Carnegie Mellon University
Software Engineering Institute

GOVERNANCE OF A SOFTWARE PRODUCT LINE: COMPLEXITIES AND GOALS

Robert Ferguson
Senior Engineer
Software Solutions Division

November 2018

Introduction

My prior blog post on product lines in DoD sustainment described the complexity of contractual relationships in a DoD software product line. Recall that a software product line is a collection of related products with shared software artifacts and engineering services that has been developed by a single organization in support of multiple programs serving multiple missions and different customers. A product line will reduce cost of development and support. In exchange, it can be a cause of conflicting priorities between customers much like the similar problem in joint program management. This blog post describes a set of guidelines and goals for establishing governance and monitoring the product line for long-term success.

The Complexity of Sustaining a Software Product Line

In this article, I refer to the core components and services as **the platform**. In the context of DoD acquisition and sustainment, each independent DoD **program** serves a unique mission. Each **program** benefits from the use of the **platform** components. Each **program** also develops its own unique capability to support its mission.

The Venn-diagram below illustrates the situation. The central circle represents the core platform components and services. Each of the three larger circles represents a single program. Each program uses some portion of the platform capability to develop its mission capability. A portion of the central platform may lie outside the context of a specific program that does not require some specific function. The programs may also overlap representing the potential for sharing additional capability across multiple programs. This shared capability also represents the potential to be captured into the platform components and services.

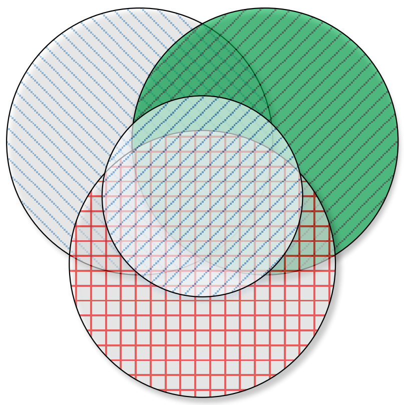

Figure: Shows programs sharing the core platform

Analysis of the Venn diagram reveals the complexity of decisions in the product line:

- Product releases must be coordinated between programs and platform.
- The combined programs provide funding for platform sustainment and development.
- The platform is responsible for recapture of technology developed by a program when the technology can support additional programs.
- The platform is responsible for "commonality and customization" capabilities of the product line.

Specific Goals of the Product Line Governance Function

Governance of the product line must support the following goals and identifiable measurable outcomes:

1. **Quality: Core components have exceptionally high quality.** Defect density of core components should be typical of best-in-class products when compared to industry benchmarks for software intensive products. The required high quality of the core can be measured. Extra costs are associated with resolving core defects for two reasons. First, programs must demonstrate that bugs are not part of program software and should be resolved instead by the platform developer. Second, the platform developer must then apply the fixes to the other programs to maintain the integrity of the product line.
2. **Cost of Sustainment: Sustainment costs of platform are reduced for participating programs.** The program community should see the cost of sustaining the platform as a fraction of a benchmark estimate for the platform cost. Working our way through an example, suppose the core is 1 million lines of code (MLOC), and there are five programs using the product line. We estimate the annual benchmark cost of the platform is 16 people (10 for code, 4 for testing, 1 for configuration management, 1 for documentation). We estimate the cost of supporting a single program is two additional people for liaison and configuration. Supporting 5 programs the annual cost looks like the following.

