A. David and P. Nielsen, "Defense science board summer study on autonomy," Defense Science Board, Washington, DC, Tech. Rep. June 2016, 2016.

[33] DARPA TTO, "Broad Agency Announcement (BAA) Offensive Swarm Enabled Tactics (OFFSET) Tactical Technology Office (TTO)," Arlington, VA, pp. 1–52, 2017.

[34] E. Frew and T. Brown, "Airborne communication networks for small unmanned aircraft systems," *Proceedings of the IEEE*, vol. 96, no. 12, 2008.

[35] Department of Defense, "Unmanned systems integrated roadmap FY2013-2038," DoD, Washington, DC, Tech. Rep., 2013.

[36] A. Finn and S. Scheding, *Developments and Challenges for Autonomous Unmanned Vehicles*. Berlin: Springer, 2010.

[37] L. Carr, K. Lambrecht, and G. Whittier, "Unmanned aerial vehicle operational test and evaluation lessons learned," Institute for Defense Analysis, Alexandria, VA, Tech. Rep. IDA Paper P-3821, 2003.

[38] E. Sahin, "Swarm robotics: from sources of inspiration to domains of application," in *International workshop on swarm robotics*, no. 631. Santa Monica, CA: Springer Berlin Heidelberg, 2005.

[39] Y. Cao, A. Fukunaga, A. Kahng, and F. Meng, "Cooperative mobile robotics: antecedents and directions," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*. Pittsburgh, PA: IEEE, 1995, vol. 23, pp. 226–234.

[40] B. T. Clough, "Metrics, schmetrics! How the heck do you determine a UAV's autonomy anyway?" in *Performance Metrics for Intelligent Systems Workshop*. Gaithersburg, MD: Air Force Research Lab, 2002.

[41] A. Minai and D. Braha, *Complex Engineered Systems: Science Meets Technology*. Berlin Heidelberg: Springer, 2006.

[42] E. Rechtin and M. W. Maier, *The Art of Systems Architecting*, 3rd ed. Boca Raton, FL: CRC Press, 2010.

[43] M. F. Stumborg, R. Reesman, and S. W. Klein, "Cultural and Organizational Impediments to Unmanned Systems in the Department of the Navy," Center for Naval Analyses, Arlington, VA, Tech. Rep. July, 2017.

[44] A. Dekker, "A taxonomy of network centric warfare architectures," Defence Sciences & Technology Organisation, Canberra, Australia, Tech. Rep., 2008.

[45] T. C. Lueth and T. Laengle, "Task description, decomposition, and allocation in a distributed autonomous multi-agent robot system," in *Intelligent Robots and Systems (IROS) 1994: Advanced Robotic Systems and The Real World Advanced Robotic Systems and The Real World*, Munich, Germany, 1994, pp. 1516–1523.

[46] R. McCune, R. Purta, M. Dobski, A. Jaworski, G. Madey, A. Madey, Y. Wei, and M. B. Blake, "Investigations of DDDAS for command and control of UAV swarms with agent-based modeling," in *Proceedings of the 2013 Winter Simulation Conference: Making Decisions in a Complex World*. Washington, DC: IEEE, 2013, pp. 1467–1478.

[47] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in UAV communication networks," *IEEE Communications Surveys & Tutorials*, no. c, pp. 1–1, 2015.

[48] C. Gao, Z. Zhen, and H. Gong, "A self-organized search and attack algorithm for multiple unmanned aerial vehicles," *Aerospace Science and Technology*, vol. 54, pp. 229–240, 2016.

[49] J. H. Holland, "Complex adaptive systems," *A New Era in Computation*, vol. 121, no. 1, pp. 17–30, 1992.

[50] X.-S. Yang, "Firefly algorithm, Levy flights and global optimization," in *Research and Development in Intelligent Systems XXVI: Incorporating Applications and Innovations in Intelligent Systems XVII*, M. Bramer, R. Ellis, and M. Petridis, Eds. London: Springer, 2010, pp. 209–218.

[51] P. Gaudiano, E. Bonabeau, and B. Shargel, "Evolving behaviors for a swarm of unmanned air vehicles," in *IEEE Swarm Intelligence Symposium*. Pasadena, CA: IEEE, 2005, pp. 317–324.

[52] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.

[53] G. Dudek, M. R. M. Jenkin, E. Milios, and D. Wilkes, "A taxonomy for multi-agent robotics," *Autonomous Robots*, vol. 3, no. 4, pp. 375–397, 1996.

[54] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: A review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, no. 1, pp. 1–41, 2013.

[55] D. Buede, *The Engineering Design of Systems: Models and Methods*, 2nd ed. Hoboken, NJ: John Wiley & Sons, Inc., 2009.

