

Vorlesung "Software-Engineering"

Prof. Ralf Möller, TUHH, Arbeitsbereich STS

- Vorige Vorlesung: Opportunistische Wiederverwendung
 - Enterprise Application Integration
 - Software-Architekturen u.a. für EAI-Anforderungen
- Heute: Organisierte / strategische Wiederverwendung
 - Software Product Lines

Danksagung

- Diese Vorlesung verwendet folgende Präsentation:
- Reuse That Pays -
Linda M. Northrop ,
ICSE 2001
- siehe:
<http://www.sei.cmu.edu/plp/presentations.html>

Reuse that Pays

A photograph of a modern building with a glass facade and a flagpole with the American flag, set against a blue sky. The building has a prominent glass section that reflects the sky and the sun. The flagpole is positioned to the left of the building. The overall scene is brightly lit, suggesting a sunny day.

Linda M. Northrop

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

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



Imagine

Reuse

not for the sake of reuse

**but reuse as a strategy to
achieve business goals**



Cummins Inc.: Diesel Engine Control Systems

Over 20 product groups with over 1000 separate engine applications

- **product cycle time was slashed from 250 person-months to a few person-months**
- **Build and integration time was reduced from one year to one week**
- **quality goals are exceeded**
- **customer satisfaction is high**
- **product schedules are met**

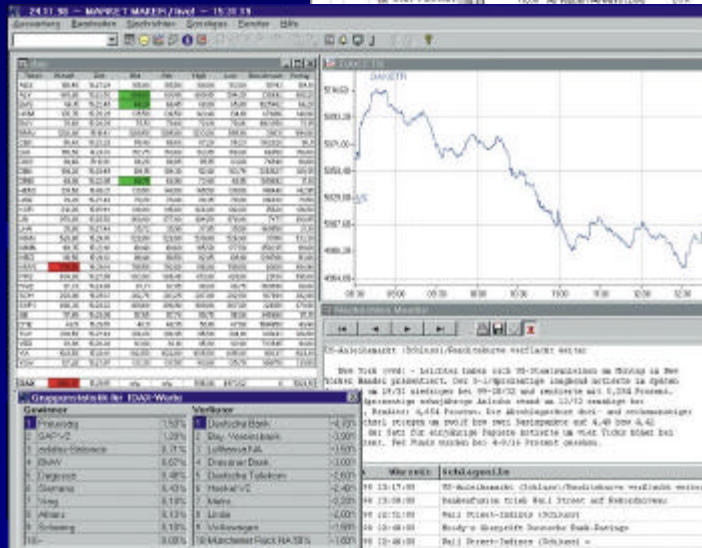




Market Maker GmbH: MERGER

Internet-based stock market software

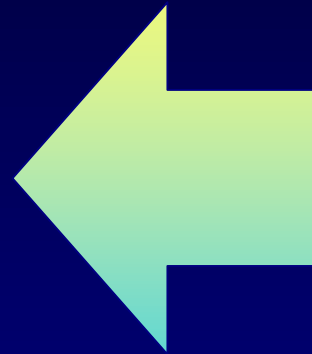
- each product “uniquely” configured
- three days to put up a customized system





How Did They Do It?

**strategic
reuse**

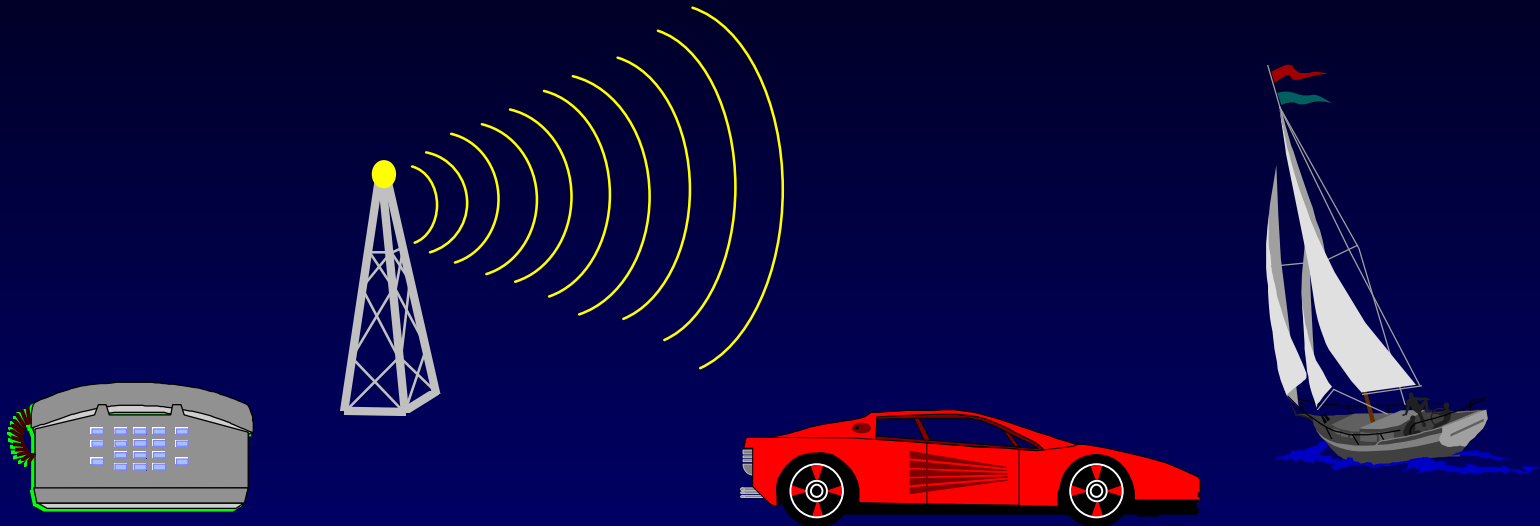


**business strategy
and
technical strategy**

employed to achieve explicit business goals



Beginning of the 21st Century



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



Universal Business Goals

High quality

Quick time to market

Effective use of limited resources

Product alignment

Low cost production

Low cost maintenance

Mass customization

Mind share

**improved
efficiency
and
productivity**

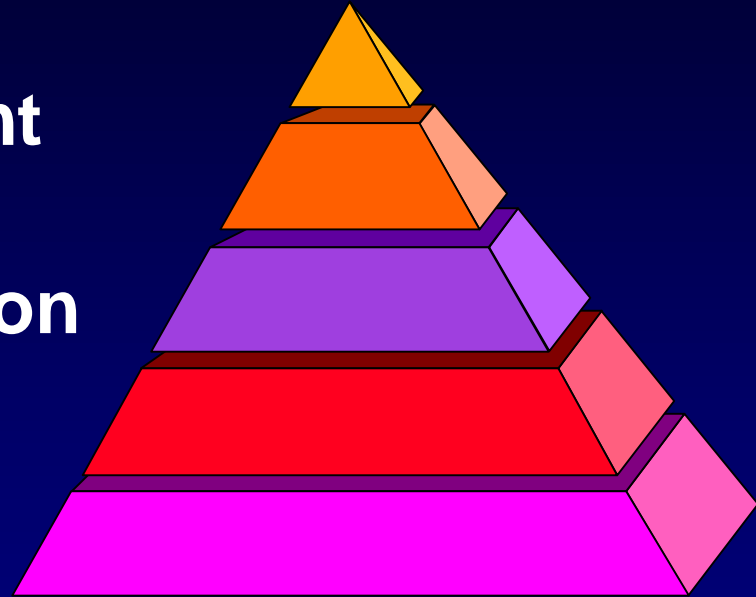


Software (System) Strategies

Process Improvement

Technology Innovation

Reuse





Ah, Reuse

First introduced at the 1968 NATO conference on software engineering

My thesis is that the software industry is weakly founded, in part because of the absence of a software components sub-industry. [McIlroy, 1969]



Reuse: a Recurring Theme-1

*Most industry observers agree that improved software development productivity and product quality will bring an end to the software crisis. In such a world, **reusable software** would abound.*

[Pressman, 1982]

*What is needed is the ability to create **templates of program units** that can be written just once and then tailored to particular needs at translation time. As we shall see, **Ada** provides a general and very powerful tool to do just this.*

[Booch, 1986]



Reuse: a Recurring Theme-2

*If one accepts that reusability is essential to better software quality, the **object-oriented** approach provides a promising set of solutions.*
[Meyer, 1987]

***Inheritance** is the most promising concept we have to help us realize the goal of constructing software systems from reusable parts.*

[Korson and McGregor, 1990]



