kompozition_

Notes on the Practice of Digital Mission Engineering and Model-Based Systems Engineering

Technical Report 01-25

Dr Dan Powell

March 10, 2025

Let's work smarter together

kompozition.com

FOREWORD

As the world becomes increasingly complex, with new technologies and capabilities being brought into service, enterprise organisations, Government agencies, and especially the ADF can no longer afford the effort and risk of applying legacy engineering approaches and methods to capability definition, acquisition and sustainment. Change is desperately needed to deliver ***'speed to capability'***. At the Mission Engineering level, outsourcing integration is no longer an option.

Traditional means of communicating intent and solution via PowerPoint, Visio diagrams, MS Word and Excel documents are failing us. This is especially the case when we try to describe systems of systems at scale.

At the mission level, the Defence sector needs to integrate, change, and/or modernise many legacy systems and subsystems of human design, many of which are complex and poorly understood systems. Fundamentally, if you intend to change your system, you must understand it.

For too long, enterprise organisations have sought to describe and communicate problems and intent using traditional documents and static drawings/views of complex systems. At scale, the distributed, disconnected, and discordant nature of these descriptions is consistently cited as a cause of acquisition and integration programs failing to deliver on time and budget and failing to meet the organisation's needs and intent.

This is a failure to communicate intent appropriately and for all parties to develop a deep, accurate, and shared understanding of the following aspects of intent:

- What we are trying to do,
- How well we are trying to do it, and
- Within what environment are we trying to do things?

In the attached technical report, Dr Dan Powell identifies and describes the challenges and limitations of current view-based approaches to Digital Mission Engineering and Model-Based Systems Engineering, supported by references to refereed and published works.

The report highlights the importance of understanding behaviour and describes a behaviour-based approach to developing integrated Digital Mission Models. It argues that integrated Digital Mission Models are sufficient for enabling analyses and supporting decision-making regarding capability gaps and changes, integration risks, and the risks to fitness for purpose and mission.

The approach and underlying technical mechanisms described in this report have been implemented in the Kompozition platform, a first-of-its-kind Digital Mission Engineering platform.

This work challenges the way the world thinks about systems. We have shared this report motivated by the recognition that the real challenge to implementing effective Digital Mission Engineering at the 'speed of relevance' is not a technical one; it is a cultural one, and the inertia of the status quo is all that stands in the way of Australia being at the forefront of the monumental change needed.

Michael Ninness
CEO, Kompozition
March 13, 2025

Notes on the Practice of Digital Mission Engineering and Model-Based Systems Engineering

Kompozition Technical Report 01-25

Dr Dan Powell

March 13, 2025

Abstract

Effective capability definition, acquisition, integration and sustainment require the ability to not only reason about what operational outcomes need to be achieved but to reason about what courses of action could be performed (what needs to be done), to what levels of performance (how well do they need to be done), and under what conditions (in what operational environment and state (when) do they need to be performed) to achieve those outcomes.

In this paper we discuss how a behaviour-based approach to developing models is not only fit for representing, validating and analysing threat/event-based scenarios, vignettes, mission threads and mission engineering threads, but that those representations enable the *synthesis* of much of the structural and compositional information required to provide significant acceleration and risk-reduction in Mission Engineering efforts intended to support capability definition, acquisition and sustainment.

1 Introduction

To have some estimable chance of achieving a desired outcome throughout the design process, we should start with a deep, accurate and shared understanding of the problem to be solved and, iteratively, the solution to that problem and the relationship between the two.

1.1 Mission Engineering

Mission Engineering: The US Department of Defence Mission Engineering Guide characterises *Mission Engineering (ME)* as an interdisciplinary process encompassing the entire technical effort to analyze, design, and integrate current and emerging operational needs and capabilities to achieve desired mission outcomes [1].

At the scale where Mission Engineering is applicable, the enabling System of Systems (SoS) integrates multiple cyber-physical systems with human and organisational systems, processes, doctrine, rules of engagement and other forms of physical, regulatory, and procedural requirements and constraints. The successful integration of systems at such a scale is dependent on understanding the interactions and relationships between the agent systems and resources in the end-to-end threads of behaviour that are expected to result in the accomplishment of the overall mission.

As highlighted in [2], Mission Engineering optimises mission outcomes over single-system solutions. As Mission Engineering must consider integrating emerging and future capabilities with extant, legacy capabilities, it must identify and manage the implicit constraints, requirements and uncertainties that come with that situation.

Mission Thread: Mission Engineering is based on the concept of a *Mission Thread*. For our purposes, we define a Mission Thread as a single, contiguous end-to-end thread of actions and interactions (including resource/information exchanges) by and between agent components of the system (cyber-physical, human, organisational, etc.), occurring under specific conditions, resulting in an effect in terms of a state change.

Mission Engineering Thread: When those actions and interactions are allocated to specific, individual (named) agents and resources that perform or enable an activity, task or part thereof, we refer to the end-to-end sequence as a *Mission Engineering Thread*.

The distinction between *Mission Threads* and *Mission Engineering Threads* is somewhat arbitrary. All activities are performed by some *Component Agent* at some level of detail. At the *Mission Thread* level, these *Component Agents* may be described as Operational Nodes, the Roles at those Nodes, Services, Organisations, Systems, Systems of Systems (ad infinitum), etc. At the *Mission Engineering Thread* level, the *Component Agents* may be the same agents at the Mission Thread level or refer to Individuals, Sub-systems, Sub-system Components, etc. refined to whatever level of detail is useful to support the purpose of the *Mission Engineering* effort.

Mission-Level Behaviour: We use the term *Mission-Level Behaviour* to refer to the sequential or concurrent, causal or temporal relationships between Mission Threads and/or their refining Mission Engineering Threads.

1.2 Digital Mission Engineering

Ring et al. [3] describe an activity-based methodology for developing *integrated* DoD Architectures. They define an *integrated* architecture as comprising the consistent set of information describing *Operational Activities* being performed by *Operational Roles* at *Operational Nodes* mapped to *System Functions* being performed by *Systems* at *System Nodes* with reference to (but not the presentation of) the *Information and Data* produced and consumed by those activities/functions. They state the requirement for *integrated*, *unambiguous* and *consistent* architectures as a precondition for conducting impact analyses, gap analyses, knowledge management for decision-making, etc. They identify that an approach to developing architectures based on the understanding of activities, their performers and their enablers, fosters the development, and in some limited cases, synthesis, of architectural information. In [4], the author of this paper presents such an activity-based approach using Behaviour Engineering and the Behaviour Modelling Language to synthesise integrated DoDAF architectures.

