Object-Process Model-Based Operational Viewpoint Specification for Aerospace Architectures

Yaniv Mordecai, Nicholas K. James, Edward F. Crawley
Massachusetts Institute of Technology
Department of Aeronautics and Astronautics
125 Massachusetts Avenue
Cambridge, MA 02139
yanivm@mit.edu, njames000@gmail.com, crawley@mit.edu

Abstract— **Remote-controlled or autonomous multi-rotor air vehicles, or drones, have become common and commercially available even to individual consumers, mostly for imaging purposes. Drones appeal to mission architects looking to extend the toolbox provided to operators performing challenging missions such as public safety operations. However, careful analysis of the operational context and concept of operations must take place before major acquisitions. The purpose of this paper is to propose a model-based operational architecture definition framework, which is based on the Department of Defense Architecture Framework (DoDAF) ontology and uses Object Process Methodology (OPM) as its underlying modeling language. Through careful mapping of DoDAF Operational Viewpoint (OV) ontology to OPM ontology, we were able to show that the entire OV ontology can be covered by a small set of objects, processes, relations among them, and constructs comprising them. We then show how to instantiate the ontology to create a model of an actual architecture of interest (AoI) while maintaining strong typing of the model elements to ensure validity, integrity, consistency, and continuous compliance with the OV. We demonstrate our approach on the case of using drones in public safety enterprises for the purpose of crowd management in massively attended events and locations. The proposed framework allows for capturing ConOps and OpsCon in a lightweight, yet robust and consistent manner, and improve communication and concept validation between operational stakeholders and enterprise architects.**

TABLE OF CONTENTS

1. INTRODUCTION

Technology and operational need evolve side-by-side and either may precede the other. However, many system capabilities have been developed due to the availability of their underlying technology, and not necessarily as a result of pure operational necessity.

Recent advances in small quadrotor or multi-rotor remotely-piloted aircraft technology triggered an ecosystem of applications and services based on such aircraft (commonly called 'drones') – surveillance, emergency response, search-and-rescue/ seek-and-destroy, logistics, traffic monitoring, vehicle guidance, etc. [1]. Many of these operational applications could not be foreseen when the technology was first introduced. At the same time, when an organization expresses an interest in utilizing quadrotor technology it must have a very good perspective on how the technology will be deployed, used, operated, and maintained, such that it would actually support the organization in achieving its goals and objectives.

Many organizations seeking to utilize drones, including military forces, law enforcement agencies, industrial and commercial enterprises, or energy facilities, may be tempted to purchase a few commercial off-the-shelf (COTS) drones due to the relatively low vehicle unit cost. These organizations may not be aware of the need to define the Concept of Operations for the enterprise that will utilize the capabilities of the drone, and the Operational Concept that will characterize the activity of the drone to support the enterprise system.

ConOps is a widely used term in aerospace and defense agencies, which describes how the system will operate, who will operate it, when and in which conditions, and in coordination with what else [2]. A ConOps is a verbal and/or graphic statement prepared for the organization's leadership that describes the assumptions or intent regarding the overall operation or series of operations of the enterprise, using one or more systems, observed as "black boxes". This statement helps stakeholders determine if and how to include any new capability in the enterprise's operation in order to meet their strategic goals and expectations or to advance towards the achieving the organization's goals and objectives [3]–[5].

978-1-7821-2734-7/20/$31.00 ©2020 IEEE

The role of ConOps for architecting aerospace systems such as space vehicles and networks, air traffic management (ATM), and autonomous vehicle ecosystems, is evident in the various publications exploring ConOps in such systems [6]–[10].

The *Operational Concept* (OpsCon) is a stakeholder-oriented description of what the system will do, what is the rationale, and how it will support the stakeholder's operations and goals. It is not intended to describe how the system will operate [3], [11].

One of the major issues with specifying ConOps and OpsCons is the lack of formal and well-defined guidance, especially for integrating the resulting specification with the enterprise model on the one hand and with the system model on the other hand. One reference on conceptual specification of operational system contexts is the Department of Defense (DoD) Architecture Framework (DoDAF) [12] – a comprehensive guide for defining enterprise architectures. DoDAF comprises multiple *viewpoints*, one of these is the Operational Viewpoint (OV). The OV is a set of views describing operational aspects. The operational views comprising the OV are partially-overlapping in content and terminology.

Interestingly, DoDAF mentions the term *Operational Concept* significantly more than it does *Concept of Operations*, although, the OV is arguably a description of the ConOps of an enterprise or organization, rather than the OpsCon of a specific system of interest. That said, it seems unlikely that the DoD will clarify this conundrum soon.

The potential of conceptual modeling to support system architecting has been identified and emphasized long ago [13]. Recently, model-based systems engineering (MBSE) adoption and utilization have been growing [14]. Model-based system specifications and design decisions are recorded in conceptual models defined in formal or semi-formal modeling language, and has an underlying common database to ensure model manageability and consistency.

Several studies explored the idea of model-based ConOps [15]–[18]. However, MBSE still remains a Rubicon to cross for operational stakeholders, because any model-based CO-OP definition framework must be lightweight, easy to learn and apply, and capable of facilitating communication between the operational stakeholders and the enterprise architects.

The need to provide a model-based unified ConOps—OpsCon (CO-OP) framework, i.e., which accounts for both the enterprise-level ConOps, and the system-level OpsCon, has led us to explore the possibility to use Object Process Methodology (OPM) for this purpose. OPM is a conceptual modeling language and a model-based systems engineering (MBSE) framework which is also standardizes as ISO-19450 [19], [20]. We argue that an OPM model that clearly defines the operational context for a system of interest (SoI) within an enterprise or another larger-scale utilizing system (the enterprise system, ES) can serve as the basis for communication with stakeholders for the purpose of CO-OP clarification and validation, as well as the basis for a specification of the SoI's architecture, its structure, function, and behavior, that would realize the OpsCon and allow it to support the ES ConOps.

In this study we were specifically interested in considering CO-OPs for the use of drones in public safety and emergency response operations. Several attempts have been made to define ConOps for drones in emergency operations [21]–[24]. However, none of those studies used a system modeling approach, which could support the ConOps evolution into a model for a system that would realize and support the proposed ConOps.

With this in mind, this paper has two purposes: a) propose a generic model-based CO-OP analysis framework, and b) explore the CO-OPs for drones in public safety emergency response using the proposed model-based framework.

2. THE DODAF OPERATIONAL VIEWPOINT

DoDAF standardizes the framing and modeling of all Department of Defense (DoD) architectures to assist decision making and information exchange [12]. Although DoDAF only governs architectures within the US DoD, it has become a standard for militaries around the globe, with similar architectures existing for NATO, the UK, France, Canada, Italy, and Australia [25].

DoDAF specifies two types of architectures: Enterprise Architectures and Solution Architectures. Enterprise Architectures discuss missions and the organizational approach to those missions, while Solution Architectures focus on systems designed to support the organizations in fulfilling their missions. There is therefore a direct mapping of DoDAF Architectures to Enterprise Architecture to ConOps, and Solution Architecture to OpsCon.

DoDAF consists of several viewpoints. Each viewpoint is a collection of descriptions referring to a system architecture aspect. DoDAF viewpoints are listed in **Table 1**. The Operational Viewpoint (OV) is one of the industry standards for describing complex systems operations. The OV is divided into nine views that "*describe the tasks and activities, operational elements, and resource flow exchanges required to conduct operations*" [12]. The nine views are listed in **Table 2**, with the typical visualization modality of each view.

OV-1 is a flexible, high-level graphic illustration used as a presentation or discussion tools (see Figure 1). OV-2 captures operand flows and *needlines* specifying inter-process dependencies. OV-3 is a matrix that associates important oper-

and flows with important attributes. OV-4 depicts the relationships among organizations and operators. OV-5a decomposes higher level processes into more specific ones that are necessary for mission completion. OV-5b shows processes and operand flows among them, but focuses on the processes, complementing OV-2. OV-6a is meant to describe operational rules that govern the sequence and timing of processes. OV-6b uses state machine notation to depict processes or process sequences, critical sequencing, timing aspects, and additional details. OV-6c is the most detailed OV view, as it aggregates the information depicted in other OV views, including operand flows, processes, locations, and the associated sequencing or timing. Each diagram should depict either a specific scenario or a critical sequence of events.

