

Product Line State of the Practice Report

Sholom Cohen

September 2002

Product Line Practice

Unlimited distribution subject to the copyright.

Technical Note
CMU/SEI-2002-TN-017

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

Copyright 2002 by Carnegie Mellon University.

NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this report is not intended in any way to infringe on the rights of the trademark holder.

Internal use. Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use. Requests for permission to reproduce this document or prepare derivative works of this document for external and commercial use should be addressed to the SEI Licensing Agent.

This work was created in the performance of Federal Government Contract Number F19628-00-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (<http://www.sei.cmu.edu/publications/pubweb.html>).

Contents

Executive Summary	v
Abstract	vii
1 The On-Board Driver-Monitoring Business Unit	1
1.1 Report Content	3
1.2 Report Structure	5
2 What Is Meant by a Product Line?	6
3 What Approaches Are Organizations Taking for Asset and Product Development?	9
3.1 Practices at Cummins, Inc.	12
3.2 Practices at Philips Medical	13
3.3 Tools Applied to Practices	14
3.4 Approach for On-Board Driver-Monitoring Systems	14
4 What Are the Benefits?	16
5 What Are the Challenges and Risks?	21
5.1 Success Criteria	21
5.2 Challenges to Successful Product Line Adoption	23
6 How to Proceed	26
6.1 Establish a Product Line Champion	26
6.2 Prepare for Change	26
6.3 Follow an Incremental Approach to Show Early Results	27
6.4 Establish Architecture as the Key Asset.....	28
6.5 Make Cost of Opting Out High.....	29
6.6 Research Topics.....	29
7 Summary	31
Appendix A Software Product Line Questionnaire	33

Appendix B	Results of the Software Product Line Questionnaire	45
Appendix C	Workshop Report.....	51
References.....		61

List of Tables

Table 1: Companies with Documented History in Software Product Line Development	6
Table 2: Product Line Definition for the On-Board Driver-Monitoring Unit	8
Table 3: Examples of Product Line Approaches and Practices	11
Table 4: Use of Commercial Tools in Product Line Practices	14
Table 5: Five-Year Development and Maintenance Costs of Monitoring Products	17
Table 6: Possible Savings Using a Product Line Approach.....	18
Table 7: Reported Benefits from a Product Line Approach	19
Table 8: Benefits of a Product Line Approach on a DoD Project	20
Table 9: Risks of Product Line Adoption	25
Table 10: Structuring the Organization for the Product Line Approach.....	27
Table 11: Workshop Attendees Making Presentations.....	52

Executive Summary

This report describes the results of an evaluation of the state of software product line practice in industry today. Based on a composite of product line experiences, the report builds a narrative description of a product line investigation. A fictitious manager, Anna Schumann, conducts the investigation to answer five basic questions:

- What is meant by product line development?
- What are the approaches that other organizations are taking?
- What is the business case (including costs, benefits, and timing) for following a product line approach?
- What are the potential challenges and risks?
- What are the first steps required to put product lines into practice in her organization?

The answers to these questions, as discovered by Schumann, form the content for this report. Managers wanting to discover the current state of product line development and the potential for software product lines within their organizations may well ask these same questions plus others. The product line approach to software development is an emerging technology. The Software Engineering Institute's (SEI's) *Framework for Software Product Line Practice* [Clements 00]¹ forms the basis for answers to these questions, but, within the framework structure, organizations are evolving new ideas and concepts to enhance their product line technology. This report explores current product line activity.

The investigation described in this report found product line practices in place throughout the software industry. A number of small, medium, and large organizations are moving to the product line approach from the traditional, isolated, product-by-product approach to software development. The market areas covered by these organizations span the commercial arena, from medical equipment to aircraft avionics, and from automotive products to financial analysis systems. Successful organizations report significant gains in productivity, software quality, and customer satisfaction by using the product line approach.

While there are barriers to product line adoption, these organizations report that the major barriers are not in the technical realm. Many specific technical hurdles remain that must be surmounted, but they are not considered "show stoppers." Practitioners consistently report barriers in management and at the organizational levels, ranging from a concentration on short-term goals to the lack of an adequate means of measuring developer performance.

¹ Version 4 of the framework is contained in *Software Product Lines: Practices and Patterns* [Clements 02].

Technically, the state of the practice is vibrant and burgeoning; on the nontechnical side, the state is healthy, but still in need of improvement.

This report presents accounts of product line experience from a variety of sources, including case studies and other technical reports. The *Framework for Software Product Line Practice* [Clements 00] and *Software Product Lines: Practices and Patterns* [Clements 02] provide much of the material for this report. The report blends information gathered from the literature with data captured from a questionnaire based on the product line framework. A product line workshop held at the International Conference on Software Reuse (ICSR), in April 2002, also provided material for the report. In presenting a state of the practice evaluation, this report reviews the impact of earlier work by the SEI and others, and it updates data from that work.

The product line questionnaire included both specific response and open-ended questions. The entire questionnaire is included in Appendix A, with a summary of results in Appendix B. The ICSR workshop brought together practitioners and researchers to consider responses gathered from the questionnaire respondents as well as factors in successful product line adoption. The results of both the questionnaire and workshop point out the interest and need for a more detailed and broadly targeted survey. Such a survey would delve further into particular product line issues and support statistical analysis according to organizational size, maturity, product line market area, and product line approaches.

Abstract

Software product lines represent a new and promising approach for fielding software systems. Companies that have launched software product line efforts report significant improvements in cost savings, quality, productivity, and time to market. Technically, they report a high degree of success in developing approaches that address the software engineering and technical management of their product lines. Organizationally, these companies report success as well as challenges that they must still overcome.

This technical note reports on the state of software product line practice in industry. The report uses a narrative approach, based on a composite of individual companies' experiences in implementing software product lines. The report blends a case study with the results of a product line questionnaire that was sent to organizations with meaningful product line experiences and with the results of a product line workshop held during the recent International Conference on Software Reuse.

1 The On-Board Driver-Monitoring Business Unit

Anna Schumann is a fictitious software development manager for a large automotive manufacturer. She oversees the development of systems for the on-board driver-monitoring business unit. This unit develops systems that monitor driver performance by checking the status of the vehicle and provide data on driving conditions and problems in various formats to the driver. The devices can also report on unsafe driving habits such as extreme acceleration, speeding, and erratic steering or braking to indicate drowsiness of the driver. Licensing organizations may use the systems for driver training and certification. In more advanced models, data are sent via wireless connections to the manufacturer's diagnostics center for early detection of anomalies and dispatch of possible corrective measures. The software department of Schumann's unit has a budget in the tens of millions of dollars.

In the past month, four different vehicle projects have come to her with the same problem—how do we maintain the connection between the on-board systems and the Web-enabled front end for our car-to-dealer maintenance system? After seeing this problem for the past month, Schumann decided to meet with her staff and come up with a common solution.

“Each of your systems has its own transaction and communication processing subsystem,” she told them. “Every time our front end changes, we go ahead and rewrite the on-board system four times, once for each of your vehicles.”

“Five times,” someone chimed in from the other side of the table. “Don't forget the SUV [sport utility vehicle] processing. They're scheduled for update next month.”

“That's what I mean,” Schumann said. “Why can't we have a single processing system, tailored to the needs of each individual system? Then, when change happens, we can adjust the baseline and release it to each system, and there's a single point of maintenance at lower cost and greater reliability.”

Schumann's staff voiced various objections:

- One system was acquired as a result of a corporate acquisition and had no common legacy of processing with the other systems.
- Two of the on-board systems interfaced with unique databases and evolved their processing independently, but following the same diagnostic rules.
- The fourth system had monitoring capabilities distributed among other functions.
- No one knew how the SUV handled monitoring and processing.

No one objected on principle to the concept of a common monitoring subsystem, but they had been down that path before, with shared routines, common object libraries, frameworks, and component-based solutions. They had experienced only marginal success. Schumann resolved to develop a business case to show that a common monitoring process, capable of tailoring to system specifics, could be built. It could be retrofitted during major upgrades to some of the legacy systems and would form the basis for new systems. But what approaches that had not already been tried and rejected could achieve these results?

Many software managers in business today face this situation, for example

- A unit manager has two legacy systems in dire need of upgrade. Both perform roughly the same functions for different sets of users, but maintenance on each system has become a nightmare. There is barely a budget for one upgrade, let alone two.
- A manufacturer produces a set of software products for other organizations that integrate the products as part of a larger system. Each product is developed as a new start, though they share much in common. How can they cut costs and get better delivery in terms of time to market?
- A contractor wins three awards to deliver systems to different organizations. The systems have much in common, but the contractor has no plans to capitalize on that commonality. However, because it can't meet the schedule for all three systems in parallel, the contractor finds that it must practice systematic methods that allow it to field all three systems by sharing software across the three development efforts.

Schumann and other software managers must address these issues to meet budgets, time-to-market, or delivery schedules. They have multiple systems with common requirements, each satisfying the specific needs of a particular market segment or mission. Yet the manager wants to achieve delivery of the systems without independent development of essentially the same software for each system. The managers may consider development from a common set of core software, but they need methods to develop the software for the systems in a prescribed way. This problem statement is really a classic definition of a *software product line* [Clements 02]. Within the product line, managers need ways to determine

- what products they will need to develop to meet future market demands
- what common software (or assets) to develop
- how to develop the assets
- how to develop the products
- how to manage the evolution of assets and products

In addition to these technical concerns, managers must create a business case to justify developing the software assets that will be used to develop systems in the product line. The managers must also identify potential challenges and risks as part of organizational planning and risk management.

To help answer her questions, Schumann must address four basic issues of product lines:

- What is meant by a product line?
- What product line approaches are organizations taking?
- What are the benefits?
- What are the challenges and risks?

She will look at case studies and other technical reports, and she will examine the results of an industry questionnaire to determine the state of product line practice. Armed with this information, she should be able to develop a plan for establishing a product line in the on-board driver-monitoring business unit. She will be able to justify the expenses for putting the plan in place through reduced development and maintenance costs and determine an organizational structure to put the plan into effect.

1.1 Report Content

This report takes the form of a narrative describing Schumann's product line investigation. It describes the results of an investigation into software product line experience—the state of the practice in industry today. Organizations considering a transition to the product line approach are the audience for this report, especially those at the management level who must make organizational decisions. By describing the path that Schumann takes to analyze product line experience, this report will help managers understand key product line issues and the state of the practice in addressing those issues.

The information presented in this report uses accounts of product line experience from a variety of sources, including case studies and technical reports. The Software Engineering Institute's (SEI's) *Framework for Software Product Line Practice* [Clements 00] (hence referred to as the framework) and *Software Product Lines: Practices and Patterns* [Clements 02] provide much of the foundation for this report. A study of reuse programs in Europe [Morisio 00] provides guidance in structuring this report. The report blends information gathered from Web- or print-based literature on software product lines with data captured from a questionnaire based on the framework. A product line workshop held at the International Conference on Software Reuse (ICSR) in April 2002 also provides material for the report. In presenting the state of the practice, this report looks at the impact of earlier case studies and reports, updating that material.

The questionnaire and workshop identified product line practices in place throughout the software industry. There are a growing number of small, medium, and large organizations moving to product line approaches from the traditional, isolated, product-by-product approach to software development. The organizations cover many market areas within the commercial arena, from medical equipment to aircraft avionics, and from automotive products to financial analysis systems. Organizations responding to the questionnaire

reported significant gains in productivity, software quality, and customer satisfaction by using the product line approach. While there are barriers to product line adoption, these organizations report that the major barriers are not technical. Many specific technical hurdles remain that must be surmounted, but the respondents do not consider them to be “show stoppers.” Practitioners consistently report barriers in management and at the organizational levels, ranging from a concentration on short-term goals to the lack of an adequate means of measuring developer performance. Technically, the state of the practice is vibrant and burgeoning; on the nontechnical side, the state is healthy, but still in need of improvement.