[56] E. Bonabeau, "Agent-based modeling: methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences*, vol. 99, no. suppl. 3, pp. 7280–7287, 2002.

[57] A. Borshchev and A. Filippov, "From system dynamics and discrete event to practical agent based modeling: reasons, techniques, tools," in *The 22nd International Conference of the System Dynamics Society*, M. Kennedy, G. W. Winch, R. S. Langer, J. I. Rowe, and J. M. Yanni, Eds. Oxford, England: Wiley, 2004, vol. 22.

[58] M. F. Munoz, "Agent-based simulation and analysis of a defensive UAV swarm against an enemy UAV swarm," M.S. thesis, Dept. of Ops. Research, NPS, Monterey, CA, 2011.

[59] M. W. Maier, "Architecting principles for systems of systems," in *INCOSE International Symposium*, no. 1. Boston, MA: Wiley, 1996, vol. 6, pp. pp. 565–573.

[60] O. Soysal and E. Sahin, "Probabilistic aggregation strategies in swarm robotics systems," in *IEEE Swarm Intelligence Symposium*. Pasadena, CA: IEEE, 2005.

[61] A. Martinoli, K. Easton, and W. Agassounon, "Modeling swarm robotic systems: a case study in collaborative distributed manipulation," *The International Journal of Robotics Research*, vol. 23, no. (4-5), pp. 415–436, 2004.

[62] H. V. D. Parunak, "Making swarming happen," in *Conference on Swarming and C4ISR*, D. Inbody, C. Chartier, D. DiPippa, and B. McDonald, Eds. Tysons Corner, VA: Joint C4ISR Decision Support Center, 2003.

[63] F. Weiskopf, T. Gion, D. Elkiss, H. Gilreath, J. Bruzek, R. Bamberger, K. Grossman, and J. Wilkerson, "Control of cooperative, autonomous unmanned aerial vehicles," in *AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles*, no. 1. Portsmouth, VA: AIAA, 2002, pp. 1–5.

[64] C. A. Petri, "Communication with automata," Rome Air Development Center, Research and Technology Division, Griffiss Air Force Base, NY, technical report, 1966.

[65] M. Zhu and R. R. Brooks, "Comparison of petri net and finite state machine discrete event control of distributed surveillance network," *International Journal of Distributed Sensor Networks*, vol. 5, no. 5, pp. 480–501, 2009.

[66] F. Palamara, V. a. Ziparo, L. Iocchi, D. Nardi, P. Lima, and H. Costelha, "A robotic soccer passing task using petri net plans," *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, no. Aamas, pp. 1711–1712, 2008.

[67] R. Hamadi and B. Benatallah, "A Petri net-based model for web service composition," in *Proceedings of the 14th Australasian database conference*, no. 4, 2003, vol. 17, pp. 191–200.

[68] M. Rao, S. Ramakrishnan, and C. Dagli, "Modeling and simulation of net centric system of systems using systems modeling language and colored Petri nets: A demonstration using the global earth observation system of systems," *Systems Engineering*, vol. 11, no. 3, pp. 203–220, 2008.

[69] A. H. Levis and L. W. Wagenhals, "C4ISR architectures: I. Developing a process for C4ISR architecture design," *Systems Engineering*, vol. 3, no. 4, pp. 225–247, 2000.

[70] K. Jensen, "Coloured petri nets and the invariant-method," *Theoretical Computer Science*, vol. 14, no. 3, pp. 317–336, 1981.

[71] M. N. Nicolescu and M. J. Matarić, "A hierarchical architecture for behavior-based robots," in *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems Part 1*, 2002, pp. 227–233.

[72] R. A. Brooks, "A robust layered control system for a mobile robot," MIT, Boston, MA, Tech. Rep. 864f, 1985.

[73] L. E. Parker, "ALLIANCE: An architecture for fault tolerant multirobot cooperation," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 2, pp. 220–240, 1998.

[74] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 912–926, 2009.

[75] M. Brambilla, C. Pinciroli, M. Birattari, and M. Dorigo, "Property-driven design for swarm robotics," in *AAMAS International Conference on Autonomous Agents and Multiagent Systems*, no. June. Valencia: International Foundation for Autonomous Agents and Multiagent Systems, 2012, pp. 139–146.

[76] M. Senanayake, I. Senthooran, J. Carlo, and H. Chung, "Search and tracking algorithms for swarms of robots: A survey," *Robotics and Autonomous Systems*, vol. 75, pp. 422–434, 2016.

[77] S. A. DeLoach, M. F. Wood, and C. H. Sparkman, "Multiagent systems engineering," *International Journal of Software Engineering and Knowledge Engineering*, vol. 11, no. 3, pp. 231–258, 2001.

