<table>
<thead>
<tr>
<th><strong>1. REPORT DATE</strong></th>
<th>JUL 2008</th>
<th><strong>2. REPORT TYPE</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>3. DATES COVERED</strong></td>
<td>00-00-2008 to 00-00-2008</td>
<td><strong>4. TITLE AND SUBTITLE</strong></td>
<td>Software Product Lines Essentials</td>
</tr>
<tr>
<td><strong>5a. CONTRACT NUMBER</strong></td>
<td></td>
<td><strong>5b. GRANT NUMBER</strong></td>
<td></td>
</tr>
<tr>
<td><strong>5c. PROGRAM ELEMENT NUMBER</strong></td>
<td></td>
<td><strong>5d. PROJECT NUMBER</strong></td>
<td></td>
</tr>
<tr>
<td><strong>5e. TASK NUMBER</strong></td>
<td></td>
<td><strong>5f. WORK UNIT NUMBER</strong></td>
<td></td>
</tr>
<tr>
<td><strong>6. AUTHOR(S)</strong></td>
<td></td>
<td><strong>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</strong></td>
<td>Carnegie Mellon University, Software Engineering Institute (SEI), Pittsburgh, PA, 15213</td>
</tr>
<tr>
<td><strong>8. PERFORMING ORGANIZATION REPORT NUMBER</strong></td>
<td></td>
<td><strong>9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>10. SPONSOR/MONITOR'S ACRONYM(S)</strong></td>
<td></td>
<td><strong>11. SPONSOR/MONITOR'S REPORT NUMBER(S)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>12. DISTRIBUTION/AVAILABILITY STATEMENT</strong></td>
<td>Approved for public release; distribution unlimited</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>13. SUPPLEMENTARY NOTES</strong></td>
<td></td>
<td><strong>14. ABSTRACT</strong></td>
<td></td>
</tr>
<tr>
<td><strong>15. SUBJECT TERMS</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>16. SECURITY CLASSIFICATION OF:</strong></td>
<td>a. REPORT unclassified</td>
<td>b. ABSTRACT unclassified</td>
<td>c. THIS PAGE unclassified</td>
</tr>
<tr>
<td><strong>17. LIMITATION OF ABSTRACT</strong></td>
<td>Same as Report (SAR)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>18. NUMBER OF PAGES</strong></td>
<td>85</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>19a. NAME OF RESPONSIBLE PERSON</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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 presentation is not intended in any way to infringe on the rights of the trademark holder.

This Presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

This work was created in the performance of Federal Government Contract Number FA8721-05-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.
Today’s Session

Introduction
Product Line Concepts
  • What
  • Why
  • How

Conclusion
Business Success Requires Software Prowess

Software pervades every sector.
Software has become the bottom line for many organizations, even those who never envisioned themselves in the software business.
Universal Business Goals

- High quality
- Quick time to market
- Market dominance
- Market agility
- Product alignment
- Low cost production
- Low cost maintenance
- Mass customization

require

IMPROVED EFFICIENCY AND PRODUCTIVITY
Software (System) Strategies

Process improvement
Technology innovation
Reuse
Few Systems Are Unique

Most organizations produce families of similar systems, differentiated by features.
A reuse strategy makes sense.
Traditional reuse strategies have had little economic benefit.
Reuse History

Focus was small-grained, opportunistic, and technology-driven. Results did not meet business goals.
Reuse History

1960s SUBROUTINES
1970s MODULES
1980s OBJECTS
1990s COMPONENTS
2000s SERVICES
Strategic Reuse is Needed for Business Benefits
Celsiustech: Ship System 2000

A family of 55 ship systems

- Need for developers dropped from 210 to roughly 30.
- Time to field decreased from about 9 years to about 3 years.
- Integration test of 1-1.5 million SLOC requires 1-2 people.
- Rehosting to a new platform/OS takes 3 months.
- Cost and schedule targets are predictably met.
Cummins Inc.: Diesel Control Systems

