# Submarine Warfare Federated Tactical Systems





This article describes the transformation of the systems engineering and integration program that produces the common combat system used across the United States Navy (USN) submarine fleet from traditional document-based systems engineering (DBSE) to model-based systems engineering (MBSE).. The topic may be of particular interest to those dealing with programs in the sustainment and evolution phase of their life cycle. For addition information, refer to the links provided in Section V, Lessons Learned below.

☐

## Contents



## Background

Modern submarines are typically in service for 20 - 40

years. Historically, each new class of submarines has been equipped with a new combat system, with a corresponding logistics and sustainment tail unique to that class. Submarines and their internal systems are commonly state-of-the-practice at launch, but most navies find it necessary to upgrade the ship's combat system at least once during the operational lifetime. The evolution of threats, technology and interoperability drives the USN to upgrade their submarine combat systems and key components including the sonar (Fages 1998) (Ford and Dillard 2009) and tactical control systems continuously (Jacobus and Barrett 2002).

Over the last three decades submarine combat systems have evolved from multiple independent systems (sonar, combat control, imaging, electronic warfare, weapon control, etc.) with manual or point-to-point interfaces into networked federations of systems (FoS). Confusingly, these component systems are often referred to as subsystems in the literature.

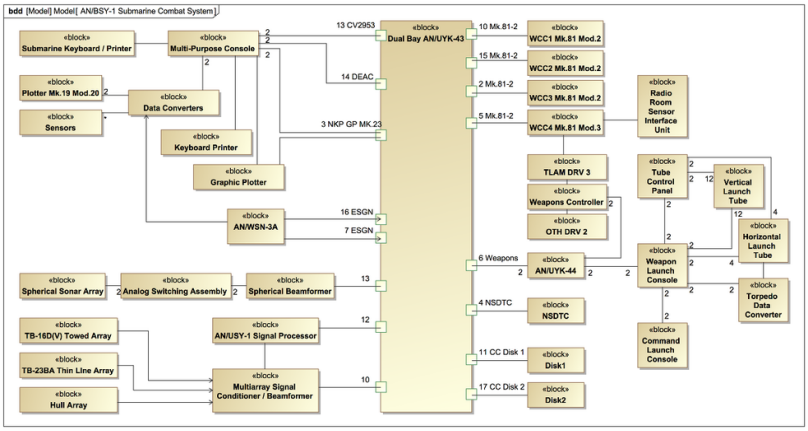


**Figure 1.** ***1985-era submarine combat system composed from independent component systems with point-to-point interfaces*** (SEBoK Original)

In the USN, each of these component systems has its own acquisition program, customer, and contractor team. Starting as legacy military systems hosted on traditional military-unique computational platforms, these systems have evolved to utilize Commercial Off-The-Shelf (COTS) computational and networking platforms, and leverage large amounts of COTS software.

As the component systems became more tightly interconnected, the acquisition customers established and collaboratively funded a systems engineering and integration (SE&I) program to manage the interfaces between systems, manage technology insertion and the obsolescence of common COTS components, and to integrate and test the production systems (Cooper,

Sienkiewicz and Oliver 2006). Starting with the Virginia class, this SE&I program was expanded to encompass both new-production and in-service submarine modernization efforts. Over time, the combat systems of the various USN submarine classes were converged into variants of a single product line (Zingarelli, et al. 2010).

## Purpose

The submarine combat system SE&I program delivers an updated production baseline annually, along with product line variants for each submarine class or subclass being built or upgraded that year. Production systems implementing this baseline are delivered to new-build submarines, and to in-service submarines being upgraded on a roughly six-year cycle. The common combat system product line is referred to as the Submarine Warfare Federated Tactical Systems (SWFTS). SWFTS is deployed by the USN on submarines of the Los Angeles (SSN 688), Ohio (SSGN 726, SSBN 730), Seawolf (SSN 21), and Virginia (SSN 774) classes, and by the Royal Australian Navy on the Collins (SSG 73) class. SWFTS is also planned for the next-generation USN Columbia (SSBN) class. Compared to the submarine combat systems that it replaced, SWFTS significantly reduces development, maintenance and training costs while delivering enhanced combat capabilities and facilitating the rapid insertion of new or improved capabilities (Zingarelli, et al. 2010).

