Software Architecture

Report on
Software Product Line

Submitted to: Dr. Hany Ammar

Submitted by: Hadeel Ahmed ElGenedy
Submitted on: May 9th 2009
Table of Contents

Introduction .................................................................................................................. 3
Software Product Line Definition: .................................................................................. 4
Software Product Line Concepts..................................................................................... 5
Binding Times................................................................................................................ 6
Moving Towards a Software Product Line: ................................................................. 9
Software Product Line Purpose ................................................................................... 10
Software Product Lines Benefits ............................................................................... 11
References .................................................................................................................. 13
Introduction

Business today holds more promise for software applications than ever. In fact, many organizations including those that never envisioned their business to be in the software business realized the importance of software applications and the power enhancements it provides. As has been the case for the last decades, software has been evading all sectors of the business and social economy, which calls for enhanced and more powerful engineering and architectural methodologies to be used to develop such software. Organizations around the globe share universal goals that distinguish a successful business from a mediocre firm. A successful business aims at high quality, marketing quick time, market dominance, market agility, product alignment, low cost products, low cost maintenance and mass customization. To achieve such high standard goals, a firm is required to improve its efficiency and productivity. To address such issues an organization can improve their process, have more technology innovation or start with software reuse. Since many organization produce similar systems, a reuse strategy can become very handy. Studies have shown the percentage of reuse in software systems is substantially high and is always increasing. Hence, the need for a strategic reuse strategy is highly important. For years, reuse strategies has been deployed for subroutine, modules, objects, components and services. This development ladder leads to the development of product lines. See Figure 1 for reuse history.

![Reuse History](image-url)

Figure 1: Reuse History
Software Product Line Definition:

A simple definition for product line is the set of related products which are produced by a single organization. However, the Software Engineering Institute \(^1\) gives a formal definition for software product line as *a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are development from a common set of core assets in a prescribed way.*

Software product line is an engineering technique that is used in creating a portfolio of similar software systems from a shared set of software assets using a common means of production. The concept is in many ways confusing with the reuse strategy. However, SPL aims at creating software artifacts that are predicted to be reused in one or more products in a well defined product line. The traditional reuse technique puts generic software components into a library hoping that a reuse will arise some day. Software product line is not just an engineering technique rather than a set of techniques, tools and methodologies.

Moreover, software product lines are not plain technical practices that can be carried out with a one phase strategy. They require strategic thinking that encourages, direct and organize the user or common sets of assets in software production. Technical resources can’t just take a decision of reusing common assets on their own. Management must be in line with such decision not to mention the enforcement to the use of such assets.

The output of product line usage is substantially high. A great economic advantage is taken by the fact that many products are similar, and it is not a co-incident rather than a business plan to produce similar products to enhance their quality and maintain a software product line on a high level of sophistication.

\(^1\) Carnegie Mellon University, Software Engineering Institute web site.
Software Product Line Concepts

Given the fair concise definition of a software product line, it is essentially definite to describe the software product line concepts. SPL can be described in four simple concepts/terms, see Figure 2 for illustration.

**Software Asset Inputs:**
This component is considered the core component for software product line, or the base one. It is a collection of software assets that vary from requirements, source code components, test cases, architecture, and documentation. The main characteristic between these components is that they can be configured and composed in different ways to create all of the products in a product line. These assets are managed by different ways to accommodate variations among the products. In practice, not all components are required some may be optional in the products to provide the variation between various products. In other cases, these assets may have internal variation points that are managed and configured in various ways to provide different functionalities within different products.

**Product Decisions:**
The product decisions in a product line form a decision model that describes optional and variable features for the products in the product line. Product decisions can be seen as choices that uniquely identify each product in a product line. A decision can be made to change a variation point in a product such variation will provide the distinction between different products in a product line.
Production mechanism and process:
The mechanism and process in this stage can be viewed as a set of tools that apply the product decisions to software assets in a product line to produce the product, or an instance of a product. In this stage, it is defined how to configure different variation points within the software assets inputs in the product line. The production mechanism is considered the means for composing and configuring the products from the software asset inputs.

Software Product outputs:
Simply it’s the product, or in other words, the collection of all products that was produced for that product line. The scope of the product line is identified by the set of software product outputs that can be produced from the software assets and decision model.

Binding Times

The previous section highlights the role of product decisions in maintaining the variation in the products produced by the product line. Moreover, this variation strategy is what makes the product line distinct from other software engineering methodologies. The product decision concept in a product line is what defines the variation between products, and the time at which such decision can be taken are referred to as binding times. However, it is possible for a single product line to utilize multiple binding times. Such decision can be made by different roles in the product line as well at different stages in the software life cycle.

Product line production exhibits different binding times such as:

- **Source Reuse time:** Decisions bound when reusing a configurable source artifact
- **Development time:** Decisions bound during architecture, design, and coding
- **Static code instantiation time:** Decisions bound during assembly of code just prior to a build
Software Product Line Overview

- **Build time:**
  Decisions bound during compilation or related processing

- **Package time:**
  Decisions bound while assembling binary & executable collections

- **Customer customizations:**
  Decisions bound during custom coding at customer site

- **Install time:**
  Decisions bound during the installation of the software product

- **Startup time:**
  Decisions bound during system startup

- **Runtime:**
  Decisions bound when the system is executing

Multiple binding times calls for partially instantiated production stages, in other words, with each variation binding a product is produced from some assets and production decisions, such product is not considered the final product, further production mechanisms should be applied resulting in a partially instantiated assets that will be applied to further decisions to reach the final product. See Figure 3 for illustration.