The product line questionnaire was circulated within the software product line community. Appendix A contains the questions from the questionnaire; Appendix B contains the numerical results. The framework [Clements 00] served as the basis for formulating questions. These questions asked for background information about the respondent’s organization and role in that organization and covered the terminology, the product line approaches in use, and the market areas of product lines at the respondent’s organization. Other questions focused on organizational structure, practices, and goals, and on the results of the product line effort.

The questionnaire gathered information regarding product line activities within industry. It was not intended to represent a scientific survey but rather to gather current information from organizations active in product line efforts. The population for the questionnaire was limited. It went primarily to participants in previous software product line meetings including the First Software Product Line Conference [Donohoe 00] and other conferences and workshops. Respondents were primarily from industry (90%), with 7% from government and 3% from academia, so the answers do reflect an industry perspective. However, the limited population and respondent return rate of about 20% restrict reporting only to raw response numbers. We cannot draw firm statistical conclusions; any correlation of results across the sample would be misleading. The results do, however, show meaningful trends within industry.

The ICSR workshop in April 2002 brought together practitioners and researchers to consider questionnaire responses as well as factors in successful product line adoption. Appendix C contains a summary of the presentations and discussions at the workshop. The results of both the questionnaire and workshop point out the interest and need for continued investigation and surveys to target specific areas within industry that are adopting software product line approaches.

1.2 Report Structure

This report covers the following topics:

- Section 2 – What is meant by a product line? This section presents basic definitions of the term *product line* and reports on terminology used by the questionnaire respondents. It also provides Schumann’s interpretation of the term, given her product line requirements.
- Section 3 – What approaches are organizations taking? This section discusses various organizational structures and practices used in industry. The questionnaire and the literature provide input to this discussion.
- Section 4 – What are the benefits? This section describes a business case approach to assess the financial benefits of a product line approach. It also summarizes benefits that the questionnaire respondents have achieved or hope to achieve through a product line approach.
- Section 5 – What are the challenges and risks? This section evaluates the possible barriers to successful product line adoption and reports on those cited in the questionnaire responses.
- Section 6 – How to proceed? This section lists five strategies to address challenges to successful adoption and also provides a brief research agenda.
- Section 7 – Summary. This section summarizes the findings of the investigation.
- Appendix A – Software product line questionnaire. This questionnaire includes both specific-response and open-ended questions.
- Appendix B – Questionnaire results
- Appendix C – Workshop report

2 What Is Meant by a Product Line?

A number of organizations are applying a product line approach to software systems. A search on the Web and in literature provided a number of companies in a variety of product line areas. Table 1 summarizes the results of that search.

Table 1: Companies with Documented History in Software Product Line Development

Company	Specifics
Area: Automotive, real-time embedded software	
Cummins, Inc.	Engine controllers
Robert Bosch, GMBH < http://www.bosch.de/k/en/start/index.htm >	Engine controllers
Danfuss Drives < http://link.springer.de/link/service/series/0558/bibs/1951/19510030.htm >	Electronic components
Area: Automotive, telematics	
Robert Bosch, GMBH < http://www.bosch.de/k/en/start/index.htm >	Driver information systems, sunroof/window controllers
Omnifleet® < http://www.omnifleet.com/product_info/productfamily.html >	Fleet maintenance software integration
Visteon Corporation < http://www.visteon.com/bluetooth/intro.html > < http://www.visteon.com/technology/automotive/telematics.shtml >	Electronics systems
Cummins, Inc. < http://www.cummins.com/na/pages/en/whoweare/index.cfm >	Electronic controls
Area: Traffic control systems	
Raytheon Company < http://www.raytheon.com/c3i/c3iproducs/c3ihtms/welcome.htm >	Highway control
Thales < http://www.airsysatm.thomson-csf.com/products/home.htm >	Air traffic control
ALCATELTransport Automation Systems, Spain < http://link.springer.de/link/service/series/0558/bibs/1951/19510053.htm >	Rail traffic control
Area: Aerospace, real-time software	
Boeing Company < http://www.isis.vanderbilt.edu/sdp/Papers/Papers.htm > (see: David Sharp, "Software Line Technologies for Military Systems")	Avionics systems
Honeywell < http://www.telelogic.com/download/userpresentation/fall_steiglitz_2.zip >	Space-based navigation systems
Rockwell < http://www.rockwellcollins.com/gs/flight2sys/ >	Cockpit management systems
Area: E-commerce	
Monetaire < http://www.monetaire.com/platform/ >	Financial planning
Market Maker Software AG < http://www.market-maker.de/03.Produkte/01.Software/MM/ >	Stock tracking
TenFold < http://www.10fold.com/ >	E-commerce solutions
ALLTEL Information Systems < http://www.alltel.co.uk/demonstration.cfm >	Banking and finance

Table 1: Companies with Documented History in Software Product Line Development (cont.)

Company	Specifics
Area: Other	
National Reconnaissance Office (Raytheon) < http://www.sei.cmu.edu/publications/documents/01.reports/01tr030.html >	Satellite command and control
Naval Undersea Warfare Center	Test range operations
U.S. Army Simulation, Training and Instrumentation Command < http://ctia.ideorlando.org/ >	Training systems
Lucent/Avaya Labs < http://www.research.avayalabs.com/department/softtech/ >	Various telecom
Philips Research < http://link.springer.de/link/service/series/0558/bibs/1951/19510199.htm > < http://link.springer.de/link/service/series/0558/papers/2290/22900390.pdf >	Medical equipment, consumer electronics
SaabTech Systems < http://www.sei.cmu.edu/pub/documents/96.reports/pdf/tr016.96.pdf >	Navy command and control (CelsiusTech)
Motorola, Inc. < http://www.motorola.com/MIMS/MSPG/Products/OEM/FLEXDecoding/index.html >	Paging solutions
Nokia Group < http://link.springer.de/link/service/series/0558/bibs/1951/19510107.htm >	Cell phones
Hewlett Packard Labs < http://www.hpl.hp.com/reuse/papers/index.htm >	Medical, bank systems
SchlumbergerSEMA < http://www.1.slb.com/smartcards/ >	Smart cards
Axis Communications AB < http://www.ipd.bth.se/msv/publications/299-055.pdf >	Networking equipment
Siemens Corporate Research < http://link.springer.de/link/service/series/0558/bibs/1951/19510019.htm >	Patient monitoring

In addition to varying in terms of market segment, the organizations responding to the questionnaire differ in their basic definition of the term *product line*. While most of the organizations do use the term *product line* to refer to their approach for fielding products, the industry questionnaire showed that the SEI definition has been adopted by 22% of the responding organizations. All the other definitions provided were local to the specific organization. Terms such as product family, application platform, and component-based development are also common throughout industry.

The SEI provides the following definition for product lines:

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 02].

Most of the responding companies agree that, regardless of the term they use, they are producing a set of related products. Some emphasize the specific approach they follow (e.g., mass customization [Krueger 01] or shared infrastructure [Batory 00]), whereas others emphasize the asset base in terms of an application platform or product library with components ready for assembly. The other feature of these definitions is the importance that organizations place on the use of a common architecture for their products. They also consider the degree of variability among products to be an important factor—some of the

organizations look for a minimum degree of variation among products, while others develop a product line that encompasses a wide degree of variation.

Products generated by these companies vary in size. Some of the organizations produce products that contain less than 100 KLOC (thousand lines of code). Others produce products with millions of lines of code. The questionnaire results are fairly uniform across this range. In addition to defining the products in the product line, organizations need to determine the extent and coverage of the assets. Again the terms vary here—some of the respondents use asset base, while others use framework or components. Some organizations develop assets that can cover almost an entire product, but the coverage varies between 50% and 90% for the responding organizations, meaning they must produce as much as 50% from scratch or uniquely for each product they field.

Nearly all the organizations responding to the questionnaire (74%) develop their products using specially built in-house assets. Others (48%) use assets derived from legacy systems. Many use both approaches. A few of the organizations commission an outside group to build assets for them, and others use commercial off-the-shelf (COTS) assets in the form of components. Some specialize in developing products that other organizations use as assets for their product lines and use their own assets for in-house tools or other products.

Schumann reviewed the literature and questionnaire results. She tried to characterize the products in her business unit in terms of the SEI’s product line definition, since it appeared to be the single most commonly used definition. The information captured in Table 2 describes the driver-monitoring product line according to that definition.

Table 2: Product Line Definition for the On-Board Driver-Monitoring Unit

Part of Product Line Definition	Examples
Set of software-intensive systems	On-board driver-monitoring systems
Sharing a common, managed set of features	Features for sampling, comparing to recommended limits, recording, reporting, and displays
(The systems must) satisfy specific needs of a particular market segment or mission (and be) developed from a common set of core assets	On-board monitoring and reporting systems Monitoring assets including common architecture and message formatting
In a prescribed way	Use of SEI’s framework practices [Clements 02] Use of PuLSE™ [IESE 02]

™ PuLSE is a trademark of Fraunhofer-Gesellschaft.

3 What Approaches Are Organizations Taking for Asset and Product Development?

The product line definition calls for applying prescribed methods for developing assets and products. Schumann's next challenge was to look at product line approaches and practices. She needed to decide which practices to apply if she chose to adopt a software product line approach. She also needed to determine what is involved in managing her organization as it makes the technical and organizational transition. The product line questionnaire gauged product line practices within organizations, and it helped Schumann determine the approaches that other companies are using.

The questionnaire responses to questions about product line practices define the scope of activity necessary to transition to a product line approach. This section includes the practices and overall approaches that the responding organizations have adopted to support product lines. The section also examines two case studies of organizational approaches and concludes with a discussion of the questionnaire responses on tool support for the requirements, modeling, and configuration management practices.

The framework identifies 29 essential product line practice areas, divided into software engineering, technical management, and organizational management categories. To sample approaches and practices, respondents were asked questions concerning 17 of the 29 practice areas, selected from among all 3 categories. The remainder of this section covers many of the software engineering and technical management practices used in asset and product development. The responses on other practice areas are discussed as follows:

- “Building a Business Case” and “Market Analysis” are summarized in Section 4.
- “Technical Risk Management” and “Organizational Risk Management” are summarized in Section 5.2.
- “Structuring the Organization” is summarized in Section 6.2.

According to the questionnaire results, nearly all the responding organizations apply the following practice areas:

- “Scoping” – Determine what is in the product line and what is out. Establish the assets to be used to produce product line members.
- “Understanding Relevant Domains” – Consider the areas of commonality within a product line and why products vary. Develop some representation of that understanding, captured as use cases or some other requirements representation. Alternatively, an architecture, tool, or generator may capture the variation.

- “Requirements Engineering” – Within areas of commonality, provide a requirements specification. This specification will apply to the assets built for the product line and to individual products.
- “Architecture Definition” – Use a common architecture for all products in the product line. This architecture will include the components to be used to build products and the connections between those components. The architecture also includes quality factors (such as performance and availability) that may lead to variations among products.
- “Component Development” – Apply well-established software development practices to produce code-level components. These strategies may include design patterns or other approaches from object technology.
- “Configuration Management” – Control the change process and evolution of both assets and products. For product lines, configuration management includes controlling evolution both in time (multiple releases of a single product) and in space (addition of new products in the product line).

