The Application of Graph Theory to Modeling, Simulation, Analysis Looping and Trust to Quantify Mission Success

Robert K. Garrett, Jr.[1], James P. Fairbanks[1], Margaret L. Loper[1], and James D. Moreland, Jr.[2]

1. Georgia Tech Research Institute
2. United States Department of Defense, OUSD(Acquisition & Sustainment)/ASD(Acquisition), Former Executive Director, Mission Engineering and Integration

Abstract

Mission Engineering is the quantification of the effects applied by a system-of-systems (SoS) to achieve measurable desired results. The execution of the mission is defined by a mission thread; that is the sequence of actions/processes executed by elemental systems. Typically, there are many plausible mission threads that can be executed. The domain of complex missions has been described as 'Wicked' because traditional military and space program-based Systems Engineering practices fail due to a lack of discrete phases, a dependence on context, and the non-uniqueness of a 'good-enough' mission thread. Wicked problems also tend to be unstructured with no centralized control and do not lend themselves to linear step-by-step processes. Wicked problems are inherently uncertain leading to the broader issue of trust across a mission knowledge base, and any mission level analyses. The nature of a complex mission will require an iterative approach resulting in a continuous reduction in uncertainty, an increase in trust, and refinement in the topology of the mission thread. The approach described in this paper is based upon Applied Category Theory (ACT) as a universal representation of mathematical knowledge; OODA-based decomposition of mission threads focused on Boyd's ORIENT function; and a Trust metric to provide decision makers confidence in the results.

Executive Summary

Mission Engineering is the quantification of the effects applied by a system-of-systems (SoS) to achieve measurable desired results. A mission can be constrained by or benefit from dynamic contexts. The Environment is defined as a subset of context and only accounts for one of many conditions that both help define and can alter mission execution. The execution of the mission is defined by a mission thread; that is the sequence of actions/processes executed by elemental systems. Typically, there are many plausible mission threads that can be executed. Mission threads can be decomposed into event chains. An event chain is a 'short' sequence of events with a quantifiable outcome.

Traditional system evaluation modeling and simulation (M&S) tools struggle to characterize performance at the mission level. Typically, these tools are structured, discrete-event simulations that are successfully used at the system level, but fail for many reasons when applied to a SoS within a mission context. An alternative is to move away from a single M&S approach that attempts to provide a true/false answer for mission success. Instead, embrace a toolbox of structured and unstructured M&S approaches which provide various quantitative perspectives on acceptable mission success criteria, thus providing richer insight into the mission context.

Missions can be complex if they are constructed with many moving parts that originate from sub-optimized business rules. From the combinatorics, many plausible mission threads are possible for the execution of a given mission. Missions are graphical in nature, where each node and edge in a mission graph contains both structured and unstructured information to include metadata, functional behaviors, and empirical and virtual data. Missions can be highly complex, and the datastore will be extensive with the number of nodes anticipated to exceed a 10^{+9}.

The domain of highly complex missions has been described as 'Wicked' because traditional military and space program-based Systems Engineering practices fail due to a lack of discrete phases, a dependence on context, and the non-uniqueness of a 'good-enough' mission thread. Wicked problems also tend to be unstructured with no centralized control, or simple hierarchical structure, and do not lend themselves to linear step-by-step processes. Wicked problems are inherently uncertain due to both complexity and the dynamic nature of context. This uncertainty leads to the broader issue of trust across a mission knowledge base, and any mission level analyses.

The performance of SoS to meet a mission goal is highly determined by the interstitials (the relationships between systems), the SoS themselves, and the operational context. The dynamic interplay between the SoS and the environment preclude a single best answer, although there may be many good enough answers. In other words, there are most likely multiple plausible mission threads to achieve a given mission success measure.

Mission analyses should be conducted on a representative set of bounding mission threads, and uncertainty in the analyses must be quantified. Trust in all aspects of the mission must be measured as a basis for decisions. The nature of a complex mission will require an iterative approach resulting in a continuous reduction in uncertainty, an increase in trust, and refinement in the topology of the mission thread and their constituent event chains.

Trustworthy predictions of mission effects require a multi-faceted and layered approach. The authors propose:

Introduction

Mission Engineering is the quantification of the effects applied by a system-of-systems (SoS) to achieve measurable desired effects. A mission can be constrained or benefited by dynamic contexts. The *Environment* is defined as a subset of context and only accounts for one of many conditions that both help define and can alter mission execution. The execution of the mission is defined by a mission thread; that is the sequence of actions/processes executed by elemental systems. Typically, there are many plausible mission threads that can be executed. Mission threads can be decomposed into event chains. An event chain is a 'short' sequence of events with a quantifiable outcome.

Missions can be mathematically characterized utilizing graph-based methodologies. A mission graph is composed of systems, the environment, policy and doctrine, and connecting relationships. The systems are represented by the nodes in the graph. System behaviors tend to be governed by mathematical laws or equations (e.g. Newtonian mechanics and Maxwell's equations). The edges of the mission graph define relationships and interactions between the nodes. These relationships are quantifiable behaviors and can be utilized to pre-determine mission success criteria. There can be many edges between two nodes. Mission threads are critical paths through the graph. The success of the mission is dominated by the inter-system relationships, or interstitial space [Garrett, 2011], that space between structure or matter where integration resides.

Traditional system evaluation modeling and simulation (M&S) tools struggle to characterize performance at the mission level. Typically, these tools are structured, discrete-event simulations that are successfully used at the system level, but fail for many reasons when applied to a SoS within a mission context [Riox, 2002] [Henriksen, 2008]. Kinder has proposed moving away from a single M&S approach that attempts to provide a true/false answer for mission success. To evaluate a SoS he embraces a toolbox of structured and unstructured M&S approaches providing various quantitative perspectives on acceptable mission success criteria [Kinder, 2014] thus providing richer insight into the mission context.

Missions can be complex if they are constructed with many moving parts that originate from sub-optimized business rules. While complexity is frequently used in describing SoS, it is rarely quantified [Ladyman, 2013]. For the purposes of Mission Engineering, complexity is a measure of extent defined as an ordered triple, (n, e, p), where n is the number of nodes in the mission graph, e is the number of edges in the graph, and p is the number of defined paths, or mission threads. In the mission graph the number of edges can approach n^2 (n squared) and the maximum number of paths can approach n! (n factorial) [Guichard, 2017]. From the combinatorics, many plausible mission threads are possible for the execution of a given mission. Each node and edge in a mission graph contains both structured and unstructured information to include metadata, functional behaviors, and empirical and virtual data, i.e., Live, Virtual and Constructive [Urias, 2012]. Due to combinatorics, mission graphs are expected to be highly complex and the datastore will be extensive. Graph databases are mature technology with demonstrated scalability and extensibility ideally suited to store these extensive quantities and diversity of anticipated mission data sets. Graph visualization and query tools are readily available and compatible with the databases [Besta, 2019].

As mission knowledge increases the associated mission-graph datastore will grow significantly in complexity with the number of nodes anticipated to exceed a 10^{+9}. The domain of highly complex missions has been described as 'Wicked' because traditional military and space program-based Systems Engineering practices fail due to a lack of discrete phases, a dependence on context, and the non-uniqueness of a 'good-enough' mission thread. Wicked problems also tend to be unstructured with no centralized control, or simple hierarchical structure, and do not lend themselves to linear step-by-step processes [Rittel, 1973]. Wicked problems are inherently uncertain due to both complexity and the dynamic nature of context. This uncertainty leads to the broader issue of trust across a mission knowledge base, and any mission level analyses [Liu, 2016] [Loper, 2019].

The performance of SoS to meet a mission goal is determined by the interstitials (the relationships between systems), the SoS themselves, and the operational context. The dynamic interplay between the SoS and the environment preclude a single best answer, although there may be many good enough answers. In other words, there are most likely multiple plausible mission threads to achieve a given mission success measure. Mission analyses should be conducted on a representative set of bounding mission threads, and uncertainty in the analyses must be quantified. Trust in all aspects of the mission must be measured as a basis for decisions. The nature of a complex mission will require an iterative approach resulting in a continuous reduction in uncertainty, an increase in trust, and refinement in the topology of the mission thread and their constituent event chains.

This paper proposes a multi-faceted, inherently unstructured and iterative approach of modeling, simulation and analysis looping (MSAL) to better assess mission success quantitatively. [Loper and Garrett, 2015]. Through realistic examples, the creation of a multi-layered, multi-dimensional mission model, and event chains will be demonstrated. The starting point will be a United States Department of Defense (US DoD) proposed Mission Engineering and Integration process. This process will then be rigorously expanded and applied to a generic mission within a surrogate example utilizing a city neighborhood involving people transiting about the neighborhood and participating in the functions of education and work, with an underlying behavior of cheating. The focus of the effort is on the establishment of a rigorous mission schema based on Applied Category Theory along with an approach to mission functional decomposition based on the Observe, Orient, Decide, Act (OODA) Loop [Boyd, 1987]. This demonstration creates the necessary graphical basis and the mathematical foundation to which MSAL can be applied. Finally, an overarching approach to quantify trust will be presented.

Applied Category Theory

While knowledge about a SoS can be stored and manipulated as a knowledge graph, this formulation cannot capture the mathematical nature of the physical interactions. In wicked problems, we find each sub-domain governed by different mathematical principles. For example, in the electromagnetic domain of radio communication, the system is governed by Maxwell's equations and solving equations for phases and amplitudes, but these radio communications are carrying messages on a social network, which is governed by stochastic processes over a discrete communication graph. In order to integrate these vast differences in fundamental dynamics, we must store information in the unified theory of mathematics provided by Applied Category Theory (ACT).

Our software approach[1] implements a rapidly developing field of mathematics called ACT, which understands physical and computational systems through the lens of *categories* [Halter et. al, 2019]. A category is a mathematical structure, built from *objects* (things) and *morphisms* (relationships between things), where the structure comes from *composition* of morphisms. The traditional presentation of mathematics centers around Set Theory, where the objects are *sets* and the morphisms are *functions* with the traditional definition of function composition. Almost any mathematical object can be viewed as a category, for example a graph is a category with *vertices* as its objects and *paths* as its morphisms. In a graph, you compose paths by concatenating them head to tail. Chemical, Biological, and Ecological systems can be viewed as categories with species as the objects and reactions as the morphisms. To reactions f, g can be composed if the products of f are the reagents of g. Processes in Systems Engineering can be modeled as a category, for example a *co-design* can be modeled as a category where the objects are resources and the components that provide input resources, produce output resources [Censi, 2017]. ACT seeks universal representations of mathematical knowledge that transcend domains and disciplines. The ACT approach is inherently computational and universal, which makes it an ideal framework for studying Mission Engineering and Integration.

[1] SemanticModels.jl is developed on GitHub at https://github.com/jpfairbanks/SemanticModels.jl with documentation hosted at https://aske.gtri.gatech.edu/docs/latest/.

By taking the ACT perspective, we can build mathematical and computational tools for analyzing systems across diverse domains. The unified framework of categories allows for representing different mathematical frameworks as examples of a common algebraic structure. This unification of heterogenous modeling frameworks allows us to build tools that are specialized enough to exploit structured knowledge about the application area, but general enough to write software against a common interface. One example of where this approach can shine is the modeling of mission threads as a graph. While existing graph based techniques treat the edges as the primary structure and build hierarchical representations of systems for either understanding or computational efficiency, the ACT approach takes the hierarchical design of the network as primary and deals directly with the consequences of that hierarchy.