Over 20 product groups with over 1,000 separate engine applications

- Product cycle time was slashed from 250 person-months to a few person-months.
- Build and integration time was reduced from one year to one week.
- Quality goals are exceeded.
- Customer satisfaction is high.
- Product schedules are met.
National Reconnaissance Office/ Raytheon: Control Channel Toolkit

Ground-based spacecraft command and control systems

• First system had 10 times fewer defects than usual.
• The incremental build time was reduced from months to weeks.
• The system development time and costs decreased by 50%.
• There was decreased product risk.
Market Maker GMBH: Merger

Internet-based stock market software

- Each product is “uniquely” configured.
- Putting up a customized system takes three days.
Nokia Mobile Phones

Product lines with 25-30 new products per year versus 5 per year originally. Across products there are

- varying number of keys
- varying display sizes
- varying sets of features
- 58 languages supported
- 130 countries served
- multiple protocols
- needs for backwards compatibility
- configurable features
- needs for product behavior
- change after release
How Did They Do It?

SOFTWARE PRODUCT LINES
Reuse History: From Ad Hoc To Systematic
Today’s Session

Introduction

*Product Line Concepts*

- *What*
- Why
- How

Conclusion
What Is A Software Product Line?

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.

- a new application of a proven concept
- an innovative, growing concept in software engineering
Software Product Lines

Product lines
• take economic advantage of commonality
• bound variation

BUSINESS GOALS/
APPLICATION DOMAIN

is satisfied by

ARCHITECTURE

used to structure

COMPONENTS
and SERVICES

CORE ASSETS

pertain to

PRODUCTS

share an

are built from

Software Engineering Institute
Carnegie Mellon

Software Product Lines
Linda Northrop
© 2008 Carnegie Mellon University
How Do Product Lines Help?

Product lines amortize the investment in these and other core assets:

- requirements and requirements analysis
- domain model
- software architecture and design
- performance engineering
- documentation
- test plans, test cases, and test data
- people: their knowledge and skills
- processes, methods, and tools
- budgets, schedules, and work plans
- components and services

PRODUCT LINES = STRATEGIC REUSE
The Key Concepts

Use of a core asset base in production of a related set of products
The Key Concepts

Use of a core asset base in production of a related set of products

- Architecture
- Production Plan
- Scope Definition Business Case

Business Case Scope Definition

© 2008 Carnegie Mellon University
Software Product Lines Are Not

Clone and own: single-system development with reuse
  • modifying code as necessary for the single system only
Fortuitous small-grained reuse
  • reuse libraries containing algorithms, modules, objects, or components
Just component-based or service-based development
  • selecting components or services from an in-house library, the marketplace, or the Web with no architecture focus
Just versions of a single product
  • rather, simultaneous release and support of multiple products
Just a configurable architecture
  • a good start, but only part of the reuse potential
Just a set of technical standards
  • constraining choices without an architecture-based reuse strategy
Software Product Lines Are

Software product lines involve strategic, planned reuse that yields predictable results.
Widespread Use of Software Product Lines

Successful software product lines have been built for families of among other things

- mobile phones
- command and control ship systems
- satellite ground station systems
- avionics systems
- command and control/situation awareness systems
- pagers
- engine control systems
- mass storage devices

- billing systems
- web-based retail systems
- printers
- consumer electronic products
- acquisition management enterprise systems
- financial and tax systems
- medical devices
- farm fish management software
Specific Examples - 1

- **akvasmart**: Feed control and farm management software
- **Boeing**: Bold Stroke Avionics
- **ABB**: Asea Brown Boveri
  - Gas turbines, train control, semantic graphics framework
- **Dialect**: Internet payment gateway infrastructure products
- **Axis Communications**: Computer printer servers, storage servers, network camera and scanner servers
- **E-Com Technology Ltd.**: Customized solutions for transportation industries
- **Ericsson**: AXE family of telecommunications switches
- **LG**: Elevator control systems
- **Nokia**: Mobile phones, mobile browsers, telecom products for public, private and cellular networks
- **GM**: Software for engines, transmissions and controllers
- **LSI Logic**: RAID controller firmware for disk storage units
- **Lucent Technologies**: 5ESS telecommunications switch
- **NASA**: Interferometer product line
- **Software Engineering Institute | Carnegie Mellon**
Specific Examples - 2