The questionnaire also asked respondents about other software engineering and technical management practice areas including the following:

- Architecture Evaluation
- COTS Utilization
- Testing
- Mining Existing Assets
- Data Collection, Metrics, and Tracking
- Tool Support

The approaches that an organization uses in applying the practices will vary depending on a variety of factors:

- market areas that their products address
- size of the organization and maturity factors
- prior experience in the product line

To evaluate the effectiveness of these practices, participants in the ICSR workshop titled “Industrial Experience with Product Line Approaches” constructed a matrix that showed the various product line approaches and the product line practices most commonly adopted. This matrix, shown in Table 3, also includes information from product line case studies, such as those developed for Cummins [Clements 02] and Philips [America 00]. The contents of a column represent the practices applied by one organization following a specific approach, based on the practices in the framework. The matrix illustrates the variety of approaches that organizations have used for successful product line adoption. Because of the mix of participants at the ICSR workshop, the matrix shows a bias towards generator capabilities where assets or products are commissioned from an outside organization. This trend results from the biases within the workshop group and should not be taken as an industry norm.

Table 3: Examples of Product Line Approaches and Practices

Approach Practices	In-house development of features and assets (Philips Medical Systems, Cummins, Inc.)	Development of assets for other organizations (Salion)	Commissioning or purchasing assets (Programare)	Acquiring product line from outside vendor (Semantic Designs, Inc.)
Scoping	Organization recognizes need for producing multiple systems and determines boundaries for product line.	Organization recognizes need for supporting parts supply businesses and determines customer system capabilities that it must support.	Customers request support in database design. Organization recognizes and defines need for database schema definition.	Manufacturer recognizes need for a product line of robot controllers. Product line organization defines scope of application generator capability to build controllers.
Understanding Domains	Analyzes current, future, and legacy systems for common features and need for variation	Understanding evolves as needs are addressed for growing numbers of businesses.	Works with vendor to define database structure and requirements	Domain is well defined in terms of types of operations performed and primitives.
Requirements Engineering	Develop database of requirements or models to capture common requirements for multiple systems.	Uses GEARS tool from Big Lever to support mass customization	Vendor defines database requirements, SQL, and application generator to organization.	Requirements for members of product line are captured in generator.
Architecture Definition	Defined common architecture with specific points where variation may take place	Evolved as a result of developing multiple systems; now maintaining common architecture asset	Defined by database generator capability	Controller architecture is defined by generator.
Component Development	Develop common components with variation points to accommodate variety of capabilities or specialization of common capabilities.	All common components developed by organization; unique components may be developed by organizations or by businesses if they acquire assets.	Component design is captured in generator.	Component design is captured in generator.
Configuration Management	Internal management of both assets and configuration of each product in the product line	Organization hosts and maintains system for businesses or delivers full capability to customer businesses.	Vendor maintains control of tool; organization controls specific database applications.	Generator can reconstruct any instance.

Schumann considered the approaches taken by Cummins, Inc. and Philips as most closely related to the goals she hoped to achieve. The relationship of their work to that of Schumann's business unit makes them a reliable comparison.

3.1 Practices at Cummins, Inc.

Cummins, Inc. is a major producer of diesel engines for vehicles ranging from small trucks to large earth-moving equipment. Engines are also used for electrical power generation, maritime power plants, and various types of machinery such as drilling equipment, locomotives, millers, and winches.

Over a period of five years in the early 1990s, Cummins transitioned from seeing itself as a company producing iron (characterized as the "piston-head" mentality) to an organization producing software (the "digit-head" mentality). Cummins products contained little or no software until the late 1980s, but by 2000 they included software systems for engine control, each containing 100-200 KLOC. Until the mid-90s, each system requiring software was developed uniquely and from scratch. With 12 engine families and 8 new fuel systems and hardware controllers, this silo or stovepipe approach would eventually use all the company's resources. Cummins managers determined that they could no longer afford to concentrate software development efforts on individual products [Dager 00].

The approach adopted by Cummins followed the application of key product line practices. Cummins decided to build a set of assets that could be shared among the engine families, yet would be tailorable to the needs of specific systems. They captured their understanding of relevant engine controller domains, developed a common architecture definition, built components to support engine controller implementation, and put in place a corporate-wide policy that required business units to adopt a common development approach.

This approach decreased the "time to first engine start from 250 person months to just a few person months" [Clements 02, pg. 432]. Cummins develops 75% of new projects from existing assets. Quality has improved dramatically and projects are consistently on schedule. Cummins has seen an increase in customer satisfaction, and developers are more flexible and able to support new projects because their knowledge is highly transferable. The product line approach also supported Cummins' move in new business areas such as the industrial engine market. The process has been embraced to the extent that individual business units are motivated to create their own product lines, a trend that Cummins resists in order to maintain uniformity across the organization.

3.2 Practices at Philips Medical

Philips Medical supports a product line of medical imaging systems [America 00]. These systems are characterized by

- maturity and complexity
- life criticality
- small numbers of systems
- collaboration across many domains including hardware design, information technology (IT) infrastructure for integration into hospital systems, medical application experts, and mechanical designers

Philips has developed this product line consolidating the work of over 100 developers across several departments. These departments previously developed products in isolation, with little commonality in underlying concepts. These projects were developed over a period of several years. Their complexity required a durable architecture that could survive changes over their lifetime of 10 years or more.

Philips has formalized several practices for institutionalizing the product line approach, including the development of various requirements and object models. Their models must be independent of any single-system implementation to support their use across the product line. Philips uses the term *variation point* to characterize the specific factors that will vary from one system in the product line to another. A variation point also defines a high-level concept and specializes that concept for individual instances. Variation points may exist for many devices in a specific system; while one system may have none, others may have three or four of the devices. There may also be attributes common to the product line that take on different values for each system.

Philips uses quality factors to characterize the architecture of its product line [Pronk 00]. Specifically, the architecture must be able to

- support integration of new functions and peripherals
- provide extensions of the system without a requirement for system-wide upgrades
- enable efficient testing that does not require the testing of all possible configurations of functions and peripherals

Philips applies a component-based approach where components are tested and replaced in isolation.

Philips has had limited experience with this approach. This approach appears to support the isolation of updates that are required when peripherals are changed. Philips is confident that the approach works in mature areas with well-defined requirements and relatively slow evolution. The approach, which may not be appropriate in market areas with rapid changes

due to new technology, supports variation in functions and capabilities for more powerful hardware and peripherals.

3.3 Tools Applied to Practices

Software tools play a role in most of the product line practice areas. The questionnaire examined tool support in two specific practice areas (“Requirements Engineering” and “Configuration Management”) and also asked about modeling and design that spans several practice areas. Most of the responding organizations are using vendor-supplied tools to support their development. The remainders use some combination of vendor or homegrown tools. Table 4 provides the percentage of respondent organizations using specific tools.

Table 4: Use of Commercial Tools in Product Line Practices

Product Line Practice	Percent Using Tool
Requirements Engineering	Requisite Pro [®] 26% Doors [®] 19% Slate 3% Others or homegrown 50%
Modeling or Design	Rational [®] Rose [®] 55% Rhapsody [®] 7% Artisan Real Time Studio [®] 3% Others or homegrown 39%
Configuration Management	Clear Case [®] 48% CCC 3% CM Synergy 3% Others or homegrown 45%

The questionnaire also showed that the respondents recognize the need to improve these tools for use with product lines. Only 30% expressed satisfaction with their chosen requirements engineering tools, and 65% reported the need for improvements in Rational Rose.

3.4 Approach for On-Board Driver-Monitoring Systems

Schumann examined these case studies and concluded that a product line approach has benefited other organizations like her own. Her product line definition and scope (Section 2) could be supported in a systematic fashion by specific product line practices. The questionnaire respondents also gave her confidence that a product line approach could succeed, and she concluded that a product line approach would be appropriate for her

business unit. With five systems already fielded, she felt confident, given the Philips results. Like the medical systems, driver-monitoring systems require interfacing to numerous peripheral devices across a bus. These devices vary in numbers and capabilities based on the category of vehicle, but a common architecture should accommodate the frequent changes in these devices and the addition of new peripherals.

Schumann wants to develop an architecture that can survive several generations of these devices so that the same system can be used for multiple model years. In this way, on-board monitoring has similar architecture requirements to those of a medical product line. Schumann was also encouraged by Cummins' results that show the ability of an organization in the automotive industry to produce a wide range of products built on a common software architecture. Schumann sees products in her product line varying in terms of device interfaces and in use (i.e., in various applications such as driver monitoring, automotive performance, routine maintenance, and special maintenance). The devices can also report unsafe driving habits such as extreme acceleration, speeding, and erratic steering or braking; in addition, the devices may be used for driver training and certification.

4 What Are the Benefits?

The benefits that Cummins, Philips, and other organizations have realized with product line approaches are impressive. The questionnaire responses show that nearly all organizations expect to benefit in increased productivity and lower development costs by using a product line approach. Many have realized these benefits. But Schumann wondered how to measure the expected results in her business unit in terms of reduced costs. If we look at Schumann's situation, she has the following needs:

- a real-time monitoring architecture that can be tailored for specific applications
- a need to interface through wireless connections to servers at dealer, repair, and manufacturer sites
- a Web-enabled front end to the server to provide driver information at the above sites and to the driver when not in the car
- reduced maintenance costs as the Web infrastructure is modified
- a desire to achieve the 75% level of reuse achieved by Cummins in its products
 - a market of at least five systems today—some undergoing major upgrades and some number of new systems in the future

Given this context, building the cost data for the business case may not be easy, but these points helped direct Schumann.

Schumann used an approach based on SEI experience [Clements 02, Section 6.1].² The business case approach, called the A-B-C method, looks at three factors in measuring cost impacts:

- applications – number of applications to be built over a given time period
- benefits – projected cost savings or other return
- costs – actual costs incurred in using assets

Schumann's business unit has five initial systems and assumes the production of three new systems per year. These systems will be derived from assets at a rate of 75%, meaning there will be 25% product-specific software in each system. Current systems containing approximately 125,000 delivered source instructions cost approximately \$12 million to develop. Over their expected life of 8 years, they accrue as much as \$3 million per year in maintenance costs. The business unit assumes that assets for these products will be approximately 100,000 source instructions also costing \$12 million. Schumann bases her

² This information will also be documented in a future SEI report by Sholom Cohen to be titled *Developing a Business Case: Case Study*.

numbers on historic development costs within the business unit with some increased costs of fielding assets for systematic reuse.

Table 5 shows the expected costs, based on historical data, for development and maintenance over a five-year period. (Maintenance appears as “Number of Years in Maintenance” in the table, with maintenance costs beginning after the first year.) The redesign of the existing five systems occurs in the first two years, and then there are three new systems in the following year. These data are developed assuming the business unit does not follow a product line approach.

Table 5: Five-Year Development and Maintenance Costs of Monitoring Products

Program Name	Lines of Code	Development Cost (in millions)	Maintenance Cost per Year (in millions)	Number of Years in Maintenance	Total Cost (including Development and Maintenance) (in millions)
Small vehicle	120000	12.0	3.0	4.00	24.00
Large vehicle	140000	14.0	3.5	4.00	28.00
Small truck	126000	12.6	3.15	3.00	22.05
Large truck	150000	15.0	3.75	3.00	26.25
SUV	150000	15.0	3.75	3.00	26.25
New 1	130000	13.0	3.25	2.00	19.50
New 2	130000	13.0	3.25	2.00	19.50
New 3	130000	13.0	3.25	2.00	19.50
Assets	100000	12.0	3.00	4.00	24.00

Using the values from Table 5, the costs of developing these systems are calculated based on systematic reuse of the assets under a product line approach, as shown in Table 6. The cumulative costs include the costs of the individual products and assets, and their maintenance costs. The 25% cost of reuse covers the costs of understanding the assets and tailoring them for use in each system. This cost also includes changing the development processes within the business unit.