Reuse: a Recurring Theme-3

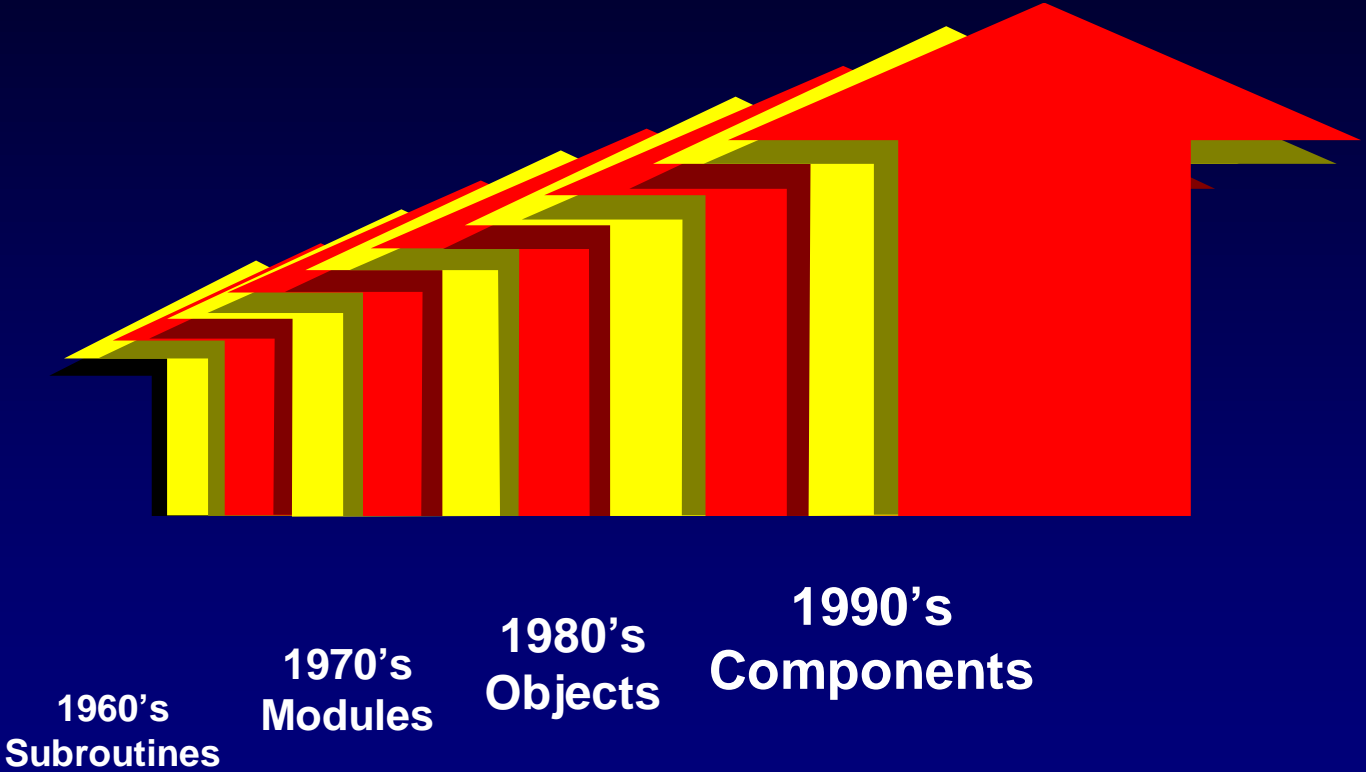
*A fundamental problem in software reuse is the lack of tools to locate potential code for reuse...**information retrieval systems** have the power and flexibility to ameliorate this problem. [Frakes and Nejme, 1987]*

*Reusable components would be schematized and placed in a **large library** that would act as a clearing house for reusable software, and royalties would be paid for use of reusable components.*

[Lubars, 1988]