**PHILIPS**
High-end televisions, PKI telecommunications switching system, diagnostic imaging equipment

**RICOH**
Office appliances

**SALION**
Revenue acquisition management systems

**BOSCH**
Automotive gasoline systems

**SIEMENS**
Software for viewing and quantifying radiological images

**TELVENT**
Industrial supervisory control and business process management systems

**testo**
Climate and flue gas measurement devices

**alltel**
Support software

**MOTOROLA**
Pagers product line

**symbian**
EPOC operating system

**NAVSEA**
Test range facilities

**U.S. ARMY**
Command and control simulator for Army fire support

**Rockwell Collins**
Commercial flight control system avionics, Common Army Avionics System (CAAS), U.S. Army helicopters

Software Product Lines
Linda Northrop
© 2008 Carnegie Mellon University
Today’s Session

Introduction

Product Line Concepts

• What
• Why
• How

Conclusion
Real World Motivation

Organizations use product line practices to:

- achieve large scale productivity gains
- improve time to market
- maintain market presence
- sustain unprecedented growth
- achieve greater market agility
- compensate for an inability to hire
- enable mass customization
- get control of diverse product configurations
- improve product quality
- increase customer satisfaction
- increase predictability of cost, schedule, and quality
Example Organizational Benefits

Improved productivity
• by as much as 10x

Increased quality
• by as much as 10x

Decreased cost
• by as much as 60%

Decreased labor needs
• by as much as 87%

Decreased time to market (to field, to launch...)
• by as much as 98%

Ability to move into new markets
• in months, not years

Note: Each of the above is based on an individual, documented product line effort.
### Costs Of A Software Product Line

<table>
<thead>
<tr>
<th>Core Assets</th>
<th>Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Architecture</strong></td>
<td>Must support variation inherent in the product line</td>
</tr>
<tr>
<td><strong>Software Components</strong></td>
<td>Must be designed to be general without a loss of performance; must build in support for variation points</td>
</tr>
<tr>
<td><strong>Test Plans, Test Cases, Test Data</strong></td>
<td>Must consider variation points and multiple instances of the product line</td>
</tr>
<tr>
<td><strong>Business Case and Market Analysis</strong></td>
<td>Must address a family of software products, not just one product</td>
</tr>
<tr>
<td><strong>Project Plans</strong></td>
<td>Must be generic or be made extensible to accommodate product variations</td>
</tr>
<tr>
<td><strong>Tools and Processes</strong></td>
<td>Must be more robust</td>
</tr>
<tr>
<td><strong>People, Skills, Training</strong></td>
<td>Must involve training and expertise centered around the assets and procedures associated with the product line</td>
</tr>
</tbody>
</table>
Economics Of Product Lines

*Software Product-Line Engineering: A Family-Based Software Development Process*
Economics Of Product Lines

*Software Product-Line Engineering: A Family-Based Software Development Process*
Today’s Session

Introduction

*Product Line Concepts*

- What
- Why
- *How*

Conclusion
Necessary Changes

The product line architecture is central to success.
Why Is Software Architecture Important?

Represents *earliest* design decisions

- hardest to change
- most critical to get right
- communication vehicle among stakeholders

*First* design artifact addressing

- performance
- modifiability
- reliability
- security

Key to systematic *reuse*

- transferable, reusable abstraction

The *right architecture* paves the way for system *success.*
The *wrong architecture* usually spells some form of *disaster.*
Product Line Practice

Contexts for product lines vary widely, based on:

- nature of products
- nature of market or mission
- business goals
- organizational infrastructure
- workforce distribution
- process discipline
- artifact maturity

But there are universal essential activities and practices.
The SEI Framework for Software Product Line Practice℠

The SEI Framework for Software Product Line Practice is a conceptual framework that describes the essential activities and twenty-nine practice areas necessary for successful software product lines.

