2024 NDIA MICHIGAN CHAPTER
GROUND VEHICLE SYSTEMS ENGINEERING
AND TECHNOLOGY SYMPOSIUM
DIGITAL ENGINEERING / SYSTEMS ENGINEERING TECHNICAL SESSION
AUGUST 13-15, 2024 - NOVI, MICHIGAN

IMPLEMENTING MODEL-BASED PRODUCT LINE ENGINEERING TO MANAGE A MIXED PORTFOLIO OF LEGACY, UPGRADE, AND NEW DEVELOPMENT SYSTEMS FOR SELF-PROPELLED HOWITZER

Jake Engle[1], Daisy Yakovich[2], Erin Torchala[3], Jeff Greene[4]

[1, 2] Strategic Technology Consulting – an Arcfield Company, Chantilly, VA
[3, 4] PM Self-Propelled Howitzer Systems, PEO Ground Combat Systems, Detroit Arsenal, MI

ABSTRACT

This paper summarizes the benefits, technical approach, and lessons learned from applying a Model-Based Product Line Engineering (MBPLE) approach to manage a mixed portfolio of systems at Project Manager (PM) Self-Propelled Howitzer Systems (SPHS) within Program Executive Office (PEO) Ground Combat Systems (GCS). The theme of the paper is to marry the concepts of Model-Based Systems Engineering (MBSE), flexible acquisition strategies, and product line engineering to describe the approach that STC has assisted PM SPHS in implementing for their program office's digital transformation initiative. Lessons learned, best practices, and economic benefits of utilizing MBPLE are highlighted throughout.

Citation: J. Engle, D. Yakovich, E. Torchala, J. Greene, "Implementing Model-Based Product Line Engineering to Manage a Mixed Portfolio of Legacy, Upgrade, and New Development Systems for Self-Propelled Howitzer". In *Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium* (GVSETS), NDIA, Novi, MI, Aug. 13-15, 2024.

1 INTRODUCTION

This paper explores essential considerations in product lines in Defense Acquisition and offers insights into strategies that practitioners can employ to gain efficiencies in the management of complex portfolios with common and unique entities. These strategies are aimed at fostering operational proficiency within an organization.

Nearly all endeavors in systems and software engineering unfold within the framework of an existing product line. Rarely are new development systems truly novel and unique; new systems are often evolutionary versus revolutionary. Systems that are dependent on a system of systems approach for emergent behaviors are rarely all developed at the same time, thus legacy system of systems interoperability needs will further constrain the newly developed system, such as the desire for a future howitzer to still fire legacy ammunition while simultaneously enabling future ammunition. And when developing a new system, rarely does anyone strive to confine themselves to a singular edition or flavor during development. Product lines are pervasive across diverse industries, spanning aerospace and defense, automotive, medical, consumer electronics, industrial automation, etc.

As commercial and defense organizations globally strive for competitive advantage via reduced time to market (or fielding), cost savings, and scalability, the demand for efficient approaches to produce systems and software product lines becomes universal. The need to achieve these goals while managing complexity often consumes engineering teams' efforts, diverting them from seizing opportunities to innovate and fully exploit new product ideas. Using traditional approaches of standing up a dedicated team to manage each new variant has an unintended consequence of requiring more liaison effort between variant-owning teams to coordinate development and changes to common parts, which does not add value and tends to slow down each variants work, and this gets slower as additional variants must be coordinated with. A common engineering team and system modeling environment to define the portfolio and its assets helps reduce these liaison needs. This is a fundamental perspective of Digital Engineering: in an authoritative source of truth (ASoT) we all see the same data, across the lifecycle and use the information for a multitude of purposes without duplication. When managing multiple variants, the ASoT needs to address the overall portfolio, not just a model of a single variant.

The traditional "clone and own" reuse approach, where each product variant is custom fit to a customer's needs, is no longer sufficient. The SPHS program management office, along with many other Army program offices, have utilized the “clone and own” methodology throughout history to evolve the M109 self-propelled howitzer (SPH) fleet – the previous variant becoming the foundation for the next with subsystem and component-level upgrades. Over the course of time in the production and fielding process, managing operational readiness of a (weapons) system may erode due to the rapidly emerging threats. Managing emergent threats and adding capabilities is no longer a single variable trade study, but rather a multi-variable optimization problem and requires a new approach. Although having adequate capabilities at the time of initial fielding, most complex systems need to be upgraded as threats evolve. As shown in current conflicts the way the Army fights can change very quickly, which places an emphasis on the system architecture’s ability to be rapidly adopted to include the systems already fielded (modular or open architecture standards are an enabler of this system characteristic).