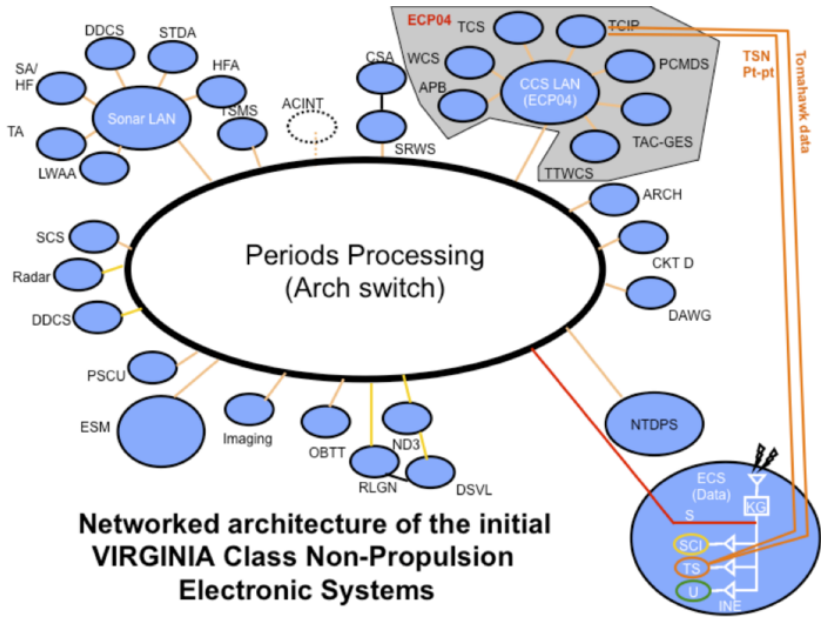


**Figure 2. Contemporary submarine combat systems are networked with converged data interfaces** (Produced by Lockheed Martin for US Navy. Approved for Public Release by US Navy, #16-348, June 2016)

## Challenges

The USN submarine fleet encompasses substantial platform variability between class, sub-classes, and even individual ships within a sub-class. The RAN Collins class contributes additional variability. Platform variability drives combat system variability.

SWFTS is a Federation of Systems, with each platform hosting a subset of 40 systems produced by 20 different program offices. As is common with system of systems (SoS) and FoS, there is no central program office that can command the compliance of all of the component system programs. Instead, the evolution of SWFTS is executed through negotiation and consensus.

Many baselines must be produced each year: new common hardware baselines are introduced in odd years, while new common software baselines are introduced in even years (Jacobus, Yan and Barrett 2002). In addition, multiple incremental developmental baselines are established each year. Once the annual production baseline for the product line is defined, variants must be developed for each submarine class or subclass built or upgraded that year (Mitchell 2012).

Like most other defense programs, the SWFTS SE&I program is under constant pressure to accomplish more with decreasing resources. There has been steady increase in SE scope despite decreasing budgets. Program leadership has responded in part through continuous SE process improvement. Improvements have included test automation, changes in the requirements management processes and tools (spreadsheets to IBM® Rational® DOORS® to OMG® SysML®), refined tooling for change management, and the DBSE to MBSE transition that is the focus of this case study. Substantial Return on Investment (ROI) has been achieved with each major SE process or tooling improvement (Mitchell 2014, Rogers III and Mitchell 2021).

## Systems Engineering Practices

During 2009 the SWFTS SE&I program conducted a Model Driven Architecture (MDA) study to determine if MDA should be the next step in the program's continuous SE process improvement. The MDA study predicted a positive ROI from converting to MBSE. In January 2010 the SWFTS SE&I program office kicked-off a three-year effort to develop and validate a SWFTS FoS

model and MBSE process. In 2013 a SWFTS baseline was developed using both DBSE and MBSE in parallel. The DBSE products were used to validate the MBSE products. Based on that successful validation, the SWFTS SE&I program transitioned to MBSE for all ongoing work.

Until the transition in 2013, SWFTS SE was performed using traditional DBSE. Requirements were managed in DOORS, and reviewed and used by engineers in the form of massive spreadsheets with hundreds of columns and thousands of rows. The design of each variant was documented in Microsoft Office files. Baseline Change Requests (BCR) were documented in briefings, and analyzed by all component system programs in parallel for potential impact. Approved BCRs were manually merged into DOORS and into revised baseline documents for each effected variant.

Starting in 2010, the customer community invested in a three-year MBSE transformation effort. The engineering team performed an in-depth tool trade study to select and set up the MBSE environment. That trade study resulted in the selection of MagicDraw™ for system modeling, using Teamwork™ as the model repository.

