

Software Product Line Considerations

Paul M. Drumm

Abstract

Software product line considerations are not to be taken lightly when deciding to move your organization from a single project, silo based approach to a core asset based system. A multitude of organizational attributes must be considered before making the final decision on pursuing a product line model for your organization. An organization must determine what their motivation is for moving to a software product line, assess their ability to transition to a software product line, determine the scope of the software product line, and choose a methodology to achieve a software product line that best fits their organizational environment if they are to succeed in moving to a software product line. This paper presents a summarized overview of these four concepts after defining what a software product line is.

Introduction

Mary and Bob are two software project managers that work for a large software company. They are discussing software product lines and development models in light of a new project their company has taken. Mary argues that they should build a flexible software product line and then mold a development model around it. Bob argues that they should choose the proper development model and that the software product line is not necessary since the development model can be considered a software product line. Bob also argues that the software product line is just a managerial strategy to produce the intended software product.

Neither Mary nor Bob's argument is correct and the position of this paper is in disagreement with both. They are taking two narrow views that do not account for other considerations when deciding whether to build a product line or continue with the existing system. Of their arguments, neither considers the existing product set nor how the new project aligns with the current product set. Furthermore, Bob is definitely wrong in saying that product lines are just a managerial strategy to produce the intended software as this is clearly lacking in the definition of a software product line which is discussed in this paper. An organization should not equate software product lines with development or managerial strategies to produce software. An organization should approach software product lines with four things in mind which are discussed further in the paper:

- 1) Organizational reasons to move to software product lines
- 2) Assessing the organizations' ability to move to software product lines
- 3) What the scope of the software product line will entail
- 4) Choosing a methodology to achieve a software product line that is synergistic with the organizations capabilities.

It is only after examining these four concepts that you can make an organizational statement about moving to a software product line.

Software Product Line Definition

A definition of Software Product Lines (SPL) is first in order. Clements and Northrop by far have been quoted the most with their definition so I repeat it here.

A software product line is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way [Clements 02a].

SPL involves reuse at the highest level. The reuse is a planned event in an SPL. The reuse is enforced by your organizations policies and procedures as constraints and eventually becomes ingrained into one entity, the SPL. Reuse is done at a large scale across all facets of the software development life cycle (SDLC). Requirements, components, architectures, deployments, process control, etc; all are involved in reuse and all may be tailored to each product in your product line in a slight variation of one another but that variation has been built into the SPL architecture as a designated point of variation.

Determining Motivation for Software Product Lines

Many software development organizations have found themselves in a position where they either on purpose or by necessity have made a collection of similar products or are planning a set of similar products based on an existing product. Once in this position, then either by choice or by someone's fortuitous knowledge, a chain of events is then set into play where the realization comes that having a product line would be advantageous to the current state of the organization. Those events can take many forms and each one should be considered to see if it applies to your organization.

1. Multiple products with the same characteristics will cause economies of scope - A realization may come that that having similar products may cause some cost saving benefits of some form and that all it would take is a study of some sort to see if that may be the case.
2. Shortage of resources – A shortage of resources may exist, such as programmers, requirements analysts, or testers, that may be alleviated by conforming existing products into some sort of unified product which may decrease the resource shortage affect in some way.
3. Redundancy and waste. - A realization everyone is doing the same thing but on different projects for different customers may come to pass and that combining the efforts with slight variations on a single product would make it realize economies of scope.
4. Time to Market for future products. – Turning to a core product line will bring a shorter time to market for future products going to different customers. When you achieve your product line template, all that needs to happen is to punch out a few custom slots and give it to our customer.

5. Addition of new product to existing product assets. – You may acquire a product from another company only to find that it is very similar to existing products with a slight variation. There may be other companies on the market in which you would like to acquire certain of their products in order to consolidate your product line and make it complete. Having a product line would make this a very efficient process during acquisition and integration.

6. Reuse – Realizing, that, unlike an automobile manufacturer who makes a new, albeit the same, engine for every car, that you can build just one engine, be it a requirements engine, or a test engine, or a deployment engine, and put it in every product.

Once you have a coherent set of motivations a vision document should be assembled to guide your organization as they work through the assessments, scope, and methodologies.

