Available online at www.sciencedirect.com

SCIENCE DIRECT®

Science of Computer Programming 57 (2005) 73–87

Science of Computer Programming

www.elsevier.com/locate/scico

A Product Line engineering practices model

François Coallier, Roger Champagne*

Department of Software and IT Engineering, École de technologie supérieure, 1100, rue Notre-Dame Ouest, Montréal (Québec), H3C 1K3, Canada

Received 15 March 2004; received in revised form 5 August 2004; accepted 19 October 2004
Available online 13 January 2005

Abstract

This paper describes work in progress towards the elaboration of a Product Line practices model that combines concepts proposed by various authors. The strengths of existing Product Line frameworks and models are summarized and a new model is proposed in the form of 31 Product Line practice areas, grouped in five categories. An important objective of this Product Line practices model is that it should be easily incorporated into existing development methodologies, while remaining aligned with existing systems engineering standards.

Keywords: Modeling; Product Lines; Software engineering; System analysis and design

1. Introduction

The software industry is constantly seeking new ways to improve competitiveness. One approach to remain competitive is to attempt to reduce costs through reuse of various software elements and related artifacts. Software Product Lines (SPLs) are a relatively new concept in the software industry, but they have already demonstrated a potential for tremendous benefits such as high reuse, reduced time-to-market and requiring less staff to produce software.

* Corresponding author. Tel.: +1 514 396 8825; fax: +1 514 396 8684.
E-mail address: Roger.Champagne@etsmtl.ca (R. Champagne).

0167-6423/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.scico.2004.10.006

Although the potential benefits of SPLs are overwhelming, implementing such a Product Line is by no means a small feat. In most of the successful documented case studies [3], the benefits are clearly demonstrated, but the organizations involved are also quick to remind us that implementing a Product Line is not something that happens by itself [1], and most of these organizations went through tough times in the process of implementing their Product Line, as a number of things can go wrong.

We believe that developing and managing a Product Line has some commonalities with what is referred to in the IT world as enterprise architecture. Enterprise architects, like Product Line architects, must:

- have a vision that transcends the deliverable of the current projects;
- have a vision that is aligned with their organization's business objectives, not only the current requirements of its stakeholders;
- develop an architecture that will enable the long term vision, not only the current requirements;
- integrate in their architecture components that must be reused among many products (or applications in the case of the enterprise architects).

The work in progress described in this paper is an attempt to build a Product Line practices model that is inspired from the enterprise architecture paradigm, and that also builds on the work performed at the SEI [4] in this area. A major difference between our Product Line practices model and the SEI framework is that ours has an explicit system wide scope. Another important source of inspiration for some concepts we propose is the work of Dikel et al. [6]. The primary motivation behind the development of our model is to enable assessment of organizations developing software-intensive systems for Product Line capability. The proposed model has been used once by one of the authors for a Product Line practices assessment for a major manufacturer of transportation systems. However, the actual results of the assessment are proprietary information and are not discussed in this paper, which focuses on the model developed for the assessment and the rationale behind its design.

This paper is structured as follows. In Section 2, we take at look at Product Line history and motivate our work. In Sections 3 and 4, we summarize the characteristics of the two main sources of information, namely the SEI's Product Line Practice Framework and the VRAPS model by Dikel et al., that inspired the proposed model, which is in turn presented in Section 5. We discuss integration of Product Line practices with current development methodologies in Section 6. Section 7 summarizes the contributions of our research and describes follow-up work to be carried out in the future.

2. Product Lines: History and motivation

Definitions for Product Lines abound. A quick search on an Internet search engine returned many hits. Here is one of them [13]:

> *"A group of products marketed by an organization to one general market. The products have some characteristics, customers and uses in common and may also share technologies, distribution channels, prices, services and other elements of the marketing mix."*

