



**Carnegie Mellon
Software Engineering Institute**

Software Product Lines

**Linda Northrop
Product Line Systems Program**

**Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213**

This work is sponsored by the U.S. Department of Defense.



Carnegie Mellon
Software Engineering Institute

Today's Talk

Introduction

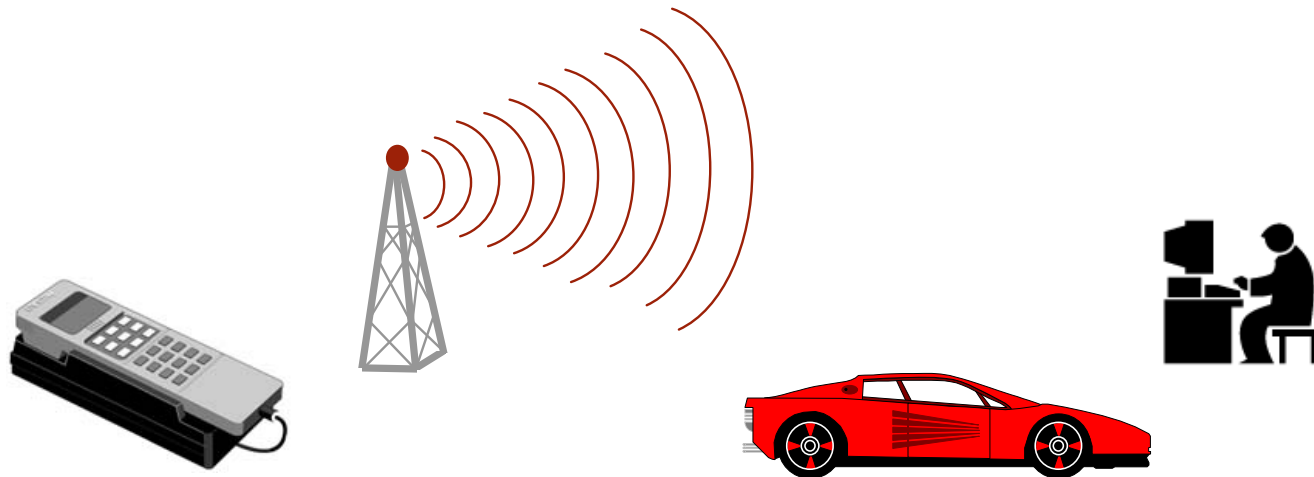
Product Line Concepts

- What
- Why
- How

Conclusion



Business Success Requires Software Prowess



Software pervades every sector.
Software has become the bottom line for many organizations who never envisioned themselves in the software business.



Universal Needs

Deploy new products (services) at a rapid pace

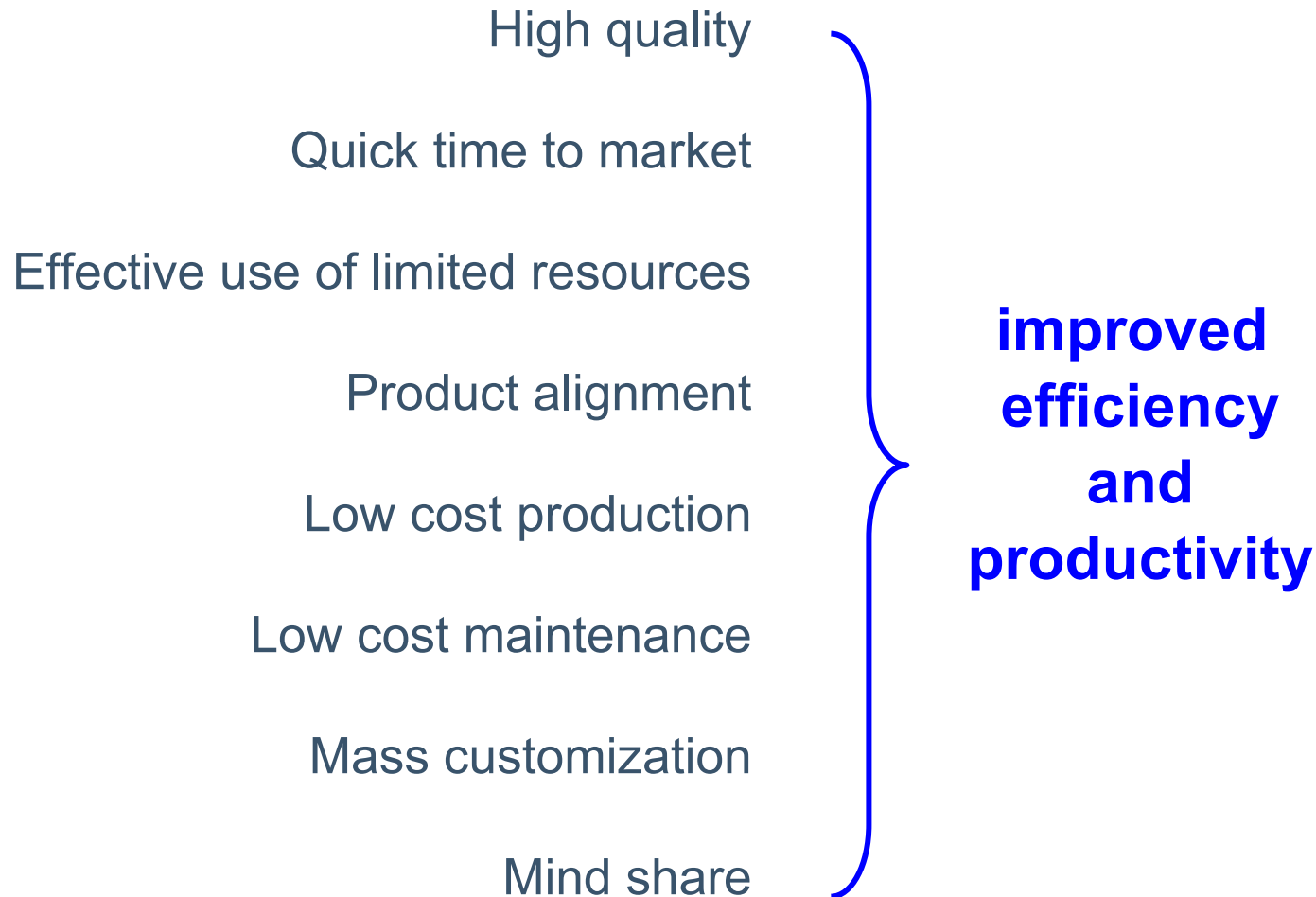
Accommodate a growing demand for new product features across a wide spectrum of feature categories

Exploit a rapidly changing technology base

Gain a competitive edge



Universal Business Goals





The Ultimate Universal Goal



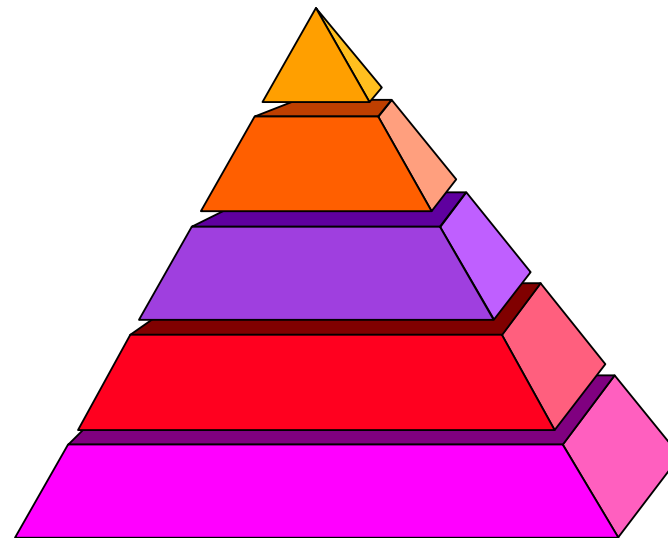


Software (System) Strategies

Process Improvement

Technology Innovation

Reuse





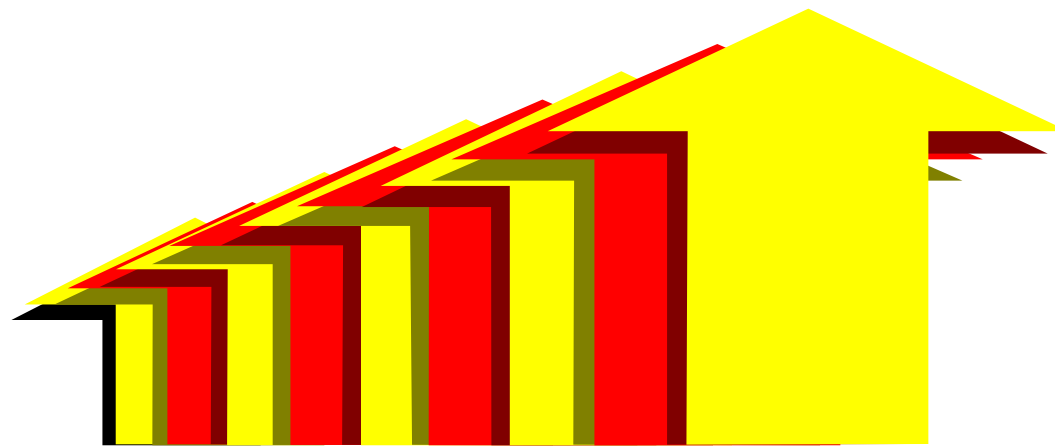
Few Systems Are Unique



Most organizations produce families of similar systems, differentiated by features.



Reuse History



1960's
Subroutines

1970's
Modules

1980's
Objects

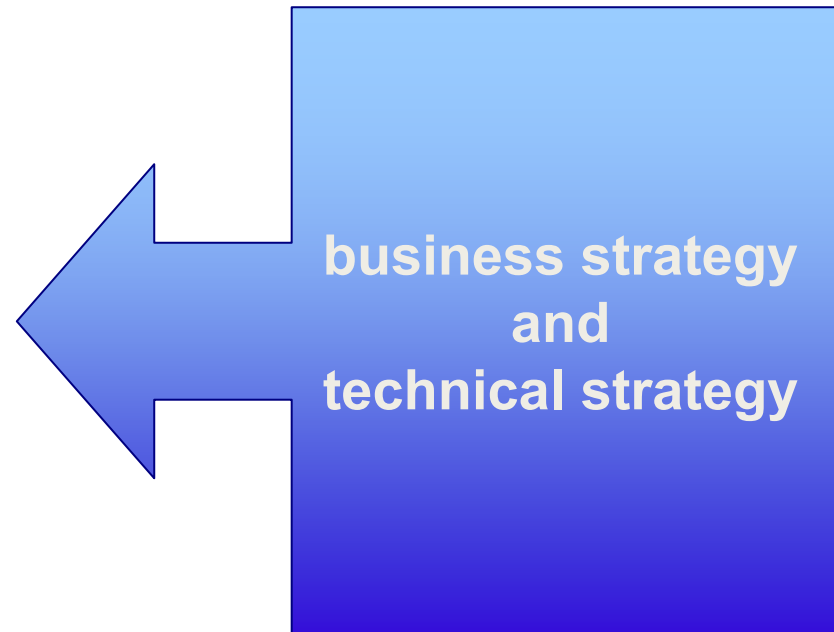
1990's
Components

Focus was small-grained and opportunistic.
Results fell short of expectations.



Imagine Strategic Reuse

**strategic
reuse**

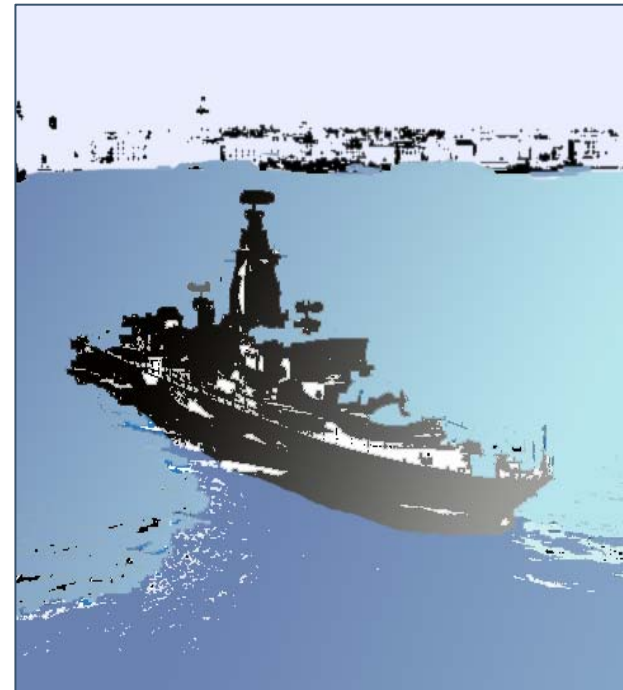




CelsiusTech: Ship System 2000

A family of 55 ship systems

- 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.
- Performance/distribution behavior are known in advance.
- Customer satisfaction is high.
- Hardware-to-software cost ratio changed from 35:65 to 80:20.