Table 1. DoD Architecture Framework Viewpoints

Viewpoint	Purpose
All Viewpoint (AV)	describes the overarching aspects of architecture context that relate to all viewpoints
Capability Viewpoint (CV)	articulates the capability requirements, the delivery timing, and the deployed capability
Data and Information Viewpoint (DIV)	articulates the data relationships and alignment structures in the architecture content for the capability and operational requirements, system engineering processes, and systems and services
Operational Viewpoint (OV)	**includes the operational scenarios, activities, and requirements that support capabilities**
Project Viewpoint (PV)	describes the relationships between operational and capability requirements and the various projects being implemented
Services Viewpoint (SvcV)	the design for solutions articulating the Performers, Activities, Services, and their Exchanges, providing for or supporting operational and capability functions
Standards Viewpoint (StdV)	articulates the applicable operational, business, technical, and industry policies, standards, guidance, constraints, and forecasts that apply to capability and operational requirements, system engineering processes, and systems and services
Systems Viewpoint (SysV)	design for solutions articulating the systems, their composition, interconnectivity, and context for supporting operational and capability functions

Table 2. The DoDAF Operational Viewpoint

View	Purpose	Typical Representation
OV-1	High Level Operational Concept Graphic	Graphic, Picture, or Artist Impression
OV-2	Operational Resource Flow Description	Diagram
OV-3	Operational Resource Flow Matrix	Matrix
OV-4	Organizational Relationships Chart	Diagram
OV-5a	Operational Activity Decomposition Tree	Diagram
OV-5b	Operational Activity Model	Diagram
OV-6a	Operational Rules Model	Diagram or Table
OV-6b	State Transition Description	Diagram or Matrix
OV-6c	Event-Trace Description	Diagram

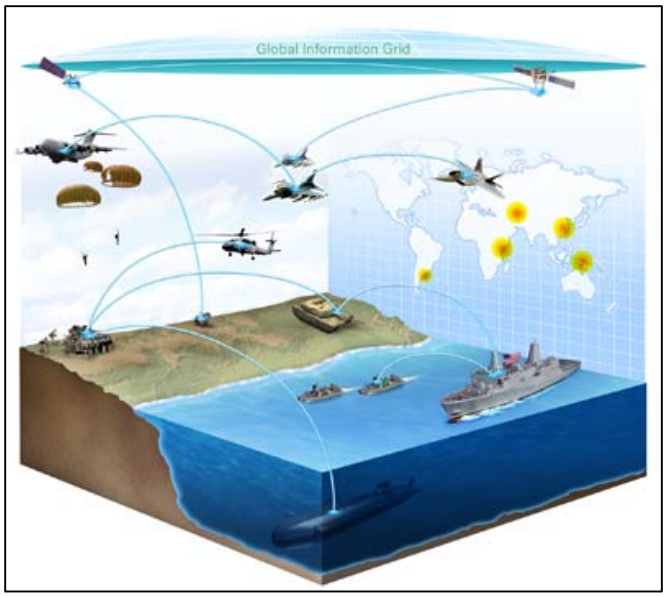

Figure 1. Example of a high-level graphical illustration of a ConOps. Source: [26]

3. OBJECT-PROCESS METHODOLOGY

OPM is a conceptual modeling language and model-based systems engineering paradigm for complex and dynamic systems and processes. OPM was standardized as ISO 19450 [19], [20]. OPM relies on the minimal universal ontology principle, whereby stateful objects (things that exist), processes (things that occur), and relations among them constitute a necessary and sufficient ontology for describing any conceivable system in the universe [19]. OPM's lightweight vocabulary includes ~20 terms.

OPM is visual and textual at the same time. The visual representation is a set of Object-Process Diagrams (OPDs), which are organized hierarchically. OPDs at all levels of the hierarchy retain and allow the same symbol notation, which makes it highly-consistent at all decomposition levels. Thus, OPM has only one kind of diagram. Structural, procedural, and functional aspects can reside jointly or exclusively within any OPD. Processes are represented by ellipses, objects by rectangles, and object states by rountangles inside the object rectangle. Objects and processes can be either informatical or physical, and either systemic or environmental (external to the boundaries of the system). Links express static and dynamic relations.

OPM's textual representation consists of sentences in Object-Process Language, OPL – a subset of English. Each sentence corresponds to an OPD construct – a set of linked things or states – and vice versa. Each OPD is accompanied by an OPD Specification (OPS) – a set of OPL sentences. The OPS is formal, machine-readable, and color-coded.

In this paper, we use OPM as our modeling language for several reasons. OPM has been suggested in the past for model-

ing operational enterprises, architectures, concepts, and scenarios, [18], [27]–[30]. It has thus been validated for this purpose. Additional properties that make OPM suitable for operational concept modeling are its minimal notation, bimodal graphical-textual representation, conceptual orientation, evolutionary nature, model executability, and standardization.

OPM's minimal notation consists only of stateful objects, processes, and relations, providing for simplicity, easy adoption and implementation, and high flexibility – without compromising on formality. OPM's robust ontology caters to both modeling and meta-modeling, which allows both ontology specification and instantiation.

OPM's graphic-textual presentation and automatically-generated OPL text specifications eliminate the need to spend precious time on describing diagrams textually. More importantly, it eliminates ambiguity and subjective interpretation of graphical drawings, which is extremely important in operational concepts definition and validation. DoDAF explicitly requires textual descriptions to accompany diagrams and other visual representations [12], and OPM provides a consistent and formal description to comply with this requirement. The text also serves as the basis for additional model analysis, e.g. coverage analysis, version differences, etc.

OPM's conceptual orientation means that it is first and foremost intended for high-level concepts, scenarios, and overall architecture. Thus it lends itself naturally to conceptual modeling of operational ideas and perceptions, and for the representation of operational concepts.

OPM is an ISO standard. As ISO 19450, OPM is situated for adoption as a meta-standard for standard authoring and protocol specification and will be easy to incorporate into existing and future standards and protocols. The process needed for standardizing a modeling language can take many years, and while the ability to standardize other modeling languages exists, ignoring an ISO-standardized modeling convention and 're-inventing the wheel' makes little sense.

There are two software tools for creating OPM models: OPCAT and OPCloud. OPCAT[1] [31] is a freely available desktop tool with built-in simulation capabilities, which has been used by thousands of academic and professional users around the world and utilized in hundreds of scientific papers over the last two decades, however it is based on obsolete desktop software technology, and its development has ended. OPCloud[2] [32] is a relatively new cloud-based modeling studio, which is still under development and evaluation, but has already been shown to be useful for various domains including medicine [33] and industry [34]. In this paper, we use OPCloud as a modeling tool and framework.

Figure 2 shows a screenshot of OPCloud while editing a model for a DoDAF-compliant architecture. The user interface includes a control bar (top abroad), a graphical OPD modeling area (center right), an OPL area (bottom right), a hierarchical OPD structure (center left), and a catalogue of objects and processes (things) in the model (bottom left). Modeling is performed by dragging a new object or process from the control bar or an existing thing from the catalogue into the OPD area. The OPL text is updated automatically in response to changes in the OPD.

The highlighted OPL sentence at the bottom of the OPL area corresponds to the highlighted OPD elements. The highlighted OPL reads: "**The Operational Viewpoint is an instance of DoDAF Operational Viewpoint.**". Accordingly, the objects **The Operational Viewpoint** and **DoDAF Operational Viewpoint** are highlighted.

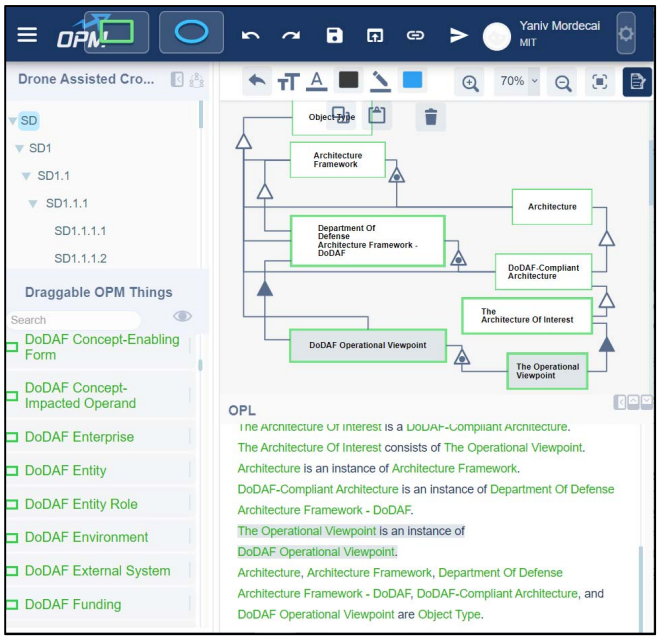

Figure 2. System Diagram of an OPM Model of an AoI, and its DoDAF-Compliant Operational Viewpoint

4. A Model-Based CONOPS Definition Framework

In this section we explain how OPM can be used to specify DoDAF-compliant Operational Viewpoints that would be informative, compact, and consistent. In fact, this approach is designed for future extension to encompass the entire scope of DoDAF and to support the implementation of all the viewpoints. The highlighted OPL sentence in Figure 2 – "**The Operational Viewpoint is an instance of DoDAF Operational Viewpoint.**" – captures the idea that the specific architecture described in this model is an implementation or subset of the DoDAF pattern and that, accordingly, the OV for the AoI is an implementation of the corresponding DoDAF-OV pattern. The framework consists of a pattern specification phase, and a pattern instantiation phase. We first describe how we generated a representation of DoDAF OV using the OPM modeling language and its available symbols and constructs. We

[1] OPCAT can be downloaded for free from http://esml.iem.technion.ac.il

[2] OPCloud is accessible on-line at https://opcloud-trial.firebaseapp.com/

then explain how the pattern portion of the model is instantiated to generate a specific implementation of the OV for a specific AoI – "The Operational Viewpoint". Using the definite (the word "*The*") emphasizes the specificity of the implementation, as opposed to the pattern's genericity.

The DoDAF OV Pattern Specification

Note that each pattern-related term in the model is always names "DoDAF <Term>". This means that: a) the object or process is an item in the DoDAF pattern, which can be instantiated any number of times and b) the defined term is a DoDAF Ontology term. The modeler can instantiate any typed DoDAF object or process as a specific architecture feature. ontological terms are also defined as specializations of either an **Object Type** or **Process Type**. This allows for retrieval of all the type things in the model. Typing can be direct, i.e., using the generalization-specialization link, or indirectly, via inheritance: the properties and specializations of typed things are also typed things.