Today, most of the manufactured goods we use or encounter are the result of Product Lines: automobiles, appliances, furniture, the hamburger we eat at our favorite fast food chain (for an entertaining and interesting discussion on an enchiladas Product Line, see the "Product Lines Everywhere" sidebar in the first chapter of [3]), the list goes on and on. The advantages of using a Product Line to manufacture similar goods which differ by relatively small variations are numerous and compelling. Many different products can be built from a small number of different parts or core assets, hence reuse is high. Because the same parts are used in different products, the same people (or manufacturing equipment) can be used to build the various products (yet more reuse!). Moreover, time-to-market is usually reduced substantially once the Product Line is up and running. When he prepared a proposal for his first major rifle contract for the U.S. government, Eli Whitney promised to build 10,000 muskets of different models in two years for $13.40 each. Had anyone else but the inventor of the cotton gin made that claim, it would have been perceived as sheer madness at the time (early 19th century). The following citation, which is still enlightening today, discusses the end result of this early Product Line effort [14]:

> *"Almost eight years was required for Whitney to fill the order, because practice still showed many gaps in his system. The number of details seemed endless. However, most of the ten thousand were turned out in the last two years. In 1811, Whitney took an order for fifteen thousand, and these were turned out within only two years."*

This illustrates an important point. Although the advantages of using Product Lines are compelling, implementing a Product Line requires a lot of effort. A Product Line typically requires an important up-front investment (in Whitney's case, designing a common architecture for the various types of rifles, tooling the manufacturing facilities, adjusting the entire manufacturing process to account for "the endless number of details", etc.) and a different corporate mindset, among others.

As in the manufacturing industry, reuse and reduced time-to-market have become major preoccupations in the software industry also. Although object-oriented technology held great promise with respect to software reuse, its success in obtaining such reuse is debatable (a discussion on this particular topic is beyond the scope of this paper). However, the idea of applying the Product Line concept to software development, although a relatively recent idea (one of the better documented success stories started in the mid 1980s [2], the SEI's work on Product Lines started around 1995 [3]), has proven successful in a number of organizations and there is a growing community of SPL practitioners.

Before going any further, it is important to define what it is we mean when we discuss a *software* Product Line. We will adopt the following definition [4]:

> *"A software Product Line is 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 developed from a common set of core assets in a prescribed way."*

This definition is pretty close to what is commonly accepted as being a Product Line nowadays. If we compare this definition to the one mentioned earlier for a general Product Line (e.g. not specifically software), we notice that the new definition emphasizes concepts such as a "*managed set of features*", "*a common set of core assets*" and development

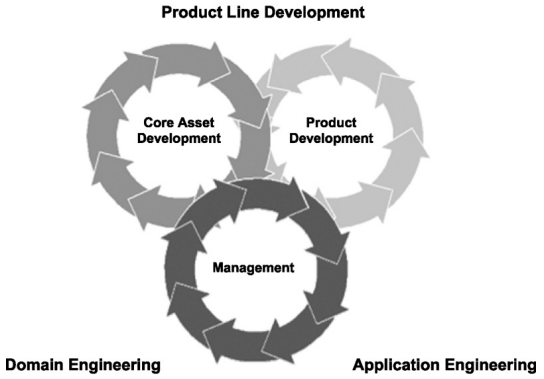

Fig. 1. The three essential activities involved in an SPL (source: http://www.sei.cmu.edu/plp/frame_report/PL.essential.act.htm).

of products from those core assets "*in a prescribed way*". These subtle additions to the definition are crucial for a successful SPL [4].

Besides providing a definition of an SPL, perhaps it is worthwhile to briefly mention what SPLs are not (for a more detailed discussion on this, see [3]). They are not about opportunistic fine-grained reuse (e.g. at the code level). Reuse in SPLs occurs at a strategic level which means reuse is planned, enabled and enforced. SPLs are also not about single-system development with reuse. A popular example of this is when an existing system is cloned and modified to suit new needs. SPLs are also more than multiple releases and versions of a product, as an SPL usually encompasses multiple products simultaneously.

As pointed out earlier, SPLs are a relatively new concept. However, an important body of work on this topic already exists, along with a growing community of practitioners. The next two sections describe two important sources of knowledge and experience related to SPLs that are particularly relevant in the context of our work.

