



**Carnegie Mellon
Software Engineering Institute**

Pittsburgh, PA 15213-3890

Developing Software Product Lines



Copyright 2020 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

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.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. The United States Government has Unlimited Rights in this material as defined by DFARS 252.227-7013.

The text and illustrations in this material are licensed by Carnegie Mellon University under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).

The Creative Commons license does not extend to logos, trade marks, or service marks of Carnegie Mellon University.

Architecture Tradeoff Analysis Method® and ATAM® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

Framework for Software Product Line PracticeSM, IDEALSM, PLQLSM, PLTPSM, Product Line Quick LookSM and Product Line Technical ProbeSM are service marks of Carnegie Mellon University.

DM20-0360



Schedule: Day 1

→ 8:30 – 9:00	Introductions, Background, and a Product Line Refresher
9:00 – 9:15	The Adoption Factory Pattern
9:15 – 9:30	The What to Build Pattern
9:30 – 10:00	Building A Business Case for a Software Product Line
10:00 – 10:15	BREAK
10:15 – 12:00	Group Exercise: Writing a Business Case
12:00 – 13:00	LUNCH
13:00 – 14:30	The Cold Start Pattern
14:30 – 14:45	BREAK
14:45 – 16:30	Group Exercise: Writing a Concept of Operations



Course Objectives

This course will help you carry out the activities necessary for starting and sustaining a software product line.

It will expose you to key decision points in a realistic software product line environment.

It will help you answer

- What do we do *now*?
- What do we *tomorrow*?
- And, how do we do it?



How This Course Relates to Others

This course is one of six courses in the SEI Software Product Line Curriculum and is designed to be taken after you complete the Software Product Lines course.

Completion of this course is a requirement for earning three SEI certificates:

- Software Product Line Professional
- Product Line Technical ProbeSM (PLTPSM) Team Member
- PLTP Leader

Courses	Software Product Line Professional	PLTP Team Member	PLTP Leader
Software Product Lines	■	■	■
Adopting Software Product Lines	■	■	■
Developing Software Product Lines	■	■	■
PLTP Team Training		■	■
PLTP Leader Training			■
PLTP Observation			■

SM Product Line Technical Probe and PLTP are service marks of Carnegie Mellon University.



Course Outcomes

After taking this course, you should

- know how to determine whether a software product line approach is right for your organization
- be able to sketch a business case, concept of operations, attached process for a product line architecture, and a production plan for a software product line
- know the most important patterns for product line practice and when and how to apply them
- know the major artifacts involved with launching a product line



Course Strategy

This course will combine lectures, group exercises, and directed discussions:

- Lectures will provide necessary background.
- Group exercises will give you hands-on experience in creating key software product line artifacts.
- Directed discussions will give you an idea of what is involved in making product line decisions.

A running example – a *pedagogical product line* – for the fictional company **Arcade Game Maker** will be used throughout the course.

AGM
Example





Course Outline - 1

Introduction, Background, and a Product Line Refresher:

Provides brief review of software product line basic concepts: essential activities, framework, practice areas, and patterns

An Overall Vision of a Product Line Organization: Shows how a software product line development effort can be structured around the practice areas specified in the **Adoption Factory pattern**. Also introduces the Arcade Game Maker pedagogical product line and its background and context, which will apply throughout the course

Finding the Right Mix of Products: Introduces **What to Build** as the best pattern for helping an organization define (initially) its product line. Also describes the constituent practice areas, delving into a few in detail

Exercise: Writing a business case



Course Outline - 2

Launching the Product Line: Introduces **Cold Start** as the best pattern for getting an aspiring product line organization off the ground. Describes the constituent practice areas, delving into a few in detail. Also compares Cold Start to the In Motion pattern and discusses when to use both patterns

Exercise: Writing a concept of operations

Developing the Core Asset Base: Introduces **Product Parts** as the best pattern for showing an organization how to set up its production capability and its core asset base. Describes the constituent practice areas, delving into a few in detail.

Exercises:

- 1. Developing the architecture for a product line**
- 2. Writing a production plan for the product line**



Course Outline - 3

Other Patterns: Provides a quick look at other relevant patterns, such as Monitor, In Motion, Assembly Line, and Product Builder

Conclusions, Wrap-Up, and Q&A: Provides a review of the **Adoption Factory** pattern and the course—its major takeaways and conclusions.



Schedule: Day 1

8:30 – 9:00	Introductions, Background, and a Product Line Refresher
9:00 – 9:15	The Adoption Factory Pattern
9:15 – 9:30	The What to Build Pattern
9:30 – 10:00	Building A Business Case for a Software Product Line
10:00 – 10:15	BREAK
10:15 – 12:00	Group Exercise: Writing a Business Case
12:00 – 13:00	LUNCH
13:00 – 14:30	The Cold Start Pattern
14:30 – 14:45	BREAK
14:45 – 16:30	Group Exercise: Writing a Concept of Operations



Schedule: Day 2

8:30 – 9:15	Developing the Core Asset Base: The Product Parts Pattern
9:15 – 10:00	Architectures for Product Lines
10:00 – 10:15	BREAK
10:15 – 12:00	Exercise: Architecture and the Core Asset Base
12:00 – 13:00	LUNCH
13:00 – 13:30	Production Plans
13:30 – 15:00	Group Exercise: Writing a Production Plan
15:00 – 15:15	BREAK
15:15 – 15:45	Other Patterns
15:45 – 16:15	Group Discussion: Measurement
16:15 – 16:30	Conclusions, Q&A



Rules of Engagement

We will be very busy over the next two days. To complete everything and get the most from the course, we will need to follow some rules of engagement:

- Your participation is essential.
- Feel free to ask questions at any time.
- Discussion is good, but we might need to cut some discussions short in the interest of time.
- Please try to limit side discussions during the lectures.
- Please turn off your cell phone ringers and computers.
- Let's try to start on time.
- Participants must be present for all sessions in order to earn a course completion certificate.

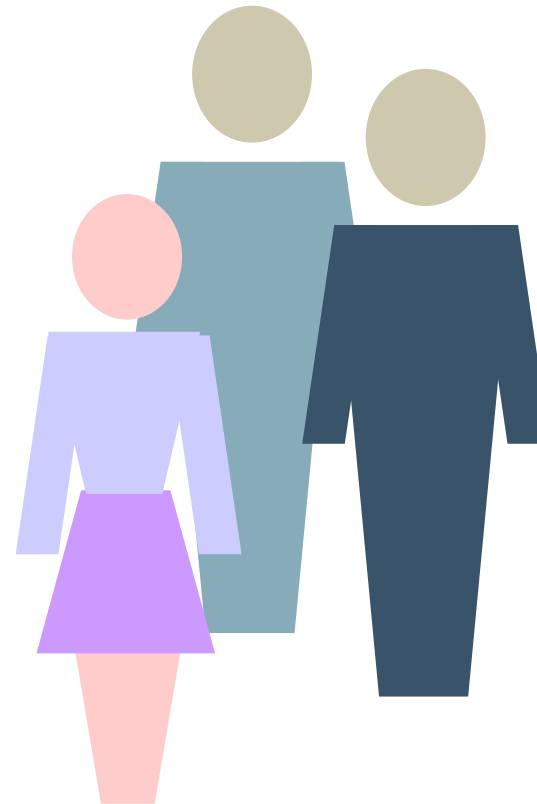


Introductions

Who am I?

Who are you?

Why are you here?





Session Outcomes

After this session, you should

- regain familiarity with the definition and basic concepts of software product lines
- regain familiarity with the three essential activities
- remember the SEI *Framework for Software Product Line Practice*SM, along with its structure and goals, and remember what a practice area is
- have a good understanding of the three practice area categories and the practice areas they contain

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



Refresher: What Is a Software Product Line?

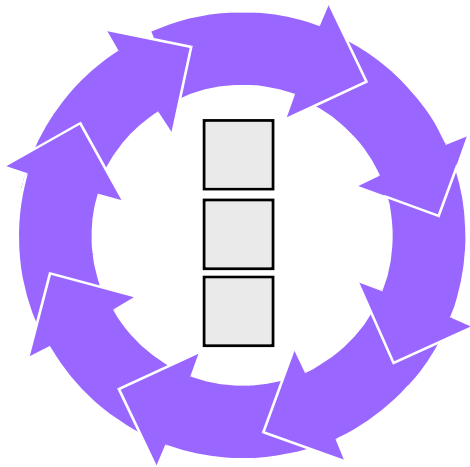
A software product line is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.¹

¹ Clements, P. & Northrop, L. Software Product Lines: Practices and Patterns. Boston, MA: Addison-Wesley, 2001.



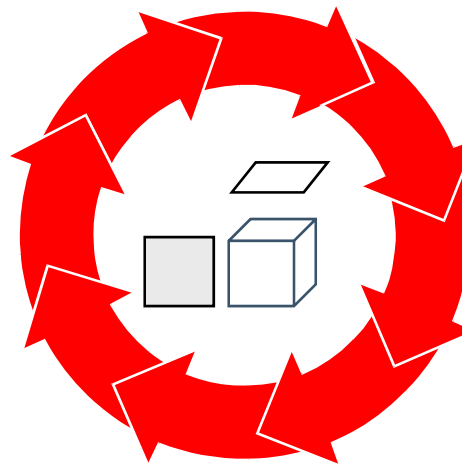
The Key Concepts

Use of a core
asset base



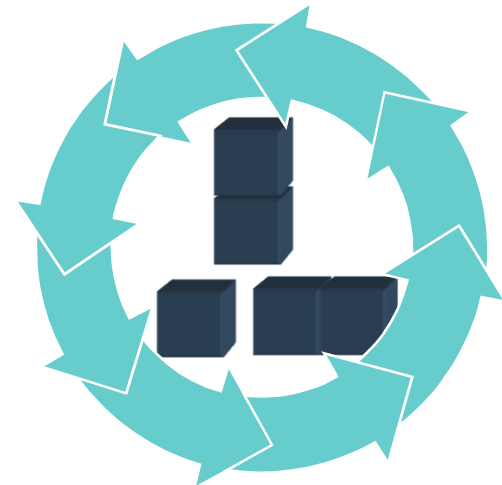
Architecture

in the production



Production Plan

of a related
set of products



Scope Definition
Business Case

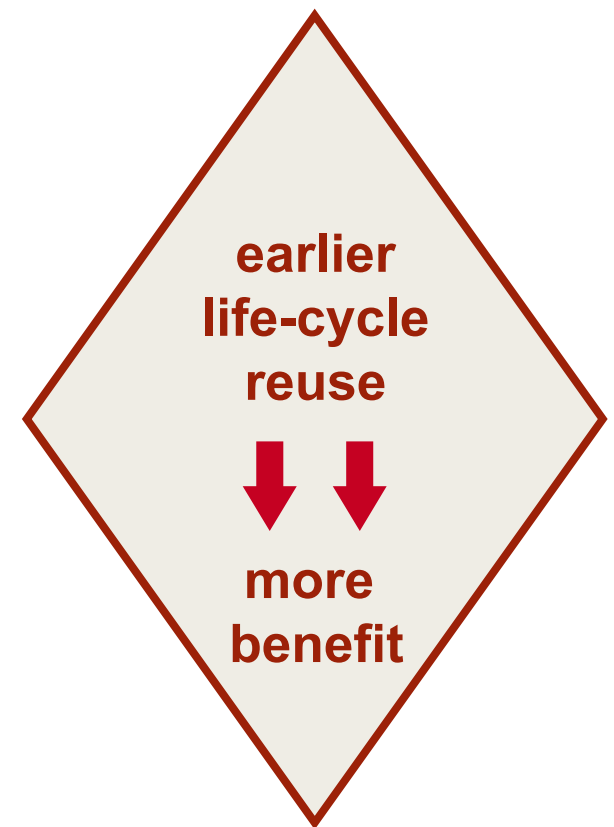


How Do Product Lines Help?

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





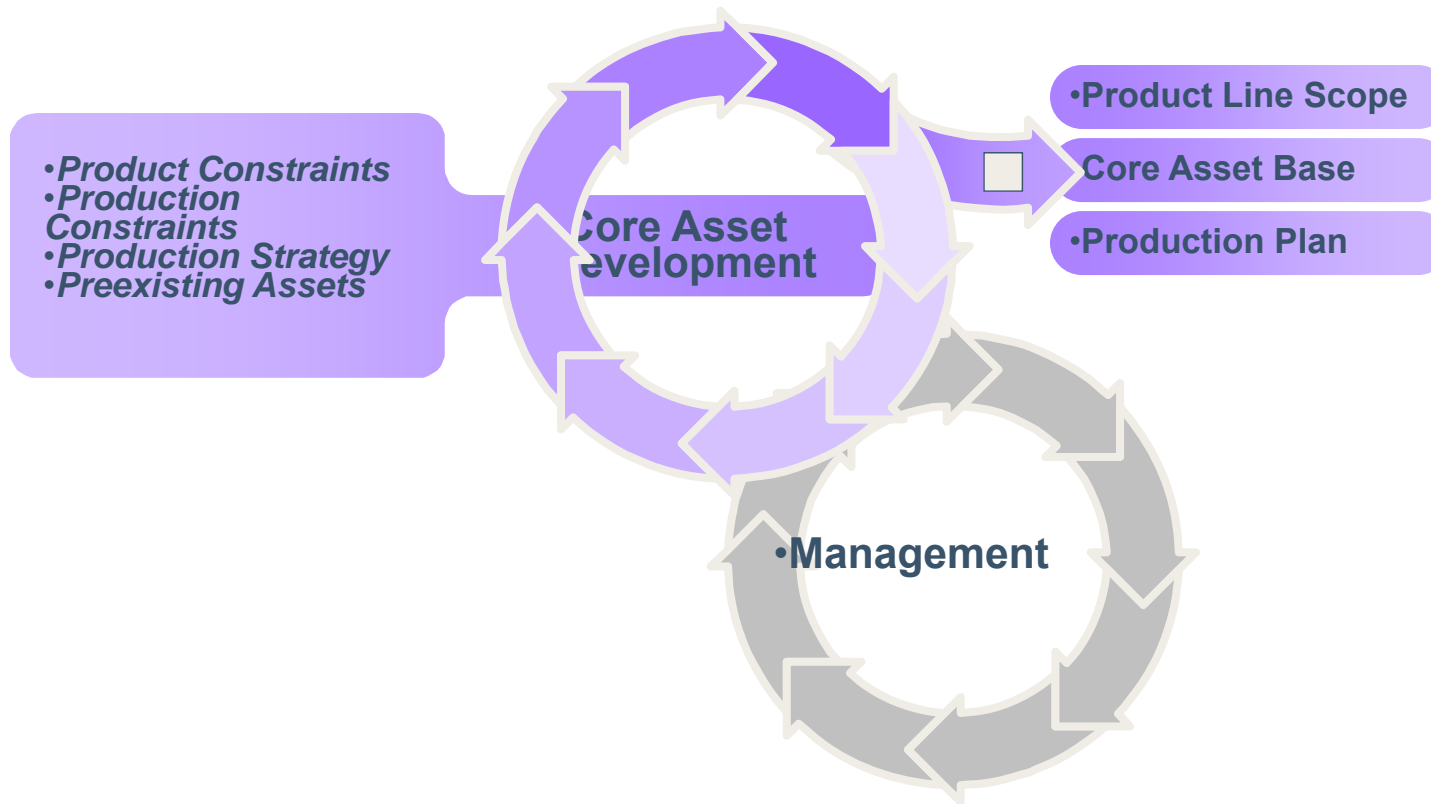
The Three Essential Activities



- All three activities are interrelated and highly iterative.
- Core asset development may precede product development.
- Existing products can be mined for core assets.
- There is a strong feedback loop between core asset development and product development.
- Strong management at multiple levels is needed throughout.

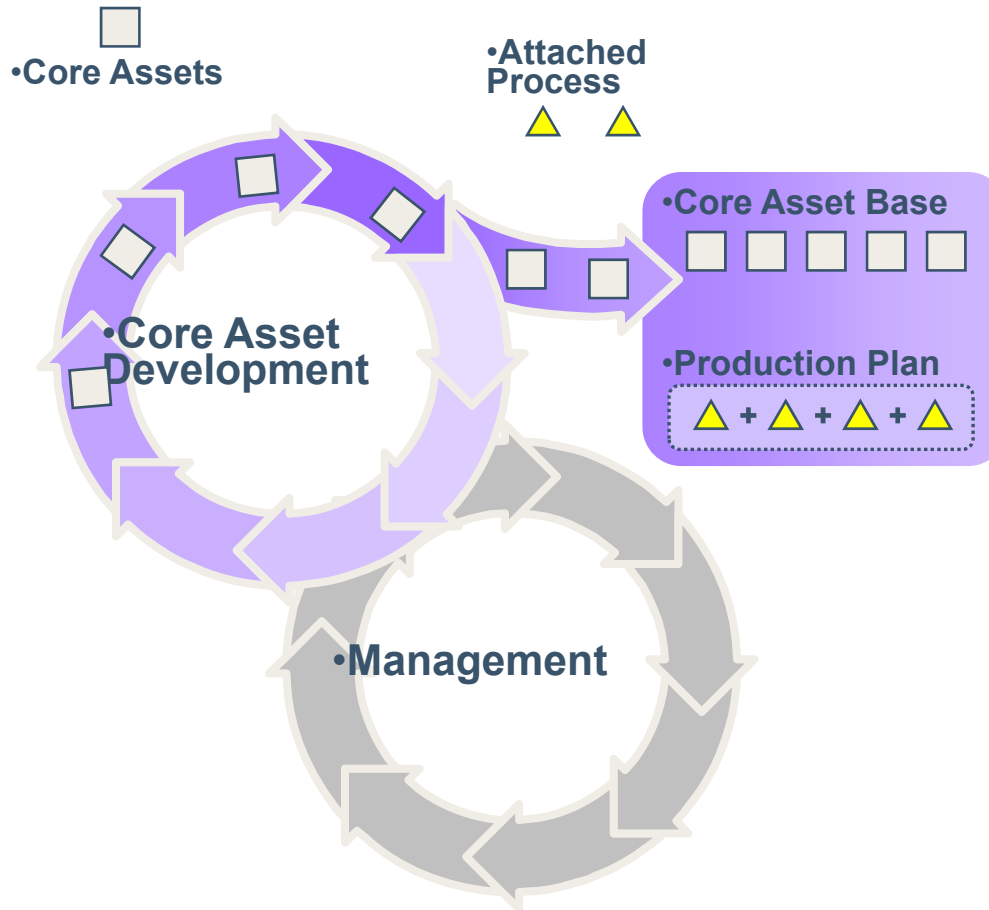


Core Asset Development



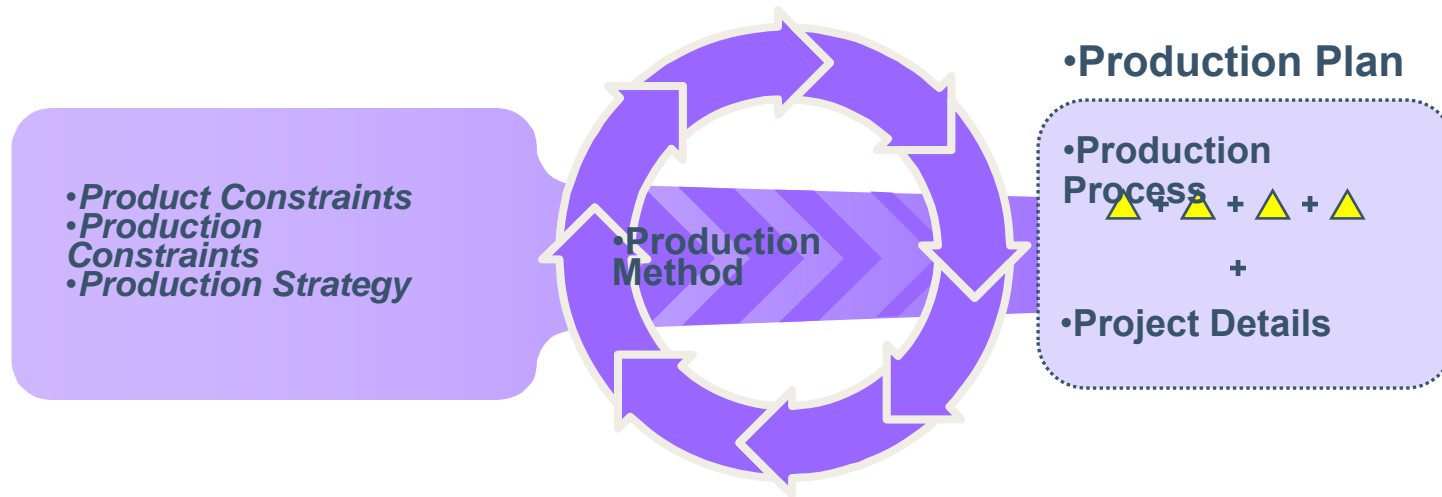


Attached Processes



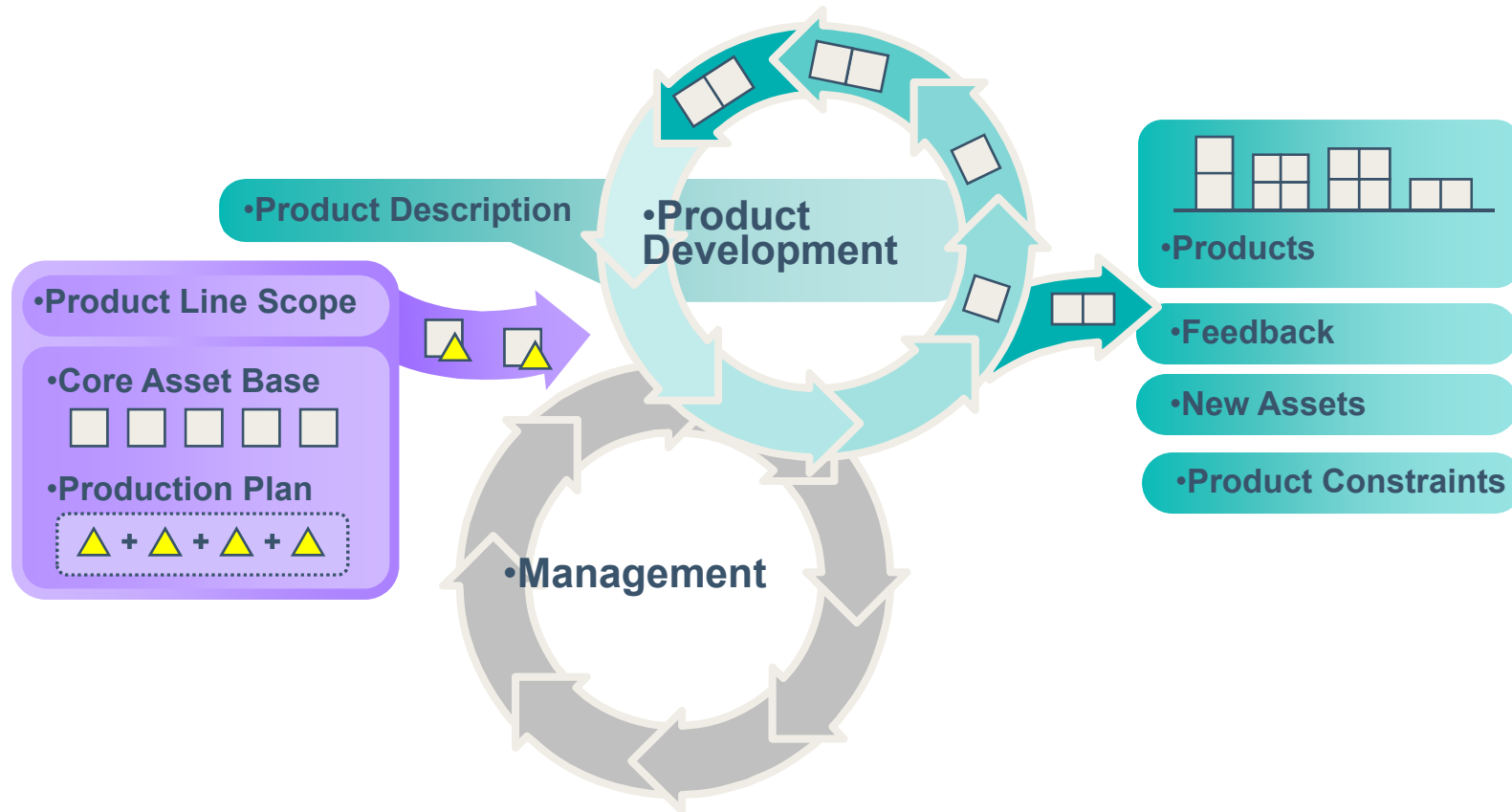


Product Line Production Plan



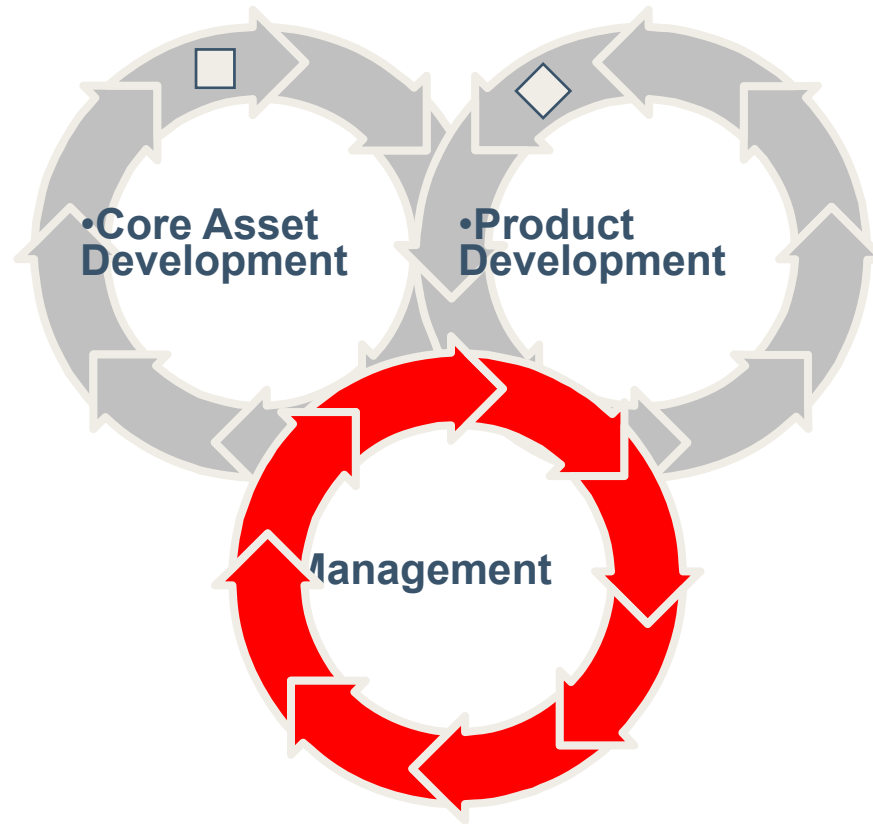


Product Development





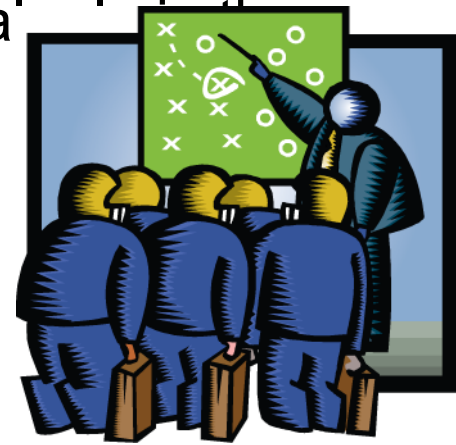
Management





Management

- Management at multiple levels plays a critical role in a 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





The SEI *Framework for Software Product Line Practice* - 1

Beneath the level of the essential activities are essential practices that fall into *practice areas*: bodies of work or a collection of activities that an organization must master to successfully carry out the essential work of a product line.