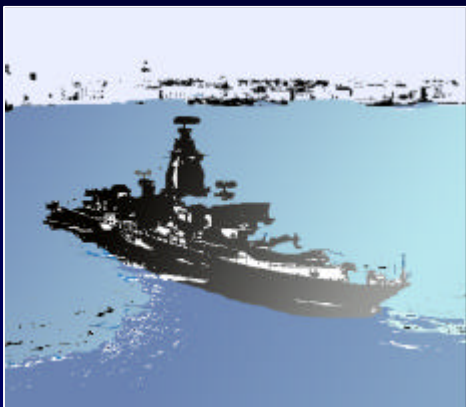
Reuse History



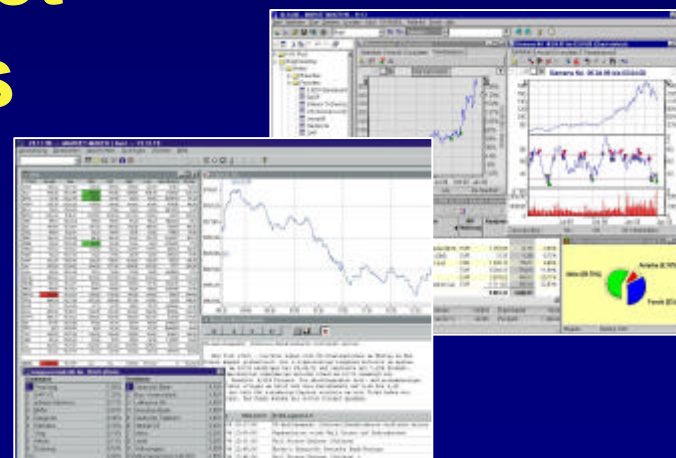
Results fell short of expectations



Strategic Reuse is Different

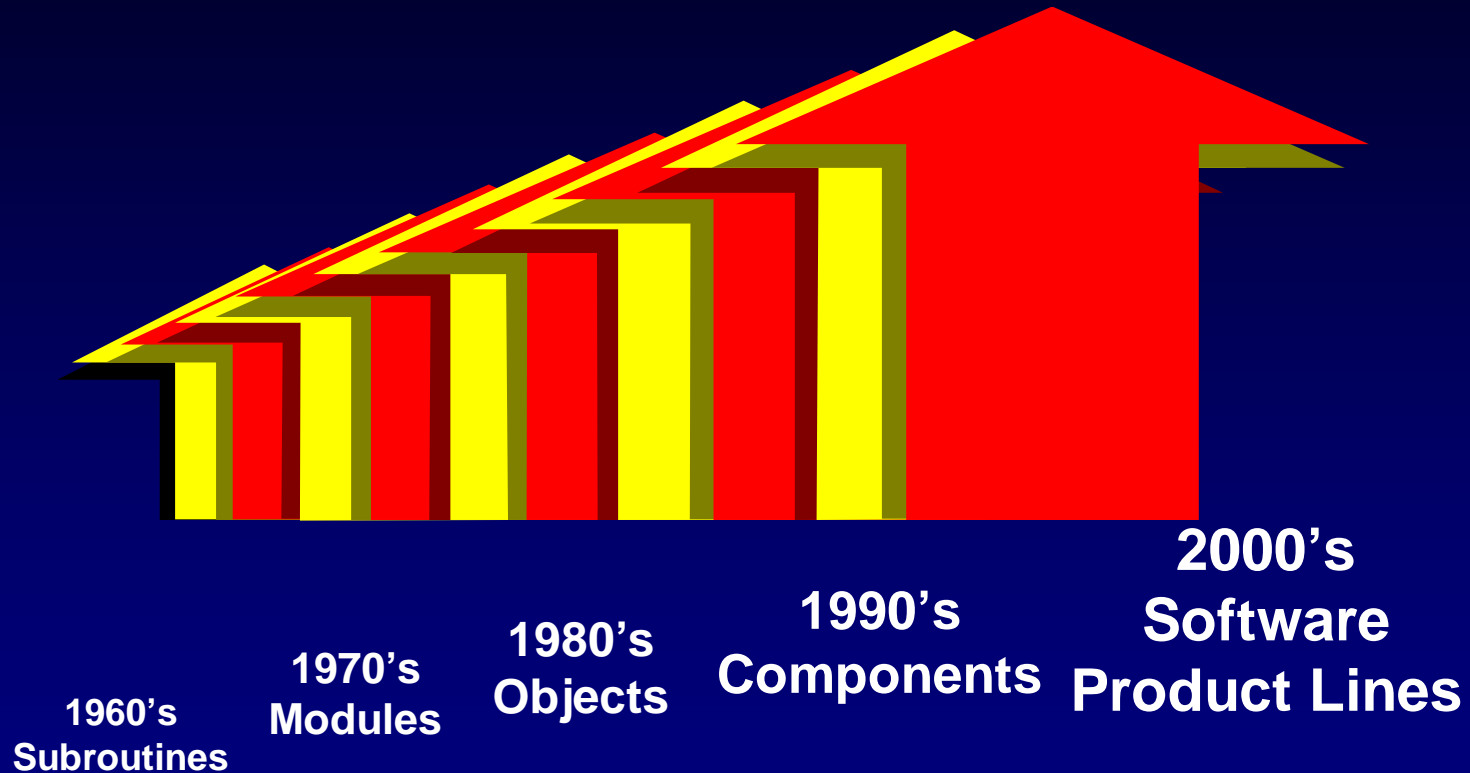


Software Product Lines





Reuse History: From Ad-Hoc to Systematic



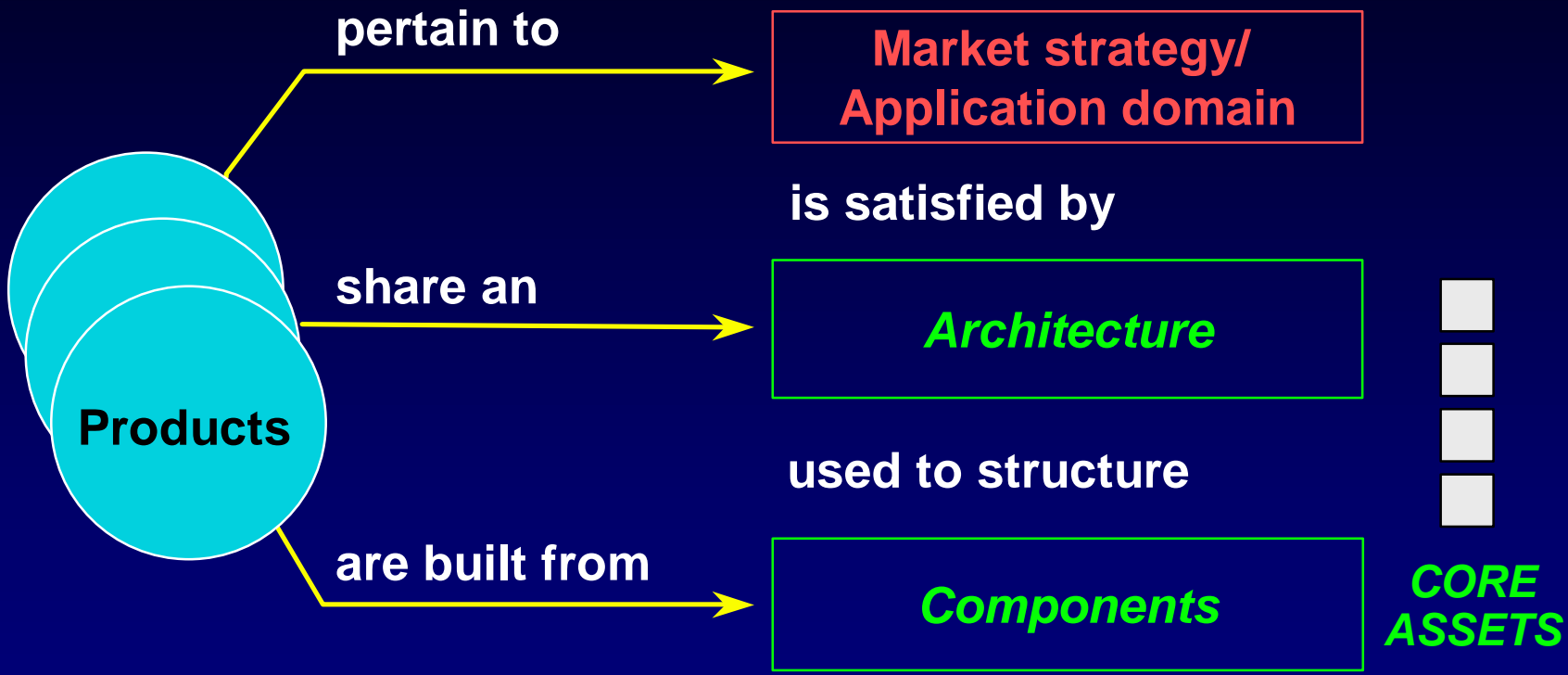


What is a Software Product Line?

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



Software Product Lines



Product lines

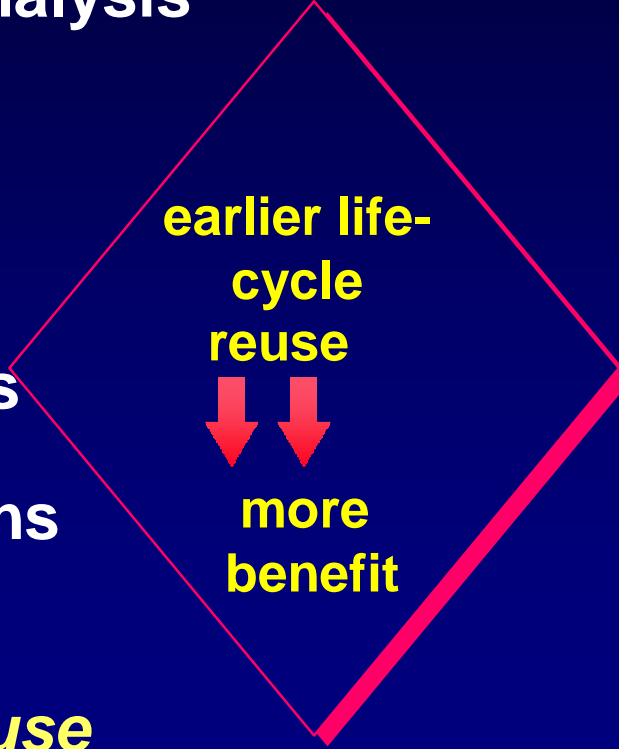
- take economic advantage of commonality
- bound variability



How Do Product Lines Help?