Digital Mission Model: We extend the definition of an *integrated* architecture in [3] to define the concept of a *Digital Mission Model*, a single consistent digital thread that integrates the following minimal set of information:

- The **composition** of the system of interest, enumerating the parts (component agents) of the system that act and interact in that system's behaviour (eventually including solution components),
- the **structure** of the system being changed, describing the relationships (including interfaces) between the parts of the system and between the system and its context,

- the **behaviour** of that system – the threads of action, interaction and exchange performed and enabled by parts/actors/agents, *using*[1] resources, under various contextual preconditions and events, achieving (with some probability) a change in state,
- the **relationships** between composition, structure and behaviour,
- the **constraints** and **requirements** (including mission metrics, performance parameters and requirements, etc.) that are imposed by context or intent on the composition, structure and behaviour, and
- any **uncertainty** implicit in the behaviour, components, constraints, requirements, and the relationships between these things.

We can then define the concept of *Digital Mission Engineering*.

Digital Mission Engineering (DME): Digital Mission Engineering refers to the process of using integrated *Digital Mission Models* (Digital Threads) to represent Mission Objectives and Metrics, Intent, Mission Threads, and Mission Engineering Threads with allocation to Performers, Enablers, Resources, Requirements, Constraints, and the Relationships between these aspects. Digital Mission Engineering requires and yields integrated models that enable the design and assurance of system solutions integrating technology, people, organisations, processes, etc., to optimise achieving positive mission outcomes through action and interaction while reducing risks that threaten those outcomes.

1.3 The Status Quo of Mission Engineering and Model-Based Systems Engineering

At the time of writing, most existing Mission and Systems Engineering approaches, including often-employed Model-Based Systems Engineering (MBSE) approaches, do not result in the development of *integrated* digital models. As Firesmith [5] recognises, many MBSE efforts are faced with challenges of distributed and disparate information, multiple architectures with multiple disconnected, often inconsistent and incomplete architectural views, compounded by information access and control issues across this distributed information. In many cases, this situation makes it infeasible to use these information artefacts as a model for the analyses required for effective decision-making in a Mission Engineering context.

These models were often created to directly represent the suggested views of architectural frameworks, which many have mistaken for a list of specific diagrams (to be pasted into documents and PowerPoints) without considering the actual information content of those views and the necessary relationships between them. Taking a decompositional (view-based or product-centric) approach to describing architecture often results[2] in inconsistent, incomplete and unintegrated architecture descriptions, and therefore, models that are not useful for supporting understanding, analysis and decision-making.

Consider the partial UAF Resources Process Flow diagram (NAFv4 L4, DoDAFv2 OV-5) representation of a Mission Engineering Thread in Figure 1. This view is taken from the US DoD Mission Architecture Style Guide [6], a document intended to be an appendix to the Mission Engineering Guide to provide guidance on the development, presentation and analysis of model-based mission architectures. It is evident from this view that some of the properties required

[1] We use the term *using* in reference to "behaviour" and "resources" to encompass the production, consumption, processing, transformation, interrogation and exchange of those resources (including information and data) as an effect of the behaviour being performed

[2] Decomposed views of information are generally composable. However, the composer must always hold in mind information regarding many aspects of the system and their relationships at any one time (they must represent the Digital Mission Model mentally before they create it). Unfortunately, this is an untenable burden on human short-term memory for most, at a relatively small scale.

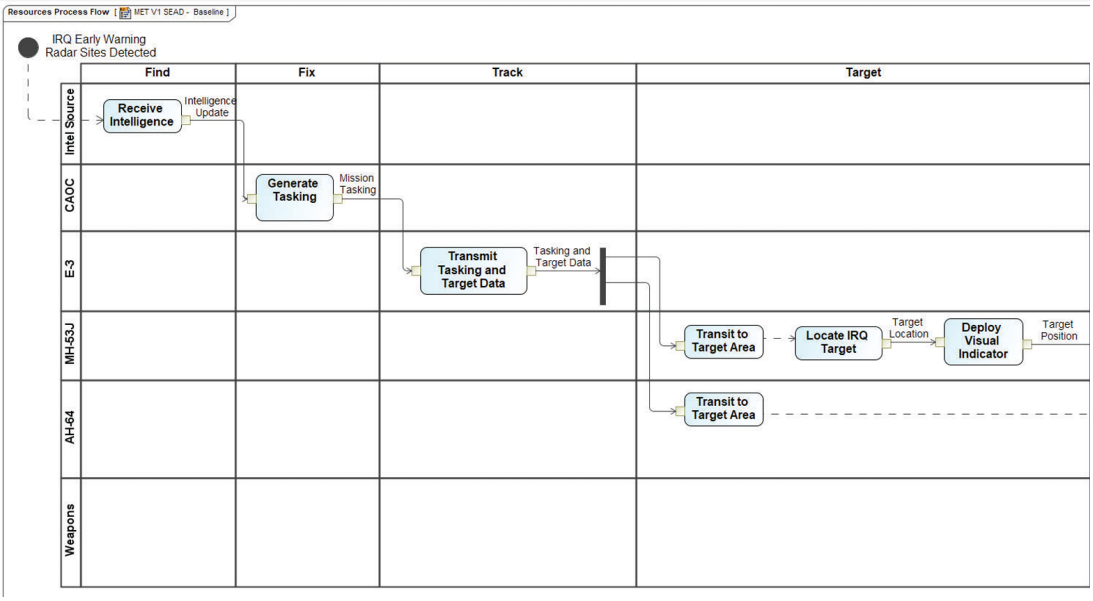


Figure 1: Baseline Suppression of Enemy Air Defence Mission Engineering Thread

of an integrated architecture are missing or have been violated. This leads to an architecture whose products are appropriate for inclusion in a document but do not comprise an integrated architecture useful for analysis and decision-making.

Below we list some of the issues within and between views of this style guide as an exemplar of the kind of common architectural issues that limit a model's use in a Digital Mission Engineering decision-making context. Where possible, we use terminology from the guide[6].

- The vocabulary used in the diagram's *Initial Node* is inconsistent with previous views in the document. This node depicts the triggering event (starting conditions) for the behaviour where one or more "IRQ Early Warning Radar Sites" have been detected. The Order of Battle (depicted as a UAF Resource Structure diagram in [6]) does not enumerate an "IRQ Early Warning Radar Site", the name used in that view is "Early Warning Radar (EWR) Site". While trivial, this shows a simple inconsistency of vocabulary in the model-based architecture that could limit the architecture's use as a *Digital Mission Model*.