Assessing Organizations Ability to Transition to Software Product Lines

Another consideration that should not be overlooked is that most sources [Chastek01] [Clements02a] recommend a survey, probe, or fact-finding mission within the organization to help determine your readiness for SPL. This will consume many resources but is a necessary investment. The Product Line Technical Probe (PLTP) is one of the recommended methods of assessing whether an organization is ready for product line adoption [Clements 02a] [SEI 07a]. The PLTP uses the Software Engineering Institutes' (SEI) Framework for Software Product Line Practice as the reference model for the probe [Clements 04a].

The PLTP is basically interviews with the stakeholders using the framework's 29 practice areas which are grouped into three essential product line activities; Software Engineering, Technical Management, and Organizational Management. There are pretexted questions laid out from each of the 29 practice areas in order to give the PLTP a repeatable look, feel, and results. The questions are designed so that a good response can be elicited from the interviewee. The various stakeholders from your organization, both internal and external and may include your customers, are all part of the interview process but they may have different question sets applied to them from each of the 29 practice areas. The people selected for the interviews should be subject matter experts (SME) that are highly knowledgeable about the organization and should be able to function as a contributor to the interview which is usually a group interview. The interview may not ask questions from all 29 practice areas if some do not pertain to your organization. The PLTP team should also consider some light training in team building and interview skills if they have not been through a PLTP for a while.

There are three phases to be considered in the technical probe. The preliminary phase, the technical probe phase, and the follow-on phase (optional). The preliminary phase is usually a small meeting with some SMEs. This first phase will generally set the context for the next phase. The PLTP teams works with the organizations representatives to gain some initial insight into the organization and to help set goals for the effort. Some preliminary context questions may be given to the organization in advance [Clements02a]. This gives them time to think about the context and also to gain an understanding of the context of the organization as it stands. What results from this phase is an understanding of what questions to ask in the interviews from the 29

practice areas, to whom the questions should be asked, and some predetermined times and schedules for all those involved.

The technical probe phase is the heart of the PLTP. An initial meeting is held with the PLTP participants and then the scheduled interviews begin with groups of similar stakeholders. The data gathered during these interviews is held in confidence in order to obtain the high quality of information as possible. Questions are asked from each of the 29 practice areas and the best response is elicited from the group. During the PLTP, some preliminary results are gathered and analyzed so that practice areas that did not get enough information or have questionable information can be revisited in another interview. This is an iterative process and it may happen several times so the team has adequate and high quality information to present. After all the interviews are over the team prepares a final report. Each of the applicable 29 practice areas are looked at to see how the organization compares to what is needed for a product line based organization. If an organization is weak in a certain area then a recommended course of action can be instituted.

The final phase in the PLTP is the Follow-On Phase and may be optional. It is most beneficial to those organizations that had weaknesses. The organization may need assistance with planning the improvements and a SEI Product Line Planning Workshop [SEI 07b] may be used to tailor an action to improve the weak practice area.

How to Determine the Scope of the Software Product Lines

Scope is very important when considering moving to a SPL. It determines what will be in your core asset collection and affects your core asset and product development, which are two of the three essential activities of fielding a product line [Clements02a]. A core asset is an artifact that can be common to all products in a SPL. It may be requirements management, an abstract base of software classes for a search engine, or a deployment methodology, all of which will have certain touch points or variations points that enable you to tailor that core asset to the product you are developing or considering to develop.

The scope should not be too over reaching or too small and will be determined by the vision of your SPL and the motivating factors behind it. If you make your scope too large then you may have an overabundance of core assets. This may preclude any financial gains from having economies of scope and the organization may revert back to a “product at a time” methodology of making software. On the other hand, if you make your asset base too small then your variation points will have a smaller delta to the base asset and you will spend too much time tailoring that asset to a particular product or making too many variation points in that asset. It may also cause you to create more custom components for each product or you may not be able to offer enough products to the market and your stakeholders.

You should carefully consider what is common across your existing product base and look 3-6 years down the road at what you will make or what the market may demand of you. This is not to say that your choice of core assets will be set in stone once you make that decision. Your scope will change and iterate as your organization adjusts to stakeholder requests and market conditions.

The table below shows the considerations you should take on the inputs to looking at the product line scope [Clements02a]. The same inputs also apply to the core assets and production plan.