Table 6: Possible Savings Using a Product Line Approach

Program Name	Cumulated Cost Without Reuse (in millions)	Product Line Savings (50% from assets, 25% cost)	Cumulated Savings with Product Lines	Cumulated Product Line Costs	Cumulated Savings over 5 years
Small vehicle	24.00	9.00	9.00	12.00	-3.00
Large vehicle	52.00	10.50	19.50	12.00	7.50
Small truck	74.05	8.27	27.77	15.00	12.77
Large truck	100.30	9.84	37.61	15.00	22.61
SUV	126.55	9.84	47.46	15.00	32.46
New 1	146.05	7.31	54.77	18.00	36.77
New 2	165.55	7.31	62.08	18.00	44.08
New 3	185.05	7.31	69.39	18.00	51.39

Schumann determined that these results justified the investment. Starting with the \$12 million project to develop assets, the cumulated savings would be over \$50 million in 5 years. Schumann based this savings estimate on several relatively conservative assumptions:

- Schumann assumed that assets would provide 50% of the total software for each system in the product line. If a product costs \$10 million built totally from scratch, with product line reuse, that cost would be cut in half. The program goal for asset use was 75% of each product, so 50% seemed to be a safe assumption.
- Potential savings would be reduced by 25% to account for a reuse cost. The reuse cost means that the savings would be reduced by one quarter to allow for learning how to use the assets, building tools, and making overall adjustments to the product line culture. Using the example of \$10 million, the actual savings realized would be \$5 million minus \$1.25 million (the cost of reusing the assets). Again, she chose 25% because it seemed like the most pessimistic cost of reuse from various product line articles.

Schumann planned to take these figures to her finance department to build in a more precise business justification. Making the case precise would include taking into account discount rates and other economic measures. At this first level, however, the \$50 million in savings during the first 5 years did not seem like a bad return. Schumann also expects other results; like Cummins, her business unit expects to realize high quality in the software products, greater transferability of development staff, and the ability to expand its market for on-board systems. The savings may be plowed back into corporate research to identify and begin production of new products, the savings may be reflected as corporate profit, or the savings may be passed on to customers in the form of lower costs for an advantage over the business unit's competitors.

While there are obstacles to overcome (see Section 5), all the companies that responded to the product line questionnaire reported some of the benefits shown in Table 7 [Clements 02, Chapter 2].

Table 7: Reported Benefits from a Product Line Approach

Benefit	Percent Reporting
Improved quality	52%
Reduced costs	45%
Improved productivity	39%
Improved ability to meet customization demands	39%
Reduced time to market	30%

A project recently completed for the U.S. Department of Defense (DoD), the Control Channel Toolkit (CCT), summarized measurable and intangible benefits [Clements 01]. The DoD recognized the results shown in Table 8 in the initial use of an asset base on an actual operational system for ground-based satellite command and control. While benefits will vary based on the particular characteristics of a product line and the development organization, the questionnaire respondents indicated that these benefits are generally achievable. Other, less tangible benefits reported by CCT project staff included significantly lower staff attrition, general satisfaction among developers with the CCT approach, praise from the asset users for the selection of and support for the variation points, and greater professional satisfaction with the product line approach. Developers thought that the pedestrian tasks had already been done and the focus was on the interesting, mission-specific capabilities. The risks of system failure were felt to be fewer, and the pressure on the program was substantially lower.

Table 8: *Benefits of a Product Line Approach on a DoD Project*

Factor	Benefits
Quality	One-tenth the typical number of discrepancy (defect) reports was submitted for a system of this type. The problems identified were all local ones, with localized fixes, having no ripple effects and no effect on the architecture.
Performance	Use of the CCT improved performance over results predicted without it. In identified places where reuse may lead to timing problems, CCT variation points can be exercised to apply mechanisms that circumvent CCT software and apply faster algorithms.
Integration time	Incremental builds were completed in weeks rather than months, as was the case for the non-CCT portion. This approach is a direct carryover from the incremental approach to development.
Code volume	The number of design objects for subsystems using CCT is lower than planned by 34% to 88% with a similar reduction in actual source-code size. Total SLOC (source lines of code) developed by Spacecraft Command and Control (C2) are 76% less than planned.
Productivity	<p>Smaller development staff required (15 versus 100 for other similar systems)</p> <p>Overall costs cut by 50%</p> <p>Overall schedule cut by 50%</p> <p>Documented flexibility in meeting requests for modifications by the Spacecraft C2 System customer</p> <p>CCT treated like a COTS product (Initial training was required; then development proceeded on the basis of domain specification, interface definitions, and reuse guide.)</p>

5 What Are the Challenges and Risks?

From her study, Schumann could see that successful product line adoption correlates to a number of success factors. Some are technical, such as deep domain expertise, but most often nontechnical factors, such as an early adopter mentality, contribute to long-term success. Organizations that are succeeding in product line development exhibit one or more of these factors. However, no technological innovation is without risks. Most of the companies responding to the questionnaire identified reluctance of their organizations to adopt a new technology. This reluctance applied almost equally to developers (32%), management (36%), and organization-wide staff (52%). Other organizations reported difficulties relating to the large initial investment and the significant lead time necessary to realize benefits, while some organizations do not believe that such an investment or lead time is required. This section presents challenges to achieving success in product lines as well as the criteria most often observed in companies that have succeeded.

5.1 Success Criteria

How does an organization define success? For Schumann, success includes achieving lower costs in fielding products, adding new features to products, attaining more rapid time to market, and expanding the set of products she can deliver. The benefits listed in the previous section cover the range of goals that organizations set for achieving success in adopting a product line approach. Organizations that have successfully developed product lines generally share a number of common characteristics. While success can come without all these criteria being met, success generally has come to those organizations that exhibit most of the following characteristics to some degree [Clements 02, Section 12.3]:

- *deep domain expertise*: Organizations must have significant experience in developing products before adopting a product line approach. This experience provides knowledge of commonalities within the product line domains that are likely to yield significant reuse possibilities. Often, organizations migrate to a product line, directly capitalizing on the experience as they field new systems and capturing the knowledge in an architecture or requirements database. Organizations such as Cummins and Philips clearly have the rich legacy of successful product development and domain understanding necessary to field a product line.
- *process discipline*: Developers in product line organizations must cooperate in the development of assets and products. The processes they use must be applied consistently both within asset creation and product development. Some organizations choose to concentrate asset development into an independent organization, while others disperse asset creation among product groups [Bosch 01]. The product line questionnaire responses revealed about an even split between these two approaches: 42% within the

product group and 39% in independent asset groups. In either case, the groups must work collaboratively and be able to share assets, tools, processes, and products.

- *architectural excellence*: Product lines are generally built around a product line architecture that may either be built into a generator or be a separate asset. Success in the product line requires an architecture that can support variation discovered in understanding product line domains. The successful organizations use the architecture not only as a blueprint for building products, but also as a mechanism for organizing software development, of both assets and products. Several questionnaire respondents included comments about the extent to which architecture influences their overall product line strategy. The architecture requires continuous improvement as products are fielded, so it may also require reevaluation. Successful organizations must plan for the sustainment of the product assets, especially architecture, and allocate resources to that task.
- *strong champion*: Successful organizations must have one or more individuals who maintain the product line vision through the development of assets and initial product line products. Such a champion must understand the product line concepts and have the authority to make change happen. Ron Temple clearly represented this individual at Cummins [Clements 02].
- *management commitment*: A product line will not survive lacking strong management commitment. The product line commitment may run against the demands of individual product deliveries, but management must align those demands without sacrificing the product line concept. Successful product line organizations often have put scarce resources into asset development without seeing tangible returns for two years. On the other hand, questionnaire respondents reported that management used product line terminology but did not grasp the strategic nature of product line adoption or the commitment required to achieve success. Product line adoption requires serious management buy-in. Lacking this buy-in, however, the product line may never see its initial products, or it will soon see products that deviate from the product line architecture and other assets.
- *change of mindset*: The organization must shift its thinking from product-by-product development to the product line perspective. Cummins has emphasized the importance of this change. Developers, marketers, and management must realize that they no longer create, market, and sell individual, stand-alone products. Rather, they are creating, marketing, and selling the product line; individual products are an outgrowth or even a side effect of building and maintaining the product line. Everyone in the organization must recognize the contribution that each makes to the product line and the benefits that product line efforts bring to the organization.

Case studies reveal two final criteria, not only for success but also as strong motivators:

- *desperation*: The case studies for both CelsiusTech (now Saab Systems) [Brownsword 96] and Cummins [Dager 00, Clements 02] show that the necessity of adopting product line approaches is a strong motivator for achieving success. In both cases, meeting key business goals necessitated adoption of an approach that could capitalize on the commonality among a number of systems that were being fielded simultaneously. Once they adopted product line approaches, both organizations reported that these approaches were essential to their continued success.
- *new companies*: A new organization can often succeed where inertia will drag down the innovative elements within an older company. Salion, a company described briefly in

Appendix C, organized itself to support automotive and other manufacturing organizations. Its business model was to offer manufacturers an application service, tailored to the manufacturer's factory operations, that supports parts procurement and delivery. From the start, Salion built its business around the ability to provide a core capability with components tailorable to each user's needs. An established company would, no doubt, find it difficult to shift from individual products to a product line mentality, but as a new organization with an innovative mindset, Salion found it natural to take on the product line approach. Companies entering new markets or developing new ways of delivering their products, such as Web-based delivery, may also find product lines a natural way of doing business.

Schumann evaluated her situation against these criteria. Her situation wasn't desperate yet. However, she could see a time when her budget could not continue to support so many independent developments. It would be better to fix things now rather than wait until they became critical. Schumann's group contained significant expertise in several of the monitoring areas, yet much of the expertise was located in individual products. Experts in several key areas were not in her business unit at all, but belonged to other technology groups. She would face a challenge in bringing together the appropriate levels of expertise.

Schumann foresaw a problem in managing the software process as well. The degree of independence among the development groups led to different development approaches, not unlike those at Cummins before it adopted a product line approach. In addition, several of the development groups came from organizations that Schumann's company had acquired over the years. These groups still maintained their own development approaches. A move to a product line would require careful introduction of new sets of process, possibly in an incremental fashion as new products fall under the product line.

Schumann had considerable authority to manage her organization and was developing a vision of how product lines would provide benefits. However, a significant change would require approval from higher level management. She would have to obtain this approval through the business case that she was developing and the benefits that she would quantify for senior management.

5.2 Challenges to Successful Product Line Adoption

While many technical issues pose risks, questionnaire respondents listed failure to achieve buy-in throughout an organization as the risk that most often interferes with product line adoption. Schumann recognized this risk—the driver-monitoring product line would simply not achieve its goals if the organization did not accept product lines as a way of doing business. The success criteria included the adoption of specific technical practices, especially those that are required to achieve architectural excellence. While appropriate guidance exists in these key technical practice areas, if they are not adopted and applied across the product line organization, asset developers will create assets that are never used, or product developers will assume design approaches for a new product that are a mismatch with the assets. The product line organization may either fail to institute beneficial practices or institute practices that are not appropriate to the particular organizational situation.