- Some of the information and data elements on the *Function Object Flows* (Item/Information Flows) between activities/tasks in the Mission Engineering Thread are not represented in any other views. That is, they are undefined other than their use in this view. This is possibly due to an Information Model (NAFv4 L7, DoDAFv2.02 DIV-2) not being presented in the style guide. The omission of resource (information and data) structural views with trace to the behaviour that *uses* those resources is considered a risk to the model being able to support the analysis of Mission Engineering Threads. In this case, "Target Position" looks to be produced by an activity called "Deploy Visual Indicator" from the input of "Target Location". Assuming that "Target Position" and "Target Location" are different information concepts (difficult to tell as they are both undefined in the Resource Structural views), "Target Position" appears in no other views other than the alternates and extensions to this Mission Engineering Thread.

- The above is also reflected in the naming of the activities (Function Actions). As recommended by almost all MBSE guidance, these activities are named using *verb phrases* of the form `Action Object*` with the `Subject` being identified by the horizontal swimlane (according to the guidance within [6]). The information contained within these phrases should also be consistent with the remainder of the model. For example, "Transit to Target Area" is an action that requires the existence of a resource called "Target Area", "Locate IRQ Target" introduces the entity "IRQ Target" which appears to also be referred to as "Target" within the same view. A complete, consistent, integrated model would represent these resource/information entities and would link those entities to the behaviours that use them. Based on the information available in [6], this is not the case. The failure to integrate domain resource models with the models of the behaviours that use them imposes limitations on the model's usefulness for supporting effective analysis and decision-making.

Other guidance for developing operational-level models, such as that provided by [7] and [8] compound these issues of model understandability and subsequent use by promoting the development of multiple, largely structural behavioural views instead of integrated, dynamic behavioural views. These Architecture Frameworks' view-based approaches result in the development of vocabulary inconsistencies similar to those outlined above. Additionally, to get an understanding of integrated *Mission-Level Behaviour*, each reader must be able to consume and internalise the relationships between multiple behavioural views, usually:

- L2-Logical Scenario (DoDAF OV-2) and L3-Node Interaction (DoDAF OV-3) views identifying interactions between nodes,
- L4-Logical Activity (DoDAF OV-5a,b) and L6-Logical Sequence (DoDAF OV-6c) views describing the causal, informational and temporal flows of behaviour, and
- L5-Logical State (DoDAF OV-6b) views describing the states and state transitions of the component agents.

Even if these views were to be developed and maintained as consistent artefacts, the distribution of behavioural information makes developing a shared understanding (particularly with non-technical stakeholders) difficult, limiting the use of such models to support capability decision-making.

Neither [6] nor [7] provide an exemplar structural resource model describing the Resource entities (including Information and Data) and their Relationships *used* within the Mission-Level Behaviours. This would typically be represented in a NAF L7-Information Model or a DoDAF DIV-1/2 Conceptual/Logical Data Model. The omission of such a view in these guides indicates models that are incomplete with respect to the definition of a *Digital Mission Model*. The state of the resources produced, consumed, processed, transformed, observed, interrogated, and exchanged by a behaviour affect or are affected by that behaviour. Integrating resource models with behavioural models is likely a precondition to making effective capability decisions about that behaviour.

This brief analysis of some published examples is entirely based on the architectural models published in [6, 7, 3]. The actual architectural models (not the document versions) may relate the disparate and undefined resource, information and data elements, activities, states and performance parameters to enable analysis. Even if this is the case - and we consider this unlikely based on the prevalence of such issues in real-world modelling examples - the above discussion illustrates the folly of attempting to communicate integrated architectures in documents and slides instead of in integrated models.

2 Behaviour-Based Approaches to MBSE

The main thesis of this paper is that taking a behaviour-based approach to modelling, one that focuses initially on creating an expressive, human-readable (therefore validatable) yet formal representation of Mission Threads and Mission Engineering Threads, can yield *Digital Mission Models*, at the speed of relevance for what is being modelled, that enable:

- analyses such as gap and impact analyses and support decision-making (through executable or simulated behaviour[3]),
- rapid capability definition from the description of the behavioural/functional, performance expectations, constraints, measures and other requirements in the Digital Mission Model. Artefacts representing capability definition and their relationship to mission become an integrated part of the Digital Mission Model,
- capability acquisition at the intersection of intent and operational need/requirement, architecture and design assurance, test and evaluation, integration and deployment, and
- mission-focussed capability sustainment through the linking of configuration item level enablers and performers to the behaviours they enable, traced to the mission level behaviours and their contribution to objectives.

Behaviour-based approaches are also useful when a component agent's structure and internal behaviour are unknown. Behaviour-based approaches can describe the requirement for interaction and exchange between *black-box* components.

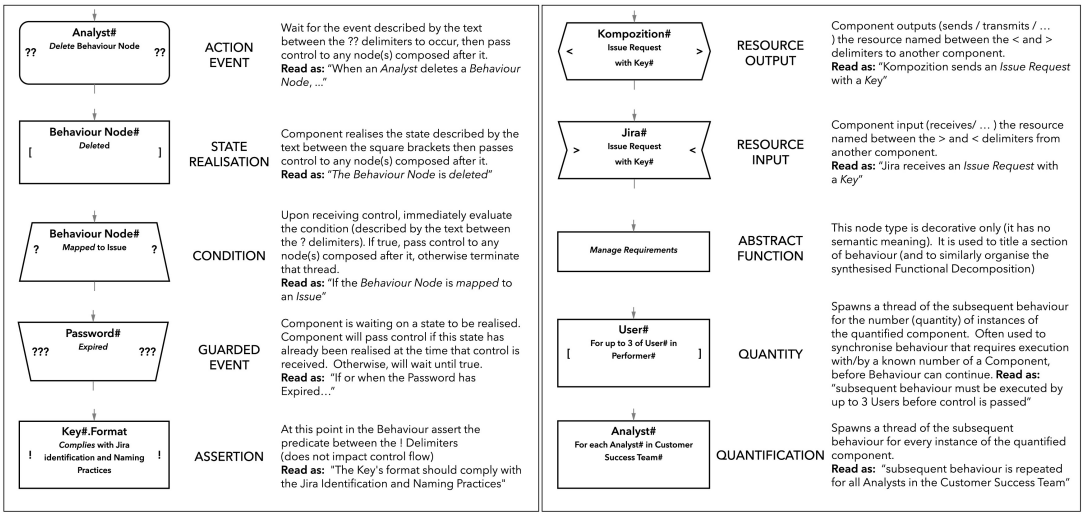
2.1 Representing Dynamic Integrated Behaviour in the Behaviour Modelling Language

Behaviour Engineering [9] is a discipline enabling the constructive and incremental modelling of the composition, structure and behaviour exhibited by the agents and elements of a problem space. The Behaviour Modelling Language (BML), the formally grounded graphical language supporting Behaviour Engineering, is expressive, human-readable and supportive of the incremental production of completely integrated and consistent behavioural, compositional and structural views from natural language descriptions of the problem space and intent. These integrated models characterise not only the interactions between elements of the problem space but also the consequences that the behaviour of those elements and their environment have on these interactions, enabling the ability to reason about a system as more than just an aggregation of parts and interfaces (what the system is) - but in terms of what that system does and what effect that has on the system and contextual state.