Since we focus on operational aspects, we begin with defining the DoDAF OV as an ontological template of the model. Figure 3 shows an OPD of the DoDAF OV structure. This diagram lists all the OV's views, as indicated by the aggregation-participation link (black triangle) between the object **Department of Defense Architecture Framework – DoDAF** and each of the OV pattern objects, called "**DoDAF <OV-id> <OV-name>**". The diagram also defines the OV objects as typed architectural descriptions, as shown by the generalization-specialization link (blank triangle) between the object **Architectural Description** and each of the OV-# objects.

The purpose of the model is not to describe DoDAF, but rather to allow generating an actual architecture that complies with the DoDAF ontology. The use of the preamble "DoDAF" in conjunction with specialization of the special and reserved **Object Type** and **Process Type**, allows easier textual analysis of model representations – especially the OPL textual specification that accompanies the diagram structure. It allows for clear distinction between pattern elements and their instances in the actual architecture. In addition, it allows for easier analysis of associations, provided that each feature in the actual architecture is defined as an instance of a pattern feature. This approach resembles the "strongly-typed" software engineering approach, whereby software entities must always be clearly classified.

The next step is to specify the ontological terms mentioned in DoDAF for each operational view. A preliminary analysis that we have conducted indicates that there are 100 different words or phrases that could count as terms in the OV ontology, i.e., part of the OV 'vocabulary'. The complete list is provided in Appendix B. This list was built by careful analysis of the descriptions of the views in the DoDAF manual. Interestingly, none of the terms recur in all the views, and only a small number of terms recur in more than four different views: "operational activity", "location", "mission", "interaction", "resource", "resource flow", and "timing". The majority of the terms (65 out of 100) are mentioned in only one view as specific to the semantics of the view (i.e. important as part of the information that the view is intended to provide). This includes several variants or specializations of broader terms. For instance, "resource" is specialized into information, personnel, materiel, funds, etc.

Our proposed OPM-based ontology representation has a unique representation for each term, which is either an object, a process, or a construct of objects, processes and relations among them. The OV ontology mapping to the OPM ontology of stateful objects, processes, and relations among them, minimizes the required term space. Some DoDAF OV ontology terms cannot be defined by a single element but rather as a construct of two, three or more connected elements, or as a diagram depicting any number of elements. Yet, it is clear, after specifying the DoDAF OV ontology, that any DoDAF OV concept can be represented with OPM.

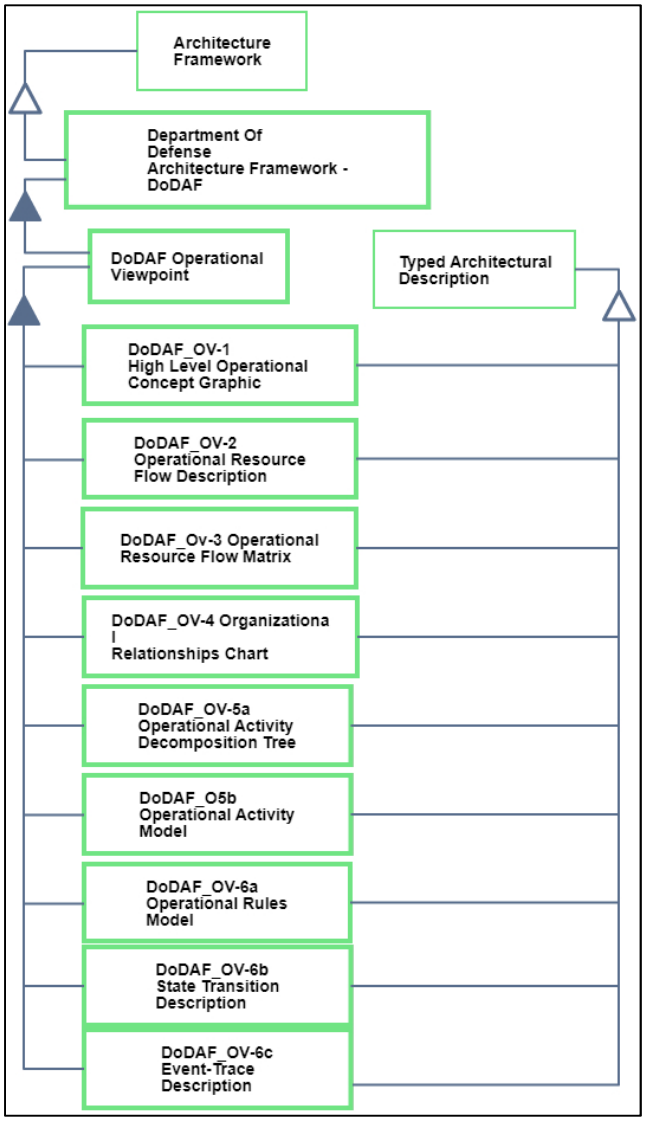

Figure 3. An OPM Model of the DoDAF Operational Viewpoint

We also attempted to find terms of similar or common semantics within DoDAF, and define additional hierarchies and associations that would result in a small set of reusable terms. For instance, "architecture", "system", "organization", "person", etc. are all commonly defined in our representation as "entity". Attributes like "location" and "capability" can be exhibited by any entity, but this characterization is done only once, at the highest level. Similarly, DoDAF mentions some items explicitly as "resource", but essentially many other ontology items are "resources": they can be transferred, allocated, and consumed by various entities. Thus, items like "operational plan", "mission", and "task", which represent different pieces of "information" exchangeable by entities, are in fact "information resources", amenable to any operation on resources defined above. This definition also eliminates the ambiguity around operational aspects that are defined both *about* the architecture and *within* the architecture, such as the mission that the architecture is asked to execute.

A useful ontology does not cover only the set of terms but also their relations. Therefore, the modeling process included the appropriate modeling of relations among the terms in a way which helps the operational analyst understand how to connect instantiated terms to each other according to the rules or guidance defined by the architecture framework. For instance, connecting *resources* to *activities*, or *capabilities* to *entities*, is done in several layers, e.g. using *needlines*, *resource allocations*, and *resource flows*, inter alia.

Since relations are essentially constructs, some complex relation patterns are defined using diagrams that capture the constructs, instead of a single link which may be easier to draw but not sufficiently informative. For instance, the *line of responsibility* and *needline* concepts are mapped to in-zoomed objects that capture multiple connections associated with the so-called "line", obviously allowing more depth and breadth than what a single link line. The DoDAF manual describes the *needline*, for instance, both as a simple line connecting two entities in a resource exchange matrimony. Thus, needlines can be captured in OPM using a tagged structural link with the phrase "needs X from". The link would be connect a resource consumer to a resource provider (Figure 4).

Figure 4. Simple Representation of the *Needline* term

There are several limitations in the simplified needline representation. First, it does not treat the resource (or in fact, resource bundle) as an object in its own right, which is especially important in a viewpoint that is focused on resource exchange. Second, it does not cover critical considerations such as the location of the resource relative to the locations of the consumer and provider. Third, the simplified needline link cannot be associated with any other element in the model – requirements, capabilities, missions, etc.

The extended representation we propose consists of an OPM construct rather than on a simple link. The needline is represented as a bona fide object, **DoDAF Needline**, and is then in-zoomed to reveal several relations and interactions, as illustrated in Figure 5. The DoDAF Needline construct is preferable for capturing the Needline idea due to several reasons:

A. The construct supports the Resource Bundle as a separate term, and as an ad-hoc aggregation of resources. This allows resources to be members of several bundles according to evolving circumstances. For instance, a drone can be provided to the operating unit by a drone distribution center, along with additional resources like a field terminal and additional batteries. In another context, the drone's *service* is provided as a resource to an intelligence or response unit. In both cases, the drone, which is both an entity and a resource, is allocated to the relevant consumer – first with the logistic and supporting infrastructure it needs to operate, and later with the payload and available output data it provides to the analysts.

B. The needline construct designates any entity as a consumer or producer of a resource bundle in the context of the specific needline. This allows the same entity to assume the role of consumer of one resource bundle, and that of producer of another resource bundle.

C. The construct accounts for the initial and final location of the resource, but the same location attribute is amenable to extension of possible intermediate states along the route of the resource. This is indicated in OPCloud by a small state-like hint inside the DoDAF Resource Location attribute, indicating additional states.

D. As an object, the needline can be associated with other terms and model items including requirements, stakeholders, capabilities, missions, scenarios, situations, etc.

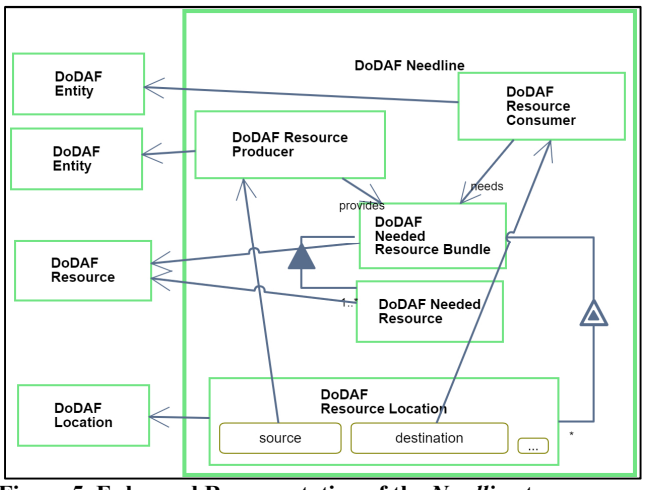

Figure 5. Enhanced Representation of the *Needline* term

A final example of an interesting pattern is the concept pattern. Crawley et al. define *concept* as an idea that maps function to form, and as the essential building block of the architecture [2]. The function of a system is a process that it performs in order to serve a purpose. In doing so it affects one or more operands. The form is any combination of structural elements that serve as an instrument for the function. Figure 6 shows the basic pattern of the Crawley Concept.