Once a system is modeled as a graph, graph analytic techniques such as pathfinding, centrality, and community detection can be used to analyze the system. Pathfinding techniques are used to explore paths through a graph. An example of pathfinding is Google Maps where several of the shortest routes between two points are calculated in terms of distance and time. In the mission model the technique could be used to explore alternate mission threads or event chains. Centrality is used to explore the role of nodes in the mission graph. Centrality provides a metric of connectedness. It also entails finding nodes that have significant control or influence, these could be vulnerable choke points in communications or decision making. Community detection algorithms are based on finding relations and behaviors within the group. These groups could be a structure of resiliency or subsequent failure. The algorithms used for these techniques are mature and available as open source tools e.g., Apache Spark with GraphX[2]. These techniques will provide insight enabling changes to the topology of the mission model and/or mission threads [Fairbanks et. al, 2015]. However, the ACT perspective opens up a whole new set of tools for analyzing systems such as comparison of networks with metadata via optimal transport [Patterson 2019].

Mission Engineering is an inherently multi-domain problem where the dynamics of the problem appear too complex to be mathematically modeled. However, this is true when trying to identify a single set of mathematical rules for modeling all aspects of the mission. When you separate the mission into each domain and model them separately, the mission engineering process is amenable to mathematical analysis. However, since the rules for different domains are diverse, traditional simulation software development techniques fail to give a unified treatment of the system, which is essential for building large scale software for accurately modeling a complex mission. It is only through the ACT paradigm that we can see how these different mathematical modeling frameworks are examples of categories with various axioms. We can then build software that works with explicit representations of the axioms to build a unified software ecosystem for mathematical modeling and computer simulation of complex multi-domain missions. Building a model is insufficient for mission engineering in wicked problems, the models you build must be used to reason about the world and make decisions. This decision making process requires that the models be tractable either analytically, or numerically. The ACT perspective gives you a framework for analyzing systems with symbolic algebra, with an easy transition to numerical analysis when there is no analytic solution, which is usually the case for wicked problems.

The DoD Mission Engineering and Integration Process

The US DoD is working toward the establishment of a Mission Engineering discipline. DoD has defined a 10-step mission-based process for Mission Engineering and Integration (MEI) through the Office of the

[2] https://spark.apache.org/, https://spark.apache.org/graphx/

Undersecretary of Defense for Acquisition and Sustainment Mission Engineering and Integration Guidebook [DoD, 2020]. This paper will start with the DoD example, but our work is not limited to military applications. The MEI steps are:

(1) Identify the missions and tasks.
(2) Define mission success and desired effect.
(3) Identify mission success factors.
(4) Identify conditions for each mission success factor.
(5) Map mission success conditions to mission tasks.
(6) Identify critical conditions for each mission task.
(7) Map systems into mission tasks.
(8) Define appropriate scoring criteria for each mission task.
(9) Apply the scoring criteria.
(10) Manage the assigned mission areas.

The process begins as a language-based analysis with the collection of mission information. This data is retrieved from doctrine and policy, and includes an initial mission thread(s) defining context and the operational mission environment. The next step in the process is an event decomposition of the mission thread, to create a series of discrete, sequenced *effects/kill chains* (these are referred to as event chains throughout the paper) which are composed of a set of tasks (e.g., track threat, detect hostile intent, neutralize threat). The event chains are represented as a path in a graph, where each node/system in the path is subjectively ranked as red/yellow/green based on defined success criteria. In this analysis the edges are only implicitly addressed. As the analysis proceeds, tactical systems replace conceptual nodes in the chains. Effects/kill chains are then reconfigured to maximize *green* capability. Based on these analyses, the mission threads are reconfigured and the analysis repeated as goals evolve. Context, and the environment are at best implicit, and no mission graph with supporting metadata are created. The Guidebook uses a simplistic, notional Air Warfare example to demonstrate the process. The scoring criteria for this example are shown in Table 1 and the scored event chain, using nodes, edges, paths (n, e, p), is shown as a string diagram in Figure 1.

Table 1. Evaluation criteria used for scoring the notional Air Warfare example

Mission Task	Critical Conditions
Track	• Can the platform provide a weapon quality track? • Is the required range to each target in a group provided? • Does the platform track targets sufficiently when it has less than complete mission data?
Identification: Commit	• Is the potential target classified sufficiently well to commit resources to that area?
Identification: Engage	• Are all Blue and Red forces correctly identified? • Is a correct decision (engage Red and do not engage Blue) made on each track? • For each track to be engaged, is an engagement order issued? • Is an appropriate/preferred weapon ordered for the engagement? • Does the platform detect targets of interest at a range to support or exceed the desired weapons employment or enhance decision time?

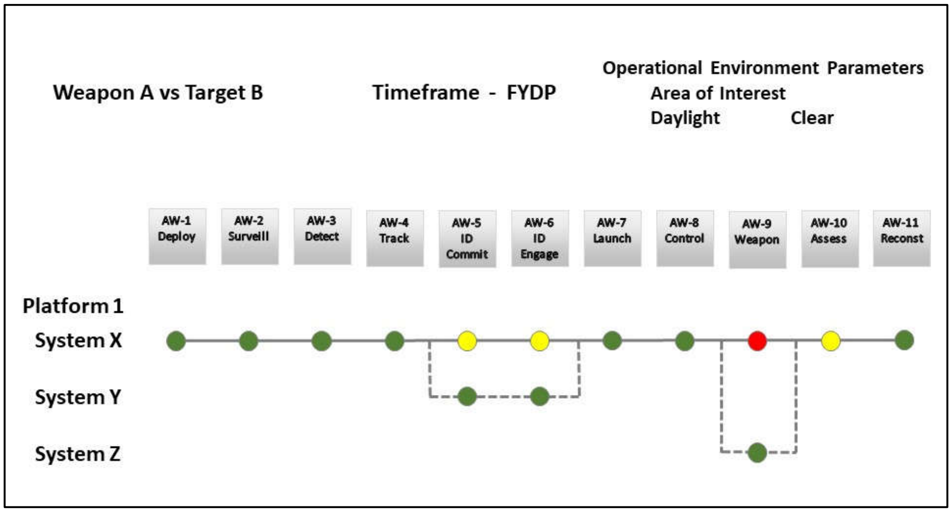

Figure 1. Scored event chain applied to a SoS consisting of three systems has complexity (14, 15 ,4)

Functional Decomposition, OODA and the Orient Function

Many military-based event chains are deliberately system-centric in their construction to fit within a required System Engineering process. Evaluation of communications and messaging beyond their system boundaries (particularly in virtual and constructive testing) are usually only implicitly considered. System level event chains also tend to be system unique in lexicon with a detailed level of abstraction that is unnecessary for mission analysis. This provides new challenges when aggregating diverse systems into a mission-based SoS. A SoS/mission event framework with an appropriate level of abstraction that explicitly represents communication across the SoS and provides for contextual awareness is desirable.

In the Air Warfare example in Figure 1, the functions of information technology (e.g., communications, information technology, data analytics, artificial intelligence) are represented as edges in the graph. These edges are the interstitials [Garrett, 2011], which are the domain of integration and interoperability across the mission. The interstitials thus play a dominate role in mission success and need to be explicitly represented in the event chain. The process to address the interstitials are based on John Boyd's OODA loop, shown in Figure 2 [Boyd, 1987].

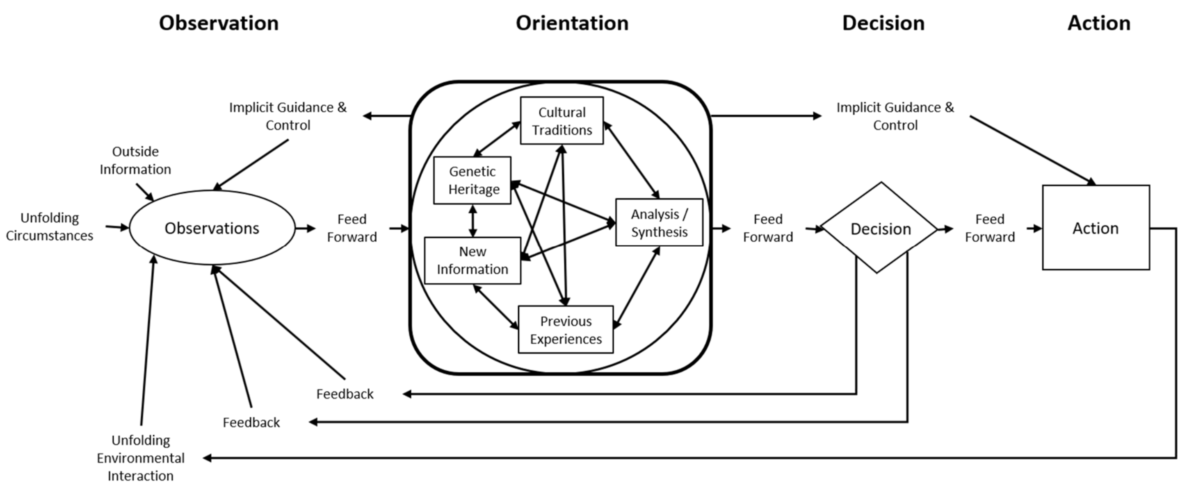

Figure 2. John Boyd's OODA Loop

Where Observe, Decide and Act are system functions, Boyd defined Orient as a multi-faceted and iterative hub between them. It is Orient that is suited to represent the interstitial functions. Boyd defined five sub-functions to Orient, which provide contextual awareness; external communications were not considered.

- *New Information*, *Previous Experience* and *Analysis/Synthesis* are straight forward involving data processing and extend readily to mission engineering.
- *Genetic Heritage* and *Cultural Conditions* involve inference, and are a means of assessing the local environment including social context within the mission environment.

These last two Orient sub-functions reduce uncertainty, and enable better decisions and informed actions [Boyd, 1976]. A sixth sub-function, communication, is added to explicitly address messaging across the SoS. The communication sub-function is more than having the means to communicate (e.g., the pipe); it includes what flows on the pipe (syntax and semantics, quality, trustworthiness, timeliness) and the unique needs of the two systems (nodes) that are connected by the pipe (edge). Thus, communication is part of the Interstitial Space, a foundational characteristic between every system/sub-system in the mission. The mapping from the Boyd fighter pilot perspective to a SoS mission perspective is shown in Table 2.

Table 2. Translation of Boyd's Orient function to a suitable SoS construct for Mission Engineering

Boyd 1987	**Mission Engineering and Integration 2020**
Cultural Traditions	Parse the physical and natural environment relevant to the moment
Genetic Heritage	Parse the human environment relevant to the moment
New Information	Parse and analyze the OBSERVE/Sensor network data, and update the knowledgebase

Previous Experience	Mine the historical data, LVC, etc. – update the knowledgebase
Analysis/Synthesis	Re-calculate ‘real-time’ ability to meet the mission goal and effects chain goal(s)
Communication	Communicate

Figure 3 maps the Air Warfare events from Figure 1 to OODA functions. Interestingly there are no ORIENT steps to enable data flows and communication. To create an event chain that represents the networked SoS, these OODA functions are then explicitly interspersed with Orient functions. Figures 4 shows plausible Observe and Decide event chains, with the Orient function explicitly represented where Orient is one of the first four sub-functions in Table 2. In these event chains, the *communication sub-function* is represented as a directed edge. The loops about the Orient functions represent iterative processing and can add significant complexity. These graphs are not unique solutions to the event chain but represent plausible paths within a mission thread. Presenting event chains and mission threads as OODA-based graphs will set the stage for subsequent quantitative analyses.