The Framework, originally conceived in 1998, is evolving based on the experience and information provided by the community.

Version 4.0 – in *Software Product Lines: Practices and Patterns*

Version 5.0 –

℠ Framework for Software Product Line Practice is a service mark of Carnegie Mellon University.
SEI Information Sources

Case studies, experience reports, and surveys

Workshops and conferences

Applied research

Collaborations with customers on actual product lines
The Three Essential Activities

Core Asset Development

Product Development

Management
Core Asset Development

Product Constraints
Production Constraints
Production Strategy
Preexisting Assets

Core Asset Development

Product Line Scope
Core Asset Base
Production Plan

Management
Attached Processes

Core Asset Development

Attached Process

Core Asset Base

Production Plan

Management

Core Assets
Product Line Production Plan

Production Plan

Production Process
△ + △ + △ + △ +

Project Details

Production Method

Product Constraints
Production Constraints
Production Strategy
Management
Management

Management at multiple levels plays a critical role in the successful product line practice by

- achieving the right organizational structure
- allocating resources
- coordinating and supervising
- providing training
- rewarding employees appropriately
- developing and communicating an acquisition strategy
- managing external interfaces
- creating and implementing a product line adoption plan
- launching and institutionalizing the approach in a manner appropriate to the organization
Managing A Software Product Line Requires Leadership

A key role for software product line management is that of champion. A champion must

• set and maintain the vision
• ensure that the appropriate goals and measures are in place
• “sell” the product line up and down the chain
• sustain morale
• deflect potential derailments
• solicit feedback and continuously improve the approach
Essential Product Line Activities

Each of these is essential, as is the blending of all three.
Different Approaches - 1

**Proactive:** Develop the core assets first.
- Develop the scope first and use it as a “mission” statement.
- Products come to market quickly with minimum code writing.
- Requires upfront investment and predictive knowledge

**Reactive:** Start with one or more products.
- From them, generate the product line core assets and then future products; the scope evolves more dramatically.
- Much lower cost of entry
- The architecture and other core assets must be robust, extensible, and appropriate to future product line needs.
Different Approaches - 2

**Incremental:** In either a reactive or proactive approach, it is possible to develop the core asset base in stages, while planning from the beginning to develop a product line.

- Develop part of the core asset base, including the architecture and some of the components.
- Develop one or more products.
- Develop part of the rest of the core asset base.
- Develop more products.
- Evolve more of the core asset base.
- …
## Alternate Terminology

<table>
<thead>
<tr>
<th>Our Terminology</th>
<th>Alternate Terminology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Line</td>
<td>Product Family</td>
</tr>
<tr>
<td>Software Core Assets</td>
<td>Platform</td>
</tr>
<tr>
<td>Business Unit</td>
<td>Product Line</td>
</tr>
<tr>
<td>Product</td>
<td>Customization</td>
</tr>
<tr>
<td>Core Asset Development</td>
<td>Domain Engineering</td>
</tr>
<tr>
<td>Product Development</td>
<td>Application Engineering</td>
</tr>
</tbody>
</table>
Driving The Essential Activities

Beneath the level of the essential activities are essential practices that fall into practice areas.
A *practice area* is a body of work or a collection of activities that an organization must master to successfully carry out the essential work of a product line.
Practice Area Descriptions

Each practice area is described with
- an introductory description
- aspects that are peculiar to product lines
- its application to core asset development
- its application to product development
- example practices
- associated risks
- further reading
Three Categories Of Practice Areas