An architecture framework is essentially a guidebook for generating and recording concepts for complex systems. Several OV terms are expressly defined as concepts, such as ConOps and OpsCon. Since they are used interchangeably and confusingly in DoDAF, it is important to avoid confusion and provide formal patterns adhering to the ontology.

A Concept of Operations (ConOps) pertains to the Enterprise that uses or intends to use a given architecture to support its operations in order to reach some enterprise-level goal. The reason for the enterprise to need some architecture's capability to reach its goals is some environmental factor that prevents the enterprise from reaching its goals. This is illustrated in Figure 7. It was interesting to find visual and semantic similarities to Dori's notion of the System Diagram as the core of an untyped OPM system model [19]. The main difference between the two is the core process, which in our approach is the enterprise activity rather than the system's main function. The ConOps model also captures the notion that the environment stands between the enterprise and its goals.

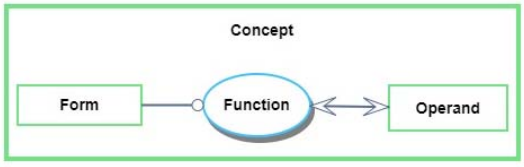

Figure 6. Basic Representation of a Crawley Concept [2]

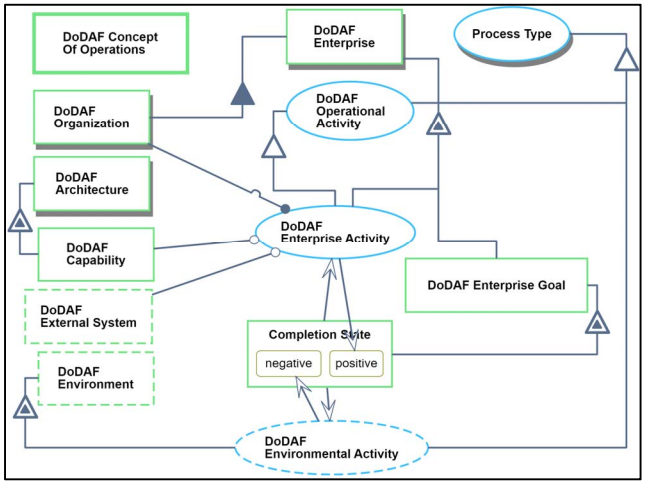

Figure 7. OPM Construct defining a Concept of Operations

An Operational Concept (OpsCon) pertains to the Architecture being analyzed, and therefore it is supported or enabled by a structural part of the architecture. The function that implements or realizes the concept is a function of the architecture. The operand that the function acts upon, and that the concept impacts, is best described as a resource, because the focus of the OV is resource exchange among operational entities. The pattern that defines the OpsCon according to the Crawley concept definition and in accordance with the DoDAF ontology is shown in Figure 8. This diagram maps each component of the OpsCon to a common term of the DoDAF OV ontology. With this pattern we can formally define and visualize multiple OpsCons to cover each part of the architecture whose way of fulfilling its purpose is not trivial.

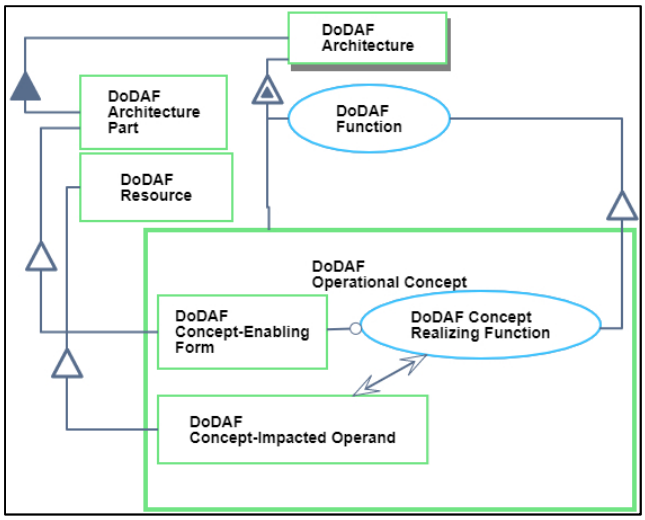

Figure 8. OPM Construct defining an Operational Concept

Each one of the nine views has a separate ontological diagram defining the patterns that are described as part of the view. Some of the patterns are not simple, apparently as a result of the fact that the DoDAF specification itself is text-based, and that it does not propose visualizations of the requested information. Constructing such visualizations could help DoDAF architects detect complications that are difficult to overcome in order to comply with the information visualization requirements of each view.

The **OV-3: Operational Resource Flow Matrix**, for instance, is specified in the OPD and corresponding OPL text shown in Figure 9 and Figure 10, respectively. OV-3 requires the operational analyst to cover *needlines* between resource productions and resource consumptions of *resource bundles*, *resource routing*, *resource attributes*, *interoperability requirements*, *capabilities*, and *locations* (places where activities take place). Obviously, a single matrix cannot contain all the possible connections and interactions among instances of the above terms. A set of diagrams that are based on a strong pattern definition, complemented by a formal textual specification in OPL, serves as an alternative collection of resource flow descriptions even if not necessarily visualized as matrices. It is also possible to organize these diagrams in a matrix-like layout, which may enhance readability.

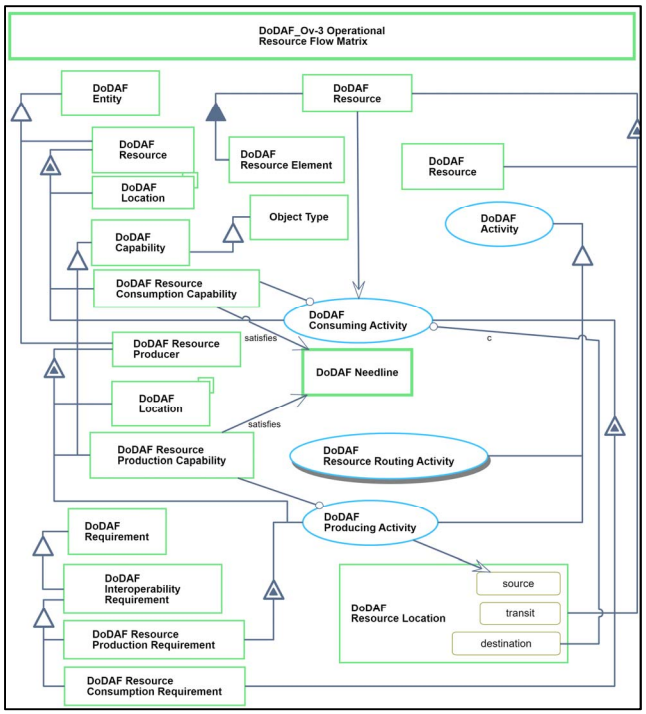

Figure 9. OV-3 Operational Resource Flow Matrix Ontology

1.	**DoDAF Resource Location** of **DoDAF Resource** can be **destination**, **source** or **transit**.
2.	**DoDAF Interoperability Requirement** is a **DoDAF Requirement**.
3.	**DoDAF Consuming Activity**, **DoDAF Producing Activity**, and **DoDAF Resource Routing Activity** are **DoDAF Activity**.
4.	**DoDAF Resource Consumption Requirement** and **DoDAF Resource Production Requirement** are **DoDAF Interoperability Requirement**.
5.	**DoDAF Consuming Activity** exhibits **DoDAF Resource Consumption Requirement**.
6.	**DoDAF Resource** exhibits **DoDAF Resource Attribute** and **DoDAF Resource Location**.
7.	**DoDAF Resource** consists of **DoDAF Resource Element**.
8.	**DoDAF Resource Consumption Capability** and **DoDAF Resource Production Capability** are **DoDAF Capability**.
9.	**DoDAF Producing Activity** exhibits **DoDAF Resource Production Requirement**.
10.	**DoDAF Resource Producer** exhibits **DoDAF Location** and **DoDAF Resource Production Capability**, as well as **DoDAF Producing Activity**.
11.	**DoDAF Resource Consumer** exhibits **DoDAF Location** and **DoDAF Resource Consumption Capability**, as well as **DoDAF Consuming Activity**.
12.	**DoDAF Resource Consumption Capability** satisfies **DoDAF Needline**.
13.	**DoDAF Resource Production Capability** satisfies **DoDAF Needline**.
14.	**DoDAF Producing Activity** of **DoDAF Resource Producer** requires **DoDAF Resource Production Capability**.
15.	**DoDAF Producing Activity** of **DoDAF Resource Producer** yields **DoDAF Resource Location** of **DoDAF Resource** at state **source**.
16.	**DoDAF Consuming Activity** of **DoDAF Resource Consumer** requires **DoDAF Resource Consumption Capability**.
17.	**DoDAF Consuming Activity** of **DoDAF Resource Consumer** occurs if **DoDAF Resource Location** of **DoDAF Resource** is at state **destination**, otherwise **DoDAF Consuming Activity** of **DoDAF Resource Consumer** is skipped.
18.	**DoDAF Consuming Activity** of **DoDAF Resource Consumer** consumes **DoDAF Resource**

Figure 10. DoDAF OV-3 Ontology OPL Specification

The complete OV pattern, consisting of the ontology representation for the nine views comprising the OV, is available as a shareable OPCloud model[3], which may be shared with relevant stakeholders, potential users, and interested readers.