Schumann's study found the following as some of the more common risks that organizations must address [Clements 02, Section 6.5.5]:

- *no identified product line champion*: Without a champion given sufficient management authority, product lines cannot succeed. All the documented product line success stories had such a champion, at least through the adoption phase. The champion is the individual who communicates the vision to management and developers, maintains the vision throughout the adoption, keeps developers on track when difficulties arise, and maintains a high level of enthusiasm. Where no single individual has these qualities and roles (i.e., there is no product line champion), adoption generally does not happen. (This is also the case for technology areas other than software product lines.)
- *approach mismatch*: Software product lines should be a strategy that meets specific business goals. While they vary among organizations, the product line goals are always predicated on exploiting systematic reuse among products. If the products being developed do not have sufficient commonality to warrant a product line approach, any launching effort will fail due to a lack of tangible results. In other words, a product line approach to software is a means to achieve an end and is not the end itself.
- *inadequate management commitment*: The product line champion must convince management of the viability of a product line approach to obtain its commitment for proper funding, staffing, or allocation of other resources. While pilot projects are often used to obtain such commitment, they must focus on applications in the organization's main line of business and generate real and immediate results. The champion must be able to convince management to back efforts in high-visibility units with some of the key personnel from those units. Without these key people, the pilot is not likely to succeed. The champion must obtain commitment at the proper levels and within certain time frames for successful adoption. Even a well-executed pilot in an obscure business unit or an isolated domain may do little to advance the adoption effort. Management is less likely to invest in a business unit that does not perform well or is not considered critical to the company's success. These efforts must focus on developing the commitment at high organizational levels that may have more strategic perspectives on the business situation and future needs.
- *insufficient staff commitment*: The product line champion must also convince the technical staff of the viability of a product line approach. Without the commitment of technical staff, they will likely torpedo the effort. A major aspect of any product line launch should focus on educating and achieving the staff's buy-in. As a mitigation step, the organization must involve technical staff in the definition of plans and processes.
- *insufficient bounding*: A frequent complaint among organizations seeking to adopt product lines is the long lead time and costly effort required to attain measurable benefits. The plan for an initial product line capability must be timed to realize and justify the investment. Some organizations limit the initial product line scope to create only high-payoff assets and incrementally expand the scope of coverage. Others limit the market focus to address a small number of products in the initial product and concentrate on delivering valuable benefits within the needed time frame. Over time, they will bring more products under the product line approach.
- *inappropriate levels of interaction*: The product line approach must span organizational boundaries. Management, engineering, and marketing staff must work collaboratively in product line adoption and application. Inadequate collaboration among these groups may produce unattainable goals, marketing of products that cannot be developed within the

product line, or assets and production plans that are insufficient to address real needs. Marketers must be aware of the product line's ability to support the needs of a new customer, and management must set goals within the adopted approach to limit unsupported deviations from the product line. Still, product line approaches should ensure that the product line can evolve to accommodate future products.

- *inappropriate standardization*: Organizations often err in assuming that adoption of standards will automatically support product lines. Unfortunately, if institutionalization occurs too quickly, inappropriate or obsolescent standards may be instituted, and innovation may be terminated prematurely. On the other hand, if standardization opportunities are missed or delayed, there may be redundant or unnecessarily divergent efforts. Too much of too many products will become system unique, and the organization will miss opportunities for optimally effective practices.
- *insufficient tailoring*: Just as architecture or components require some degree of tailoring across a product line, the organization must often tailor practices across development groups or products. Applied inappropriately, practices within the organization may result in suboptimal performance or unsupported deviations from the preferred practices.
- *failure to evolve the approach*: If the approach to software product lines is not continuously improved over time, practices will likely become ineffective, and unsupported deviations will arise out of necessity.
- *ineffective dissemination*: The product line champion must take responsibility for developing and distributing the appropriate level and type of documentation, adequate training, and effective product line support. These products are essential for a successful product line launch. The launch will not occur in the planned form or produce the desired outcomes in the required time frame if there is inadequate preparation.

No single organization will face all these risks, but results of the questionnaire showed that all the responding organizations recognized at least three or four of the problems listed in Table 9. Other risks identified in the questionnaire included

- skeptical or uninformed customers
- impulse within the organization to fall back on traditional, one-at-a-time methods
- improper timing of product line introduction (e.g., midst of critical product release)

Table 9: Risks of Product Line Adoption

Problem	Percent Responding
Organizational resistance	52%
Management resistance	36%
Developer resistance	32%
Concerns over large investment	45%
Lack of properly trained staff	29%
Inability to measure impact	19%
Concerns of long lead time	18%

6 How to Proceed

Next, Schumann had to decide how to proceed. What must an organization do to overcome resistance while addressing the risks and achieving the benefits listed in the previous section? Based on the success criteria cited in Section 5.1, Schumann determined that successful organizations must address several key objectives:

- Have a strong product line champion to advocate the product line cause.
- Prepare management and development personnel for the changes.
- Follow an incremental development approach that can show concrete results early.
- Make architecture the key asset and use domain understanding and requirements engineering as the basis for generating the architecture.
- Make the cost of opting out of the product line too high for the individual projects.
- Identify applied research areas, both in development and process areas, to close the gap between theory and practice.

6.1 Establish a Product Line Champion

Schumann saw herself as the product line champion, leading the strategic planning and long-term focus on outcome and payback. Adoption across the organization requires collaboration among functions as diverse as management, engineering, marketing, sales, support, and administration. Seeing an organization through the adoption process and initial operational state requires a leader who is respected by higher level management, development organizations, and customers. The leader must have technical knowledge to understand the architectural and production issues and be an excellent planner and manager with skills in financial analysis and strategic planning. These characteristics are those of the product line champion. The champion must also be committed, along with management, to a multi-year effort from kickoff to routine production in the product line.

6.2 Prepare for Change

Many organizations are unprepared for the changes required for product line adoption. While technically able and committed to minimize costs and time to market while maximizing quality, they have been optimized to deliver single systems. The training, compensation, and promotion criteria are all geared towards rewarding the success of managers and developers of individual projects. The notion of sharing resources across an organization is foreign and regarded as suspect. The product line champion together with the technical staff must develop

adequate training and processes to support the product line approach and ensure their dissemination.

The organization must also consider organizational structures. Table 10 summarizes the questionnaire results for structuring the organization for product lines:

Table 10: Structuring the Organization for the Product Line Approach

Organizational Structure	Percent Responding
Separate core asset development unit supplying assets to product groups	39%
Core asset development as part of product group	42%
Business units share responsibilities for asset development	13%
Hierarchy of core asset development teams and product group asset teams	3%
Other	23%

As product line champion, Schumann must also secure funding for the asset development effort. She will use the results of her study and the questionnaire to prepare her business case that will establish the scope of the product line and define the expected benefits. It will also cite the successful achievements of other organizations and propose an organizational structure for core asset and product development from within her business unit. Armed with this support material, Schumann feels confident that she will be able to make a strong presentation to management. Product lines represent a new and challenging approach for attacking many of the software engineering difficulties that have plagued the industry, and Schumann is committed to making the case for product line adoption.

6.3 Follow an Incremental Approach to Show Early Results

A good first step is to develop an example system to elaborate the possibilities for product lines. A recommended approach [Weiss 99] is to gather some of the best development people for a short period (a week or less) and work a design problem related to real project needs that the organization faces. This study will provide the opportunity to examine various techniques. The problem should exhibit a significant degree of variation and cross various domains that apply to the organization's products.

For Schumann's organization, the scope of design issues includes the family of systems that would apply to both cars and trucks (one major point of variation). Systems in the product line would be used for a variety of purposes:

- on-board diagnosis
- driver performance (safety, economy, emission control)
- routine and special maintenance
- driver training and certification

The example development effort will also examine a variety of subsystems or domains including real-time monitoring, databases, displays, and others. In addition, the effort will examine issues of variability along a number of views:

- domains - In addition to automotive expertise, what other domain experts are needed?
- methods - Employ groups to discuss variation methods that cut across domains, such as automatic software generation or aspect orientation.
- product development - What types of assets are needed to build a variety of products, and what approaches for product development should the asset base support?

Questions to address in this example system include the following:

- Is the process of understanding variability within a domain the same regardless of domains? In the driver-monitoring example, does database design require a different approach from the real-time monitoring domain?
- How does each group handle variation in the domain?
- What support can specialized tools provide?

The example development will investigate and bring out issues that cover requirements, architecture, component design, testing, group organization, product development, and so forth. The product will be a report for management and development staff covering numerous aspects of the product line plus instruction in possible methods for the asset-building phase. From this starting point, an organization can develop follow-up activities to begin putting its own product line approach in place.

6.4 Establish Architecture as the Key Asset

While many elements within the product line organization will compete for resources, product line architecture must be given priority. Based on an understanding of relevant domains and requirements engineering, the architecture partitions the solution space, supports development of assets and products, provides organization of development teams, enables attainment of quality goals, and gives concrete objectives to plans and schedules. Inadequate investment in architecture will result in components that cannot work together properly and a flimsy structure for product development; products will not meet quality factors and will not be maintainable.

Organizations must establish a plan for exploiting architectural excellence within the organization. Single-product development tends to reward accomplishments on a single

project. The organization must make changes in the reward system to provide incentives to product architects who use and strengthen the asset base. The organization must also inculcate direct collaboration among projects in the product line. Enforcing use of the architecture may require a training program in asset structure and use for product development groups.

6.5 Make Cost of Opting Out High

Many organizations decide to pursue a product line approach but, at the same time, let individual projects make their own decisions about participating in the product line. The success stories cited above required projects to be a part of the product line or pay a stiff penalty—usually paying a share of the product line development and sustainment costs out of their own budget. For example, an organization may take asset development costs out of individual project budgets. The understanding is that the projects will now have much of their applications pre-built as product line assets. Projects not using assets will pay double for those elements of their system, perhaps making it impossible for them to meet their own budgets.

6.6 Research Topics

Respondents to the product line questionnaire most often cited architecture as the single hardest element in a product line to create and record. At the same time, they regard the product line architecture as essential to the product line's success. This suggests several areas for applying research within the product line group. The following architecture issues were among those cited by the respondents:

- developers not trained in architecture methods. One respondent described his developers as experts in assembly language, but not in large-scale project development practices.
- inability to evolve architecture with product. Keep component code and architecture in sync.
- defining component interfaces. What architecture standards exist for defining the interface and the required variations across the product line? What degree of variation is allowed within the architecture, and what variation requires architecture change?
- closing conceptual gap between product line architecture and product architecture. What is the relationship of the product line's architecture to the individual product's architecture? How does the product group document its use of the product line architecture to ensure adequate feedback of results to other product line developers? What implementation decisions were made for each product?

A second major area of concern involves extracting and documenting commonality and variability in a collection of existing systems and forecasting those characteristics for future developments. The following issues were among those cited in this category of the questionnaire:

- ability to define a domain. What are the key areas of expertise required to develop a set of related products? What techniques are useful in documenting domain understanding?
- closing the conceptual gap between domain understanding, requirements, and architecture. Organizations find it difficult to record variability requirements, yet without them they cannot identify the points of variation within an architecture. What tools or approaches support domain modeling? How does one use models, use cases, or other analysis results to make architectural decisions such as what elements of a component asset interface must be exposed?
- variation mechanisms and when to apply them. Though not a direct response on the questionnaire, organizations must distinguish variability as a characteristic or quality of the architecture or other assets from variability mechanisms supporting specific types of variability. Jan Bosch has contrasted variability mechanisms according to a product line approach [Bosch 02]. Further research in this area could benefit the product line organization.

A final area in need of further research includes practices for organizational management:

- Determine the essential product line metrics. What metrics must an organization keep to assess the impact of product lines? How are these captured and how are they reported?
- Establish funding and financial models. What funding approaches should an organization consider? Should product line development come from a corporate research program or from technology programs within specific business units? Should the organization capitalize the costs of developing software assets? If so, how should they be capitalized and amortized?
- Perform marketing and technology forecasting. How does an organization determine the ideal scope for a product line, knowing that the product line must support products over a multi-year period beginning some years out? What technology investigations are necessary to maintain currency within the product line?
- Evaluate alternative organizational structures. Can an organization compare alternative organizational strategies in advance of product line adoption? How might results differ given centralized asset development versus asset development distributed to individual business units?
- Determine the effect of product lines on operations. What stages must an organization go through to realize the product line approach as its standard way of doing business? How are managers, developers, suppliers, and customers affected by the transition?

7 Summary

Schumann drew several conclusions from her investigation. Software product lines represent a new approach for software development that can provide real benefits to an organization. While not all situations are appropriate for a product line approach, certain factors generally lead an organization to consider adoption of a product line approach:

- need for multiple products sharing similar requirements
- recognized expertise and a desire for market leadership (or the desire to maintain that leadership)
- customer base to support products for some years to come
- strategic vision within the organization

These conditions were present in the case studies summarized in this report and the characterizations provided by the respondents to the product line questionnaire. From the technical perspective, many companies have been very successful in developing and fielding products using a product line approach. Practices in the management area are less mature but are recognized as an area for further investigation.