Cummins Inc.: Diesel Engine 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

- increased quality by 10X
- incremental build time reduced from months to weeks
- software productivity increased by 7X
- development time and costs decreased by 50%
- decreased product risk

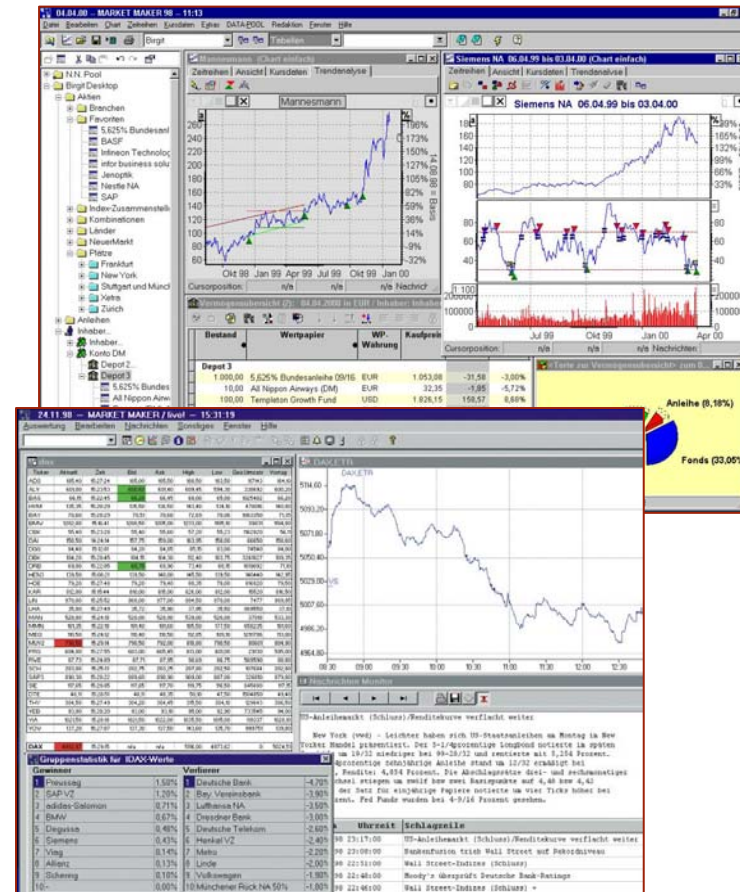




Market Maker GmbH: MERGER

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

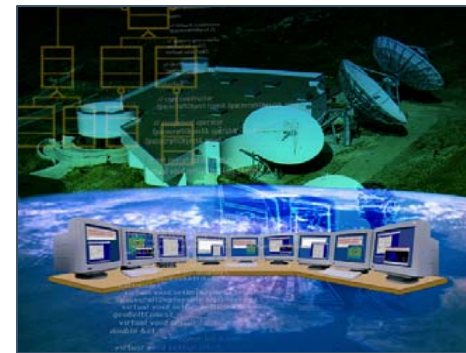
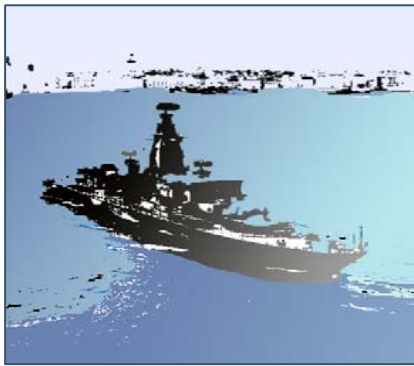
Across products there are

- 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

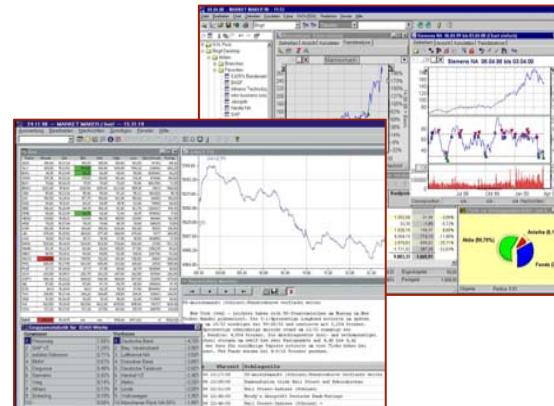




How Did They Do It?

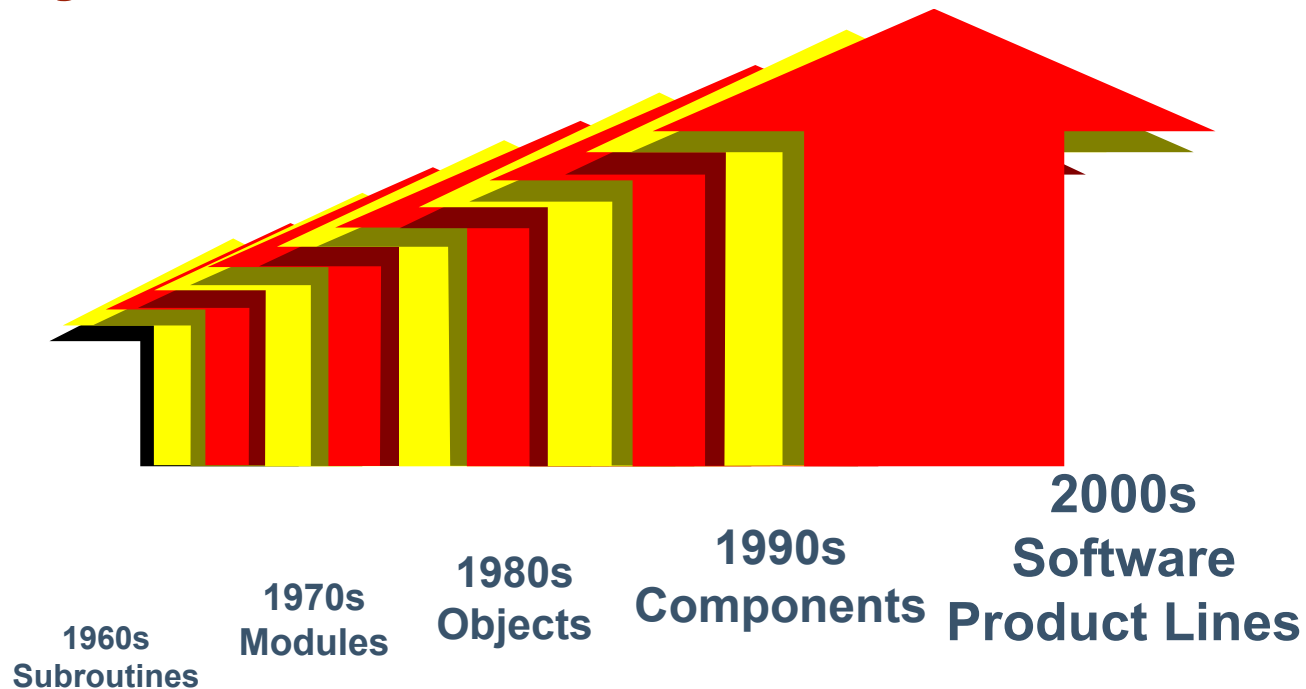


Software Product Lines





Reuse History: From Ad Hoc to Systematic





Carnegie Mellon
Software Engineering Institute

Today's Talk

Introduction

Product Line Concepts

- What
- Why
- How

Conclusion

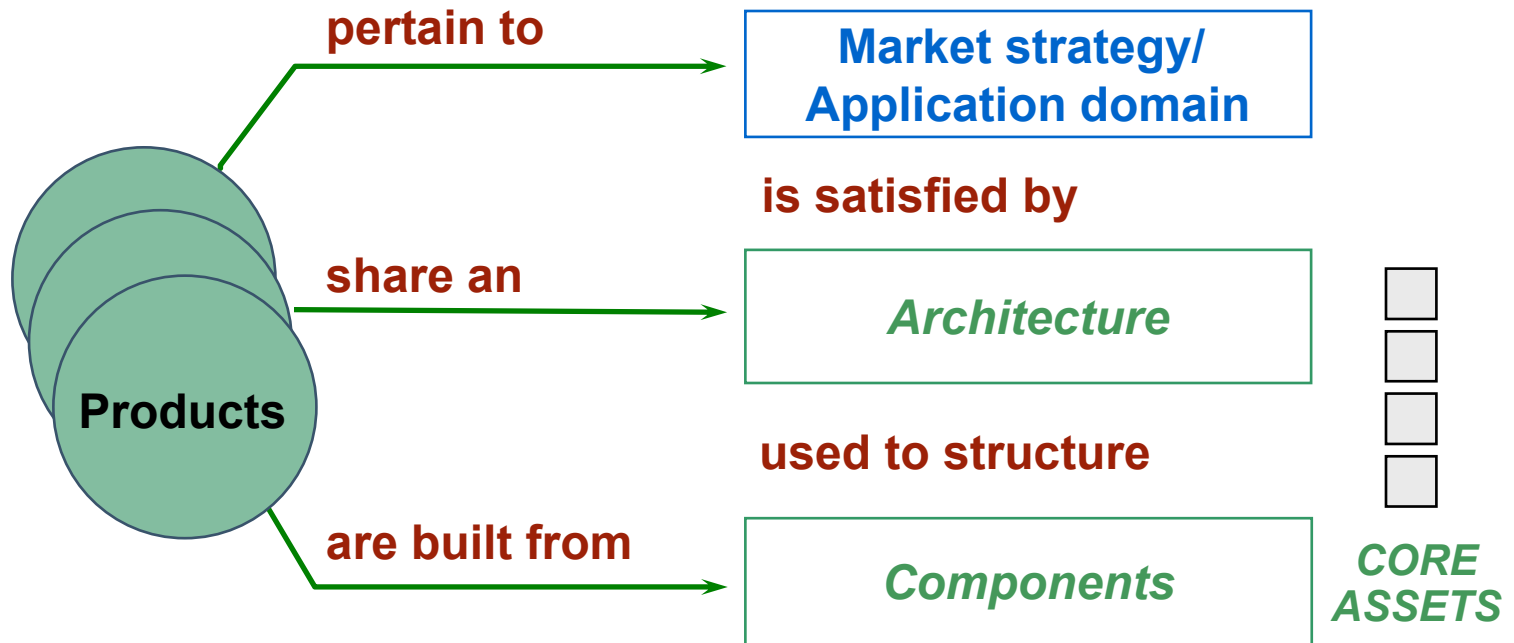


What Is a Software Product Line?

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.



Software Product Lines



Product lines

- take economic advantage of commonality
- bound variability

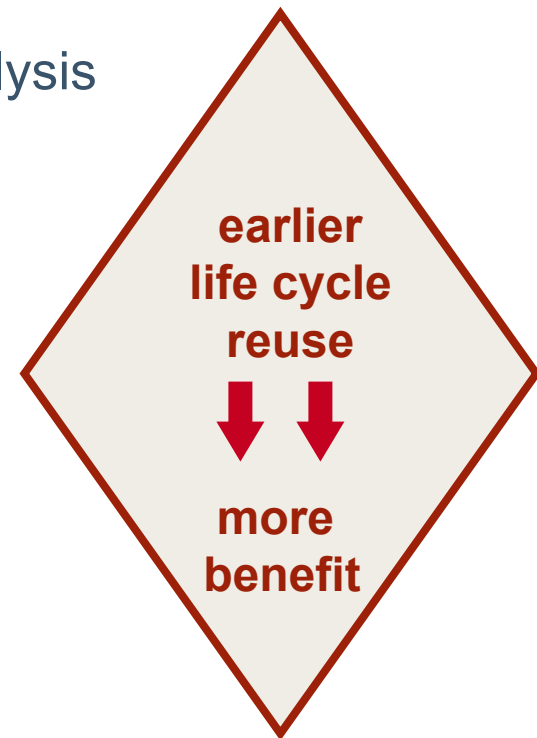


How Do Product Lines Help?