Using MBSE to define a comprehensive SPH Architecture Model where these analyses can inform future decisions on an optimal future system solution is a key component of the program’s Digital Engineering (DE) Strategy. The goal of the model is to readily configure current and future potential variants to enable maximum flexibility to the PM when determining an acquisition strategy. The acquisition strategy will define what is to be procured, and the PLE-based model can quickly define the architecture that industry needs to fit within or build to for the given acquisition scenario.

The contemporary landscape demands agile and expedited deliveries. Customers now expect frequent and modular updates, lower development costs, and swift deliveries to stay abreast of competition and evolving operational landscapes. “The Department [of Defense] is transitioning to a culture of performance and affordability that operates at the speed of relevance… We will prioritize speed of delivery, continuous adaptation, and frequent modular upgrades.” – James Mattis, former U.S. Secretary of Defense, April 2018. In the Defense landscape, we must be able to rapidly evolve our systems and processes to counter rapidly emerging threats from our adversaries.

Besides managing a capability upgrade that gets rolled into new production as soon as possible, an MBSE modeling approach also helps in retrofitting the fielded vehicles, and similarly resolving obsolescence redesign issues. In essence, the system as designed, produced, and then sustained is always evolving – we cannot think about a system as a single configuration anymore, so everything becomes an evolving variation of the common platform. It is no longer acceptable to spend 10 years developing a system. PMs need to have the ability to field capabilities fast, and incrementally and continuously upgrade them overtime to keep pace with emerging threats and evolving operational needs.

2 MODEL-BASED PLE

(1) Technical Approach

Feature-based PLE is an approach to software and system development that focuses on creating a family of related products or systems by managing and reusing common features and components. This approach is particularly valuable for organizations that need to develop a range of products with variations while maximizing efficiency and maintaining consistent quality. STC employs this Feature-based PLE approach within a modeling environment – coining the term Model-based Product Line Engineering (MBPLE). Their approach to building scalable products (System Models) leverages the Feature-based PLE strategy in accordance with ISO/IEC 26580:2021. STC's MBPLE Framework seamlessly integrates into the Feature-based PLE approach identified above and has been a proven and repeatable solution framework for a multitude of customers.

(2) Economic Benefits

Feature-based PLE provides strategic advantages that enable organizations to achieve efficiency, scalability, cost savings, and adaptability in their product development processes. Many PMs have the mission statement of providing overmatch capabilities over time, not just fielding a single system instance, such that Feature-based PLE puts the design authority decision on the overall product portfolio to develop a strategic upgrade/modernization plan across multiple variants over an extended time period, something that was not done when viewing each variant as a single product instance. Feature-based PLE introduces the concept of lifecycle time to the portfolio, not just the time it takes to develop or field a single program of record. For example, the PM SPHS mission statement emphasizes the PM's broader responsibility as a continuum: *" Continuously Develop, Produce and Sustain Field Artillery Self-Propelled Howitzer Systems to ensure the King of Battle remains dominant on the battlefield."* Ironically, although with new technologies change is coming faster and new threats are entering the battlefield at reduced cost and increased capability making them more pervasive, these realities cause the PM to take a longer-term, more strategic perspective as to the evolution of their platforms. These advantages contribute to a more competitive and responsive organizational structure in today's dynamic business environment.