3. The SEI's Product Line Practice Framework

The SEI Product Line Practice (PLP) Framework was first published in 2000. The PLP Framework is built around what its authors term the "three essential activities", namely Core Asset Development, Product Development, and Management. Core Asset Development is mainly concerned with defining the Product Line scope and producing the core assets from which the products will be built. Product Development consists in turning these core assets into products, and Management oversees both these activities at the project and organizational level. Fig. 1 illustrates the three essential activities and their relationships. It is worth noting that Core Asset Development is also often referred to as Domain Engineering, while product Development is often referred to as Application Engineering. It is also important to note that there is no clear starting or ending point in this figure. The activities are highly iterative in nature and where a Product Line actually

Table 1
The practice areas in the SEI's Product Line practices framework [4]

Software engineering practice areas	Technical management practice areas	Organizational management practice areas
1. Architecture Definition	1. Configuration Management	1. Building a Business Case
2. Architecture Evaluation	2. Data Collection, Metrics, and Tracking	2. Customer Interface Management
3. Component Development	3. Make/Buy/Mine/Commission Analysis	3. Developing an Acquisition Strategy
4. COTS Utilization	4. Process Definition	4. Funding
5. Mining Existing Assets	5. Scoping	5. Launching and Institutionalizing
6. Requirements Engineering	6. Technical Planning	6. Market Analysis
7. Software System Integration	7. Technical Risk Management	7. Operations
8. Testing	8. Tool Support	8. Organizational Planning
9. Understanding Relevant Domains		9. Organizational Risk Management
		10. Structuring the Organization
		11. Technology Forecasting
		12. Training

starts and ends will depend on the specific context of an organization. For instance, if a company is starting a new Product Line from scratch, perhaps the core assets will first be developed, and then products built from them. In the case where an organization already has a number of products, perhaps the core assets will actually start as a portion of (if not all) existing products or their constituting elements.

In its current state and based on publicly available documentation, the framework includes 29 practice areas grouped into three categories, namely practices related to Software Engineering, Technical Management and Organizational Management. Table 1 summarizes the Practice Areas in the framework.

In this framework, the categories for practice areas are based on the skills required to perform the activities of the practice areas. In other words, similar skills are required to perform the activities in Software Engineering Practice Areas, while similar skills (but different from the previous set) are required to perform activities related to the Technical Management Practice Areas. We will later see, when we present our model, that we chose a different approach to establish practice area categories. The SEI's PLP Framework also describes each practice area in some detail. For each Practice Area:

- aspects that are peculiar to Products Lines are discussed;
- application of the practice area to Core Asset Development and Product Development is described;
- specific practices are identified and described;
- risks associated to the practice area are identified;
- references to more detailed material are provided.

Everything discussed thus far in this section is available in the Web version of the framework [4]. The book version [3], however, includes additional material, including discussions on patterns, an assessment method for Product Line capability termed the Product Line Technical probe, and a few case studies. The Product Line Technical Probe (PLTP), which is briefly discussed in [12], is described as follows:

Table 2
The five VRAPS principles for software architecture [6]

Vision is the mapping of future value to architectural constraints as measured by how well the architecture's structures and goals are clear, compelling, congruent, and flexible.
Rhythm is the recurring, predictable exchange of work products within an architecture group and across their customers and suppliers.
Anticipation is the extent to which those who build and implement the architecture predict, validate and adapt the architecture to changing technology, competition, and customer needs.
Partnering is the extent to which architecture stakeholders maintain clear, cooperative roles and maximize the value they deliver and receive.
Simplification is the intelligent clarification and minimization of both architecture and the organizational environment in which it functions.

> *"The Product Line Technical Probe (PLTP) is a method for examining an organization's readiness to adopt or ability to succeed with a software Product Line approach. The PLTP utilizes a focused series of structured interviews of small peer groups within the organization, followed by data analysis. The PLTP utilizes the SEI's Framework for Software Product Line Practice as a reference model both in the data collection and in the data analysis. The results of the PLTP include a set of findings, which characterize an organization's strengths and challenges relative to its Product Line effort, and a set of recommendations. The findings can provide input to the SEI's Product Line Planning Workshop. The facilitated Product Line Planning Workshop helps to develop an action plan with the goal of making the organization more capable of achieving Product Line success to support identified business goals."*

At first glance, since our specific goal is to assess Product Line capability at our industrial sponsor's organization, the SEI's PLTP is exactly what we need. However, as pointed out in the previous citation, the PLTP is not in the public domain and anyone interested in having their organization assessed for Product Line capability must mandate the SEI and pay for their consulting services. The book associated to the PLP framework does have sample questions from the PLTP, but only for one of the 29 Practice Areas. As far as we can tell, these sample questions are not available on the Web.