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

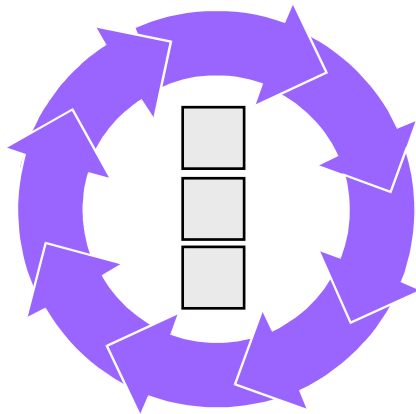
product lines = strategic reuse



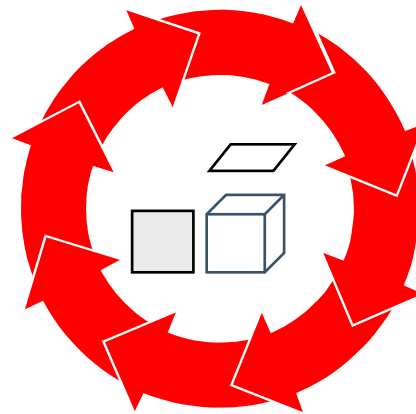


The Key Concepts

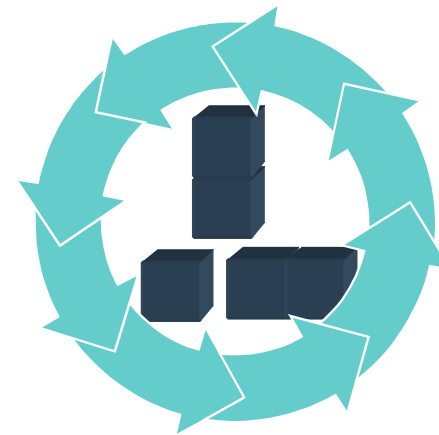
**Use of a core
asset base**



in production

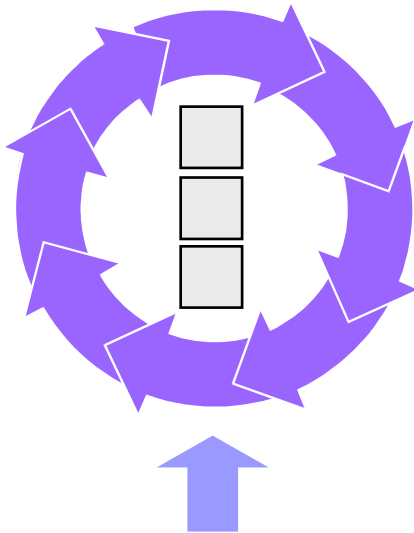


**of a related
set of products**



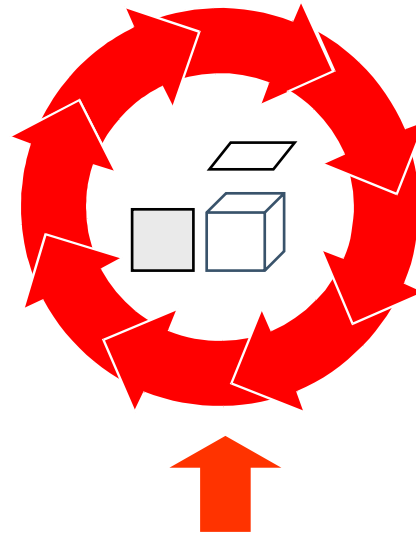
The Key Concepts

Use of a core
asset base



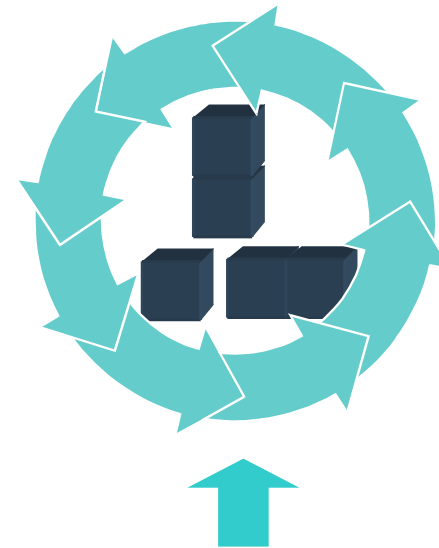
Architecture

in production



Production Plan

of a related
set of products



Scope Definition
Business Case



Software Product Lines Are Not - 1

Fortuitous Small-Grained Reuse

- Reuse libraries containing algorithms, modules, objects, or components
- Benefits depend on
 - software engineer's predisposition to use what is in the library
 - suitability of library contents for particular needs
 - successful adaptation and integration of library units into the rest of the system
- Reuse is not planned, enabled, or enforced nor are results predictable



Software Product Lines Are Not - 2

Single-System Development with Reuse

- Borrowing opportunistically from previous efforts
- Modifying as necessary for the single system only
- Asset base never cultivated

Just Component-Based Development

- Selection of components from an in-house library or the marketplace
- Missing a product line architecture and a production plan as well as management infrastructure



Software Product Lines Are Not - 3

Just a Configurable Architecture

- Involves use of a reference architecture or application framework
- Does not involve the planned reuse of other assets

Versions of Single Products

- Involves sequential release of products over time.
- No simultaneous release/support of multiple products

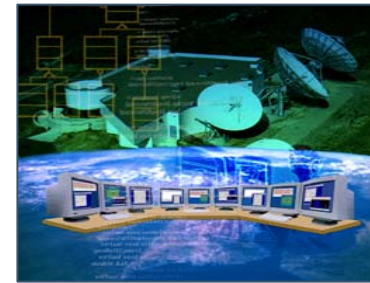
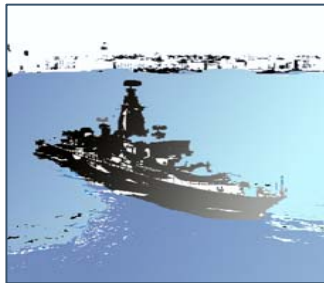
Just a Set of Technical Standards

- Constraints to promote interoperability and to decrease the cost associated with maintenance and support of commercial components
- Does not provide assets and production capability



Product Lines Are

Software product lines involve strategic, planned reuse that yields predictable results.





Commercial Examples

Successful software product lines have been built for families of

- mobile phones
- command and control ship systems
- ground-based spacecraft systems
- avionics systems
- command and control/situation awareness systems
- pagers
- engine control systems
- billing systems
- web-based retail systems
- printers
- consumer electronic products
- acquisition management enterprise systems



Carnegie Mellon
Software Engineering Institute

Today's Talk

Introduction

Product Line Concepts

- What
- Why
- How

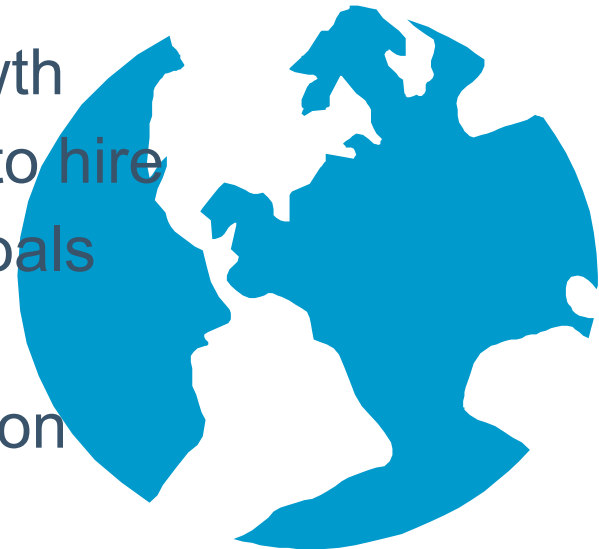
Conclusion



Real World Motivation

Organizations use product line practices to:

- achieve large scale productivity gains
- improve time to market
- maintain market presence
- sustain unprecedented growth
- compensate for an inability to hire
- achieve systematic reuse goals
- improve product quality
- increase customer satisfaction
- enable mass customization
- get control of diverse product configurations





Summary: Organizational Benefits

Improved productivity
by as much as 10x

Decreased time to market (to field, to launch...)
by as much as 10x

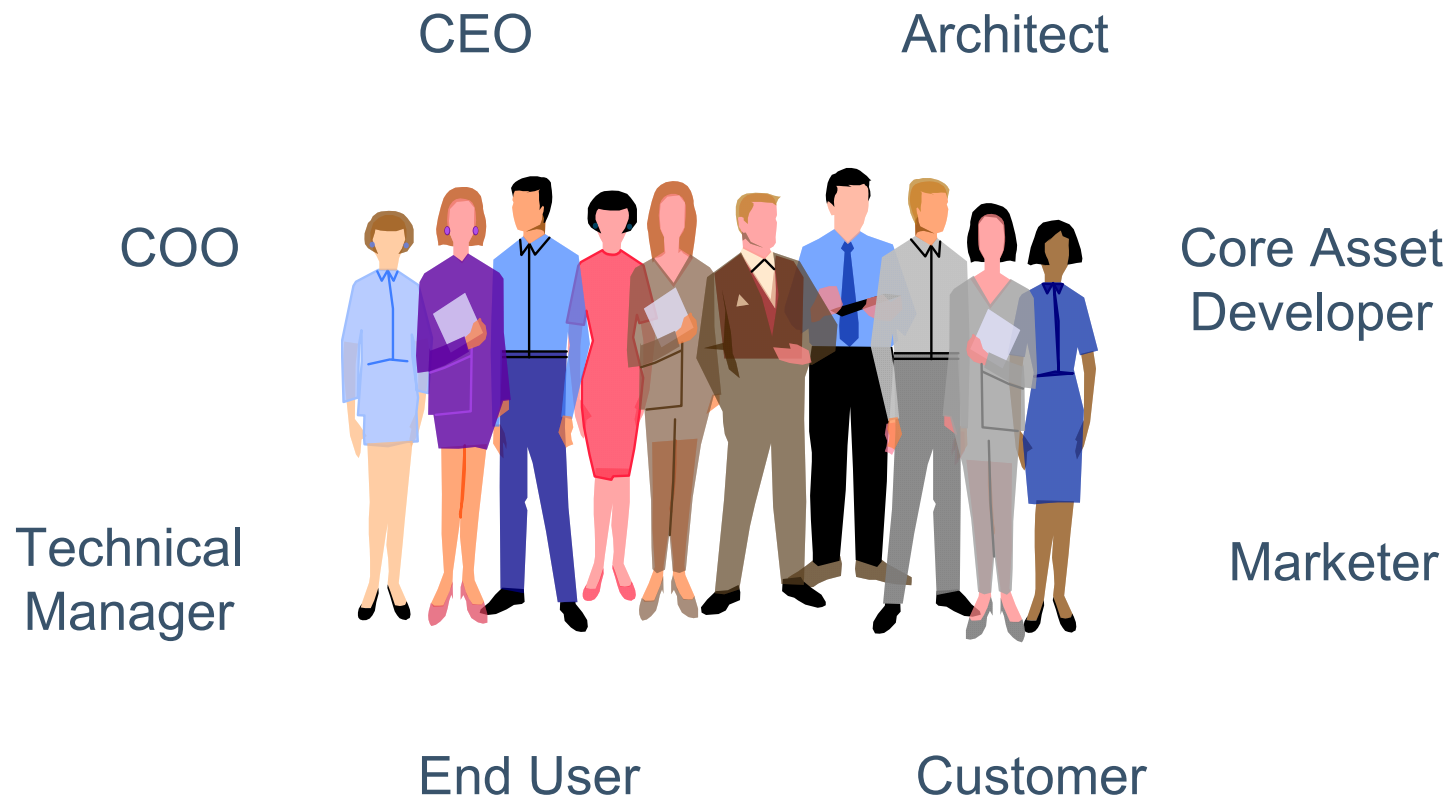
Decreased cost
by as much as 60%

Decreased labor needs
by as much as 10X fewer software developers

Increased quality
by as much as 10X fewer defects

*Product line practice permits predictable
“faster, better, cheaper.”*

Individuals Who Benefit





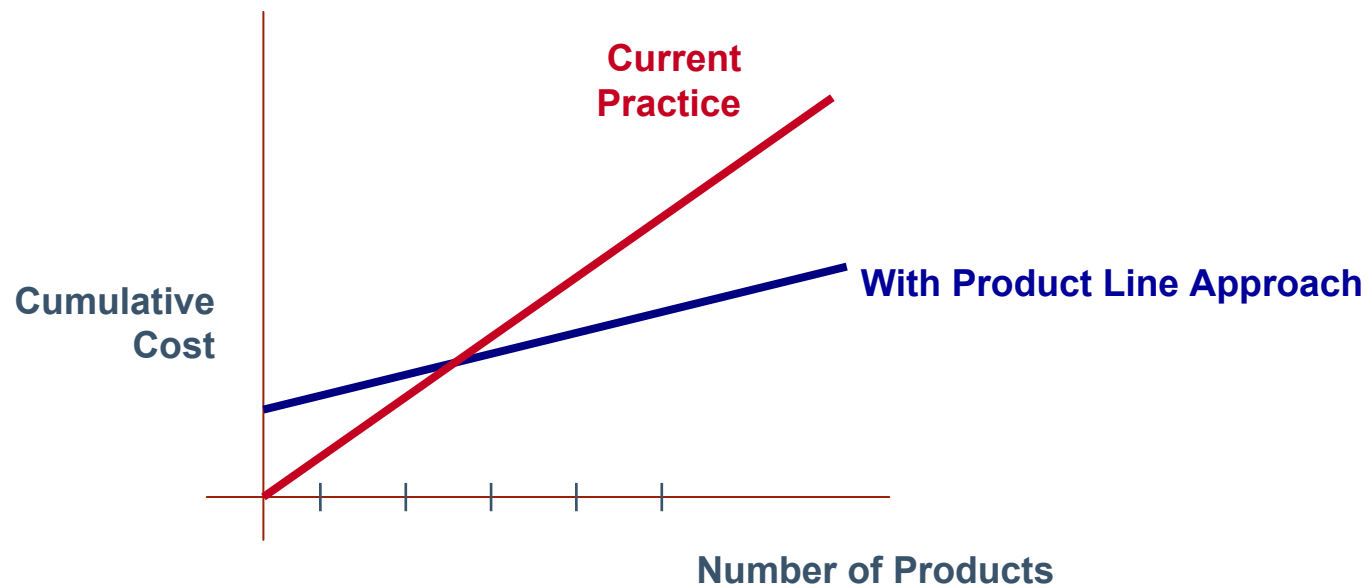
Costs of a Software Product Line

Core Assets	Costs
architecture	must support variation inherent in the product line
software components	must be designed to be general without a loss of performance; must build in support for variation points
test plans, test cases, test data	must consider variation points and multiple instances of the product line
business case and market analysis	must address a family of software products, not just one product
project plans	must be generic or be made extensible to accommodate product variations
tools and processes	must be more robust
people, skills, training	must involve training and expertise centered around the assets and procedures associated with the product line