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

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



earlier life-
cycle
reuse



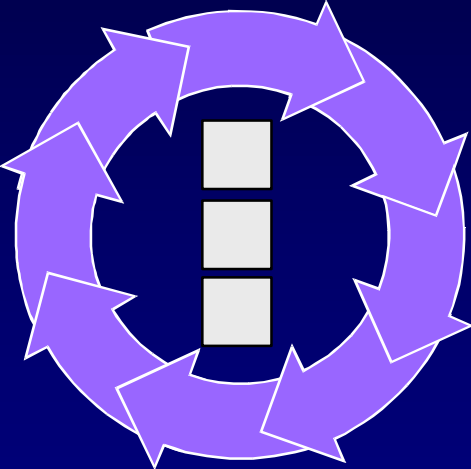
more
benefit

product lines = strategic reuse

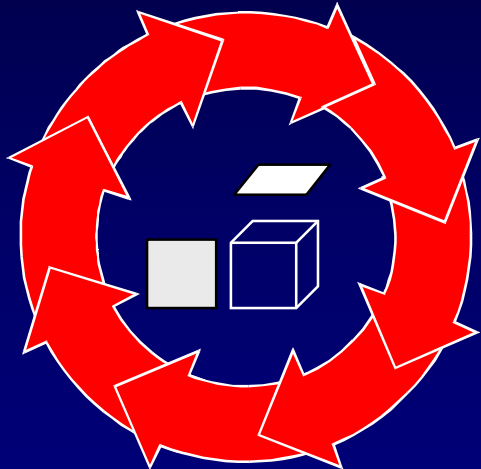


The Key Concepts

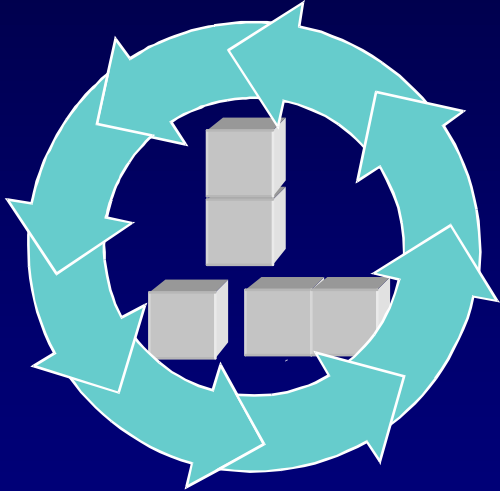
**Use of a
common
asset base**



in production



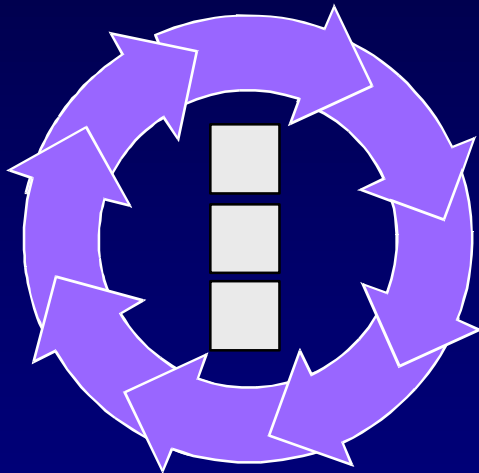
**of a related
set of products**





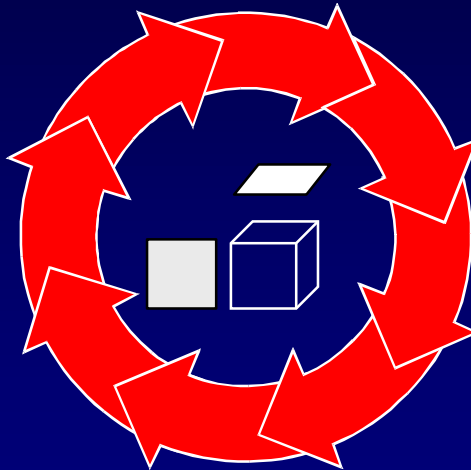
The Key Concepts

**Use of a
common
asset base**



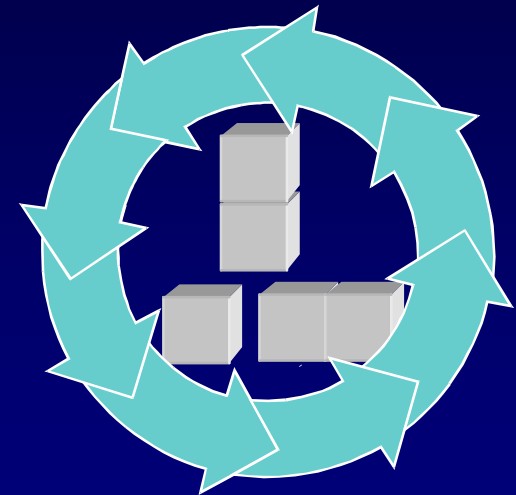
Architecture

in production



Production Plan

**of a related
set of products**



**Scope Definition
Business Case**



Software Product Lines Are Not

Just

- **libraries of objects, components, or algorithms**
- **reuse when the software engineer is so inclined**
- **reuse with no repeatable production process**
- **a configurable architecture**



Software Product Lines Are Not

Just

- libraries of objects, components, or algorithms
- reuse when the software engineer is so inclined
- reuse with no repeatable production process
- a configurable architecture

=

*Opportunistic
Reuse*



Organizational Benefits

Improved productivity

by as much as 10x

Decreased time to market (to field, to launch...)

by as much as an order of magnitude

Decreased cost

by as much as 60%

Decreased labor needs

by as much as 10X fewer software developers

Increased quality

by as much as 10X fewer defects



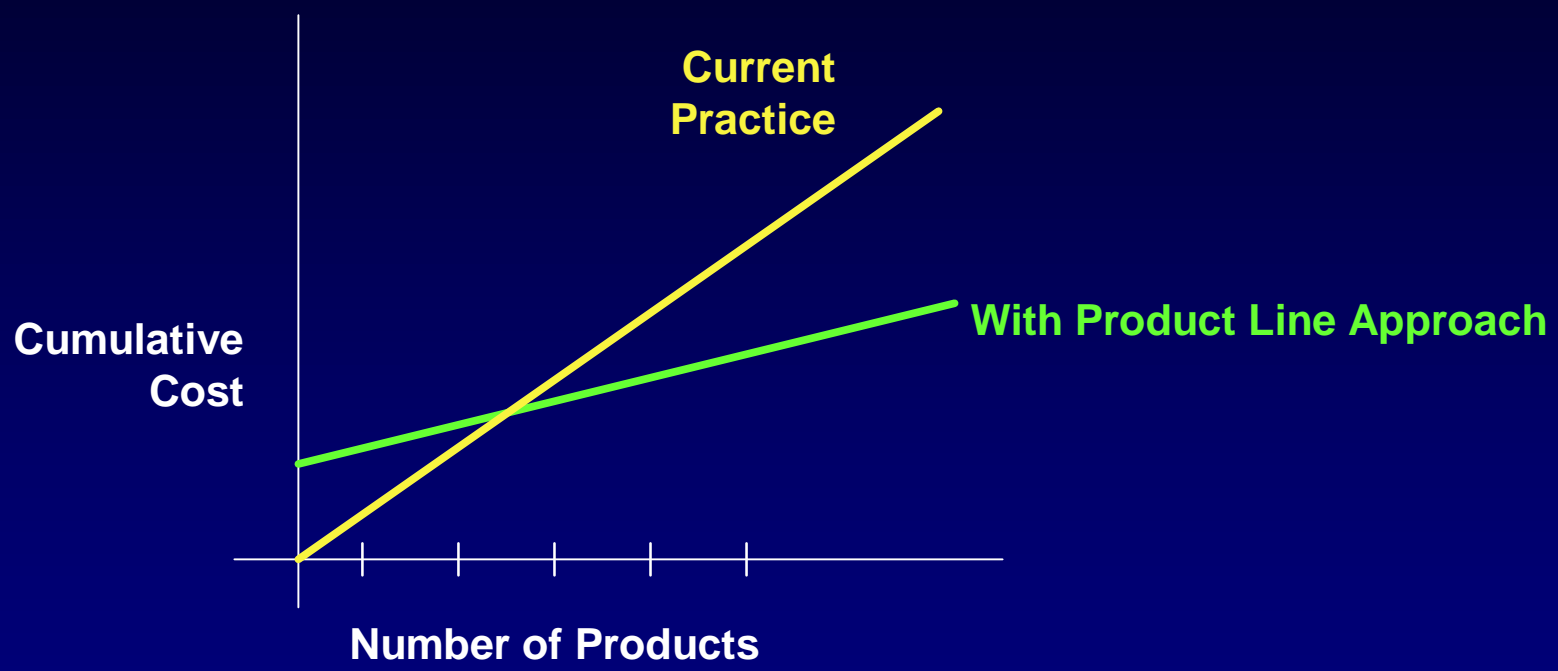
Costs of a Product Line

Asset	Costs
architecture	must support variation inherent in the product line
software components	must be designed to be general without loss of performance; must build in variation points
performance modeling and analysis	reusing the analysis may constrain processor allocation
test plans, test cases, test data	must consider variation points and multiple instances of product line
project plans	Single plans will be largely dependent upon degree of reuse
tools and processes	must be more robust
people, skills, training	Must involve training and expertise centered around the assets and procedures associated with the product line





