2023 NDIA MICHIGAN CHAPTER
GROUND VEHICLE SYSTEMS ENGINEERING
AND TECHNOLOGY SYMPOSIUM
DIGITAL ENGINEERING / SYSTEMS ENGINEERING (DE/SE) TECHNICAL SESSION
AUGUST 15-17, 2023 - NOVI, MICHIGAN

MODEL-BASED ACQUISITION KICKSTART: Lowering the Barrier-to-Entry for Model-Based Deliverables

Eric Alexander[1], Jason Kolligs, PhD[1], Kate Kovalovsky[1]

[1]Strategic Technology Consulting, Toms River, NJ

ABSTRACT

In the current acquisition landscape, the typical program office is proliferating and drowning in a plethora of documents to manage their increasingly complex efforts. Model-based approaches allow System Acquirers (SA) (i.e., Program offices) to articulate system and contractual needs more effectively and precisely. Model-based approaches allow System Developers (SD) to focus their energy on designing complex solutions, instead of spending time and effort preparing documentation to check-the-box; or worse, to create inconsistencies and confuse the SA. There is an immense opportunity to utilize state-of-the-art Digital Engineering (DE) technology and Model-Based Systems Engineering (MBSE) methods to its full potential to streamline the acquisition and system development lifecycle. One of the opportunities is to address the current understanding of a "deliverable". This paper outlines the current beliefs concerning deliverables and presents a desired future state with recommended constructs for implementation.

1. INTRODUCTION

There are several types of deliverables with any given engineering effort. This paper focuses on the delivery of model-based deliverables, namely systems models. After describing the current state of MBSE, DE, and deliverables, three (3) future-state implementation options are presented with pros and cons. As system models are components of MBSE, and MBSE is a supporting pillar of DE, the evaluation of each implementation options considers system understanding, model analysis, DE infrastructure, configuration management (CM), and digital threads. The paper concludes with a recommendation to implement a new paradigm in "delivering" a system design.

2. CURRENT STATE

Modern programs are investing a large amount of time and money into DE Strategy and Implementation initiatives. While these are a necessary element, they also require Contracting & Legal aspects to be successful. Imagine being handed the keys to a Ferrari (your new DE tools, processes, etc.) and then being told to put it in neutral while being pulled by two carriage horses (your document-centric deliverables and processes). The current contractual and legal expectations regarding the formatting and media for system specifications are arbitrary. System requirements convey fundamental information to ensure accountability and contractual obligations [1]. Model diagrams are viable as system requirements and therefore system models can serve as system specifications [2]. “Delivering” a system model is better than delivering a document-based collection of natural language requirement expressions [3]. However, as long as organizations maintain a document- or file-based exchange mechanism, they will be limited to the speed and quality of document-based processes instead of realizing the value proposition of DE technologies and streamlined processes.

Figure 1 below represents this evolution of deliverables from document-centric to a hybrid of documents and models, and proposes a future state where the concept of a “deliverable” is challenged, leading to model-based acquisition programs.

3. RE-THINKING DELIVERABLES

Most major defense acquisition programs are still very prescriptive on document-based formats for deliverables. However, to realize a future state that is not constrained by document- or file-based exchange mechanisms for contract deliverables, a paradigm shift is needed in the concept of the deliverable. Currently, most people understand deliverables as things that are *delivered*. The concept and connotation of *delivered* is still construed as *in-hand*, to include hardcopies and file transfers. This makes logical sense, however there are ways to improve the quality of a shared understanding of the current design maturity by adopting deliverables as things that are *made-available* to the system acquirer. This perspective maintains the original consideration of a deliverable, i.e. something is still delivered; however it now also includes the ability to view the desired information in its current infrastructure. The potential benefits of items being “made-available” vice “delivered” include a reduction in the infrastructure needed to enable the files, more immediate “delivery”, and more immediate feedback on said deliverables.

4. IMPLEMENTATION OPTIONS

There are various implementation options to make something available to review & sign-off. Recall that this paper focuses on systems models as deliverables for all things system definition.

Systems models provide an opportunity to harness the power of digital tools and infrastructure to enable distributed collaboration in a common framework (across networks, organizations, industries, classification levels, etc.). When necessary for contractual and legal requirements, automated, on-demand document generation