While many companies have succeeded in adopting a product line approach as their standard way of doing business, most of the questionnaire respondents are either beginning the adoption process or are in transition. Few of the respondents have adequate data to show meaningful results yet, but they remain committed due to the promise of financial and other benefits. A future study of many of these companies a year or two from now would reveal the extent of product line adoption and the realization of those benefits. That study would also give a wider sampling of success factors, risks, and the critical first steps (or steps to avoid) in product line adoption.

Several sets of studies have helped define the field and are frequently cited:

- Studies by Jan Bosch and colleagues. Bosch has concentrated on many of the organizational issues including organizational structures for asset and product development, organizational maturity, and mechanisms for dealing with variability.
- Institute for Experimental Software Engineering reports. These reports cover methods for product line scoping, domain analysis, product line architecture, product development, and business issues.
- SEI book and reports. The SEI *Framework for Software Product Line Practice*, as the name implies, provides a structure for describing essential software product line practice areas, introduces patterns for applying those practices, and presents case studies of

successful product line adoption. SEI reports cover the gamut of product line practices from making the business case to component strategies.

- Information Technology for European Advancement (ITEA). ITEA is an eight-year strategic pan-European program for research and development in embedded and distributed software. The Engineering Software Architectures, Processes, and Platforms for System Families (ESAPS) and Concepts to Application in System-Family Engineering (CAFÉ) are two projects sponsored by ITEA that bring together leading exponents of software product line practices.
- Software Productivity Consortium (SPC). The SPC was an early advocate of software product line approaches. Many of the consortium member organizations have applied and benefited from the Product Line Management and Engineering (PLME) methodology developed by the SPC.³

Still, there are areas for further research. To identify the appropriateness of specific methods for a particular situation, we must begin to correlate organizational, product, and practice characteristics. The questionnaire provides ample evidence of the range of approaches that organizations are taking, but a more systematic study could reveal the correlation between those approaches, the size of the organization, the size of products in the product line, and such factors as the maturity of the product line. The results of that study would support the organizational planning needed for making the strategic decisions of product line adoption.

³ The SPC's Web site at <www.software.org> contains numerous references and articles on its product line approach, most of which are available to members only. Morisio, Tully, and Ezran report on the application of SPC approaches by Thomson-CSF (now Thales) for training simulation systems [Morisio 00].

Appendix A Software Product Line Questionnaire

Industrial Experience with Product Line Approaches

This questionnaire will examine your current status in adopting software product line approaches. The *Framework for Software Product Line Practice* serves as a reference model for the questions [Clements 02]. The results we obtain from this survey will be available to the community and will help us organize subgroups at an upcoming workshop on Industrial Experience with Product Line Approaches. We will hold the workshop at the International Conference on Software Reuse, on April 16, 2002. Both survey participants and those who did not participate but are interested in discussing the results are welcome to attend. The workshop description is located at http://www.sei.cmu.edu/plp/icsr_workshop/.

The survey results should help us gauge the state of product line practice. We are looking here for industry experience but are not limiting the survey to industry participants. We want to understand the contributions coming from academic organizations or independent researchers and consultants as well. We want to obtain information from four types of organizations:

1. those who already have product line activities in place
2. those working on transitioning to product line approaches
3. those considering the adoption of product lines
4. those who started a product line activity and later abandoned it for whatever reason

We hope that organizations will use the results to compare their progress with that of others and to identify areas for further investigation.

There are also several different business models that we want to examine:

- development of software products derived from a common asset base for use within the enterprise
- development of a platform that serves as a baseline for tailoring and use by other organizations
- commissioning of product line assets from an outside organization
- companies that have attempted product line practices, but have not yet succeeded in widespread adoption or have abandoned the concept

We also want to examine specific aspects of your product line:

- market or mission area
- products already fielded
- product line practices used
- organizational patterns
- economic or other benefits
- factors leading to favorable (or unfavorable) conclusions

Information About Yourself (Optional)

This information is optional. We will keep all responses anonymous, but if you would like us to contact you or send you follow-up information, please supply information. We will keep it confidential.

User Information

- May we contact you to follow up on the answers you are providing regarding your product line activities?

First Name:

Last Name:

Job Title:

Company

Phone:

FAX:

Email:

City:

State:

Country:

ZIP

Information About Your Organization (For consultants: the organization you work with)

How would you classify your organization?

- industry
- government
- academia
- consulting
- other

What is the size of your software development organization?

- 1-50 persons
- 50-300
- 300-1000
- 1000 or larger

What is your position in your organization? (check all that apply)

- software manager
- asset developer
- product developer
- organization policy setter
- researcher
- consultant
- other

What is the scope of the answers you are providing?

- organization-wide
- asset development level
- single or multiple product development
- combination of above or other. Please explain:

Product Line Questions

In your organization, what terms do you use for product line?

- product line
- product family
- application platform
- application framework
- component-based development
- other



What is your definition of software product line or your alternate term?

Please use the forms below to provide your definitions. If alternate terms are used, please include and define them. Also include any citations for your work.

Definitions



Citation(s)



What is your stage in product line adoption?

- currently operating multiple product lines. If so, how many in your organization?



- operating one product line
- developing assets for first product line
- determined to adopt product lines, but adopting new processes and organizational structure
- considering product line adoption
- attempted to adopt a product line approach but did not succeed (Please provide a brief explanation.)



Which best describes the focus of your product line approach?

- We build (or plan to build) product lines using specially built in-house assets.
- We built (or plan to build) product lines using legacy in-house assets.
- We commission an external organization to supply assets based on our specifications.
- We purchase assets (components, COTS) from external organizations.
- We commission an external organization to supply assets and products.
- We built assets or a platform used by external groups for developing their products.
- Other. Please provide an explanation.



What market or mission area do your product lines address?

- aerospace (including embedded and avionics)
- automotive (including embedded and telematics)
- business, e-commerce
- computer systems related (printers, routers, displays, etc.)
- consumer electronics (entertainment related)
- manufacturing
- medical
- telecommunications
- defense weapon related or other (Please explain.)



What technical product line practices are you using? Please provide details or citations below.

- Scoping
- Understanding Relevant Domains
- Requirements Engineering
- Architecture Definition (use of a common architecture for each product line)
- Architecture Evaluation (use of a common architecture for each product line)
- Component Development Strategies (sharing components between products)
- Product Line Testing
- Configuration Management. What percentage of code for a typical product is derived from assets?



- Legacy Mining
- Data Collection, Metrics, and Tracking
- Tools Support

Please provide any details or citations below.

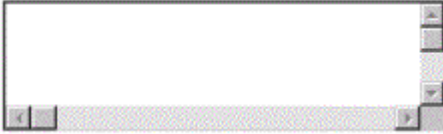
(Examples: What variability mechanisms do you use? Do you use generator technology?)



What tools are you using for product line support? Please provide details or citations below.

Requirements Engineering

- | | | | |
|---|------------------------------------|--|--------------------------------------|
| <input type="checkbox"/> Requisite Pro | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> DOORS | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> SLATE | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> Others or homegrown (Please provide any details or citations below.) | | | |



Modeling

- | | | | |
|---|------------------------------------|--|--------------------------------------|
| <input type="checkbox"/> Rational Rose | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> Artisan Studio | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> Rhapsody (I-Logix) | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> Others or homegrown (Please provide any details or citations below.) | | | |



Configuration Management

- | | | | |
|---|------------------------------------|--|--------------------------------------|
| <input type="checkbox"/> Clear Case | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> CCC | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> CM Synergy | <input type="checkbox"/> Satisfied | <input type="checkbox"/> Needs improvement | <input type="checkbox"/> Unsatisfied |
| <input type="checkbox"/> Others or homegrown (Please provide any details or citations below.) | | | |



How would you characterize your product line organizational structure?

- Core asset development is performed within product group(s).
- Business units share responsibilities for asset development.
- Separate core asset development unit supplies assets to product groups.
- Organization consists of hierarchy of core asset development units: corporate wide supplying assets to domain engineering units within product groups.
- Other (Please provide any details or citations below.)

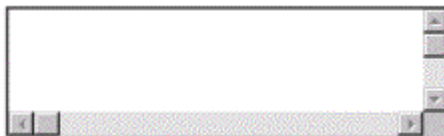


What led you to adopt your product line approach? Please provide details or citations below.

- reduce time to market
- increase market share
- improve productivity in features per new product per time period
- reduce costs
- improve ability to meet customization demands
- improve quality
- Other (Please explain.)



Please provide any details or citations below.



What concrete benefits have you achieved that you attribute to your product line approach?

(Please provide details or citations below.)

- reduced time to market
- increased market share
- improved productivity in features per new product per time period
- reduced costs
- improved ability to meet customization demands
- improved quality
- Other (Please explain.)



Please provide any details or citations below.



What difficulties did you encounter while adopting product line approaches?

- developer resistance
- management resistance
- organizational resistance to cross-organization efforts
- large initial investment
- inability to measure impact
- lack of necessary skills
- too much lead time to be of benefit
- Other (Please provide details.)



Where do you obtain your information about product line approaches?

Papers:

An empty rectangular text box with a light gray background and a thin black border. It has a scroll bar on the right side and a small icon in the bottom-left corner.

Books:

An empty rectangular text box with a light gray background and a thin black border. It has a scroll bar on the right side and a small icon in the bottom-left corner.

Other:

An empty rectangular text box with a light gray background and a thin black border. It has a scroll bar on the right side and a small icon in the bottom-left corner.

What kind(s) of information do you write down when you document the software product line?

An empty rectangular text box with a light gray background and a thin black border. It has a scroll bar on the right side and a small icon in the bottom-left corner.

What kind(s) of information do you have the hardest time writing down about a software product line, and why?

An empty rectangular text box with a thin black border and a light gray background. It has small square handles in the corners for resizing.

Additional comments:

An empty rectangular text box with a thin black border and a light gray background. It has small square handles in the corners for resizing.

Appendix B Results of the Software Product Line Questionnaire

Returns on Questionnaire

Total sent out: approximately 150

Total responses: approximately 31

Respondent Information

Organization

- 71% industry
- 10% consulting
- 7% government
- 3% academia
- 10% no response or other

Organization size

- 32% 1-50
- 23% 51-300
- 19% 301-1000
- 19% 1001+
- 7% N/A

Respondent Role

Title (more than one allowed)

- 55% - researcher
- 42% - consultant
- 20% - other (50% of these identified architect as role)
- 19% - product developer
- 16% - asset developer
- 16% - software manager
- 7% - organization policy maker

Scope of answers

- 42% - organization wide
- 26% - some combination
- 19% - single or multiple product developments
- 3% - asset development level
- 10% - other

Terminology

For product line (more than one allowed):

- 55% - product line
- 39% - product family
- 39% - component-based development
- 26% - application platform
- 16% - application framework

Others

- Aspects layer
- Business domains
- Multi-mission (from military)
- Platform or platform-based development
- Product library
- Product suite
- Product population
- Software mass customization

Definitions

- 22% use SEI definition

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 02].

- Other definitions
 - generally refer to “set of systems”
 - emphasize specific approach they follow (mass customization or generative)
 - emphasize the asset base in terms of an application platform or product library with components ready for assembly
 - place importance on use of a common architecture for their products
 - degree of variability among products an important factor – Some organizations look for a minimum degree of variation among products, while others develop a product line that encompasses a wide degree of variation.