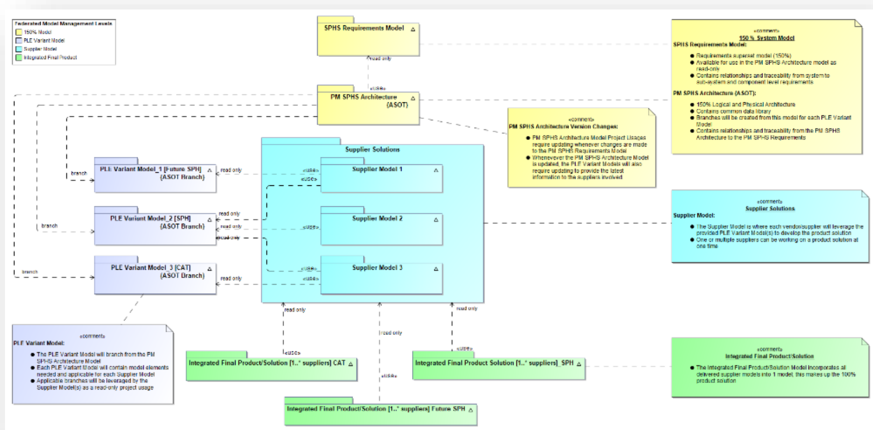
3 PM SPHS MBPLE APPROACH

The SPHS project office was presented with a challenge not currently addressed within the Army. While other organizations are developing models for a new platform from the ground up leveraging contracted digital data artifacts, PM SPHS must account for legacy platforms – often with inconsistent or incomplete document-centric data sources – in addition to future systems. This unique challenge further solidifies the need to implement MBPLE to maximize efficiencies, along with many broader benefits to the organization.

To address this challenge, PM SPHS contracted STC to assist in the establishment and population of an MBSE model for the SPHS portfolio, focusing on the Extended Range Cannon Artillery (ERCA) products, leveraging PLE practices. Although the original MBSE project intent was to enable reuse of common requirements across multiple system variants, it was quickly determined that reuse was desired across all the system definition elements to maximize acquisition flexibility to meet future program needs. As the PM is asked to manage more platforms without a significant increase in personnel, the PM must become more efficient in how they execute the work, while the future of artillery programs continues to evolve in parallel to reflect world events.

First, to better capture how we meet the operational capabilities defined by the Fires Community (hence forth referenced as user), the system boundary was redefined. A firing platform cannot execute the capabilities required on its own – driving, firing, and resupplying. The crew, ammunition, and the associated resupply vehicle are all required to successfully execute the required capability. Therefore, the system boundary line was redrawn to include firing platform and associated resupply vehicle along with required crew and ammunition. This redefined boundary thus informs how the system is viewed in the SPHS model. Also, with the crew considered part of the platform, it enables reallocation of manual tasks (such as loading) from the crew to an autoloader as automation is introduced within future variants.

As previous SPH variants had difficulties in Operational Test due to inadequate training, the inclusion of the crew within the system boundary places an emphasis on human interface optimization as well as sufficient training. This inclusion allows the model-generated crew task list to become an evaluation of the performance value of automation upgrades in a consistent framework. Therefore, the Feature-based PLE model is not really a model of a single system, but is flexible to represent the current vehicle, an all-new vehicle, or any type of upgrade in-between. This approach was selected to ensure the SPHS MBSE model was sustainable and provided the ability to evaluate individual variant decisions within the context of the overall SPHS portfolio.



Another benefit from this approach is the ability for the PM to plan wholistic upgrade paths across multiple iterations. This approach provides a deeper understanding of the system and identifies the behavior, requirements, and architecture decisions that have the greatest impact on the design. We can add technology enablers or overdesign components earlier in the evolution plan to ease the implementation of a future upgrade without having to redesign components multiple instances over time. The strategy is a longer-term, continuum, not a single product delivery.

identify decisions can be changed on the current upgrade, such that enablers are added or parts are overdesigned such that it is easier to do a future upgrade, and redesigned parts for the current upgrade do not have to be removed and redesigned again – the strategy is a longer-term continuum, not just a single product delivery.
Next, a MBPLE approach was developed consisting of Behavior, Architecture, and Requirement Specification models within the Teamwork Cloud. The models are implemented via the MagicDraw plugin.

The SPHS model starts at the portfolio level, establishing “150%” models – “buckets” of all possible and available elements (behaviors, architectures, and requirements) that could be allocated to specific platforms within the system and across the portfolio in alignment with PLE practices. With this set up, the integrated models can generate numerous configurations supporting multiple acquisition strategies – system level new start, subsystem upgrade, component upgrade, software upgrade, etc. – that can then support an RFP release with a specific acquisition effort (see figure below).