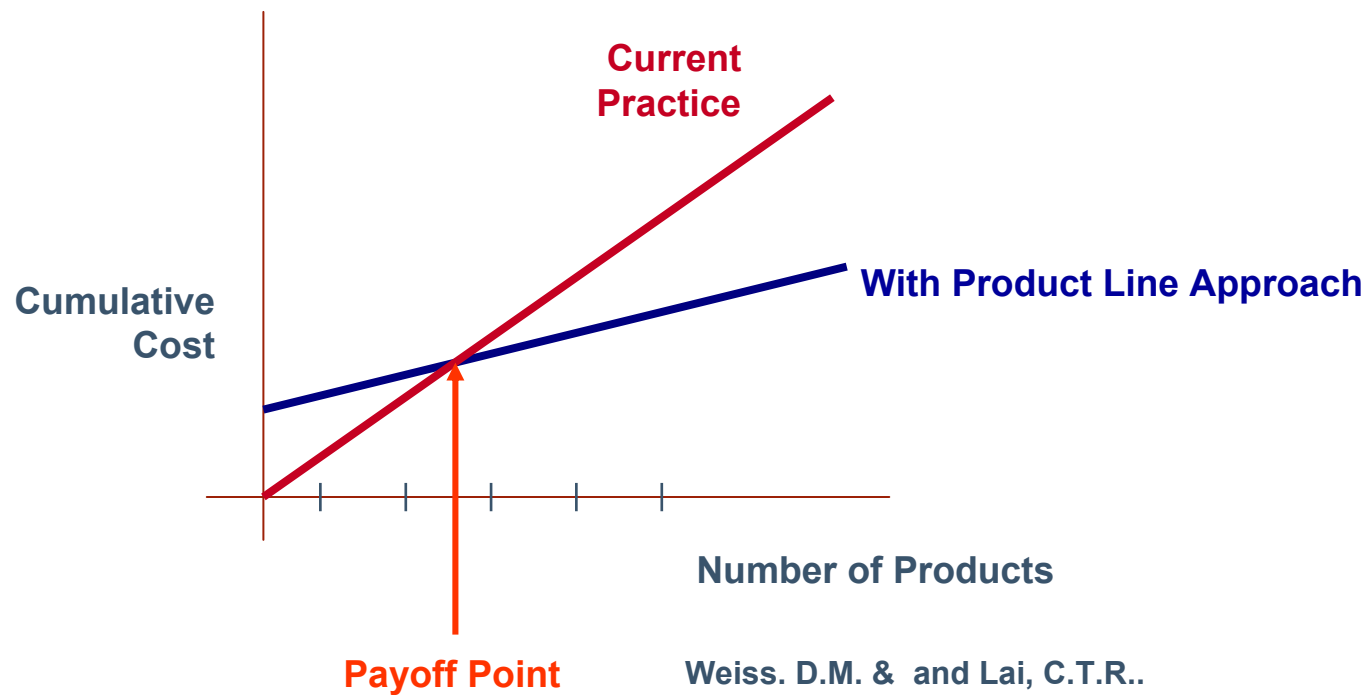
Economics of Product Lines



Weiss, D.M. & Lai, C.T.R..
*Software Product-Line Engineering:
A Family-Based Software Development Process*
Reading, MA: Addison-Wesley, 1999.



Economics of Product Lines



Weiss, D.M. & and Lai, C.T.R..
*Software Product-Line Engineering:
A Family-Based Software
Development Process*
Reading, MA: Addison-Wesley, 1999.



Carnegie Mellon
Software Engineering Institute

Today's Talk

Introduction

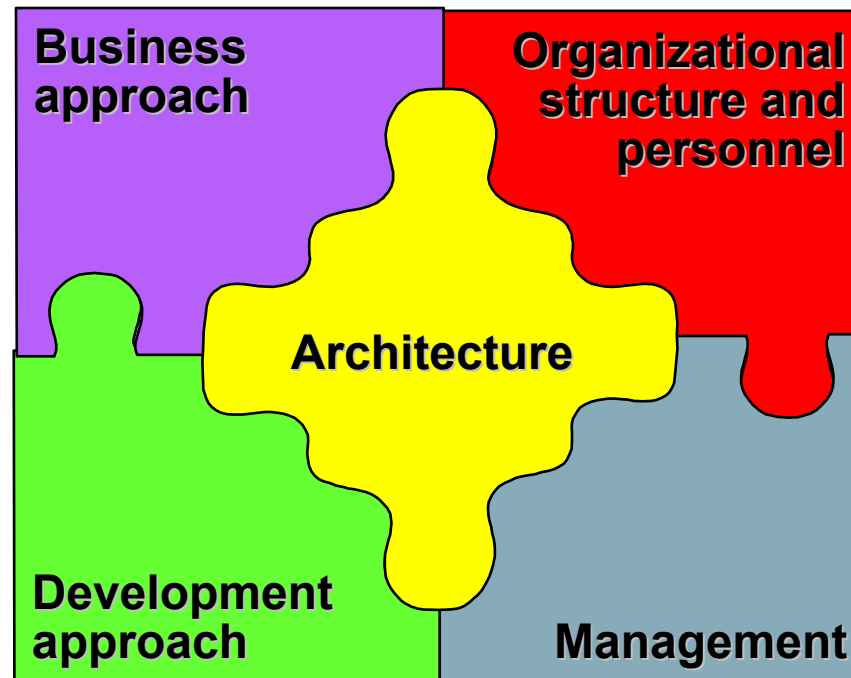
Product Line Concepts

- What
- Why
- How

Conclusion

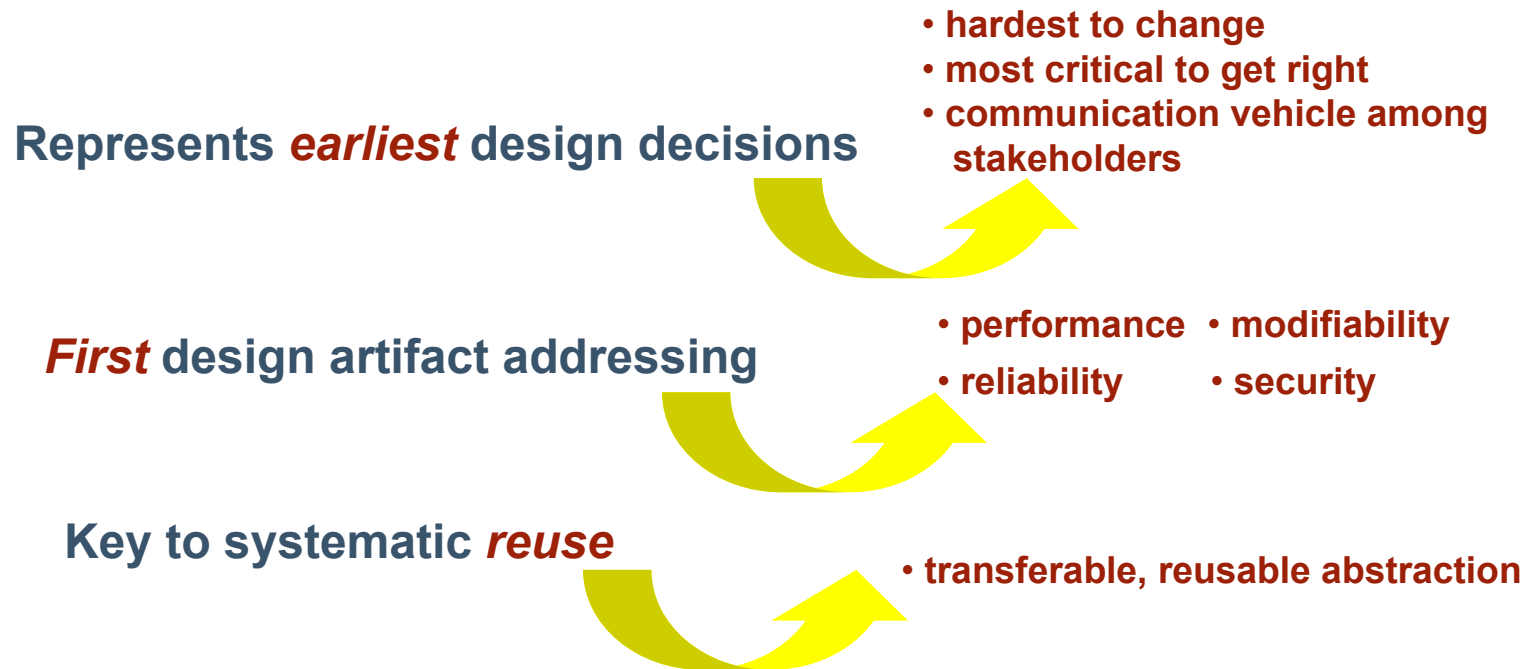


Necessary Changes



The product line architecture is the foundation of everything.

Why is Software Architecture Important?



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.



Carnegie Mellon
Software Engineering Institute

A Framework for Software Product Line PracticeSM

A description of the essential activities and practice areas form a conceptual framework for software product line practice.

This Framework is evolving based on the experience and information provided by the community.

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

Version 4.2 – <http://www.sei.cmu.edu/plp/framework.html>



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





The Nature of the Essential Activities

All three activities are interrelated and highly iterative.

There is no “first” activity.

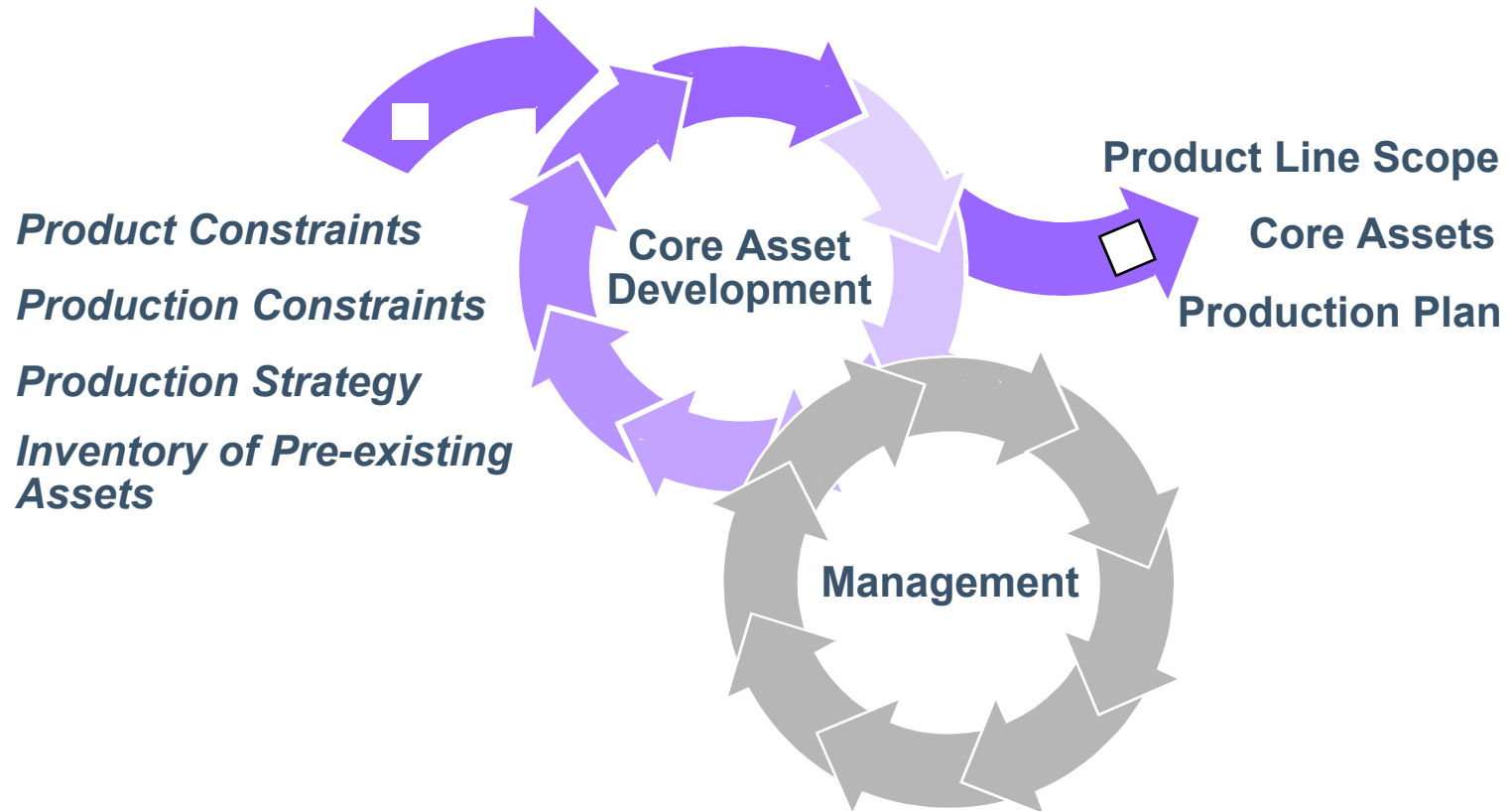
- In some contexts, existing products are mined for core assets.
- In others, core assets may be developed or procured for future use.

There is a strong feedback loop between the core assets and the products.

Strong management at multiple levels is needed throughout.