Inputs	Core Asset Development	Outputs
<p>Product Constraints</p> <ul style="list-style-type: none"> ○ Commonalities and variations ○ Physical constraints ○ Markets ○ External Interfaces ○ Standards <p>Styles, patterns, and frameworks</p> <ul style="list-style-type: none"> ○ Large grained components ○ Protocols ○ Patterns ○ Preferred architectural style <p>Production constraints</p> <ul style="list-style-type: none"> ○ Regulations ○ Standards ○ COTS ○ Legacy issues <p>Production Strategy</p> <ul style="list-style-type: none"> ○ Top down or bottom up ○ Assembly patterns ○ COTS ○ Costing <p>Inventory of Preexisting Assets</p> <ul style="list-style-type: none"> ○ Legacy components ○ Process Asset Library ○ Frameworks ○ CMM/CMMI practices 		<p>Product Line Scope</p> <ul style="list-style-type: none"> ○ Products ○ Common Features ○ Quality Attributes <p>Core Asset List</p> <ul style="list-style-type: none"> ○ Common Architecture ○ Process Specs ○ Training ○ Evolution <p>Production Plan</p> <ul style="list-style-type: none"> ○ Usage of assets ○ Process Plan ○ Audience ○ Training

Choosing Methods to Achieve Software Product Lines that fit your Organization

There are various methods to help your organization achieve a mature SPL status one of which to consider is the SEI *Framework for Software Product Line Practices* [Clements 04a] and its associated case studies and product line practices patterns [Clements 02a]. These works break down the three essential activities of fielding a product line into Core Asset Development, Product Development, and Management. It then goes on to field 29 practice areas your organization should be following, if deemed appropriate, to achieve a successful implementation of SPL. These practice areas are also the basis of the Product Line Technical Probe discussed earlier in the paper. Two of these activities, Core Asset Development and Product Development, are the focus of the Software Engineering practice areas that provide the technical prowess to build and grow the core assets and the products in the SPL. The remaining essential activity, Management, has two practice areas broken out of it. Technical Management practice area is for

building and growing core assets and the products in the SPL. Organizational Management practices for coordinating the SPL at the organizational level. These are listed below with hyperlinks [Clements 04a].

Software Engineering	Technical Management	Organizational Management
Architecture Definition	Configuration Management	Building a Business Case
Architecture Evaluation	Data Collection, Metrics, and Tracking	Customer Interface Management
Component Development	Make/Buy/Mine/Commission Analysis	Developing an Acquisition Strategy
COTS Utilization	Process Definition	Funding
Mining Existing Assets	Scoping	Launching and Institutionalizing
Requirements Engineering	Technical Planning	Market Analysis
Software System Integration	Technical Risk Management	Operations
Testing	Tool Support	Organizational Planning
Understanding Relevant Domains		Organizational Risk Management
		Structuring the Organization
		Technology Forecasting
		Training

Each practice area is decomposed into several parts.

1. Definition and Description; A high-level summary of the practice followed by a more detailed analysis.
2. Aspects Peculiar to Product Lines: How that practice is specifically imposed on the product line.
3. Applications to Core Asset Development (CAD): How to apply the practice to the CAD.
4. Application to Product Development (PD): How to apply the practice to the PD.
5. Specific Practices: A more detailed description of how to carry out the practice in the SPL.
6. Practice Risks: Gives you some insight into what could go wrong.

Applying the practices areas is more of a tailoring of the schema of the practice to your specific circumstances. It should be remembered that this is a “Framework” and should be approached in that fashion at all times. For a method of how the changes are introduced into the organization, you may want to look at the SEI’s IDEAL model as a model for implementing change [McFeeley 96].

PuLSE or Product Line Software Engineering [Bayer 99] is another methodology one’s organization might consider when pursuing a SPL. It provides a tailorable method for the inception and deployment of SPL. It has three main components; the deployment phases, the technical components, and the support components. This model was born out of a necessity for a more product-centric methodology as the domain centric models at the time were not proving

effective. The PuLSE method claims that it can establish SPL across a wide variety of enterprise domains by a) being product centric throughout the methods, b) having the ability to customize each component, c) being able to incrementally introduce the system, and d) providing a maturity scale for evolving the processes.

PuLSE starts with the deployment phases. These are the Initialization where the enterprise is baselined. This flows into the Infrastructure Construction where the SPL infrastructure is scoped, modeled, and architected. This flows into Infrastructure Usage, which gives you the products via instantiations. Finally, the Evolution and Management interacts with the other three to grow the infrastructure components and to manage its course.