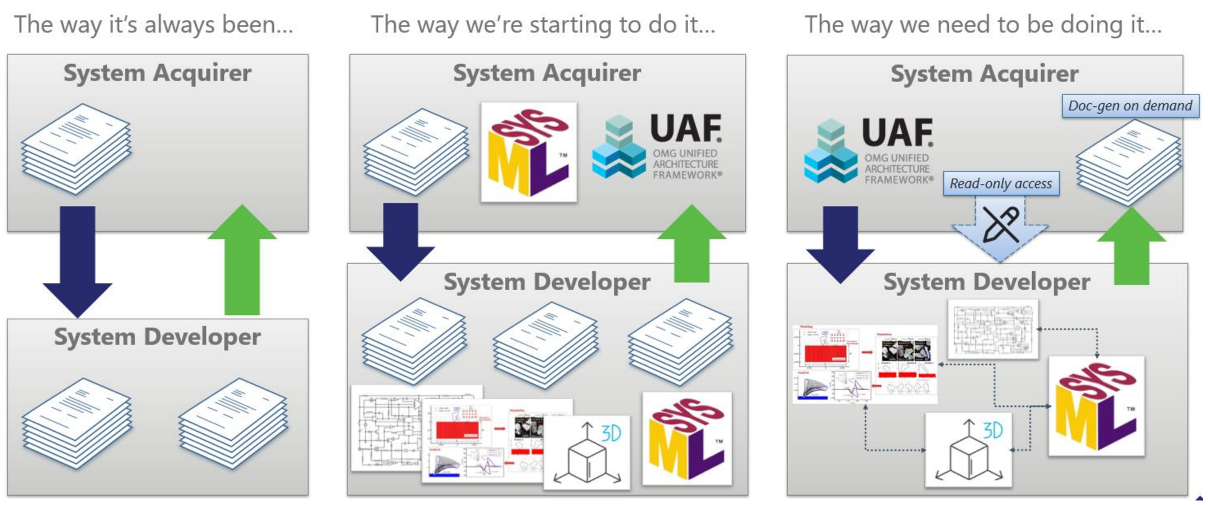
The way it's always been...
The way we're starting to do it...
The way we need to be doing it...
System Acquirer
System Developer
UAF
OMG UNIFIED ARCHITECTURE FRAMEWORK®
SysML
Doc-gen on demand
Read-only access
3D

Figure 1 – Evolution of Model Based Deliverables

can create necessary documentation directly from the model.

The question is if a *delivery* of the system model, i.e. a model file(s) sent to the system acquirer, is truly necessary as a "deliverable". Or, can acquirers designate trusted agents to gain access to the system developer's Digital Engineering Ecosystem (DEE) to truly observe and navigate the full digital thread and conduct a more thorough review of the design throughout the engineering lifecycle? If the design maturity is better understood on a rolling baseline, how much more efficient can system technical reviews become? While major technical reviews like System Requirements Review (SRR), System Functional Review (SFR), Preliminary and Critical Design Review (PDR, CDR) will need to take place for most acquisition programs based on acquisition policy, the process of reviewing a plethora of documents prior to the review and conducting a meaningful technical exchange can be drastically improved upon using model-based approaches and access to dynamic baselines as proposed here.

Research into best practices do not identify proven solutions for delivering the full digital thread as part of a deliverable. Digital thread tools are deeply nested into the environment in which they are deployed and thus cannot be extracted with all the connected models and linkages. Therefore, programs requesting model-based deliverables are simply getting pieces of the puzzle, separate from the connective tissue (digital thread) that allows for robust impact analysis to make informed decisions.

This paper identified three (3) approaches: 1) send the model frequently, 2) direct, read-only access to the model, and 3) remote repository synchronization. Here are some pros and cons of system model delivery options:

Option 1: Send the Model Frequently

Pros:

- Gives the system acquirer a much more approachable understanding of system development maturity by reviewing work in progress (bi-weekly, monthly submissions), as opposed to digesting all the content 30 days before a major review.
- Allows the system acquirer to set up a Teamwork Cloud (TWC) on their end and use system developer model(s) for internal analysis that perhaps the system developer(s) should have awareness of.
    - Applies in a situation where you may have multiple competitive primes working on a contract.

Cons:

- File, SaveAs in Cameo leads to some model organization issues (extra attention needed for containment tree structure- especially with used projects), and the risk of model-mix up on the receiving end when publishing to system acquirer's TWC.

Option 2: Direct (read-only) access to the model(s) / DEE

Pros:

- No need to SaveAs and send a model file (or collection of files) to the acquiring office repeatedly.
- Reduces risk of "model-mix up" on the receiving end when the acquiring office's modeling team reassembles in their TWC (project usage versions, unresolved reference errors, etc.).
- The true digital thread can be traversed by acquirer to have a better total system understanding (digital thread is not easily transferred between developer and acquirer environments).

Cons:

- The acquiring office will have people with access to the living, breathing system model (read-only) in its Authoritative Source of Truth (ASoT). This could create a sense of excessive oversight and lead to inefficient modeling practices to make sure everything is always "review-worthy".
- Recommended solution: use branches in TWC for development and releases and manage permissions accordingly.