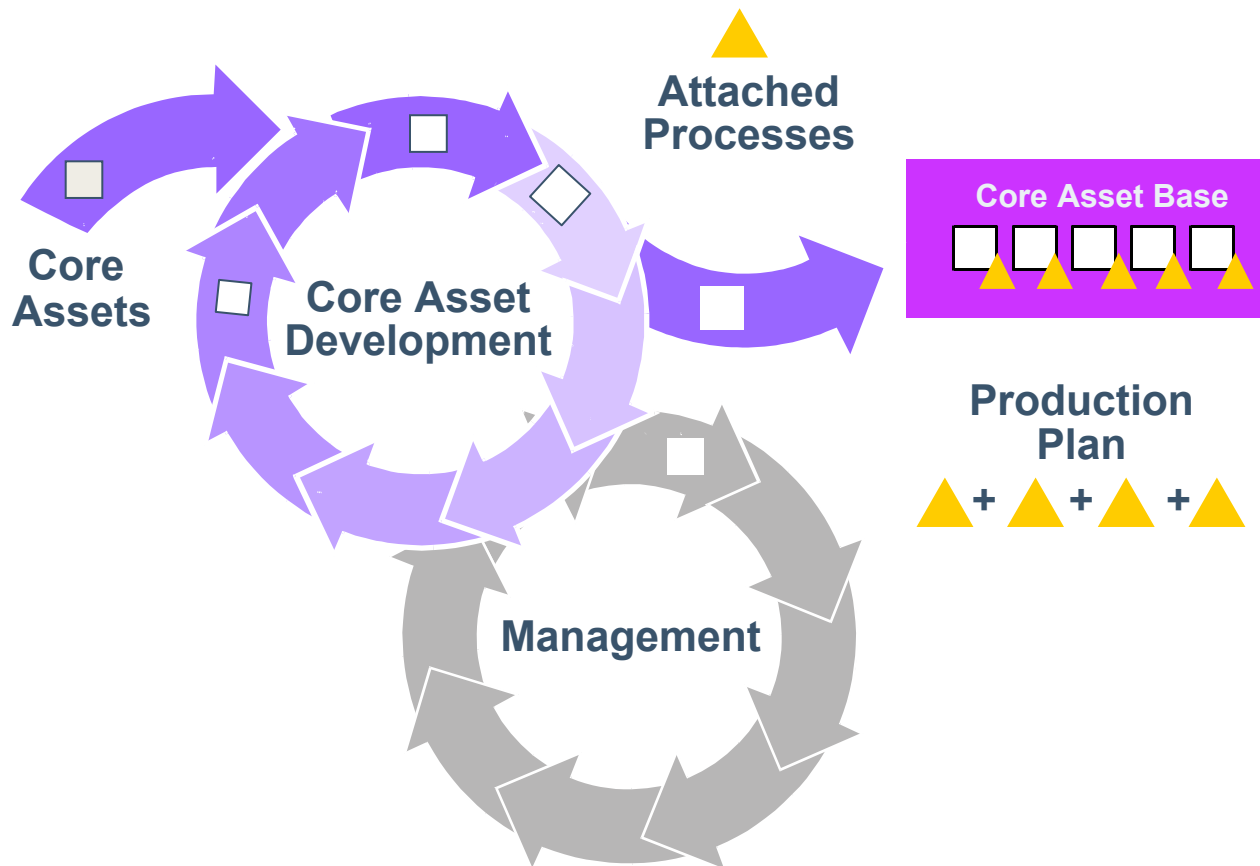


Core Asset Development



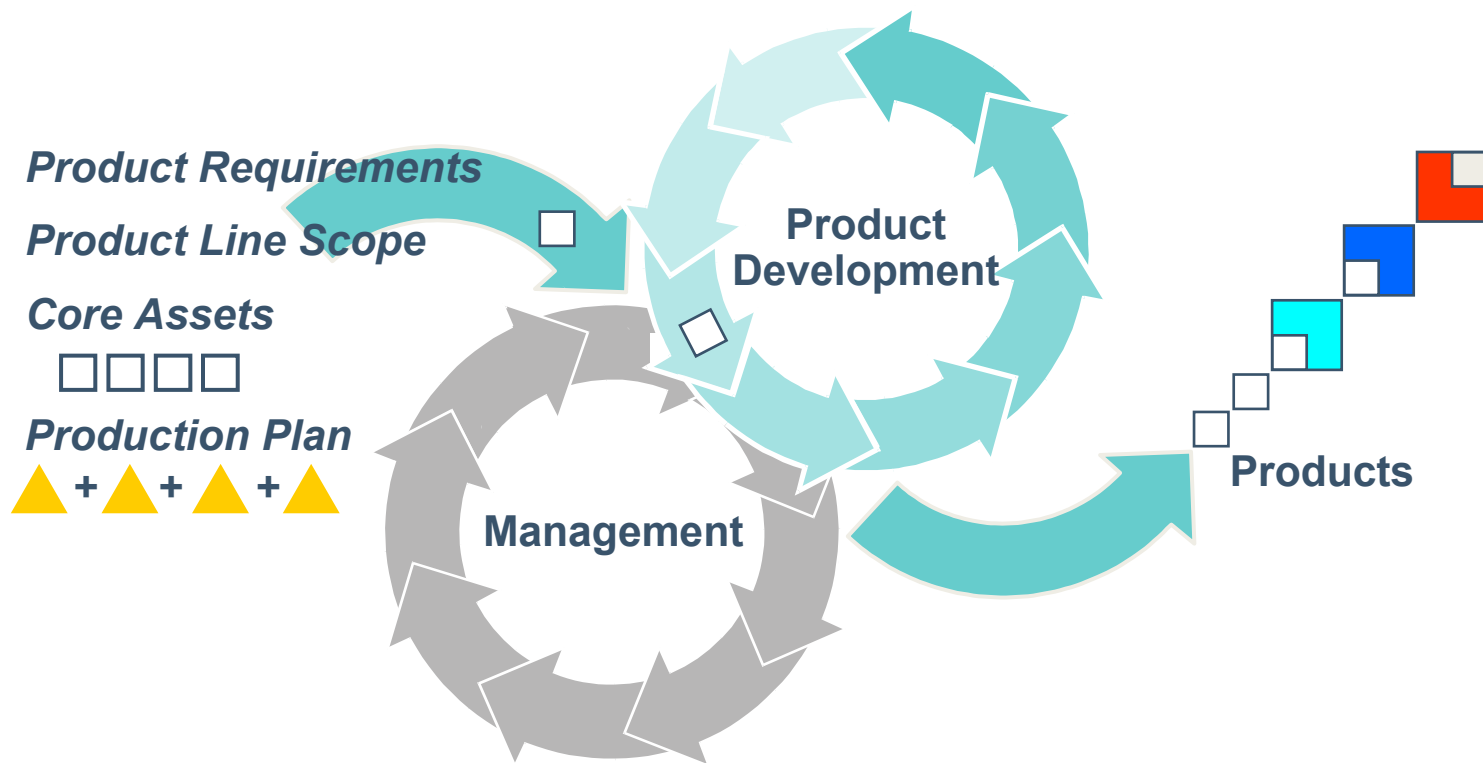


Attached Processes

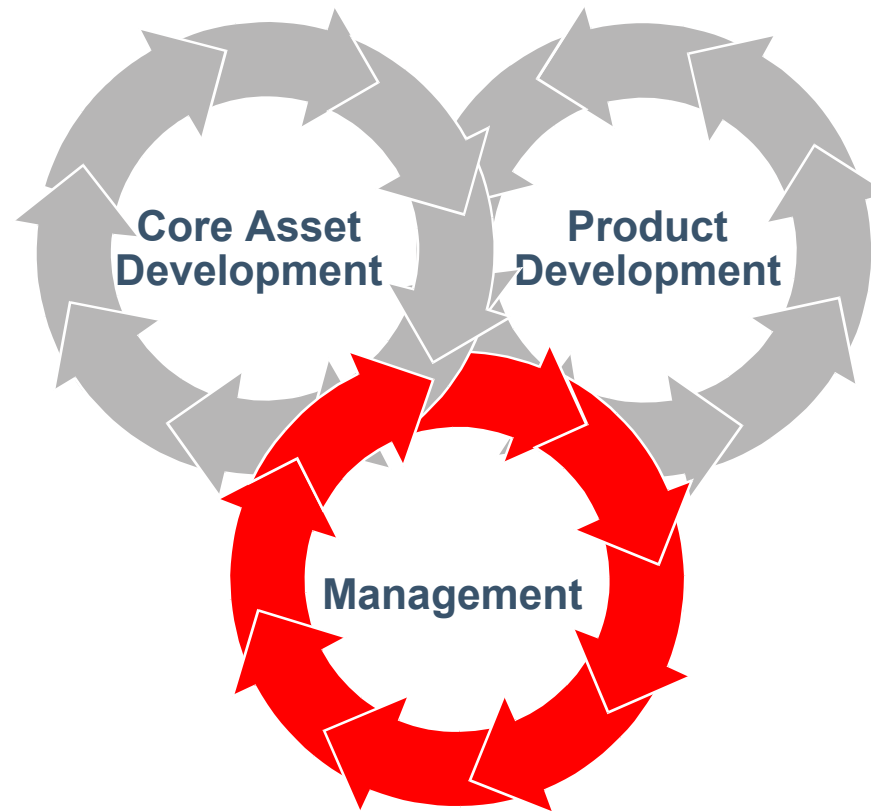




Product Development



Management

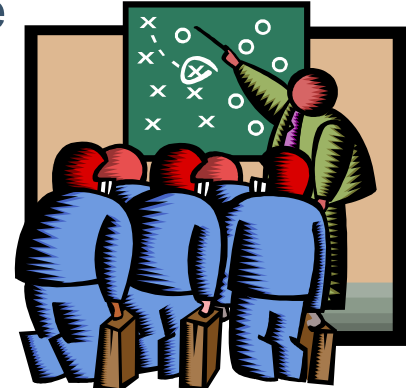




Management

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





Carnegie Mellon
Software Engineering Institute

Managing a Software Product Line Requires Leadership

A key role for a software product line manager is that of champion.

The 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.
- ...



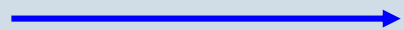
Alternate Terminology

Our Terminology

Product Line



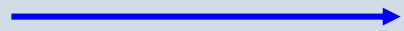
Core Assets



Business Unit



Product



Core Asset Development



Product Development



Alternate Terminology

Product Family

Platform

Product Line

Customization

Domain Engineering

Application Engineering



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.

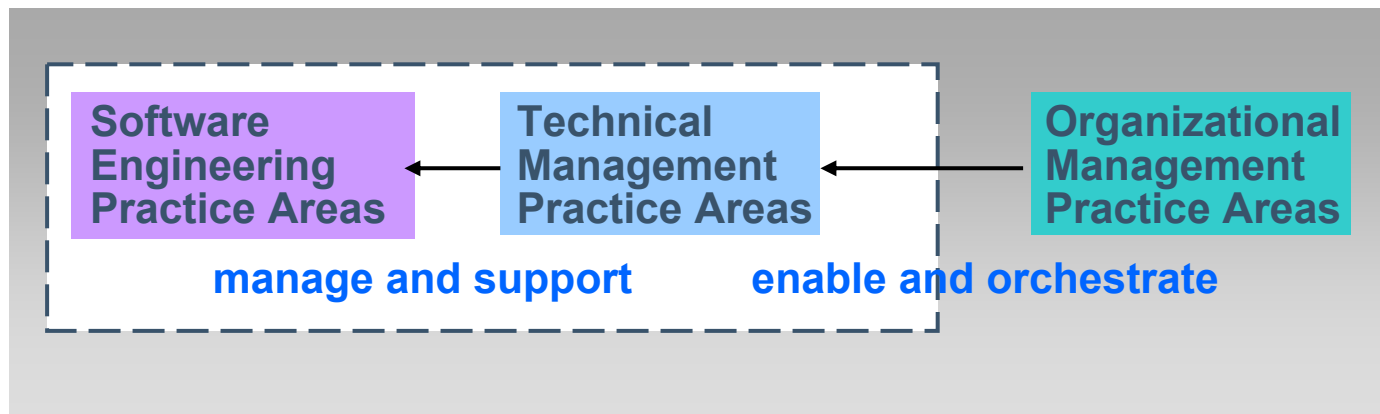


Practice Areas Categories



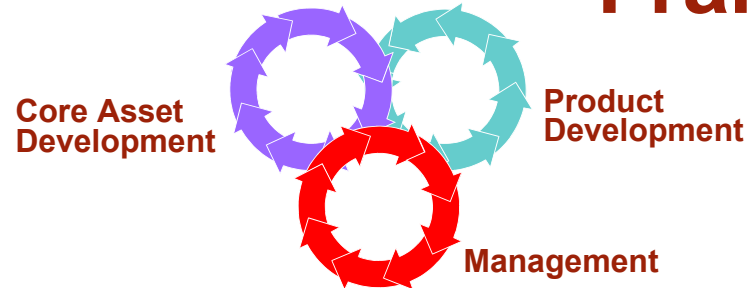


Relationships among Categories of Practice Areas





Framework



Essential Activities

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

Practice Areas



Architecture Definition

The software architecture of a software system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them.¹

Architecture is

- the blueprint for a project
- the carrier of most system quality attributes
- a forum for resource tradeoffs
- a contract that allows multi-party development
- an essential part of complex systems

Bass, L.; Clements, P. & Kazman, R. *Software Architecture in Practice, 2nd Edition*. Reading, MA: Addison-Wesley, 2003.