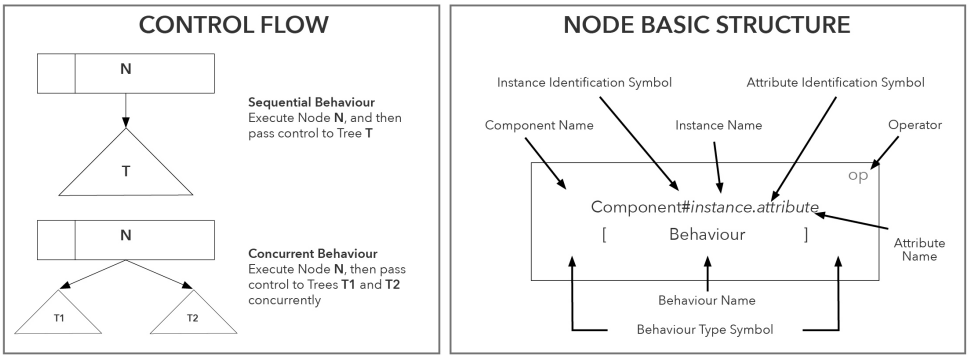
For decades, the Behaviour Engineering method has been successfully applied to the analysis of large-scale and complex specifications and their resulting systems (and Systems of Systems). The ability to constructively build wholly integrated, human-readable models of such large and complex systems has effectively reduced risk in programs. Facilitating understanding and communication has resulted in the early identification of large numbers of major issues related to capturing and preserving intent (requirements defects) previously missed in such projects [10]. It was recognised [11] that such applications of modelling intent at scale can prevent enormous waste in terms of projects failing to deliver to intent and/or budget and schedule.

The Behaviour Modelling Language (BML) represents individual behaviours and the integrated dynamic behaviour of interacting Components in a Behaviour Tree. It represents state

[3]The investigation of executable and simulated behaviour will be the subject of a future paper



(a) Behaviour Nodes



(b) Behaviour Node Detail and Behaviour Composition

Figure 2: Behaviour Modelling Language Node Syntax and Behaviour Composition

transition, action events, events of state, conditions, resource flow, control flow, threads and non-functional constraints in a single view. The behaviour tree enables the constructive integration of behaviour fragments into a single model, using sequential, concurrent and guarded compositions to represent all end-to-end threads of behaviour of interest within the problem space. Figure 2 gives a summary of the key elements of the Behaviour Modelling Language notation.

2.2 Example

Consider the image in Figure 3. This is a Behaviour Modelling Language representation of the Initial Node and Find activities depicted in Figure 1.

> When an Early Warning Radar Site (part of the IRQ (Red) Order of Battle) is Detected, The Intel Source receives Intelligence and then transmits Intelligence Updates to the CAOC.

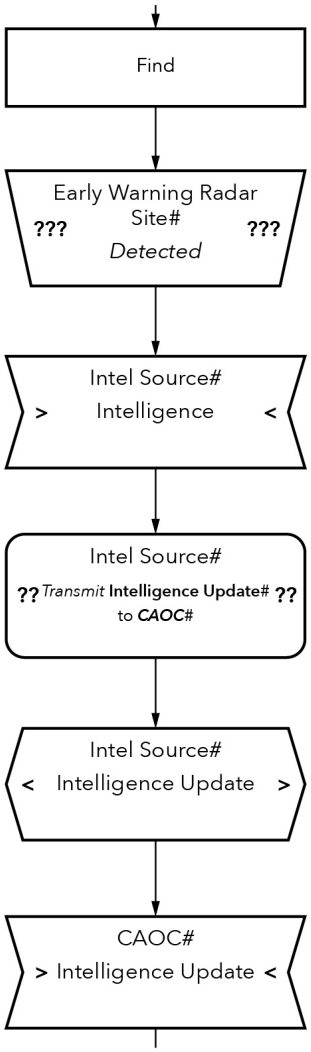


Figure 3: Example Behaviour Tree representation.

The following discussion and figures illustrate that information implicit in a Behaviour Modelling Language representation can be used to synthesise information required of an *Digital Mission Model*.

2.2.1 Generated Activity/Function Decomposition and Allocation

The action events in the Behaviour Tree directly describe the activities/functions being performed, their allocation to performers and their sequence. In the example, the activites/functions `Receive Intelligence` and then `Transmit Intelligence Updates to the CAOC` are automatically identified, organised as part of the `Find` function and allocated to a performer called `Intel Source` (refer Figure 4). This is a representation of the information content of a DoDAF

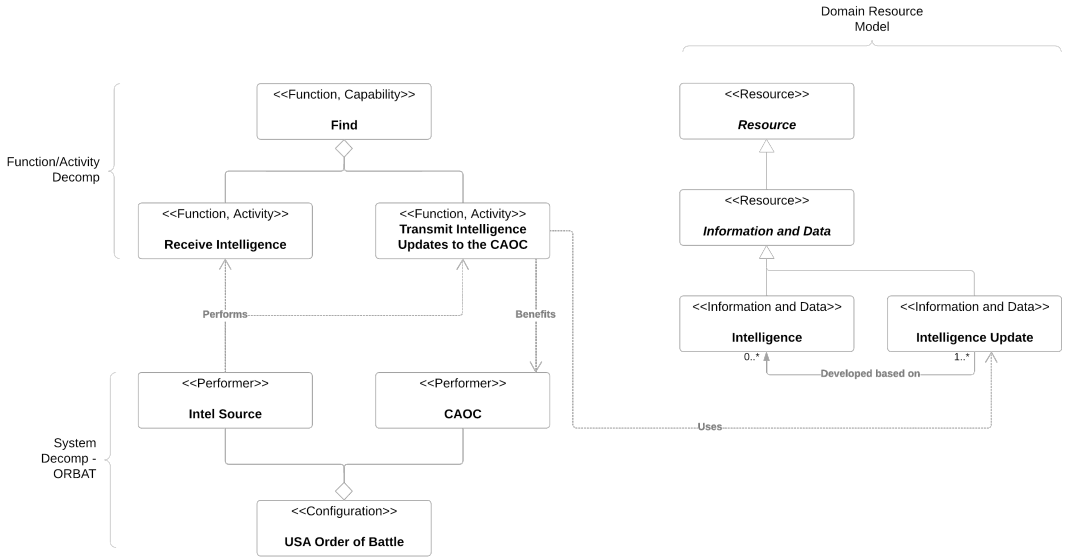


Figure 4: Partial Functional Architecture and Domain Resource Model synthesised from Behaviour in Fig. 3.