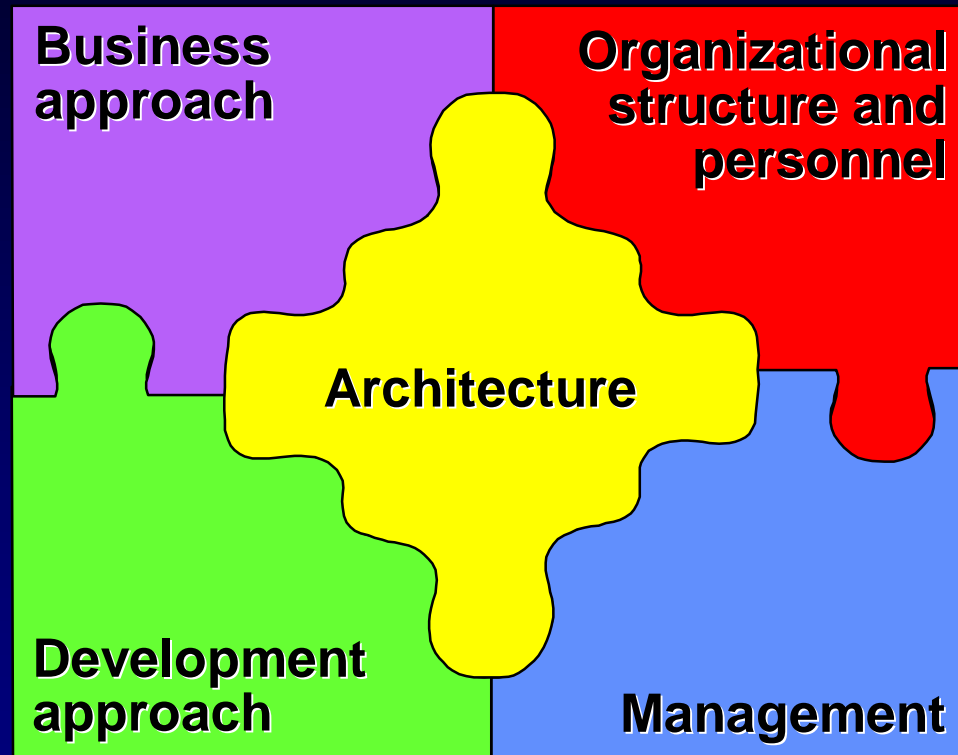
Economics of Product Lines



Derived from data supplied by
Lucent Technologies
Bell Laboratories Innovations



Necessary Changes



**The architecture is the
foundation of everything.**



Product Line Practice

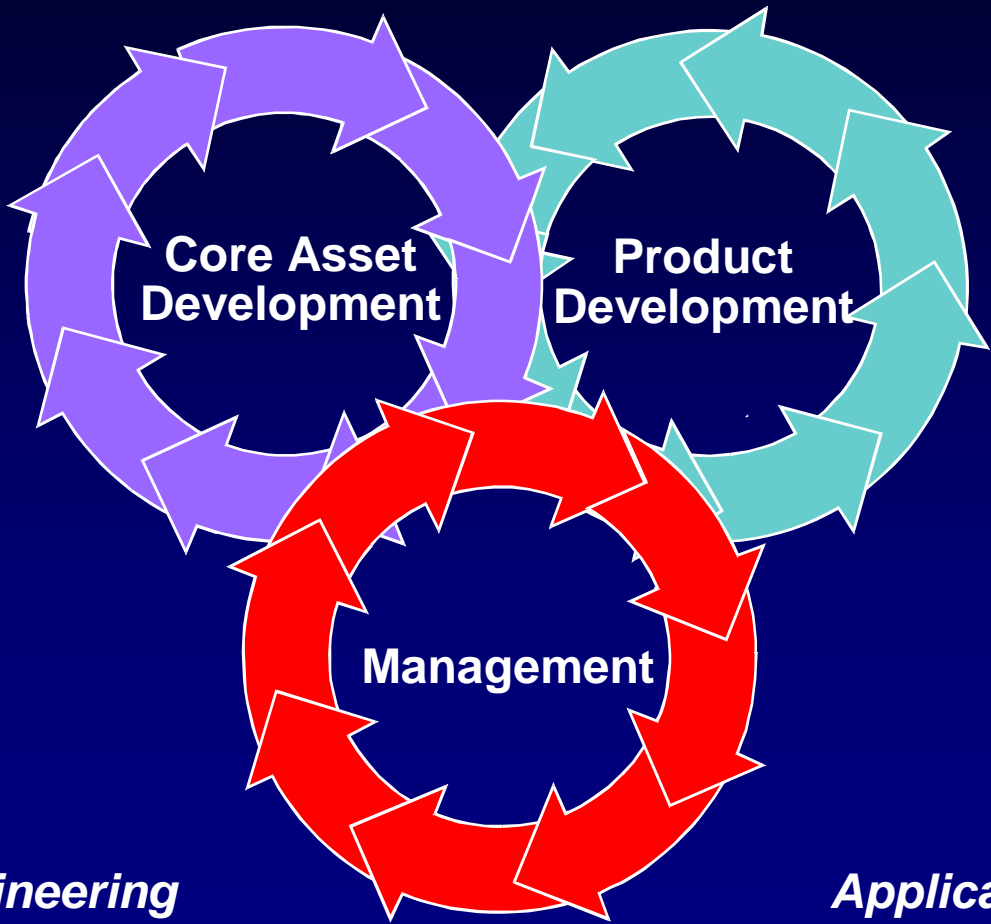
Contexts for product lines **vary** widely

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

But there are **universal essential** activities and practices.