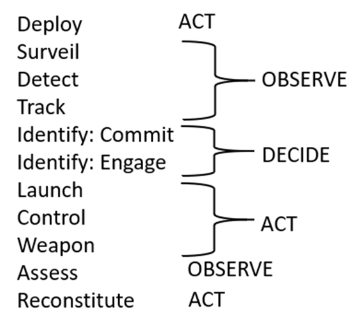

Figure 3. Air Warfare kill/event chain mapped to the OODA functions

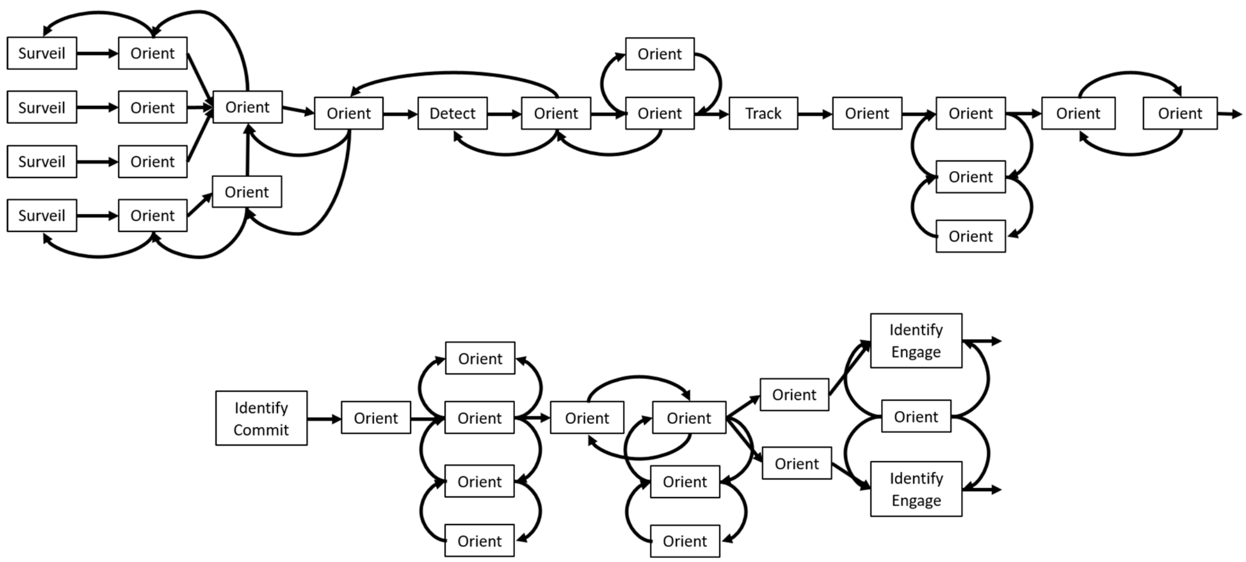

Figure 4. A plausible event chain for the Air Warfare example with complexity (30, 50, ∞) given the looping in sensor network configurations

Interrogating the Mission with Modeling, Simulation, Analysis Looping

MSAL is an iterative approach suited to provide a variety of perspectives for quantitatively evaluating the ability of an SoS to meet mission goals [Loper and Garrett, 2015]. MSAL is created and executed upon a graphical mission model and an initial set of event chains (e.g., Figures 1 and 4) and mission threads or paths, that traverse the model. As discussed earlier, traditional system modeling languages are not appropriate for a SoS evaluating a complex mission. The mission model should instead be built using ACT-based mathematical specifications. The mission model is a basis of a graph database where each node and edge in the model is the basis of independent data stores containing metadata, structured and unstructured data, and essentially extensible sub-databases. Multiple associations, the edges between nodes in the databases, can be made arbitrarily within the ACT-based specification. Graph databases have demonstrated scalability, extensibility, and a robust open source toolset that includes visualization, query languages, and graph analytics. They are also readily integrated with machine learning and artificial intelligence tools. The other advantage of a graph database is that a simulation can be built upon the mission model, i.e., the database is a flexible, reconfigurable simulation framework providing a consistent interface standard. The mission model consists of the following information:

1. Environment data,
2. Relevant systems and resources needed to execute the mission,
3. Context including constraining policy and doctrine,
4. Mission goals,
5. Compilable behaviors, node or edge, to support modeling and/or simulation, which could include conditional probability tables and supporting data, and
6. Live, Virtual and Constructive data, real-time or historical.

Once the initial event chains and mission threads have been created, MSAL can interrogate event chains by Bayesian Inference when experimental data is available. The first step of this process is to establish the event chains as paths through the mission graph consistent with mission goals. The second step is to perform functional decomposition and merge the event chain graphs establishing mission processes. The OODA loop provides a means for such a functional decomposition. The event process can then be converted from a cyclic multidigraph into a unique, directed acyclic graph to enable the use of Bayesian techniques, e.g., Markov Chains. This process will likely create several directed acyclic graphs, i.e., a parallel/sequential process, for each mission thread. The third step is to collect and analyze existing data sets and if possible establish statistical distributions about each mission process. From these analyses the probability of event success, $P_{A|B,C,\ldots}^{Success}$, may be inferred using Bayesian techniques. If the probability of event success is unacceptably low or there is insufficient data, then experimentation will be used to change mission thread topology.

Experimentation about the event chains can be conducted through the use of Uncertainty Quantification (UQ) coupled with agent-based simulation run on scenarios. Scenarios are instances of a mission thread. The use of simulation explicitly introduces a temporal component to the scenario. Agent-based simulations are a key tool to evaluate mission success when inter-entity relationships have similar or greater importance than the performance of individual entities. Agent-based techniques readily deal with networks and inter-agent interactions to include human social factors, non-linearity in agent behavior and/or coupling, or the absence of explicit mathematical solutions. Further, due to their fast running nature, hundreds to thousands of runs can be realized per day [Bonabeau, 2002]. The Dempster-Shafer-based UQ engine will propagate both epistemic and aleatoric uncertainties through iterative simulation producing bounds about a multi-dimensional, optimized performance surface [Dempster, 1968] [Shafer, 1976]. Any detail within the uncertainty bounds is but high frequency noise. The intent is not to simulate the totality of a mission, but to provide insight where needed within the mission thread due to a lack of empirical data, or a change in context. This simulation-based evidence could lead to changes in mission thread topology, inter-system relationships/behaviors, or populate conditional probability tables. The optimal solution is not desired, instead the goal is to minimize uncertainty while maintaining acceptable performance. The authors believe the characteristics of the uncertainty bounds are more important to decision makers than the absolute nature of an optimized performance prediction. The use of UQ in simulation is mature technology; an example is DAKOTA from Sandia National Laboratories[3], an open source tool for driving simulations with iterative analysis method containing Dempster-Shafer theory of evidence capability.

Various components of MSAL have been demonstrated; Mabrok [2017] used ACT to create a robust foundation for Model-Based Systems Engineering focusing on the relationships between components explicitly including requirements. ACT has been used to rigorously couple agent-based simulation with Bayesian inference to create hybrid methods consistent with MSAL [Beheshti, 2013]. Henkel [2015] has applied graph database techniques to enable robust queries and data analyses of models, simulation results, and metadata in Biology. Key to the approach was maintaining relationships between the models, simulations, and metadata to facilitate reuse. GraphPool [Lange, 2016] is an application built upon a graph datastore to enable management of substantial data generation from concurrent simulation. Vehicle to vehicle information flow in a dynamic multi-layered traffic network was modeled and simulated upon a graph-based framework [Kim, 2016]. This emerging work gives confidence to build the multi-faceted MSAL.

[3] https://dakota.sandia.gov/, page 53 of the theory manual

Using an ACT basis for integration enables the rigorous implementation, data management and extensibility of software. ACT provides for a mathematical foundation used to build the multi-layered mission graph. This foundation can be viewed as graphical maps defining the functional structure and sub-structures of objects and morphisms. The graph then is both the model and the data schema upon which the MSAL process is built. Mission thread construction begins using ACT to create string diagrams [Jacobs, 2019] facilitating the rigorous and consistent execution of subsequent analyses. Categories and functors become the mathematical architecture of agent-based simulation as well as managing the data from thousands of iterative simulation runs and the associated UQ analysis.

A Graphical Example

These graph-based concepts will be demonstrated on a city model representing a mission environment upon which mission threads can be placed and analyses conducted. The city is based on the *City Anatomy Framework*; *City Anatomy provides a hierarchically sound and well-established description, identification, nomenclature, and classification of all city systems, subsystems and interactions…*" [Agreement, 2015]. The framework is language-based and, at the highest level, is composed of three interacting layers: the city *Structure*, *Interactions* and *Society* as shown in Figure 5. A graphical instance of the framework is shown in Figure 6, containing 68 nodes and 211 edges. This graph is math-based and testable. In this graph the nodes represent the environment (green), the infrastructure (yellow), and the built domain (blue). The edges of the graph represent flows (e.g. electricity, water, etc.), interactions, or hierarchical relationships (i.e., 'subsets of'). Specific details of the city like roads and buildings would be substructures embedded within the graph.

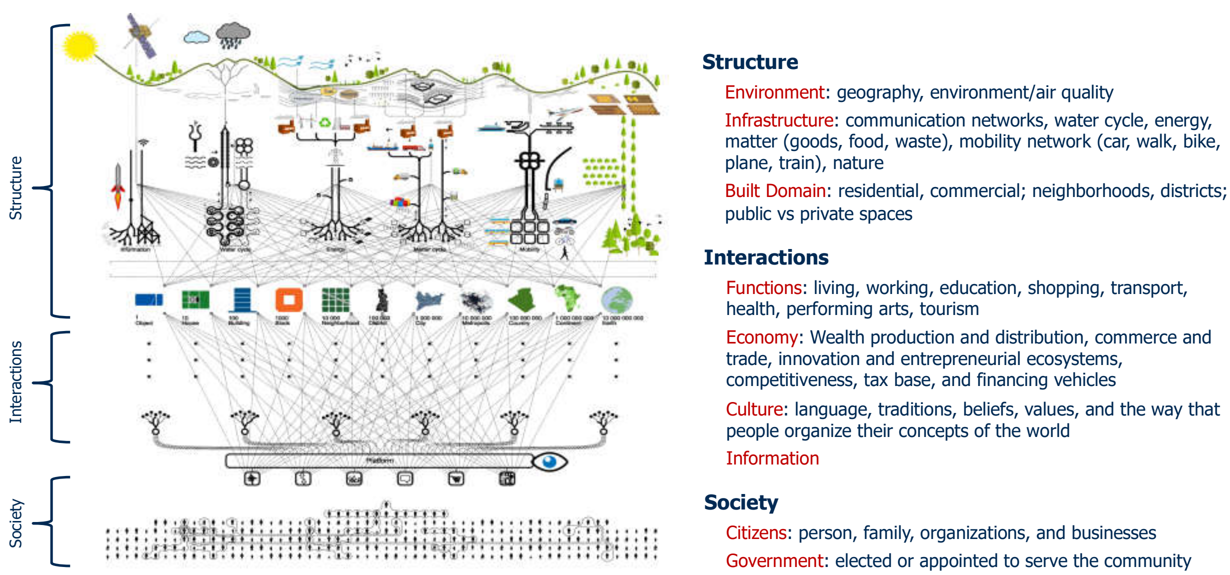

Figure 5. The city anatomy framework, as basis for modeling a city

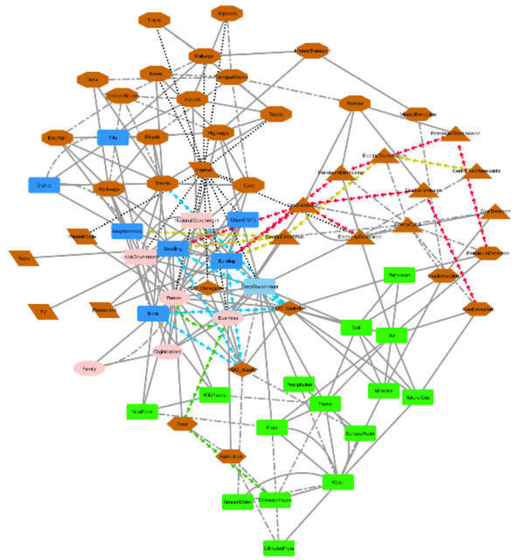

Figure 6. A graphical instance [Shannon, 2003] of the city anatomy framework containing all the information in Figure 5

A notional mission involving college students on the Georgia Institute of Technology (GaTech) campus will be used as an example for creating event chains; the first step in implementing MSAL. The example will be static and lead to the creation of an OODA-based event chain. The mission graph will be multilayered and consistent with the City Anatomy framework.