[78] M. A. Day, M. R. Clement, J. D. Russo, D. Davis, and T. H. Chung, "Multi-UAV software systems and simulation architecture," in *International Conference on Unmanned Aircraft Systems*. Denver, CO: IEEE, 2015, pp. 426–435.

[79] D. Cohen, M. Lindvall, and P. Costa, "An introduction to agile methods," *Advances in Computers*, vol. 62, no. C, pp. 1–66, 2004.

[80] S. Kazadi, "On the development of a swarm engineering methodology," in *IEEE International Conference on Systems, Man and Cybernetics*. Waikoloa, HI: IEEE, 2005, vol. 2.

[81] R. Boskovic, Jovan, Prasanth, Ravi, Mehra, "A multi-layer autonomous intelligent control architecture for unmanned aerial vehicles," *JACIC*, vol. 1, no. 12, pp. 605–628, 2004.

[82] A. Kolling, P. Walker, N. Chakraborty, K. Sycara, and M. Lewis, "Human interaction with robot swarms: A survey," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, pp. 9–26, 2016.

[83] S. Easterbrook, R. Lutz, and R. Covington, "Experiences using lightweight formal methods for requirements modeling," *IEEE Transactions on Software Engineering*, vol. 24, no. 1, pp. 4–14, 1998.

[84] S. Agerholm and P. G. Larsen, "A lightweight approach to formal methods," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1641, no. October 1998, pp. 168–183, 1999.

[85] E. M. Clarke, J. M. Wing, R. Alur, R. Cleaveland, D. Dill, A. Emerson, S. Garland, and Others, "Formal methods: state of the art and future directions," *ACM Computing Surveys*, vol. 28, no. 4, pp. 626–643, 1996.

[86] A. Valmari, "The state explosion problem," in *In Lectures on Petri nets I: Basic models*, W. Reisig and G. Rozenberg, Eds. Springer Berlin Heidelberg, 1998, ch. The state, pp. 429–528.

[87] J. Woodcock, P. G. Larsen, J. Bicarregui, and J. Fitzgerald, "Formal methods: practice and experience," *ACM Computer Surveys*, vol. 41, no. 4, pp. 1–40, 2009.

[88] M. B. Revill, "UAV swarm behavior modeling for early exposure of failure modes," M.S. thesis, Dept. of Sys. Eng., NPS, Monterey, CA, 2016.

[89] K. Giammarco and M. Auguston, "Monterey Phoenix demonstration 2016-09-01," Monterey, CA, 2016.

[90] M. Auguston, "System and software architecture and workflow modeling language manual," Monterey, CA, pp. 1–64, 2017.

[91] K. Giammarco, "SYS 650 system architecture and design lab manual," Monterey, pp. 1–135, 2016.

[92] DARPA TTO, "DARPA service academies swarm challenge (SASC) rule book," Arlington, VA, 2017.

[93] DARPA TTO, "Service academies swarm challenge (SASC) operations guide," Arlington, VA, 2017.

[94] J. Elston and E. W. Frew, "Hierarchical distributed control for search and tracking by heterogeneous aerial robot networks," in *Proceedings of the IEEE International Conference on Robotics and Automation*. Pasadena, CA: IEEE, 2008, pp. 170–175.

[95] J. Tisdale, A. Ryan, M. Zennaro, X. Xiao, D. Caveney, S. Rathinam, J. K. Hedrick, and R. Sengupta, "The software architecture of the Berkeley UAV platform," in *Proceedings of the IEEE International Conference on Control Applications*. Munich, Germany: IEEE, 2006, pp. 1420–1425.

[96] P. Almeida, G. M. Gonçalves, and J. B. Sousa, "Multi-UAV platform for integration in mixed-initiative coordinated missions," in *IFAC Proceedings Volumes*, R. Germinet and A. Dolgui, Eds., no. 20. Saint-Etienne, France: Elsevier Ltd, 2006, vol. 39, pp. 70–75.

[97] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Mg, "ROS: an open-source robot operating system," in *ICRA workshop on open source software*, no. 3.2. Kobe, Japan: IEEE, 2009, vol. 3, p. 5.

[98] I. Bekmezci, O. K. Sahingoz, and S. Temel, "Flying ad-hoc networks (FANETs): a survey," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1254–1270, 2013.

[99] J. S. Jang and C. Tomlin, "Design and implementation of a low cost, hierarchical and modular avionics architecture for the DragonFly UAVs," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*. Monterey, CA: AIAA, 2002, pp. 4465–4477.