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



The SEI *Framework for Software Product Line Practice* - 2

The goals of this Framework include

- identifying practice areas that an organization developing software product lines must master
- defining practices in each practice area, where current knowledge is sufficient to do so

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

The latest version is available at

<http://www.sei.cmu.edu/productlines/framework.html>



Software Engineering Practice Areas

These practices are necessary for applying the appropriate technology to create and evolve both core assets and products:

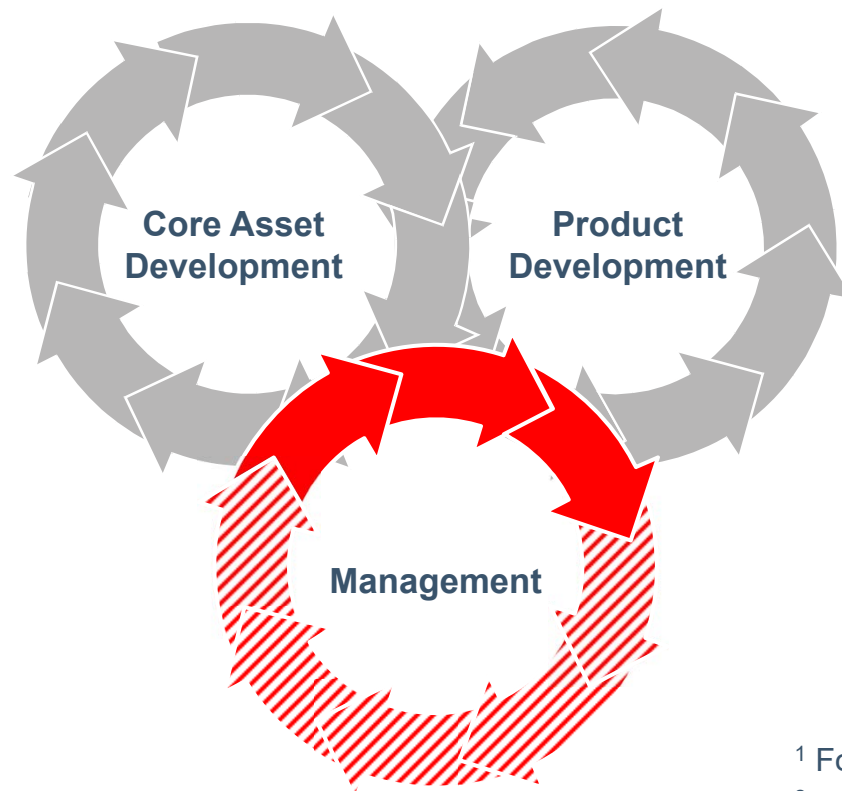


- Architecture Definition
- Architecture Evaluation
- Component Development
- Using Externally Available Software¹
- Mining Existing Assets
- Requirements Engineering
- Software System Integration
- Testing
- Understanding Relevant Domains



Technical Management Practice Areas

These management practices are necessary for the development and evolution of the core assets and products:



- Configuration Management
- Measurement and Tracking¹
- Make/Buy/Mine/Commission Analysis
- Process Discipline²
- Scoping
- Technical Planning
- Technical Risk Management
- Tool Support

¹ Formerly called “Data Collection, Metrics, and Tracking”

² Formerly called “Process Definition”



Organizational Management Practice Areas

These practices are necessary for the orchestration of the entire product line effort:



- Building a Business Case
- Customer Interface Management
- Developing an Acquisition Strategy
- Funding
- Launching and Institutionalizing
- Market Analysis
- Operations
- Organizational Planning
- Organizational Risk Management
- Structuring the Organization
- Technology Forecasting
- Training



Schedule: Day 1

8:30 – 9:00	Introductions, Background, and a Product Line Refresher
 9:00 – 9:15	The Adoption Factory Pattern
9:15 – 9:30	The What to Build Pattern
9:30 – 10:00	Building A Business Case for a Software Product Line
10:00 – 10:15	BREAK
10:15 – 12:00	Group Exercise: Writing a Business Case
12:00 – 13:00	LUNCH
13:00 – 14:30	The Cold Start Pattern
14:30 – 14:45	BREAK
14:45 – 16:30	Group Exercise: Writing a Concept of Operations



Session Outcomes

After this session, you should

- regain familiarity with the concept of product line practice patterns and why they are useful
- know how a pattern is described
- understand how the Adoption Factory pattern defines the vision for an entire product line organization
- know the subpatterns and practice areas that make up the Adoption Factory pattern
- understand some of the available entry points into and paths through the Adoption Factory pattern



Running Example Used for This Course

We have developed a **pedagogical software product line** to serve as a running example throughout this course.

- The fictional company **Arcade Game Maker (AGM)**, a producer of computer games, wishes to develop its products as a software product line.
- We will develop plans and other documents necessary to help it along.
- For the remainder of this course, we all work in the office of AGM's vice president of product development (VPPD).

The pedagogical software product line is available at
<http://www.sei.cmu.edu/productlines/ppl>



Special Formatting Used to Highlight AGM-Specific Information

The formatting includes four things:

1. this special AGM Example icon



2. this character in the slide title



3. a red rule going down the left side of the text



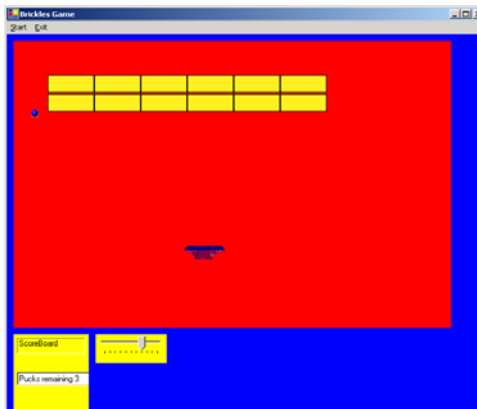
4. bright-blue-colored text



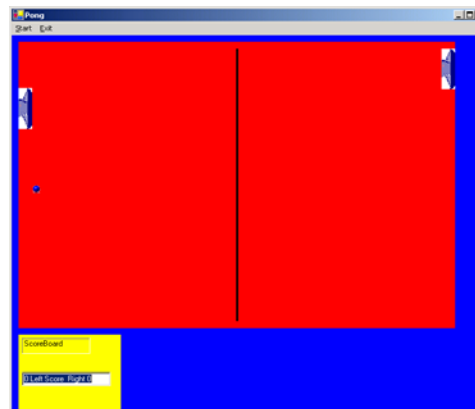
▶ AGM - 1

AGM makes computer games.

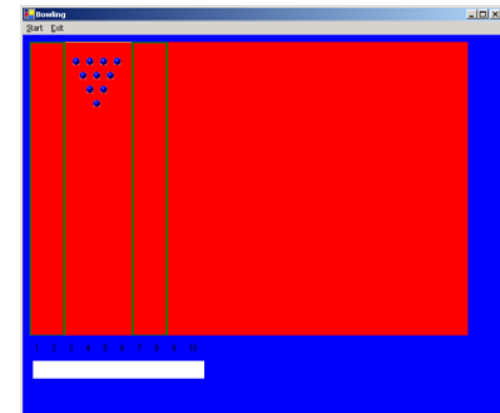
They currently produce three products:



Brickles



Pong



Bowling



▶ AGM - 2

AGM maintains a presence in three different markets:

1. personal computer freeware
2. wireless devices
3. a customizable version for company advertising

Games can vary by user interface and rules and by the platform on which they run. These are successful products.

However, trouble is looming on the horizon!

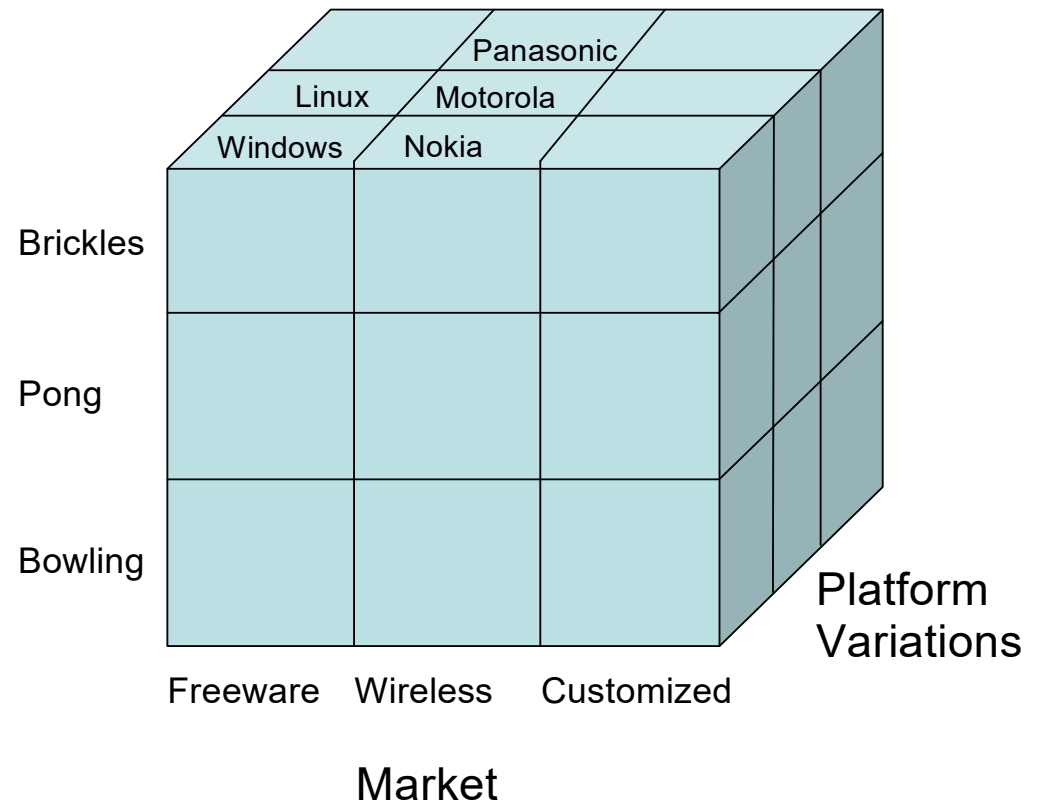


▶ AGM - 3

The number of platforms our games must run on is increasing. Marketing is demanding more and more product variations with shorter time to market.

Games

The complexity of our products will soon be out of control. Mass hiring is not an option.





▶ AGM - 4

It has become clear that

- AGM is building multiple systems with significant commonality.
- There is an overall fit for a software product line approach.

AGM has articulated goals that it is trying to achieve with a software product line approach. The benefits of successful product lines match the goals of the organization.

The AGM VPPD¹ is familiar with the SEI *Framework for Software Product Line Practice*.

¹ Vice President of Product Development



▶ AGM - 5

The VPPD¹ has launched a software product line effort at AGM:

- This means there is sufficient support within the organization to launch a software product line adoption effort. The VPPD is willing to spearhead the effort.
- AGM has embraced the Adoption Factory pattern as the basis for its product line adoption plan.

Our job is to help the VPPD develop the product line.

¹ Vice President of Product Development



Review: Product Line Practice Patterns

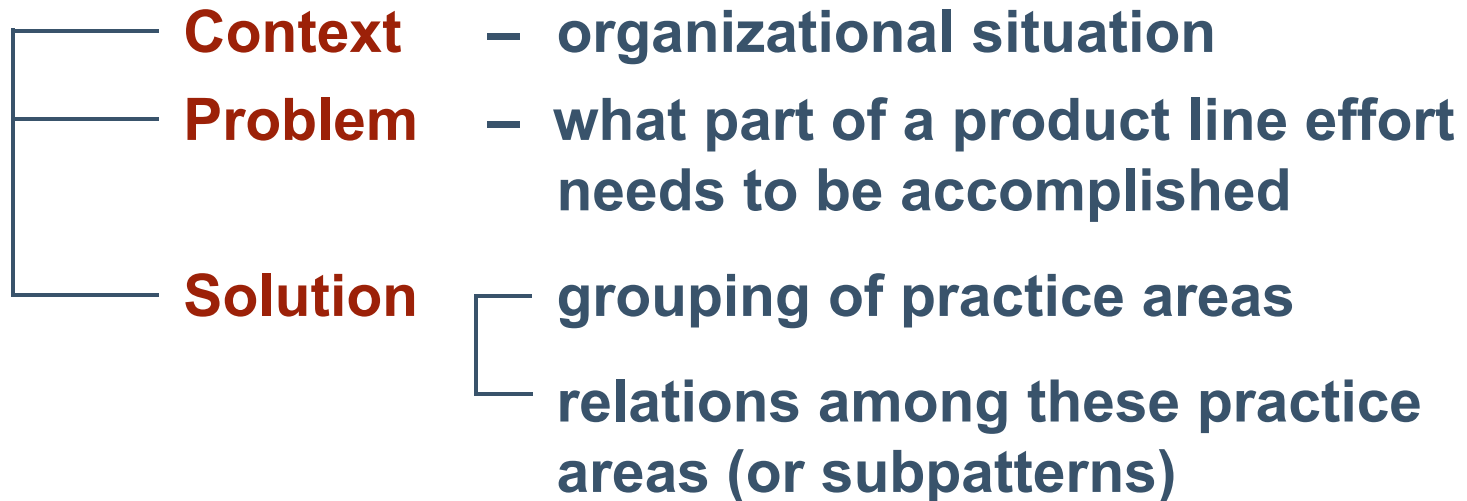
Product line practice patterns

- are ways of expressing solutions to common context/problem pairs
- address recurring product line problems, such as
 - How do I get started?
 - What products should my product line include?
 - How do I build a core asset base?
- codify existing, well-proven software product line experience
- provide an additional common vocabulary for understanding product lines
- can be combined to build complex product line solutions



Software Product Line Practice Pattern

Pattern





Building the Software Product Line

We now turn to a pattern that describes a software product line organization in full swing—the **Adoption Factory pattern**.

We'll use this pattern as a picture of the ultimate desired state and roadmap for how to get there.



Adoption Factory Pattern - 1

Name: The **Adoption Factory** pattern is a composite pattern* that describes the entire product line organization.

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

Problem: To map the entire product line effort

* A composite pattern is one composed of other patterns (subpatterns).



Adoption Factory Pattern - 2

Solution: Fielding a product line involves

- deciding what to build
- building and running the production capability
- preparing the organization
- designing and providing the product parts
- running the assembly line to produce products
- monitoring the process
- defining processes that are important during adoption



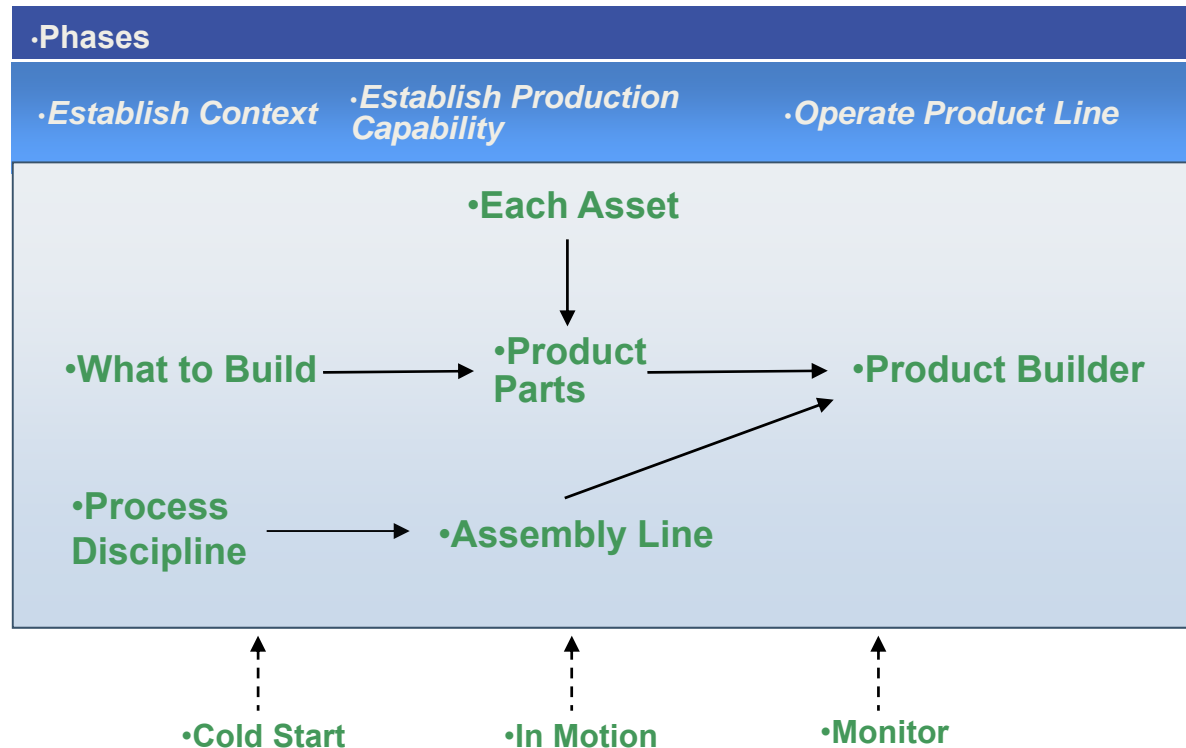
Adoption Factory Pattern - 3

Static Structure: The **Adoption Factory** pattern consists of the following subpatterns:

- Assembly Line
 - Each Asset
 - Cold Start
 - In Motion
 - Product Builder
 - Product Parts
 - Monitor
 - What to Build
- ...plus one practice area
- Process Discipline



Adoption Factory Pattern - 4



→
•Informs and information flow

- - - - ->
•Supports

Dynamic View



Adoption Factory Pattern - 5

Many paths through the Adoption Factory pattern are possible. (They are discussed more in the SEI Adopting Software Product Lines course.)

Which one you choose depends on your context and how much preliminary work or thinking has been done about software product lines.

Also, you don't always complete one thing before you start another.

We will explore one path through the Adoption Factory pattern, based on the hypothetical situation of AGM and the adoption plan it built.



► Using the Adoption Factory Pattern at AGM

The Adoption Factory pattern embodies the overall concept of product line adoption and will help us explain our vision of AGM to those above and below the office of the VPPD¹.

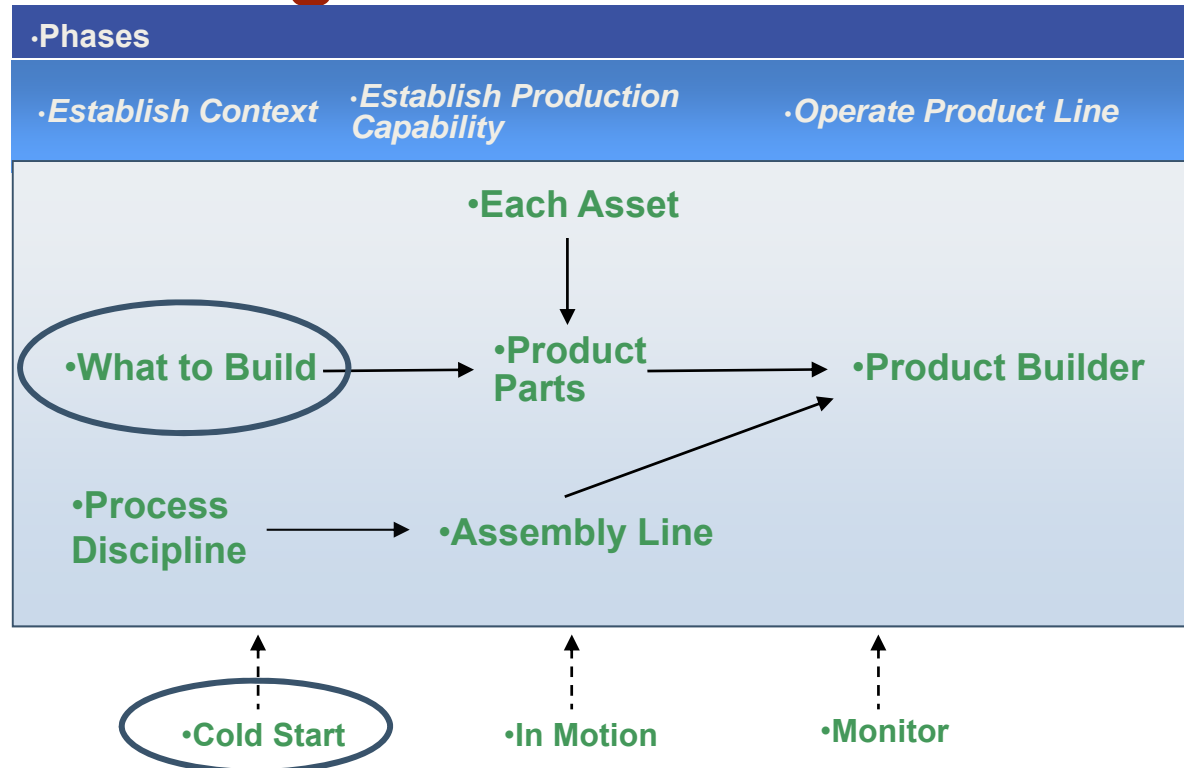
However, since the Adoption Factory pattern is composed of other patterns; we cannot carry it out directly.

Rather, we must choose one or more subpatterns to start with.

¹ Vice President of Product Planning



▶ A Starting Point for AGM



First, AGM will work to establish context:

- For products, this is carried out by the What to Build pattern.
- For the organization, the Cold Start pattern is what we need.

Since AGM already has a product set, following the What to Build pattern should be relatively easy. The VPPD chooses to first concentrate there.



Schedule: Day 1

8:30 – 9:00	Introductions, Background, and a Product Line Refresher
9:00 – 9:15	The Adoption Factory Pattern
 9:15 – 9:30	The What to Build Pattern
9:30 – 10:00	Building A Business Case for a Software Product Line
10:00 – 10:15	BREAK
10:15 – 12:00	Group Exercise: Writing a Business Case
12:00 – 13:00	LUNCH
13:00 – 14:30	The Cold Start Pattern
14:30 – 14:45	BREAK
14:45 – 16:30	Group Exercise: Writing a Concept of Operations



Session Outcomes

After this session, you should

- understand the static and dynamic structure of the What to Build pattern
- understand how the What to Build pattern is useful and how it can help an organization decide whether the software product line approach is right for it
- be able to use the What to Build pattern to reach a decision about product line adoption



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



What to Build Pattern - 2

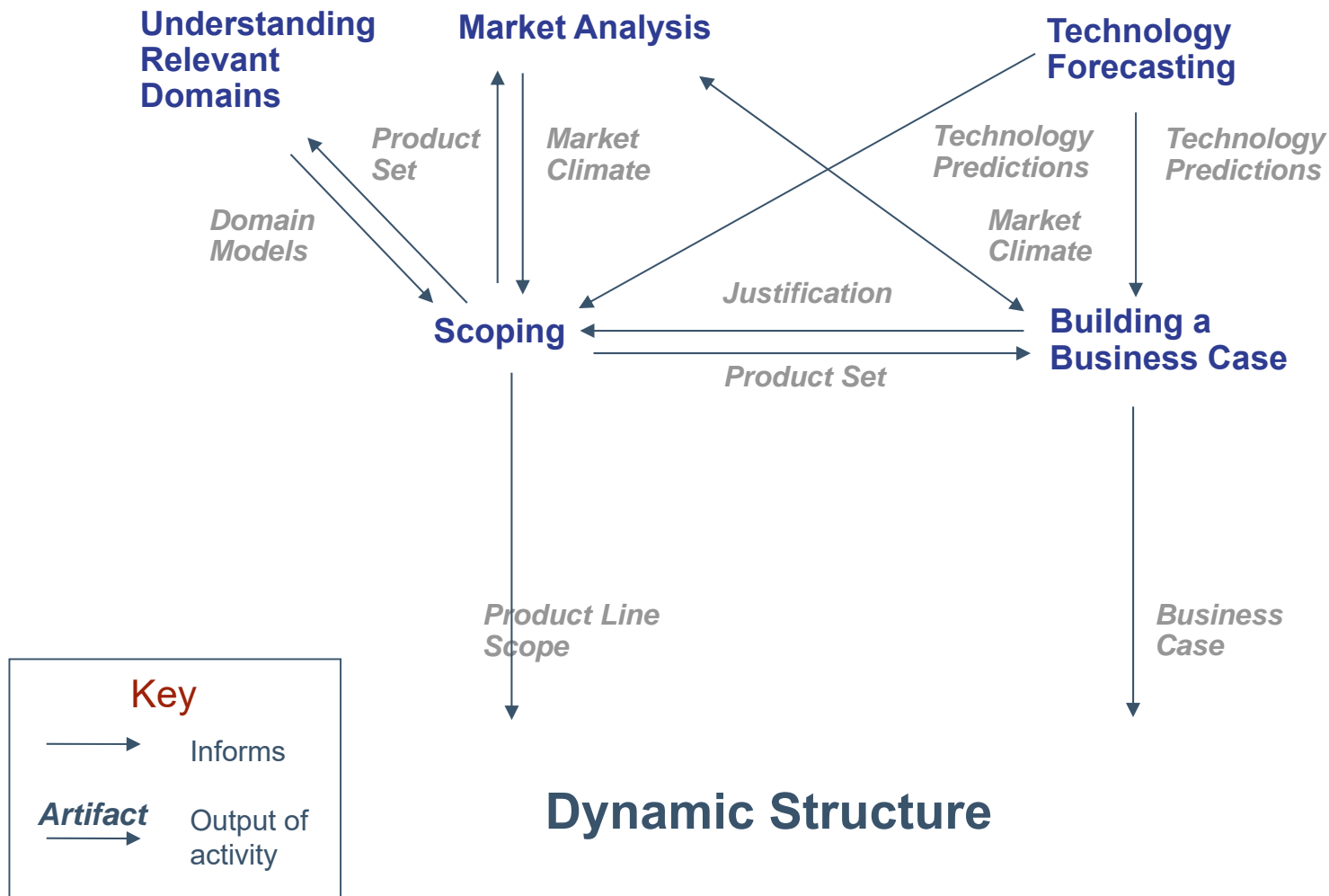
Solution: Determining what to build requires information related to the product area, technology, and market; the business justification; and the process for describing the set of products to be included in the product line.