- Organizational Management Practice Areas
  - Enable and orchestrate

- Technical Management Practice Areas
  - Manage and support

- Software Engineering Practice Areas
## PRECINCT AREAS

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

<table>
<thead>
<tr>
<th>Software Engineering</th>
<th>Technical Management</th>
<th>Organizational Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture Definition</td>
<td>Configuration Management</td>
<td>Building a Business Case</td>
</tr>
<tr>
<td>Architecture Evaluation</td>
<td><strong>Measurement and Tracking</strong></td>
<td>Customer Interface Management</td>
</tr>
<tr>
<td>Component Development</td>
<td>Make/Buy/Mine/Commission Analysis</td>
<td>Developing an Acquisition Strategy</td>
</tr>
<tr>
<td>Using Externally Available Software</td>
<td><strong>Process Discipline</strong></td>
<td>Funding</td>
</tr>
<tr>
<td>Mining Existing Assets</td>
<td>Scoping</td>
<td>Launching and Institutionalizing</td>
</tr>
<tr>
<td>Requirements Engineering</td>
<td>Technical Planning</td>
<td>Market Analysis</td>
</tr>
<tr>
<td>Software System Integration</td>
<td>Technical Risk Management</td>
<td>Operations</td>
</tr>
<tr>
<td>Testing</td>
<td>Tool Support</td>
<td>Organizational Planning</td>
</tr>
<tr>
<td>Understanding Relevant Domains</td>
<td></td>
<td>Organizational Risk Management</td>
</tr>
</tbody>
</table>

**Key:**

- **New Name and Substantial Change**
### PRACTICE AREAS

<table>
<thead>
<tr>
<th>Software Engineering</th>
<th>Technical Management</th>
<th>Organizational Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture Definition</td>
<td>Configuration Management</td>
<td>Building a Business Case</td>
</tr>
<tr>
<td>Architecture Evaluation</td>
<td>Make/Buy/Mine/Commission Analysis</td>
<td>Customer Interface Management</td>
</tr>
<tr>
<td>Component Development</td>
<td>Measurement and Tracking</td>
<td>Developing an Acquisition Strategy</td>
</tr>
<tr>
<td>Mining Existing Assets</td>
<td>Process Discipline</td>
<td>Funding</td>
</tr>
<tr>
<td>Requirements Engineering</td>
<td>Scoping</td>
<td>Launching and Institutionalizing</td>
</tr>
<tr>
<td>Software System Integration</td>
<td>Technical Planning</td>
<td>Market Analysis</td>
</tr>
<tr>
<td>Testing</td>
<td>Technical Risk Management</td>
<td>Operations</td>
</tr>
<tr>
<td>Understanding Relevant Domains</td>
<td>Tool Support</td>
<td>Organizational Planning</td>
</tr>
<tr>
<td>Using Externally Available Software</td>
<td></td>
<td>Organizational Risk Management</td>
</tr>
<tr>
<td>Key:</td>
<td>New Name and Substantial Change</td>
<td>Structuring the Organization</td>
</tr>
<tr>
<td></td>
<td>Substantial Change</td>
<td>Technology Forecasting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Training</td>
</tr>
</tbody>
</table>
Dilemma: How Do You Apply The 29 Practice Areas?

Organizations still have to figure out how to put the practice areas into play.

*Twenty-nine is a big number.*
Help To Make It Happen

<table>
<thead>
<tr>
<th>PRACTICE AREAS</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Engineering</td>
<td>Technical Management</td>
<td>Organizational Management</td>
</tr>
</tbody>
</table>

GUIDANCE

- Case Studies
- Patterns
- Probe
- Curriculum
Case Studies

**CelsiusTech** – CMU/SEI-96-TR-016
http://www.sei.cmu.edu/publications/documents/01.reports/96.tr.016.html

**Cummins, Inc. Software Product Lines: Practices and Patterns**

**Market Maker Software Product Lines: Practices and Patterns**

**NRO/Raytheon** – CMU/SEI-2001-TR-030
http://www.sei.cmu.edu/publications/documents/01.reports/02tr030.html

**NUWC** – CMU/SEI-2002-TN-018
http://www.sei.cmu.edu/publications/documents/02.reports/02tn018.html

**Salion, Inc.** – CMU/SEI-2002-TR-038
http://www.sei.cmu.edu/publications/documents/02.reports/02tr038.html

**U.S. Army** – CMU/SEI-2005-TR-019
http://www.sei.cmu.edu/publications/documents/05.reports/05tr019.html
Help To Make It Happen