[100] J. Bachrach, J. Beal, and J. McLurkin, "Composable continuous-space programs for robotic swarms," *Neural Computing and Applications*, vol. 19, no. 6, pp. 825–847, 2010.

[101] S. Sheard, S. Cook, E. Honour, D. Hybertson, J. Krupa, J. Mcever, D. Mckinney, P. Ondrus, A. Ryan, R. Scheurer, J. Singer, and J. Sparber, "A complexity primer for systems engineers," *INCOSE Complex Systems Working Group*, no. July, pp. 1–17, 2015.

[102] R. Parasuraman, S. Galster, and C. Miller, "Human control of multiple robots in the RoboFlag simulation environment," in *IEEE International Conference on Systems, Man and Cybernetics*. Washington, DC: IEEE, 2003, vol. 4, pp. 3232–3237.

[103] B. Browning, J. Bruce, M. Bowling, and M. Veloso, "STP: skills, tactics, and plays for multi-robot control in adversarial environments," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 219, no. 1, pp. 33–52, 2004.

[104] J. D. McLurkin, "Stupid robot tricks: A behavior-based distributed algorithm library for programming swarms of robots," M.S. thesis, Dept. of Elec. Eng. and Cptr. Sci., MIT, Boston, MA, 2004.

[105] D. Karaboga and B. Akay, “A comparative study of artificial bee colony algorithm,” *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 108–132, 2009.

[106] C. W. Reynolds, “Flocks, herds and schools: a distributed behavioral model,” *ACM SIGGRAPH computer graphics*, vol. 21, no. 4, pp. 25–34, 1987.

[107] R. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43, 1995.

[108] Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, “Adaptive particle swarm optimization,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, no. 6, pp. 1362–1381, 2009.

[109] H. Hakli and H. Uğuz, “A novel particle swarm optimization algorithm with Levy flight,” *Applied Soft Computing Journal*, vol. 23, pp. 333–345, 2014.

[110] G. M. Viswanathan, V. Afanasyev, S. V. Buldyrev, S. Havlin, M. G. E. Da Luz, E. P. Raposo, and H. E. Stanley, “Levy flights in random searches,” *Physica A: Statistical Mechanics and its Applications*, vol. 282, no. 1, pp. 1–12, 2000.

[111] T. Stevens and T. H. Chung, “Autonomous search and counter-targeting using Levy search models,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2013, pp. 3953–3960.

[112] W. Ren, R. W. Beard, and E. M. Atkins, “Information consensus in multivehicle cooperative control,” *IEEE Control Systems Magazine*, no. April, pp. 71–82, 2007.

[113] A. S. Jaimes and M. Jamshidi, “Consensus-based and network control of UAVs,” in *5th International Conference on System of Systems Engineering*. Loughborough, United Kingdom: IEEE, 2010, pp. 291–298.

[114] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*, 2nd ed. Cambridge, MA: Cambridge University Press, 2010.

[115] L. E. Parker, “Distributed algorithms for multi-robot observation of multiple moving targets,” *Autonomous Robots*, vol. 12, no. 3, pp. 231–255, 2002.

[116] M. Mitchell, *Complexity: A Guided Tour*. New York: Oxford University Press, 2009.

[117] G. B. Lamont, J. N. Slear, and K. Melendez, “UAV swarm mission planning and routing using multi-objective evolutionary algorithms,” in *Proceedings of the IEEE Symposium on Computational Intelligence in Multicriteria Decision Making*. Honolulu, HI: IEEE, 2007, pp. 10–20.

[118] D. Shilane, J. Martikainen, S. Dudoit, and S. J. Ovaska, "A general framework for statistical performance comparison of evolutionary computation algorithms," *Information Sciences*, vol. 178, no. 14, pp. 2870–2879, 2008.

[119] D. P. Bertsekas, "The auction algorithm for assignment and other network flow problems: a tutorial," *Interfaces*, vol. 20, no. 4, pp. 133–149, 1990.

[120] L. Liu and D. A. Shell, "Optimal market-based multi-robot task allocation via strategic pricing," *Robotics: Science and Systems*, vol. 9, no. 1, pp. 33–40, 2013.

[121] J. J. Tritten, "Naval perspectives for military doctrine development," Naval Doctrine Command, Norfolk, VA, Tech. Rep., 1994.

[122] NATO, "Administrative publication–6 (AAP-6): NATO glossary of terms and definitions," NATO, Brussels, Belgium, Tech. Rep., 2013.

[123] W. P. Hughes, *Fleet Tactics and Coastal Combat*, 2nd ed., U.S. Naval Institute, Ed. Annapolis: U.S. Naval Institute, 2000.