Static Structure: The following practice areas address the solution and provide the structure for the **What to Build** pattern:

- Market Analysis
- Understanding Relevant Domains
- Technology Forecasting
- Building a Business Case
- Scoping



What to Build Pattern -3





What to Build Pattern - 4

Application:

- Practices from the “Market Analysis,” “Understanding Relevant Domains,” and “Technology Forecasting” practice areas can be conducted in parallel by separate groups.
- The pattern is especially applicable to those well versed in the marketplace.
- It can be carried out by any size organization.
- It can (and should) continue to be applied after a product line already exists.



Schedule: Day 1

8:30 – 9:00	Introductions, Background, and a Product Line Refresher
9:00 – 9:15	The Adoption Factory Pattern
9:15 – 9:30	The What to Build Pattern
→ 9:30 – 10:00	Building A Business Case for a Software Product Line
10:00 – 10:15	BREAK
10:15 – 12:00	Group Exercise: Writing a Business Case
12:00 – 13:00	LUNCH
13:00 – 14:30	The Cold Start Pattern
14:30 – 14:45	BREAK
14:45 – 16:30	Group Exercise: Writing a Concept of Operations



Session Outcomes

After this session, you should

- understand a recommended structure for a business case document
- know the technical and organizational inputs that feed the business case
- see how the business case is built in conjunction with other activities
- know the basic modeling constructs of the SIMPLE product line economics model
- be able to build a complicated SIMPLE formula that models a product line scenario
- be ready to write a business case for a software product line



► What Are the Right Products for AGM?

We need to write a *business case* for AGM, because it will determine the right products for AGM.

The What to Build pattern shows that the business case is influenced by three things:

1. technology predictions
2. market climate
3. an initial product set (scope)

Our first task is to assemble this information. We ask the VPPD to gather together the company's technology experts, market experts, and product experts to produce three initial reports.

These initial reports will lay the groundwork but should be updated as needed throughout the life of the product line.



▶ 1. AGM's Technology Predictions

The VPPD has chartered a number of forward-looking technology-exploration projects.

- These studies have investigated the impact of new platforms such as the N-Gage from Nokia, the implications of the life cycle of the Mobile Information Device Protocol (MIDP) set of standards, and the viability of TinyOS to support game play on ultra-small devices.
- These studies do not affect the current product space, as they identified no new technologies that are market mature. The MIDP study showed that the planned product line should remain with the MIDP1.0 standard, since the MIDP2.0 standard will not be widely supported until after the launch times of the current products.

The studies do influence the identification of variation points but not their current implementations.

You can see the AGM Technology Forecast at
www.sei.cmu.edu/productlines/ppl.



▶ 2. AGM's Market Climate - 1

The number of platforms and venues for computer games continues to expand. AGM's vice president of product planning (VPPP) has chartered a regular update of the Marketing and Product Plan (MPP).

The most notable current opportunity is for wireless devices such as PDAs, cell phones, and the various permutations of these devices. The new N-Gage product line of game "decks" from Nokia presents interesting possibilities, although current sales are disappointing.



▶ 2. AGM's Market Climate - 2

The most likely emerging opportunity will be the infotainment aspects of vehicles. Several new devices are being developed to provide a game platform at various points in automobiles, planes, and other vehicles.

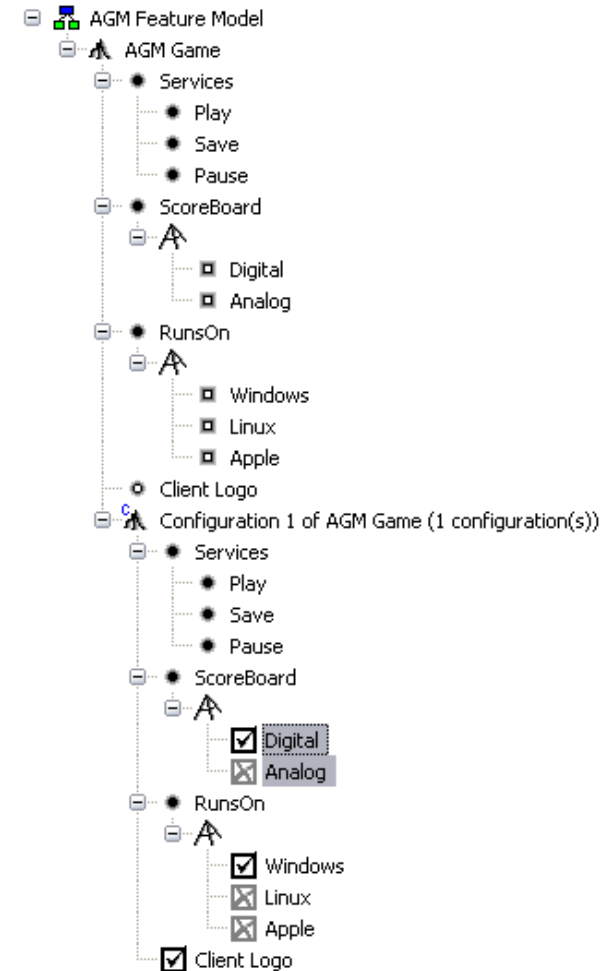
Additional information is provided in the AGM Marketing and Product Plan (MPP), which will be used in the first exercise.

(You can see AGM's Marketing and Product Plan in the "Background" booklet.)



Variation representation

- Feature models are used to represent the commonality and variability among products in a product line.
- Unfilled circles represent optional features.
- The bottom portion shows a product configuration. All variation has been resolved.





▶ 3. AGM's Initial Product Set (Scope) - 1

AGM's product/feature map was constructed using an Eclipse plug-in that supports feature modeling

The screenshot shows the Eclipse Variant Management interface for a project named 'firstV.xfm'. The main window displays a hierarchical tree of features for 'ArcadeGame'. The tree includes features like 'configuration', 'graphicsDisplay', 'locatorInput', 'communication', 'gameRules', 'operatingEnvironment', 'playGame', and 'qualities'. Each feature is marked with a symbol indicating its type: a solid arrow for mandatory, a question mark for optional, and a double-headed arrow for alternative. The 'communication' feature is highlighted with a blue selection box. A legend box on the right side of the screenshot defines the symbols: a solid arrow for 'mandatory', a question mark for 'optional', and a double-headed arrow for 'alternative'.

!	mandatory
?	optional
↔	alternative



▶ 3. AGM's Initial Product Set (Scope) - 2

We know that the current product set will be carried forward.

- 3 games
- Each game implemented on 3 platforms

We are also planning to add new products based on the new platforms shown earlier. These new products come from the “Understanding Relevant Domains,” “Market Analysis,” and “Technology Forecasting” practice areas.

The new products will include additional games, such as a pinball game, and additional platforms for all of the games. The scope of the product line will be stated to provide for this eventual evolution but will not include specific products beyond the three games.

Additional information is provided in the AGM Scope document which will be used in the first exercise.



Making the Case

A business case is essentially a justification for a course of action. There are many ways to justify a product line, but they all must relate the proposed product line to the strategic goals of the organization.

The business case writers will use forecasts about technology trends and market trends to justify the products that are included in the product line.

We will consider how to build an economic model that can support assertions made in the business case.



▶ Market Analysis for AGM

As part of his continuing responsibilities, AGM's VPPP¹ commissioned an analysis of the markets in which AGM is interested.

The resulting MPP² identifies trends in the game industry and, in this case, a niche opportunity.

The MPP provides a feature model for the envisioned products that technical managers can use to estimate the effort required to produce the product.

The MPP provides important input into the business case by predicting sales figures and suggesting the price point at which each product can be sold. This data helps determine release dates, quality attributes, and production techniques.

¹ Vice President of Product Planning

² Marketing and Product Plan



► Scoping for AGM

Representatives of the VPPP and the VPPD cooperate to “scope” (determine which products belong to) the product line. The initial product set is a starting point.

A balance is struck between having sufficient commonality to achieve strategic levels of reuse (for the VPPD) and sufficient variability to satisfy a number of markets (for the VPPP).

This activity will produce scenarios that are analyzed as part of the business case activity. The results of these analyses will be fed back into the scoping activity.

VPPD = Vice President of Product Development

VPPP = Vice President of Product Planning



Business Case Content

The content of the business case includes

1. a mapping of the business goals to the product line strategy
2. a description of alternative approaches to meeting the goals
3. analysis of the strengths/weaknesses/opportunities/threats associated with the product line (SWOT¹ analysis)
4. analysis of strategic factors that influence the product line strategy
5. comprehensive scenario analysis including cost/benefits
6. recommended course of action

¹ Strengths/Weaknesses/Opportunities/Threats



1. Mapping of Business Goals

The context section of the business case describes how the recommendations made in the business case will support the goals of the organization.

One way this is done is to list the quality attributes of the products and the production process that are related to each goal.

For example, an organizational goal of “greatly reduced time to market” can be addressed by “achieve strategic levels of reuse” in the production process.



2. Alternative Approaches

There are several ways to meet the specified business goals.

A scenario is created for each feasible approach. The scenario describes the approach and defines the range of values for any variable in the scenario.

For example, the scoped set of products can be produced within the specified time frame by building each product separately if the organization would hire x number of additional personnel. The cost/benefit analysis should provide for comparing this approach to the product line approach.



▶ 3. SWOT Analysis (for AGM)

The business case includes a SWOT analysis to capture the elements of risk associated with the product line:

Strengths – the organization’s experience in the domain

Weaknesses – difficulty maintaining expertise in a wide range of platforms

Opportunities – mass customization, which could open a new market for specialized games

Threats – open source availability of much game software



4. Analysis of Strategic Factors

This is the heart of the business case. In this section, factors that affect the achievement of the strategic goals are identified and discussed.

For example, the increasing variety of intelligent platforms opens new markets for our products, as long as we port to those platforms. The cost of the ports versus their affect on sales is analyzed.

AGM
Example



This section also contains the cost/benefit analysis that uses the strategic factors, as well as inputs from the scope and Marketing and Product Plan, to analyze the costs and benefits of each approach that has been defined.



5. Comprehensive Scenario Analysis (including cost/benefit analysis)

Structured Intuitive Model for Product Line Economics (SIMPLE)

- The business case argues for a particular development scenario by reducing the costs and benefits to a common denominator: money.
- Even the quality attributes of the products and the degree to which business goals are achieved are all reduced to money.
- SIMPLE provides a basic starting point from which a model can be developed.
- Each function in the model can be implemented in a variety of ways depending on the context of your product line.



Basic Costs

$$\sum_{i=1}^n C_{prod}(product_i, t) \quad (A)$$

The cost of building a product as a stand-alone product can be represented by Equation A above.

C_{prod} is a *cost function*. Given actual parameters, the function returns the portion of the cost of building *product_i* during the time interval *t*.

This expression is used to evaluate the approach in which the organization hires sufficient staff to build *n* standalone products. The value returned by C_{prod} includes all costs associated with that staff and other related costs (including severance costs for when the project is completed).



Cost Benefit Analysis – Cost Functions

$$C_{org}(t) + C_{cab}(t) + \sum_{i=1}^n (C_{unique}(product_i, t) + C_{reuse}(product_i, t)) \quad (B)$$

Four basic cost functions support the modeling of most of the costs of a software product line:

1. C_{org} returns the cost to an organization of adopting the product line approach for its products.
2. C_{cab} returns the development cost to develop a core asset base suited to satisfy a particular scope.
3. C_{unique} returns the development cost to develop unique software that itself is not based on a product line platform.
4. C_{reuse} returns the development cost to reuse core assets in a core asset base.

Rather than rigorously defined mathematical functions, these are invitations to do thought experiments to come up with a reasonable cost (or monetary benefit) estimate in each area.



Cost Benefit Analysis: Benefit Functions

$$\sum_{j=1}^{nbrBen} B_{ben_j}(t) \quad (C)$$

A general benefit function can be used to model as many benefits as necessary.

B_{ben} returns the value derived from a particular benefit.

- The difference between two release dates can be modeled as a difference in revenue.
- Increased quality or customer satisfaction might be modeled as increased revenue from increased sales and as a reduction in costs for handling trouble reports.

The benefit of reduced cost is captured in the cost functions.
Be careful not to count that benefit twice!



Example Scenario - 1

Cost of building the family in a stand-alone fashion:

$$\begin{aligned} & \rightarrow \sum_{i=1}^n \underline{C_{prod}(product_i, t)} & (A) \\ & = 5 \times 3PY \\ & = \underline{15PY} \end{aligned}$$

Suppose

- There are 5 products, all roughly the same size and complexity:
 $n=5$
- Each product costs about 3 person-years (PY) to build as a stand-alone project:

$$C_{prod}(product_i, t) = 3PY$$



Example Scenario - 2

Cost of building the products as a software product line:

$$C_{org}(t) + C_{cab}(t) + \sum_{i=1}^n (C_{unique}(product_i, t) + C_{reuse}(product_i, t)) - \sum_{j=1}^{nbrBen} B_{ben_j}(t) \quad (D)$$

$$= 1PY + 2PY + 5(1.5PY + 0.225PY)$$

$$= \underline{11.625PY}$$

Suppose

- We ignore benefits for now.
- It takes 1 PY to turn the organization around: $C_{org} = 1PY$
- It takes 2 PY to build the core asset base for these 5 products:
 $C_{cab} = 2PY$
- Each product is about 1/2 core assets and 1/2 unique content:
 $C_{unique}() = 50\% * 3PY = 1.5PY.$
- We take $C_{reuse}() = 15\%$ [1] of one-half of 3PY = $15\%(1.5PY) = 0.225PY$ [2].



Example Scenario - 3

Cost savings of the product line approach

= Equation A – Equation D

$$\sum_{i=1}^n C_{prod}(product_i, t) - (C_{org}(t) + C_{cab}(t) + \sum_{i=1}^n (C_{unique}(product_i, t) + C_{reuse}(product_i, t))) \quad (E)$$

= 15 PY –

11.625 PY

= **3.375 PY** (ignoring benefits)



Example Scenario - Footnotes

[1] This value is the accepted value for opportunistic reuse. A product line will have a smaller value than 15%, because there is no search or qualification cost. Boehm's COPLIMO model uses a figure of around 5%. We use 15% as a very conservative estimate in this example but suggest that each company measure its local cost.

[2] C_{unique} and C_{reuse} are both functions of the fraction of each product that the core asset base provides. As that fraction rises, C_{unique} decreases and C_{reuse} increases.

[3] This savings is calculated over a single time period (the period of building the products), and the scenario does not take into account any changes to the products or core assets over time.

A web site is available for modeling product line scenarios using SIMPLE:

<http://simple.sei.cmu.edu>



A Return to Scoping

The analyses used to build the business case may not always show that a product line is the best course of action.

The volatility of the domain may simply not support the use of the product line strategy.

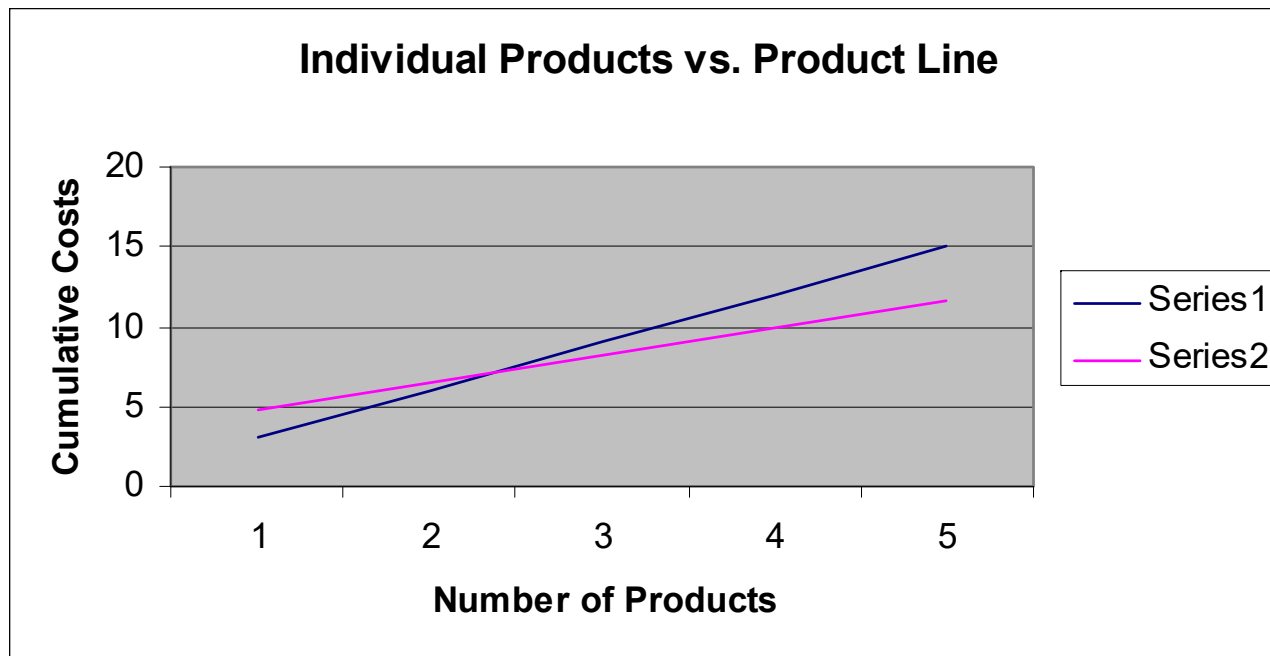
The variation among products may be too great. By narrowing the scope, the variation is reduced, and the product line approach may look better when you run the numbers a second time.



6. Recommended Course of Action

The final section of the business case summarizes the analyses of the previous sections and recommends a course of action.

The recommended actions are justified by the contents of the other sections in the business case.





Schedule: Day 1

8:30 – 9:00	Introductions, Background, and a Product Line Refresher
9:00 – 9:15	The Adoption Factory Pattern
9:15 – 9:30	The What to Build Pattern
9:30 – 10:00	Building A Business Case for a Software Product Line
→ 10:00 – 10:15	BREAK
→ 10:15 – 12:00	Group Exercise: Writing a Business Case
12:00 – 13:00	LUNCH
13:00 – 14:30	The Cold Start Pattern
14:30 – 14:45	BREAK
14:45 – 16:30	Group Exercise: Writing a Concept of Operations



Session Outcomes

After this session, you should

- understand what is needed to construct a business case that supports adoption of a software product line strategy
- be able to structure those analyses in a document



► Exercise: Build a Business Case for AGM

Goal: Explore the development of the business case for the AGM product line.

Preparation: In the Background book, take about 15 minutes to read

- the AGM Product Line Overview
- the AGM Market and Product Plan
- the AGM Scope document
- the partial AGM Business Plan that your team will complete during this exercise.

Activity: Form teams of size 3-4. Take about 45 minutes to complete the partial business case in the Exercise book

- Complete section 3 (SWOT analysis), page 5
- Complete sections 5.1.1, 5.1.2, and 5.1.3, pages 8 and 9. Record any assumptions you make.
- Complete section 5.3 (table of SIMPLE formulas), page 12. Here you only need to provide relative sizes (e.g., “high,” “medium,” “low”) to compare the approaches against each other.



► Taking Stock of AGM's Situation

The What to Build pattern has revealed that software product lines are a sound strategy for AGM.

The VPPD's business case has convinced the CEO, CFO, and CTO who have given the go-ahead to plan the necessary changes. They are concerned about changing everything at once, so an incremental approach is called for.

At this point, we have the following artifacts:

- an initial scope definition
- a market forecast
- a technology forecast
- a business case for the product line

These are our first core assets!



What to Build Lives On



A software product line can operate for a number of years depending on the size and complexity of the products and the release schedule.

The market climate and technology predictions can change in ways that affect the viability of the product line.

The What to Build pattern is reapplied according to the usual planning cycle for the product line organization. Plans are reviewed and updated when appropriate.



Schedule: Day 1

8:30 – 9:00	Introductions, Background, and a Product Line Refresher
9:00 – 9:15	The Adoption Factory Pattern
9:15 – 9:30	The What to Build Pattern
9:30 – 10:00	Building A Business Case for a Software Product Line
10:00 – 10:15	BREAK
10:15 – 12:00	Group Exercise: Writing a Business Case
 12:00 – 13:00	LUNCH
 13:00 – 14:30	The Cold Start Pattern
14:30 – 14:45	BREAK
14:45 – 16:30	Group Exercise: Writing a Concept of Operations



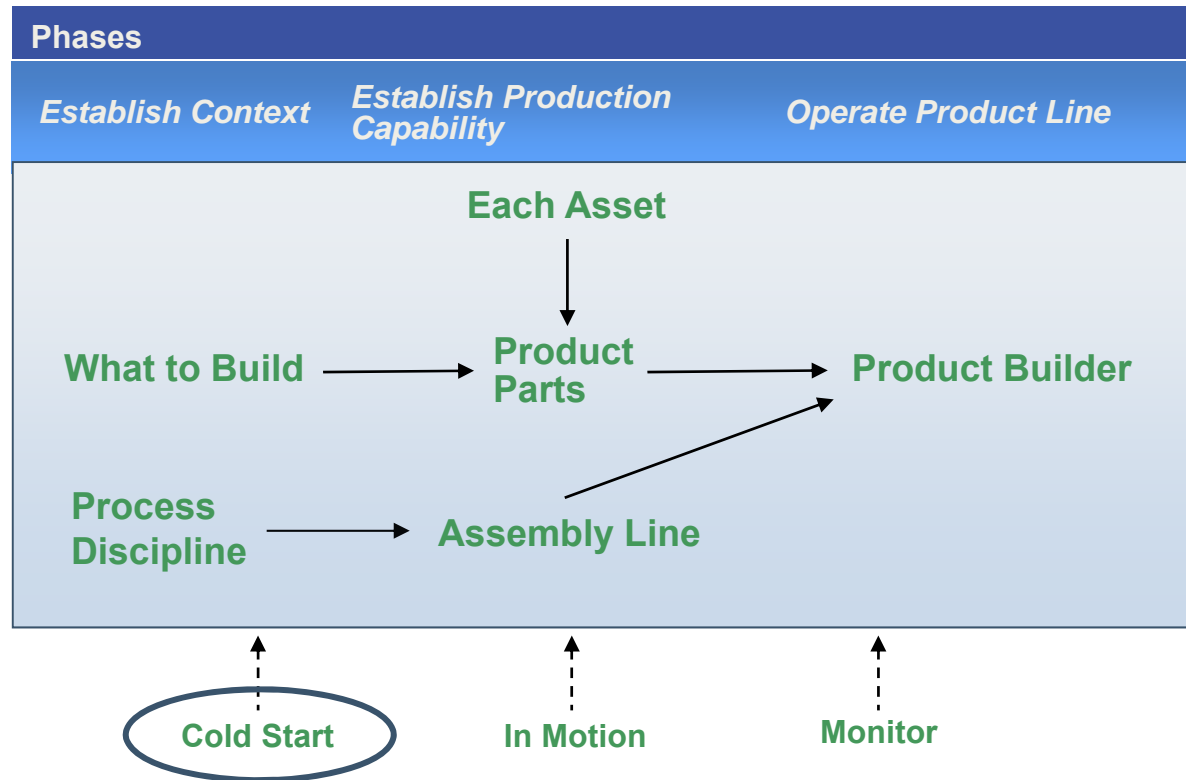
Session Outcomes

After this session you should

- understand when to use the Cold Start pattern
- know which practice areas are part of the Cold Start pattern
- have a good understanding of each constituent practice area
- understand how each practice area would be handled or carried out in the AGM example
- be ready to develop a Concept of Operations for a software product line organization



Reminder: Adoption Factory Pattern



→
•Informs and information flow

- - - - ->
•Supports

Dynamic Structure



Cold Start Pattern - 1

Name: The **Cold Start** pattern consists of practice areas that should be used when any organization is launching a software product line for the first time

Context: An organization is launching its first software product line.

Problem: To effectively prepare the organization for its first software product line production



Cold Start Pattern - 2

Solution: The organization must shape itself so it can effectively use a product line approach to turn out products.