Architecture Definition: Aspects Peculiar to Product Lines

A product line architecture must

- apply to all members of the product line (even if their functions and qualities differ)
- embody the commonalities and variabilities of the family members
- include specific mechanisms for variation



Architecture Definition: Specific Practices

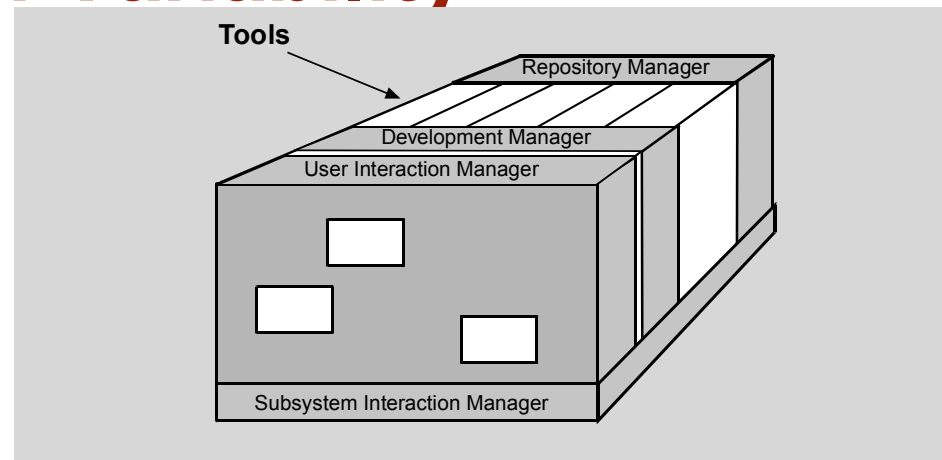
Architecture variability mechanisms

- component replacement, omission, replication
- parameterization (including macros, templates)
- compile-time selection of different implementations (e.g., *#ifdef*)
- OO techniques: inheritance, specialization, and delegation
- configuration and module interconnection languages
- generation and generators
- aspect-oriented programming
 - an approach for modularizing system properties that otherwise would be distributed across modules
- application frameworks

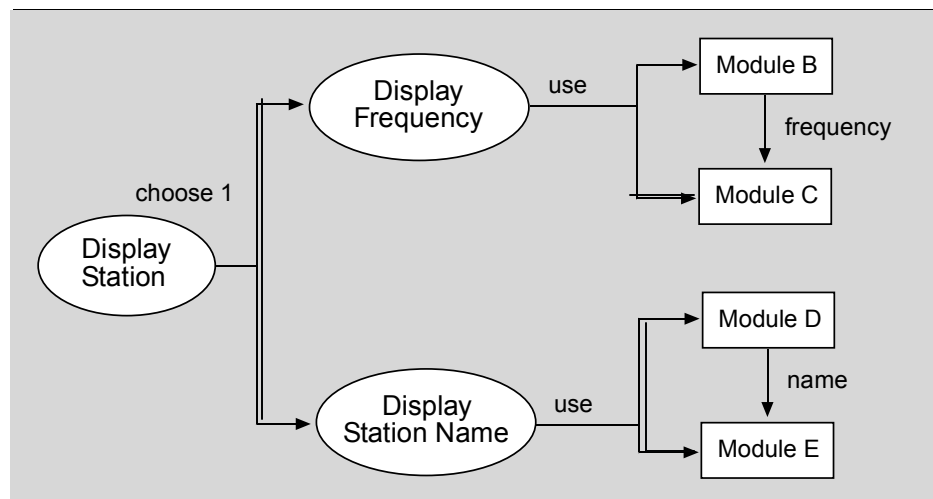


Examples of Variability

Reference architectures with slots for plug-in components



Variation points within a family of products:
Document with a decision tree that shows the choices available





Carnegie Mellon
Software Engineering Institute

Important Concepts

Localization

Variability mechanism

Conditional process

Supporting elements

Dependencies

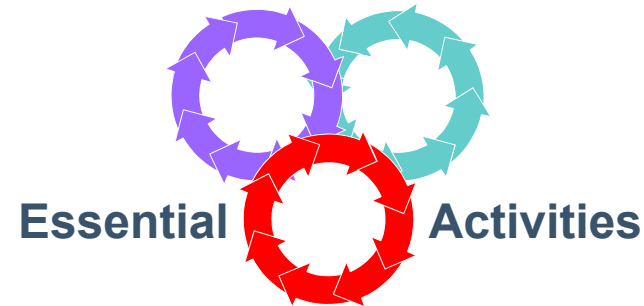


Dilemma: How Do You Apply the 29 Practice Areas?

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

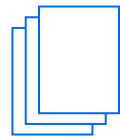
Twenty-nine is a big number.

Help to Make It Happen



Practice Areas

Software Engineering	Technical Management	Organizational Management
-----------------------------	-----------------------------	----------------------------------

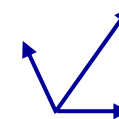


Case Studies

Guidance



Patterns



Probe



Carnegie Mellon
Software Engineering Institute

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>

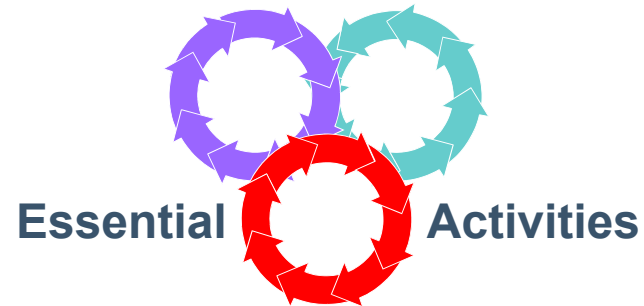
NUWC – *CMU/SEI-2002-TN-018*

<http://www.sei.cmu.edu/publications/documents/02.reports/02tn018.html>

Salion, Inc. – *CMU/SEI-2002-TR-038*

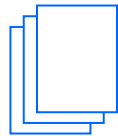
<http://www.sei.cmu.edu/publications/documents/02.reports/02tr038.html>

Help to Make It Happen



Practice Areas

Software Engineering	Technical Management	Organizational Management
-----------------------------	-----------------------------	----------------------------------

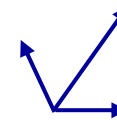


Case Studies

Guidance



Patterns



Probe



Patterns Can Help

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

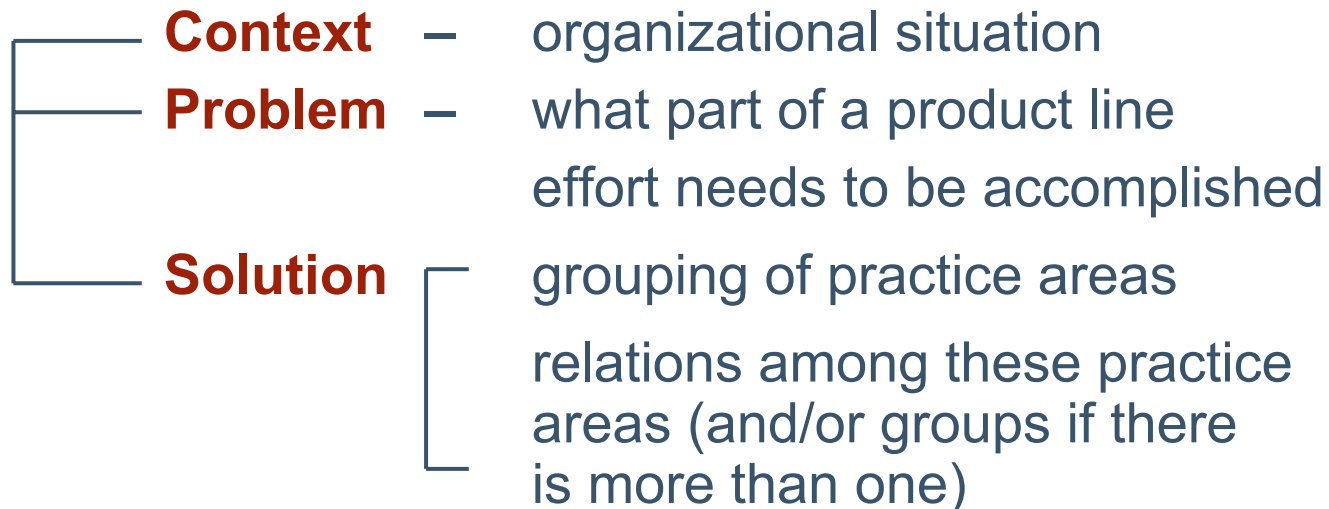
Patterns have been found to be useful in building architecture, economics, software architecture, software design, software implementation, process improvement, and others.

Patterns assist in effecting a divide and conquer approach.



Software Product Line Practice Pattern

Pattern



What to Build Pattern - 1

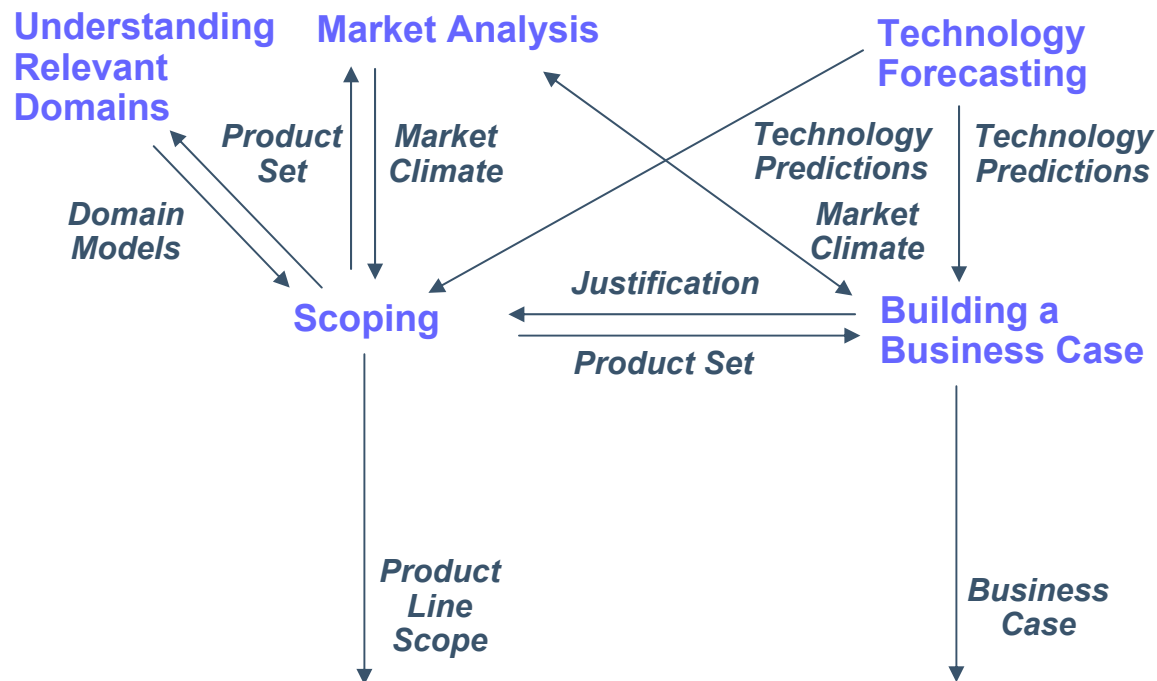
Name:

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.

What to Build Pattern - 2



Dynamic Structure



Factory Pattern - 1

Name:

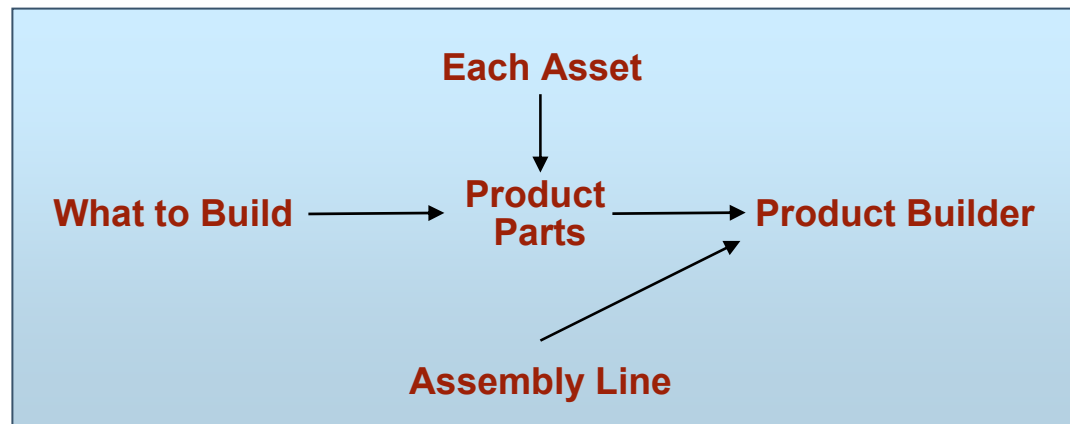
The **Factory** patterns is a composite pattern that describes the entire product line organization.

Context:

An organization is considering (or fielding) a product line.