ESSENTIAL ACTIVITIES

Core Asset Development

Product Development

Management

PRACTICE AREAS

Software Engineering | Technical Management | Organizational Management

GUIDANCE

Case Studies

Patterns

Probe

Curriculum
Patterns Can Help

Patterns are a way of expressing common context and problem-solution pairs.

Patterns have been found to be useful in building architecture, economics, software architecture, software design, software implementation, process improvement, and others.

Patterns assist in effecting a divide and conquer approach.
Software Product Line Practice Patterns

- **Pattern**
  - **Context**: Organizational Situation
  - **Problem**: What part of a product line effort needs to be accomplished
  - **Solution**: Grouping of practice areas
    Relations among these practice areas (and/or groups if there is more than one)
What To Build Pattern - 1

Name:
The *What to Build* pattern helps an organization determine what products ought to be in its software product line – what products to build.

Context:
An organization has decided to field a software product line and knows the general product area for the set of products.

Problem:
To determine what products should be included in the product line

Solution:
Determining what to build requires information related to the product area, technology, and market; the business justification; and the process for describing the set of products to be included in the product line.
What To Build Pattern - 2

Understanding Relevant Domains

Market Analysis

Technology Forecasting

Scoping

Domain Models

Product Set

Market Climate

Technology Predictions

Building a Business Case

Justification

Product Set

Product Line Scope

Business Case
## Current Set Of Patterns

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly Line</td>
<td></td>
</tr>
<tr>
<td>Cold Start</td>
<td>Warm Start</td>
</tr>
<tr>
<td>Curriculum</td>
<td></td>
</tr>
<tr>
<td>Each Asset</td>
<td>Each Asset Apprentice</td>
</tr>
<tr>
<td></td>
<td>Evolve Each Asset</td>
</tr>
<tr>
<td>Essentials Coverage</td>
<td></td>
</tr>
<tr>
<td>Factory</td>
<td>Adoption Factory</td>
</tr>
<tr>
<td>In Motion</td>
<td></td>
</tr>
<tr>
<td>Monitor</td>
<td></td>
</tr>
<tr>
<td>Process</td>
<td>Process Improvement</td>
</tr>
<tr>
<td>Product Builder</td>
<td>Product Gen</td>
</tr>
<tr>
<td>Product Parts</td>
<td>Green Field</td>
</tr>
<tr>
<td></td>
<td>Barren Field</td>
</tr>
<tr>
<td></td>
<td>Plowed Field</td>
</tr>
<tr>
<td>What to Build</td>
<td>Analysis</td>
</tr>
<tr>
<td></td>
<td>Forced March</td>
</tr>
</tbody>
</table>
Help To Make It Happen

ESSENTIAL ACTIVITIES

PRACTICE AREAS

| Software Engineering | Technical Management | Organizational Management |

GUIDANCE

Case Studies
Patterns
Probe
Curriculum
What Is An SEI Product Line Technical Probe (PLTP)?

The SEI PLTP is a method for examining an organization’s readiness to adopt or ability to succeed with a software product line approach.

• It is a diagnostic tool based on the SEI Framework for Software Product Line Practice.