Option 3: Remote Repository Synchronization

Pros:

- Acquirer to System Developer TWC synchronization for seamless traceability.
- Models can be synchronized on major revision (e.g., new requirement baseline etc. from Acquirer Model, new technical baseline revision from Developer Model).
- Models can be synchronized on any commit (for an extremely tight integration between distributed teams).
- Cross domain / classification synchronization (one-way).
- Maintain configuration control of the model elements and traceability without air gaps in the transfer of models up into higher classification levels.

Cons:

- Authentication limitations prevent full implementation on DoD networks.
- Still requires a human in the loop to ensure setup and configuration on both sides.

Approach	**Frequently Sent Model**	**Read-Only Access to SD Model / DEE**	**Remote Repository Synchronization**
System Understanding	Can be reviewed more frequently than 30 days before a major review.	Can be published at the same frequency as the first approach.	Can be updated daily as an automated process.
Model Analysis	SA will have model files and can conduct any desired analysis within their own infrastructure. Digital threads will still have to be established on the SA side.	SA only has access to analysis developed by SD.	SA will have their own synched version of the model and can conduct any desired analysis within their own infrastructure. If the entire DEE is mirrored, then the threads should be maintained. Otherwise, there could still be DE challenges on the SA side.
DE Infrastructure	Requires SA to have same infrastructure as SD.	SA only needs a web-browser to enable full review and provide feedback.	Requires SA to have the same infrastructure, plus additional infrastructure to support the synchronization
Configuration Management	Version of the model is only relayed in the filename.	Requires branching of the model and must be published by the SD.	Requires process to determine auto updates or human in the loop version updates for repo.
Digital Threads	May exist.	Can be fully traversed.	Once recreated both via infrastructure and synchronization of applicable components, can be fully traversed.

5. Other Considerations

5.1. Document Generation

It is the expectation of DE best practices that reviews and archives happen in ASoTs when possible. In the near term, it is still widely accepted that documents will be required for formal milestone reviews & contractual archives etc. However, system models should serve as the sources for document generation and not stand-alone duplicative sources of the same document-based content. Otherwise, the total number of products that require work increase, rather than using the models as the ASoT and documents as derivative work products on demand.

The concept of document generation is truly just scaffolding while engineering efforts exist in a hybrid DE implementation state where documents are still seen as necessary to provide technical documentation.

The Velocity Template Language (VTL) is a general-purpose scripting language used in a wide variety of software applications and is fully supported by Cameo. VTL templates can be produced to create the shell of a document that is typically produced manually, then in Cameo the template is run to pull content from the model to populate the sections that the template defines. Creating the template takes some investment, but each subsequent publish of that report from the model will buy back the time it would have taken for manual updates, consistency checks, and coordination across a distributed team.

5.2. Standardizing MBSE CDRLs

The question remains as to how a model-based CDRL can be conceptually standardized for acquisitions. The Systems Modeling Language (SysML) details the terms of defining the concrete and abstract semantics of the systems model itself, but has limited information regarding the presentation of data that is meeting contract data requirement list (CDRL) views. Many Data Item Descriptions (DIDs) have been updated to include the notion of leveraging a relational database to communicate architecture, requirements, and traceability type deliverables. Several more DIDs from the Air Force and Missile Defense Agency have been taking great steps to call out a Digital Systems Model that encompasses SysML, UML, and other specialty engineering considerations.

In addition to these new DIDs, using ISO 42010 as a guide, views and viewpoints can be explicitly produced in a systems model by the System Acquirer and sent out with a Request for Proposal (RFP) to supplement these DIDs. Having the desired model-based views and viewpoints defined in a model would allow the SD to trace their model content (diagrams) to the requested views (via an <<expose>> relationship).

6. CONCLUSION

To realize true digital transformation in a timely fashion, the power of digital tools and infrastructure must be harnessed to enable distributed collaboration in a common framework (across networks, organizations, industries, classification levels, etc.) without proliferating documents for historical purposes. This paper encourages the industry to challenge the contractual paradigm of a “deliverable” and start interacting more efficiently as distributed Acquirer-Developer teams. This can be done by 1) implementing a sound MBSE approach within a solid DE ecosystem that manages system technical data and 2) interacting with systems models more frequently and efficiently to expedite the process of technical interchange and baseline approval to make our acquisition programs move faster on delivering the end item to the warfighter.

7. REFERENCES

[1] J. Kolligs and D. Thomas, “The Origins of Requirements”, *IEEE Systems Journal*, vol. 15, no. 3, pp. 3692-3702, 2021.

[2] J. Kolligs and D. Thomas, “On the Viability of Diagrams and Drawings as System Requirements”, *Systems*, vol. 11, no. 4, pp. 176, 2023.

[3] J. Kolligs, “The Case for Alternate Media Requirement Expressions in Systems Engineering”, Ph.D. Dissertation, College of Eng., Univ. of Alabama in Huntsville, 2022.