Those in charge must

- fund the effort
- put in place an organizational structure that designates core asset developers, product developers, assembly line builders, and so forth
- provide training for the people involved
- prepare customers for the new approach
- develop an acquisition strategy for any suppliers that will be involved
- develop a Concept of Operations
- establish an organizational risk management program
- create a product line adoption plan



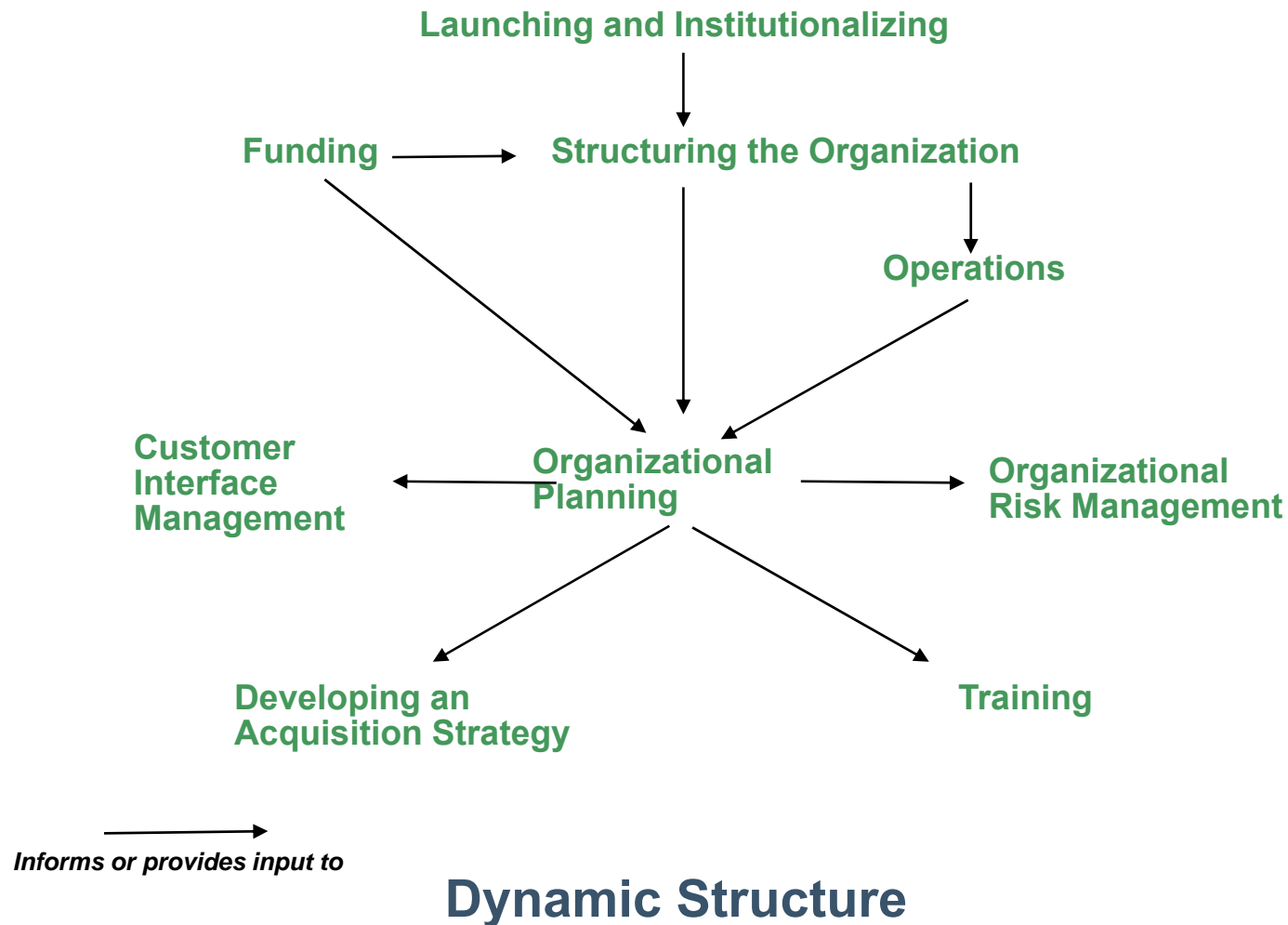
Cold Start Pattern - 3

Static Structure: The **Cold Start** pattern consists of the following practice areas:

- Launching and Institutionalizing
- Funding
- Customer Interface Management
- Developing an Acquisition Strategy
- Operations
- Organizational Planning
- Organizational Risk Management
- Structuring the Organization
- Training



Cold Start Pattern - 4





Organizational Planning



→
Informs or provides input to



Organizational Planning - 1

At this point, organizational planning gives us some advance warning about the plans that need to be produced:

- **product line adoption plans**
 - how to transition an organization to a product line approach
 - covered under the “Launching and Institutionalizing” practice area
- **core asset funding plan**
 - Funding the development and maintenance of the core asset base is likely to be done at the organizational level.
 - covered under the “Funding” practice area
- **configuration management (CM) plan**
 - CM reaches across all the core asset and product-building projects and possibly even across product lines.
 - Hence, it is appropriate to plan CM at the organizational level.



Organizational Planning - 2

- **tool support**
 - Tool support that is common across products in an organization should be planned organizationally.
- **training plan**
 - It pays to consider training at the cross-product level.
 - Include course content, who provides it, who attends, a schedule, and resource allocation.
 - covered under the “Training” practice area
- **organizational structure plan**
 - details the transition steps and shifts in responsibility, outlines any logistical or physical relocation, and assigns schedules and resources
 - covered under the “Structuring the Organization” practice area



Organizational Planning - 3

- **risk management plan**
 - assigns people to participate in the process, accounts for any training or other preparation required, and lays out an engagement schedule
 - covered under the “Organizational Risk Management” practice area
- **priorities for core asset development**
 - resolve competing needs among projects when new core assets become available
 - especially important when new core assets are being developed



Organizational Planning - 4

Organizational planning also helps us understand that our plans will have interdependencies induced by the product line approach:

- Organizational plans will have dependencies with project plans.
- Project plans will have external dependencies among other project plans.
- Organization-level plans may be necessary to coordinate project-to-project dependencies.
- In a product line context, the project plans can relate to core asset development, product development, or the activities that cross between them.



Organizational Planning - 5

Organizational plans themselves (or parts of them) make fine core assets.

Ideally, reusable plans should be tailorable in the same fashion as other core assets—that is, they have defined points of commonality and variation.

Cost, effort, and schedule estimates may be useful candidates, particularly for reuse, as are work breakdown structures, goals, strategies, and objectives.



► Organizational Planning for AGM

The VPPD

- makes sure that key players' planning skills are up to the task
- makes sure that planning tools are in place and people know how to use them
- builds a list of organizational plans that that AGM will need, and uses this list as a checklist later
- makes sure to remember the dependencies among plans so that when planning tasks are assigned, they take into account those dependencies.



Launching and Institutionalizing



→
Informs or provides input to



Launching and Institutionalizing

This practice area is about systematic elevation of an organization from a given state of product line sophistication to a higher state of product line sophistication.

Carrying it out involves the timely application of the other practice areas and patterns (especially Adoption Factory), as appropriate to the needs and capabilities of an organization

Launching a product line is a type of technology change project, undertaken to help organizations prepare themselves to adopt a new technology or a new way of doing business.

How to make the change is highly dependent on the context of the organization; an invariant sequence of steps to execute the project is inappropriate.



Launching and Institutionalizing: Specific Practices - 1

Use pilot projects to

- Reduce risk.
- Learn more about the organizational and technical issues associated with the product line effort.
- Build advocacy.
- Run a controlled experiment to test specific ideas or concerns.



Launching and Institutionalizing: Specific Practices - 2

Use an organizational diagnostic, such as the SEI Product Line Technical Probe,SM to

- Find the organization's blind spots.
- Find out where the organization lacks necessary expertise in one of the practice areas (especially one that tends to manifest itself early in the product line life cycle, such as scoping or requirements engineering).
- Learn where to focus resources judiciously.
- Provide input for an adoption plan.

SM Product Line Technical Probe is a service mark of Carnegie Mellon University.



Launching and Institutionalizing: Specific Practices - 3

Develop product line goals, objectives, and strategies:

- Establish an appropriate set of goals, which are validated by supporting rationale.
- Determine objectives and high-level, measurable indicators of progress toward those goals.
- Consider holding an internal workshop for articulating and capturing the goals, objectives, and strategies that will serve as the foundation for building an adoption plan.
- Revisit those goals, objectives, and strategies as the adoption progresses.



Launching and Institutionalizing: Specific Practices - 4

Create a suitable Product Line Adoption Plan that

- describes how product line practices will be rolled out across the organization
- may provide the definition of processes, the initiation of practice areas (e.g., “Training”), or the selection and implementation of pilots
- should address the entire organization. For example, what does everyone else do while the pilot project is running?
- is a primary result of the launching and institutionalizing effort



Launching and Institutionalizing: Specific Practices - 5

Initiate process improvement as a basis for launching and institutionalizing:

- Process discipline provides a foundation for product line practice.
- If an organization doesn't have sufficient process discipline, launching a product line is going to be problematic.



► L&I Outcomes for AGM

We help the VPPD to make the following decisions related to launching and institutionalizing product line practice:

- Keep working with patterns.
- Launch a training program via the “Training” practice area.
- Hold an internal workshop to establish goals and objectives, and increase buy-in. Use the goals and objectives from the business case as a “starter” set.
- Use that workshop to choose which products become members of the product line first.
- Determine what other product teams do in the meanwhile.
- Write an adoption plan based on all of the above.

The VPPD also decides to take the SEI Adopting Software Product Lines course.



Launching and Institutionalizing: Discussion

1. What *should* second-wave product teams at AGM be doing in the meanwhile?
2. Who should attend AGM's goals-and-objectives workshop?

AGM
Example



3. How would you incrementally “roll out” product line practice across the entire organization once the pilot project has served its purpose?



► What's Next for AGM?

The VPPD now has an idea for how the adoption of software product line practice is going to be rolled out across AGM.

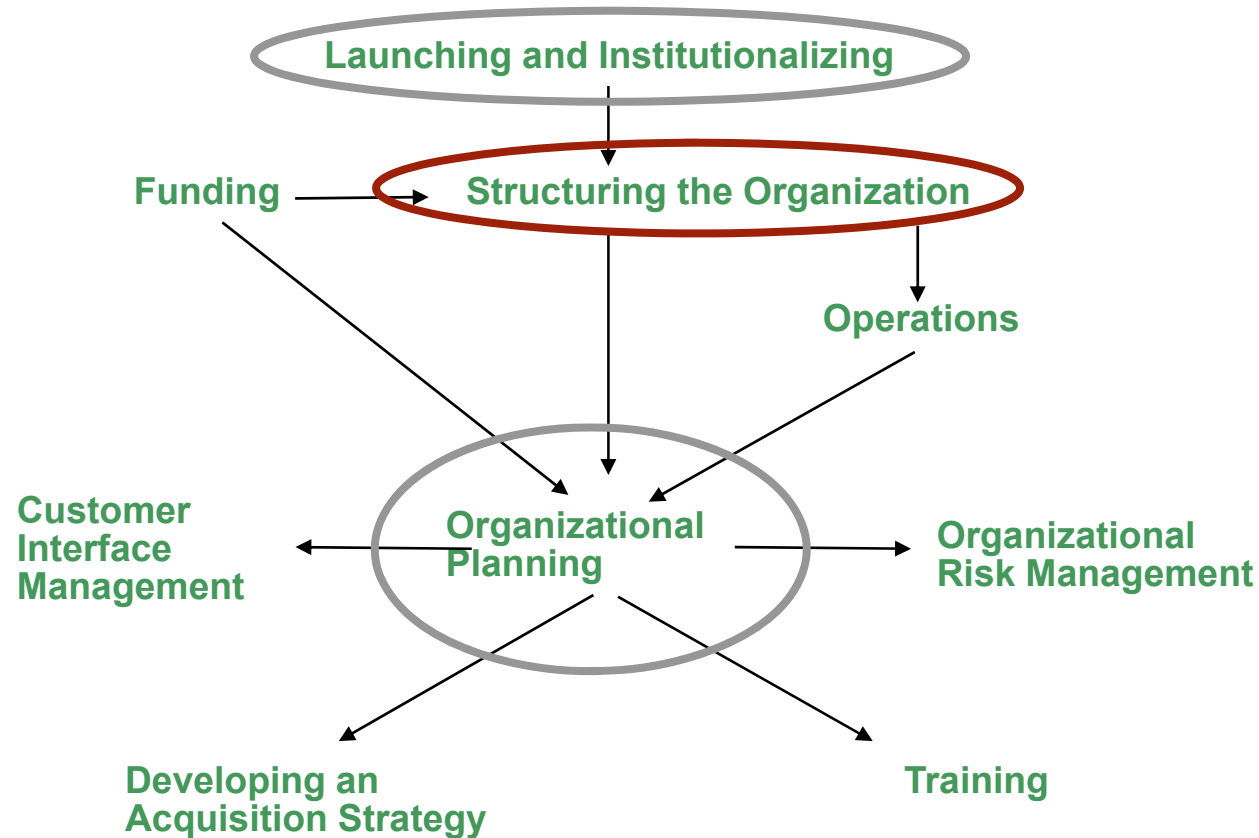
Some decisions are tentative and will need to be revisited. For instance, choosing which products are going to be in the first wave may depend on setting up the core asset base, which hasn't been done yet. But educated guesses are possible.

The VPPD now wants help in deciding what the major organizational units of his organization and their responsibilities are going to be. Hence, we next turn our attention to

Structuring the Organization



Structuring the Organization



→
Informs or provides input to



Structuring the Organization - 1

An organizational structure should be chosen to at least determine which unit or units

- produce and maintain the **architecture** for the product line
- determine the **requirements** for the product line and product line members
- design, produce, and maintain the product line's **core assets**
- produce **products**
- **determine the processes** to be followed and measure or ensure compliance with them
- maintain the **production environment** in which products are produced
- **forecast new trends, technologies**, and other developments that might affect the future of the product line



► Structuring the Organization at AGM

Recall that the “Organizational Planning” practice area has alerted us to the need for the following plans and organizational units to produce and implement them:

- product line adoption
- core asset funding
- configuration management
- tool support
- training
- structuring the organization
- risk management
- priorities for core asset development

The VPPD will
create these plans.



Structuring the Organization - 2

Structuring the organization is *largely* about answering the following questions:

- Who develops and maintains the core assets?
- Who develops and maintains products?

Two main organizational structures occur most often in product line organizations (although several other variations are possible):

1. a separate core asset group
2. an integrated product and core asset group

Structuring the Organization: Discussion

1. What effects do you think the following have on choosing an organizational structure?
 - size of the organization
 - number of products (and product projects)
 - volatility of core assets
 - amount of manual coding or unique code each product requires
 - time to market requirements
 - organizational process maturity
 - sophistication of development tool support
2. What artifact(s) result from making this decision?

3. Which organizational structure for AGM do you recommend to the VPPD?

4. What will you measure or look for to see if AGM made the right decision?

**AGM
Example**





► What Tangibles Do We Have So Far? - 1

We have

- an initial scope definition
 - a market forecast
 - a technology forecast
 - a business case
- } What to Build
- a blueprint for the organization based on patterns
 - an intent to launch a training program
 - a set of goals and objectives for the product line
 - an identified set of stakeholders invested in the successful rollout of product line practice
 - an adoption plan, describing a planned phase-in of the product line strategy across products
- } Launching and Institutionalizing
- an organizational chart and a charter for each unit
- } Structuring the Organization



► What Tangibles Do We Have So Far? - 2

We have

- an initial scope definition
- a market forecast
- a technology forecast
- a business case

} a picture of where we are

- a blueprint for the organization based on patterns
- an intent to launch a training program
- a set of goals and objectives for the product line
- an identified set of stakeholders invested in the successful rollout of product line practice
- an adoption plan, describing a planned phase-in of the product line strategy across products

} a picture of what our organization will look like in the future

- an organizational chart and a charter for each unit

} a vision of our organization's capabilities



► What's Next for AGM?

Given an organizational structure decision, the VPPD is now in a position to turn to any of several possibilities in the Cold Start pattern, in order to decide how the units

- are to be funded
- should manage organizational risk
- are going to present a unified, consistent interface to the customer
- are going to jointly manage acquisition policy

All of this is leading up to a definitive description of the day-to-day operations of the product line—an operational concept.



Funding



→
Informs or provides input to



Funding

Funding is largely about deciding who pays for the core assets.

Many schemes are possible. Major ones include

- first product(s) pay
- taxing every product
- taxing every product group
- using R&D or other corporate funds



Funding Strategies

<div style="text-align: center;"> Activities to be Funded </div> <div style="text-align: center;"> Funding Strategies </div>		Planning and Analysis	Product Line Development		Product Line Sustainment and Evolution	Product Development Sustainment and Evolution
			Infrastructure Development	Asset Development		
1	Product-specific funding (individual customer, for example)			XX		XXX
2	Direct funding from corporate sponsor/program	XXX	XX	XX		
3	Product line organization's discretionary funds	X	XXX	X	X	
4	First product (project) funds effort	XX	XX	XX	X	XXX
5	Multiple projects banded together to share costs	XXX	XX	XXX	XX	X
6	Taxing of participating projects		X	X	XXX	
7	Product-side tax on customers			X	XXX	
8	Fee based on core asset usage				XXX	
9	Prorated cost recovery		X	XX	X	X



Funding: Discussion

1. What funding model do you think the VPPD should recommend for AGM? (Remember that the funding model can change as the product line grows and evolves.)

AGM
Example



2. How does our selection of an organizational structure constrain or affect our funding model choices?
3. What artifact(s) result from making this decision?

4. What will you measure or look for to see if AGM made the right decision?

AGM
Example





Funding: Final Word

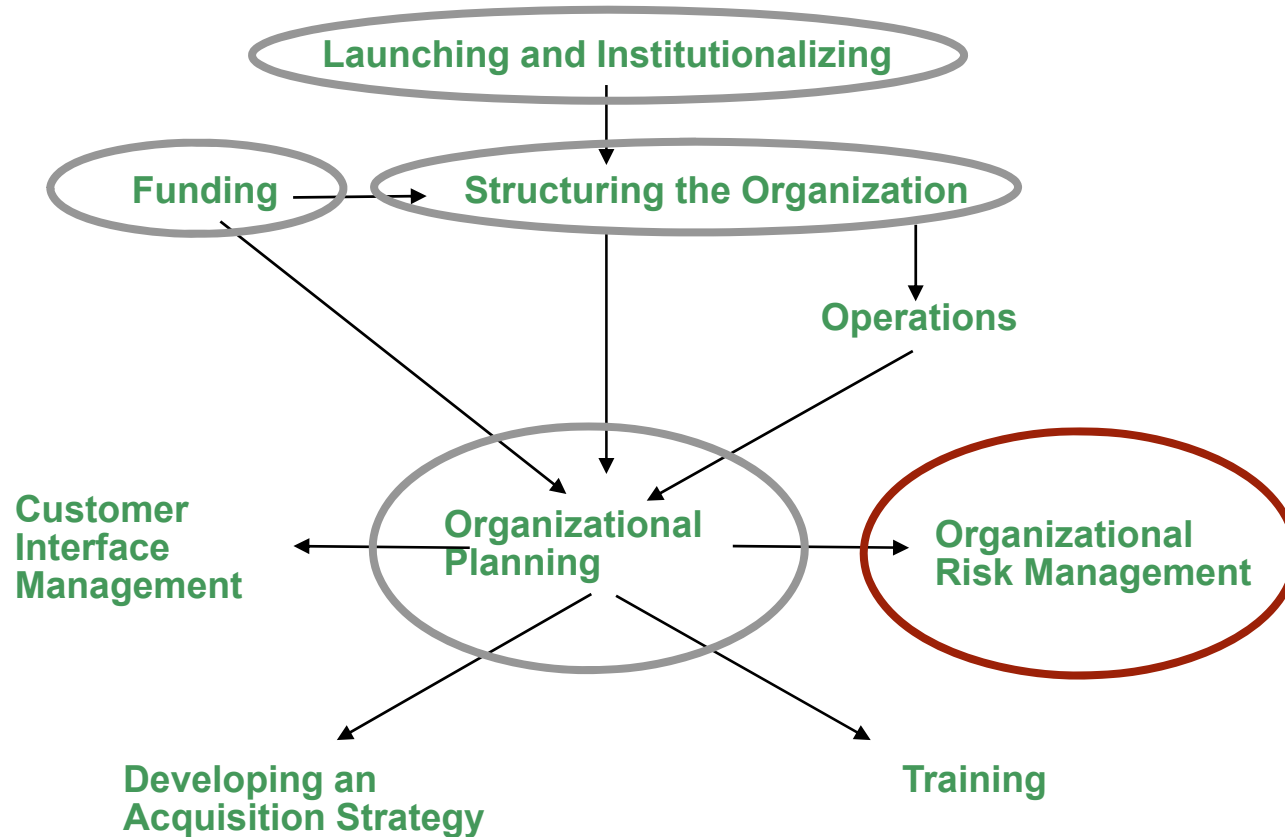
A funding policy should be chosen that will further the goals of the overall software product line strategy.

The rationale for the choice should be communicated clearly to the parties involved. Ideally, they should have a voice in setting the policy.

Measures should be put in place to see if the goals are being met. If they are not, the funding policy should be revisited.



Organizational Risk Management

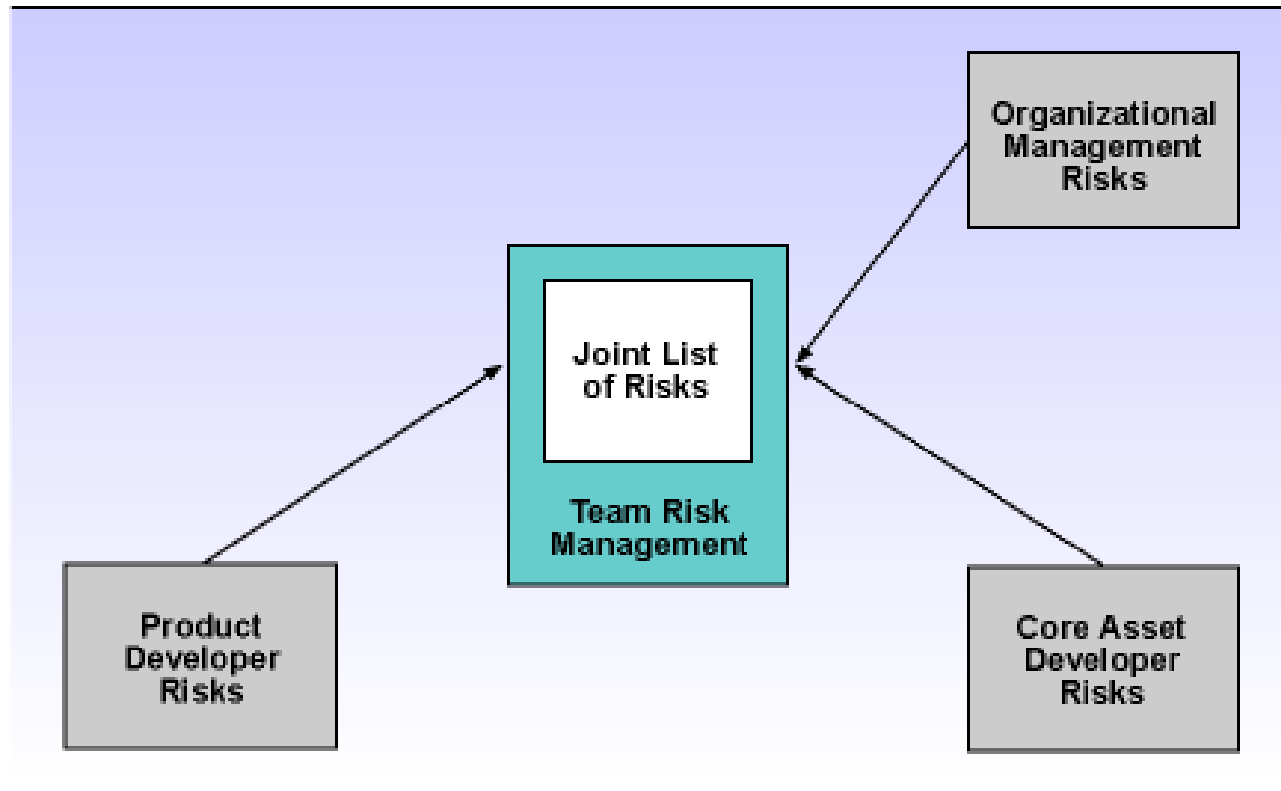


→
Informs or provides input to



Organizational Risk Management

Recommended practice: Team Risk Management





Organizational Risk Management: Discussion

1. Who should be on the organizational risk management team?

AGM
Example



2. What activities should they perform? How often?
3. What artifact(s) result from this practice area?



Customer Interface Management





Customer Interface Management - 1

In all organizations, there is a need to understand and manage the commitments between producers and customers.

What the customer sees needs to be managed:

- Who are the customer representatives and what are their customer interface responsibilities?
- What are the standard product offerings and the preplanned feature variation?
- What are the corresponding cost, schedule, and quality benefits?
- What is the product line strategy for future features and evolution?



Customer Interface Management - 2

The ground rules for transacting business need to be defined:

- What protocol must be followed, and what policies and procedures apply?
- How are customer requirements to be negotiated and managed?
- How will a disciplined interface with the customer be enforced?

The customer representatives of a product line organization typically include

- marketers
- a product manager
- domain experts
- a users group coordinator
- other individuals explicitly assigned roles and responsibilities that require them to interface with the customer.



Customer Interface Management - 3

In a product line approach

- Customers are urged to choose from standard product offerings, relinquishing some flexibility in return for cost and time-to-market advantages.
- Specialized requirements can be accommodated but must be considered individually and have special cost and schedule implications to which both parties must agree.
- Strict product line organizational interfaces are enforced.
- Customers may form user groups to give the *market* a voice and drive requirements for product evolution jointly.
- The organization should establish centralized product support for customers.



Customer Interface Management: Discussion

1. What are some of the ways that marketers in a software product line organization will behave differently?
2. What artifact(s) result from this practice area?
3. To what extent should customers be discouraged from requesting new features? To what extent should they be encouraged?