Mission Model

The mission model for this example is a subset of Figure 6. The mission environment is Atlanta Georgia, specifically the GaTech neighborhood. The details of Atlanta and GaTech add additional layers within the model. For reference of complexity and consistency with the City Anatomy taxonomy, this example will explicitly represent:

- Structure layer components: *Built Domain* - Neighborhood, Buildings and Dwellings; *Infrastructure* - *Mobility Network* comprised of Streets, Bike Paths and Walkways, and *Communication Network.*
- Society layer components: *Citizens* – students and professors at GaTech.
- Interaction layer components: *Functions* - Working, Education and Transport.

Atlanta is divided into 12 Districts each with a City Council member[4], and GaTech is in District 3. Atlanta has 242 neighborhoods; one of which is GaTech[5]. The neighborhood population is about 8000 with approximately 90% being college aged. There are around 970 dwellings where nearly 90% are non-family households. Figure 7 is a map of the GaTech campus from Open Street map[6], along with a close up view of the center of campus. This map is composed of 45 nodes and 592 edges along with extensive metadata for each. This is significant complexity and parsing the data is necessary for subsequent analyses.

[4] http://cbatl.org/atlanta-city-district-maps/ and https://citycouncil.atlantaga.gov/home
[5] http://documents.atlantaregional.com/NN/Profiles/AtlantaProfiles/E02.pdf
[6] https://www.openstreetmap.org/?edit_help=1#map=16/33.7757/-84.3999

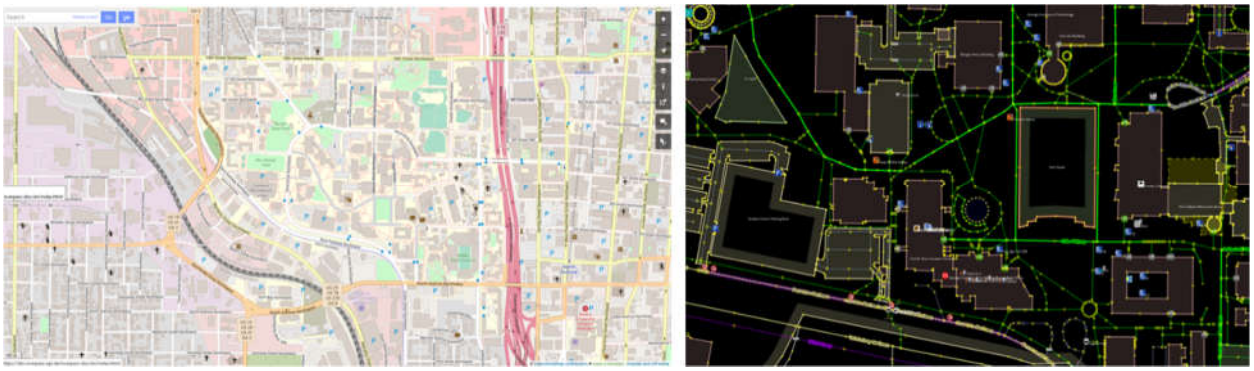

Figure 7. Images of the GaTech campus from Open Street Map. The image on the right shows the detail in the dataset consisting of the node and edges representing buildings, streets, walkways, etc.

Mission Threads

The mission is defined as groups of students from a class working to solve a homework assignment. One of the groups decides to cheat by plagiarizing from classmates, with their mission goal being to fool the professor that their work is original. ACT will be used to rigorously define the mission schema, a precursor to the mission thread. Figure 8 shows three representations of the mission schema [Joyal-Street, 1991]. The top left representation is programming syntax that could be used by a subject matter expert. The bottom view is the String diagram that is easy to visualize and understand. The top right representation is an algebraic expression that would be easy to manipulate with algorithms. Each string is an entity type, and each box is a process that matches entity production with requirements. The input code used to generate the diagram and the corresponding algebraic expressions are included as well. This String diagram also necessarily defines a database schema, and can be used to construct a mission simulator.

```
d = @program Transit (s1::Team, s2::Team, p::Prof)
    s1, a1 = class(s1,p)
    s2, a2 = class(s2,p)
    s1 = north(s1)
    sol1 = solve(s1, a1)
    s1 = east(s1)
    s2 = east(north(s2))
    sol2 = plagiarize(s2, sol1)
    s1 = west(s1)
    g1 = gradea(sol1, p)
    g2 = gradea(sol2, p)
    return g1, g2
end

display(d)
```

a)

$$
\begin{aligned}
&(id_{Team} \otimes id_{Team} \otimes (\Delta_{Prof} \cdot ((\Delta_{Prof} \cdot (\Delta_{Prof} \otimes id_{Prof})) \otimes id_{Prof}))) \\
&\cdot (id_{Team} \otimes \sigma_{\mathrm{Team,Prof}} \otimes \mathrm{id}_{\mathrm{Prof \otimes Prof \otimes Prof}}) \cdot (\mathrm{class} \otimes \mathrm{class} \otimes \mathrm{id}_{\mathrm{Prof}} \otimes \mathrm{id}_{\mathrm{Prof}}) \\
&\cdot ((\mathrm{north} \cdot \Delta_{\mathrm{Team}}) \otimes \mathrm{id}_{\mathrm{Assgn}} \otimes (\mathrm{north} \cdot \mathrm{east}) \otimes \lozenge_{\mathrm{Assgn}} \otimes \mathrm{id}_{\mathrm{Prof}} \otimes \mathrm{id}_{\mathrm{Prof}}) \\
&\cdot (\mathrm{id}_{\mathrm{Team}} \otimes \sigma_{\mathrm{Team,Assgn}} \otimes \mathrm{id}_{\mathrm{Team \otimes Prof \otimes Prof}}) \\
&\cdot ((f \cdot g \cdot (\mathrm{id}_{\mathrm{Assgn}} \otimes \sigma_{\mathrm{Assgn,Team}}) \cdot (\mathrm{id}_{\mathrm{Assgn}} \otimes \mathrm{plagiarize})) \otimes \mathrm{id}_{\mathrm{Prof}} \otimes \mathrm{id}_{\mathrm{Prof}}) \\
&\cdot (\mathrm{id}_{\mathrm{Assgn}} \otimes \sigma_{\mathrm{Assgn,Prof}} \otimes \mathrm{id}_{\mathrm{Prof}}) \cdot (\mathrm{gradea} \otimes \mathrm{gradea})
\end{aligned}
$$

where $f = ((\mathrm{solve} \cdot \Delta_{\mathrm{Assgn}}) \otimes (\mathrm{east} \cdot \mathrm{west} \cdot \lozenge_{\mathrm{Team}}) \otimes \mathrm{id}_{\mathrm{Team}})$ and $g = (\sigma_{\mathrm{Assgn,Assgn}} \otimes \mathrm{id}_{\mathrm{Team}})$ b)

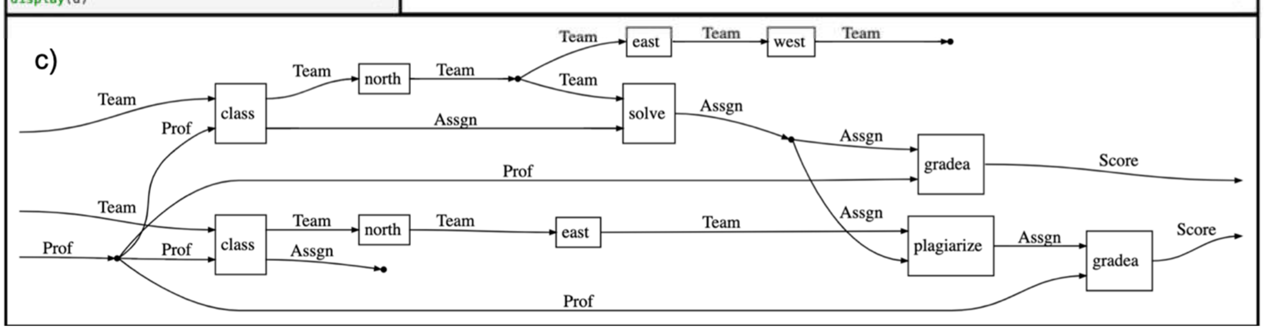

Figure 8. An example mission thread of students plagiarizing an assignment in a college class. The mission thread is show in three representations of a schema-level mission thread; a) a programing syntax that is easy to write, b) a formula that is easy to computationally manipulate, and c) a string diagram that is easy to visually interpret and understand.

The next step is to explicitly define mission resources and relations. The resources (top left view of Figure 9) are the students, **Team**, taking a class together**;** the professor, **Prof,** teaching them; and the homework assignment, **Assgn.** The product of the mission is the grade the professor gives the assignment, **Score**. There are two subsets of **Team**, $\mathbf{S_1}$ and $\mathbf{S_2}$. The relations (right view of Figure 9) are **class**, **north**, **east**, **solve**, **west**, **gradea**, and **plagiarize**. **Class** is the process where the professor and students create a learning environment resulting in an assignment given to **Team**. **North**, **east** and **west** are the motions the **Team** takes transiting about the campus. **Solve** is the process of completing the homework. **Gradea** is the professor's process for grading the homework. **Plagiarize** is the process of team $\mathbf{S_2}$ copying the homework from an element of $\mathbf{S_1}$. The mission will track the movement of the individuals of Team and Prof through the campus as the students complete their assignment and receive a grade.

A Modeling Category for Student Misconduct Missions

Generating Objects

```
Person::Ob
Team::Ob
Material::Ob
Homework::Ob
Prof::Ob
Score::Ob
Test::Ob
Assgn::Ob
```

Selected Generating Axioms

```
north · south     == id
south · north     == id
east  · west      == id
west  · east      == id
north · east      == east · north
south · east      == east · south
(copy⊗id)·copy    == (id⊗copy)·copy
(merge⊗id)·merge  == (id⊗merge)·merge
```

Generating Morphisms

```
north: Team → Team
south: Team → Team
east: Team → Team
west: Team → Team
meet: Team⊗Team → Team⊗Team
exchange: Team⊗Team → Team⊗Team
acquire: Team⊗Material → Team
deposit: Team → Team⊗Material
comm: Team⊗Team → Team⊗Team
class: Team⊗Prof → Team⊗Assgn
exam: Team⊗Prof → Team⊗Test
presentation: Team⊗Prof → Team⊗Assgn
gradea: Assgn⊗Prof → Score
gradet: Test⊗Prof → Score
gradeh: Homework⊗Prof → Score
solve: Team⊗Assgn → Assgn
take: Team⊗Test → Test
obsfucate: Assgn → Assgn
plagiarize: Team⊗Assgn → Assgn
```

Figure 9. An example category presentation. One lists the concepts and relationships between those concepts along with any axioms of the theory. These axioms allow symbolic computation techniques to reason about models analytically before running an explicit simulation of the model numerically.

The **Team** and **Prof** transit on *streets*, *walkways* and *bike paths* between their *dwellings* and *buildings* containing classrooms, offices and study areas. Once **Team** and **Prof** arrive at the appropriate building they participate in the *work* and *education* interactions. Transit is represented by a series of event chains composed of the primitive movement operations **north**, **east**, **and west**. Each of the processes of **class**, **solve, gradea**, and **plagiarize** are represented by an event chain. A nearest neighbor analysis of the City Anatomy graph show that *persons*, *buildings* and *dwellings* may have significant impact on the outcome of the event chain. The integration of these event chains would then become the mission thread. Based on the number of possible primitive events, there is a combinatorial explosion of mission complexity.