Focus of Product Line Approach (more than one possible)

- 74% - build own assets and products
- 48% - use legacy assets to build products
- 29% - use COTS assets
- 22% - commission other organizations to build assets
- 13% - build assets or platform used by other organizations to build products
- 3% - commission external organization to build products

Market Areas (more than one possible)

- 32% - medical
- 19% - aerospace
- 19% - telecommunication
- 19% - business, e-commerce
- 13% - automotive
- 10% - defense weapon related
- 10% - consumer electronics
- 10% - manufacturing
- 3% - computer systems
- 23% - other (software tools, insurance, embedded systems, large equipment, civil engineering)

Sizes of Products

Percent of product typically developed from assets: 50-90%

Code size

- 13% - Under 50KLOC
- 17% - 50 – 200KLOC
- 22% - 200KLOC – 1MLOC (million lines of code)
- 27% - 1MLOC+
- 17% - N/A
- 4% - could not tell

Practices Applied (more than one possible)

- 55% - Scoping
- 65% - Understanding Relevant Domains
- 58% - Requirements Engineering
- 87% - Architecture Definition
- 35% - Architecture Evaluation
- 87% - Component Development Strategies
- 39% - Product Line Testing
- 77% - Configuration Management
- 26% - Legacy Mining
- 32% - Data Collection, Metrics, and Tracking
- 42% - Tools Support

Tools

Category	Name	Percent Using Tool	Among Users Percent Satisfied	Among Users Percent Saying Needs Improvement	Among Users Percent Unsatisfied
Requirements Engineering	Requisite pro	26	27	62	11
	Doors	19	16	68	16
	Slate	3		100	
	Others, homegrown	52			
Modeling	Rose	55		65	6
	Artisan	3		100	
	Rhapsody	7		100	
	Others	35			
Configuration Management	Clear Case	48	27	67	
	CCC	3		100	
	CM Synergy	3		100	
	Others (Microsoft Source Safe, CVS, homegrown)	45			

Organizational Structure

- 42% - Core asset development as part of product group
- 39% - Separate core asset development unit supplying assets to product groups
- 13% - Business units share responsibilities
- 3% - Hierarchy of core asset development teams and product group asset teams
- 3% - N/A

Product Line Goals Set by the Organization (more than one possible)

- 67% - Reduce costs
- 65% - Reduce time to market
- 55% - Improve productivity
- 55% - Improve quality
- 45% - Meet customization requirements
- 13% - Increase market share
- 19% - Other (small effort produces much of product, new markets, customer value, manage complexity, extensibility)

Product Line Results Achieved to Date (more than one possible)

- 45% - Reduce costs
- 30% - Reduce time to market
- 30% - Improve productivity
- 52% - Improve quality
- 39% - Meet customization requirements
- 7% - Increase market share
- 7% - Other (product management flexibility, organizational alignment, ability to sell upgraded systems to customers)

Greatest Difficulties

- 52% - Organizational resistance
- 45% - Large initial investment
- 36% - Management resistance
- 32% - Developer resistance
- 29% - Lack of skills
- 19% - Inability to measure impact
- 16% - Too much lead time
- 19% - Other (customer skepticism, drive to return to traditional ways of development, poor timing such as during product release)

Appendix C Workshop Report

Participant Presentations

The Industrial Experience with Product Line Approaches Workshop brought together software product line practitioners and researchers at the International Conference on Software Reuse, which was held in Austin, Texas on April 15-18, 2002. Participants exchanged ideas and experiences, discussed current and emerging practices, and introduced next-generation concepts. The organizers set the following goals for the workshop:

1. Develop a better understanding of the state of the practice in software product lines.
2. Review an evaluation tool in the form of a questionnaire that practitioners and researchers can use to gauge the state of product line practice.

This workshop explored the state of product line practice across the software industry. The variety of techniques that organizations have used to capture product line results was another area of interest. The workshop covered product line characteristics such as

- market area
- products already fielded
- product line practices used
- organizational patterns
- economic or other benefits

The workshop attracted representatives from several types of organizations. Some of the organizations have already experienced considerable savings in using a product line approach for software system production. Other organizations were attracted to the idea but are in varying stages of integrating product line practices into their operations. Table 11 lists the presenters at the workshop.

Table 11: Workshop Attendees Making Presentations

Name	Organization	Product Line Activity
Ira Baxter	Semantic Design, Austin, TX	Constructor technology
Dale Churtett	Salion, Austin, TX	Automotive manufacturing support
Craig Cleaveland	Independent consultant	Constructor technology
Sholom Cohen	SEI	Researcher and consultant
Charles J. Krueger	BigLever Software, Austin, TX	Tool developer
Dirk Muthig	Institute for Experimental Software Engineering (IESE), Kaiserslautern, Germany	Researcher and consultant
Luiz Paolo and Alves Franca	Programare, Rio de Janeiro, Brazil	Database technology

Other participants included Pat Donohoe and Bob Krut of the SEI and Krzysztof Czarnecki of Daimler Chrysler.

Sholom Cohen, SEI

This presentation summarized the results of the product line questionnaire. Cohen also summarized the goals of the workshop for the participants.

Charles Krueger, BigLever Software, Inc.

This presentation focused on the use of tool support for product line approaches. BigLever has developed a tool named GEARS that organizes components and variants for integration into product line products. GEARS establishes a production line incorporating an infrastructure and an instantiation used to generate product instances. Krueger emphasized the need to minimize the paradigm shift for the organization adopting a production line. He recommended the use of lightweight approaches.

The presentation discussed four approaches.

1. **Proactive** strategy (the “big-bang” approach). Create assets and infrastructure up front; a year or two later, you have a product line. This is not exactly a wrong approach, just heavyweight and highly risky. Tasks include domain analysis and scoping, creating a product line architecture with variation points, design, and implementation.
2. **Reactive** strategy. This approach is also known as “delta engineering.” A “generic product” emerges based on the experience of the previous N products. The generic product supports turning out product N+1. BigLever has used this approach with Salion. GEARS stores the information about the generic product and the deltas that are required to generate a new product. The generic product includes 250 KLOC including some

COTS software. Ninety percent of individual Salion products come from the generic product. This approach requires good abstract thinkers to drive the effort but otherwise routine engineering once the approach is in place.

3. **Extractive** strategy. This approach uses a collection of existing products. Typically, an organization that has fielded 5 to 15 products realizes that it could benefit from reuse of software. It needs to extract the reusable software from the existing products. BigLever has a project with Avaya for telephony servers following this approach. Variation management in these systems is very difficult; there is a real need to create a production line.
4. **Refactorive** strategy. In this approach, an organization has created a production line and then noticed particular patterns with respect to the variations requested by customers. It becomes possible to refactor some of the customizations into the general production capability so that the level of customer-specific variation is reduced in future projects.

In practice, Krueger reported that organizations use combinations of the different strategies and often use configuration management for product line variation management. The rule is “variations manage the production line rather than the product line.” The configuration management capability can be treated as a single product. The products are *outside* of configuration management; they are generated, disposable outputs of the configuration manager.

Dale Churtett, Salion

Salion builds manufacturing supply systems for automobile manufacturers. Salion’s business model uses application service providers (ASPs) as the means for making the supply systems available to the factory. Salion relies on rapid customer implementation (under 60 days). Their business model offers several options to customers when they buy a particular solution from Salion. The products share much of the same infrastructure. However, despite their similarity, they are implemented in completely different ways.

The new development culture at Salion is based on agile development. They use the Rational Unified Process (RUP), using a lightweight version. Salion also applies some of the best practices from extreme programming. In addition, they apply component-based development and handle mass customization by using a reactive approach supported by BigLever Software’s GEARS tool. This effort began about a year ago.

The GEARS application is in support of quotation process software for various parts suppliers. The quotation process includes steps for pre-quote and post-quote, with separate versions for each customer. Salion recognized a need for a platform of core software that could be customized, but it could not adequately address customization given its existing approach. The marketing group wanted a turnkey approach with the ability to turn features in products on and off and license those features separately. BigLever was originally brought to support configuration management for these various versions. Salion found GEARS to be an approach that offered the configurable customization they needed. GEARS creates a configuration file that gets rid of all the coded selections to turn features on and off. Salion

has already worked with two customers and is now working on its third customer project. The mass customization software can now generate five products from a basic capability.

A second element of the approach relies on constantly applying clone detection. Clone detection identifies identical or highly similar software code segments. These segments may be in a single file, in multiple files of a single product, or spread across multiple products. Salion's cloning problem stemmed from overuse of cut-and-paste programming. Programmers had cut segments from one area in a product and pasted them in other areas. They then made minor modifications. Using DMS, a clone detection tool from Semantic Design, Salion identified 12.9% of the code (27 KLOC) as cloned code. (In fact, this is a fairly typical number, according to Ira Baxter, who has performed clone detection for several companies.) Maintaining this code is a significant cost overhead. If the code is cloned only in one additional place, that's 13.5 KLOC that should not be maintained if the clones could be managed as a single, reused asset. DMS also discovered bugs in the code. It provided a better understanding of system structure than reverse engineering Rational Rose did and exposed areas of poor design, both in the application programming interface (API) and the implementation.

Churtett listed the following benefits of clone detection:

- Clones were removed in areas that are customization hot spots.
- User interface code was easier to maintain.
- It is a powerful tool when coupled with an agile development process that promotes continual refactoring.
- There is no need to worry about cut-and-paste programming because every few months the clone detection will clean up the code.
- Clone detection results drive the design for the next generation of the product (e.g., exception-handling framework, user interface framework).
- The design for the next generation of the product will contain half the code of the previous generation.

Dirk Muthig, IESE

IESE follows a quality improvement program for product line adoption. They have identified domain boundaries or product line scope as a key problem area. The key challenges are finding what can be reused, documenting the assets, evaluating best candidate systems for reuse, and using assets in multiple products. IESE measures the success in terms of the number of people who are using a technology and are satisfied with it.

Muthig reported on a model for analyzing product line application. His approach characterizes a product line in terms of the variety of products versus the change rate. A company may produce a large number of products with slight variations. This is typical of

consumer products such as cell phones. At the other extreme, a company may produce a small number of products with significant variations. Large-scale systems with a long product life, such as medical equipment or military command and control systems, are typical. Using this model, Muthig believes that it is possible to predict the types of product line approaches that will be most productive.

Luiz Paulo, Programare

Programare uses a generative approach for producing database management systems (DBMSs). The generators emerged over a period of 10 years of mostly manual development of DBMSs. The approach is based on a high-quality development team, with a strong product line champion of both the asset and product development sides.

Paulo has introduced companies to both the DMBS generation tool and to the process for fielding products. Initially, Programare only developed the deliverable software, but now it also delivers the ability to generate products in the form of the generator.

Independent of the technology, Paulo recommended starting with simpler problems and moving to complex ones. For database systems, the initial efforts may focus on the schema and then move to the interface and the client side.

Craig Cleaveland, Independent Consultant

Cleaveland's work has focused on finding the relationship between software product lines and the technologies used to support them (generative, mass customization, etc.). He considers how to get the biggest return on investment. For a telecommunications company, Cleaveland supported analysis of a high-level abstraction for call processing. The approach used the Specification and Description Language (SDL). The products were for voiceover Internet Protocol (IP) for cable networks. The company's initial approach did not use sophisticated tools; it involved using Visio for the SDL diagrams (specifically Visio's Extensible Markup Language [XML] export capability) and then converting the XML to actual code. This example illustrates the use of proven technology, but the approach required reading and annotation of the generated XML.

To date, this organization has developed components (e.g., call processing part, user interface part, hardware device part). It does not have concrete measurements available. The organization feels that the product line approach is an advance over the prospect of a large development organization producing code that was not understandable at a higher level and that does not provide good mapping between SDL diagrams and code.

Ira Baxter, Sematic Design

The core tool of Semantic Design is the Design Maintenance System (DMS). The tool has been in limited use for full maintenance, but it has been used widely in automatic generation of applications. Baxter emphasized the concept of “recursive automation,” where elements of the tool are themselves generated by the tool or where problems are recast in terms of solutions that the tool already offers. DMS is, itself, composed of several domains, for example, lexical parsing, grammars, attribute computations, rewrite rules, and parlance (which is a special-purpose programming language).