4. The VRAPS model

Dikel et al. propose a model that concentrates on the intersection of software architecture (in general, not specifically Product Line architecture) and organization. The model is built around five core organizational principles, namely Vision, Rhythm, Anticipation, Partnering and Simplification, hence the name VRAPS model. The definitions of these five core principles are given in Table 2.

Dikel et al. also propose a framework that builds on and completes the VRAPS model by defining criteria, patterns and antipatterns, all of which enable concrete use of the core principles. Criteria are used for assessing a principle, e.g. to determine if and how well

Table 3
VRAPS principles and associated criteria [6]

Principle	Criteria
Vision	When vision is in place, 1. The architect's vision aligns with what his or her sponsors, users, and end customers are trying to accomplish 2. Practitioners trust and use the architecture 3. Tacit knowledge about architecture and components is visible and accessible to users
Rhythm	When rhythm is working, 1. Managers periodically reevaluate, synchronize, and adapt the architecture 2. Architecture users have a high level of confidence in the timing and content of architecture releases 3. Explicit activities are coordinated via rhythm
Anticipation	When anticipation is working, 1. Architecture capability is regularly enhanced to respond to • Anticipated risks and requirements of architecture customers and their customers, • Market-driving standards and evolving technology, and • Changes in strategic business directions 2. Technical and business risks and opportunities are evaluated through quick releases of review and development 3. Features, budgets, plans, or schedules are adapted when it is recognized that critical estimates or assumptions are incorrect
Partnering	When partnering is working, 1. The architect continually seeks to understand who the most critical stakeholders are, how they contribute value, and what they want 2. Clear, compelling agreements exist between stakeholders 3. Both policies and informal rules of social conduct enforce cooperation
Simplification	When simplification is working, 1. Developers continue to use the architecture over time, reducing overall cost and complexity 2. The architecture group clearly understands the essential minimal requirements and builds them into core elements that are shared across one or more applications 3. Long-term budget and action ensure that elements are removed from the core when • they are not shared, or add unnecessary complexity and • there is a clear business case

the principle is being practiced. The authors propose three criteria for each principle, as summarized in Table 3. Patterns in this context are basically things that should be done to solve problems in specific situations. On the other hand, antipatterns describe what not to do, or in some cases solutions that are applied in the wrong context. Numerous patterns and antipatterns are discussed in this text, and references to yet many others are also provided.

Although the term "Product Line" does not appear in the book's title, it appears at numerous occasions in the text and Product Line architecture case studies and examples are used to illustrate the proposed concepts. The same authors were previously involved in Product Line specific work [7], and this work seems to have influenced the VRAPS model substantially. Our model was also strongly influenced by the emphasis that Dikel et al. put on the importance of an articulated product vision, which they note as a necessary condition in organizations that successfully deployed architectural practices across Product Lines.

Dikel et al. also propose, along with the VRAPS model, an assessment method (it is actually termed *benchmarking* in this text) similar to the aforementioned Product Line Technical Probe. However, in the case of the VRAPS model, the assessment method is described in some detail. The actual process used to assess a number of organizations is described, some templates are provided and some actual findings resulting from assessments across four relatively large organizations are discussed. The templates provided in [6] are actually also available on a companion Web site [11].

5. The proposed model

Before discussing the model we propose, we first summarize the salient characteristics of the PLP Framework and the VRAPS model discussed in the previous sections. Both approaches have commonalities, as they both strongly emphasize concepts such as software architecture. Moreover, in both cases, non-technical issues are identified are a major concern for Product Line success.