The technical components interact with the deployment set phases to provide engineering operations to the SPL. There are six components to it. 1) Customizing which tells us how to perform the Initialization phase. 2) Scoping the SPL based on product definitions. 3) Modeling the SPL. 4) Architecting the SPL by developing reference architecture and relating it to the model. 5) Instantiating – how to enact the usage phase. 6) Evolving and managing which deals with how to integrate odd products and interactions with configuration management.

The last group is the support components. These provide constraints and instructions to the other components in the deployment phase. You have 1) the Project Entry Points that provides tailoring to project types, 2) Maturity Scale, which provides an integration and growth path for the SPL, and 3) Organization Issues, which give tailoring guidelines to the organizations structure and management of SPL. Each one of the PuLSE phases and components has specific actions laid out for it. However, these actions are tailored to your enterprise context in order to make it work.

One of the more interesting features of this model is in the Construction phase. It works through three sub channels called PuLSE –Eco (Economic Scope), PuLSE-CDA, and PuLSE-DSSA (Domain Specific Software Architecture) [Bayer 99] which flow one after another. The inputs for this come from stakeholder information, business objectives, and system information. The PuLSE-Eco lays out the SPL candidates, performs a mapping and documentation of them, a benefits analysis, and then a final product map or scope. The PuLSE-CDA takes this one-step further, refines that scope, and further decomposes them. Storyboards and decision models are built and passed to the PuLSE-DSSA. Out of this phase, a reference architecture and decision/configuration model emerges [Bayer 99].

There are other methods available and hybridizations of these methods [SEI 05b]. This section was not meant to be a detailed analysis on available models but to show some differences in what may be available to your organization. The main take away from this section is that you should consider a variety of methodologies and frameworks and then decide which method will work best when tailored for your organization.

Conclusion

Determining if your organization should create and pursue software product lines should not be a single thought consideration. A myriad of high-level concepts must be approached at all levels

of your organization. You should first start with what is motivating your organization to migrate towards the SPL concept. Next, you should assess whether your organization is conceptually ready for a SPL by using an assessment methodology such as PLTP. You must then consider the scope of your SPL. This should be done with the utmost thought and consideration as you will have to accommodate this scope for time period while you get your SPL off the ground. You should look 2-5 years out while creating the scope and you can always evolve the scope later as your markets and requirements change. The method you chose to proceed with a product line should be a framework or other model that you can tailor to your organization such as PuLSE or SEI's Framework for Software Product Lines. Lastly, a SPL is not just a managerial strategy but also a conceptual approach that lays the architecture for your core assets, your products, and your organization.

References

- [Chastek 01] Chastek, G.; Donohoe, P.; Kang, K.; & Thiel, S. [*Product Line Analysis: A Practical Introduction*](#) (CMU/SEI-2001-TR-001, ADA396137). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, <<http://www.sei.cmu.edu/publications/documents/01.reports/01tr001.html>> 2001.
- [Clements 02a] Clements, Paul C.; Northrop, Linda M. *Software Product Lines: Practices and Patterns*, Addison-Wesley, 2002.
- [Clements 04a] Clements, Paul & Northrop, Linda. *A Framework for Software Product Line Practice*, V4.2. <<http://www.sei.cmu.edu/productlines/framework.html>> (2005).
- [SEI 05b] Graaf, Bas; O'Brien, Liam; Capilla, Rafael. [*R2PL 2005—Proceedings of the First International Workshop on Reengineering Towards Product Lines*](#) (CMU/SEI-2006-SR-002), <<http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06sr002.pdf>> (2005).
- [SEI 07a] Northrop, Linda. *The Product Line Technical Probe*. <<http://www.sei.cmu.edu/productlines/pltp.html>> (2007).
- [SEI 07b] Jones, Larry. *Product Line Action Planning Workshop*. <http://www.sei.cmu.edu/productlines/products_services/ap_workshop.html> (2007).
- [Bayer 99] Bayer, J.; et al.: .PuLSE: A Methodology to Develop Software Product Lines., 5th Symposium on Software Reusability (SSR'99), 1999.
- [McFeeley 96] McFeeley, B., "IDEAL; A Users Guide for Software Process Improvement", An SEI Handbook, CMU/SEI-96-HB-001, Feb 1996