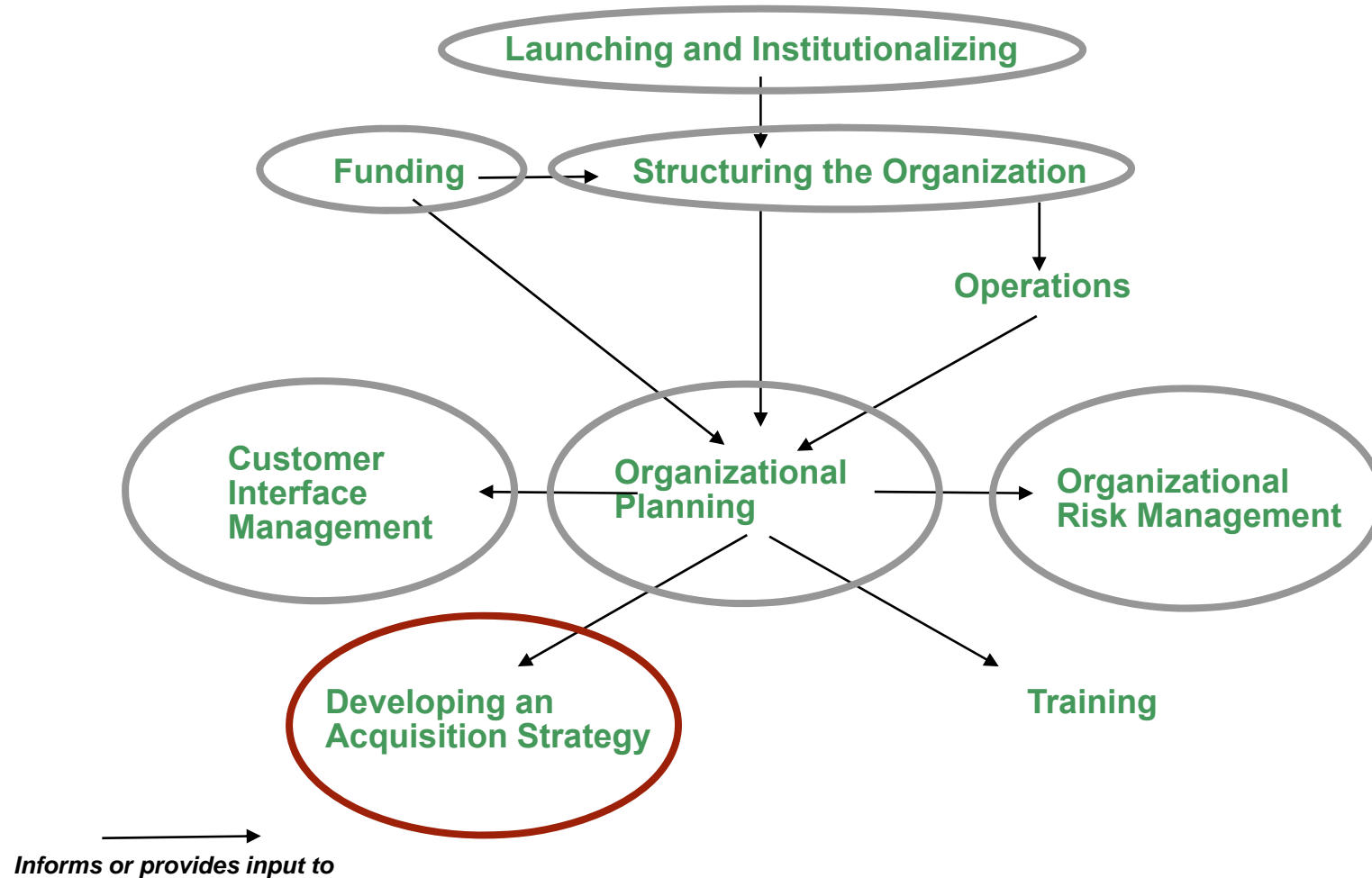
4. Who are AGM's customers? What would the customer interface for those people look like?

AGM
Example





Developing an Acquisition Strategy





Developing an Acquisition Strategy

This practice area is about managing our organization's purchases and subcontracts (if any) as a coherent whole, across the entire product line.

We want to prevent individual product teams from interfacing directly with suppliers in a conflicting way.

In the AGM product line, our major acquisitions are tools—AGM acquired Microsoft Visual Studio for the first increment's implementation. Palm OS's SDK, which is implemented in Eclipse, will be acquired for the second increment. (It is open source, but AGM still takes an acquisition approach to bring the tool into the development environment.) The VP of Purchasing is responsible for developing this, see Memo05-01 in the Background Material.

**AGM
Example**





Developing an Acquisition Strategy

The acquisition strategy team will be informed by

- the product line scope
- the chosen funding strategy
- the architecture for existing and planned products
- technology forecasts

Discussion:

1. What is our result? What do we recommend to the VPPD?

AGM
Example





Training





Training - 1

The focus is on establishing a core competence in the creation and usage of core assets:

- It is not enough, for example, to send people to a course in object-oriented technology or software reuse and then expect them to build product lines.
- Product line training must be viewed as a strategic activity that should be planned accordingly.

The primary outcome is a training plan that

- identifies training needs
- describes how those needs will be met
- addresses how skills will be maintained
- addresses how the new training is related to existing training: does it augment the existing training? Replace it? Add to it?



Training - 2

Possible training elements for a software product line approach include

- introductory course on product line concepts and terminology
- overview of the organization's current and planned product lines
- overview of the proposed development process, including changes in existing processes, organizational structure, and roles
- presentation of the Concept of Operations (CONOPS)
- training in specific product line practices or concepts (e.g., software architecture)
- training in supporting technologies (e.g., tools)



Training - 3

Possible training implementation choices include

- in-house courses
- external courses
- short tutorials, brown-bag lunches
- Webcasts
- all-hands meetings
- pilot projects
- mentoring



▶ Summary of AGM's Training Needs

What training needs/opportunities have we identified so far?

- AGM's business case for software product lines: why we're doing what we're doing
- the rollout (adoption) plan from the "Launching and Institutionalizing" practice area
- the goals and objectives from the "Launching and Institutionalizing" practice area
- our new organizational structure, roles, and responsibilities
- our funding policy
- risk management
- what our planning documents are and how they fit together
- customer interface procedures



► Training for AGM: Discussion

So far there is only a memo from HR setting policy on training for AGM (see Memo04-05 in the Background Materials). The functional team leads must work with HR to plan training.

Propose a training curriculum for AGM:

- Identify training needs.
- Develop a training inventory: a series of training elements.

For each training element, say

- what it is and what it conveys
- the format of the training (e.g., course, tutorial, meeting)
- who should attend it
- who should teach it
- how long you expect it to last (e.g., one hour, one day, one week)



Operations





Operations - 1

The end result is to produce a CONOPS:

- It describes the day-to-day operation of the product line organization so that everyone works towards the same goals.
- Without it, the organization is mostly just a collection of staffed units, poised and willing to do the right thing, but unsure of precisely what that is.

Who is responsible for the CONOPS?

- usually the product line manager who must make sure it's well documented, effectively communicated, and closely followed



Operations - 2

The CONOPS puts together the organization's management strategies, policies, and practices with respect to the product line. It

- facilitates a common understanding among members of the organization as to how products are fielded and how the production capability is evolved and maintained
- serves as a baseline when the organization considers alternatives in its approach as warranted by changing conditions
- describes how the organizational units work together to execute their defined processes
- spells out how the organizational structure populates and nurtures the core asset base and how it uses the core asset base to build products



A Sample CONOPS Table of Contents -1

1 Overview

- 1.1 Identification
- 1.2 Document Map
- 1.3 Concepts
- 1.4 Readership
- 1.5 Product line guiding principles
- 1.6 Vision for this product line
- 1.7 Reference models
- 1.8 Metrics for success

2 Operations

- 2.1 Organizational structure
- 2.2 Communication
- 2.3 Operational tools

3 Core Asset Development

- 3.1 Core Asset Building Roles
- 3.2 Production constraints
- 3.3 Implications of the Production Strategy
- 3.4 Attached processes
- 3.5 Core asset development scenarios
 - 3.5.1 Core asset acquisition
 - 3.5.2 Core asset packaging and delivery
 - 3.5.3 Core asset sustainment



A Sample CONOPS Table of Contents -2

4 Product Development

- 4.1 Product Building Roles
- 4.2 Implications of the Production Strategy
- 4.3 Using the core assets
- 4.4 Modifying the core assets
- 4.5 Providing feedback about assets
- 4.6 Product development scenarios
 - 4.6.1 Constructing a product using only assets
 - 4.6.2 Constructing a product with unique requirements

5 Management

- 5.1 Launch and Adoption Strategies
- 5.2 Management roles
 - 5.2.1 Vice Presidents
 - 5.2.2 Product Line Manager
 - 5.2.3 Team Leader
 - 5.2.4 Project Lead
- 5.3 Organizational structure, roles and responsibilities
- 5.4 Challenges and risks to successful implementation
- 5.5 Continuous improvement
- 5.6 Management scenarios
 - 5.6.1 Add product to the product line
 - 5.6.2 Begin developing a product
 - 5.6.3 Assessing progress



A Sample CONOPS Table of Contents -3

6 Interconnections

- 6.1 Core assets and Product building
- 6.2 Core assets and Management
- 6.3 Product Production and Management
- 6.4 Escalation
- 6.5 Risk management

Appendix: Operational Procedure Resolutions



What Else Does Operations Entail? - 1

The product line manager should

- define and communicate the **product line vision**
 - by posting the vision publicly (e.g., on the Web); crafting slide presentations for organizational and technical managers; and discussing the vision at opportune times, from brown-bag lunches to management seminars
- create personal and organizational **performance objectives** that embrace the product line goals and strategies
- establish **promotion and reward structures** that provide real benefits to individuals who
 - follow the documented product line approach to building products
 - design and build core assets that are, in fact, reusable
 - contribute to the improvement of the product line effort



What Else Does Operations Entail? - 2

The product line manager should (cont'd.)

- communicate **product line progress** early, openly, and often
- **remove from the critical path** individuals who are barriers to product line success
- **champion** the product line at higher levels of management



Using Scenarios to Define Operations

Scenarios can be created to help the organization understand the product line operations. Examples include

- using the product line architecture and other core assets to build a product
- developing product-specific assets
- submitting new or modified components to the core asset base
- updating the core asset base and migrating the changes into existing or in-development products
- determining whether a candidate product is in or out of the product line scope
- delivering product line systems to customers
- supporting the implementation and maintenance of the development and execution environments for product line systems



Schedule: Day 1

8:30 – 9:00	Introductions, Background, and a Product Line Refresher
9:00 – 9:15	The Adoption Factory Pattern
9:15 – 9:30	The What to Build Pattern
9:30 – 10:00	Building A Business Case for a Software Product Line
10:00 – 10:15	BREAK
10:15 – 12:00	Group Exercise: Writing a Business Case
12:00 – 13:00	LUNCH
13:00 – 14:30	The Cold Start Pattern
→ 14:30 – 14:45	BREAK
→ 14:45 – 16:30	Group Exercise: Writing a Concept of Operations



Schedule: Day 1

8:30 – 9:00	Introductions, Background, and a Product Line Refresher
9:00 – 9:15	The Adoption Factory Pattern
9:15 – 9:30	The What to Build Pattern
9:30 – 10:00	Building A Business Case for a Software Product Line
10:00 – 10:15	BREAK
10:15 – 12:00	Group Exercise: Writing a Business Case
12:00 – 13:00	LUNCH
13:00 – 14:30	The Cold Start Pattern
14:30 – 14:45	BREAK
→ 14:45 – 16:30	Group Exercise: Writing a Concept of Operations



Session Outcomes

After this session, you should be able to

- formulate the operational rules for a software product line organization
- capture those rules in a Concept of Operations (CONOPS) document



► Exercise: Write a CONOPS for AGM

Goal: To gain experience defining the concept of operations for a product line

Actions:

- Form teams of at least 4 people
- In the Exercise book, read the exercise overview.
- Agree who will play which roles.
- Each person reads his or her role description(s). Do *not* read other role description(s).
- Complete the partial CONOPS document in three places by playing your roles according to the scripts..



Developing Software Product Lines

DAY 2





Schedule: Day 2

→	8:30 – 9:15	Developing the Core Asset Base: The Product Parts Pattern
	9:15 – 10:00	Architectures for Product Lines
	10:00 – 10:15	BREAK
	10:15 – 12:00	Exercises: Architecture and the Core Asset Base
	12:00 – 13:00	LUNCH
	13:00 – 13:30	Production Plans
	13:30 – 15:00	Group Exercise: Writing a Production Plan
	15:00 – 15:15	BREAK
	15:15 – 15:45	Other Patterns
	15:45 – 16:15	Group Discussion : Measurement
	16:15 – 16:30	Conclusions, Q&A



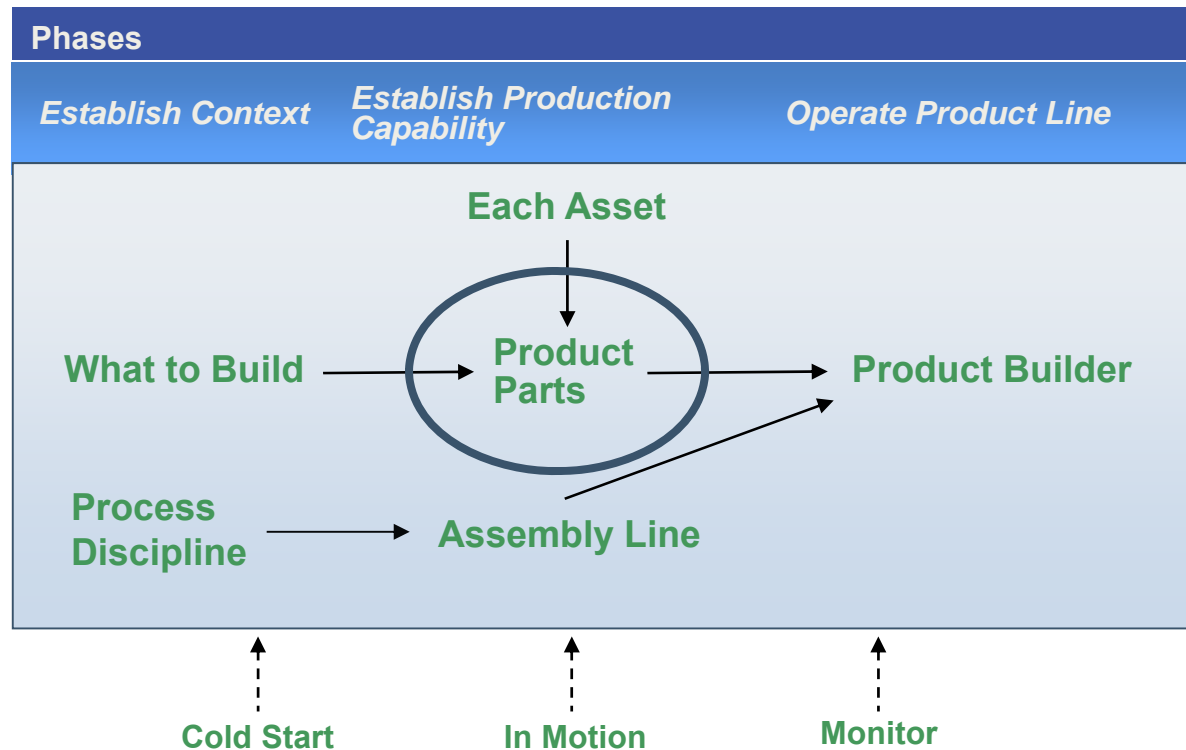
Session Outcomes

After this session, you should

- understand the practice areas needed to produce the core assets
- be able to evaluate the options for populating the core asset base



Reminder: Adoption Factory Pattern



→
•Informs and information flow

- - - - ->
•Supports

Dynamic Structure



Product Parts Pattern - 1

Name: Product Parts

Context: An organization knows what products are to be included in the product line and has designated knowledgeable individuals or groups to develop the core assets that will be used to develop the products.

Problem: To develop the core assets that will be joined together to form the products in the software product line



Product Parts Pattern - 2

Solution: The core assets of interest to this pattern include

- the product line requirements
- the product line architecture
- the components
- their reusable test-related artifacts

Each core asset needs to be equipped with an attached process.

The best source for each component needs to be determined (built, mined, bought, contracted).

Each core asset needs to be tested, and the suite of core assets needs to be integrated and tested.



Product Parts Pattern: Static Structure

The Product Parts pattern is composed of four subpatterns:

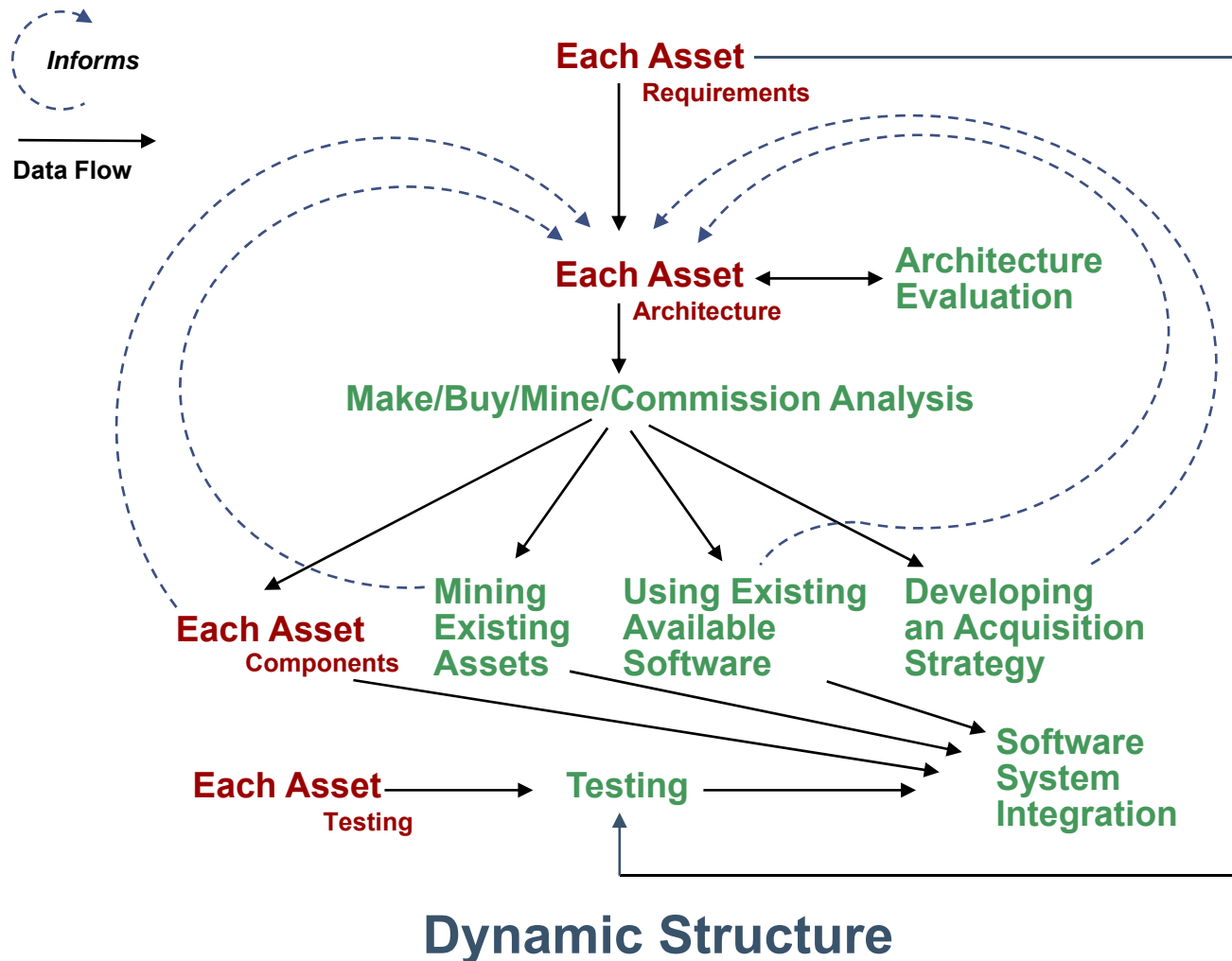
- Each Asset for requirements
- Each Asset for architecture
- Each Asset for components
- Each Asset for test-related artifacts

and seven practice areas:

- Architecture Evaluation
- Make/Buy/Mine/Commission Analysis
- Mining Existing Assets
- Using Externally Available Software
- Developing an Acquisition Strategy
- Testing
- Software System Integration



Product Parts Pattern - 3





Lecture Outline

The Product Parts pattern involves the

- ➔ • **Each Asset subpattern**
- Seven practice areas
 - “Architecture Evaluation” practice area
 - “Make/Buy/Mine/Commission Analysis” practice area
 - “Mining Existing Assets” practice area
 - “Using Externally Available Software” practice area
 - “Developing an Acquisition Strategy” practice area
 - “Testing” practice area
 - “Software System Integration” practice area



Each Asset Subpattern

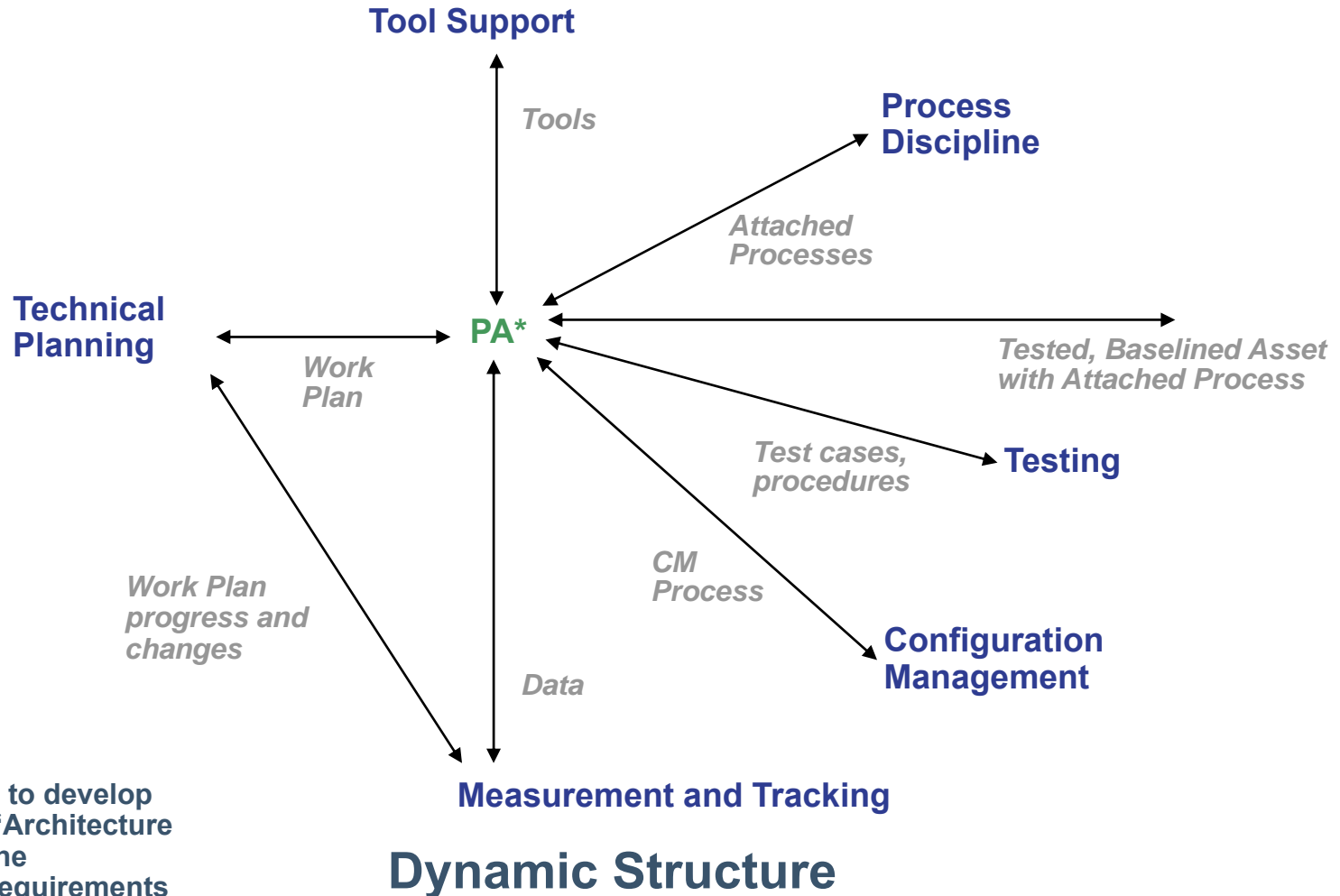
Each Asset defines those practice areas involved every time an asset is created, including

- Tool Support
- Technical Planning
- Process Discipline
- Testing
- Configuration Management
- Measurement and Tracking

We will discuss each very briefly in this context.



Each Asset Pattern



•PA* = practices to develop the asset, e.g., “Architecture Definition” for the architecture, “Requirements Engineering” for requirements, etc.



Tool Support

Each asset needs tool support. A tool may support the asset over several phases of its life cycle, such as a UML modeling tool for problem specification and solution design. A tool may be dedicated to a single activity in the life cycle, such as JUnit for testing. Configuration management tools handle all types of assets.

AGM has adopted Eclipse as the basic platform upon which a variety of tools can be built. Modeling, coding, and testing can be done in one environment. The fmp2 plug-in for feature modeling, the Omondo community version for UML modeling, Java development and JUnit test framework are all used. Functional team leads provide the tools for their particular skill area.

**AGM
Example**





Technical Planning

Technical plans should exist for creating and maintaining each asset. These plans give the schedule and the allocated resources for each activity. In particular, technical planning coordinates the schedules and resources being applied to each asset. This coordination allows for consistency and optimization.

AGM has a product line manager and core asset and product team leaders. These people are responsible for conducting technical planning for every asset and producing production plans, test plans, and development plans.

**AGM
Example**





Process Discipline

Each asset requires process discipline. In particular, each asset has an attached process that must be defined and followed. Each attached process is used to tell the product builders how to use the asset in building products. There will also be a process for building the asset itself.

AGM has a method engineering team (bringing together process, modeling languages, and tooling) that matrixes into teams of engineers who have experience in the technical content of the process. This team provides process definition knowledge, technology selection, and modeling expertise to the technical community.

AGM
Example





Testing

Each asset must be examined for correctness and other qualities. Testing provides a discipline that guides the evaluation of any asset. The evaluation can be guided by scenarios. The SEI Architecture Tradeoff Analysis Method[®] (ATAM[®]), a technique for Architecture Evaluation, uses scenarios to guide the examination of the architecture.

AGM has a comprehensive testing plan. Each phase of each process has exit criteria that must be passed before the next phase is entered. These criteria are usually evaluated using the “Testing” practice area. Individual developers conduct unit tests, integration and system tests are conducted by the testing team.

AGM
Example



[®] Architecture Tradeoff Analysis Method and ATAM are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.



Configuration Management

Each asset will be used to produce products, but not every asset will be used in every product. There is a need to manage multiple configurations of the assets. The product line strategy stresses traditional models of configuration management. Multiple assets, and multiple versions of those assets, must be shared among multiple product configurations.

AGM has been a traditional one-product-at-a-time company. Nothing was shared between projects. Each project was started on its own. Adopting the product line strategy will require AGM to establish a configuration management capability.