Mission Threads and OODA

To complete a mission thread a functional decomposition of the schema must be conducted. For this example, we will be building an event chain, a subset of the overall mission thread. An OODA-base event chain is a Directed Acyclic Graph where the edges represent the movement of people or information consistent with the schema, Figure 8. There are a set of edges representing **Team** transiting to and from the processes **class, solve** and **plagiarize**. There are a set of edges representing **Prof** transiting to and from **class** and **gradea**. There are also two types of edges: external communication between sets $\mathbf{S_1}$, $\mathbf{S_2}$ and/or **Prof**, and internal communication within a set. The nodes for $\mathbf{S_1}$, $\mathbf{S_2}$ and **Prof** are OBSERVE, ORIENT, DECIDE and Act where the Orient nodes are:

1. Parse the physical and natural environment relevant to the moment,
2. Parse the human environment relevant to the moment,
3. Parse and analyze the OBSERVE data,
4. Mine historical data,
5. Reassess the ability to meet the mission goal and effects chain goal(s).

The Orient nodes are the essence of learning from observation and experimentation.

The concept of creating an OODA-based event chain will be demonstrated on the process **class**. Using **Prof** as an example, a plausible functional decomposition for **class** could begin as:

- **Prof** walking to **Class,** is an ACT within the *Transport* function
- **Prof** checks the IT infrastructure and reassess the ability to meet the lesson goal is an ORIENT. This transitions from *Transport* to *Working*.
- **Prof** observes **Team,** OBSERVE
- **Prof** processing state of **Class** and **Team,** ORIENT 1, 2 & 3
- **Prof** decides to start lesson, DECIDE
- **Prof** teaching in **Class** with **Team**, ACT
- **Prof** teaching in **Class** with Team, ORIENT 2, 3 & 5

The ORIENT function could be represented as multiple entities or bundled as a single entity type. Figure 10 is an instance of an event chain for **class**. The class begins as above with the entrance of the entities of **Prof** and **Team**. In class a learning environment is established, teaching material presented, a question and answer session followed with an assignment given to **Team**. Team breaks up during class into to subsets, $\mathbf{S_1}$ and $\mathbf{S_2}$. The class ends with a communication between students in $\mathbf{S_1}$ and $\mathbf{S_2}$ requesting assistance to cheat, then $\mathbf{S_1}$ and $\mathbf{S_2}$ transit away in two different directions.

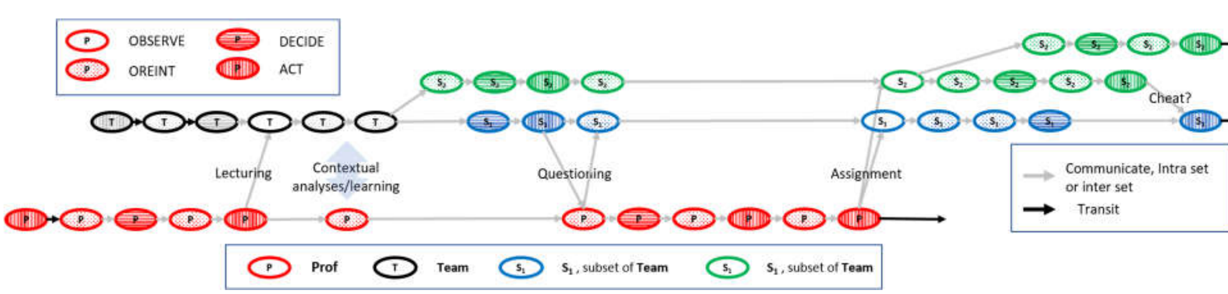

Figure 10. An event chain for **Class**

Mission Mathematics

In the ACT paradigm, you define a model as a syntactic expression that describes a model as a formula, then give that model semantics that you can view as a meaning, behavior or implementation. In terms of Mission Engineering, the wiring diagram (or equivalent program or formula) is the syntactic representation of the mission, and the semantics of the mission are given by a relational algebra database instance or a run of a simulation in a traditional M&S context.

For example, a mission thread containing movement operations is a relation on places that satisfy the constraints of that thread. If the mission thread was $p = north \cdot east \cdot north$ then the semantics of that mission thread is the set of all pairs of points $(x_1, y_1), (x_2, y_2)$ such that there is a path $(x, y + 1) \rightarrow (x + 1, y + 1) \rightarrow (x + 1, y + 2)$ in the transportation network. In a city aligned to a grid, we have the axiom that $north \cdot east = east \cdot north$ so an automated reasoning algorithm could compute analytically that $north \cdot east \cdot north = north \cdot north \cdot east$ and thus would know that the semantics of a path p is given by the set $\{(x, y), (x + 1, y + 2) \mid x, y\}$. The ACT paradigm connects syntactic descriptions of models to their mathematical behavior by the principle of compositionality. Let F denote the map from descriptions of systems to their behaviors, we say that F is compositional if $F(f \cdot g) = F(f) \cdot F(g)$ for all pairs of models f, g. This property can be characterized by Fpreserving the algebraic structure of composition $(\cdot)$. Algebraic properties and maps that preserve them are essential to the mathematical analysis of systems.

By formalizing our mission engineering frameworks as algebraic objects (categories), we can build computer algebra systems that can analyze mission threads within those frameworks. When implementing the semantics of mission engineering in a software system, a relation database can store the data and database joins implementing the composition rule for the semantics. In the example above, the composition of path is implemented as a database join because if you want to compose a path $p \rightarrow q$ with a path $r \rightarrow s$ you are looking for all the paths $p \rightarrow q, r \rightarrow s$ where $q = r$. This is precisely the notion of a database join and could be implemented in SQL as `select t1.start, t2.end from paths as t1 join paths as t2 on t1.end == t2.start`. In this way the map F transforms descriptions of paths into relations on pairs of points that turns composition of paths into joins of relations. When we say that mission engineering software can generate a simulation of the mission thread, it is exploiting these structure preserving maps to go from wiring diagrams of the mission thread to a computer program that computes the semantics of the mission thread. In the example above, that means converting sequences of directions into SQL queries that compute sets of paths.

When implementing the semantics of a mission thread in real-world data, you have to mine the structures from data you have available. For that purpose we turn to the Open Street Map as a source for transit networks. The mission thread specifies that a set of event chains for **Transit** would be a set of paths along *Streets*, *Walkways* and *Bike Paths* through the campus. There would be the set of paths for **Prof** and a set of paths for each element (student) in the set **Team**. These paths are obtained from shortest path applications and a temporal component could be added with routing applications, e.g., the proprietary Google Maps or the open source Grasshopper Maps. The parsing and interrogation of Open Street Map data can be readily achieved by the use of open source tools like OSMnx tool [Boeing, 2017]. A parsed data set from OSMnxset from which routing analyses could be conducted is shown in Figure 11. In this data set there are 5057 nodes and 6373 edges and 12 defined categories of streets, walkways, bike paths, parking lots, etc. For brevity we will not execute the routing analysis in this paper.

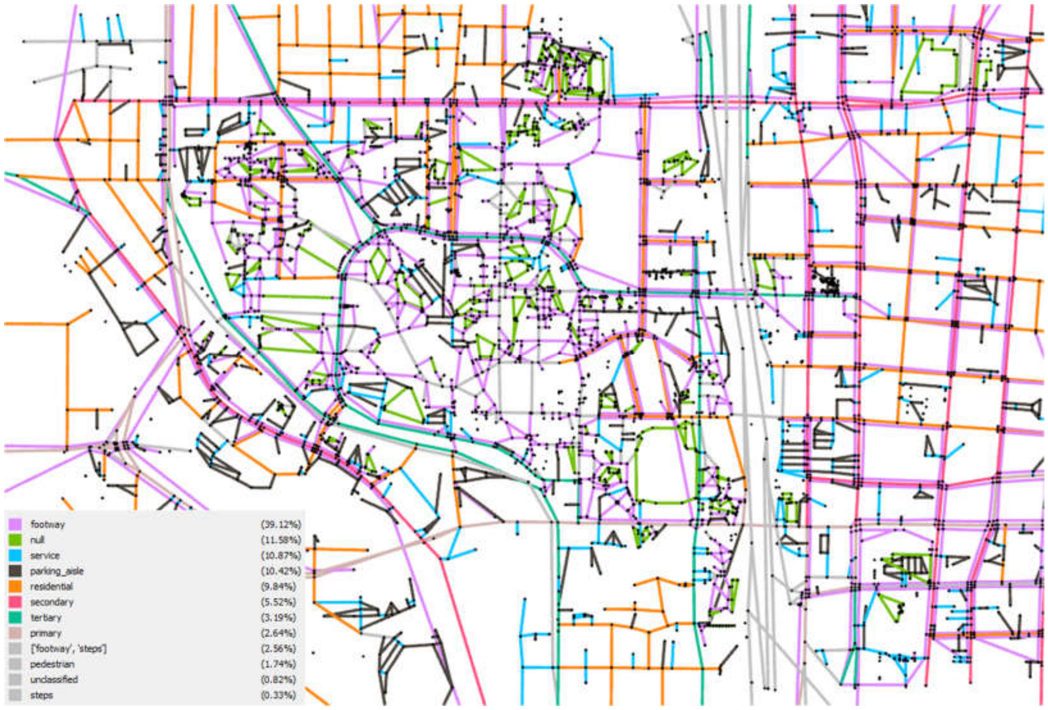

Figure 11. The mobility network in the GaTech neighborhood, parsed from Open Street map, is presented as an undirected graph with 5057 nodes and 6373 edges

Trust, an Overarching Measure of Mission Confidence

There is an increasing desire to use things (e.g., cellphones, physical devices, vehicles, and other systems embedded with electronics, software, sensors, actuators) in lieu of humans in dangerous or routine situations, and also to make things more intelligent such that they can deliver personalized services. The growth of the Internet of Things (IoT) introduces increasing complexity and scale, and raises questions about the trustworthiness of this emerging technology. The approach used to establish trust in the IoT will be the starting point to establish a trust protocol for mission engineering.

Defining Trust

The connection between people and things is complex, and creates a set of trust concerns. Trust should be considered at two levels: (1) whether a thing trusts the data it receives or trusts the other things it interacts with (machine to machine, or M2M) and (2) whether a human trusts the things, services, data, or IoT offerings that it uses (human to machine, or H2M / M2H). This leads to the idea that trust is multi-dimensional.

Ahn et al. [2007] described the concept of multi-dimensional trust by different agent characteristics, such as quality, reliability, and availability. For Matei et al. [2009], trust refers to the trustworthiness of a sensor, whether it has been compromised, the quality of data from the sensor, and the network connection. Grandison and Sloman [2000] define trust as the belief in the competence of an entity to act dependably, securely, and reliably within a specified context. To address behavior uncertainty in agent communities, Pinyol and Sabater-Mir [201]) define three levels of trust based on human society: security, institutional, and social. Leisterm and Schultz [2012] identify technical, computational, and behavioral trust, but focus primarily on a behavioral trust indicator. The idea that trust is a level of confidence is captured by Voas et. al [2018]: the probability that the intended behavior and the actual behavior are equivalent given a fixed context, fixed environment, and fixed point in time. Lastly, recognizing trust is multi-dimensional, NIST defines it as "… the demonstrable likelihood that the system performs according to designed behavior under any set of conditions as evidenced by characteristics including, … security, privacy, reliability, safety and resilience" [NIST, 2017].