Thanks to the robust and compact ontology of OPM, the diagrams are similar to each other, using and reusing the same model elements in different contexts. Theoretically, the entire ontology could be placed on a single OPD, thus showing the connections among the views and the shared terms. However, this would also generate a truly complicated diagram that would be difficult to follow and less usable, because eventually this would only be a pattern diagram and not an instance diagram. In this study we maintain the boundaries defined in DoDAF among the views. Future research can and will attempt to remove the boundaries, redefine some of the ontology, and eliminate redundancies and ambiguities.

The DoDAF OV Pattern Instantiation

Based on the complete OV pattern, we can instantiate a model of a specific AoI, in order to solve a specific problem or address an actual need. There are three possible instantiation approaches: a) separate models, b) overwriting the pattern model, and c) branching from the patten model. The details of each approach are described next.

A. Creating a separate model while referring to the pattern model as a general reference.

The first approach allows us an unrestricted, clean sheet model building, and is based on the common approach for modeling new systems in research and practice. This approach allows the modeler to generate a high-level outline of the system of interest quite fast, without the additional overhead. However, this easy start and quick deployment of an initial concept is devastating in the longer term – both to the model and the system, because it does not allow the modeler to maintain compliance to a special modeling standard or convention, and because it generates an unmanageable plethora of elements that have no rigorous domain-specific ontology to underpin the model. For example, we can quickly list relevant users, stakeholders, activities, and functionalities, but without tying these objects and processes to the corresponding ontology terms, the model will not be able to clarify the meaning of each element, and may lead to a critical confusion point in which the model might be abandoned.

B. Overwriting the pattern model terms with specific terms that embody the ontology term semantics.

The second approach is the exact opposite: it exploits the ontological model by redefining the ontology as the specific system model. While this approach allows for strong connection to the original ontology, and for completeness of the specific implementation, it is limited to very simple implementations that do not require more than one or two implementations for each ontology concept. Terms may be specified twice in the ontology in order to support the representation of relations between instances of the same term – such as organization, activity, etc. For instance, if our ontology lists the capability term only once, we will only be able to define one

[3] Please contact the main author to request access.

specific capability for our architecture. The de-generalized term used as the basis for the singular instantiation will no longer be able to serve as the basis for another instance, unless the modeler keeps a copy of the ontology as reference, which resembles the first approach.

C. Branching out the specific model from the pattern model.

The third approach is similar to the second approach in that it starts from the ontology model, but rather than greedily exploiting the existing model constructs, it instantiates each pattern or term for use. This principle must be observed in all cases, even if the modeler is 99.99% sure that there will be only one instance for some of the terms. The modeler would also need to consult the ontology before introducing any new feature into the instantiated architecture, in order to determine if it can be typed according to the ontology or not. Indeed, some features of the model may not be covered by the ontology. The upside of this approach is the assurance that the instantiated architecture always maintains compliance to the ontology, and that only minor extensions that are not covered by the ontology are included in the architecture. This approach is much closer to current engineering practices in software programming, hardware design, mechanical design, etc., which strongly rely on available building blocks, libraries, templates, reusable structures, and part catalogues. Obviously, this approach is applied in this study, and is gaining momentum as an MBSE paradigm.

In order to simplify and shorten the instantiation process, we have added placeholders for two-three view instances for each typed DoDAF view. The instantiation of the architecture framework takes place in System Diagram 1 (SD1) – an unfolded view of **The Architecture Of Interest** (AoI). Since legacy models have always started with the specific architecture or system of interest in SD1, we preserve SD1 for instantiated AoI models, and SD2, SD3, etc. for reference ontologies (including but not limited to DoDAF) and patterns. We expect future versions of OCPLOUD to support template importing that will allow separate management of template models and ad-hoc importing into SD2, SD3, etc. of any reference AF, methodology, ontology, etc. At present, a template model with SD1 as a placeholder for an instantiated AoI and SD2 as the reference ontology should be made available to users, to simplify the creation of additional AoI models, especially when the reference ontology may be extended (e.g., to capture additional DoDAF viewpoints or additional architecture frameworks) or evolved to cover updates and revisions to the original ontologies represented in the model.

Figure 11 shows an example of the instance architecture description – the OV-4 View Set, which consists of three placeholder view instances of the DoDAF OV-4 Organizational Relationships Chart. Each placeholder can be renamed as the specific architecture analyst sees fit, and as needed to cover the operational organizational aspects of the AoI. For instance, one chart can be a simple organization chart listing the various involved organizations, in a hierarchical structure, while a second chart can show the relationships among the organizations, and a third chart can show one or more command structures (e.g., one for routine and one for emergency or one for domestic and one for global missions).

Another chart can show the stakeholder requirements that each organization has defined for the architecture. Like each operational view, the **OV-4 View Set** also exhibits a corresponding **OV-4 OntologyMapping** object. **OV-4 OntologyMapping** is unfolded in a separate diagram in which the elements specified within the view and its sub-views are mapped to the ontology, be it DoDAF or another. The purpose of this ontology mapping diagram is to remove clutter from the main diagram presenting the useful information. Typing the **OntologyMapping** objects enables easy retrieval and analysis of ontology mappings across the model, e.g. for consistency, coverage, etc.

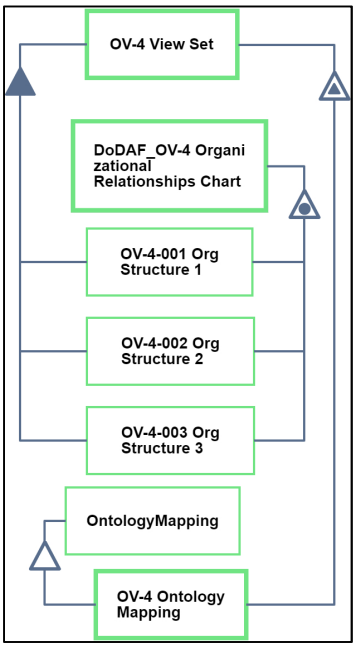

Figure 11. Instantiated Organizational Relationships View (OV-4) with three placeholders

There is no clear requirement in DoDAF for a documentation methods. The architect is therefore free to decide how to record the information, how to distribute it among the charts, and how to integrated the multiple aspects. The architect can easily switch to the ontology through the "show unfolded" option in the command halo of the OV-4 pattern model.

Figure 12 demonstrates the power of this approach, backed up by OPM's robust ontology and OPCloud's modeling capabilities. This figure shows a diagram of the instantiated architecture labeled **OV-5a-001 Activity Decomp 1**. While the name is kept generic for template purposes, the details inside this diagram indicate that this first instance of the DoDAF OV-5a Activity Decomposition view describes the command & control activity, which is generally part of any complex system at some level or another. The process Activity A100 Conducting Command And Control and five of its primary sub activities are instantiated from DoDAF Operational Ac-

tivity, while three supporting activities are specified as Logistic Support or Training Activities. This example also employs a second ontology, OODA Loop [35], which is not part of DoDAF OV, as a reference standard for command and control. This is both allowed and expected, since DoDAF does not provide reference architectures for any conceivable architectural notion.

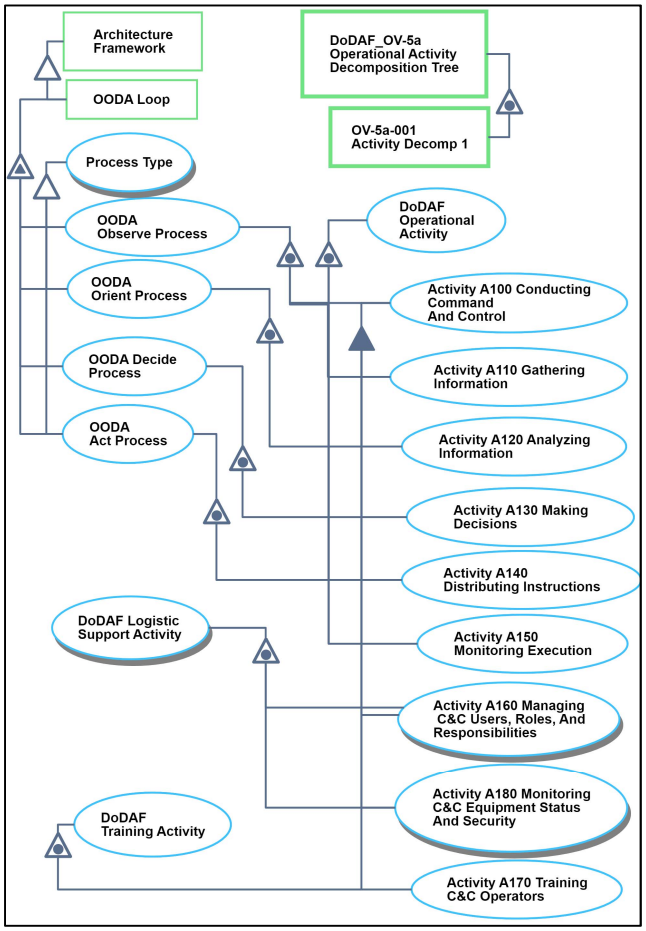

Figure 12. Instantiated Activity Decomposition View (OV-5a) for Command & Control activities, concurrently relying on DoDAF and the OODA Loop

The OODA Loop consists of four phases – Observe, Orient, Decide, and Act (OODA). In our instantiation, OODA Loop is a secondary reference ontology, and we instantiate its four terms into five command and control activities – one for each OODA phase, but also an additional "Observe" activity for monitoring execution. The OODA Loop can also be added to SD2 as one of several possible reference architectures for command and control that other operational architects can choose from, so that even if it is a convention, its adoption does not remain the preference or caprice of a specific modeler, as is often the case in untyped architecture models.