[124] A. S. Hernandez, T. Karimova, and D. H. Nelson, "Mission engineering and analysis: innovations in the military decision making process," in *Proceedings of the American Society for Engineering Management*. Huntsville, AL: American Socitey for Engineering Management, 2017.

[125] S. Tzu, *The Art of War*, T. Cleary, Ed. Boston, MA: Shambhala, 2003.

[126] J. Arquilla and D. Ronfeldt, "In Athena's Camp: preparing for conflict in the information age," Rand Corporation, Santa Monica, CA, Tech. Rep., 1997.

[127] S. J. Edwards, "Swarming and the future of warfare," Ph.D. dissertation, Pardee Rand Graduate School, 2010.

[128] D. Hart and P. Craig-Hart, "Reducing swarming theory to practice for UAV control," in *IEEE Aerospace Conference Proceedings (IEEE Cat. No.04TH8720)*. Big Sky, MT: IEEE, 2004, vol. 5, pp. 3050–3063. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1368111

[129] J. Arquilla and D. Rondfeldt, "Swarming and the future of conflict," *Defense*, 2000.

[130] B. Zweibelson, "Let me tell you about the birds and the bees: Swarm theory and military decision-making," *Canadian Military Journal*, vol. 15, no. 3, 2015.

[131] J. Arquilla, "The new rules of war: How to fight smaller, cheaper, smarter," *Foreign Policy*, no. 178, pp. 60–67, 2010.

[132] P. Scharre, "American strategy and the six phases of grief," pp. 1–4, 2016. Available: warontherocks.com

[133] J. F. Dunford, "Gen. Dunford's remarks and Q&A at the center for strategic and international studies," 2016. Available: http://www.jcs.mil/Media/Speeches/Article/707418/gen-dunfords-remarks-and-qa-at-the-center-for-strategic-and-international-studi/

[134] U.S. Joint Forces Command, "Joint Publication (JP) 5-0 joint operation planning," DoD, Washington, DC, Tech. Rep. August, 2011.

[135] Department of Defense, "Irregular warfare: countering irregular threats joint operating concept," DoD, Washington, DC, Tech. Rep. Version 2.0, 17 May 2010, 2010.

[136] J. Richardson, "A design for maintaining maritime superiority," U.S. Navy, Tech. Rep., 2016.

[137] D. Stefanus, "Embracing the dark battle," *U.S. Naval Institute Proceedings*, vol. 143, no. 4, pp. 27–31, 2017.

[138] T. Brown and S. Doshi, "Test bed for a wireless network on small UAVs," in *AIAA 3rd Unmanned Unlimited Technical Conference*. Chicago, IL: AIAA, 2004, pp. 20–23.

[139] U. o. C. Berkeley Robotics Lab, "BEAR: Berkeley Aerobot Team." Available: https://robotics.eecs.berkeley.edu/bear/

[140] G. T. R. Facility, "Georgia Tech Research Facility UAV." Available: http://www.uavrf.gatech.edu/

[141] S. Chaumette, R. Laplace, C. Mazel, R. Mirault, A. Dunand, Y. Lecoutre, and J. N. Perbet, "CARUS, an operational retasking application for a swarm of autonomous UAVs: first return on experience," in *Proceedings of the IEEE Military Communications Conference MILCOM*. Baltimore, MD: IEEE, 2011, pp. 2003–2010.

[142] C. Daniel, Kai; Wietfeld, "Using public network infrastructures for UAV remote sensing in civilian security operations," *Homeland Security Affairs*, vol. 3, 2011.

[143] D. W. Courtney, "Implementation of secure 6LOWPAN communications for tactical wireless sensor networks," M.S. thesis, Dept. of Elec. Eng. and Cptr. Sci., NPS, Monterey, CA, 2016.

[144] D. R. J. Olsen and S. B. Wood, "Fan-out: measuring human control of multiple robots," in *Proceedings of the SIGCHI conference on Human factors in Computing System*, no. 1. Vienna, Austria: ACM, 2004, vol. 6, pp. 231–238.

[145] B. Hilburn, P. G. Jorna, E. A. Byrne, and R. Parasuraman, "The effect of adaptive air traffic control (ATC) decision aiding on controller mental workload," *Human-automation interaction: Research and practice*, pp. 84–91, 1997.

[146] M. Lewis, "Human interaction with multiple remote robots," *Reviews of Human Factors and Ergonomics*, vol. 9, no. 1, pp. 131–174, 2013.

[147] Department of Defense, "Autonomy community of interest (COI) test and evaluation, verification and validation (TEVV) working group," Office of the Assistant Secretary of Defense For Research & Engineering, Washington, DC, Tech. Rep. May, 2015.