Factory Pattern - 2



Cold Start In Motion Monitor

→
Informs

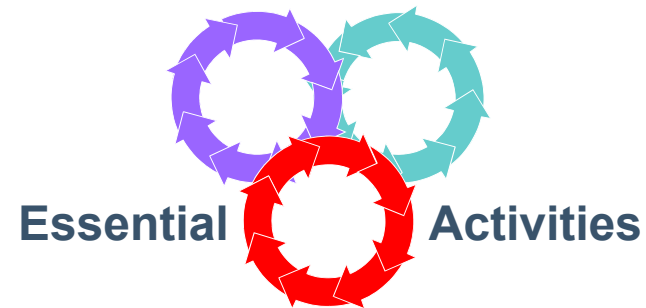
Dynamic Structure



Current Set of Patterns

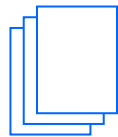
Pattern	Variants
Assembly Line	
Cold Start	Warm Start
Curriculum	
Each Asset	Each Asset Apprentice Evolve Each Asset
Essentials Coverage	
Factory	Adoption Factory
In Motion	
Monitor	
Process	Process Improvement
Product Builder	Product Gen
Product Parts	Green Field Barren Field Plowed Field
What to Build	Analysis Forced March

Help to Make It Happen



Practice Areas

Software Engineering	Technical Management	Organizational Management
-----------------------------	-----------------------------	----------------------------------

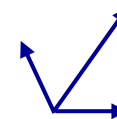


Case Studies

Guidance

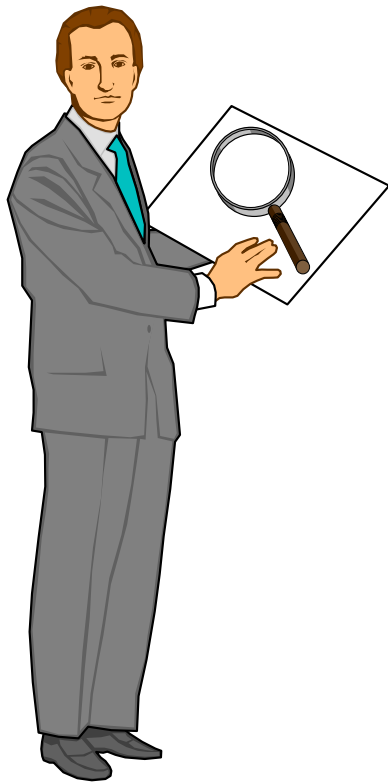


Patterns



Probe

What Is a Product Line Technical Probe?



A method for examining an organization's readiness to adopt or ability to succeed with a software product line approach

- diagnostic tool based on the SEI Framework for Software Product Line Practice
- Practice areas are the basis of data collection and analysis.



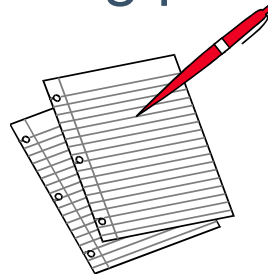
PLTP Outcomes

Set of findings that portray organizational

- strengths
- challenges

with regard to a product line approach

Findings can be used to develop an action plan with the goal of making the organization more capable of achieving product line success.





PLTP Applicability

When an organization

- is considering adopting a software product line approach
- has already initiated a software product line approach





Getting There

Product line adoption involves moving from some form of developing software-intensive systems with a single-system mentality to developing them as a software product line.



The Adoption Endgame

Effectively achieve an **operational product line**.

- have
 - a core asset base
 - supportive processes and organizational structures
- develop products from that asset base in a way that achieves business goals
- improve and extend the software product line adoption effort as long as it makes sense



Barriers to Product Line Adoption

Cost, cost, and
cost





Barriers to Product Line Adoption

Time, time,
and time





Time Needed for Product Line Adoption

Time is needed to

- launch the product line effort
 - educate
 - address cultural barriers
- define supportive processes and organizational structures
- develop a core asset base
- lead the organization to an operational product line
- continue to do business

An organization can't go out of business while adopting a product line approach.



More Barriers

Lack of knowledge

Need for organizational change

Cultural resistance

Lack of sufficient management support

Lack of necessary talent

Incompatible development processes

Globalization of workforce

Stove-piped mentality

No clear path to follow

Others?????

Factors Influencing Adoption

Organizational Context

product line readiness 

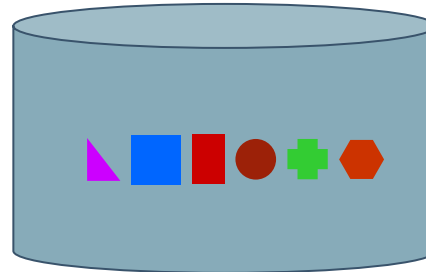
barriers 

enablers 

unique  characteristics

culture 

other ongoing activities 



Factors Influencing Adoption

Organizational Context

product line readiness 

barriers 

enablers 

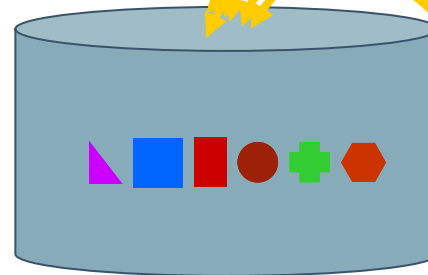
unique  characteristics

culture 

other ongoing activities 

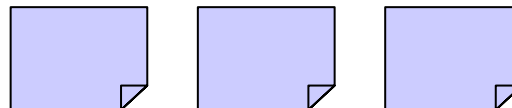
Adoption Support

-  The Framework
-  product line adoption roadmap
-  product line approaches
-  change models
-  change management mechanisms
-  planning process



Product Line Adoption Plan

Product Line Action Plans



Factors Influencing Adoption

Organizational Context

product line readiness 

barriers 

enablers 

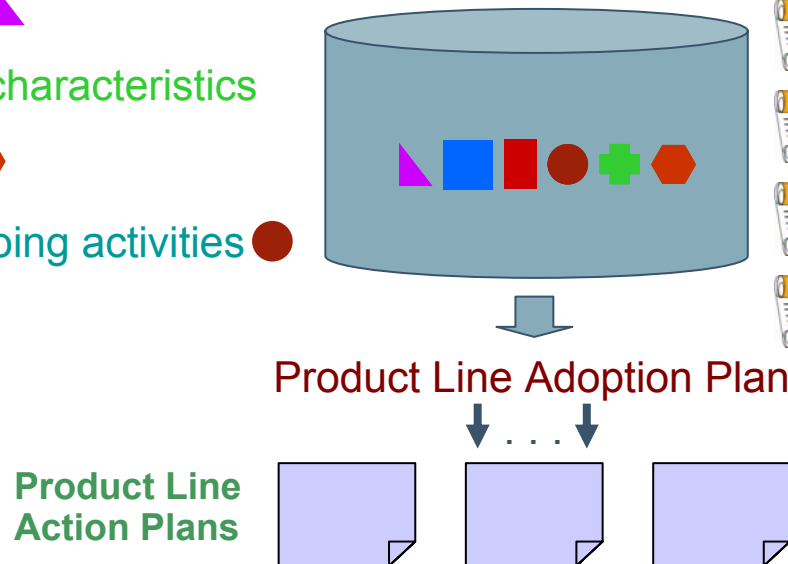
unique  characteristics

culture 

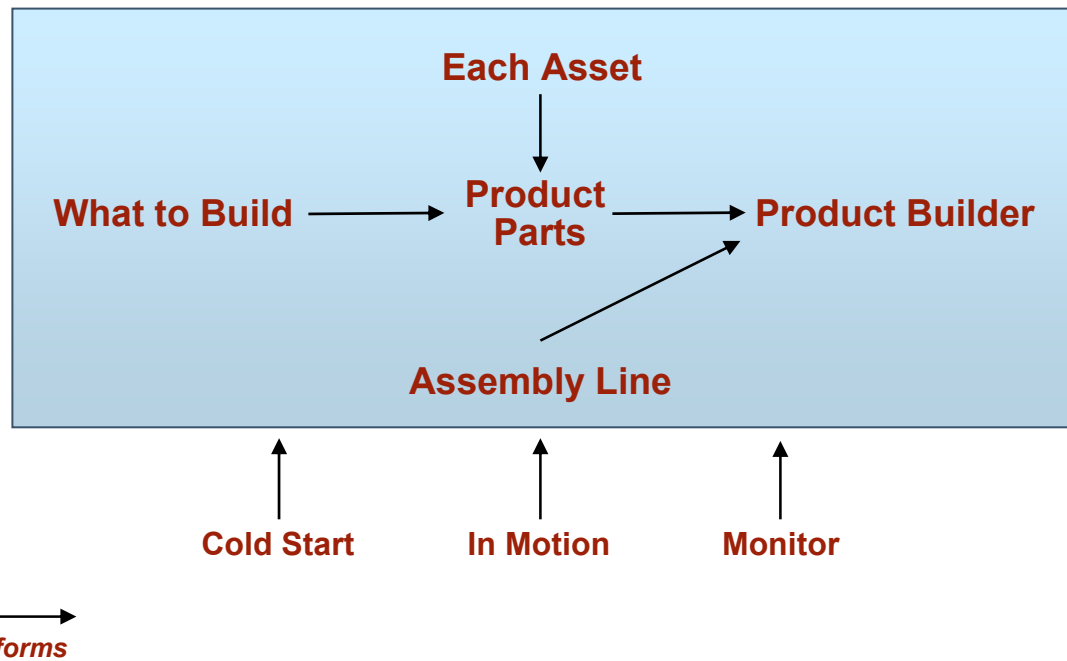
other ongoing activities 

Adoption Support

-  The Framework
-  **product line adoption roadmap**
-  product line approaches
-  change models
-  change management mechanisms
-  planning process



Factory Pattern Revisited



Dynamic Structure



A Variant for Adoption

The **Factory** pattern is already a roadmap for the entire product line organization:

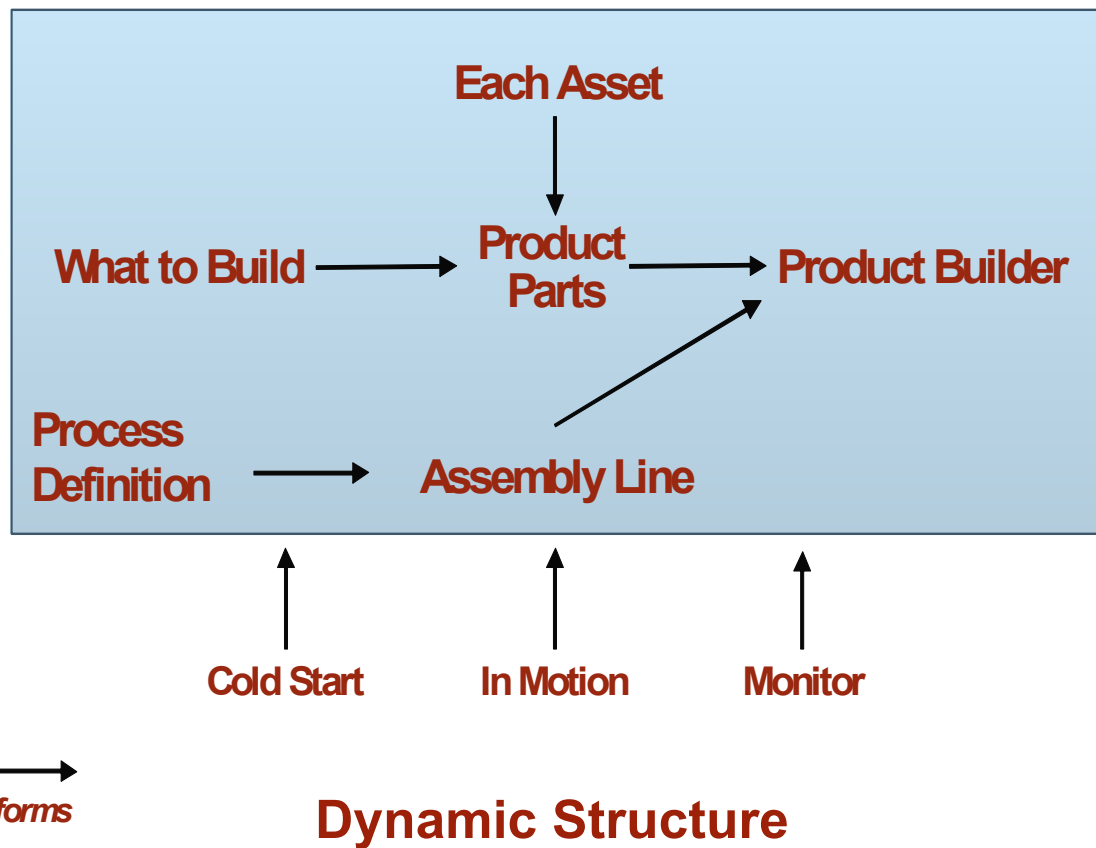
- a top-down view of the product line organization
- a blueprint for a divide-and-conquer strategy

Organizations that lack the ability to define and follow processes, even lightweight or agile ones, need to address that deficiency early in their adoption path.

Even though the “Process Definition” practice area is part of the Assembly Line pattern, it is called out separately in a variant on the **Factory** pattern.