Platform sustainment cost must include cost of support to programs:

*Benchmark Cost + overhead*Number of programs=Platform Cost*

*Example: 16+2*5 = 26 people*

Program cost of platform components is reduced by cost sharing but adds some overhead:

Platform Cost/number of programs + overhead = Program Cost

Example: 16/5 +2 = 5.2 people

3. **Reduced time to add features to programs: New features in the core can be rapidly deployed to multiple programs in the product line.**
   Rapid deployment is achieved by product design that isolates variability to a smaller number of specific components. It is also achieved because propagating an improvement to one program simplifies the effort required to provide the same functionality to another program. The high quality of the core platform contributes significantly to the reduced time to deliver.
4. **Reduced time to distribute capability between programs. The core platform has the strategic capability to recapture newly developed program capabilities into the core platform.**
   Capability captured by the core platform provides all the product line capabilities to other programs (documentation, testing, etc.). Cost of recapture should be weighed against the cost of changing multiple individual programs.
5. **Releases occur on a regular schedule. Releases are synchronized within a reasonable timeframe to avoid increasing the difficulty of managing multiple different configurations of the core.**
   Multiple configurations of the platform add complexity to the sustainment and support, and, thus, increase costs and cause quality problems.

Each of the above five goals suggests potential measures.

1. Measure defect density, frequency of bug reports and estimate changes in mission-capable availability due to defect removal.
2. Monitor annual cost of platform sustainment and cost per program for support of platform
3. Track time to propagate technology to other programs after initial development.
4. Track time to recapture program developed capability into platform functionality.
5. Releases from platform occur within agreed schedule. Number of configurations of platform is controlled.

Developing a Governance Policy

Governance requires a high-level decision authority, which often contributes to a slower decision process. Therefore, policy must be carefully articulated to allow most decisions to be made at the platform or program level to minimize the effects of the slower process.

The policy should address the following:

Programs share funding of core platform support. Funding of the platform can be considered as similar to a license fee for normal support in implementing program changes. Certain program change requests may require changes to the core platform. These changes may incur additional costs that must be supported by the program office. Usually, core platform changes should be supported by the program office requesting the change. However, changes to the underlying technology may need broader financial support. The product line governors should be prepared to help negotiate this larger scale request across the program community.

Programs and core platform plan and execute a regular release cycle, probably between 12 and 24 months. While the actual program releases are all independent, any discussion of changes to platform core, including documentation, testing and support, should be discussed jointly by programs. If programs operate independently, then the core platform will struggle to synchronize releases and will experience new sources of rework. The other participants in the product line need an opportunity to negotiate schedule concerns.

Platform team leads release decisions for changes to the core affecting commonality and customization. Product lines are known for the ability to isolate product variation for customization. In fact, the concept of variability separates product lines from small-grained reuse. See *A Framework for Software Product Line Practice, V5* (page 6). Changes to core products affect all elements of the product line—code, documentation, test cases, etc. Insistence by programs on making frequent changes can adversely affect the careful attention to the quality and support for variability suggested by the preceding reference.

Programs and platform jointly evaluate new technology to determine whether changes to the core are appropriate. Changes to system technology are often suggested to improve various aspects of system performance, safety or security. Changing the core platform to support the technology will affect core architecture, components, and modeling (variability and customization). These are expensive efforts, often requiring investments across multiple releases. See *Framework* paper page 9.

Platform uses a consistent estimation method for changes to the core platform. The product line concept introduces some additional complexity since the core platform essentially develops an infrastructure supporting the various programs. The governors must understand the variable parts of the cost equation when funding the core platform. A useful paper describing the estimation process was developed by Rolls-Royce. See Nolan, et. al *A Case for Product Lines*.

Looking Ahead: A Shared Commitment

A product line in the DoD requires a shared commitment across a number of programs. Because each program has its own program management office (PMO), some governance structure is needed to maintain the alignment of goals, thus keeping the product line viable across multiple releases of the products.

In the next post in this series, I will describe a decision process for release management. It is designed to support both a single product and the core program for a product line.

Additional Resources

- Read the first post in this series.
- Read the series by Mike Phillips on Efficient and Effective Software Sustainment.
- View the podcast Software Sustainment and Product Lines by Mike Phillips and Harry Levinson.

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412/268.5800 | 888.201.4479
Web: www.sei.cmu.edu
Email: info@sei.cmu.edu