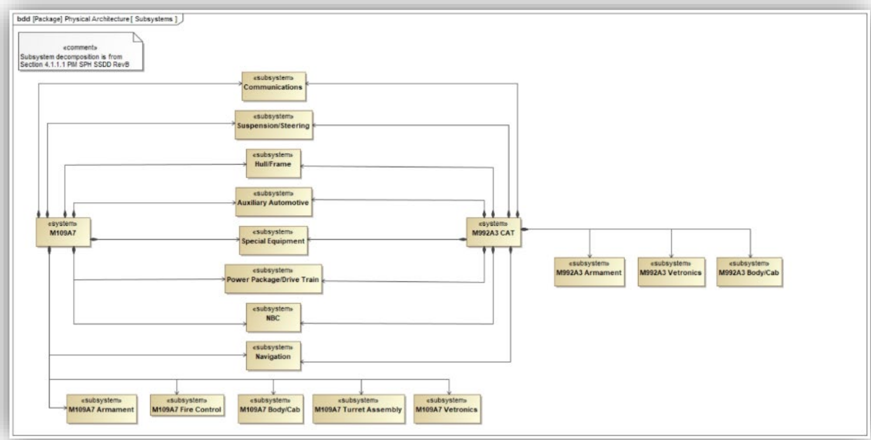
To generate and populate model data, functional-specific working groups were established to validate assumptions and data supporting the 150% model level as well as the decomposition to each platform – legacy, current, and future.

Ultimately, the SPHS MBPLE approach is enabling proper understanding and documentation of the systems across the portfolio, bringing the SPHS office into the digital age while delivering the right systems to the Field Artillery community ensuring operational dominance.

4 Lessons Learned

***Standardization and Guidelines*:**

- Establish standardized guidelines and best practices for PLE within the organization. This includes naming conventions, documentation standards, and modeling conventions to ensure consistency. To ensure model interoperability, it is desired that these standards be common across PEO GCS and the supporting Development Commands (DEVCOMs).



***Adaptive Acquisition Strategies*:**

- Flexible acquisition strategies can help mitigate the impact of changing budgets, technology advancements, and evolving military needs. Adoption of an agile acquisition process can be more responsive to these changes.

Lifecycle Cost Management:

- Managing lifecycle costs is a critical factor in PM SPHS, including the importance of evaluating long-term costs and making design decisions that can reduce sustainment expenses.

Open Systems Architecture (OSA):

- PM SPHS encourages OSA to foster competition, innovation, and vendor diversity, thus reducing vendor lock-in, improving technology refresh cycles, and reducing long-term costs.

Commonality and Standardization:

- *P*M SPHS's framework promotes the standardization of components and interfaces across different SPH Family of Vehicle platforms. The value of common components in streamlining maintenance, logistics, and training, as well as reducing development costs, are crucial lessons learned. The figure below shows an architectural view where the common and unique subsystems are laid out in the model for reuse.

***Modularity and Interoperability*:**

- PM SPHS emphasizes the importance of designing modular systems that can be integrated across different platforms. This approach promotes interoperability and flexibility in deploying various components and subsystems across a range of self-propelled howitzer systems, resulting in benefits of modularity for rapid adaptability and cost-effective upgrades.

Digital Engineering and Simulation:

- PM SPHS leverages digital engineering and simulation tools to accelerate design, development, and testing processes. Lessons learned include the importance of digital tools for reducing development timelines, improving product quality, and enhancing collaboration among teams.

5 REFERENCES

[1] ISO/IEC 26580:2021, Software and systems engineering – Methods and tools for the feature-based approach to software and systems product line engineering

[2] International Council on Systems Engineering. "Feature-based Systems and Software Product Line Engineering: A Primer," Technical Product INCOSE-TP-2019-002-03-0404,available at https://connect.incose.org/Pages/Product-Details.aspx?ProductCode=PLE_Primer_2019, downloaded January 2020.