![Figure 3: Multiple Binding Times](image-url)
As different production decision produce variation in products, also different production mechanism and process produces different products. In fact, production is a key discriminator between the different software product line approaches. There are three distinct characteristics that define the production mechanisms, see Figure 4 for illustration.

The first characteristic of production mechanism is automation. Automation refers to production in software product lines that can be fully automated, completely manual or somewhere in between. In fully automated approaches there is sufficient information that enables the user take product decisions that automatically generate the product output. On the other hand, an example on a completely manual approach is a textual production plan where software engineers interpret and follow directions in the plan and in the product decision to be able to tailor, integrate and provide the sufficient mechanism to create the product.

Production in a product line is similar to the traditional software engineering as it is not a one-shot activity. That’s where the second property of a product line production phase can be defined, which is periodicity. Periodicity refers to the fact that products may need to be periodically reproduced to reflect enhancements to the software assets or decisions.

The last property that distinguishes a production mechanism from the other is the role. Product line approach define a separate human role for the production activity called the application engineering, which is a different role from the domain engineering role which is responsible for engineering the software asset inputs. Other approaches do not make such difference, but rather give separate roles for approaches with manual production, and single role for fully automated production approaches.
Moving Towards a Software Product Line:

Many organizations aim at moving towards a software product line, but lack the appropriate measures and strategies to get started. As mentioned, adapting an approach is not a one-shot activity but rather an iterative steps and stages that need to be implemented on both the management and technical levels. The below figure 5 illustrated the five main stages that an organization member should process before trying to achieve a software product line approach in his working organization.

To get started on a product line strategy in an organization, one has to become informed about the software product lines. The knowledge about the software product line must be utilized and enhanced to the maximum. One has to take advantage of the resources available to them to have all the required information about software product line.

The next step in the process is to assess the software product line opportunity within the targeting organization. Pros and cons of having a product line approach should be assessed in reference to the organization. Such assessment can be essential in defining whether a product line has a place in the organization or will do more harm than benefits upon implementation.

Once the assessment has been made and it has been defined that a software product line fits within an organization, the team should be assembled. Those team members must be capable and motivated to lead the transition to software product lines, as change is not always the best case scenario for most people. The team should include individuals that provide strong, clear, constant and consistent leadership.
As any person starts out their life with a role-model, a software product line should start in an organization with an ideal success story. After a team has been made, and enough information about software product line and the organization has been gathered, a clear strong vision must be stated. Such vision can be brought from an ideal success story that an organization wish to accomplish. This vision will be used for the team as their ultimate goal for promotion, success and acceptance of the software product line in the organization.

Finally, it is essential to promote for the software product line in the organization by showing its quick benefits, thus gaining much acceptance within the community. One way to do this is by doing a pilot project it both addresses a small but complete part of the problem and solution. The purpose behind this step is to energize the change agents in the new team and highlights the points that a product line was implemented for in the first place, hence builds a spirit for the effort that has been taken.

**Software Product Line Purpose**

At first glance, the definition for software product line can be confused with much architecture that enhances the engineering reuse strategy. However, this is not the case with software product line. The key objective of software product line is to reduce the time, effort, cost and complexity of creating and maintaining a product line of similar software systems. Such efficiency can be achieved through the below means:

- **Capitalize on commonality:**
  This can be achieved by sharing and consolidation between the software asset inputs which in return prevents duplication, rework and divergence. Hence reduce time, effort and cost.

- **Manage Variation:**
  The decision model supports this point by defining the variation points in configuring the software assets thus making the logic and dependencies between the product and the assets as explicit as possible.
Software Product Lines Benefits

The main benefit behind a software product line approach is its ability to efficiently create many copies of the same product this can be referred to as mass production. On the other hand, mass customization as well is considered a major advance in the manufacturing business and software engineering as well. Mass customization refers to the ability to efficiently create many variations of a product. This variation as discussed earlier can be achieved by the product decisions that configure the assets and produce various products accordingly.

Moreover, software product line techniques can enhance the productivity of software engineers. The reduction in the effort and cost that is required to develop, deploy and maintain a collection of similar software products. Such increase in the productivity was reported in case studies\(^2\) and range between factors of 2 to 3. The below graph (Figure 6) shows the effort required to develop, deploy and maintain a collation of similar software products. The greater the total number of products, the greater the total effort and cost.

![Figure 6: Benefits of Software product line](image)

The productivity increase by SPL does not only affect a company’s finance but rather have a strategic top line benefits. Such productivity benefits can lower the prices hence increasing the profit margin for products which in return provide strategic competitive advantage.

Moreover, scalability is considered a major advantage for software product lines. In general an organization that uses SPL approach should scale without constraints to the number of products are optimal for their business. From figure 6 one can see that both traditional and SPL approaches do have limits to the number of products that can be effectively developed and maintained. However, the graph in figure 7 shows a steep rise in effort when an approach reaches its complexity limit. The point is that SPL approaches can often scale to more products than the traditional software engineering techniques can hold.

\(^2\) Charles W, Krueger, “Benefits of Software Product Lines”.
In conclusion, the scalability enhancement by the software product lines, does not just benefit an organization on the number of products that are produced but have other strategic impacts as well. Scalability provides strategic competitive advantages as it allows going into larger number of markets with tailored products. In addition, scalability helps in creating larger numbers or more narrowly focused products within a market. Finally it provides greater variety and more targeted products than competitors.
References