**AGM
Example**





Measurement and Tracking

Each asset must be tracked through its life cycle, especially the development portion. Metric data is collected to assist with decision making. Data collection provides input to future applications of the “Building a Business Case” and “Operations” practice areas. The Goal-Question-Metrics (GQM) approach helps define what to measure.

AGM has a minimal metrics program that produces mainly descriptive measures. The program will need to define appropriate metrics for each asset. Metrics are defined by almost every team in the organization to help them quantify their processes and products.

AGM
Example





Lecture Outline

The Product Parts pattern involves the

- Each Asset subpattern
- • **Seven practice areas**
 - “Architecture Evaluation” practice area
 - “Make/Buy/Mine/Commission Analysis” practice area
 - “Mining Existing Assets” practice area
 - “Using Externally Available Software” practice area
 - “Developing an Acquisition Strategy” practice area
 - “Testing” practice area
 - “Software System Integration” practice area



Architecture Evaluation

Identifying defects among the early design decisions made during architecture definition results in a large savings of time later in the development process.

The ATAM was developed by the SEI to analyze the tradeoffs made during architecture definition. Scenarios based on the required system qualities are used to explore and evaluate the architecture.

AGM's ATAM exercise determined the following:

The results of the ATAM showed the need to modify the existing architecture to include user and system models that allow the user to control more aspects of the game. The detailed design process will consider whether to do this through menu selections or a configuration file. The next step is to revise the architecture to accommodate these models.

AGM
Example





Make/Buy/Mine/Commission Analysis

Every asset comes into being through one of these actions. This practice area also provides feedback to practice areas such as “Architecture Definition.” The commercial availability of a particular product will influence the precise structure of the architecture. Decisions made in this practice area may be revised based on an architecture evaluation.

AGM has been able to take advantage of a number of open source products for its wireless set of products. Palm OS has released an IDE for building applications for the Palm. This is an Eclipse open source plug-in. It was used by core asset developers and product builders during the second increment.

AGM
Example





Mining Existing Assets

Most product lines are started by organizations that have already been producing software for some time. A number of techniques exist for reverse engineering and extracting architecture information from the existing code. Existing use cases, domain models, and architectures provide starting points for the corresponding product line assets.

AGM has been producing game software for sometime. It has a large inventory of legacy assets. While the user interface and operating system interface must be replaced, the core domain model and a number of the essential abstractions are applicable and useful. They will be recast in a new implementation using C# for the freeware games and Java micro edition for the wireless versions. The core asset development team has this responsibility.

**AGM
Example**





Using Externally Available Software

One approach to reducing time to market is to build products by integrating existing products to provide the required behavior. For example, the applications in the Microsoft Office suite are available with an application program interface (API) so that glue code can invoke the functionality of the applications. The “Using Externally Available Software” practice area provides feedback to the “Architecture Definition” practice area.

There are several game engines on the market, but among other weaknesses they require too much overhead for the wireless platforms. Therefore, a team of architects and core asset developers made the decision not to pursue any of them.

AGM
Example





Developing an Acquisition Strategy

The “Make/Buy/Mine/Commission Analysis” and “Using Externally Available Software” practice areas can require that the organization acquire assets that are not developed in-house. Acquisition requires organizational roles for interfacing with external sources, coordinating asynchronous release schedules, and certifying that deliveries meet specifications. These skills are needed whether the assets are purchased, commissioned, or open sourced.

AGM assigned an assistant product line manager to oversee these roles. Individual engineers with domain expertise were assigned to own assets such as the Palm IDE. Those engineers monitored new releases, tested them, and introduced them only at acceptable points in the product line life cycle. This strategy is used for open source software as well as vendor-supplied products.

AGM
Example





Testing

Every asset should be examined to ensure that it meets its requirements. Testing is an activity in which an asset is examined in a systematic manner by using the asset as it was intended to be used. Test cases (how the asset is to be used) and test data (offering a specific use of the asset) are large investments and must be managed for use over as many products and as many versions of the asset as possible.

AGM established a three-tier test program: (1) Assets are tested in isolation as they are created. (2) Their interactions with other assets are examined as the asset is integrated with other assets. (3) The final product is tested to ensure that it meets its requirements. AGM's goal is to find 90% of the defects at unit test time, using functional and structural tests. A strongly defined architecture helps enable this.

**AGM
Example**





Software System Integration

Software assets must be combined to produce the final product. The “Software System Integration” practice area relies on architecture definition to define interfaces for software assets and to ensure that the interfaces only allow compatible assets to be integrated. Integration handles the merging of assets from a variety of sources and at a variety of times. Integration may be achieved once an hour, day, or week. The sooner assets are integrated, the sooner the interaction effects of integrating two assets will become clear.

AGM defined interfaces between major architecture components. A modified Model-View-Controller architecture pattern is the basis for the products. Using a standard architecture pattern makes defining interfaces much easier. The architecture team made this selection.

**AGM
Example**





Schedule: Day 2

8:30 – 9:15	Developing the Core Asset Base: The Product Parts Pattern
→ 9:15 – 10:00	Architectures for Product Lines
10:00 – 10:15	BREAK
10:15 – 12:00	Exercises: Architecture and the Core Asset Base
12:00 – 13:00	LUNCH
13:00 – 13:30	Production Plans
13:30 – 15:00	Group Exercise: Writing a Production Plan
15:00 – 15:15	BREAK
15:15 – 15:45	Other Patterns
15:45 – 16:15	Group Discussion : Measurement
16:15 – 16:30	Conclusions, Q&A



Session Outcomes

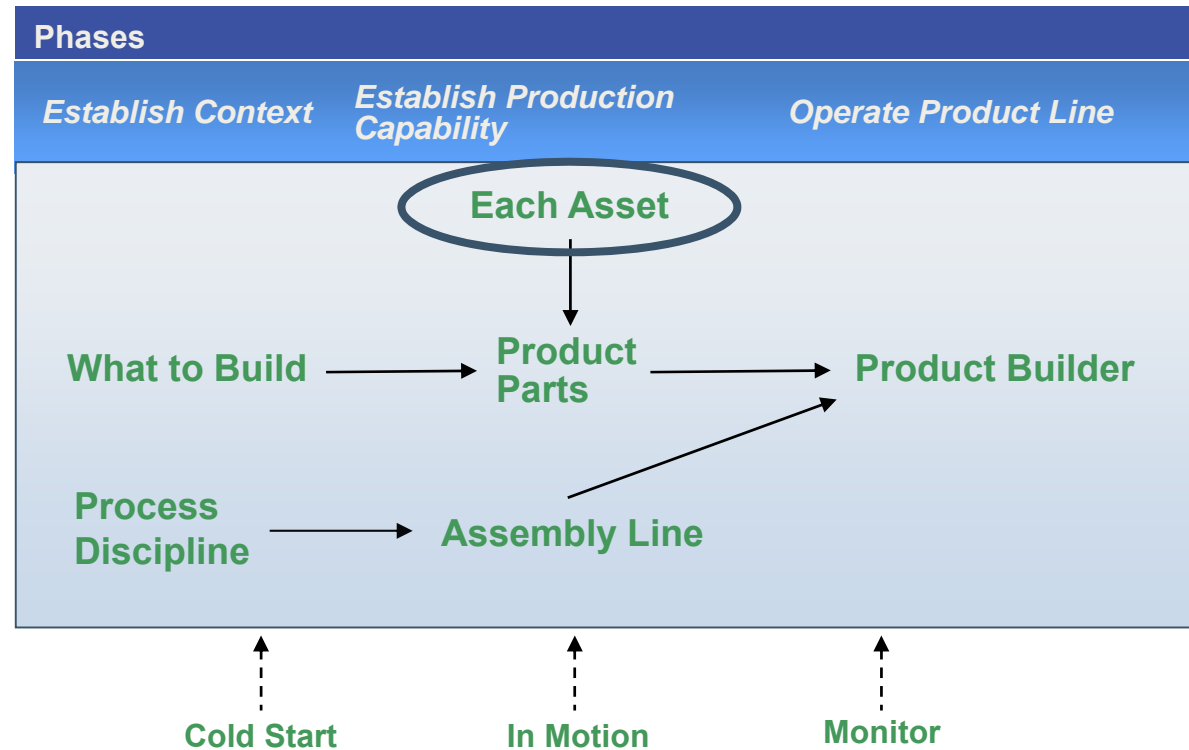
After this session, you should

- understand approaches to creating an architecture for a software product line
- know the major variation mechanisms available in architecture to support a product line
- be familiar with an approach to documenting an architecture
- be familiar with a method for evaluating a software architecture



Reminder: Adoption Factory Pattern

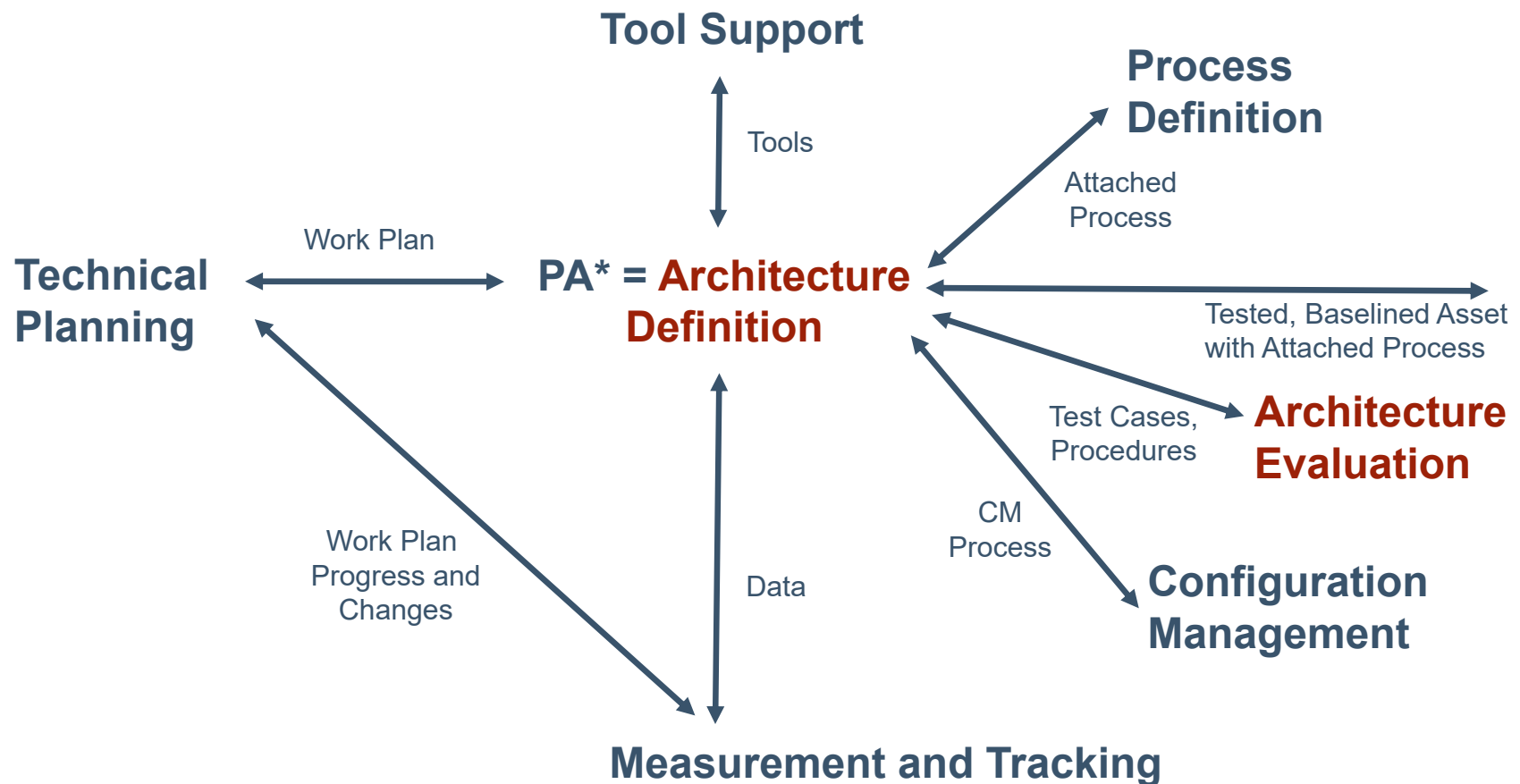
Let's look at the Each Asset pattern for Software Architecture.



Dynamic Structure

Architecture and the “Each Asset” Pattern

What must we produce?





Lecture Outline: Architecture-Related Practice Areas

➔ Architecture Definition

Architecture Evaluation

Other architecture-related needs identified in Each Asset

- configuration management (CM) plan for architecture
- tool support for architecture
- work plan for architecture
- data and metrics to track success of architecture
- attached process for architecture



Architecture Definition

This practice area refers to the creation and communication of the software architecture for the product line.

What do we mean by software architecture?

*The software architecture of a program or computing 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

¹ Bass, L.; Clements, P.; & Kazman, R. *Software Architecture in Practice, Second Edition*. Boston, MA: Addison-Wesley, 2003.



Abstraction

Architecture is an abstraction of a system. The architecture

- defines the system's elements and how they relate to one another
- suppresses details of what the elements do internally and purely local information about the elements; private details are not architectural.

A product line architecture therefore defines the software elements that constitute the core assets, as well as all of their supporting artifacts.



Multiple Structures

Systems can and do have many structures:

- No single structure can be *the* architecture. The set of candidate structures is not fixed or prescribed.
- Relationships and elements may be runtime
 - “sends data to,” “invokes,” or “signals”
 - processes or tasks
- Relationships and elements may be non-runtime
 - “is a sub-module of,” “inherits from,” or “is allocated to team X for implementation”
 - a class or library
- Different structures provide engineering leverage for different quality attributes



Architecture Is a Communication Vehicle

Architecture provides a common frame of reference in which competing interests may be exposed and negotiated. These interests include



- negotiating requirements with users
- keeping the customer informed of progress and cost
- implementing management decisions and allocations
- **in a product line, insuring that the needs of product developers and other product stakeholders are met**



Architecture and Quality Attributes

Architecture permits/precludes the achievement of a system's desired quality attributes; for example

If you desire...	In general, you need to pay attention to...
High performance	the frequency and volume of inter-element communication
Modifiability	elements' responsibilities and their interactions. (Interactions need to be limited.)
Security	inter-element communication. (It needs to be managed and protected.)
Reusability	inter-element dependencies. (They should be minimized.)

In a product line architecture, the architecture must satisfy the quality attributes of the entire family of products



Architectures for 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

The product line architecture is informed primarily by

- the product line's scope definition
- the product line requirements specification(s).

Architectural *variation mechanisms* are often used to make the architecture work for the entire set of products.



Architecture Variation Mechanisms

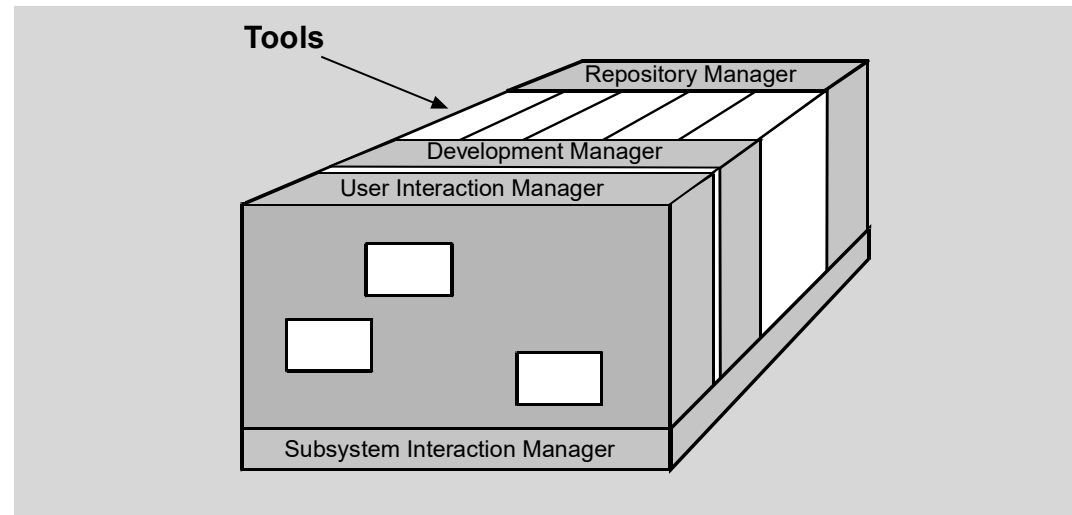
Common variation mechanisms include

- replacement, omission, and replication of architectural elements
- object-oriented (OO) techniques
 - inheritance
 - specialization
 - delegation
 - application frameworks
- parameterization (including macros and templates)
 - Special case: compile-time selection of different implementations or implementation fragments (e.g., *#ifdef*)
- generation and generators
- aspect-oriented programming
 - an approach for modularizing system properties that otherwise would be distributed across modules



Example of Variation

Reference
architectures with
slots for plug-in
components



This is an example of what kind of variation mechanism?



Understanding Architecturally-Significant Requirements

The organizational goals and the system properties required by the business are rarely understood, let alone fully articulated.

Software quality attribute requirements are seldom documented, which results in

- goals not being achieved
- inevitable conflict between different stakeholders

Architects *must* identify and actively engage stakeholders in order to

- understand the real constraints of the system
- manage the stakeholders' expectations
- negotiate the system's priorities
- make tradeoffs



Product Line Analysis - 1

Product line analysis (PLA) is

- early requirements engineering for a product line
- the link between the recognition of a business opportunity and the design of a product line architecture

PLA is stakeholder focused.

The benefits of PLA include

- systematic identification of opportunities for large-grained reuse across a product line
- clarified and refined assumptions about the product line scope
- early feedback on the technical feasibility of the product line



Product Line Analysis - 2

PLA creates a requirements model comprising four interrelated work products:

1. The *use-case model* specifies the product line stakeholders and their key interactions with the product line.
2. The *feature model* specifies the stakeholders' views of the product line. It captures the functional features of products and the software quality attributes of the product line and its products.
3. The *object model* specifies the product line responsibilities that support those features, and the commonality and variations of those responsibilities.
4. The *dictionary* defines the terminology used in the work products and supports a consistent view of the product line requirements.



Creating the Software Architecture

There are architecture definition methods and guidelines, many of which focus exclusively on the functional requirements.

It is more desirable, however, to create an architecture based on the quality attribute drivers.

One way to approach quality attribute-driven architecture creation is to use **architectural tactics and patterns** and a method (such as the SEI's Attribute-Driven Design) that capitalizes on both.



Architectural Patterns

An architectural pattern¹

- is found repeatedly in practice
- is a package of design decisions
- has known properties that permit reuse
- describes a class of architectures
- is described by a set of element types, a set of interaction mechanisms or connectors, a topological layout of these elements, and a set of constraints on topology, element behavior, and interaction mechanisms

¹ Buschmann, F. *Pattern-Oriented Software Architecture: A System of Patterns*. New York, NY: Wiley, 1996.



Example Architectural Patterns

Independent Component Patterns

- communicating sequential processes
- communicating parallel processes
- event systems
 - implicit invocation
 - explicit invocation

Data Flow Patterns

- batch sequential
- pipe-and-filter

Data-Centered Patterns

- blackboard
- repository

Virtual Machine Patterns

- interpreters
- rule-based systems

Call-Return Patterns

- main program and subroutine
- object oriented
- layers



Patterns and Tactics

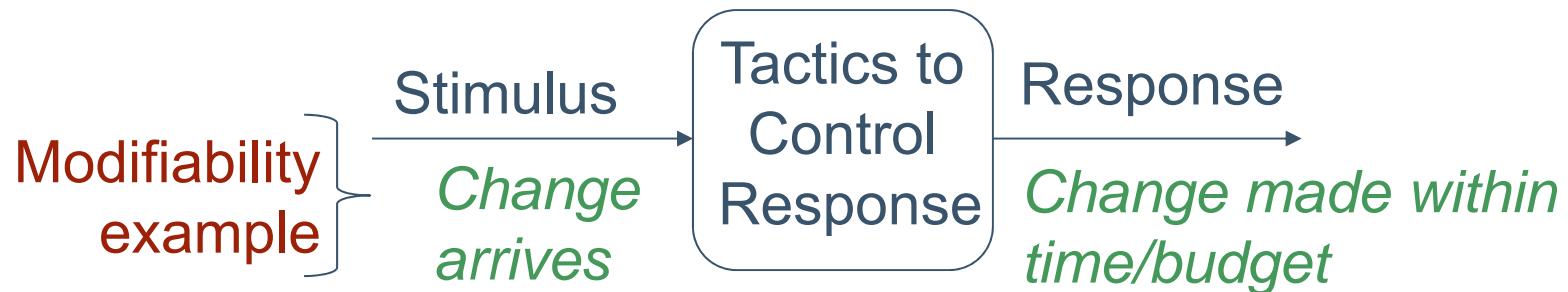
Any pattern implements several different *tactics*, often to promote various quality attributes.

- Tactics are the “building blocks” of design from which architectural patterns are created.
- A *tactic* is a design decision that is influential in the control of a quality attribute response.

Any implementation of a given pattern will also embody choices regarding tactics that will differ from other implementations of that pattern.



Tactics



Each tactic is a design option for the architect. For example

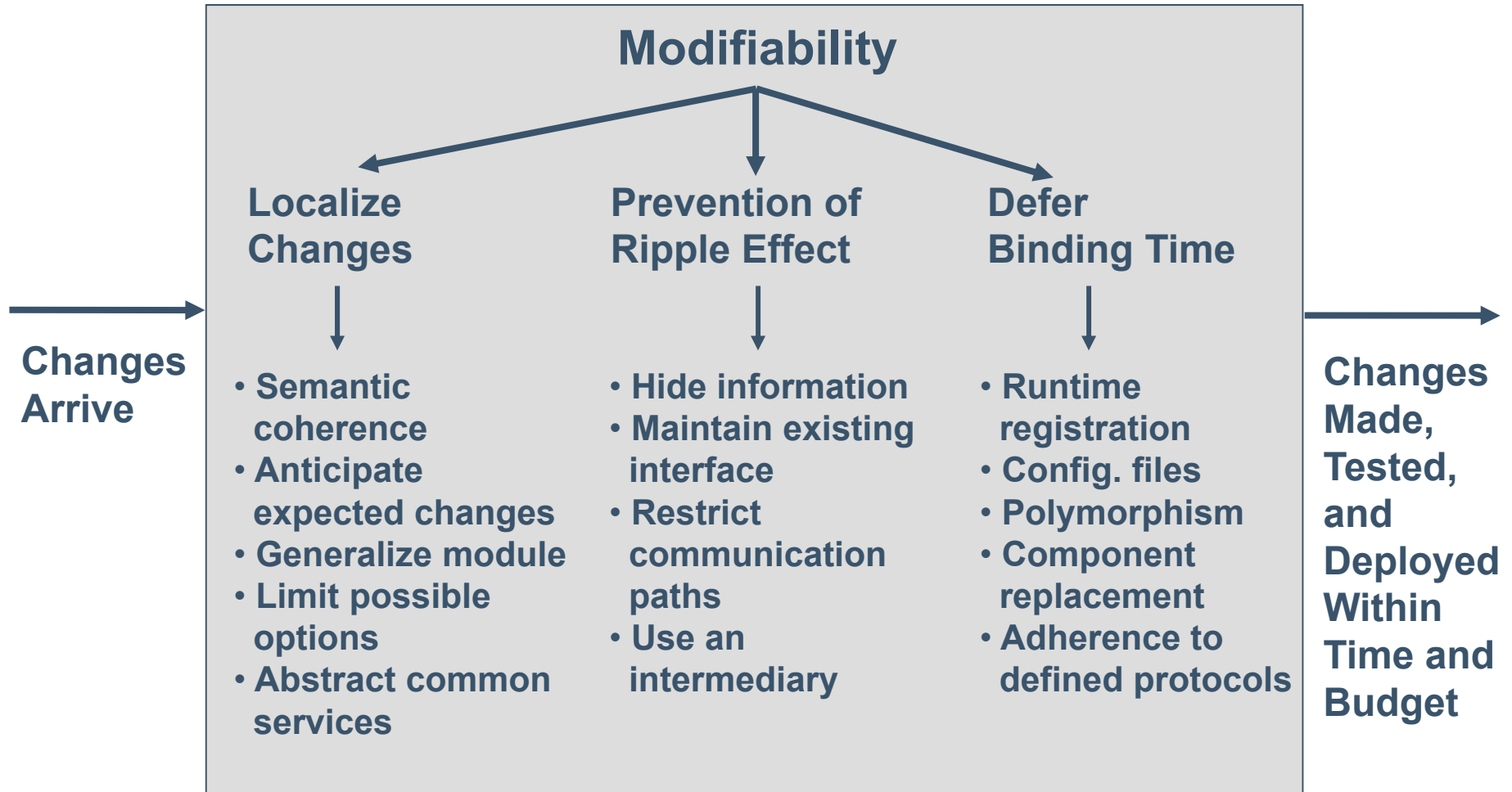
- to promote availability, we might employ redundancy
- to promote security, we might employ authentication
- to promote testability, we might employ monitors
- to promote performance, we might employ concurrency.

There are catalogs of tactics for quality attributes.

Product line architectures are frequently concerned with modifiability – changing the architecture to suit individual products.



Example: Tactics for Modifiability¹



¹ Bass, Clements, & Kazman. *Software Architecture in Practice, 2nd Edition*. Addison-Wesley, 2003.
Carnegie Mellon University, Software Engineering Institute, © 2020 Carnegie Mellon University. This document is the property of Carnegie Mellon University and is not to be distributed, copied, reproduced, or used in any form without the prior written approval of Carnegie Mellon University.



Architecture Definition: Practice Risks

An inadequate architecture—one that's not up to the task of supporting the correct range of products—can result from

- the lack of a skilled architect
- garbage in, garbage out (bad requirements)
- poor communication
- management and culture shortfalls (e.g., inadequate support for architecture creation or developers being allowed to develop without conforming to the architecture)
- no evaluation being performed
- poor tools
- overparameterization
- freezing the architecture too early or too late



Lecture Outline: Architecture-Related Practice Areas

Architecture Definition

➔ **Architecture Evaluation**

Other architecture-related needs identified in Each Asset

- configuration management (CM) plan for architecture
- tool support for architecture
- work plan for architecture
- data and metrics to track success of architecture
- attached process for architecture



Architecture Evaluation for Product Line Architectures

Architectures for product lines need to be evaluated for their ability to support the full range of product architectures, in addition to the “normal” suite of functional and quality properties.