Work on trust management is often divided into two areas: security-oriented and non-security- oriented [Terzis, 2009]. Security-oriented trust adopts a restricted view, where trustworthiness is equated to the degree to which an entity or object is considered secure. This traditional view sees trustworthiness as an absolute property that an entity either has or doesn't have. This is often accomplished by determining the credentials an entity possesses, and iteratively negotiating how to disclose certified digital credentials that verify properties of trust. An example of this is GTRI's Trustmark program (GTRI, 2013), which facilitates federated identity and attribute management (i.e., the reuse of digital identities and associated attributes) in enterprise systems. Identity reuse requires trust between entities that assert attributes and entities that rely on such assertions. The rules and requirements for establishing such trust comprise an identity trust framework. Non-security-oriented trust adopts a wider view similar to the social sciences. This includes a view of trust as a mechanism for achieving, maintaining, and reasoning about the quality of service and interactions. Trust is determined on the basis of evidence (personal experiences, observations, recommendations, and reputation) and is situational, meaning an entity's trustworthiness differs depending on the context of the interaction. A goal of trust management is about managing the risks of interactions between entities, which includes the notion of malicious and selfish behavior. Since non-security oriented trust is similar to the human notion of trust, areas such as computer-mediated trust between users, building human trust in computer systems, and human-computer interaction are part of the continuum of interest.

Trust and OODA

OODA is about creating processes and implementing procedures that cycle through the loop quickly and ultimately lead us to action, whether it's automated or not. Consider the plagiarism example of Figure 8, once **Prof** has observed the **Team**, it is parsing the physical and natural environment, parsing the human environment, and parsing and analyzing the OBSERVE data that leads to the conclusion that the lesson, ACT, can begin. It is this ORIENT processing that enables the action to occur. In a similar manner **Team** OBSERVEs the lecture and the subsequent ORIENT processes the lesson, i.e., learning, leading to the action of questioning. Likewise when a member of **Team S_2** cheats, they DECIDE after reassessing their ability to successfully complete the homework assignment. The benefit is from ORIENT processes, which provide the insights that inform the decisions and subsequent actions needed. Where observe, decide and act are system functions, orient is multi-dimensional. The five ORIENT sub-functions (mentioned earlier) which provide contextual awareness and external communications are fundamental to trust, especially in settings such as IoT and smart cities. Collecting new information, previous experience and the ability to analyze/synthesize data are precursors to computing trust. However, it is the ability to parse the physical and natural environment (genetic heritage) and human environment (cultural conditions) relevant to the moment, that enables us to compute trust based on evidence, i.e. the non-security oriented trust described above. Further, the syntax and semantics of the message, timeliness, and verification the message was successfully conveyed are all key to computing trust. In settings such as smart cities, where the urban environment is in constant change, these orient sub-functions reduce uncertainty, and enable better decisions and informed actions.

Trust and MSAL

To tie this together in a system model for MSAL, we adopt a layered trust framework defined by Yan et al. [2014]. These layers work together to create an environment in which things and humans can interact and make trustworthy decisions. The layers in the framework include (i) physical perception, which perceives physical environments and human social life; (ii) a network layer that transforms and processes perceived environment data; and (iii) an application layer that offers context-aware intelligent services in a pervasive manner. The fourth layer, trust management, represents the cyber-physical social relationships that connect layers. Figure 12 depicts these layers, with trust objectives. A trustworthy system relies on the cooperation among layers. "Ensuring the trustworthiness of one … layer (e.g., network layer) does not imply that the trust of the whole system can be achieved" [Yan et al., 2014].

Figure 12. IoT Trust framework

MSAL is designed to reduce mission uncertainty, providing decision makers quantitative measures of mission success to include mission thread and event chain topologies. There are, however, other criteria in addition to MSAL-based data that are important in making evidence-based decision. Therefore, trust in decision making must be considered from a mission perspective, including both security and non-security-oriented properties. Determination of trust requires analysis of the ORIENT functions across a mission thread. Thus, trust is a based on managing the interstitials, as described below.

Trust at the Application layer – "What to trust" – is where the individual simulations reside (e.g., agent based, equation based, AI based) that execute the mission model. Since no one simulation contains everything important to the mission model, it is conceivable that strength of trust comes from having many simulations with different assumptions and world context. In other words, an ensemble of simulations with different perspectives has the effect of reducing risk by giving greater understanding to sensitivity and uncertainty. First, each simulation is tested with the same data set (parameter distributions, not scenarios). Aleatoric and when appropriate epistemic parameter distributions will be defined and their effects upon simulation outcome evaluated [Ferson, 2007]. Single parameter sensitivity studies will be used to assess the most impactful parameters for subsequent study. Optimization will provide insight into areas of high gradients and parameter interactions within the performance space. UQ will then define confidence bounds about the performance space for each simulation. For example, Figure 13 shows the effects of a parameter study with a single simulator. Each prediction of the path of hurricane Cristobal, from UKMET[7] simulator, is shown as a separate line; these are instances of bounding mission threads. Since the same input distributions should yield similar results, all of the ensemble simulations should show similar trends (e.g., uncertainty volumes). If one of the simulations produces results very different that the rest of the ensemble, there is reason to question that simulation. In other words, the comparison of ensemble results will lead to an understanding of which simulations to trust, and how much.

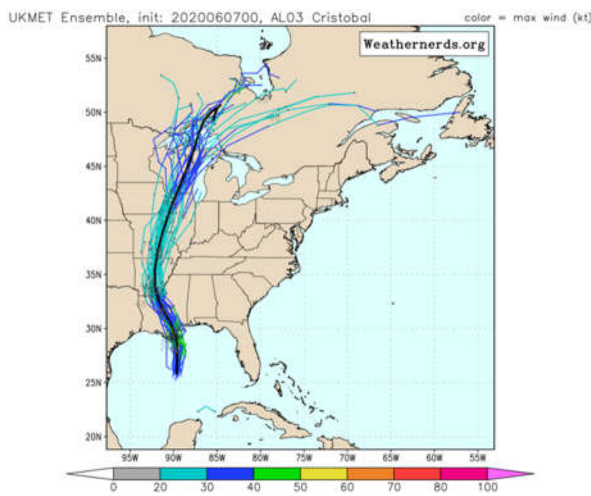

Figure 13. Parameter sensitivity analysis from a single simulation

Trust at the Network layer – "When to trust" – is where the mission graphs reside. A mission graph is composed of systems, the environment, policy and doctrine, and connecting relationships (which can be related to the structure, society and interactions dimensions of the City Anatomy). This layer captures both a visual and mathematical behavior of people and mobility. At this layer, mission graphs are structured using ACT, and transformed using graph analytics (quantifying/assess topologies, centrality, clustering). Temporal changes to topology, based on context, is one particular focus of trust in this layer.

Trust at the Physical Perception layer – "Where to trust" – At this layer mission threads and reside. A mission thread is a sequence of actions/processes with quantifiable outcomes. There are many plausible mission threads that can be executed, and each thread has a temporal component. Simulator specific scenarios are created for each mission thread along with establishing the input parameters. An ensemble of simulators is run on each mission thread and the aggregate outcome is used to establish error bounds. It is by considering the ensemble of all possible mission threads that provides the best insight, and enables us to derive a greater understanding of trust.

To illustrate this concept, consider how simulation is used in hurricane tracking. In Figure 14, an ensemble of simulations is run on the mission threads, and the aggregate outcome shows the error bound of all

[7] A medium-range (3 to 7 day) numerical weather prediction model operated by the United Kingdom METeorological Agency. It has a resolution of 75 kilometers and covers the entire northern hemisphere. Forecasters use this model along with the ECMWF and GFS in making their extended forecasts (3 to 7 days). https://forecast.weather.gov/glossary.php?word=ukmet

possible outcomes. The ensemble is from the European Centre for Medium-Range Weather Forecasts (ECMWF), which has pioneered a system to predict forecast confidence. This system is the Ensemble Prediction System (EPS). What the ensemble shows us is that over time context evolves and dispersion increases.

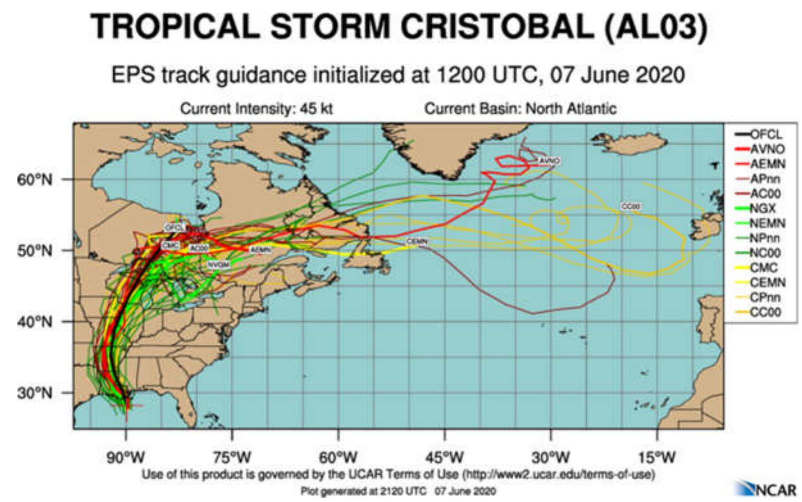

Figure 14. Ensemble hurricane prediction

Trust Management layer – "How to trust" – The trust management layer can be thought of in terms infrastructure and the systems which check that programs, e.g. databases, queries, analytics, etc., are sound. The type systems put constraints on application programs such that each function can only be applied to the right "type" of value. Since all functions are annotated with the input type and return type, a compiler can reason over an expression to determine if the expression is "well-typed." As you refine the types of a program to be more precise about which sets of values are distinguished at the type level, you get stronger guarantees that programs that type check are trustworthy. In this way, as you use ACT in constructing models, you get more precise constraints about what kinds of mission threads are valid scenarios for your simulation. A software system that uses the principles of ACT can validate these scenarios before executing them based on a logical notion of well-formed expressions. In functional programming, careful work needs to be done upfront to establish the types and the signatures of the functions in order to capture the "business logic" into the type system. The analogous process in the ACT approach is to design a category whose objects are types of values and whose morphisms are processes where the signature of each morphism respects the domain knowledge of experts in the systems under examination. Once this work to capture the expert knowledge is done, computational systems can automatically validate statements about these systems including statements about the existence of particular mission threads. This computational validation yields higher trust in the resulting modeling and simulation systems.

Mission engineering software can check the well-formedness of mission threads by checking that whenever two processes are composed sequentially, the output type of the first process matches the input type of the second process. The structure of a Generalized Algebraic Theory contains all the information necessary to make these type judgements. That a mission thread is well-formed is not sufficient to say that it is relevant to a mission engineering analysis or that a set of well-formed mission threads is exhaustive, but it is necessary for a mission thread to "make sense" within the scope of the system. This requirement can allow mission thread planners to avoid the consideration of meaningless scenarios in a machine enforceable way. Many simulation developers in DoD communities will object to analyses conducted with a simulation software because the scenarios analyzed “don’t make sense from a domain perspective.” These objections are informal arguments against certain mission threads, and by formalizing this process as checking the well-formedness of expressions, we can automate those judgements, which will accelerate mission engineering timelines. Database systems perform these checks on queries as part of their runtime systems. Before a RDBMS will execute a query it will check that all of the columns referenced by the query exist and any functions called in the query support the datatypes that the schema uses. In this way they will invalidate the comparison of a DATETIME with an INTEGER unless appropriate conversion functions are used. While these constraints are sometimes frustrating to developers, they are very valuable for ensuring the correctness and integrity of large scale software systems. While traditionally these type checking techniques are only used in the design and implementation of programming languages, the ACT approach widens their applicability to large areas of mathematics and scientific computing. Our mission engineering approach applies these techniques to the analysis of mission threads.