Once the modeling environment was installed, the MBSE transformation team architected, developed and populated a SysML-based model of the SWFTS FoS interfaces. The team updated the SWFTS SE process to take advantage of the new MBSE environment, with the constraint that the MBSE process produce SE products that were effectively identical to those produced by the DBSE process.

As the SWFTS model was developed, unanticipated benefits emerged. Capturing the architecture in a model improves the depth and quality of baseline products due to the fact that, unlike the spreadsheets it replaced, the model inherently captures relationships between elements. Using the model, one can drill down and explore various aspects of the architecture, e.g., a) the network design that supports data exchanges; b) component systems that use a particular network Virtual Local Area Network (VLAN); c) service level requirements levied upon data providers.

In 2013 the new MBSE environment and process was used to produce a set of SWFTS baseline SE products. In parallel, the DBSE process produced equivalent products. The two sets of baseline products were compared in detail, with all differences traced back to

root causes.

The relationships in the model also support automated integrity and consistency checking between model elements, which is also called model concordance. Analysis of these relationships identified a significant number of minor differences that were all traced to errors in the DBSE products. This validated the MBSE process, and demonstrated that while those initial MBSE baseline products were more labor-intensive than the DBSE baseline products, the new MBSE process produced higher quality products. After this validation, the SWFTS SE&I program switched over to MBSE as their basic process.

Since that transformation, SE process improvement has continued apace. Requirements management has moved from DOORS to the system model in MagicDraw. As of 2016, the system model is the baseline for requirements, architecture and the FoS design. BCR impact analysis is now performed in model, leveraging capabilities of the toolset for automated assistance. Variants are documented in the system model as system configurations. Most SE products are generated from the system model on demand.

The MBSE process has been matured and refined over subsequent baseline development efforts. As the team climbed the learning curve, the average cost of processing a BCR declined. New SE artifacts capturing specific aspects of the FoS model were conceived and auto-generated from the model to improve communication with targeted audiences. Where before the transition to MBSE it was not cost-effective to manually generate tailored SE artifacts for individual systems, developing scripts to auto-generate tailored artifacts proved cost effective and improved the efficiency of the component system developers. Additional scripts were developed to tailor the modeling tool user interface and focus it on the specific activities performed by the SWFTS SE&I engineers. MagicDraw™ has very extensive modeling capabilities with a correspondingly expansive user interface, and tailoring the user interface both reduced the learning curve for new engineers coming onto the program and streamlined routine SWFTS SE activities.

While the initial scope of MBSE was limited to managing the interfaces between component systems, once the transition was successful MBSE started expanding out to encompass additional SWFTS SE&I tasks. MBSE is now beginning to spread into the component system

programs as well as the overall submarine combat system SE&I program.

## Return on Investment Achieved by Transitioning to MBSE

After the MBSE process reached a reasonable level of maturity, a retrospective analysis (Rogers III and Mitchell 2021) was performed to determine if the anticipated cost savings had been achieved. This analysis compared the requirements database-managed 2010 baselines, which were generated using the mature common requirements baseline and common change management process institutionalized by 2008, with the 2014 baselines built using the mature MBSE process. This analysis made a quantitative comparison of the efficiencies between the legacy interface requirements management process using the IBM DOORS® toolset, and the model-based interface requirements management process employing the MagicDraw™ toolset.

These two distinct SWFTS baseline updates provided a good case study as both

- utilized the same high-level SWFTS process and contract,
- involved updating the lead boat in a new group of submarines added to the SWFTS SE&I program, and
- involved updates to a similar number of submarine classes.

The second bullet is key, since it suggests that the overall level of complexity of the requirements changes is similar between the two sets of baselines.

The 2010 baseline developed using DOORS® will be referred to hereafter as the Legacy Process Baseline. The 2014 baseline developed using MBSE with the MagicDraw™ toolset will henceforth be referred to as the MBSE Baseline. The timing of these two baselines in the context of the MBSE transition is shown in Figure 3.

*Figure 3. Timing of the Legacy Process Baseline and the MBSE Baseline relative to other SE process improvement. (Rogers III and Mitchell 2021)*

Since a requirement modification is the basic unit of systems engineering work depicting development needs

at the FoS level, it is a useful metric to compare the scope of the updates. As can be seen from Table 1, the MBSE Baseline involved 42% more interface requirements changes than the Legacy Process, while consuming only 16% more hours. Another way of looking at these numbers is that the SE hours per requirement decreased from 12.1 in the Legacy baseline to 9.9 in the MBSE baseline. This is equivalent to saying that the MBSE process is 18% more efficient than the older DBSE process. This exceeded the 13% improvement projected by the 2009 SWFTS pilot study (Mitchell 2014).