There are also some differences between the two approaches. Both discuss an assessment method, which in our case is an important factor as assessment is the ultimate goal of our work, but the Product Line Technical Probe is only outlined and sample questions provided for one of the 29 practice areas. In the case of the VRAPS model however, an assessment process is described and actual templates used to conduct interviews during this process are provided. Our sponsor also found that concepts set forth in the VRAPS model were more concrete and perhaps easier to put into operation.

The model we propose in this section basically builds on the concepts from the SEI PLP framework and Dikel et al.'s VRAPS model. However, our model puts a stronger emphasis on system considerations, where the previous two mostly concentrate on software. In the case of our sponsor, co-development of hardware and software and extensive use of legacy systems is omnipresent. Hence, the core assets actually include both hardware and software. Moreover, this organization grew primarily by acquisition and some of the organizations acquired already had Product Lines implemented. In this context, one of the goals of our sponsor is to extend the concept of Product Line one step further and define a top-level Product Line for the entire organization, hence a system approach.

Our model is built in such a way that it remains compatible with the SEI CMMI [5] continuous representation and the ISO/IEC 15504 [9] standard, which the continuous CMMI representation is compatible with. Alignment with these standards is also motivated by our system focus as well as our sponsor's approach to Product Line practices as part of an engineering process improvement program.

Another driving factor in our model's design consists in focusing on Product Line-specific practices. We do not discuss practices that are common in a non-Product Line context.

Our model presently consists of 31 practice areas organized in five categories. We mentioned earlier that the PLP Framework categorizes practice areas based on skills required to perform the related activities. Instead of using skills to define our categories, we decided to group our practice areas based on activities to remain consistent with the CMMI and the approaches currently used in the software and system engineering capability areas. This choice is also motivated by the fact that our goal is to assess various organizations for Product Line capability. We therefore need to ask questions to the people involved in the Product Line for these assessments. We found it easier to ask questions based on activities, as they (ideally) have clear input and output artifacts, and it is relatively straightforward to define questions such as "Show me artifact XYZ and explain to me how it is used". We defined five categories of practice areas as follows:

- Product Line Management;
- Architecture Management;
- Requirements Management;
- Assets Development and Management;
- Product Synthesis and Support.

A quick look at our categories list reveals that they are more or less organized on a life-cycle approach (although no specific life-cycle is implied). This seemed a relatively natural way to group practice areas, and was also influenced by our desire to keep the proposed model aligned with current development methodologies, which we discuss in the next section.

A closer look at these practice areas should bring out that we are mainly dealing with operational processes. By operational we mean that these processes are part of the organizational fabric and are continuously active. This is quite different from project processes which are instantiated when a project is initiated, and are terminated when the project is finished. Development capability models such as the CMMI mostly deal with project-type processes, although they do include operational processes such as Process Management.

Since Product Lines can have quite a long life-cycle it is normal that their processes, and the organizational structure associated with them, are more permanent in nature, leading to a fundamental paradigm shift compared to a project-based approach. It is thus normal that comparing a Product Line framework and the CMMI as done in [10] will turn out some differences. On the other hand, we believe that a closer integration of Product Line practices and engineering capability models such as the CMMI is possible, which is one of the motivations of the work in progress presented in this paper.

The following subsections summarize each of our practice area categories and associated practice areas.

5.1. Product Line management

The practice areas in this category are mostly derived from the VRAPS model with some of our own additions. They are summarized in Table 4 and are mainly concerned with practices at the organizational level. These practices are operational in nature.