Product Line Essential Activities

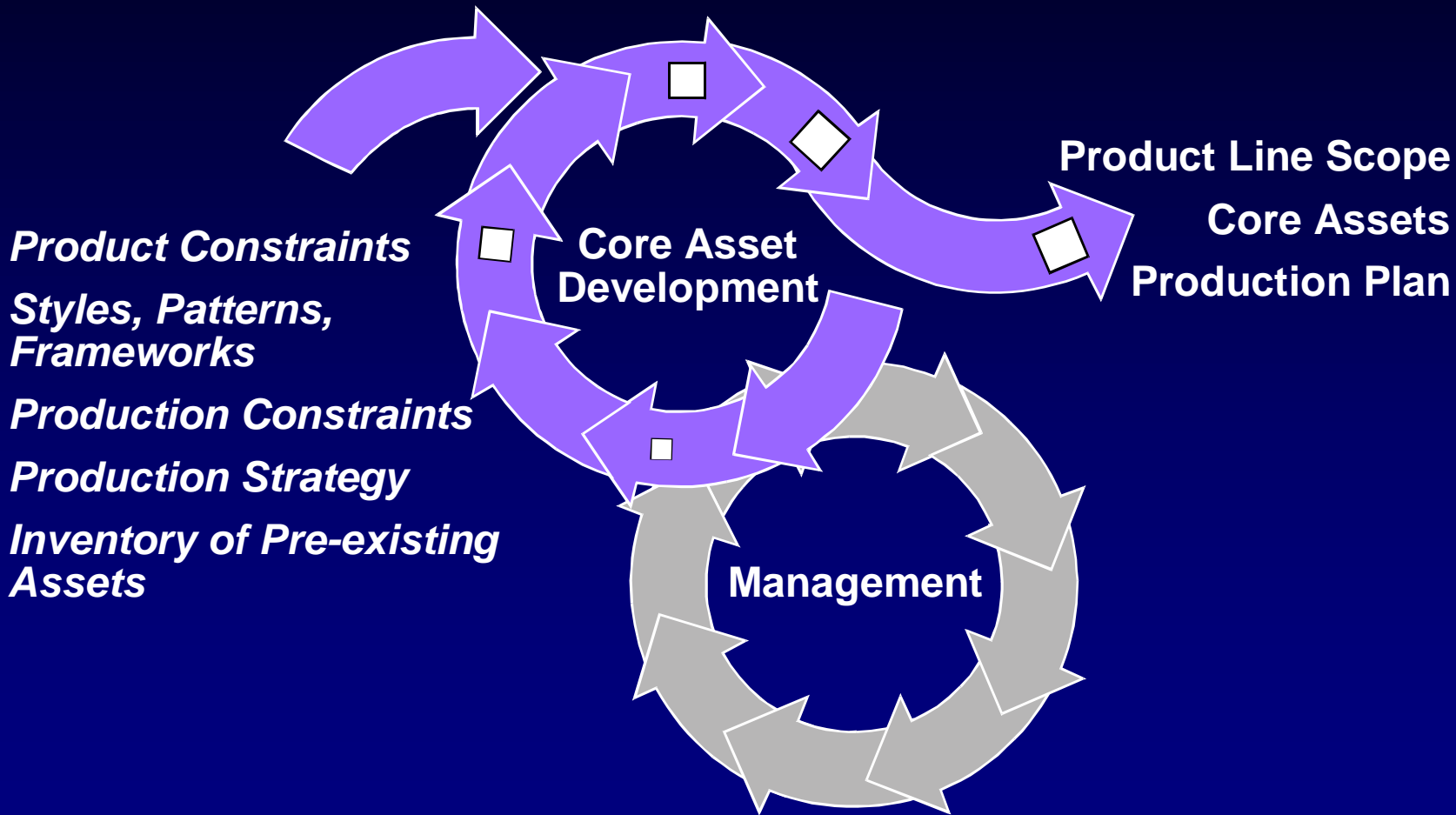


Domain Engineering

Application Engineering

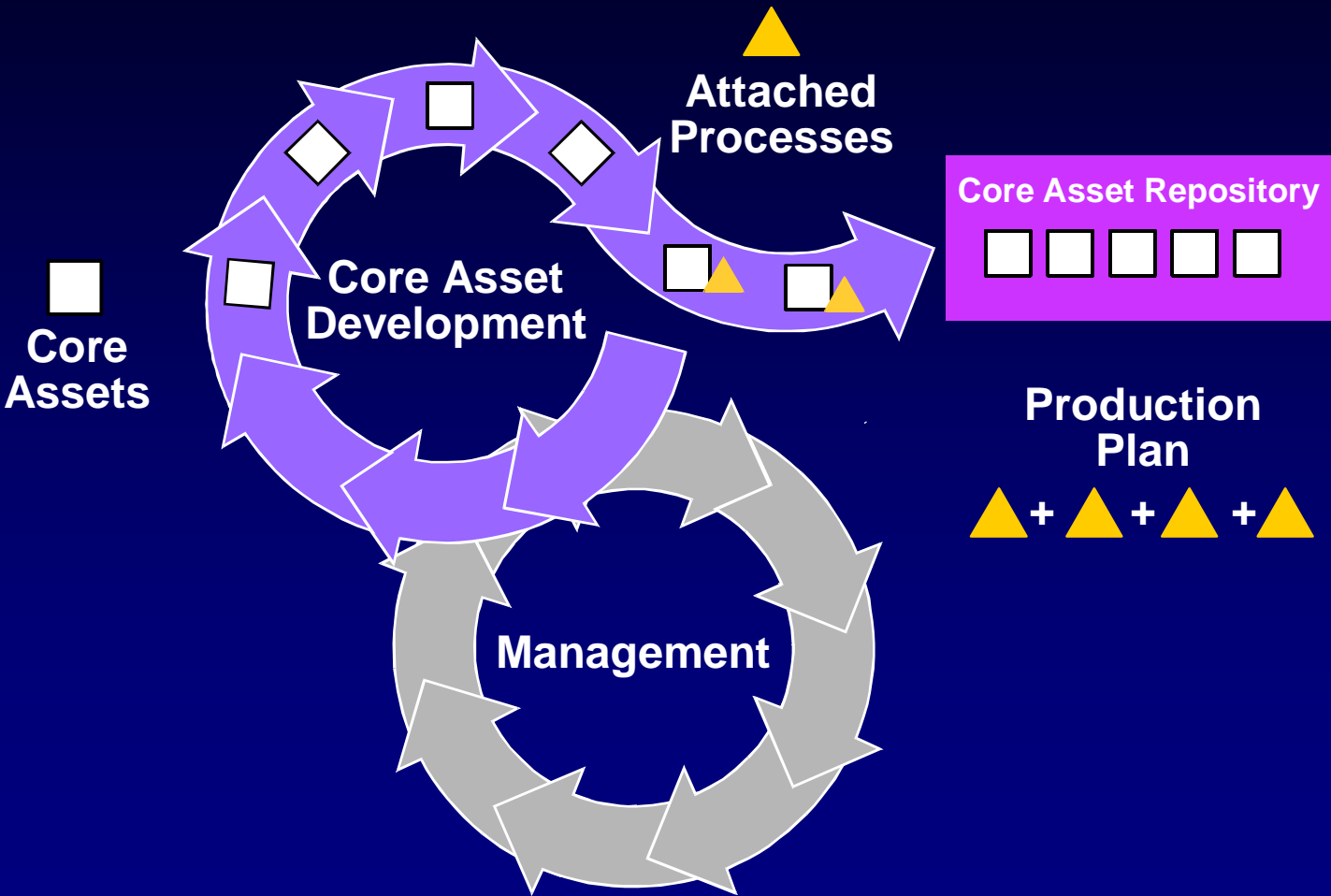


Core Asset Development



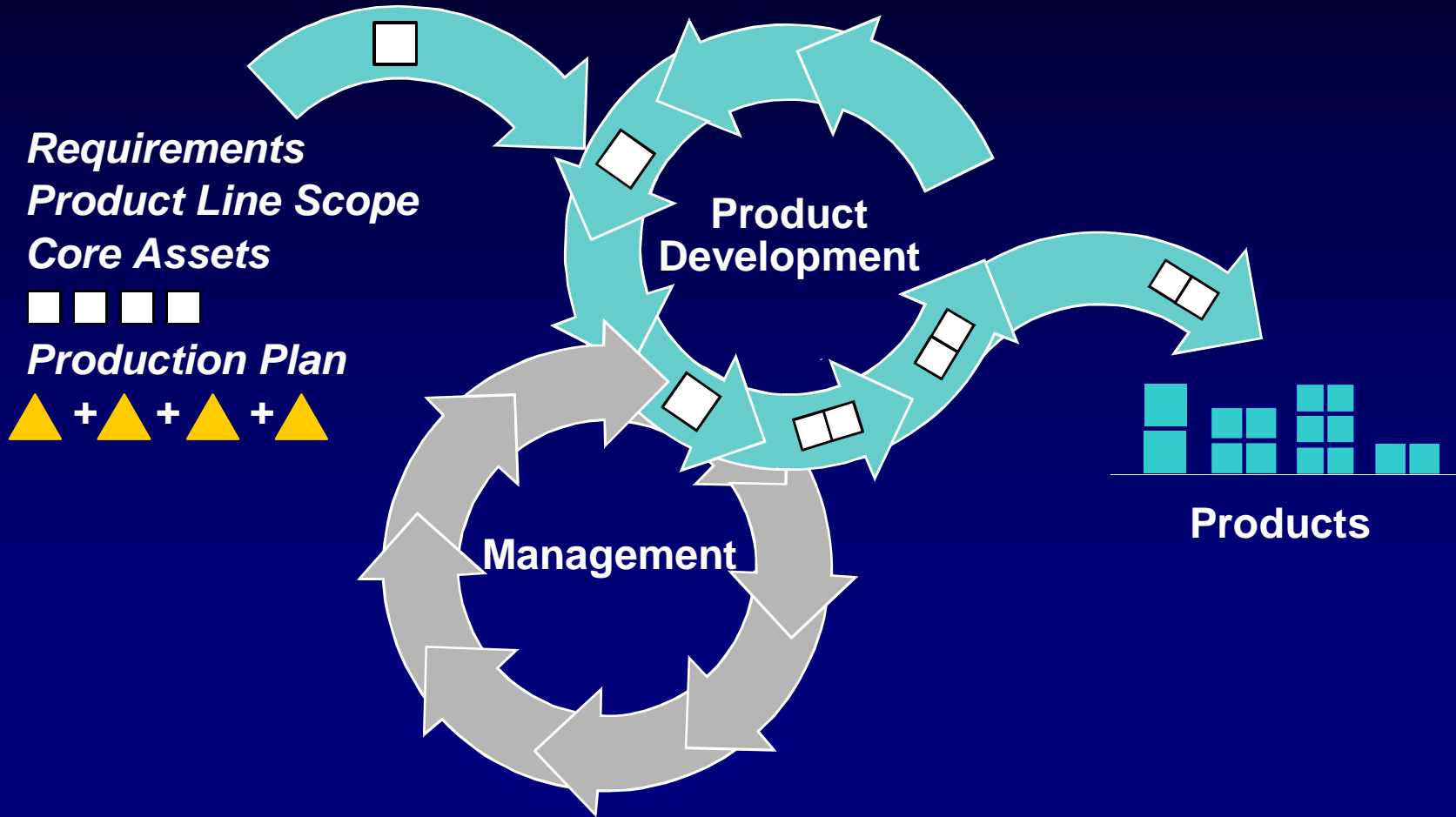


Attached Processes



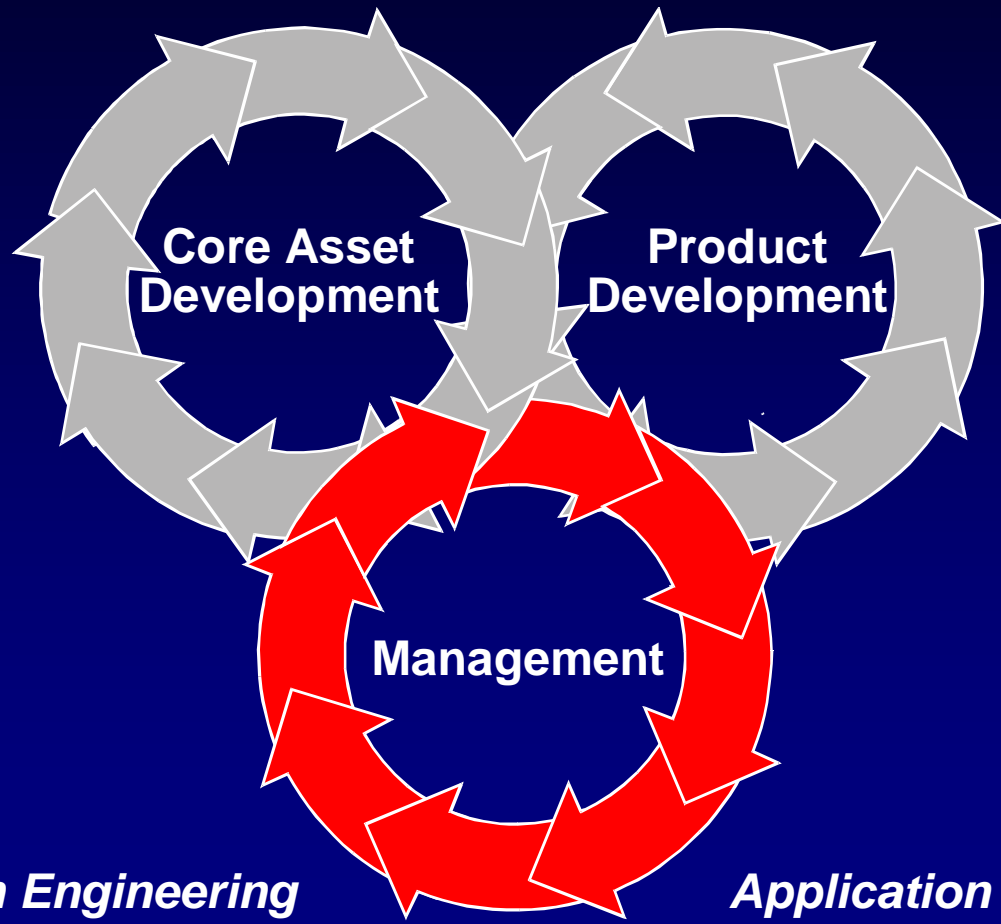


Product Development





Management



Domain Engineering

Application Engineering



Management

Management plays a critical role in the successful building of a product line by

- allocating resources
- coordinating and supervising
- achieving the right organizational structure
- rewarding employees appropriately
- providing training
- developing and communicating an acquisition strategy
- managing external interfaces
- creating and implementing a product line adoption plan



***Managing a software product line
requires leadership.***