*Table 1. Summary of the SWFTS MBSE ROI Analysis Baselines. (Rogers III and Mitchell 2021)*

In addition to the decrease in labor hours per requirements change, measurable improvements in quality were found. A 9% reduction in total interface defects were discovered in the MBSE Baseline compared to the Legacy Process Baseline. In addition, there was an 18% shift of defect discovery from platform integration testing to laboratory integration testing with the MBSE Baseline. Estimates of the cost savings achieved by shifting defect eradication from platform integration to laboratory integration range from 1.6x (Rogers III and Mitchell 2021) to 4x (Feiler et al. 2013), but in any case these savings can contribute significantly to the overall ROI.

# Lessons Learned

Seven learning principles (LPs) (Friedman and Sage 2005) were derived that address the more broadly applicable areas of systems engineering knowledge, and inform the areas of the SEBoK that are most strongly related to the case study. They are:

- Requirements traceability (LP1);
- Communications (LP2);
- Productivity (LP3);
- Quality (LP4);
- Managing Change (LP5);
- Managing variants (LP6); and
- Life cycle (LP7).

**Requirements traceability** LP1: MBSE improves traceability in multiple dimensions, but maintaining requirements traceability between a traditional database

and the MBSE system model can be challenging. While DOORS, MagicDraw and Teamwork can interoperate to provide requirements management and traceability, the combination is fragile. Without careful configuration management, synchronization can lead to database corruption. If DOORS and Teamwork are on separate servers, maintaining the connection can run afoul of ever-evolving corporate information assurance policies.

Requirements can be managed using the SysML language inside the system model quite effectively. This approach can reduce the resources needed to keep the system model in sync with a traditional requirements database system and increase overall SE productivity.

**Communications** LP2: Tailored SE products generated from the system model can substantially enhance communications both within the technical team and between customer stakeholders.

Graphical depictions of the system model often communicate better to human stakeholders than massive spreadsheets and textual documents. Further, the enhanced precision driven by modeling can reduce miscommunications between both technical and programmatic stakeholders.

Having the architecture and design in a system model makes it affordable to generate specialized SE products on demand for particular communications needs while keeping all SE products in concordance. Even technical stakeholders who thought they understood the design can find new insights by looking at it in different representations.

**Productivity** LP3: MBSE increases productivity by enhancing communications within the team, automating routine tasks and through cost avoidance.

DBSE processes often require substantial revision to achieve the potential productivity gains of MBSE. In particular, review processes should be modified to take advantage of the tooling.

The modeling tools selected constrain how you can practically re-engineer SE processes. Automation can replace a great deal of routine SE work (document generation, identifying potential impacts of changes, etc.).

Developing strong modeling style guidelines and specialized representations, along with training materials to indoctrinate new team members as they join

the program, is worth the investment. MBSE does require a trained cadre of modelers, but not all systems engineers have to become skilled modelers.

To effectively quantify the benefits of MBSE, a program needs to plan metrics collection carefully, and then stick to the plan long enough to collect meaningful data.

**Quality** LP4: Much of the ROI from the MBSE transition can be in improved quality. Improving the quality of SE products enhances early discovery of defects, which reduces integration costs. It also reduces latent defects in systems delivered to the customer, reducing maintenance costs and increasing customer satisfaction.

Models are less tolerant of imprecision than documents. The increased precision improves SE product quality, both in reduced defect generation and in reduced defect escape.

The automation of product generation can make specialized SE products affordable, further enhancing system quality.

**Managing change** LP5: How a proposed system change is understood and executed is fundamentally different between a model- and a document-centric approach. In the document-centric approach, the focus is on “What should my final output look like?” In the model-based approach, the focus is on “What does this change mean to the system? Which other parts of the system are impacted by this change?”

Change management is hard. When moving from DBSE to MBSE you need to think carefully up front about what approach you are going to take, and then design the system model to facilitate that approach. Change management also impacts tool selection, since different tools align better with different approaches.