APPENDIX A: Terminology

Feature-based PLE Terminology and Definitions (all terms are from ISO/IEC 26580:2021)

3.1
bill-of-features
specification for a member product (3.8) in the product line (3.16), rendered in terms of the specific features (3.4) from the feature catalogue (3.5) that are chosen for that member product

3.2
bill-of-features portfolio
collection comprising the bill-of-features (3.1) for each member product (3.8) in a product line (3.16)

3.3
domain supersets
collection comprising the feature catalogue (3.5) and shared asset supersets (3.18)

3.4
feature
characteristic of a member product (3.8) in a product line (3.16) that distinguishes it from other member products in the product line

Note 1 to entry: Features can a) express the customer-visible or end-user-visible variability among the member products in a product line, or b) distinguish implementation variability not directly visible to a customer or end user except through non-functional differences such as price, performance, noise, weight, energy and more.

Note 2 to entry: In feature-based PLE, features express differences among member products. A capability or other characteristic common to all member products in the product line is not modelled as a feature.
Note 3 to entry: See Annex A for the definition of this term in ISO/IEC 26550.

3.5
feature catalogue
model of the collection of all the feature (3.4) options and feature constraints (3.6) available across the entire product line (3.16)

3.6
feature constraint
formal relationship between two or more features (3.4) that is necessarily satisfied for all member products (3.8)

3.7
feature language
syntax and semantics for the formal representation, structural taxonomy, and relationships among the concepts and constructs in the feature catalogue (3.5), bill-of-features portfolio (3.2), and shared asset superset (3.18)variation points (3.20)

3.8
member product
product belonging to the product line (3.16)
[SOURCE:ISO/IEC 26550:2015, 3.15, modified — The preferred term "application" has been removed.]

3.9
mutually exclusive
alternatives from which at most one is selected

3.10
mutually inclusive
alternatives from which zero or more are selected

3.11
PLE factory
technological, organizational, and business infrastructure and processes to support a PLE factory configurator (3.12) producing product asset instances (3.14) from shared asset supersets (3.18) based on a bill-of-features (3.1) for a member product (3.8)

3.12
PLE factory configurator
automated mechanism that produces assets for a specific member product (3.8) by processing the bill-of-features (3.1) for that member product, and exercising the shared assets' (3.17) variation points (3.20) in light of the feature (3.4) selections made in that bill-of-features

3.13
PLE factory development environment

toolset for creating, organizing, assembling, and maintaining a collection of elements in a feature catalogue (3.5), bill-of-features portfolio (3.2), shared asset supersets (3.18), and a hierarchy of a product line (3.16) of product lines

3.14
product asset instance
instantiation of a shared asset (3.17) specific to a member product (3.8), automatically produced by the PLE factory configurator (3.12), corresponding to a bill-of-features (3.1) for that member product

Note 1 to entry: A product asset instance is analogous to an application asset (ISO/IEC 26550) with the proviso that it is produced by the PLE factory configurator.

3.15
product instances
collection comprising the bill-of-features portfolio (3.2) and product asset instances (3.14)

3.16
product line
family of similar products with variations in features (3.4)
Note 1 to entry: See Annex A for the definition of this term in ISO/IEC 26550.

3.17
shared asset
software and systems engineering lifecycle digital artefacts that compose a part of a delivered member product (3.8) or support the engineering process to create and maintain a member product

Note 1 to entry: A shared asset is analogous to a domain asset (ISO/IEC 26550).

Note 2 to entry: Typical shared assets are requirements, design specifications or models for mechanical, electrical, and software, source code, build files or scripts, test plans and test cases, user documentation, repair manuals and installation guides, project budgets, schedules, and work plans, product calibration and configuration files, mechanical bills-of-materials, electrical circuit board and wiring harness designs, engineering management plans, engineering drawings, training plans and training materials, skill set requirements, manufacturing plans and instructions, and shipping manifests.

3.18
shared asset superset
representation of a shared asset (3.17) that includes all content needed by any of the member products (3.8)

3.19
variant
alternative that can be used to realize a particular variation point (3.20)
[SOURCE:ISO/IEC 26550:2015, 3.28, modified — the word "one" at the beginning of the definition has been removed; "may" has been changed to "can"; "particular variation points" has been changed to "a particular variation point"; note 1 to entry has been removed.]

3.20
variation point
identification of a specific piece of shared asset superset (3.18) content and a mapping from feature (3.4) selection(s) to the form of that content that appears in a product asset instance (3.14)