This example, demonstrating dual reliance on the primary DoDAF reference, as well as on a complementary reference such as OODA Loop, demonstrates the power of this approach in specifying an AoI while maintaining compliance with multiple standards and conventions. While specifying the pertinence of decomposed activities to the OODA Loop reference was not mandatory for DoDAF-compliance, it is used here to rationalize the decision by the operational architects to adopt this paradigm, indicating that it was not an unfounded decision. Similarly, we can rely on any ad-hoc reference architecture, standard, or protocol to rationalize resource allocation strategies, organizational and command structures, security and safety requirements, etc. This example shows that the DoDAF-compliant modeling does not prevent compliance with other standards and frameworks.

5. Example: Drone-Assisted Crowd Management

In this section we review a simple example of an operational solution for crowd management using drones. Crowd management allows public safety and emergency response agencies to observe and control crowd behaviors and dynamics in attraction areas such as large-scale public events, festivals, conferences, sporting events, and massive public facilities such as transportation centers and airports. Crowd management can be used for detecting and releasing bottlenecks, removing obstructions, opening and closing access ways, channeling crowds through desired routes or area, and detecting anomalies in crowd movement and behavior.

Drones are useful for taking photos and videos of the crowd and sending them to a ground terminal for further treatment. It is relatively easy to purchase and deploy an imaging drone, but this is where the operational issues start creeping in: what skills are required from the operator of the drone? How do you fly a drone above a crowd in a safe and secure manner and according to aviation authority regulations and recommended practices? Where is the information analyzed, and by whom? How do multiple drones act together to cover a wide area – should they be divided to sectors, or follow a path that covers the entire area in appropriate intervals? Can we use the drones to carry out crowd management tasks, e.g., carrying a loudspeaker to communicate messages to the crowd, carrying a communication relay to allow communication with connected devices on the ground, setting the drone to serve as a guidance device for crowds and field teams, and so on. Questions such as how it should be done, what are the training and logistic needs, etc. must be asked before setting out to purchase a large number of drones and operationalize their usage. Failure to do so may result in failure to support mission needs in the critical moments and significant waste of funds.

The purpose of this paper is to demonstrate a methodological framework and not to specify a complete ConOps for drone-assisted crowd management or an OpsCon for crowd-management drones. Such specifications should be done by stakeholders and may last for years and reach hundreds of pages of information. We therefore demonstrate a highlight of the framework for the case of drone-assisted crowd management, with the intention of persuading potential adopters that using this approach is beneficial in several ways: a) saving time and effort for modeling and specifying the architecture; b) assur-

ing quality and compliance to standard architecture specification frameworks and patterns; c) providing higher confidence to operational stakeholders that their needs, concerns, and conceptions are recorded, validated, and understood by the architects; and d) assuring architects that their specification will be informative, consistent, easy to understand, and maintainable, and therefore used, utilized, and implemented by engineering and development teams.

Drone-Assisted Crowd Management ConOps

The primary purpose of specifying the OV is to clarify and validate the ConOps. Figure 13 shows a diagram which is the instantiation of the OV-1 Concept Graphic, representing the ConOps according to the pattern defined in the Ontology. The Public Safety enterprise is the primary beneficiary of this architecture, which is intended to provide **Drone-Assisted Crowd Management Capability** to support the enterprise's **Crowd Management** activity. This activity is handled by various organizations, such as municipal services and authorities, law enforcement agencies, and emergency response agencies. The main goal of the enterprise is to assure **Crowd Safety**, while **Environmental Activity** and **Crowd Behavior** undermine this effort, which must constantly be preserved. In this example **Crowd Safety** is a goal of the **Public Safety Enterprise**, but even though it might seem like an attribute of the **Crowd**, it is not, since the **Crowd** as a collection of individuals does not exhibit an inherent, aware perspective on its collective safety.

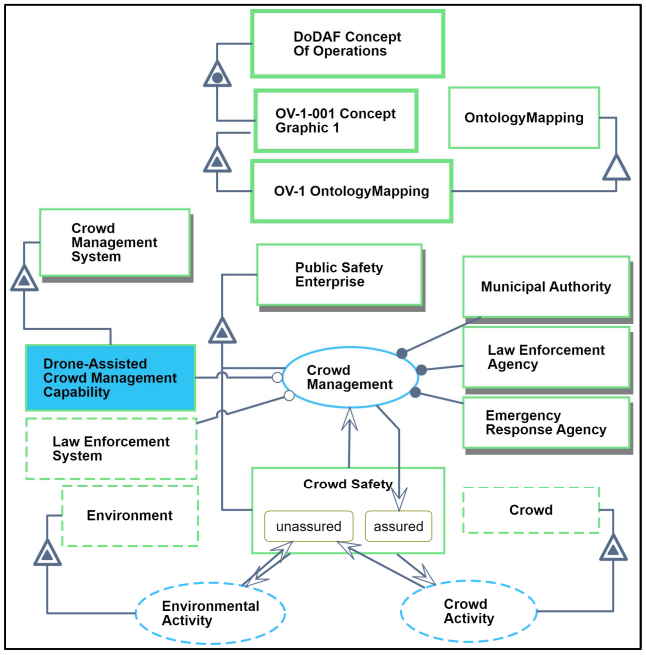

Figure 13. Instantiated Concept Graphic (OV-1) showing the ConOps for Crowd Management

In order to reduce clutter and unnecessary visual load, the typing of the ConOps elements is done in a separate Ontology Mapping diagram, which is an unfolded view of the object **OntologyMapping**. Figure 14 includes a subset of the OPL text of the Ontology Mapping diagram, which refers to the typing of the architecture elements according to the DoDAF ConOps terms. For example, **Crowd Management is an instance of DoDAF Enterprise Activity** simply specifies Crowd Management as the primary enterprise activity in this model. This set of sentences can be very useful for model analysis, verification, integrity and inconsistency checks, etc.

1. **Crowd Safety** can be **unassured** or **assured**.
2. **Emergency Response Agency**, **Law Enforcement Agency**, and **Municipal Authority** are instances of **DoDAF Organization**.
3. **Crowd Management System** is an instance of **DoDAF Architecture**.
4. **Drone-Assisted Crowd Management Capability** is an instance of **DoDAF Capability**.
5. **Crowd Management** is an instance of **DoDAF Enterprise Activity**.
6. **Crowd Safety** is an instance of **DoDAF Enterprise Goal**.
7. **Crowd Activity** and **Environmental Activity** are instances of **DoDAF Environmental Activity**.
8. **Crowd** and **Environment** are instances of **DoDAF Environment**.
9. **Law Enforcement System** is an instance of **DoDAF External System**.
10. **Drone-Assisted Crowd Management** is an instance of **DoDAF Concept Of Operations**.
11. **Public Safety Enterprise** is an instance of **DoDAF Enterprise**.

Figure 14. A Mapping of Drone-Assisted Crowd Management ConOps Terms to the DoDAF ConOps Ontology

In addition to providing a high-level ConOps view, the model should also describe the main resource needs and operational activities included in or supporting Crowd Management, as defined in the rest of the views of the OV. The first resource needed for crowd management is a ***Mission***. Figure 15 and Figure 16 specify the **Mission Assignment Needline** in graphics and text, respectively. A **Mission Assignment Needline** is a ***DoDAF Needline***, which means that it must account for a Needed Resource, its Producer and Consumer, and its source and destination locations. The specification of this needline asserts that there are two separate sub-organizations within the **Law Enforcement Agency**: an **Operations Center** and a **Crowd Management Unit**. A **Crowd Management Mission** is perceived as yet another sort of **Law Enforcement Mission**. In order to be assigned to the **Crowd Management Unit**, it must be *relocated* (in a virtual sense) from the **Law Enforcement System** to the **Crowd Management Mission Control System**, a newly specified subsystem that this solution will require.

The elaboration and specification of the OV for Crowd Management can continue in a similar manner by instantiating additional operational views according to their respective ontologies. It is important to associate items with their ontological stereotypes, preferably in an OntologyMapping diagram. The model is built to support as many instantiated views as the enterprise architect needs. The main difference between such a model and an untyped model is that the backbone for this model is the ontological decomposition and organization of viewpoints and views, while the backbone of the untyped model is the functional and structural decomposition of the system. Since an enterprise architecture may be multi-dimensional and multi-faceted, it appears that an ontological backbone is more suitable. However, it is clear that the OPM framework supports such a decomposition.