Table 4
Proposed model: Product Line management practice areas

Practice	Details	Purpose of the practice
Define Product Line Scope	Definition of the boundaries of the Product Line.	To define the scope of the Product Line, thus ensuring that its boundaries are well defined. These boundaries should be documented as feature ranges.
Define Product Line Vision	Vision, in marketing terms, of the objectives and evolution of the Product Line.	To ensure that there is a documented vision, in marketing terms, of the objectives of the Product Line and of its evolution during its expected life.
Perform Product Line Market Analysis	Documentation of the actual market of the Product Line. Definition of its potential market(s), given hypotheses such as features, pricing, competition, etc. Must include services and support if applicable.	To ensure that a marketing analysis of the Product Line has been performed and its result documented. This analysis should cover both actual and potential markets as per probable scenarios. This marketing analysis must then be maintained during the lifetime of the Product Line.
Perform Product Line Strategic Analysis	Analysis of marketing and technical strategies to fulfill the Vision.	To elaborate strategies for the evolution of the Product Line so that the vision is fulfilled. This implies the exploration of different products succession scenarios through the lifetime of the Product Line.
Manage Product Line Evolution	Management of the evolution of the Product Line.	To ensure that the architecture of the Product Line and its vision are always consistent during the lifetime of the Product Line.
Establish and Maintain a Product Line Business Model	Organization, Accountabilities and Funding Model to manage and evolve the Product Line.	To ensure that the economics of the Product Line is well understood, the management accountabilities of the Product Line well defined and the funding of the Product Line activities secured.
Manage Product Line Performance	Monitoring of performance (sales, profits, quality, . . .) of the Product Line. Corrective actions or improvement activities.	To monitor all aspects of the performance of the Product Line, as well as collate, process and communicate the collected data and ensure that action is taken when required. This includes corrective actions or improvement activities related to either the Product Line and its components or the processes associated with the Product Line.
Perform Product Line Communication and Support	Communication to stakeholders of Product Line vision, evolution plan, etc.	To communicate in an efficient and timely fashion to the different stakeholders of the Product Line (marketing, customers, designers, maintenance and support personnel, etc.) the information they require on the Product Line. This includes the vision, evolution plans, etc.

Table 5
Proposed model: Architecture management practice areas

Practice	Details	Purpose of the practice
Define an Architecture Vision	Architecture vision definition — derived from Product Line vision.	To ensure that the vision of the Product Line architecture is well defined. This vision should be documented as architecture features and quality attributes ranges. Full traceability between the Product Line vision and its architecture should be maintained.
Define and Maintain the Architecture Requirements	Elaboration of the detailed requirements for the Product Line architecture.	To elaborate and document the detailed requirements of the Product Line architecture. These requirements should be as technology independent as possible, unless there are constraints. These detailed requirements should have full traceability to the architecture vision as well as the Product Line requirements and the Product Line evolution strategy.
Define the Architecture	Design of the Product Line architecture.	To define the architecture of the Product Line. This includes its expected evolution through time. Full traceability to the architecture requirements should be maintained.
Simplify and Optimize the Architecture	Optimization and simplification of the Product Line architecture.	To simplify and optimize the Product Line architecture to ensure that requirements are met while total life-cycle costs are minimized and time-to-market is optimized.
Perform Architecture Risk Management	Includes technology management.	To proactively manage, on a Product Line life scale, the risks associated with the different elements of the Product Line architecture. This includes the technologies used, or planned to be used.
Define and Implement COTS/OSS Utilization	Optimization of Commercial-Off-The-Shelf (COTS) and Open-Source Software (OSS) utilization in the Product Line.	To optimize usage of COTS and OSS usage in the Product Line. The activities of this practice need to be done concurrently with those associated with simplification, optimization and risk management.
Perform Core Assets / Product Partitioning	Partitioning between core reusable architectural assets and components and final product assembly.	To identify the architectural elements in the Product Line that are reusable, either as components or as part of a platform.

5.2. Architecture management

The practice areas in this category are mostly derived from the VRAPS model with some of our own additions. They are summarized in Table 5 and are mainly concerned with the elaboration and management of the Product Line architecture. These practices are also principally operational in nature.

Table 6
Proposed model: Requirements management practice areas

Practice	Details	Purpose of the practice
Elicit Requirements	Elicit requirements from product end users and other stakeholders.	To elicit requirements for the Product Line from the end-users of the products and other stakeholders such as the marketing, sales, product design and support personnel.
Perform Requirements analysis and consolidation	Consolidation and analysis of stakeholder requirements.	To analyze and consolidate the elicited requirements.
Establish and Maintain Requirements specifications	Detailed specifications of the Product Line requirements. Allocation between core assets requirements and product synthesis instances.	To derive from the consolidated requirements the detailed Product Line requirements, and identify core assets requirements.
Perform Requirements verifications and validation	Verification and validation of Product Line requirements, with specific activities for the core assets.	To ensure the correctness of the Product Line requirements, both intrinsically and within the constraints of a Product Line environment.
Manage Requirements	Managing requirement changes through Product Line versions and releases.	To manage Product Line requirements as the Product Line develops and evolves.