[148] L. G. Shattuck, "Transitioning to autonomy: a human systems integration perspective," Naval Postgraduate School, Monterey, CA, Tech. Rep., 2015.

[149] T. B. Sheridan and W. L. Verplank, "Human and computer control of undersea teleoperators," Massachusetts Institute of Technology Man-Machine Systems Lab, Boston, MA, Tech. Rep., 1978.

[150] R. W. Proud, J. J. Hart, and R. B. Mrozinski, "Methods for determining the level of autonomy to design into a human spaceflight vehicle: A function specific approach," NASA, Johnson Space Center, Houston, TX, Tech. Rep., 2003. Available: http://hdl.handle.net/2060/20100017272

[151] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on Systems, Man, and Cybernetics-Part A*, vol. 30, no. 3, pp. 286–297, 2000.

[152] G. Duke, "Challenges in autonomy: the quest for full autonomy in unmanned systems in the NAS," *The Journal of Air Traffic Control*, no. Winter 2010-11, pp. 46–47, 2010.

[153] Defense Science Board, "The role of autonomy in DoD systems," OUSD(AT&L), Washington, DC, Tech. Rep. July, 2012.

[154] A. Meystel, J. Albus, E. Messina, J. Evans, D. Fogel, and W. Hargrove, "Measuring performance of systems with autonomy: Metrics for intelligence of constructed systems," 2000. Available: https://www.nist.gov/sites/default/files/documents/el/isd/ks/White{_}Paper{_}PerMIS2000.pdf

[155] H.-M. Huang, K. Pavek, B. Novak, J. Albus, and E. Messina, "A framework for autonomy levels for unmanned systems (ALFUS)," in *AUVSI Unmanned Systems North America Conference 2005*. Baltimore, MD: AUVSI, 2005, pp. 1–9.

[156] E. Tunstel, "JHU APL autonomy discussion," Monterey, CA, 2016.

[157] D. M. Rousseau, S. B. Sitkin, R. S. Burt, and C. Camerer, “Not so different after all: a cross discipline view of trust,” *Academy of Management Review*, vol. 23, no. 3, pp. 393–404, 1998.

[158] J. M. Ross, J. L. Szalma, P. A. Hancock, J. S. Barnett, and G. Taylor, “The effect of automation reliability on user automation trust and reliance in a search-and-rescue scenario,” in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, no. 19. New York: Sage Journals, 2008, vol. 52, pp. 1340–1344.

[159] Department of the Army, “Electronic warfare in operations,” Headquarters, Department of the Army, Washington, DC, Tech. Rep. February, 2009.

[160] D. Davis, K. Giles, K. Jones, and M. Jones, “ARSENL field experimentation FX29 after action report,” Naval Postgraduate School, Monterey, CA, Tech. Rep., 2017.

[161] M. Day, “ARSENL swarm vs. swarm ACS autopilot description,” Naval Postgraduate School, Monterey, CA, Tech. Rep., 2015.

[162] M. Rodano and K. Giammarco, “A formal method for evaluation of a modeled system architecture,” *Procedia Computer Science*, vol. 20, pp. 210–215, 2013.

[163] D. R. Wright, “Finite state machines,” pp. 1–28, 2005. Available: http://www4.ncsu.edu/{~}drwrigh3/docs/courses/csc216/fsm-notes.pdf

[164] M. Sipser, *Introduction to the Theory of Computation*, 2nd ed., M. Mendelsohn, Ed. Boston, MA: Thomson Course Technology, 2006.

[165] S. Innovations, “Innoslate,” 2017. Available: www.innoslate.com

[166] S. J. Edwards, *Swarming on the Battlefield: Past, Present and Future*. Santa Monica, CA: Rand Corporation, 2000.

[167] S. G. Hart and L. E. Staveland, “Development of NASA-TLX (Task Load Index): results of empirical and theoretical research,” *Advances in Psychology*, vol. 52, no. C, pp. 139–183, 1988.

[168] ArduPilot Dev Team, “Ardupilot geo-fencing in plane,” 2016. Available: http://ardupilot.org/plane/docs/geofencing.html

[169] A. D. Team, “Plane failsafe function,” 2016. Available: http://ardupilot.org/plane/docs/apms-failsafe-function.html

[170] Office of the Under Secretary of Defense (AT&L), *Systems Engineering Guide for Systems of Systems*, version 1. ed. Washington, DC: Office of the Under Secretary of Defense, AT&L, 2008, no. August. Available: http://www.acq.osd.mil/se/docs/SE-Guide-for-SoS.pdf

[171] G. Dyson, *Darwin Among the Machines: The Evolution of Global Intelligence*. New York: Basic Books, 1998.

[172] DASN (RDT&E), "Mission Area "System of Systems" Systems Engineering Guidebook, draft, 2011."