The Next Step

We have described the application of MSAL to both a military and smart city example. We now consider a third case study, inspired by the COVID-19 Pandemic, that can exercise all the aspects of MSAL and trust described previously. In addition, data is readily accessible and many models and simulations are open source [CDC, 2020]. A potential mission could be the phased reopening of the GaTech campus for the fall 2020 semester, while anticipating a second wave of the pandemic to strike Atlanta during the semester. A mission goal of successfully completing academic requirements for all students with no deaths to students, faculty and employees provides for a plausible example.

Building upon the Existing Epidemiology Toolset

There are three types of epidemiology simulations; equation-based, agent-based and discrete event. Deterministic, equation-based simulations are the classic epidemiology simulations and have been around for decades [Sharkey, 2008]. They are homogenous in nature and should be applied carefully to large populations [Hethcote, 1994]. Critics of equation-based techniques argue that detailed social behaviors are necessary for credible simulation results [Shen, 2020]. Agent-based simulations, sometimes referred to as stochastic simulations, explicitly represents population heterogeneities, i.e., networks with behavior functions [Grefenstette et al, 2013] [Ferguson et al, 2005]. Many agent-based simulations are massively parallel with capability to add substantial detail [Halloran, 2020]. Massively parallel discrete event simulations have also been developed to capture ever increasing detail [Perumalla, 2020]. There are also federated models and simulations creating integrated products to aid in pandemic decision making [Stein et al, 2012] and supply logistics [Araz, 2013].

A thorough comparison including single parameter, parametric analyses were made of three massively parallel, agent-based simulators using the ‘same’ scenario to address the effectiveness of various mitigation strategies in an influenza pandemic [Halloran, 2008]. The analytics used and the comparisons made are simplistic. The authors describe significant differences in the implementation of social networks that make direct comparisons difficult. The case can be made that this simulation is focused on the OODA OBSERVE and ACT function, where decisions are pre-scripted and the orient sub-functions are ignored. All three of these simulators are open source and can be readily applied to this proposed mission.

Typically, epidemiology scenarios are not unambiguously defined. The CDC recently defined COVID-19 scenarios as the traditional epidemiological simulation parameters with values typical with an upper and lower bound without societal context [CDC, 2019]. Parameter distribution details are not referenced. In other words, the goal of these scenarios is not to look at or explore the shape of curve by pulling from distributions, rather it is simply establishing a nominal upper/lower bound. Language-based descriptions of simulated social networks are sometimes provided by authors but not with sufficient detail to understand their algorithms. The use of OODA-based event chains and mission threads would add explicit clarity to the analysis. The use of graph analytics and visual inspection upon the mission graph and mission threads could provide more insight into social behaviors (heterogeneity) than functional descriptions embedded within the simulation. For example, a clustering and community analysis of a mission graph could readily point out areas of high potential transmission enabling simulation over a more simplistic mission thread.

A 2006 review of pandemic influenza modeling by the National Academy of Sciences [NAS, 2007] resulted in a series of recommendations to prepare for future pandemics. These recommendations include improved methods for estimating model and parameter uncertainty, and defining broader outcome measures to include the costs and benefits of intervention strategies. The current state of epidemiological modeling and simulation has yet to address these recommendations. Parameter aleatoric uncertainty is rarely quantified and epistemic uncertainty ignored. Parametric sensitivity studies are, when reported, conducted one-at-a-time. Multiparameter optimization and uncertainty quantification studies to quantify parameter interactions are nonexistent. Massively parallel simulations are routinely run on various social networks, yet the network graphs are rarely displayed and graph analytics not used.

Using MSAL and Trust

The Application Layer of a COVID-19 investigation would consist of a set of about 6 simulators that the epidemiological community is currently using to study the effects of mitigation on COVID-19 incidence. Parameter sensitivity, optimization and UQ would be conducted for each simulator building upon the CDC scenarios. These results gauge trust in the individual simulations about a similar input distribution and would be the basis to select the group of simulators for subsequent work.

CDC reports simulation results with their COVID-19 forecasting, it begins to addresses the question whether a simulation is consistent with expectations and the data. Two week projections, shown in Figure 15, provide an interesting visual understanding of the wide range of forecasts of individual simulations, based on their underlying assumptions and data. However, they only show simple trends based simplistic parameter sensitivity runs (nominal, upper and lower bounds), they do not look at the shape of the parameter distributions. We need to look at where there are peaks and valleys in the bounding volume, not simply the shape of a flat curve. Unlike hurricane tracking simulations where there is more integration of past data into future predictions, this is simply a collection of simulation runs without subsequent analyses.

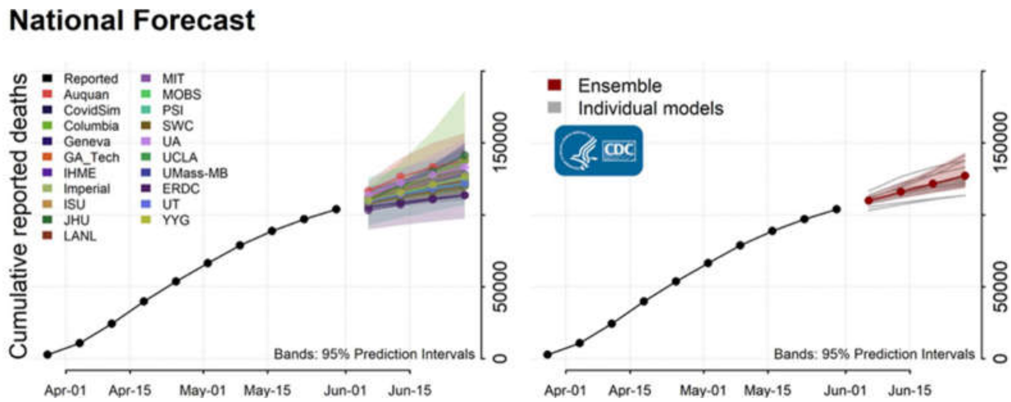

Figure 15. CDC National COVID-19 Forecasting[8]

The Network Layer is the unstructured component that deals with concepts that are heterogeneous and not readily quantified by traditional equation-based techniques. It includes the graphical constructs of the mission model. The mission model would be built upon our previous work starting with Figures 5 and 6, and explicitly add relevant networks and network detail, e.g. the mobility network of Figure 11. Preliminary event chains would be defined based on the City Anatomy Interactions of functions, economy, culture and Information. These event chain graphs would be very much like the plagiarism example above where physically passing a homework assignment would be analogous to passing a virus. The event chains would be OODA-based with an emphasis on the ORIENT sub-function to enable subsequent trust analyses. In this construct economy would look at the cost of options such as social distancing, isolation, quarantine, hybrid classes (in person and on-line); culture would include behaviors/actions about social distancing, isolation, quarantine, hybrid classes; and information would look at the effects of messaging on economy and culture changes. Thus, event chains would include cost functions and conditional probabilities of success. Graph analytics will be conducted upon the networks to look to quantify locations of homogeneity and heterogeneity, define communities and identify density centrality. Temporal changes to the graph topology are also quantified. The efficacy of network topology to define social behaviors will be evaluated and compared to social behaviors incorporated into the selected simulators.

The Physical Perception Layer is where we integrate the structured and unstructured components and use them to interrogate the mission threads. The creation of this Layer begins by defining a set of bounding OODA-based mission threads to meet the mission goal. These mission threads are built from a common mission schema and content found in the trust Network Layer. Using the simulators and insight gained from the Application Layer along with a probabilistic analysis of ensemble results will define an overarching uncertainty. The approach would be consistent with the ensemble-based probability swath [Ortt, 2018] used to express uncertainty in hurricane tracks (see Figure 14).

By decomposing the problem into structured (physical perception and application layer) and unstructured (trust network layer) parts built upon the category theory basis, one can quickly and precisely get at the key attributes that contribute to meeting the mission goal, e.g., dropping the epi curve to minimal deaths in a short time period. The use of multi-parameter UQ on many individual sims in the physical layer gets at impactful variables and particularly important combinations of variables. The network layer uses the power of graph theory and analytics to very quickly partition the problem into homogeneous and

[8] https://www.cdc.gov/coronavirus/2019-ncov/covid-data/forecasting-us.html

heterogeneous domains. Structured simulations can then focus where they are best suited; the homogeneous regions. The trust-based approach would provide quicker (cheaper), more robust, and trustworthy results than building 'federated super simulations'. The ensemble techniques then allow for appropriately focused simulations (with insight into their trustworthiness) to get at a robust uncertainty map of what we know about meeting the specific mission goal. The use of this approach would enable the epidemiology community to coordinate analysis approach while maintaining their autonomy and provide a coordinated means for decision makers to trust the simulation results.

Conclusions

This paper presents a robust approach to multi-disciplinary, mission engineering beginning with a military example. The concept is then applied to a Smart City, and finally applied to the mission of mitigating the spread of the COVID-19 Pandemic.

Mission Engineering and Integration is the definition, identification, assemblage, analysis and quantification of a SoS to achieve measurable desired effects and therefore mission success. The mission is constrained by dynamic operational context and can be inherently complex. Missions are graph-based where the number of edges can approach the number of entities squared and the maximum number of paths can approach the number of entities factorial. From the combinatorics, many plausible mission threads are possible for a given mission. However, not all paths are feasible or desirable due to the governing rules of the mission domain. Missions tend to be layered and multidimensional. They are layered in terms of abstraction; a node in a mission graph can contain very detailed substructure. They are multi-dimensional in that several interrelated functions can occur simultaneously; these functions appear as edges in the mission thread. ACT seeks universal representations of mathematical knowledge that transcend domains and disciplines. The ACT approach is inherently computational and universal, which makes it a candidate framework for studying mission engineering. ACT is suited to rigorously define a mission schema. This mathematical basis of the schema necessarily defines a corresponding database schema, and the architecture of a mission simulator.

The key to OODA-based decomposition of mission threads and event chains is Boyd’s ORIENT function. ORIENT is used as an edge in a graph to represent the function of communication. Communication encompasses the communication infrastructure, the syntax and semantics of the message, timeliness and trustworthiness, and verification that the meaning of the message was successfully conveyed. In addition, ORIENT has several entity functions related to situational awareness and information processing, defined as:

- Parse the physical and natural environment relevant to the moment,
- Parse the human environment relevant to the moment,
- Parse and analyze the OBSERVE data,
- Mine historical data, and
- Reassess the ability to meet the mission goal and effects chain goal(s).

The addition of ORIENT entities between the OBSERVE, DECIDE and ACT functions adds both robustness and complexity.

MSAL is an iterative approach suited to provide a variety of perspectives for quantitatively evaluating the ability of an SoS to meet mission goals. MSAL is created and executed upon a graphical mission model and an initial set of event chains and/or mission threads. Graph theoretic and ACT-based constituent capabilities of MSAL have been demonstrated in the literature enabling integration of a robust MSAL toolbox.

Since missions are inherently uncertain due to complexity and the dynamic nature of context, the broad issue of trust across a mission knowledge base and mission level analysis become vital to decision making. Trust can be measured for both security and non-security properties of missions, yet understood at the mission level. A Trust Framework is proposed that is comprised of four layers; Application, Network, Physical Perception and Trust Management. For mission models and simulations, the Application Layer evaluates single simulators and quantifies uncertainties. The Network Layer is graphical and contains the mission models and event chains. The Physical Perception Layer uses an ensemble approach of multi-simulators running the same mission threads similar to the technique used for hurricane prediction. Convergence of the results over time builds trust. The Network Management Layer pulls together the three layers and is constrained by an ACT basis. This is the layer of data storage, query and scripting. Metrics that enable decisions makers to quickly grasp which mission threads are feasible with what level of risk is an emerging and important area of research. The goal of MSAL is not to find the perfect answer, rather it provides a quantifiable, testable approach that enables people to make better informed decisions.