The evaluation will tend to focus on variation points.

If a product architecture differs markedly from the overall family architecture, it may need to be evaluated separately.



Architecture Evaluation: Specific Practices

The ATAM is a method that helps stakeholders ask the right questions to discover potentially problematic architectural decisions.

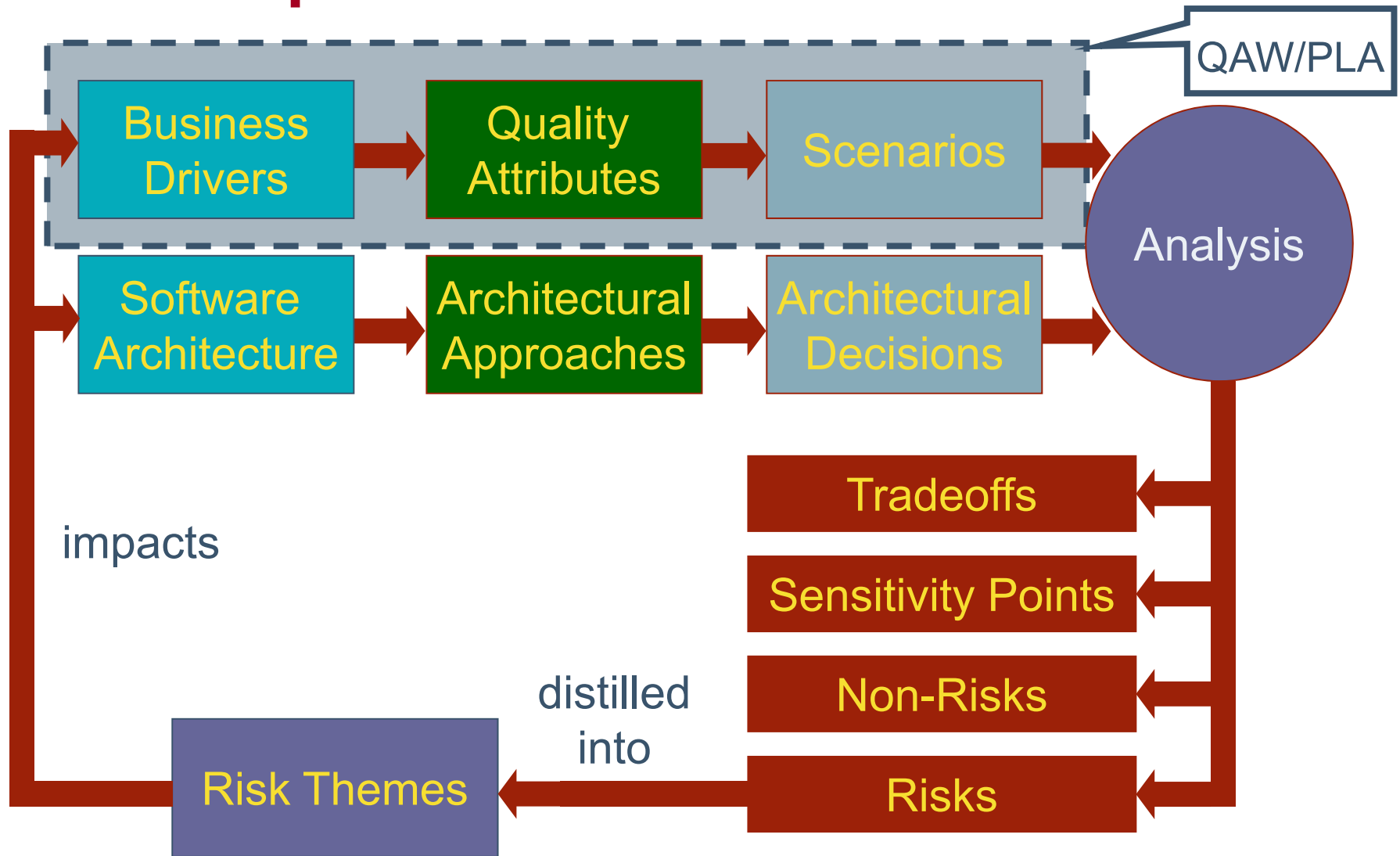
The purpose of the ATAM is *to assess the consequences of architectural decisions in light of quality attribute requirements and business goals.*

Discovered risks can then be made the focus of mitigation activities; for example, further design, further analysis, and prototyping.

Surfaced tradeoffs can be explicitly identified and documented.



Conceptual Flow of the ATAM





Schedule: Day 2

8:30 – 9:15	Developing the Core Asset Base: The Product Parts Pattern
9:15 – 10:00	Architectures for Product Lines
→ 10:00 – 10:15	BREAK
→ 10:15 – 12:00	Exercises: Architecture and the Core Asset Base
12:00 – 13:00	LUNCH
13:00 – 13:30	Production Plans
13:30 – 15:00	Group Exercise: Writing a Production Plan
15:00 – 15:15	BREAK
15:15 – 15:45	Other Patterns
15:45 – 16:15	Group Discussion : Measurement
16:15 – 16:30	Conclusions, Q&A



Session Outcomes

After this session, you should be able to

- have an appreciation for the architecture planning process
- understand variation mechanisms applied to the AGM case
- determine the actions product developers will have to take to use the software architecture to produce products
- list the contents of the AGM core asset base



Exercise 1: Architecture Planning – A Role-Playing Exercise

Divide up between architects and product line managers. The architects and managers are meeting to plan the architecture work.

- Architects: What resources and support do you need from the managers?
- Managers: What outputs and products do you expect from the architects?

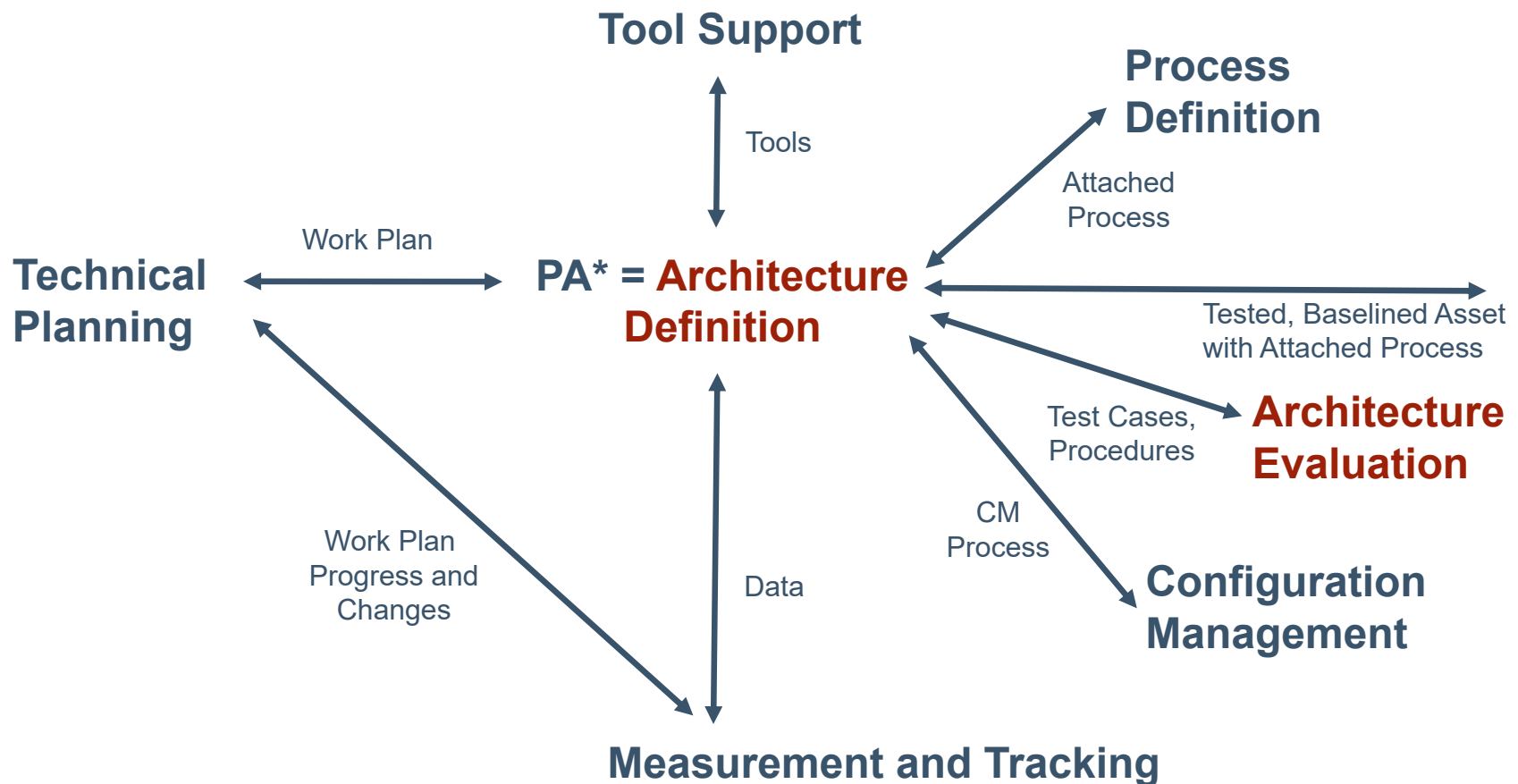
Remember the architecture-related artifacts identified by the Each Asset pattern for architecture.

AGM
Example



Reminder: Architecture and “Each Asset”

What must we produce?





Exercise 2: Architecture Definition Architecture Evaluation Other Architecture Work Products

Form teams. Within each team, complete the following assignments.

- ARCHITECTURE DEFINITION
 - Look at the scope definition. Propose a list of variation mechanisms to use.
- ARCHITECTURE EVALUATION
 - Give some scenarios especially unique for a product line architecture. Use the stimulus/environment/response form.
- PROCESS DISCIPLINE
 - Sketch an attached process for the architecture to tell how to use it for a product
- MEASUREMENT AND TRACKING
 - What metrics do you collect about the architecture to see if it's fulfilling its purpose?

AGM
Example





► Discussion: The Core Asset Base

The Product Parts pattern is about building and maintaining the core asset base.

What core assets have we identified so far? That is, list the contents of the core asset base.

If you like, you can list them from the point of view of some of their producers/consumers:

- Product developer
- Product manager
- Functional team lead
- Product line manager



Schedule: Day 2

8:30 – 9:15 Developing the Core Asset Base:
The Product Parts Pattern

9:15 – 10:00 Architectures for Product Lines

10:00 – 10:15 BREAK

10:15 – 12:00 Exercises: Architecture and the Core Asset Base

→ 12:00 – 13:00 LUNCH

→ 13:00 – 13:30 Production Plans

13:30 – 15:00 Group Exercise: Writing a Production Plan

15:00 – 15:15 BREAK

15:15 – 15:45 Other Patterns

15:45 – 16:15 Group Discussion : Measurement

16:15 – 16:30 Conclusions, Q&A



Session Outcomes

After this session, you should be able to

- derive a production strategy from the business goals of a product line
- collect the information needed for the production plan
- identify the product qualities that affect product production



Production Planning - 1

The primary function of a software product line is to produce products.

Establishing a *product production capability* is a high priority.

This capability is a function of the strategic goals of the product line and how the architecture handles variation.

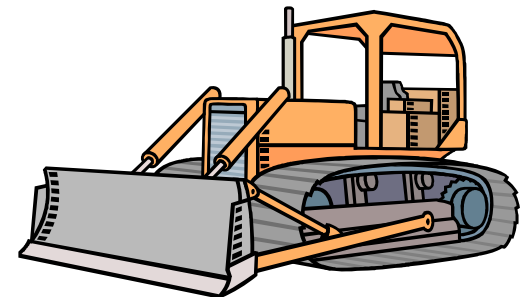
The production strategy is derived from the product line goals. The production plan is defined by the choices made by the asset builders in response to the production strategy.



What Will We Need?

The production plan describes

- the assets needed to build a product
- how those assets will be used
- the skills needed to produce the product.



Production Strategy - 1

The *production strategy* sets out how the strategic goals of the product line will be achieved through product production.

AGM has stated their production strategy as follows:

GOALS: We will position ourselves as the leading provider of rapidly customized, high-performance, low-cost games by producing products that are easily modified, have better performance than our competitors, are sufficiently low cost to deter potential competitors from entering the market, and require sufficiently few resources to allow their use on any embedded computer.

STRATEGY: We will produce the initial products using a traditional iterative, incremental development process using a standard programming language, IDE, and available libraries. We will create domain-based assets, including a product line architecture and software components, for the initial products in a manner that will support a migration to automatic generation of the second and third increment products.

**AGM
Example**





Production Strategy - 2

The production strategy defines a number of aspects of development, including

- the required expertise of the product developers
- how the product developer identifies the product to be built
- the product development process
- the technical environment used to build the software products

The production strategy should identify qualities that ensure that the production plan supports the goals of the product line.

For AGM, these qualities, described in the business case, include

- flexibility – Abstraction mechanisms keep the products flexible.
- simplicity – As many choices as possible are hidden.
- modularity – Assets are defined and implemented as modules.

AGM
Example





Buildability

Buildability is a quality of the architecture that is directly related to product production.

Factors affecting buildability include the

- complexity of the relationships among components in the architecture
- degree of automation possible in product production
- ease with which assets can be modified

Planning product production early in the life of a product line ensures that the architecture and other core assets will support buildability.



Variability

The techniques used to provide variability directly affect production planning:

- Some of the techniques require access to source code. Others handle variability as data.
- Some techniques require programming skills, while others do not.

At AGM, variability is provided via class inheritance. A new Game class is created for each new product. Variability is also provided by passing the constructor of the GameBoard class a different set of parameters. These mechanism are a result of choices made by the method engineers, the architects, and the developers.

The AGM production strategy requires programmer skills and programming tools.

**AGM
Example**





Developing the Production Strategy

Begin by identifying those product line goals that are affected by product production.

Establish priorities among these goals using the SWOT analysis in the business case and the product line analysis results.

Determine broad production technology directions that will help achieve these goals.

Develop a statement that unites these goals with the technologies to describe the intent of the product production capability; for example

- “We will produce the initial products using a traditional iterative, incremental development process...”



Production Plan - 1

The production plan is an implementation of the production strategy. It describes how the core assets are coordinated to produce products. It is intended as the main communication vehicle between the core asset developers and the product builders.

The AGM architecture uses inheritance and parameterization to provide variability.

The AGM production plan describes which objects are used as parameters to which other objects in order to produce the desired product. It also describes which core asset classes can be used as parents of product-specific classes.

AGM
Example





Production Plan - 2

The production plan expands on the Concept of Operations (CONOPS) by providing a more complete description of the process by which products are created.

The production plan specifies the following:

- the inputs needed to build a product
- the activities that result in a completed product
- the roles and responsibilities of the product developers
- the interactions needed with other groups in the organization
- the schedule and resources associated with building the product



Production Plan - 3

The Production Plan provides specifics such as:

- What tools to use and how to use them
- The location of repositories
- Techniques for creating the product-specific portions

The AGM production plan includes directions for beginning a new product:

```
Start new ClassLibrary in a Visual Studio Project
using {game name}Definitons as its name. Use
this for new classes other than the game
definition itself.
```

```
Start a new Windows Application in a Visual Studio
Project using the name of the game.
```

AGM
Example





Attached Process

The generic Production Plan is accompanied by an *attached process* that describes how to specialize the plan to a product-specific production plan.

That attached process provides guidance on how to

- select and order the process steps that are needed, based on the product definition
- develop the product's bill of materials, listing all the assets that will be used for this specific product
- create the cost estimates and time schedules for building the product
- tailor the parts of the product plan's core asset that must be changed

The result of applying this process is the product-specific production plan.



Product-Specific Production Plan

The core asset team produces the generic production plan for the products developed in the product line.

Each product team derives a product-specific production plan from the generic production plan.

Doing so may be as simple as editing a generic makefile, or it may require extensive work developing a bill of materials and schedule for a complex product.

At AGM, the product builder creates a new class for the product and populates it with the appropriate Sprites that will show the action on the screen. The makefile is automatically updated to include the new class as the root of the aggregation hierarchy.

**AGM
Example**





Product Production Process

The production plan describes the process for building a product in the product line.

The entry criteria for this process are defined in the CONOPS and typically require that the product planning and approval activities be completed prior to building a product.

The product-building process is described using the process definition style used for other processes in the organization.

The exact contents of the process depends on the level of institutionalization, but contains some subset of the Product Builder pattern and is included in Section 4 of the plan outline.



Outline of Production Plan - 1

1 Overview

- 1.1 Identification
- 1.2 Document Map
- 1.3 Using This Document
- 1.4 Concepts
- 1.5 Readership
- 1.6 Timeline

2 Strategic View of Product Development

- 2.1 Assumptions
- 2.2 Qualities
 - 2.2.1 Product Qualities
 - 2.2.2 Production Process Qualities
- 2.3 Products Possible From Available Assets
- 2.4 Production Strategy

3 Overview of Available Core Assets

- 3.1 Source Code Naming Conventions
- 3.2 Analysis-Level Assets
- 3.3 High-Level Design Assets
- 3.4 Source Code
- 3.5 Test Cases
 - 3.5.1 Unit Tests
 - 3.5.2 Integration Tests
 - 3.5.3 System Tests
- 3.6 Inputs and Dependencies
 - 3.6.1 Inputs
 - 3.6.2 Dependencies
- 3.7 Variations
 - 3.7.1 Absorbing vs. Reflecting
 - 3.7.2 Event Handling



Outline of Production Plan - 2

4 Detailed Production Process

- 4.1 Identify, Define, and Analyze the Product Incrementally
- 4.2 Design the Product
- 4.3 Build the Product
- 4.4 Test the Product

5 Tailoring Production Plan to Product-Specific Production Plan

6 Management Information


- 6.1 Schedule
- 6.2 Production Resources
- 6.3 Bill of Materials (BOM)
- 6.4 Product-Specific Details
- 6.5 Metrics

7 Attached Processes

- 7.1 Constructing the Production Plan
- 7.2 Changing the Production Plan



Schedule: Day 2

8:30 – 9:15	Developing the Core Asset Base: The Product Parts Pattern
9:15 – 10:00	Architectures for Product Lines
10:00 – 10:15	BREAK
10:15 – 12:00	Exercises: Architecture and the Core Asset Base
12:00 – 13:00	LUNCH
13:00 – 13:30	Production Plans
 13:30 – 15:00	Group Exercise: Writing a Production Plan
15:00 – 15:15	BREAK
15:15 – 15:45	Other Patterns
15:45 – 16:15	Group Discussion : Measurement
16:15 – 16:30	Conclusions, Q&A



Session Outcomes

After this session, you should be able to

- to plan how to produce products in your product line
- capture those ideas in a production plan



► Exercise: Production Planning

Goal: Explore the development of the production plan for the AGM product line.

Preparation: Assign individual team members to review the vision and strategic objectives in the AGM Product Line Introduction and the AGM architecture description focusing of how the architecture handles variability. Take about 15 minutes for this preparation. Everyone should read the partially complete AGM Product Line Production Plan. You will work with the plan during this exercise.

Activity: Follow the steps given below for developing the production plan template. As you go fill in the missing material in the partial production plan template.

1. Establish the product line context. Review the assumptions and goals (see the information in the MPP, scope document and the product line introduction), develop alternative approaches, and compare those approaches.
2. Develop a list of qualities that will be one of the drivers for developing the production process. (see section 2.2.2 in the Production Plan Template)
3. Identify, define, and explain metrics that would be useful during product production. (See section 6.5 in the Production Plan Template)
4. Be prepared to report your answers to the rest of the class.



Schedule: Day 2

8:30 – 9:15	Developing the Core Asset Base: The Product Parts Pattern
9:15 – 10:00	Architectures for Product Lines
10:00 – 10:15	BREAK
10:15 – 12:00	Exercises: Architecture and the Core Asset Base
12:00 – 13:00	LUNCH
13:00 – 13:30	Production Plans
13:30 – 15:00	Group Exercise: Writing a Production Plan
→ 15:00 – 15:15	BREAK
→ 15:15 – 15:45	Other Patterns
15:45 – 16:15	Group Discussion: Measurement
16:15 – 16:30	Conclusions, Q&A



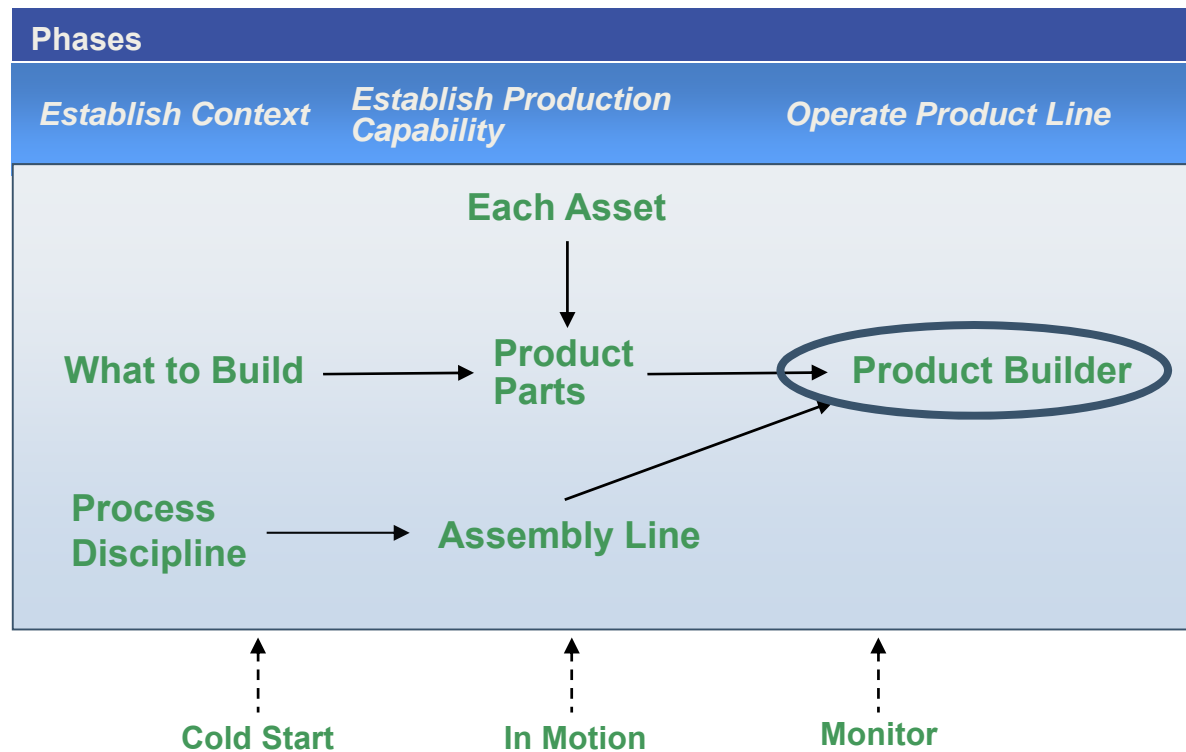
Session Outcomes

After this session, you should be able to

- enumerate additional issues that must be addressed when producing products, operating a product line on a continuing basis, and measuring the operation of the product line
- identify the practice areas in which your organization must have skills when producing products, operating a product line on a continuing basis, and measuring the operation of the product line



Reminder: Adoption Factory Pattern



→ *Informs and information flow*
- - - → *Supports*

Dynamic Structure



Product Builder Pattern - 1

Name: The Product Builder pattern consists of practice areas that should be used when any product in the product line is being developed.

Context: An organization has already established the production plan, the production capability, and the core asset base and has designated knowledgeable individuals or groups to develop a product that has been determined to be in the product line.

Problem: To develop a product from the core assets using the production plan



Product Builder Pattern - 2

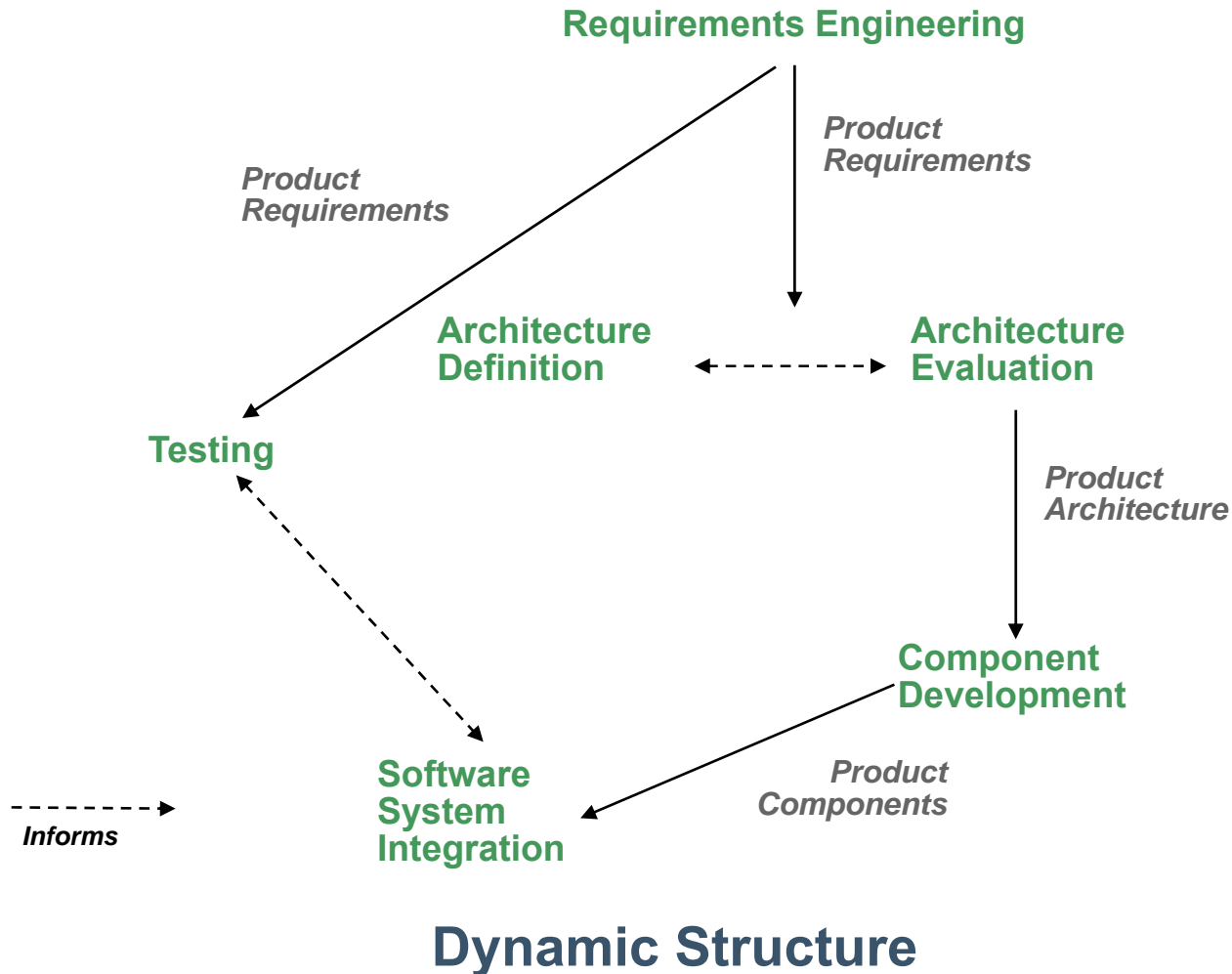
Solution: The production plan is followed using the established production capability to create an instance of the product line. Any additional components are developed and integrated into those assembled from the core asset base. The components are integrated and tested according to the production plan.