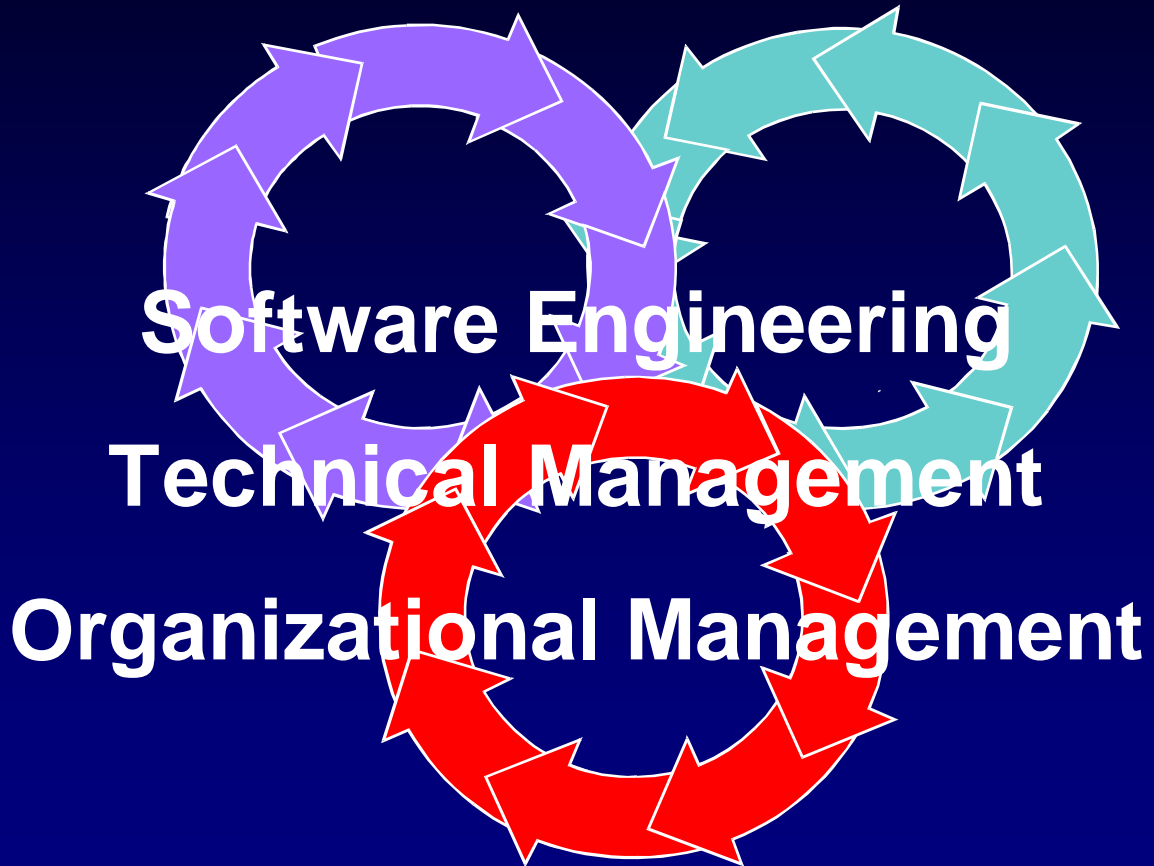
Driving the Essential Activities

Beneath the level of the essential activities are essential practices that fall into practice areas.

A **practice area is a body of work or a collection of activities that an organization must master to successfully carry out the essential work of a product line.**



Practice Areas Categories





The 29 Practice Areas

Software Engineering

Architecture Definition
Architecture Evaluation
Component Development
COTS Utilization
Mining Existing Assets
Requirements Engineering
Software System Integration
Testing
Understanding Relevant Domains

Technical Management

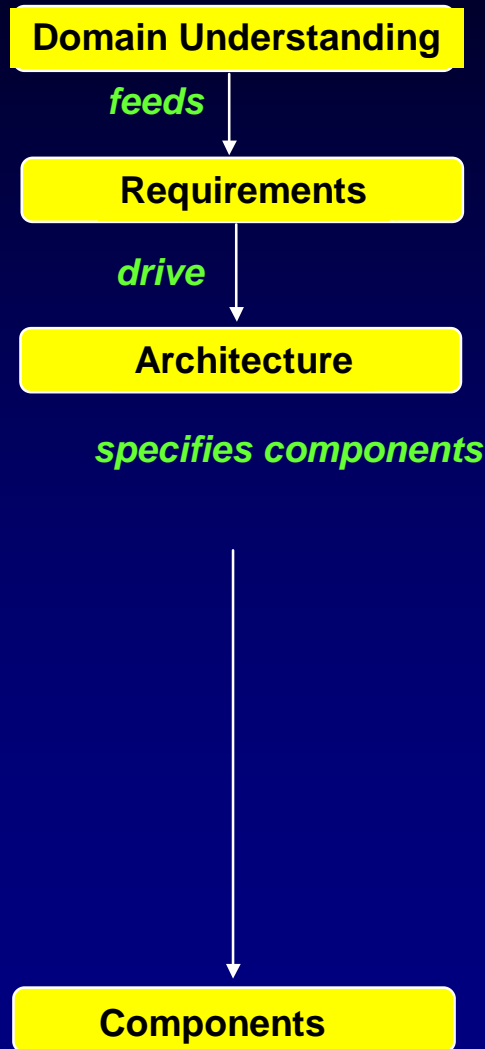
Configuration Management
Data Collection, Metrics, and Tracking
Make/Buy/Mine/Commission Analysis
Process Definition
Scoping
Technical Planning
Technical Risk Management
Tool Support