• The 29 practice areas are the basis of data collection and analysis.
Help To Make It Happen

PRACTICE AREAS

<table>
<thead>
<tr>
<th>Software Engineering</th>
<th>Technical Management</th>
<th>Organizational Management</th>
</tr>
</thead>
</table>

GUIDANCE

Case Studies | Patterns | Probe | Curriculum
# The SEI Software Product Line Curriculum

## Three Certificate Programs

<table>
<thead>
<tr>
<th>Five Courses</th>
<th>Software Product Line Professional</th>
<th>PLTP Team Member</th>
<th>PLTP Leader</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Product Lines</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Adopting Software Product Lines</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Developing Software Product Lines</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PLTP Team Training</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PLTP Leader Training</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLTP Lead Observation</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

✓: course required to receive certificate
The Entire Picture

**PRACTICE AREAS**

| Software Engineering | Technical Management | Organizational Management |

**GUIDANCE**

- Case Studies
- Patterns
- Probe
- Curriculum

**ADOPTION FACTORY**

---

Software Product Lines
Linda Northrop
© 2008 Carnegie Mellon University
The SEI Adoption Factory Pattern

<table>
<thead>
<tr>
<th>Focus Areas</th>
<th>Phases</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>Establish Context</td>
<td>Establish Production Capability</td>
<td>Operate Product Line</td>
<td></td>
</tr>
<tr>
<td>Process</td>
<td></td>
<td>Each Asset</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Organization</td>
<td></td>
<td>What to Build</td>
<td>Product Parts</td>
<td>Product Builder</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Process Discipline</td>
<td>Assembly Line</td>
<td></td>
</tr>
</tbody>
</table>

- What to Build 
- Process Discipline 
- Assembly Line 
- Product Parts 
- Product Builder 
- Cold Start 
- In Motion 
- Monitor 

Informs and information flow 
Supports
## Associated Practice Areas

<table>
<thead>
<tr>
<th>Establish Context</th>
<th>Establish Production Capability</th>
<th>Operate Product Line</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Product</strong></td>
<td>• Marketing Analysis</td>
<td>• Requirements Engineering</td>
</tr>
<tr>
<td></td>
<td>• Understanding Relevant Domains</td>
<td>• Architecture Definition</td>
</tr>
<tr>
<td></td>
<td>• Technology Forecasting</td>
<td>• Architecture Evaluation</td>
</tr>
<tr>
<td></td>
<td>• Building a Business Case</td>
<td>• Mining Existing Assets</td>
</tr>
<tr>
<td></td>
<td>• Scoping</td>
<td>• Component Development</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Using Externally Available Software</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Software System Integration</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Testing</td>
</tr>
<tr>
<td><strong>Process</strong></td>
<td>• Process Discipline</td>
<td>• Make/Buy/Mine/Commission</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Configuration Management</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Tool Support</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Measurement and Tracking</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Technical Planning</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Technical Risk Management</td>
</tr>
<tr>
<td><strong>Organization</strong></td>
<td>• Launching and Institutionalizing</td>
<td>• Launching and Institutionalizing</td>
</tr>
<tr>
<td></td>
<td>• Funding</td>
<td>• Funding</td>
</tr>
<tr>
<td></td>
<td>• Structuring the Organization</td>
<td>• Structuring the Organization</td>
</tr>
<tr>
<td></td>
<td>• Operations</td>
<td>• Operations</td>
</tr>
<tr>
<td></td>
<td>• Organizational Planning</td>
<td>• Organizational Planning</td>
</tr>
<tr>
<td></td>
<td>• Customer Interface Management</td>
<td>• Customer Interface Management</td>
</tr>
<tr>
<td></td>
<td>• Organizational Risk Management</td>
<td>• Organizational Risk Management</td>
</tr>
<tr>
<td></td>
<td>• Developing an Acquisition Strategy</td>
<td>• Developing an Acquisition Strategy</td>
</tr>
<tr>
<td></td>
<td>• Training</td>
<td>• Training</td>
</tr>
</tbody>
</table>

- Make/Buy/Mine/Commission
- Configuration Management
- Tool Support
- Measurement and Tracking
- Technical Planning
- Technical Risk Management
- Measurement and Tracking
- Technical Risk Management
- Organizational Risk Management
- Customer Interface Management
- Organizational Planning
Today’s Session

Introduction

Product Line Concepts

• What
• Why
• How

Conclusion
In A Nutshell

Software product lines epitomize the concept of strategic, planned reuse. The product line concept is about more than a new technology. It is a new way of doing one’s software business. There are essential product line activities and practices areas as well as product line patterns to make the move to product lines more manageable.