OV-5a - Operational Activity Decomposition Tree. Where a Behaviour Tree represents the behaviour of lower-level entities, the view synthesised would represent the corresponding DoDAF System and Service views.

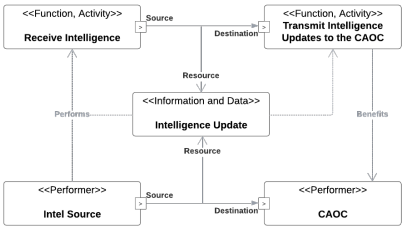
The resources (including information and data) produced, consumed, processed, transformed, observed, interrogated, and exchanged by that activity/function, and the relationships between those resources are able to be extracted from the description of behaviour as shown in Figure 4. `Intelligence` and `Intelligence Updates` are identified as Resources, added to the ontology, related based on the behaviour[4] and linked to the activities/functions that use them. The generated information shown in Figure 4 extends the information content of a NAF L7 / DoDAF OV-1/2.

2.2.2 Generated Event-State Transitions and Information Exchange Requirements

There is sufficient information implicit in the description of a BML action event to be able to synthesise the description of the expected resultant state. In the example of Figure 3, the postcondition state is generated as the `CAOC` having *Received* the transmitted `Intelligence Update` as a consequence of the *Transmit* action event. To maintain the integrated status of the model, the state is generated against the CAOC and linked to the causal *Transmit* action event.

In this example, the action *Transmit* implies a likely need for a resource/information exchange. This information leads to synthesising the I/O behaviour nodes following the action-event in Figure 3. The information exchange behaviours are linked to the respective function, performers and resources for consistency. This behaviour implies the existence of interfaces to enable the information exchange at the functional level and between the performers. These interfaces are synthesised and associated with all relevant model artefacts as shown in Figure 5. These in-

[4]For brevity and inclusion in this document, behavioural assumptions leading to these depicted relationships have been omitted from the representation of behaviour



ID - Name	Source	Destination	Resource	Characteristics and Constraints	Requirement Trace	Issues	Status
- Intel Source to CAOC - Intelligence Update	Intel Source	CAOC	Intelligence Update	*Not characterised*	UAF-ME-SRC-0001	**UAFME-ISS-1**: Where do Intelligence Updates come from? [Major, Incompleteness, Open] Modellers asumption is that Intelligence Updates are developed based on the received Intelligence.	Proposed

Figure 5: Synthesised identification and allocation of a proposed Interface linked to the behavioural information exchange.

terfaces can be subsequently characterised/designed/refined with recognition of their dynamic behavioural context.

2.2.3 Managing Vocabulary

In the example of Figure 3, the precondition (triggering event) to the Mission Engineering Thread states that an instance of an `Early Warning Radar Site` has been *Detected*. As mentioned above, the source for this model fragment refers to an "IRQ Early Warning Radar Site", and in the ORBAT of the source [6] "Early Warning Radar (EWR) Site".

In order to create integrated models, we must be able to resolve and manage vocabulary inconsistencies. The assumptions made to manage vocabulary must be a part of the integrated model and linked to the record of any aliases and all related model artefacts.

2.2.4 Provenance

For provenance, the Behaviour Model and all synthesised and linked information should be traced to a managed model artefact that represents the source from which the model was produced. Even if no documented source is identifiable and a model is produced through workshops, interviews or similar, a record of the workshop or interview should be created and linked to the behaviour and all related model artefacts. The artefact representing the source is linked to the behaviour and all synthesised model artefacts to enable provenance to be determined, regardless of the view taken.

2.2.5 Managing Uncertainty

Many models are created from source information that is incomplete, inconsistent, or otherwise ambiguous. For a model derived from such sources to be useful, the ambiguity of the source information must be recognised, and *Issues*[5] must be recorded and linked to the source information and all relevant model artefacts, so that uncertainty may be considered at all stages of model development and model use. The uncertainty posed by the issues can then be managed. The

[5]We use the term "Issue", but feel free to use other terms such as "defect", "opportunity for improvement", etc.

linking to synthesised artefacts is important as the managed issues that have not been resolved represent residual uncertainty and risk related to using those model artefacts.

2.2.6 Growing the Tree

Consider the issue that the relationship between the Intelligence received and the Intelligence Update being subsequently transmitted is not easily understood from the model. Such an issue would be resolved using the model, closing the issue (ideally through consultation with appropriate subject matter experts). The resolution of issues is often an iteration of an elicita-