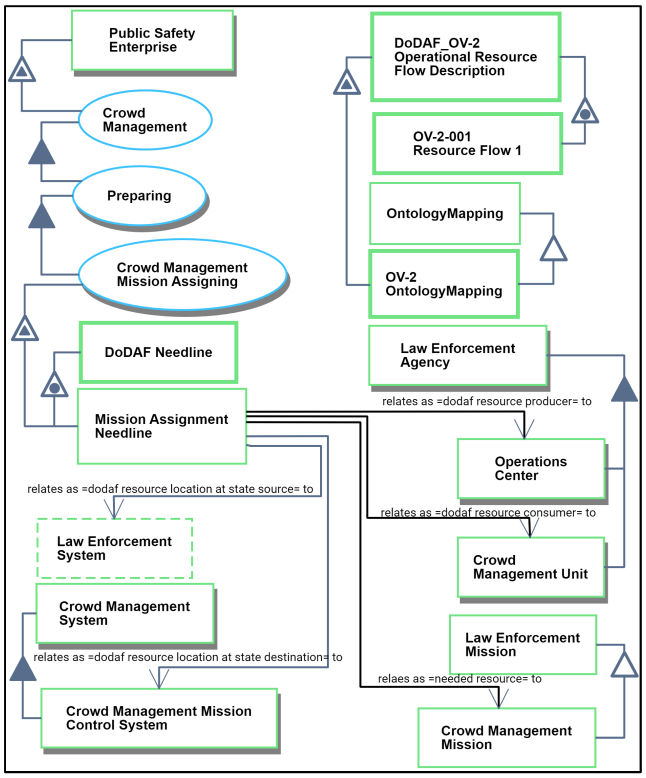

Figure 15. An OPM diagram of an instantiated Resource Flow Description (OV-2) showing the Mission Assignment Needline

1. **OV-2-001 Resource Flow 1** is an instance of **DoDAF_OV-2 Operational Resource Flow Description.**
2. **DoDAF_OV-2 Operational Resource Flow Description** exhibits **OV-2 OntologyMapping.**
3. **OV-2 OntologyMapping** is a **OntologyMapping.**
4. **Law Enforcement Agency** consists of **Crowd Management Unit** and **Operations Center.**
5. **Mission Assignment Needline** relates as =DoDAF Resource Producer= to **Operations Center**.
6. **Mission Assignment Needline** relates as =DoDAF Resource Consumer= to **Crowd Management Unit**.
7. **Mission Assignment Needline** relates as =Needed Resource= to **Crowd Management Mission**.
8. **Crowd Management Mission** is a **Law Enforcement Mission.**
9. **Crowd Management System** consists of **Crowd Management Mission Control System.**
10. **Mission Assignment Needline** relates as =DoDAF Resource Location at state destination= to **Crowd Management Mission Control System**.
11. **Public Safety Enterprise** exhibits **Crowd Management.**
12. **Mission Assignment Needline** relates as =DoDAF Resource Location at state source= to **Law Enforcement System**.
13. **Crowd Management** consists of **Preparing.**
14. **Preparing** consists of **Crowd Management Mission Assigning.**
15. **Mission Assignment Needline** is an instance of **DoDAF Needline.**
16. **Crowd Management Mission Assigning** exhibits **Mission Assignment Needline.**

Figure 16. An OPL Description of OV-2-001, Specifying the Mission Assignment Needline, its relations and components.

6. DISCUSSION AND CONCLUSION

This paper presents a novel and fresh approach to model-based architecture specification based on an architecture framework like DoDAF. Using OPM as a modeling language and the new OPM modeling tool OPCloud as a modeling vehicle, we were able to capture the entire ontology of the DoDAF OV, and instantiate some of it to demonstrate the application of the OV to describe ConOps for actual architectures and complex systems. Our approach is different from previous modeling approaches, including the traditional OPM modeling approach in that it emphasizes and observes explicit mapping of each architecture feature to the corresponding ontology term. It makes the model less straightforward, but more robust, as it allows for informed, contextualized, and ontologically-grounded model analysis, model quality assurance, consistency analysis, etc. While such analyses are beyond the scope of the current paper, we have shown that useful information can be extracted from the OPL text describing the model, such as a report on instantiated pattern terms. This allows, for example, to trace each actual element to a pattern element, and the implementation of each pattern element by architecture features.

There are several benefits to the proposed framework, as shown in the framework definition and demonstration sections of this paper (sections 3 and 4). First, OPM is a lightweight, minimal yet robust, and easy to learn modeling language, which makes the modeling process relatively easy, simple, and intuitive. Second, the model includes all the patterns, and the modeler has only to instantiate or classify the actual architecture's elements. Third, the OPL text that accompanies the diagrams is extremely useful for analysis, verification, validation, and even proofing. Fourth, the way the model is built it is easy and effective to display to stakeholders and decision makers and save time and energy editing the content before various important or critical reviews. Fifth, the model also supports the adoption of any additional standard, framework, or protocol – as shown with the inclusion of the OODA Loop ontology – and this allows the operational analysts to employ additional ontologies and terminologies supporting detailed feature decompositions.

Future research will focus on three directions. First, we plan to extend the framework to cover the entire DoDAF with the rest of its viewpoints. This will also allow for a complete representation of the OpsCon for the Solution Architecture, which complements the ConOps for the Enterprise Architecture. In addition, we plan to develop analytical tools that would help generate results and insights based on the OPL text. Finally, we intend to show that this approach is useful not only for describing architectures but also for comparing them to each other.

APPENDICES

A. ACRONYMS AND ABBREVIATIONS

AoI	Architecture of Interest
ConOps	Concept of Operations
CO-OP	Concept of Operations and/or Operational Concept
DoDAF	Department-of-Defense Architecture Framework
MBSE	Model-Based Systems Engineering
OPD	Object-Process Diagram
OPM	Object-Process Methodology
OPS	Object-Process Specification
OpsCon	Operations Concept
OV	Operational Viewpoint / Vie

B. ONTOLOGY TERMS OF THE DODAF OV

Below is a list of words and phrases identified as DoDAF OV ontological terms (its 'vocabulary'). There are 100 distinct terms including similar, overlapping or synonymous terms.

#	Ontology Term	Recurs	Views
1	action	2	OV-6a, OV-6c
2	activity	1	OV-6c
3	activity decomposition	1	OV-5a
4	activity model	1	OV-5b
5	activity sequence	3	OV-5a, OV-5b, OV-6b
6	activity state	1	OV-6b
7	activity-activity relation-ship	2	OV-5a, OV-5b
8	activity-resource alloca-tion	1	OV-2
9	architecture	1	OV-1
10	architecture aspect	1	OV-6a
11	architecture stakeholder	1	OV-4
12	architecture-environment interaction	1	OV-1
13	architecture-external sys-tem interaction	1	OV-1
14	capability	3	OV-5a, OV-5b, OV-6c
15	capability context	1	OV-2
16	capability requirement	1	OV-2
17	collaboration need	1	OV-2
18	collaboration requirement	1	OV-4
19	command structure	1	OV-4
20	concept of operations	1	OV-6a
21	condition	2	OV-6a, OV-6b
22	conditional imperative	1	OV-6a
23	constraint	2	OV-6a, OV-6b
24	consuming activity	1	OV-3
25	dependency	2	OV-5a, OV-5b
26	doctrine	1	OV-6a
27	event	2	OV-6a, OV-6b
28	event scenario	1	OV-6c
29	event-trace	1	OV-6c
30	external activity	2	OV-5a, OV-5b
31	geography	1	OV-1
32	guidance	1	OV-6a
33	human role	1	OV-4
34	imperative	1	OV-6a
35	information flow	2	OV-5a, OV-5b
36	input/output flow	2	OV-5a, OV-5b
37	interaction	4	OV-4, OV-5a, OV-5b, OV-6c
38	interoperability require-ment	1	OV-3
39	issue	2	OV-5a, OV-5b
40	line of responsibility	2	OV-5a, OV-5b
41	location	5	OV-1, OV-2, OV-3, OV-6a, OV-6c
42	logical data model entity	1	OV-6a
43	logistic support	2	OV-5a, OV-5b
44	mission	5	OV-1, OV-5a, OV-5b, OV-6a, OV-6c
45	mission aspect	1	OV-6a
46	mission class	1	OV-1
47	mission objective	1	OV-6c
48	needline	1	OV-2
49	op. activity	7	OV-2, OV-3, OV-5a, OV-5b, OV-6a, OV-6b, OV-6c
50	op. architecture	1	OV-2
51	op. aspect	1	OV-1
52	op. capability	1	OV-3
53	op. concept	2	OV-1, OV-2
54	op. facility	1	OV-2
55	op. plan	3	OV-2, OV-5a, OV-5b
56	op. rule	1	OV-6a
57	op. scenario	1	OV-1
58	op. situation	1	OV-1
59	op. thread	1	OV-6c
60	operation	1	OV-1
61	opportunity	2	OV-5a, OV-5b
62	organization	2	OV-1, OV-4
63	organization structure	1	OV-4
64	organization type	1	OV-4
65	organizational resource	1	OV-4
66	performance parameter	1	OV-1
67	person type	1	OV-4
68	player	1	OV-1
69	problem space	3	OV-2, OV-5a, OV-5b
70	process owner	1	OV-4
71	producing activity	1	OV-3
72	relationship	1	OV-4
73	requirement	2	OV-5a, OV-5b
74	resource	4	OV-2, OV-5a, OV-5b, OV-6c
75	resource - funding	1	OV-2
76	resource - information	1	OV-2
77	resource - materiel	1	OV-2
78	resource - personnel	1	OV-2
79	resource attribute	1	OV-3
80	resource element	1	OV-3
81	resource flow	4	OV-2, OV-3, OV-5b, OV-6c
82	resource transfer	1	OV-3
83	response	1	OV-6b
84	responsibility	2	OV-5a, OV-5b
85	role	2	OV-5a, OV-5b
86	rule of engagement	1	OV-6a
87	satisfied needline	1	OV-3
88	scenario	3	OV-5a, OV-5b, OV-6c
89	sequence	1	OV-6c
90	sequence diagram	1	OV-6c
91	service function	2	OV-5a, OV-5b
92	situation	1	OV-6c
93	state transition	1	OV-6b
94	stimulus	1	OV-6b
95	system function	2	OV-5a, OV-5b
96	task	2	OV-5a, OV-5b
97	timing	4	OV-5a, OV-5b, OV-6b, OV-6c
98	training need	2	OV-5a, OV-5b
99	trigger	1	OV-6b
100	workflow	2	OV-5a, OV-5b

ACKNOWLEDGEMENTS

We thank the MIT–Technion Post-Doctoral Fellowship Program for funding this research. We would like to thank Prof. Dov Dori from Technion – Israel Institute of Technology, inventor of OPM and director of the OPCloud development program, for useful comments and tips regarding the utilization of OPCloud for this study.