In offering this capability, Semantic Design emphasizes the ability of DMS to create derivative products. Such products are the results that customers want, rather than products of an obscure, “giant recursive system.” For example, a key feature of DMS is clone detection. (See the previous section summarizing Dale Churtett’s presentation.) This capability is a generator within DMS and a product line in its own right, since there are clone detectors for various languages. Customers may not relate to clone detection, but they do relate to reduced maintenance through code reduction.

Sematic Design has also developed tools for external customers. They have developed tools to generate the programmable logic controller (PLC) code for the automotive industry. A typical device for manufacturing car doors may require 20,000 Boolean equations. With the automatic generator produced by DMS, the manufacturer is able to generate that code in about 200 lines. Semantic Design (or the manufacturer) can reconfigure code generators for different targets (all driven off the same target generator). This approach is successful, because the company founders were already familiar with factory automation before forming the company.

Discussion

The workshop participants developed the following list for further discussion of issues raised during the presentations:

1. **Business drivers:** What are the business drivers leading to product lines? How can an organization effectively use support technology (e.g., generative, software libraries, and mass customization)?
2. **Success factors:** How does an organization share information throughout levels of management (e.g., existence of asset base, adoption of approach, and aligning practices)?
3. **Obstacles:** What are the biggest obstacles to reuse across multiple products (e.g., obstacles to getting the reuse framework in place)?
4. **Quantifying success:** What is the best way to quantify success of using a product line approach (e.g., the old way costs \$X, the new way costs \$Y)? How do you measure this?
5. **Getting the message out:** Identify the candidates who are close to having a product line (and who may not be calling it a product line).

The following sections summarize these discussions.

Business Drivers

Most organizations establish productivity goals for the product line efforts. They must be able to deliver systems in a shorter time at reduced cost. They may, in some cases, trade off these factors, sacrificing cost to meet the pressures of quicker time to market or reducing costs at the expense of a longer delivery cycle. Improved quality is another driver. The repeated use of a tested architecture and assets reduces risks, improves quality, and supports the predictability of delivery.

Other organizations have turned to product lines out of desperation. The organization saw a need to deliver more products than its current staff could handle, in a shorter time period, and at lower costs to show a profit. It either could not add staff to meet schedules or could not afford to hire the additional staff given its current approaches. Previous case studies have documented this situation.

Success Factors

A major success factor is the depth of experience in the product line market area. Semantic Design was successful in the factory automation area because the founders came from a factory automation background. They could immediately relate to the problems that automotive manufacturing and the PLCs addressed.

In some cases, product line technology is seen as exotic or purely academic. Customers don't really understand it. For product line approaches to succeed, the technologist must demonstrate success in an area that is highly relevant to the organization. Case studies may suffice to initiate a technology investigation, but to obtain buy-in, practical demonstrations within the key business areas of an organization are essential. Most product lines take several years to get off the ground, and the developers must show measured progress and return to maintain momentum and support.

Baxter recommended tackling maintenance issues that are common to all organizations. All organizations have large amounts of legacy code. A major success for these organizations is reducing maintenance costs or translating the legacy code to a more maintainable form. A product line approach may offer assets to substitute for areas within the legacy code that may be addressed through common solutions. Semantic Design has experience in producing translators from obscure or obsolete languages to a more maintainable form. By addressing the short-term needs of controlling maintenance costs, the technology group may "get a foot in the door" for the more long-term product line approach.

Obstacles

The participants have found it difficult to change the mindset of organizations. The obstacle is the accepted construction-based approach versus generation or composition from assets. Where software is produced for an outside organization, the argument is the need to be customer specific versus the development of products from a common core. A related experience is the difficulty that technology groups encounter in convincing organizations to take on product lines; the technologists must sell the approach indirectly. Doing so may include solving a set of related problems, such as the maintenance improvements that Semantic Design has brought to organizations it supported. Once the organization buys into a product line approach, even indirectly and not recognizing it as such, the technologist can offer approaches for continuous improvement.

Many organizations believe that they must be able to “read” any code running in support of their operations. For automatic generation, this readability requirement may pose a problem. While the domain-specific language (DSL) may be highly readable, the actual code generated is not. Customers often do not think of software as the abstractions supported through the DSL. While component software may not have this same degree of unacceptability, organizations are still reluctant to accept software for which they do not have total control, unless that software is “shrink-wrapped” and unchanged once it leaves the developer’s organization. There are too many instances of unique customizations that have failed to deliver the promises of software reuse.

Companies tend to treat their products as a black box; they don’t care about product lines. Often, they want to know what COTS products are used, but they don’t actively seek out opportunities for systematic reuse. Customers are mainly interested in security rather than whether the product is part of a product line. They track production based on individual products and don’t collect data at a sufficiently low level to measure success. Among other issues, these obstacles manifest themselves in the inability to quantify success.

Quantifying Success

Most organizations have not been able to quantify the results of product line approaches. They have observed and reported qualitative effects, but they can’t produce meaningful numbers on return on investment or productivity improvements. Semantic Design can report on its results. Manual development of a logic controller generally takes 8-10 person months. The logic controller developed by Semantic Design required 2.5 person months. The customer in that case had expert help that may not be generally available, but this result is one concrete quantified data point.

While many studies have made the case for product lines in selected areas, they are not seen as relevant in a different context. Even where concrete results are reported, as in the Cummins case study, other organizations don’t see similarities to their own situation. The few

long-term studies of product lines that show the merits of products line approaches are seen within most organizations as insufficient to warrant installing a measurement process.

Getting the Word Out

The workshop made several recommendations for spreading success stories:

- The “nothing succeeds like success” approach works. Identify highly visible successes but use them only where the context of the organization is similar to that in the success story.
- Target individuals at a level higher than the developer: vice president of engineering, chief technical officer, and so forth.
- Product line technologists must stress the concepts of strategic customization. These product line approaches must be emphasized as a business decision. Where the business model is to beat competition through the variety of product offerings, the technology group must offer product line approaches as viable.
- Identify industry analysis organizations to undertake a study of product line approaches. Giga Information Group, Gartner, or Forrester are organizations that target technology for investigation.

Comparison of Approaches

The workshop developed a matrix to demonstrate the effectiveness of the approaches proposed by the participants. Table 3 on page 11 captures the information on those approaches.

References

- [America 00]** America, Pierre & Wijgerden, Jan van. "Requirements Modeling for Families of Complex Systems," 199-209. *Proceedings of the International Workshop on Software Architectures for Product Families*. Las Palmas de Gran Canaria, Spain, March 15-17, 2000. New York, NY: Springer-Verlag, 2000.
- [Batory 00]** Batory, Don; Johnson, Clay; MacDonald, Bob; & von Heeder, Dale. "Achieving Extensibility Through Product Lines and Domain-Specific Languages: A Case Study," 117-136. *Proceedings of the 6th International Conference on Software Reuse*. Vienna, Austria, June 27-29, 2000. New York, NY: Springer-Verlag, 2000.
- [Bosch 01]** Bosch, Jan. "Software Product Lines: Organizational Alternatives," 91-100. *Proceedings of the 23rd International Conference on Software Engineering*. Toronto, Canada, May 12-19, 2001. Los Alamitos, CA: IEEE Computer Society, 2001.
- [Bosch 02]** Bosch, Jan. "Maturity and Evolution in Software Product Lines: Approaches, Artifacts and Organization," 257-271. *Proceedings of the Second Product Line Conference (SPLC 2)*. San Diego, CA, August 19-22, 2002. New York, NY: Springer-Verlag, 2002.
- [Brownsword 96]** Brownsword, Lisa & Clements, Paul. *A Case Study in Successful Product Line Development* (CMU/SEI-96-TR-016, ADA315802). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1996. <<http://www.sei.cmu.edu/publications/documents/96.reports/96.tr.016.html>>.
- [Clements 00]** Clements, Paul & Northrop, Linda. *A Framework for Software Product Line Practice, Version 3.0* [online]. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000. <<http://www.sei.cmu.edu/plp/framework.html>>.

- [Clements 01]** Clements, Paul; Cohen, Sholom; Donohoe, Patrick; & Northrop, Linda. *Control Channel Toolkit: A Software Product Line Case Study* (CMU/SEI-2001-TR-030, ADA396286). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2001. <<http://www.sei.cmu.edu/publications/documents/01.reports/01tr030.html>>.
- [Clements 02]** Clements, Paul & Northrop, Linda. *Software Product Lines: Practices and Patterns*. Reading, MA: Addison Wesley, 2002.
- [Dager 00]** Dager, James C. "Cummins' Experience in Developing a Software Product Line Architecture for Real-Time Embedded Diesel Engine Controls," 23-46. *Software Product Lines: Experience and Research Direction, Proceedings of the First Software Product Lines Conference (SPLC1)*. Denver, Colorado, August 28-31, 2000. Boston, MA: Kluwer Academic Publishers, 2000.
- [Donohoe 00]** Donohoe, Patrick, ed. *Software Product Lines: Experience and Directions, Proceedings of the First Software Product Lines Conference (SPLC1)*, August 28-31, 2000. Denver, Colorado. Boston, MA: Kluwer Academic Publishers, 2000.
- [IESE 02]** Institute for Experimental Software Engineering. *PuLSE™ (Product Lines for Software Systems)* [online]. Kaiserslautern, Germany: Institute for Experimental Software Engineering, 2002. <<http://www.iese.fhg.de/PuLSE/>>.
- [Krueger 01]** Krueger, Charles. "Easing the Transition to Software Mass Customization," 283-293. *Proceedings of the 4th International Workshop on Product Family Engineering*. Bilbao, Spain, October 3-5, 2001. New York, NY: Springer-Verlag, 2001.
- [Morisio 00]** Morisio, Maurizio; Tully, Colin; & Ezran, Michel. "Diversity in Reuse Process." *IEEE Software* 17, 4 (July/August 2000): 56-63.
- [Pronk 00]** Pronk, Ben J. "An Interface Based Platform Approach," 331-352. *Software Product Lines: Experience and Research Directions*. Boston, MA: Kluwer Academic Publishers, 2000.
- [Weiss 99]** Weiss, David & Lai, Chi Tau Robert. *Software Product Line Engineering: A Family-Based Software Development Process*. Reading, MA: Addison Wesley, 1999.

REPORT DOCUMENTATION PAGE			<i>Form Approved OMB No. 0704-0188</i>	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE September 2002	3. REPORT TYPE AND DATES COVERED Final		
4. TITLE AND SUBTITLE Product Line State of the Practice Report		5. FUNDING NUMBERS F19628-00-C-0003		
6. AUTHOR(s) Sholom Cohen				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213			8. PERFORMING ORGANIZATION REPORT NUMBER CMU/SEI-2002-TN-017	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES				
12A DISTRIBUTION/AVAILABILITY STATEMENT Unclassified/Unlimited, DTIC, NTIS			12B DISTRIBUTION CODE	
13. ABSTRACT (MAXIMUM 200 WORDS) Software product lines represent a new and promising approach for fielding software systems. Companies that have launched software product line efforts report significant improvements in cost savings, quality, productivity, and time to market. Technically, they report a high degree of success in developing approaches that address the software engineering and technical management of their product lines. Organizationally, these companies report success as well as challenges that they must still overcome. This technical note reports on the state of software product line practice in industry. The report uses a narrative approach, based on a composite of individual companies' experiences in implementing software product lines. The report blends a case study with the results of a product line questionnaire that was sent to organizations with meaningful product line experiences and with the results of a product line workshop held during the recent International Conference on Software Reuse.				
14. SUBJECT TERMS software product line, state of the practice, business case, Framework for Software Product Line Practice, software development, case study			15. NUMBER OF PAGES 72	
16. PRICE CODE				
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	