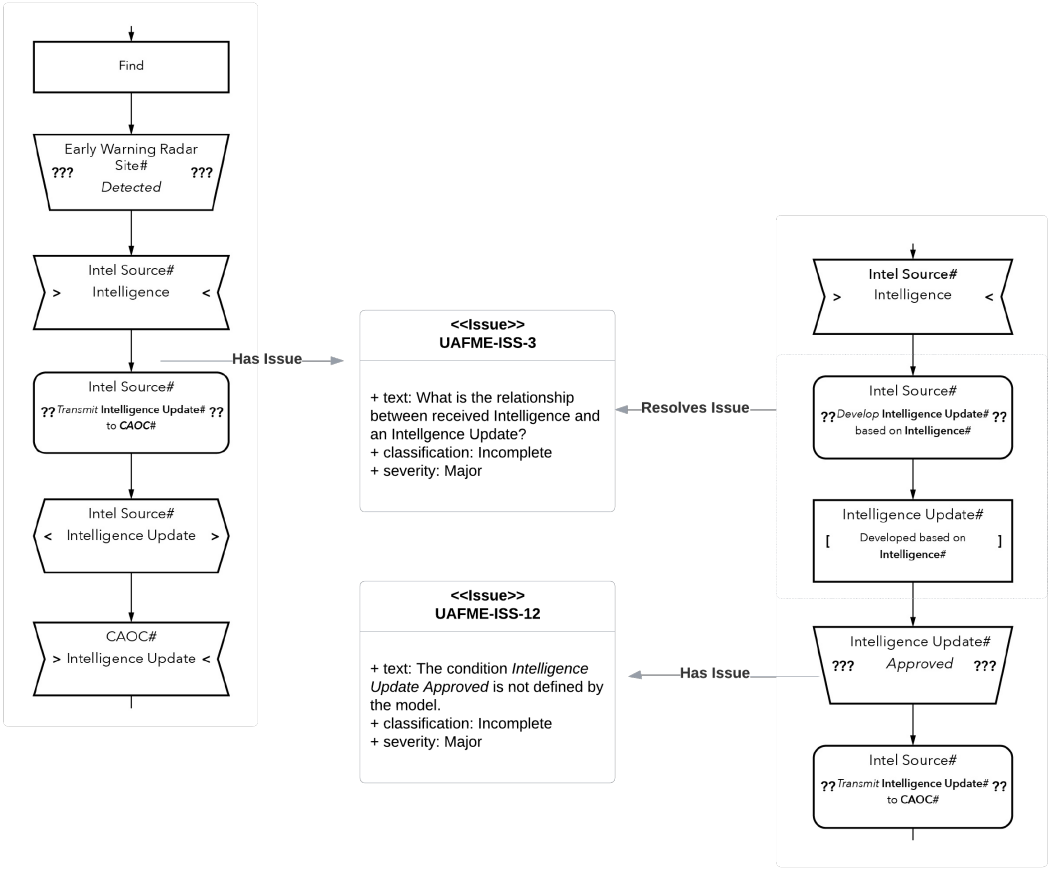


Figure 6: Example of resolving an issue through elicitation of behaviour.

tion/elaboration-modelling-validation or refinement loop.

Elaboration: We use the term *elaboration* to refer to adding information for clarification of intent without deferring to a discussion of *how* lower-level components might act and interact to achieve that intent. In the language of classical systems engineering, elaboration expands on the ”what”, not the ”how”.

Refinement: We use the term *refinement* to refer to the practice of allocating all or part of an intentional behaviour description to a description of behaviour at the next lower level of

abstraction. The process of transforming *Mission Threads* to *Mission Engineering Threads* can be achieved through a process of refinement, where direct allocation to behaviour is insufficient for modelling purpose.

In the example shown in Figure 6, the uncertainty of where "Intelligence Updates" come from is resolved through elaboration, adding information that describes that after receiving Intelligence, an Intel Source will develop Intelligence Updates, which can then be transmitted to the CAOC. For the purpose of illustrating the requirement for iteration in elaboration processes, we have introduced a fictitious rule that Intelligence Updates must be "approved" before transmission - a state that is undefined according to the integrated model. This new issue may, in turn, be resolved. This process of identifying incompleteness in knowledge and elaborating in the model to fill the gap is often referred to as "growing the tree".

While possibly not an essential property of a *Digital Mission Model*, the elaborateness and fidelity of a model are likely proportional to the usefulness of that model for supporting analyses and information synthesis.

In practice and at scale, many issues identified in models remain unresolved while the model is being used. This is often due to the knowledge required to resolve the issue not being accessible. Often such issues can be resolved as a result of subsequent refinement. A *Digital Mission Model* must tolerate such residual uncertainty. Again, the importance of linking issues to all relevant model artefacts is that useful decision-making can usually occur with understood risk in the presence of known unknowns, but rarely in the case of unknown unknowns.

2.2.7 Scale

Statements of intent like the one above may be one of thousands for describing real-world missions. We need to be able to scale the capture of the informational semantics of thousands of statements like this and integrate them behaviourally and structurally where required.

Taking a **behaviour-based approach** to describe a system is constructive in that individual segments of behaviour can be naturally composed through questions like "what happens next?" and "what had to happen before this?". Behaviour can also be refined or abstracted to any level of detail and allocated to performers and enablers at that level[6].

As illustrated above, behaviour definition provides sufficient information to synthesise much of the compositional and structural information describing the system of behaviour. Such synthesis significantly reduces manual modelling effort and risk. This is evident, even in this simple example.

The iterative nature of developing a Behaviour Tree means that the effort required for the integration of behaviour, analogous to the completion of a jigsaw puzzle, scales almost linearly. Synthesis also provides confidence in the consistency of the integrated model, mitigating the risk of the resulting model not being a useful *Digital Mission Model*.

2.2.8 Validation

Decades of conducting scenario-based walk-throughs of Behaviour Models with, often non-technical, subject matter experts to identify issues of incompleteness, inconsistency and inaccuracy indicate that Behaviour Trees enable effective validation of intent. When a behaviour model is validated and where composition, structure, etc. have been synthesised from that validated behaviour model and are consistent by construction, then we have confidence that the model is useful for supporting analysis - that it is a *Digital Mission Model*.

[6] As with the fluid definitions of terms like Mission Thread, Mission Engineering Thread, Scenario, Vignette, Use Case, etc. there are no "fixed" definitions for the levels of behaviour, performers and enablers.

In summary, we propose that the existence of the model elements and relationships discussed in this section be a test of a *Digital Mission Model*.

3 Supporting a Managed Systems Engineering Process

This section summarises some of the uses of *Digital Mission Models* that have been successfully demonstrated in industry. It is intended that each of these sections will be the subject of individual papers in the future.