PRACTICE AREAS

| Software Engineering | Technical Management | Organizational Management |

Core Asset Development
Product Development
Management

ESSENTIAL ACTIVITIES
What’s Different About Reuse With Software Product Lines?

- Business dimension
- Iteration
- Architecture focus
- Preplanning
- Process and product connection
At The Heart Of Successful Product Lines

- A pressing need that addresses the heart of the business
- Long and deep domain experience
- A legacy base from which to build
- Architectural excellence
- Process discipline
- Management commitment
- Loyalty to the product line as a single entity
Product Line Adoption and Institutionalization

Innovators and early adopters demonstrated the feasibility and the benefits of software product lines:

- CelsiusTech
- Cummins, Inc.
- Hewlett-Packard
- Motorola
- Nokia

The SEI and others have tried to lower the adoption barrier by codifying practices, writing case studies, perfecting methods useful in product line approaches, and engendering a software product line community.

Many organizations are now handsomely achieving their business goals using a software product line approach.
Sources of Knowledge
Summary of SEI Contributions

Models and Guidance

- *A Framework for Software Product Line Practice*™
- *Software Product Line Acquisition: A Companion to A Framework for Software Product Line Practice*
- Product line practice patterns
- Product line adoption roadmap
- Pedagogical product line

Methods and Technology

- Product line analysis
- Architecture definition, documentation, evaluation (ATAM®), and recovery
- Mining assets
- Production planning
- Structured Intuitive Method for Product Line Economics (SIMPLE)
- Product Line Technical Probe™ (PLTP™)
- Product Line Quick Look (PLQL)
- Interactive workshops in product line measurement, variability management, product line management
- Prediction-enabled component technology

Book

*Software Product Lines: Practices and Patterns*

Curriculum and Certificate Programs

- Five courses and three certificate programs
- Product Line Executive Seminar

Conferences and Workshops

- SPLC 1, SPLC 2, SPLC 2004; SPLC 2006; Workshops 1997-2008

Technical Reports, publications, and Web site
SEI Transition

Enable others
- Certificate Programs
- Course licensing
- PLTP Leader Certification

Foster Widespread Awareness
- Books
- Reports, articles, papers
- Five-course curriculum
- Executive seminar
- Conferences
- Workshops
- Website

Ensure practicability
- Methods
- Patterns
- Case studies
- Adoption Roadmap
- Acquisition Companion

Assist others
- Product Line Technical Probe
- Product Line Quick Look
- Practice-specific workshops
- Planning workshops

Transition

Software Engineering Institute | Carnegie Mellon
Software Product Lines
Linda Northrop
© 2008 Carnegie Mellon University
Summary

Research in software product lines was inspired by the proven benefits of product line approaches in manufacturing, buoyed by the advent of object and component technology.

The SEI has been a leader in developing a body of knowledge and a set of standard models for software product lines.

Early product line adopters, like Cummins, Inc., are now on second generation product lines that have resulted in even far greater benefits.

Service-oriented and model-driven approaches, as well as developments in collaborative philosophies and environments, are extending the power of product line practice in exciting new ways.
Final Word

If properly managed, the benefits of a product line approach far exceed the costs.
Strategic software reuse through a well-managed product line approach achieves business goals for:

• efficiency
• time to market
• productivity
• quality
• agility
Questions – Now Or Later

Linda Northrop
Director, Product Line Systems Program
Telephone: 412-268-7638
Email: lmn@sei.cmu.edu

U.S. Mail:
Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-3890

World Wide Web:
http://www.sei.cmu.edu/productlines
SEI Fax: 412-268-5758