REFERENCES

[1] S. Gupte, P. I. T. Mohandas, and J. M. Conrad, "A survey of quadrotor unmanned aerial vehicles," *Conf. Proc. - IEEE SOUTHEASTCON*, 2012.

[2] E. Crawley, B. Cameron, and D. Selva, *Systems Architecture: Strategy and Product Development for Complex Systems*. Prentice Hall, 2015.

[3] *ISO/IEC/IEEE 29148-2018 - Systems and software engineering — Life cycle processes — Requirements engineering*. 2011.

[4] INCOSE, "Systems engineering handbook: A guide for system life cycle processes and activities." International Council on Systems Engineering. Published John Wiley & Sons, Inc., San Diego, CA, USA, 2015.

[5] NASA, *NASA System Engineering Handbook*, SP-2016-61. NASA, 2016.

[6] K. J. Viets and C. G. Ball, "Validating a Future Operational Concept for En Route Air Traffic Control," McLean, VA, USA, 1999.

[7] JPDO, "Concept of Operations for the Next Generation Air Transportation System," 2007.

[8] J. J. Scholte, H. A. P. Blom, J. C. (Hans) van den Bos, and R. B. H. J. Jansen, "Management of ATM performance in operational concept development and validation: A case study," in *Eighth USA/Europe Air Traffic Management Research and Development Seminar (ATM2009)*, 2009, pp. 373–383.

[9] U. Borkenhagen, N. Makins, and M. Valmorisco, "The 'European' operational concept validation methodology (E-OCVM)," *AIAA/IEEE Digit. Avion. Syst. Conf. - Proc.*, 2006.

[10] C. H. Fleming and N. G. Leveson, "Including safety during early development phases of future air traffic management concepts," in *Proceedings of the 11th USA/Europe Air Traffic Management Research and Development Seminar*, 2015.

[11] *ANSI/AIAA G-043A-2012 Guide To The Preparation Of Operational Concept Documents*. 2012.

[12] "DoD Architecture Framework V2.02," Washington, DC, USA, ASD(NII)/DoD CIO, 2009, 2011.

[13] D. Dori, "Conceptual Modeling and System Architecting," *Commun. ACM*, vol. 46, no. 10, pp. 62–65, 2003.

[14] A. M. Madni and M. Sievers, "Model-based systems engineering: Motivation, current status, and research opportunities," *Syst. Eng.*, vol. 21, no. 3, pp. 172–190, 2018.

[15] P. Korfiatis, T. Zigh, and M. Blackburn, "Developing a graphical CONOPS to enhance model-based systems engineering," in *2012 Industrial and Systems Engineering Research Conference*, 2012, pp. 2267–2275.

[16] R. Cloutier *et al.*, "Prototype of a Graphical CONOPS (Concept of Operations) Development Environment for Agile Systems Engineering," 2013.

[17] S. J. Herzig, D. Velez, B. Nairouz, B. Weatherspoon, R. Tikidjian, and B. Muirhead, "A Model-based Approach to Developing the Concept of Operations of the Proposed Mars Sample Return Mission," *2018 AIAA Sp. Astronaut. Forum Expo.*, no. September, pp. 1–15, 2018.

[18] N. K. James, "Concept of Operations and the DoD Architecture Framework," Massachusetts Institute of Technology, 2018.

[19] D. Dori, *Model-Based Systems Engineering with OPM and SysML*. 2016.

[20] "ISO 19450 Automation systems and integration — Object-Process Methodology," International Organization for Standardization (ISO), Geneva, Switzerland, 2015.

[21] H. B. Abrahamsen, "A remotely piloted aircraft system in major incident management: Concept and pilot, feasibility study," *BMC Emerg. Med.*, vol. 15, no. 1, 2015.

[22] I. Safi'i, A. Fariz, M. F. Rahman, O. Arifianto, R. A. Sasongko, and Y. I. Jenie, "The development of operational concept and Design Requirements and Objectives (DRO) of medical transport drone in Liukang Tangaya Archipelago," *J. Phys. Conf. Ser.*, vol. 1130, no. 1, 2018.

[23] S. Lingel *et al.*, "Methodologies for Analyzing Remotely Piloted Aircraft in Future Roles and Missions," 2012.

[24] D. G. Clark, J. D. Ford, and T. Tabish, "What role can unmanned aerial vehicles play in emergency response in the Arctic: A case study from Canada," *PLoS One*, vol. 13, no. 12, pp. 1–16, 2018.

[25] M. Hause, "Rebuilding the Tower of Babel The Case for a Unified Architecture Framework," in *INCOSE Internaional Symposium*, 2013.

[26] M. Parkinson, "How To Make My ConOps Clear and Persuasive," *24 Hour Company*, 2012. [Online]. Available: http://www.24hrco.com/images/articles/html/MikeParkinson_May12.html. [Accessed: 20-Oct-2019].

[27] Y. Mordecai and D. Dori, "Model-Based Operational-Functional Unified Specification for Mission Systems," in *10th Annual IEEE International Systems Conference (SysCon)*, 2016.

[28] N. R. Soderborg, E. F. Crawley, and D. Dori, "SYSTEM FUNCTION AND ARCHITECTURE: Opm-based definitions and operational templates," *Commun. ACM*, vol. 46, no. 10, pp. 67–72, 2003.

[29] M. Sitton and Y. Reich, "ESE Framework Verification by MBSE," *IEEE Syst. J.*, vol. PP, pp. 1–10, 2018.

[30] D. Dori, "DoDAF and its OPCAT Implementation," Technion - Israel Institute of Technology, 2006.

[31] D. Dori, C. Linchevski, and R. Manor, "OPCAT – An Object-Process CASE Tool for OPM-Based Conceptual Modelling," in *1st International Conference on Modelling and Management of Engineering Processes*, 2010, pp. 1–30.

[32] D. Dori, A. Jbara, N. Levi, and N. Wengrowicz, "Object-Process Methodology, OPM ISO 19450 – OPCloud and the Evolution of OPM Modeling Tools," *Systems Engineering Newsletter (PPI SyEN)*, vol. 61, pp. 6–17, 2018.

[33] N. Levi-Soskin, R. Shaoul, H. Kohen, A. Jbara, and D. Dori, *Model-Based Diagnosis with FTTell: Assessing the Potential for Pediatric Failure to Thrive (FTT) During the Perinatal Stage*, vol. 1. Springer International Publishing, 2019.

[34] D. Dori *et al.*, "OPCloud: An OPM Integrated Conceptual-Executable Modeling Environment for Industry 4.0," in *Systems Engineering in the Fourth Industrial Revolution: Big Data, Novel Technologies, and Modern Systems Engineering*, R. S. Kenett, R. S. Swarz, and A. Zonnenshain, Eds. Wiley, 2020.

[35] P. Eggenhofer-Rehart *et al.*, "C2 Conceptual Reference Model Version 2.0," in *NATO NEC C2 Maturity Model (N2C2M2)*, 2012.

BIOGRAPHY

Yaniv Mordecai *is a post-doctoral research fellow at the Massachusetts Institute of Technology, Engineering Systems Lab. He holds a Ph.D. in information systems engineering from Technion – Israel Institute of Technology, Israel (2016), and M.Sc. (2010, cum laude) and B.Sc. (2002) degrees in industrial engineering & management from Tel-Aviv University, Israel. He is a senior systems architect with Motorola Solutions Israel, and an adjunct lecturer and researcher at the Faculty of Technology Management at Holon Institute of Technology, Israel. His research interests include model-based systems engineering, cybernetics, interoperable systems, risk and decision analysis, and operations research. Dr. Mordecai is a senior member of IEEE and a Board Member of the INCOSE Israel Chapter. In 2017, he won the IEEE Systems, Man, and Cybernetics Society Outstanding Ph.D. Diploma Award in Systems Science and Engineering, as well as the OmegaAlpha Association's Exemplary Systems Engineering Dissertation Award.*

Nicholas "Nikko" K. James *is a United States Air Force officer. He earned his MS in Aeronautics and Astronautics at Massachusetts Institute of Technology, Cambridge MA, USA (2018). His research focused on analyzing Concept of Operations modeling using the DoD Architecture Framework. In 2017, Nicholas has been named the U.S. Air Force Cadet of the Year.*

Edward F. Crawley *is the Ford Professor of Engineering, and a Professor of Aeronautics and Astronautics at MIT. He has served as the founding President of the Skolkovo Institute of Science and Technology (Skoltech) in Moscow, founding Director of the MIT Gordon Engineering Leadership Program, Director of the Cambridge (UK) MIT Institute, and Head of the Department of Aeronautics and Astronautics at MIT. Dr. Crawley is a Fellow of the AIAA and the Royal Aeronautical Society (UK), and a member of the International Academy of Astronautics. He is a member of five national academies: Sweden, UK, China, Russia, and USA. He received an S.B. (1976) and an S.M. (1978) in aeronautics and astronautics and a Sc.D. (1981) in aerospace structures, all from MIT, and has been awarded two degrees of Doctor Honoris Causa. Crawley's research has focused on the architecture, design, and decision support and optimization in complex technical systems subject to economic and stakeholder constraints. His work ranges from underlying theory to models for real systems. His recent book – System Architecture: Strategy and Product Development for Complex Systems – was published by Pearson (2016)*