Organizational Management

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

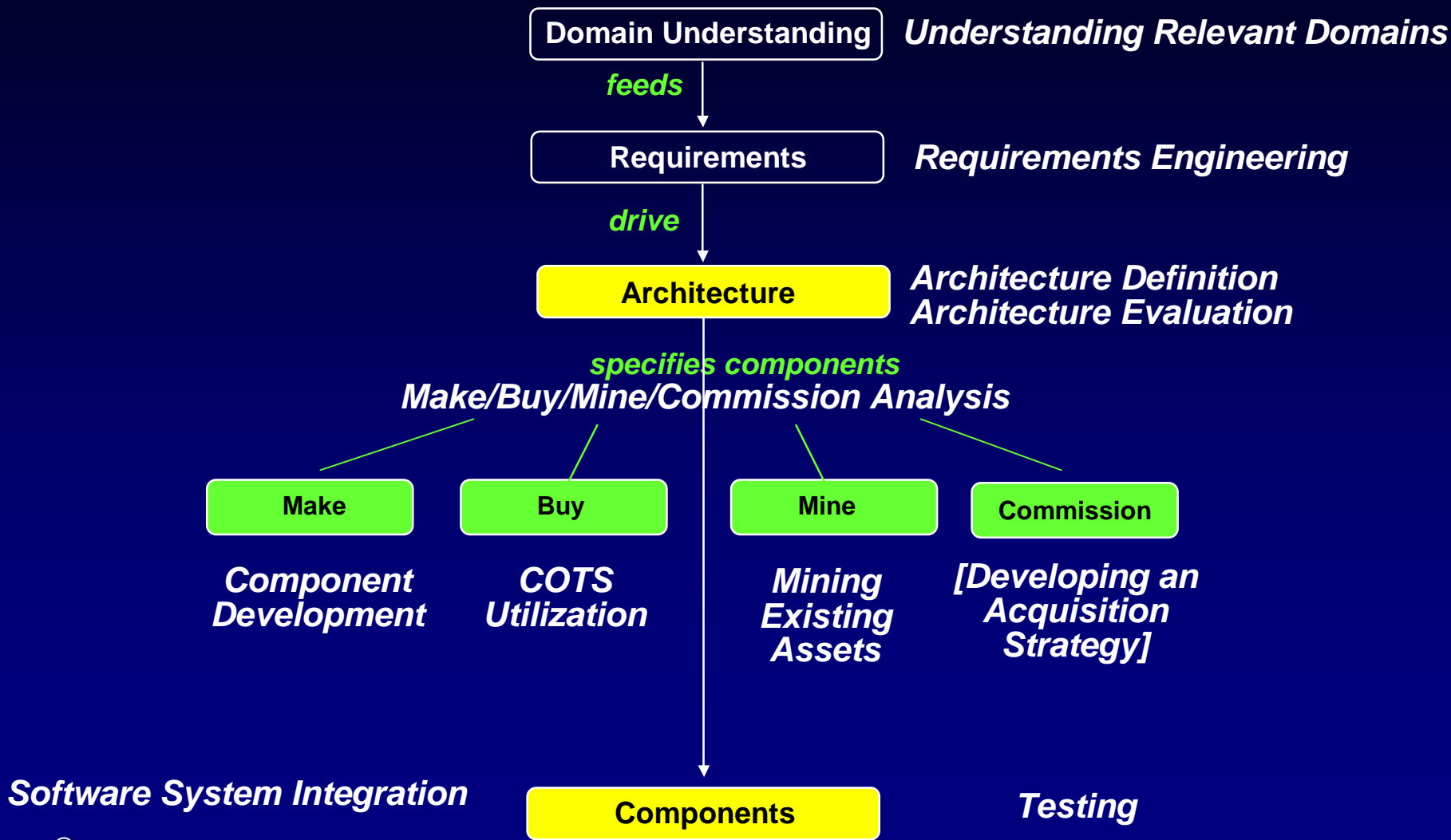


Software Engineering Practice Areas



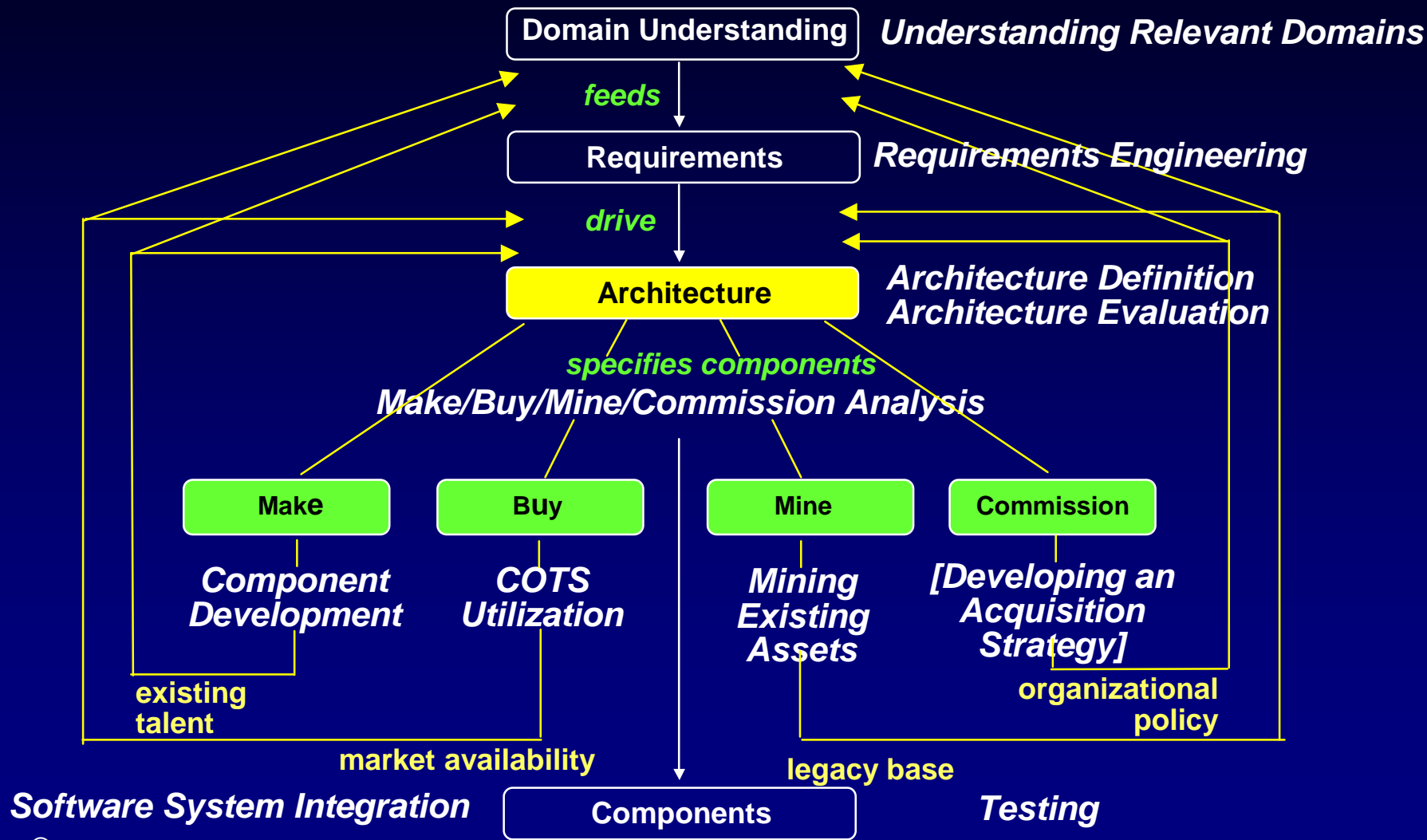


Software Engineering Practice Areas





Software Engineering Practice Areas





Closing Comments

Software product lines epitomize the concept of strategic, planned reuse.

The product line concept is about more than a new technology. It is about a purposeful re-invention of an organization, a disciplined way of doing one's software business.

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



What's Different About Reuse with Software Product Lines?

Business dimension

Iteration

Architecture focus

Pre-planning

Process **and product connection**



At the Heart of Successful Product Lines

A pressing need that addresses the heart of the business

Long and deep domain experience

A legacy base from which to build

Architectural excellence

Process discipline

Management commitment

Loyalty to the product line as a single entity



The Time is Right

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

A global realization of the ***importance of architecture***

A universal recognition of the need for ***process discipline.***

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

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

Company and inter-company ***product line initiatives.***

Rising recognition of the ***amazing cost/performance savings*** that are ***possible.***



Remaining Challenges

Definition of product line architectures

Evolution of product line architectures and assets

Product line migration strategies for legacy systems

Collection of relevant data to track against business goals

Funding models to support strategic reuse decisions

Acquisition strategies that support systematic reuse through product lines

Codified, integrated technical and management practices

Product line tool support



SEI Contribution

Practice Integration: A Framework for Software Product Lines, Version 3, <http://www.sei.cmu.edu/plp/framework.html>

- essential activities
- practice area identification and descriptions
- FAQ

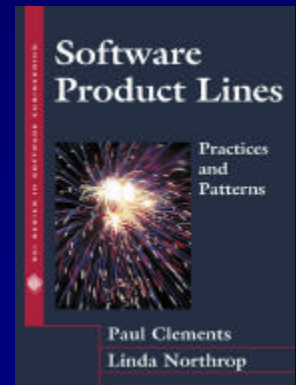
Techniques and Methods

- architecture definition – ADD
- architecture evaluation – ATAMSM
- mining assets – OAR
- product line analysis
- **Product Line Technical Probe**

Software Product Lines: Practices and Patterns

- practices
- patterns
- case studies

SPLC2 – August 19-22, 2002 – San Diego





Conclusion

If properly managed, the benefits of a product line approach far exceed the costs.

Strategic software reuse through a well-managed product line approach achieves business goals for:

- efficiency**
- time to market**
- productivity, and**
- quality**

Software product lines are reuse that pays.