[173] J. McEver, "A complexity primer for systems engineers," pp. 1–22, 2015.

[174] D. J. Snowden and M. E. Boone, "A leader's framework for decision making," *Harvard Business Review*, vol. 85, no. 11, p. 68, 2007.

[175] INCOSE, "Systems engineering vision 2020," INCOSE, San Diego, CA, Tech. Rep. September, 2007. Available: http://www.incose.org/ProductsPubs/

[176] K. Dooley, "Complex adaptive systems: a nominal definition," *The Chaos Network*, vol. 8, no. 1, pp. 2–3, 1996.

[177] K. J. Dooley, "A complex adaptive systems model of organization change," *Nonlinear Dynamics, Psychology, and Life Sciences*, vol. 1, no. 1, pp. 69–97, 1997.

[178] J. Krogstie, "Evaluating UML using a generic quality framework," *UML and the Unified Process*, pp. 1–22, 2003.

[179] F. Goethals, W. Lemahieu, M. Snoeck, and J. Vandenbulcke, "An overview of enterprise architecture framework deliverables," *SSRN eLibrary*, pp. 1–20, 2006.

[180] R. E. Giachetti, "Evaluation of the DoDAF meta-model's support of systems engineering," *Procedia Computer Science*, vol. 61, pp. 254–260, 2015.

[181] M. Amissah and H. A. H. Handley, "A process for DoDAF based systems architecting," in *IEEE Systems Conference*. Orlando, FL: IEEE, 2016, pp. 1–7.

[182] J. Zachman, "A framework for information systems architecture," *IBM Systems Journal*, vol. 26, no. 3, pp. 276–292, 1987.

[183] L. Urbaczewski and S. Mrdalj, "A comparison of enterprise architecture frameworks," *Issues in Information Systems*, vol. 7, no. 2, pp. 18–23, 2006.

[184] T. O. Group, "The Open Group TOGAF version 9.1," 2016. Available: http://pubs.opengroup.org/architecture/togaf9-doc/arch/

[185] L. Hart. (2015). Introduction to model-based system engineering (MBSE) and SysML. [Online]. Available: https://www.incose.org/docs/default-source/delaware-valley/mbse-overview-incose-30-july-2015.pdf

[186] Y. K. Lopes, S. M. Trenkwalder, A. B. Leal, T. J. Dodd, and R. Groß, "Supervisory control theory applied to swarm robotics," *Swarm Intelligence*, vol. 10, no. 1, pp. 65–97, 2016.

[187] Object Management Group, "OMG systems modeling language (OMG SysML) v.1.4," Object Management Group, Needham, MA, Tech. Rep. September, 2015. Available: http://sysml.org/docs/specs/OMGSysML-v1.4-15-06-03.pdf

[188] B. S. Blanchard and W. J. Fabrycky, *Systems Engineering and Analysis*, 5th ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2010.

[189] K. Hampson, "Technical evaluation of the systems modeling language (SysML)," *Procedia Computer Science*, vol. 44, no. C, pp. 403–412, 2015.

[190] J. A. Estefan, "Survey of model-based systems engineering (MBSE) methodologies," INCOSE, Seattle, WA, Tech. Rep., 2008. Available: http://www.omgsysml.org/MBSE{_}Methodology{_}Survey{_}RevB.pdf

[191] J. Forrester, "Some basic concepts in system dynamics," *Sloan School of Management*, pp. 1–17, 2009.

[192] A. Williams, *Autonomous systems: issues for defense policymakers autonomous systems issues for defense*, A. P. Williams and P. D. Scharre, Eds. Norfolk, VA: NATO Capability Engineering and Innovation Division, Headquarters Supreme Allied Commander Transformation, 2015, no. October.

[193] J. W. Forrester, "Industrial dynamics-a major breakthrough for decision makers," *Harvard Business Review*, vol. 36, no. 4, p. 37, 1958.

[194] J. W. Forrester, "System dynamics, systems thinking, and soft OR," *System Dynamics Review*, vol. 10, no. 2-3, pp. 245–256, 1994.

[195] J. Rushby, "Formal methods and the certification of critical systems," Computer Science Laboratory, SRI International, Menlo Park, Tech. Rep., 1993.

[196] C. B. Jones, "Formal methods light," *ACM Computing Surveys (CSUR)*, vol. 28, no. 4, p. 121, 1996.

[197] J. V. Guttag and J. J. Horning, *Larch: Languages and Tools for Formal Specification*, D. Gries, Ed. New York: Springer Science & Business Media, 2012.

[198] M. J. Spivey, *Understanding Z: A Specification Language and its Formal Semantics*, 3rd ed. Cambridge, England: Cambridge University Press, 1988.