Acknowledgements

This effort was funded by the United States Department of Defense, OUSD(Acquisition & Sustainment)/ASD(Acquisition), Mission Engineering and Integration under contract HQ0642930737.

References

Agreement, C. P. (2015). City Anatomy: A Framework to support City Governance, Evaluation and Transformation. *CPA-I_001-v2_City Anatomy, https://cpsociety. sharepoint. com/sites/cptf/CPTSC/Private% 20Documents/Publications/CPA-I_001-v2_City_Anatomy. pdf*.

Ahn, J., D. DeAngelis, and S. Barber. 2007. “Attitude Driven Team Formation Using Multi-Dimensional Trust”. In Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT ’07), 2nd–5th November, Fremont, CA, 229 –235.

Agreement, C. P. (2015). City Anatomy: A Framework to support City Governance, Evaluation and Transformation. *CPA-I_001-v2_City Anatomy, https://cpsociety. sharepoint. com/sites/cptf/CPTSC/Private% 20Documents/Publications/CPA-I_001-v2_City_Anatomy. pdf*.

Araz, O. M. (2013). Integrating complex system dynamics of pandemic influenza with a multi-criteria decision making model for evaluating public health strategies. Journal of Systems Science and Systems Engineering, 22(3), 319-339.

Beheshti, R., & Sukthankar, G. (2013, November). Analyzing agent-based models using category theory. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)* (Vol. 2, pp. 280-286). IEEE.

Besta, M., Peter, E., Gerstenberger, R., Fischer, M., Podstawski, M., Barthels, C., ... & Hoefler, T. (2019). Demystifying Graph Databases: Analysis and Taxonomy of Data Organization, System Designs, and Graph Queries. *arXiv preprint arXiv:1910.09017*.

Boeing, G. 2017. “OSMnx: New Methods for Acquiring, Constructing, Analyzing, and Visualizing Complex Street Networks.” *Computers, Environment and Urban Systems*. 65, 126-139. doi:10.1016/j.compenvurbsys.2017.05.004

Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the national academy of sciences*, *99*(suppl 3), 7280-7287.

Boyd, J. R. (1987). *A discourse of winning and losing*. Retrieved from https://danford.net/boyd/essence.htm

Boyd, J. R. (1976). Destruction and Creation. US Army Command and General Staff College. *3/31/15*.

Censi, A. (2017). "Uncertainty in Monotone Codesign Problems." *IEEE Robotics and Automation Letters* 2.3, 1556-1563.

CDC (2019). https://www.cdc.gov/coronavirus/2019-ncov/hcp/planning-scenarios.html

CDC (2020). https://www.cdc.gov/covid-data-tracker/index.html

Dempster, A. P. (1968). A generalization of Bayesian inference. Journal of the Royal Statistical Society: Series B (Methodological), 30(2), 205-232.

DoD Mission Engineering and Integration Guidebook, DoD DAU website, March 2020, Dr. James D. Moreland, Jr.

Fairbanks, J. P., Kannan, R., Park, H. and Bader, D. A. (2015). Behavioral clusters in dynamic graphs. Parallel Computing, 47, 38-50.

Ferguson NM, Cummings DA, Cauchemez S, Fraser C, Riley S, Meeyai A, Iamsirithaworn S, Burke DS (2005) *Nature* 437:209–214

Ferson, S., Nelsen, R. B., Hajagos, J., Berleant, D. J., Zhang, J., Tucker, W. T., ... & Oberkampf, W. L. (2015). *Dependence in probabilistic modeling Dempster-Shafer theory and probability bounds analysis* (No. SAND-2015-4167J). Sandia National Lab.(SNL-NM), Albuquerque, NM (United States).

Garrett Jr, R. K., Anderson, S., Baron, N. T., & Moreland Jr, J. D. (2011). Managing the interstitials, a system of systems framework suited for the ballistic missile defense system. *Systems Engineering*, *14*(1), 87-109.

Georgia Tech Research Institute (GTRI). 2013. “NSTIC Trustmark Pilot”, https://trustmark.gtri.gatech.edu, accessed 17th June 2019.

Grandison T. and M. Sloman. 2000. “A Survey of Trust in Internet Applications”. IEEE Communications Surveys and Tutorials 3(4): 2-16. Guichard, D. (2017). An Introduction to Combinatorics and Graph Theory. *Whitman College-Creative Commons, page 13*.

Grefenstette JJ, Brown ST, Rosenfeld R, Depasse J, Stone NT, Cooley PC, Wheaton WD, Fyshe A, Galloway DD, Sriram A, Guclu H, Abraham T, Burke DS. FRED (A Framework for Reconstructing Epidemic Dynamics): An open-source software system for modeling infectious diseases and control strategies using census-based populations. BMC Public Health, 2013 Oct;13(1), 940. doi: 10.1186/1471-2458-13-940. PubMed PMID: 24103508

Halloran, M. E., Ferguson, N. M., Eubank, S., Longini, I. M., Cummings, D. A., Lewis, B., ... & Wagener, D. (2008). Modeling targeted layered containment of an influenza pandemic in the United States. Proceedings of the National Academy of Sciences, 105(12), 4639-4644.

Halter M, Herlihy C, Fairbanks J. A. (2019). Compositional Framework for Scientific Model Augmentation. Halter, M., Herlihy, C., & Fairbanks, J. (2019). A Compositional Framework for Scientific Model Augmentation. *arXiv preprint arXiv:1907.03536*.

Hethcote, H. W. (1994). A thousand and one epidemic models. In *Frontiers in mathematical biology* (pp. 504-515). Springer, Berlin, Heidelberg.

Henkel, R., Wolkenhauer, O., & Waltemath, D. (2015). Combining computational models, semantic annotations and simulation experiments in a graph database. *Database*, *2015*.

Henriksen, J. O. (2008). Taming the complexity dragon. *Journal of Simulation*, *2*(1), 3-17.

Jacobs B., Kissinger A., Zanasi F. (2019) Causal Inference by String Diagram Surgery. In: Bojańczyk M., Simpson A. (eds) Foundations of Software Science and Computation Structures. FoSSaCS 2019. Lecture Notes in Computer Science, vol 11425. Springer, ChamKim, Y. H., & Peeta, S. (2016). Graph-Based Modeling of Information Flow Evolution and Propagation under V2V Communications-Based Advanced Traveler Information Systems. *Computer-Aided Civil and Infrastructure Engineering*, *31*(7), 499-514.

Joyal, A., & Street, R. (1991). The geometry of tensor calculus, I. *Advances in mathematics*, *88*(1), 55-112.

Kinder, A., Henshaw, M., & Siemieniuch, C. (2014). System of systems modelling and simulation–an outlook and open issues. *International Journal of System of Systems Engineering*, *5*(2), 150-192.

Ladyman, J., Lambert, J., & Wiesner, K. (2013). What is a complex system?. *European Journal for Philosophy of Science*, *3*(1), 33-67.

Lange, P., Weller, R., & Zachmann, G. (2016, May). GraphPool: A High Performance Data Management for 3D Simulations. In *Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation* (pp. 23-33). ACM.

Leisterm, W. and T. Schultz. 2012. "Ideas for a Trust Indicator in the Internet of Things". In Proceedings of the First International Conference on Smart Systems, Devices and Technologies (SMART 2012), 27th May – 1st June, Stuttgart, Germany, 31-34.

Liu, L., Loper, M., Ozkaya, Y., Yasar, A., & Yigitoglu, E. (2016). Machine to Machine Trust in the IoT Era. In *TRUST@ AAMAS* (pp. 18-29).

Loper, M., Garrett, R.K. Jr. (2015). A Comparison of Traditional Simulation and the MSAL Approach, Georgia Tech Research Institute Information & Communications Laboratory Technical Report ICL-DO-01-15, retrieved from https://pdfslide.net/documents/a-comparison-of-traditional-simulation-and-msal-6-3-2015.html

Loper, M.L. (2019) Simulation Trust & the Internet of Things, *Proceedings of the 2019 Winter Simulation Conference N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y-J. Son, eds.*

Mabrok, M. A., & Ryan, M. J. (2017). Category theory as a formal mathematical foundation for model-based systems engineering. *Appl. Math. Inf. Sci*, *11*, 43-51.

Matei, I., J. Baras, and T. Jiang. 2009. "A Composite Trust Model and its Application to Collaborative Distributed Information Fusion". In Proceedings of the 12th Interna-tional Conference on Information Fusion (FUSION 2009), 6th–9th July, Chicago, IL, 1950 –1957.

NAS (2007). Committee on Modeling Community Containment for Pandemic Influenza and Institute of Medicine (2007) Modeling community containment for pandemic influenza: A letter report. Available at: http://books.nap.edu/catalog/11800.html#orgs.

NIST. (2017). "Framework for Cyber-Physical Systems: Volume 2". National Institute for Standards and Technology (NIST), Publication 1500-202, http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-202.pdf, accessed 10th July 2019.

Ortt, D. (2018, April). An Improved Methodology for Expressing and Communicating Tropical Cyclone Risk. In 33rd Conference on Hurricanes and Tropical Meteorology. AMS.

Patterson E. (2019), "*Hausdorff and Wasserstein metrics on graphs and other structured data*". https://arxiv.org/abs/1907.00257

Perumalla, K. S., & Seal, S. K. (2012). Discrete event modeling and massively parallel execution of epidemic outbreak phenomena. Simulation, 88(7), 768-783.

Pinyol I. and L. Sabater-Mir. 2013. "Computational Trust and Reputation Models for Open Multi-Agent Systems: A Review". Artificial Intelligence Review 40(1):1–25.

Rioux, G. P. and Nance, R. E. (2002). Reusing simulation components: generalizing: is it possible to create all-purpose simulations? In *Proceedings of the 34th conference on Winter simulation: exploring new frontiers* (pp. 783-790). Winter Simulation Conference.

Rittel, H. W. and Webber, M. M. (1973). Dilemmas in a general theory of planning. *Policy sciences*, *4*(2), 155-169.

Shafer, G. (1976). A mathematical theory of evidence (Vol. 42). Princeton university press.

Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T. *Cytoscape: a software environment for integrated models of biomolecular interaction networks.* Genome Res, 13:11 (2498-504). 2003 Nov. PubMed ID: 14597658.

Sharkey, K. J. (2008). Deterministic epidemiological models at the individual level. Journal of Mathematical Biology, 57(3), 311-331.

Shen, C., Taleb, N. N., & Bar-Yam, Y. (2020). Review of Ferguson et al "Impact of nonpharmaceutical interventions..". *New England Complex Systems Institute*.

Stein, M. L., Rudge, J. W., Coker, R., Van Der Weijden, C., Krumkamp, R., Hanvoravongchai, P., ... & Touch, S. (2012). Development of a resource modelling tool to support decision makers in pandemic influenza preparedness: The AsiaFluCap Simulator. *BMC public health*, *12*(1), 870.

Urias, V., Van Leeuwen, B. and Richardson, B. (2012). Supervisory Command and Data Acquisition (SCADA) system cyber security analysis using a live, virtual, and constructive (LVC) testbed. In *MILCOM 2012-2012 IEEE Military Communications Conference* (pp. 1-8). IEEE.

Voas, J., R. Kuhn, P. Laplante, and S. Applebaum, 2018. Internet of Things (IoT) Trust Concerns. NIST Cybersecurity White Paper, 17th October. https://csrc.nist.gov/CSRC/media/Publications/white-paper/2018/10/17/iot-trust-concerns/draft/documents/iot-trust-concerns-draft.pdf.

Yan, Z., P. Zhang, and A. V. Vasilakos. 2014. "A Survey on Trust Management for Internet of Things". *Journal of Network and Computer Applications* 42:120-134.