Static Structure: The practice areas that address the solution and provide the structure for the Product Builder pattern are

- Requirements Engineering
- Architecture Definition
- Architecture Evaluation
- Component Development
- Testing
- Software System Integration

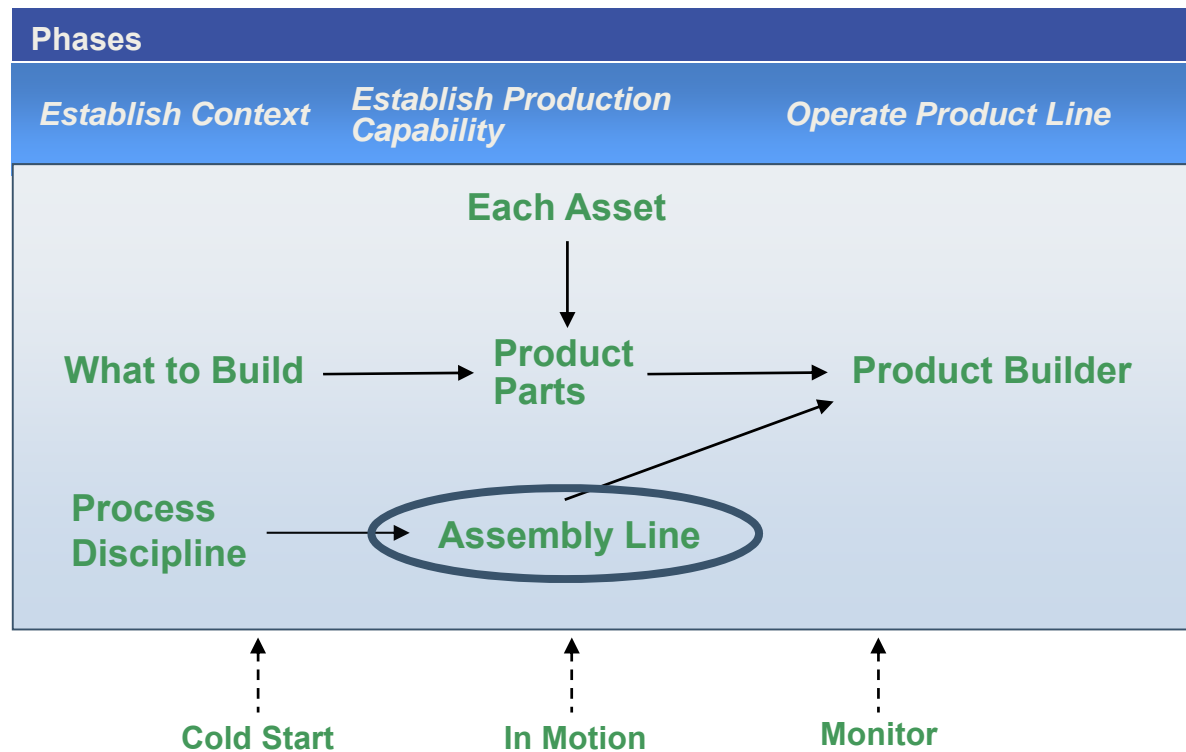


Product Builder Pattern - 3





Reminder: Adoption Factory Pattern



—————>
Informs and information flow

- - - - ->
Supports

Dynamic Structure



Assembly Line Pattern - 1

Name: The **Assembly Line** pattern should be used to set up and run the production capability of a software product line.

Context: An organization has made a decision to launch a product line effort.

Problem: To provide and use the tools and processes necessary to support the development of products from the product line's core assets



Assembly Line Pattern - 2

Solution: Think of the tools and processes that support the development of products from core assets as the assembly line for the software product line; these tools and processes provide the production capability to build products. The assembly line dictates how to assemble the products from their core asset parts. It also specifies which asset versions to use and where to find them, the schedule for the assembly, how to use automated tools to speed up the process, and how to coordinate all the activities involved in the assembly operation.



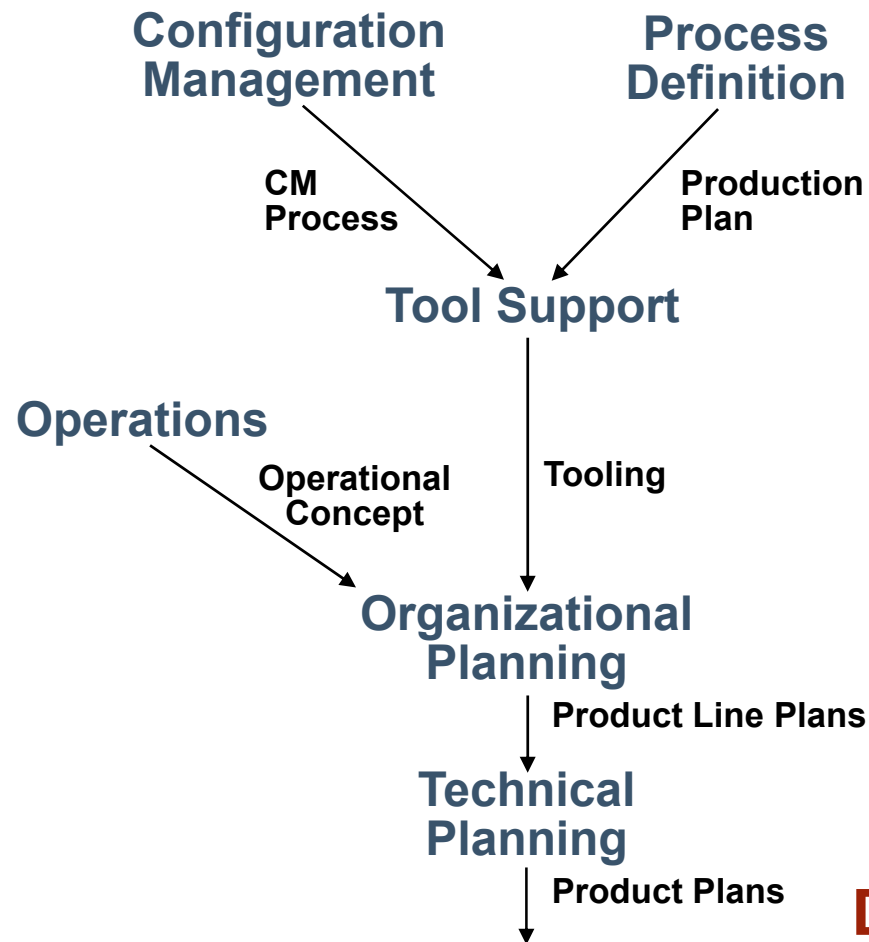
Assembly Line Pattern - 3

Static structure: The practice areas that address the solution and that provide the structure for the **Assembly Line** pattern are

- Configuration Management
- Process Discipline
- Tool Support
- Operations
- Technical Planning
- Organizational Planning



Assembly Line Pattern - 4



Dynamic Structure



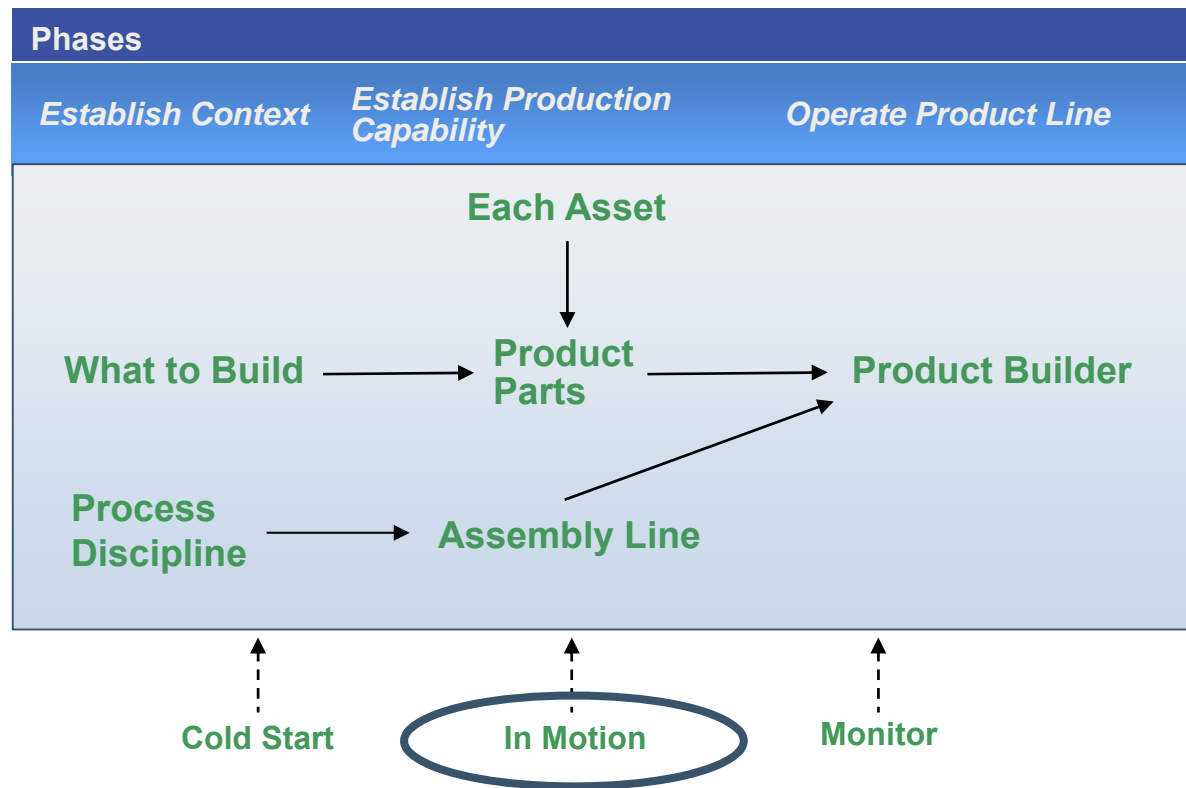
▶ Applying Assembly Line at AGM

The first increment of products was built from the first version of the architecture using C# and the VisualStudio tool suite.

The second increment was built from the first version of the architecture but using Java and the Eclipse development environment.



Reminder: Adoption Factory Pattern



→ *Informs and information flow*
- - - → *Supports*

Dynamic Structure



In Motion - 1

Name: The **In Motion** pattern consists of practice areas that keep a product line effort going after it has been launched. The In Motion pattern is really a variant of the Cold Start pattern but is described separately, because it plays a major role in product line efforts.

Context: A product line effort has been launched

Problem: To keep the product line effort going



In Motion - 2

Solution: The organization must continue to stoke the product line fire. The people in charge must ensure that

- funds to operate the effort continue to be sufficient
- operations are running smoothly
- personnel are adequately prepared for their tasks
- both customers and suppliers are in close communication and in tune with the product line effort.

Changes to the organizational structure are made as necessary.



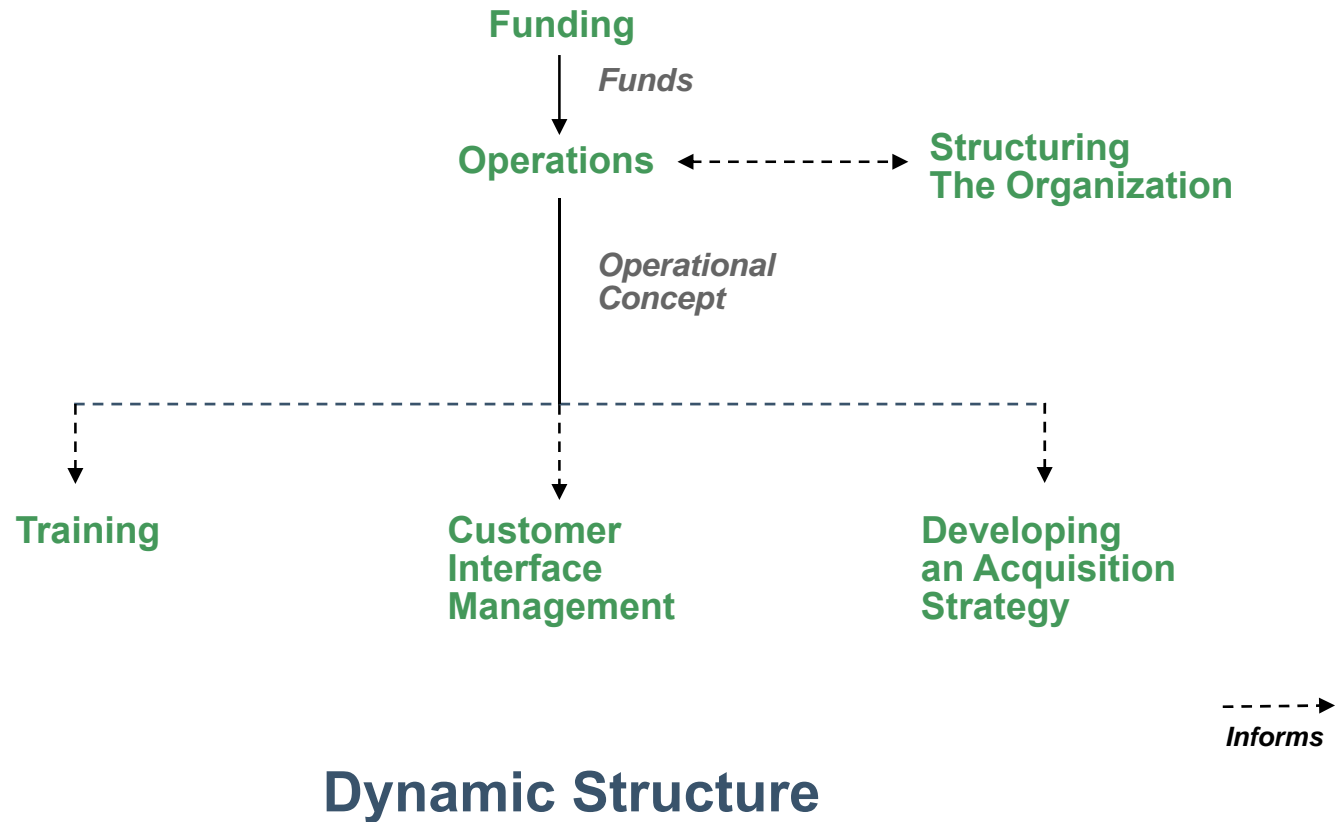
In Motion - 3

Static: The practice areas that address the solution and that provide the structure for the **In Motion** pattern are

- Customer Interface Management
- Developing an Acquisition Strategy
- Funding
- Operations
- Structuring the Organization
- Training



In Motion Pattern





In Motion - 5

Example: A robot manufacturer launched a product line for the software in its line of warehouse robots.

- Funds were secured, groups were set up and tasked, a product line manager was appointed, and a product line adoption plan was developed and is being implemented.
- The product line manager developed an operational concept, a training program, a risk management program, and an acquisition strategy.
- She has also prepared the customer for the new approach being taken to develop the software for warehouse robots.

The groups that were set up to do the product line work are functioning. The manager now wants to know what practices she needs to keep the effort in motion.



▶ Applying the In Motion Pattern at AGM

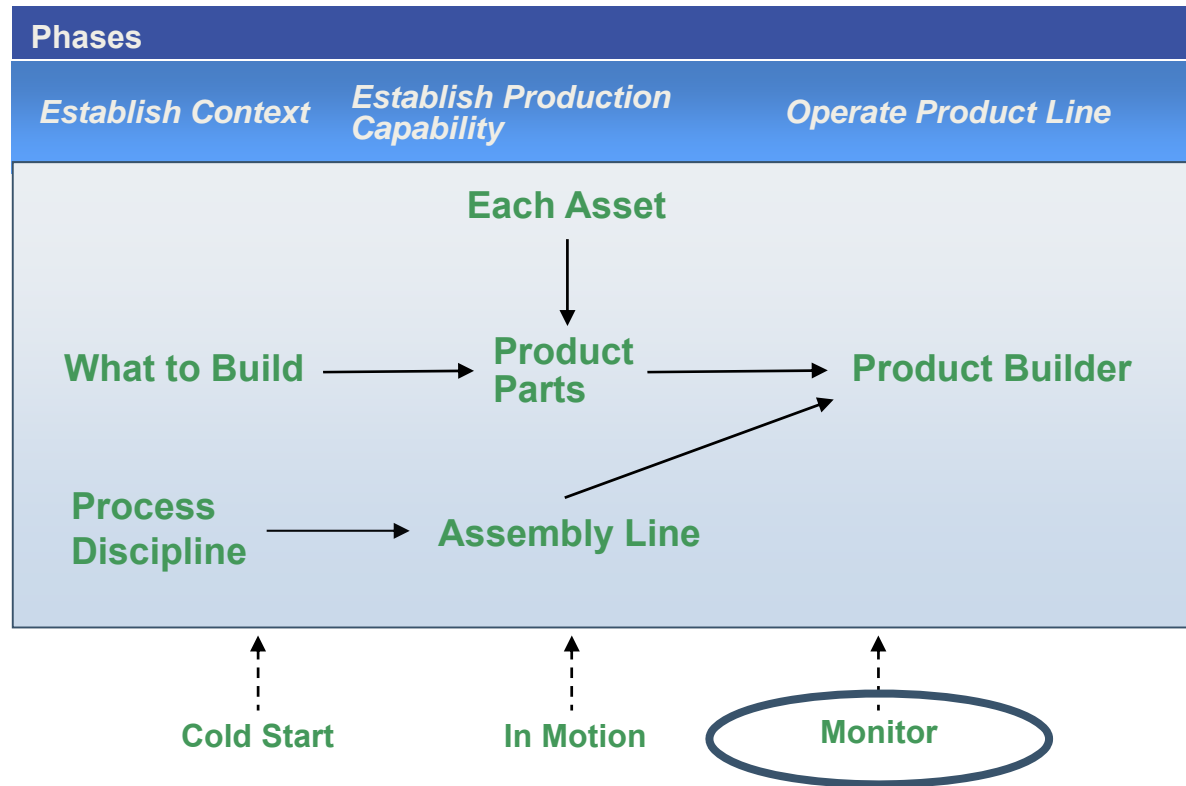
AGM funded the product line as a pilot project. The first increment of products was largely a training exercise but resulted in a set of freeware products.

The second increment produced a set of products for sale and will repay the investment. This set of products required the marketing team to train to sell a set of related products.

AGM is modifying its sales organization to reflect the relationships among products.



Reminder: Adoption Factory Pattern



→ *Informs and information flow*

- - - → *Supports*

Dynamic Structure



Monitor - 1

Name: The **Monitor** pattern consists of practice areas that all serve to monitor an ongoing product line effort and apply course corrections to keep activities on track.

Context: An organization has a software product line effort in play

Problem: To monitor the product line operation and apply course corrections when needed



Monitor - 2

Solution: Monitoring a product line operation requires routinely performed practices that keep a pulse on the organization by

- collecting and analyzing process and product data
- communicating with customers
- identifying and analyzing risks.

It also requires routinely performed practices that precipitate needed changes in product line operations.



Monitor - 3

Static structure: Two groups of practice areas address the solution and provide the structure for the **Monitor** pattern:

- The Listen Group includes these practice areas:
 - Measurement and Tracking
 - Technical Risk Management
 - Organizational Risk Management
 - Customer Interface Management
- The Response Group includes these practice areas:
 - Technical Planning
 - Organizational Planning
 - Process Discipline



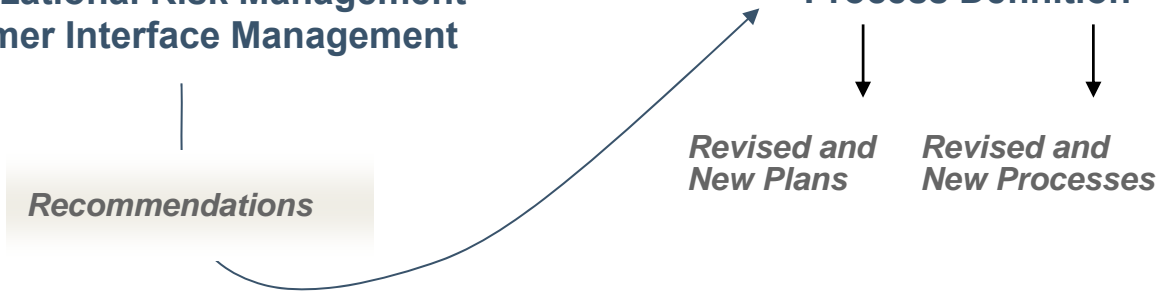
Monitor Pattern - 4

Listen Group

- Measurement and Tracking
- Technical Risk Management
- Organizational Risk Management
- Customer Interface Management

Response Group

- Technical Planning
- Organizational Planning
- Process Definition



Dynamic Structure



Monitor - 5

Examples:

- *Scenario 1:* A company has a software product line of flight simulator video games, has a core asset base, and has fielded five different games, all still being sold, from the set of assets. The company needs to keep a watchful eye on the organization to guarantee that all the product line processes are being followed, existing plans are realistic, its products are being well received in the market place, and its staff is performing according to expectations. The company also needs to be nimble in changing plans and processes to fine tune its operations or to become more responsive to its market.



Monitor - 6

Examples: (cont'd.)

- *Scenario 2:* TelecomPlus recently launched a software product line effort for its new line of mobile phones. The product line architecture has been developed, and component development is nearing completion. The first product to be built from the assets is being actively marketed, and development has just begun. The schedule to get this first product to market is tight. The company needs to ensure that its product line effort will get to market on time *and* poise TelecomPlus for its second product, which is to appear close on the heels of the first.




▶ Applying the Monitor Pattern at AGM

AGM's newly formed project management office was assigned responsibility for measurement. This office supplies trained project managers to lead core asset and product development projects. A standard set of metrics has been defined for immediate collection. Additional measures will be added as needed.

The CEO's direct reports handles risk management but is unsure how to manage risk with so many projects at one time. A consultant will be needed to train them.



Schedule: Day 2

8:30 – 9:15	Developing the Core Asset Base: The Product Parts Pattern
9:15 – 10:00	Architectures for Product Lines
10:00 – 10:15	BREAK
10:15 – 12:00	Exercises: Architecture and the Core Asset Base
12:00 – 13:00	LUNCH
13:00 – 13:30	Production Plans
13:30 – 15:00	Group Exercise: Writing a Production Plan
15:00 – 15:15	BREAK
15:15 – 15:45	Other Patterns
 15:45 – 16:15	Group Discussion : Measurement
16:15 – 16:30	Conclusions, Q&A




► Exercise: Measurement

Form into teams. Prepare answers to the following:

1. AGM adopted the product line approach to achieve what gains?
2. For each gain listed above, give examples of measurements you would want to be able to test whether those gains are being achieved.
3. What will you measure to see if your core asset base is performing as well as it could be?
4. What would you measure to see if product development was occurring as smoothly as it could be?
5. What would you measure to gauge the overall health of the product line organization?



Schedule: Day 2

8:30 – 9:15	Developing the Core Asset Base: The Product Parts Pattern
9:15 – 10:00	Architectures for Product Lines
10:00 – 10:15	BREAK
10:15 – 12:00	Exercises: Architecture and the Core Asset Base
12:00 – 13:00	LUNCH
13:00 – 13:30	Production Plans
13:30 – 15:00	Group Exercise: Writing a Production Plan
15:00 – 15:15	BREAK
15:15 – 15:45	Other Patterns
15:45 – 16:15	Group Discussion : Measurement
 16:15 – 16:30	Conclusions, Q&A



Conclusions

This course has tried to give you as much hands-on experience as possible in creating the artifacts and making the decisions associated with starting a software product line.

We have taken a pattern-based approach, as patterns leverage previous solutions to recurring problems.

The overarching pattern for a product line organization is **Adoption Factory**. It describes a software product line organization in full swing. Use it to picture the ultimate desired state and roadmap for how to get there.



Adoption Factory Pattern

Adoption Factory recognizes that fielding a product line involves

- deciding what to build
- building and running the production capability
- preparing the organization
- designing and providing the product parts
- running the assembly line to produce products
- monitoring the process
- defining processes that are important during adoption and operation



Using the Adoption Factory Pattern

Many paths through the Adoption Factory pattern are possible. (This is discussed more in the SEI *Adopting Software Product Lines* course.)

Which one you choose depends on your context and how much preliminary work or thinking has been done about software product lines.

Also, you don't always complete one thing before you start another.

We have explored one path through the Adoption Factory pattern, based on the hypothetical situation of Arcade Game Maker and the adoption plan that it built.



Other Patterns We've Seen

What to Build

- helps determine if the product line approach is right for your organization
- helps set the scope of a product line

Cold Start

- helps get the product line off the ground

Product Parts (with the **Each Asset** subpattern)

- helps orchestrate the building of the core assets

In Motion and Monitor

- helps assure that an organization stays on track



Takeaways

Experience in writing a business case

Experience in writing a concept of operations

Experience in writing a production plan

An appreciation for the role of architecture

Deeper insight into many product line practice areas

Most of all, a real sense of what's involved in running a software product line organization