3.1 Generation of Requirements within the Model

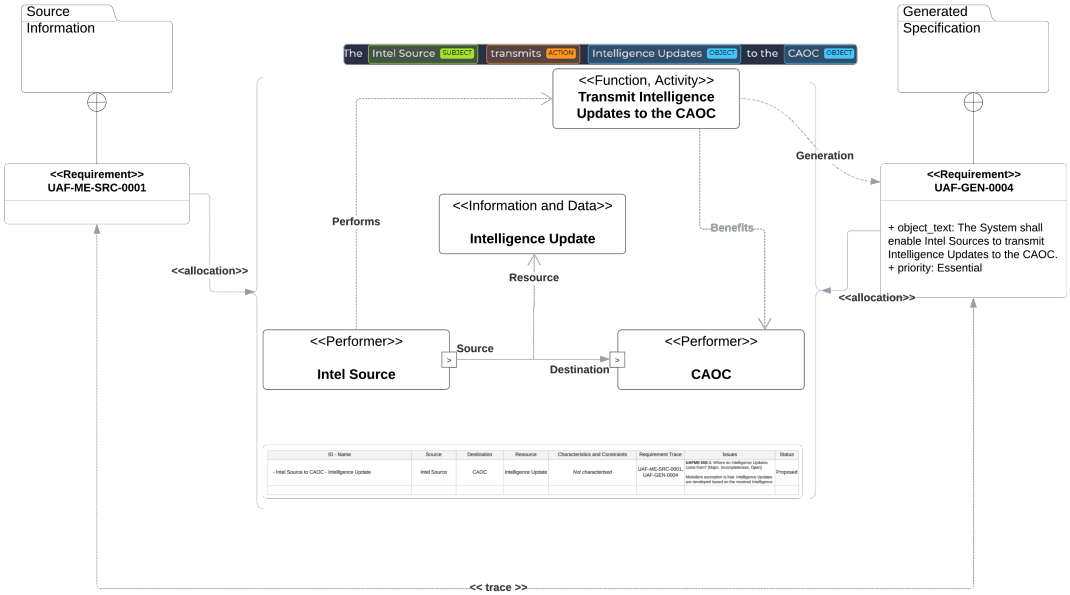


Figure 7: Generated functional requirements, traces and links.

Digital Mission Models contains sufficient, integrated information in the functional architecture alone to generate what most would refer to as the "functional requirements" at whatever level of detail or abstraction we have modelled at. Generated requirements are directly traced to the operational activities performed in pursuit of some mission objective, and traced to all parts of the architecture involved in performing or enabling those operational activities. Figure 7 provides an example of statements of intent/requirement generated from the model and linked to the relevant model artefacts.

Allocating constraints and performance parameters to model elements (particularly functions and performers) allows us to generate performance requirements, compliance requirements, etc., corresponding to the properties and paramterics (refer Figure 8). The model elements and the requirements generated from them are linked during subsequent design, development, test and evaluation with qualification requirements and evidence.

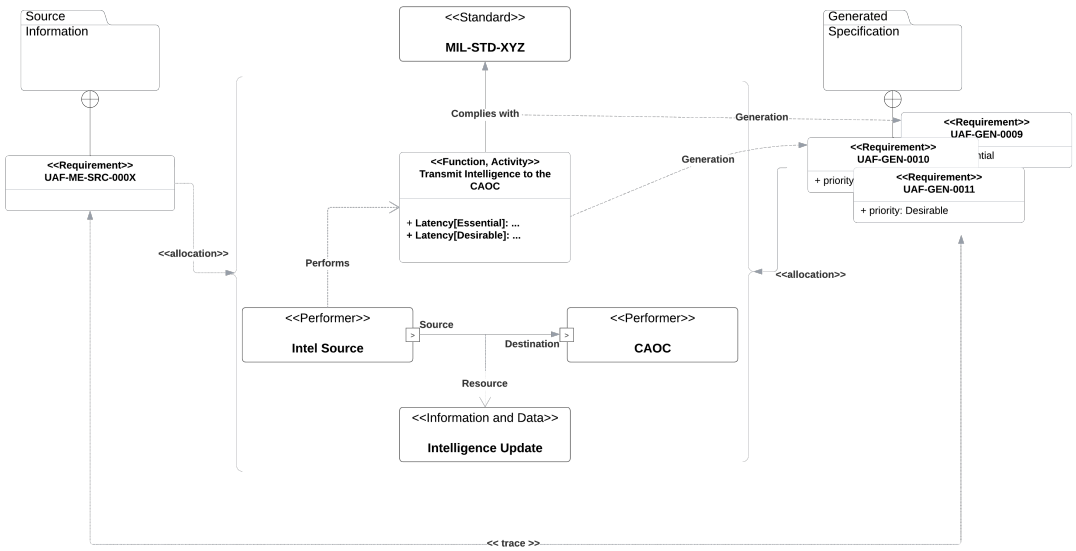


Figure 8: Generation of performance and compliance requirements from properties, constraints and relationships.

3.2 Generation of Acceptance Criteria

The Behaviour Trees within a *Digital Mission Model* contain sufficient information to enable the direct synthesis of acceptance criteria mapped to the respective model elements and any synthesised requirements. By synthesis, acceptance criteria cover all logic represented by the Behaviour Tree and any requirements linked to or synthesised from the same behaviour. This provides a significant accelerator and risk mitigator to the Systems Engineering activities of defining verification statements and methods against requirements and the subsequent design and justification of sufficient sets of tests.

The mapping of acceptance criteria to behaviour (and all related artefacts) enables the assessment of planned test coverage (say in a Test Readiness Review) and, after execution, the assessment of non-compliance against the risk to fitness for mission.

3.3 Linking to Solutions

Given a *Digital Mission Model*, designing a solution to address identified capability gaps is a mapping and recursive *refinement* and modelling exercise that typically involves the following activities:

- Modelling of the solution decomposition integrated within the ontology and allocated against behaviour, activity/function and constraint, interfaces, etc. (potentially staged over multiple design iterations/epochs/etc.),
- *Refinement* of *Mission-Level Behaviours* to describe the actions, interactions and exchanges between elements of the solution decomposition and with elements in their respective context/environment,

- Evaluation of the lower-level performer's and enabler's compliance against requirements and constraints in the *Digital Mission Model* and analysis of the consequence to fitness for mission of non-compliance or low Technology Readiness Level (TRL), and
- Recording of Objective Quality Evidence (OQE), including Test Execution Results, against solution configurations and the intent in the *Digital Mission Model* to support the analysis of fitness for mission.

The first two of these acivities are exactly those outlined above for developing a *Digital Mission Model* only applied at a lower level of abstraction. When integrated with the *Digital Mission Model*, these lower-level models enable not only the ability to reason about the corresponding lower-level systems, but their enablement of the *Mission-Level Behaviour* and the risks to those behaviours resulting from non-compliance to intent. The allocation of OQE to the mapping between solution and intent provides assurance that a solution will be fit for mission.