5.3. Requirements management

The practice areas in this category are present in the PLP framework, termed as *Requirements Engineering*. They are summarized in Table 6 and essentially consist of an extension of systems and software engineering standard practices at the Product Line level. By extending these practices to the Product Line level, they also become operational.

5.4. Assets development and management

The practice areas in this category are mostly inspired from the PLP Framework. They are summarized in Table 7 and are mainly concerned with the management of the common assets that are used across the Product Line during its life-cycle. These practices are operational.

5.5. Product synthesis and support

The practice areas in this category are mostly inspired from the PLP Framework. They are summarized in Table 8 and are mainly concerned with the development and the support of the individual products that are part of the Product Line. The first three practices are project focused while the last two are more operational and would normally be done by a more permanent organization. Most of the practices in this area should be part of the Product Development process.

Table 7
Proposed model: Assets development and management practice areas

Practice	Details	Purpose of the practice
Establish and Maintain a Core Assets Repository	Repository for core assets and associated management practices: Configuration Management, Release Management, etc.	To establish and maintain a core assets repository that is accessible to all stakeholders involved in the design and in the maintenance of the Product Line.
Ensure Core Assets Integrity	Processes to ensure that core assets are stable before being available to the production teams.	To ensure that core assets are properly verified and validated before they are made available for integration in products.
Perform Core Assets Mining	How new core assets are generated through Product Synthesis activities.	To identify and package new core asset items from product synthesis activities. These new items, once their integrity has been validated, are then made available through the repository.
Establish and Maintain a Production Process	How products are produced from the core assets.	To specify and document the process used to generate products from the core assets while maintaining the architectural integrity of the Product Line.
Perform Release Management	How (1) the content of a release, and (2) the frequency of a release are determined.	To determine the release frequency of core assets elements, so that they are synchronized to product releases and maintenance requirements, and to perform the release of these elements.
Perform Core Asset Maintenance	Adaptive, corrective and perfective maintenance of core assets.	To perform the adaptive, corrective and perfective maintenance of core assets.

6. Integrating Product Line practices into existing development methodologies

One of the objectives of the proposed Product Line practices model is to enable its use in such a way that Product Line practices can be easily integrated into existing development methodologies. We believe that this would be rather straightforward with the proposed model's structure. For instance, if an organization has a RUP-like [8] development methodology in place, the following new disciplines would need to be added:

- Product Line Management;
- Architecture Management;
- Asset Development and Management;
- Product Line Requirements Management.

Based on the fact that the workflows proposed in RUP fall in one of two categories (Development or Support), we envision that the disciplines we propose in the above list would fall in a third category, specific to systems based on Product Lines, since they transcend individual projects and are essentially operational in nature. The Product Line Requirements Management discipline would essentially reuse most of the methods and

Table 8
Proposed model: Product synthesis and support practice areas

Practice	Details	Purpose of the practice
Perform Customer Interface Management	Management of customers for tenders, product customization and maintenance.	To perform in a systematic fashion customer interface for all products part of the Product Line for tenders, product customization and maintenance.
Perform System Synthesis	Product development in the context of a Product Line: includes development practices and process to decide when functionality should be in the core assets, from COTS or custom component.	To perform optimally the development of products part of the Product Line. Reuse of core assets elements and platforms is maximized and architectural integrity of the Product Line is maintained.
Perform Customer Engineering	Engineering, installation and testing of site specific product configurations.	To perform optimally the engineering, installation and testing of site specific product configurations.
Support the Product	Product support in the context of a Product Line: includes support practices and 2–3 tiers of help desk (customer facing, core assets & COTS/OSS).	Product support for all products part of the Product Line.
Perform Product Maintenance	Product maintenance in the context of a Product Line: includes maintenance practices and process to tie up with core asset maintenance.	To perform maintenance (corrective, adaptive and perfective) of the products part of the Product Line in an integrated fashion with core asset maintenance.

tools used by the project Requirements Management discipline of RUP. These methods and tools would simply be used in a different context.