**Managing variants** LP6: The most common process for managing variants in DBSE is ‘clone and own’, where each new product family member takes the then-current baseline and ‘forks’ the baseline for evolution of the variant. This makes synchronizing changes to the core baseline across the product family a very labor-intensive process. Treating variants as deviations from a core baseline in a model greatly reduces the cost of managing variation in a product family.

Variant management is hard. You need to think about what approach you plan to take up front and design your system model to accommodate it. The selected approach

impacts tool selection and tailoring.

Design the system model to treat variants as deviations from the core baseline. Then changes to the core baseline are automatically shared among all variants, and impact to product family members is limited to any impact of core baseline change on specific variant deviations. This also facilitates commonality between variants, a key customer goal as commonality reduces logistics and training costs.

**Life cycle** LP7: MBSE can be applied early or late in the product family life cycle. While most projects using MBSE start off model-based, a program can transition to MBSE late in the life cycle.

Getting from DBSE to MBSE requires serious engineering, careful thought, planning and implementation. The SWFTS MBSE transition required three years of investment by the customer. That time and budget was spent primarily in designing and developing the system model and in re-engineering the SE processes.

Start the transition with carefully defined scope. Once that is accomplished you can expand the scope of MBSE from there.

# References

## Works Cited

Cooper, D. J., J. Sienkiewicz, and M. Oliver, "System engineering and integration for submarine combat systems in the COTS environment", NDIA Systems Engineering Conference, San Diego, CA, 23-26 October 2006.

Fages, H I. 'Submarine programs: A resource sponsor's perspective," *The Submarine Review*, July pp. 53–59, 1998.

Feiler, Peter H., John B. Goodenough, Arie Gurfinkel, Charles B. Weinstock and Lutz Wrage, "Four Pillars for Improving the Quality of Safety-Critical Software-Reliant Systems", CMU/SEI White Paper, April 2013. Accessed 10 Sept 2017 at http://www.dtic.mil/get-tr-doc/pdf?AD=ADA585679

Ford, D. N., and J. T. Dillard, "Modeling open architecture and evolutionary acquisition:

implementation lessons from the ARCI program for the Rapid Capability Insertion process ", *Proceedings of the Sixth Acquisition Research Symposium: Defense Acquisition in Transition* 2 (Apr 2009): pp. 207- 235.

Friedman, G.R. and A.P. Sage, “Case studies of systems engineering and management in systems acquisition." *Systems Engineering,* vol. 7, No. 1, pp. 84-97, 2004.

Jacobus, P., P. Yan, and J. Barrett, "Information management: The advanced processor build (tactical)", *JOHNS HOPKINS APL TECHNICAL DIGEST*, vol. 23, No. 4 Jan, pp. 366-372, 2002.

Mitchell, Steven W., "Efficiently Managing Product Baseline Configurations in the Model-Based System Development of a Combat System Product Family", INCOSE International Symposium, Rome, Italy, July 2012.

Zingarelli, M. A., S. R. Wright, R. J. Pallack, and K. C. Matto, “SWFTS - System engineering applied to submarine combat systems,” *Engineering the Total Ship,* Falls Church, VA, July 2010.

## Primary References

Mitchell, S. W., "Transitioning the SWFTS program combat system product family from traditional document-centric to model-based systems engineering", *Journal of Systems Engineering*, Vol. 17, No. 2, Spring 2014.

Rogers III, Edward B. and Steven W. Mitchell, "MBSE Delivers Significant Return on Investment in Evolutionary Development of Complex SoS", *Systems Engineering*, vol. 24, No. 6, pp. 385-408, November 2021. DOI 10.1002/sys.21592

## Additional References

Gibson, B., S. W. Mitchell, and D. Robinson, "Bridging the gap: Modeling federated combat systems," *Third International Conference on Model Based Systems Engineering*, Fairfax, VA, Sept 2010.

Mitchell, S. W., “Model-based system development for managing the evolution of a common submarine combat system," *A FCEA-GMU C4I Center Symposium on Critical Issues in C4I,* 18-19 May 2010.

Mitchell, S. W., "Complex product family modeling for common submarine combat system MBSE," *Third International Conference on Model Based Systems Engineering,* Fairfax, VA, Sept 2010.

Mitchell, S. W., ”Efficient management of configurations in the model-based system development of a common submarine combat system," *A FCEA-GMU C4I Center Symposium on Critical Issues in C4I,* 24-25 May 2011.

## Note



---

< Previous Article | Parent Article | Next Article>
**SEBoK v. 2.10, released 06 May 2024**

---