3.4 Generation of Architectural Views

As discussed, a *Digital Mission Model* represents an integrated architecture. From an integrated architecture, it is possible to generate commonly produced architectural views representing subsets of the integrated information. The effect is that taking a behaviour-based approach enables the rapid development of the following information aspects of an architecture. This was previously discussed in [4]:

- At the operational/mission-level, behaviour trees provide a direct, integrated representation of the following NAFv4 [7] and DoDAFv2.02 [8] viewpoints[7]:
  - L4 - Logical Activities, OV-5b – Operation Activity Model,
  - OV-6a – Operational Rules Model,
  - L4 - Logical Activity, OV-6b – State Transition Description, and
  - L6 - Logical Sequence, OV-6c – Event-Trace Description
- From behaviour trees and corresponding constraint and performance definition, we can synthesise the following artefacts useful for supporting capability acquisition. These generated artefacts are linked to their generative behaviours and all related model artefacts:
  - Functional Requirements,
  - Performance Requirements,
  - Interface Requirements, and
  - Constraints,
- From a behaviour tree, the information content of the following NAF and DoDAF artefacts are synthesised:
  - L1 - Node Types
  - OV-5a - Operational Activity Decomposition Tree
  - L2 - Logical Scenario, OV-2 – Operational Resource Flow Description and/or SV/SvcV-2 – Systems/Services Resource Flow Description, depending on the level of the information being modelled,

[7]The corresponding SV and SVC views are represented by behavioural models at those respective levels of abstraction

  - L3 - Node Interactions, OV-3 – Operational Resource Flow Matrix and/or SV/SvcV-6 – Systems/Services Resource Flow Matrix, depending on the level of the information being modelled, and
  - L7 - Information Model, DoDAF Data and Information models, noting that it is expected that at the mission-level of detail, the information captured in the model represents a DoDAF DIV-1/2 – Conceptual/Logical Data Model.

- When a *Digital Mission Model* is integrated with model artefacts representing system or service performers and enablers, we are also able to synthesise the information content of:
  - SV-5 – Operational Activity to Systems Traceability Matrices, and
  - SvcV-5 – Operational Activity to Services Traceability Matrices.

4 Summary

This paper has outlined the requirement for *Digital Mission Models* to realise the benefits of taking a behaviour-based *Mission Engineering* approach to capability definition, acquisition and sustainment.

Many of the key relationships required are depicted in the simplified (perhaps, oversimplified) conceptual data model of Figure 9.

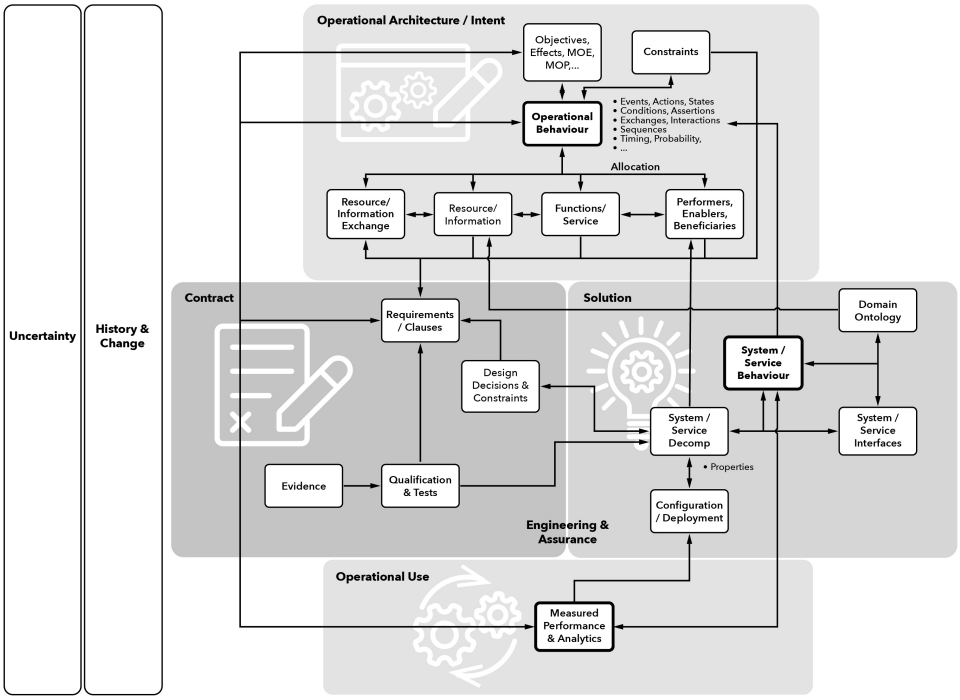


Figure 9: The Kompozition Conceptual Data Model (CDM).

This paper has attempted to argue that integrated, *Digital Mission Models* are necessary for enabling the types of analyses and engineering efforts required to realise the objectives of Mission Engineering. In practice, at the scale where Mission Engineering is applied, the status quo is characterised by distributed, inconsistent and incomplete information. As with all laws regarding entropy, turning this disordered information into ordered information requires the application of effort. At present, this effort is a human effort. Much of this effort and its associated risks are avoidable by taking a behaviour-based *Digital Mission Engineering* approach.

References

[1] Office of the Under Secretary of Defense for Research and Engineering Mission Capabilities, "Mission engineering guide version 2.0," tech. rep., October 2023.

[2] C. Kennedy and M. Efatmaneshnik, "The evolution of mission engineering within the military context," in *2024 IEEE International Symposium on Systems Engineering (ISSE)*, pp. 1–8, 2024.

[3] S. J. Ring, "Activity-based methodology for development and analysis of integrated dod architectures," 2007.

[4] D. Powell, "Producing DoDAF operational views using behaviour engineering – a scenario-based approach," in *6th Asia Pacific Conference on Systems Engineering (SETE/APCOSE 2012)*, 2012.

[5] D. Firesmith, "Mission thread analysis using end-to-end data flows - part 1." https://insights.sei.cmu.edu/blog/mission-thread-analysis-using-end-to-end-data-flows-part-1, August 2019. Accessed: 2024-10-14.

[6] Office of the Under Secretary of Defense for Research and Engineering Mission Capabilities, "Mission architecture style guide version 1.0," tech. rep., January 2025.

[7] "NATO architecture framework version 4." https://www.nato.int/cps/en/natohq/topics_157575.htm?selectedLocale=en. Accessed: 2025-02-28.

[8] "DoD Architecture Framework Version 2.02." https://dodcio.defense.gov/library/dod-architecture-framework/. Accessed: 2025-02-28.

[9] R. Dromey, "From requirements to design: formalizing the key steps," in *First International Conference on Software Engineering and Formal Methods, 2003.Proceedings.*, pp. 2–11, 2003.

[10] D. Powell, "Requirements evaluation using behavior trees-findings from industry," in *Australian Software Engineering Conference (ASWEC'07)*, 2007.

[11] Australian Research Council, "Taming complexity in large-scale systems," in *Outcomes - Results of research in the real world, 08*, 2008.