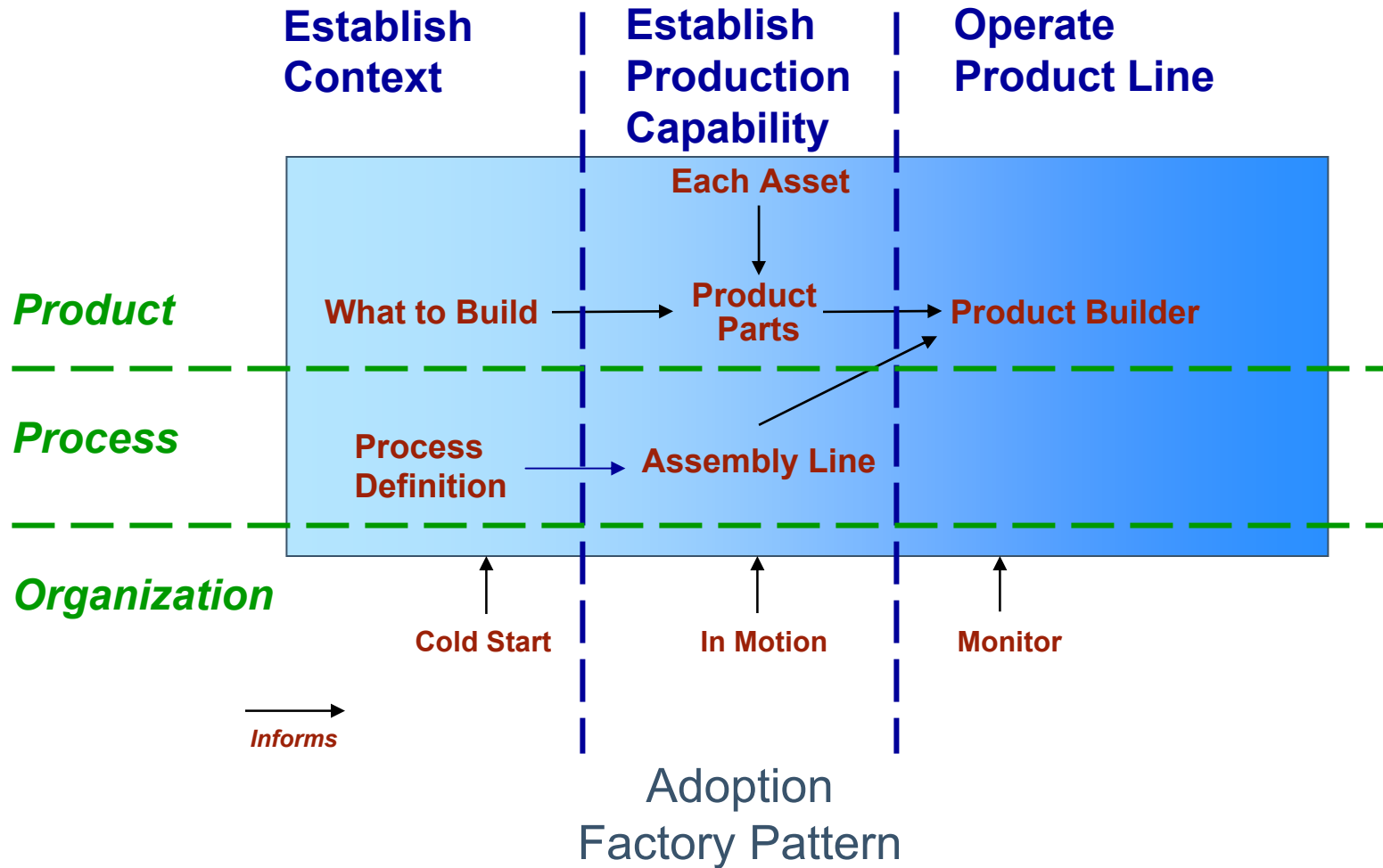
The variant is called the **Adoption Factory** pattern.

Adoption Factory Pattern





Adoption Factory Pattern





Using the Adoption Factory Pattern- 1

To use the *Adoption Factory* pattern as a roadmap

- Elaborate the practice areas associated with its subpatterns.
- Plan to master these practice areas in a continuous way that begins at the phase where they first appear.

The *Adoption Factory* pattern applies regardless of the adoption strategy chosen – proactive, reactive, or incremental.



Associated Practice Areas

	Establish Context	Establish Production Capability	Operate Product Line
Product	<ul style="list-style-type: none"> Marketing Analysis Understanding Relevant Domains Technology Forecasting Building a Business Case Scoping 	<ul style="list-style-type: none"> Requirements Engineering Architecture Definition Architecture Evaluation Mining Existing Assets Component Development COTS Utilization Software System Integration Testing 	<ul style="list-style-type: none"> Requirements Engineering Architecture Definition Architecture Evaluation Mining Existing Assets Component Development COTS Utilization Software System Integration Testing
Process	<ul style="list-style-type: none"> Process Definition 	<ul style="list-style-type: none"> Make/Buy/Mine/Commission Configuration Management Tool Support Data Collection, Metrics, Tracking Technical Planning Technical Risk Management 	
Organization	<ul style="list-style-type: none"> Launching and Institutionalizing Funding Structuring the Organization Operations Organizational Planning Customer Interface Management Organizational Risk Management Developing an Acquisition Strategy Training 	<ul style="list-style-type: none"> Launching and Institutionalizing Funding Structuring the Organization Operations Organizational Planning Customer Interface Management Organizational Risk Management Developing an Acquisition Strategy Training 	<ul style="list-style-type: none"> Data Collection, Metrics and Tracking Technical Risk Management Organizational Risk Management Customer Interface Management Organizational Planning



Using the Adoption Factory Pattern - 2

You can also use the **Adoption Factory** pattern to gauge where in the adoption process by phase your organization is and benchmark your activities by measuring yourself against the practice areas in that phase.

- We use the **Adoption Factory** pattern in the analysis part of the PLTP and also in framing recommendations.
- You can use the **Adoption Factory** pattern as an easily understood adoption vocabulary that can be shared across an organization and marks organizational progress.



Implementing the Adoption Plan

Everyone in the product line organization is responsible for implementing the Product Line Adoption Plan.

- Each person has a stake.
- Each person has a role.
- Each person needs to contribute.

Coordination and cooperation are fundamental to successful adoption.



Roles View - 1

Another instructive view of the Adoption Factory pattern depicts the type of people who need to be involved in the product line adoption effort.

The Roles View lists the typical roles associated with each quadrant of the Phases and Focus Areas view.

This view can be used for identifying staffing needs and making assignments.

Some roles may appear in multiple phases, but the tasks those roles perform will vary with the phase.



Roles View - 2

	Establish Context Phase	Establish Production Capability Phase	Operate Product Line Phase
Product-related roles	<ul style="list-style-type: none">• marketer• market analyst• domain expert• product manager• senior manager• technology scout• architect	<p>core asset developer:</p> <ul style="list-style-type: none">• requirements engineer• architect• architecture evaluator• component developer• tester• software integrator	<p>product developer:</p> <ul style="list-style-type: none">• requirements engineer• architect• architecture evaluator• component developer• tester• software integrator



Roles - 3

	Establish Context Phase	Establish Production Capability Phase	Operate Product Line Phase
Process - related roles	<ul style="list-style-type: none">• technical manager• process owner• process group member	<ul style="list-style-type: none">• technical manager• process owner• process group member• technical support• tool specialist• measurement specialist	<ul style="list-style-type: none">• technical manager• process owner• process group member• technical support• tool specialist• measurement specialist



Roles - 4

	Establish Context Phase	Establish Production Capability Phase	Operate Product Line Phase
Organization-related roles	<ul style="list-style-type: none">• product line manager• software manager• business unit or organization manager• product manager• acquisition expert• financial manager• human resource manager• training planner• training developer• trainer	<ul style="list-style-type: none">• product line manager• software manager• business unit or organization manager• financial manager• training developer• trainer	<ul style="list-style-type: none">• product line manager• product manager• business unit or organization manager• customer field representative• salesperson



Carnegie Mellon
Software Engineering Institute

Today's Talk

Introduction

Product Line Concepts

- What
- Why
- How

Conclusion



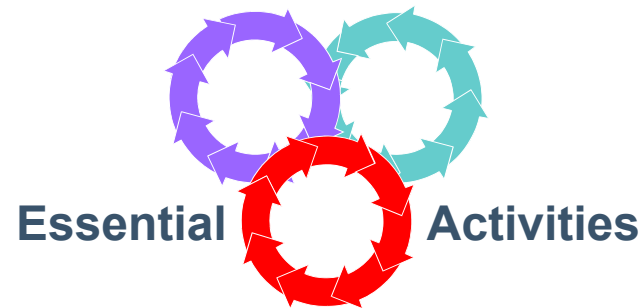
In a Nutshell

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.

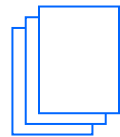
There are essential product line activities and practices areas as well as product line patterns to make the move to product lines more manageable.

The Entire Picture



Practice Areas

Software Engineering	Technical Management	Organizational Management
-----------------------------	-----------------------------	----------------------------------

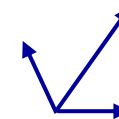


Case Studies

Guidance



Patterns



Probe

Adoption Factory



What's Different About Reuse with Software Product Lines?

Business dimension

Iteration

Architecture focus

Preplanning

Process **and** product connection



Carnegie Mellon
Software Engineering Institute

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



The Time is Right

Rapidly maturing, increasingly sophisticated software development technologies including *object technology, component technology, and standardization of commercial middleware.*

A global realization of the *importance of architecture*

A universal recognition of the need for *process discipline*

Role models and case studies that are emerging in the literature and trade journals

Conferences, workshops, and education programs that are now including product lines in the agenda

Company and intercompany *product line initiatives*

A rising recognition of the *amazing cost/performance savings* that are possible



Evidence of Progress - 1

1. More companies are reporting software product line efforts including
 - John Deere (tractor manufacturer) went from turning out one software product in ten years to turning out two products in one year.
 - Agilent (a telecom company) is using SEI Product Line Practice Patterns as a way to successfully navigate its geographically dispersed product line effort.
 - Argon Engineering (developer of communication systems that search, identify, and capture signals): reports increased customer satisfaction, shorter development cycles, and decreased costs from its software product lines.



Evidence of Progress - 2

2. Others have product line efforts underway, including

- Caterpillar
- Delphi
- Lockheed Martin
- Northrop Grumman
- Raytheon
- Robert Bosch
- Siemens
- Visteon



Evidence of Progress - 3

3. U.S. Department of Defense product line efforts that were begun in the late 1990s are now showing quantifiable benefits:

- The Naval Undersea Warfare Center (NUWC) developed the RangeWare product line concept and asset base.
- The U. S. Army Technology Applications Program Office (TAPO) and Rockwell Collins successfully developed a software product line for the cockpit software for the Army's special operations helicopters.



Evidence of Progress - 4

4. A software product line approach is being chosen for two major U.S. Army efforts.

- Force XXI Battle Command Brigade and Below (FBCB2)
- Future Combat System (FCS)

5. Both IBM and Microsoft have gotten interested in software product lines.

- IBM included “Software Product Lines” in its 2003 Global Technology Outlook.
- Microsoft uses software product lines as the underlying motivator for its proposed software factories tool environment.



Evidence of Progress - 5

6. Mainstream U.S. conferences and magazines for software developers now feature software product lines:

- OOPSLA
- Software Development East
- ICSE
- AOSD
- *IEEE Software*
- *Software Development Times*



Evidence of Progress - 6

7. Many new technology movements have a direct relationship to software product lines and may provide additional catalysts.

- OMG's Model-Driven Architecture (MDA)
- generative programming
- aspect-oriented development
- UML 2.0
- predictable assembly from certifiable components (PACC) from the SEI

8. SPLC 2004 was a resounding success with representation and presentations from major companies across the globe.



Remaining Challenges

Definition and implementation of appropriate variation mechanisms

Evolution of product line architectures and assets

Funding and business models to support strategic reuse decisions

Effective production plans that meet production constraints

Product line tool support

Ways to lower the initial cost of adoption

Summary of SEI Contributions

Practice integration

- A Framework for Software Product Line PracticeSM, Version 4.1, <http://www.sei.cmu.edu/plp/framework.html>

Techniques and methods

- product line analysis
- architecture definition – Attribute-Driven Design (ADD)
- architecture evaluation – Architecture Tradeoff Analysis MethodSM (ATAMSM)
- mining assets – Options Analysis for ReengineeringSM (OARSM)
- Product Line Technical ProbeSM (PLTP)
- Product Line Quick Look (PLQL)
- Product line practice patterns and the Adoption F

Book

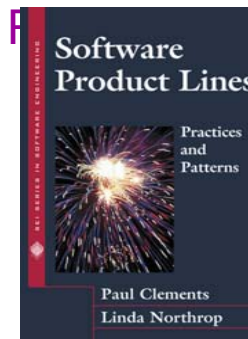
Software Product Lines: Practices and Patterns

Curriculum and Certificate Programs

- Five courses and three certificate programs

Conferences and Workshops

SPLC 1, SPLC2, SPLC 2004; Workshops 1997 - 2004





Ongoing SEI Product Line Research

Product derivation

- variability mechanisms
- production plan definition and implementation

Product line sustainment

- asset evolution

Product line adoption strategies

- economic models

Software Product Line Strategy in Context



Software Product Line Strategy in Context





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



Software product lines: Reuse that pays.



Carnegie Mellon
Software Engineering Institute

Questions – Now or Later

Linda Northrop

Director

Product Line Systems Program

Telephone: 412-268-7638

Email: lmn@sei.cmu.edu

Business Development

Product Line Systems Program

Jay Douglass

Telephone: 412-268-6834

Email: jcd@sei.cmu.edu

U.S. mail:

Software Engineering Institute

Carnegie Mellon University

Pittsburgh, PA 15213-3890

World Wide Web:

<http://www.sei.cmu.edu/ata>

<http://www.sei.cmu.edu/plp>

SEI Fax: 412-268-5758