For product synthesis, hooks will have to be added in the methodology development workflows to integrate them into a Product Line context. These hooks will take the form of modifications of some workflows, modified or new checklists, and modified artifacts. For instance, the Requirements workflow will have to be modified to, among other things, reflect that the product requirements must be expressed as reuse of a set of Product Line-wide requirements and a set of deltas [3].

7. Conclusions and future work

In this paper, we propose a Product Line engineering practices model that builds on and extends the concepts of existing frameworks and models, namely the SEI Product Line Practice Framework and Dikel et al.'s VRAPS model. The initial goal that motivated the design of this model is the capability to assess organizations for Product Line capability. We believe that our model offers a good complement to what is currently available in the public domain. It is compatible with published SEI documentation, and could be

used not only to perform assessments, but also to add Product Line practices to existing methodologies.

We intend to refine the model and elaborate on its practices, reusing as much as possible from public domain material from the SEI and complementing as required. We also intend to explore in more detail how to modify a RUP-like methodology to make it functional in a Product Line context. Finally, to further validate our work, we intend to perform on-site assessments with more industrial partners.

Acknowledgements

The authors wish to thank the anonymous reviewers who reviewed both the initial version of the paper presented at IWSSA 2004 and this one. We also wish to thank the people who attended our presentation IWSSA and the co-chairs of this Workshop, Narayanan Subramanian and Lawrence Chung. These people provided valuable comments and insights that allowed us to substantially improve this paper and the interaction also gave us some ideas for future research.

References

[1] J. Bosh, Maturity and evolution in software product lines: approaches, artifacts and organization, in: Second Software Product Line Conference, SPLC2, 19–22 August, San Diego, 2002.
[2] L. Brownsword, P. Clements, A case study in successful product line development, Technical Report CMU/SEI-96-TR-016, October 1996. Available: http://www.sei.cmu.edu/pub/documents/96.reports/pdf/tr016.96.pdf, visited 2004-Aug-04.
[3] P. Clements, L. Northrop, Software Product Lines: Practices and Patterns, Addison-Wesley, Boston, MA, 2002.
[4] P. Clements, L. Northrop, A Framework for Software Product Line Practice. Available: http://www.sei.cmu.edu/plp/framework.html, visited 2004-Aug-04.
[5] CMMI Product Team, Capability Maturity Model® Integration (CMMI^{SM}), Version 1.1, Carnegie Mellon University, Pittsburgh, PA, 2002, Available: http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr011.pdf, visited 2004-Aug-04.
[6] D.M. Dikel, D. Kane, J.R. Wilson, Software Architecture — Organizational Principles and Patterns, Prentice Hall PTR, Upper Saddle River, NJ, 2001.
[7] D. Dikel, D. Kane, S. Ornburn, W. Loftus, J. Wilson, Applying software product-line architecture, Computer 30 (8) (1997) 49–55.
[8] IBM, Rational Unified Process Overview, http://www-306.ibm.com/software/awdtools/rup/, visited 2004-Aug-04.
[9] International Organization for Standardization, Information technology — Software process assessment, ISO/IEC 15504:1998.
[10] L.G. Jones, A.L. Soule, Software process improvement and product line practice: CMMI and the framework for software product line practice, July 2002, Technical Note CMU/SEI-2002-TN-012. Available: http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tn012.pdf, visited 2004-Aug-04.
[11] D. Kane, D. Dikel, J. Wilson, Templates for Collecting “Best Practices” and Assessing Organizational Principles for Software Architecture, http://www.vraps.com/templates.jsp, visited 2004-Aug-01.
[12] Software Engineering Institute, Product Line Technical Probe, http://www.sei.cmu.edu/plp/pltp.html, visited 2004-Aug-01.
[13] Sundberg-Ferar, Product Line Definition, http://www.shapetomorrow.com/resources/p.html#productline, visited 2004-Jul-31.
[14] The Eli Whitney Museum, The Inventor: Whitney Changed the Face of the North, http://www.eliwhitney.org/inventor.htm#three, visited 2004-Jul-31.