[199] C. A. R. Hoare, *Communicating Sequential Processes*. Englewood Cliffs: Prentice-Hall, 1985.

[200] N. A. Lynch and M. R. Tuttle, "An introduction to input/output automata," *CWI-Quarterly*, vol. 2, no. 3, pp. 219–246, 1989.

[201] Z. Manna and A. Pnueli, *The Temporal Logic of Reactive and Concurrent Systems: Specification*. New York: Springer Science & Business Media, 2012.

[202] M. Auguston, "A program behavior model based on event grammar and its application for debugging," in *2nd International Workshop on Algorithmic and Automated Debugging*, no. May. Saint-Malo, France: AADEBUG, 1995, pp. 1–24.

[203] M. Auguston and C. Whitcomb, "System architecture specification based on behavior models," in *International Command and Control Research and Technology Symposium*, no. 831. Santa Monica, CA: ICCRTS, 2010.

[204] M. Auguston, "Monterey Phoenix, or how to make software architecture executable," in *Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*. Orlando, FL: ACM, 2009, pp. 1031–1038.

[205] M. Auguston, "System and software architecture and workflow modeling language manual," Monterey, CA, pp. 1–44, 2016.

[206] T. H. Chung, "Swarm CONOPS brief," Naval Postgraduate School, Monterey, CA, Tech. Rep., 2015.

[207] K. Whitcomb, Clifford Giammarco and S. Hunt, "An instructional design reference mission for search and rescue operations," Naval Postgraduate School, Monterey, CA, Tech. Rep., 2015.

[208] N. 4.0P, "172004ZAUG15 Zephyr II UAS CAT 3 Interim Flight Clearance for NPS Multi-Air Vehicle Operations with Restricted or Warning Areas," Patuxent River, MD, 2015.

[209] DARPA, "Swarm challenge program: initial planning workshop," Arlington, VA, 2014.

[210] United States Navy, "Naval Doctrine Publication 1," Department of the Navy, Washington, DC, Tech. Rep., 1994.

[211] K. Sandvik and K. Lohne, "The rise of the humanitarian drone: giving content to an emerging concept," *Millennium: Journal of International Studies*, vol. 43, no. 1, pp. 145–164, 2014.

[212] Chairman Joint Chiefs of Staff, "Joint publication 3-29 foreign humanitarian assistance," DoD, Washington, DC, Tech. Rep. January, 2014.

[213] M. O. Eberhard, S. Baldridge, J. Marshall, W. Mooney, and G. J. Rix, "The Mw 7.0 Haiti earthquake of January 12, 2010: USGS / EERI advance reconnaissance team report," U.S. Geological Survey, Reston, VA, Tech. Rep., 2010.

[214] Navy Warfare Development Command, "Foreign humanitarian assistance / disaster relief operations planning (TACMEMO 3-07.6-06)," Department of the Navy, Newport, RI, Tech. Rep. May, 2006.

[215] G. Kovács and K. M. Spens, "Humanitarian logistics in disaster relief operations," *International Journal of Physical Distribution & Logistics Management*, vol. 37, no. 2, pp. 99–114, 2007.

[216] C. M. Greenfield and C. A. Ingram, "An analysis of U.S. Navy humanitarian assistance and disaster relief operations," Naval Postgraduate School, Monterey, CA, Tech. Rep., 2011.

[217] U.S. Agency for International Development, "USAID Haiti earthquake page," 2010.

[218] C. o. N. Operations, "Visit, board, search, and seizure operations (NTTP3-07.11M)," Department of the Navy, Norfolk, VA, Tech. Rep., 2013.

[219] J. S. O. Directorate, "Command and control for joint maritime operations," *Naval War College Review*, vol. 55, no. 4, pp. 39–52, 2013.

[220] W. V. Heinegg, "The legality of maritime interception /interdiction operations within the framework of operation enduring freedom," *International Law Studies*, vol. 79, no. International Law and the War on Terror, 2006.

[221] W. Okon and D. McDaniel, "DoD architectures and systems engineering integration," Washington, DC, 2012.

[222] J. Kline, "A tactical doctrine for distributed lethality," in *Center for International Maritime Security*, D. Filipoff, S. DeBoer, M. Merighi, and J. Stryker, Eds. Annapolis, MD: CIMSEC, 2016.

[223] D. Goure, "The importance of maritime interdiction operations," Arlington, VA, Tech. Rep., 2012. Available: http://proceedings.ndia.org/6100/goure.pdf

[224] Department of Defense, "Universal Naval Task List (UNTL)," DoD, Washington, DC, Tech. Rep. January, 2007.

# Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California