System Architecture

IAP Lecture 6

Ed Crawley

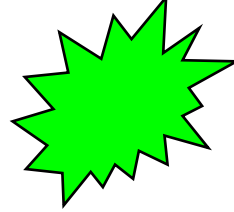
Jan 30, 2007

Rev 2.0

Today's Topics

- **Reflection on Needs, Goals for a Medium System**
- **Complexity, Modules and Interfaces**
- **Reflections on the PDP**
- **A Holistic View and Summary**
- **The Role of the Architect**
- **Summary - Learning Objectives**
- **The Architecture of SDM - a Worked Example**

Needs, Goals for Medium System



- **Who is the principal beneficiary? What are their needs?**
- **Who are other beneficiaries? What are their needs?**
- **What is the intent associated with the externally delivered process which are traceable to customer needs? Are there any intents apparent which flow from regulations, corporate strategy, competition or the corporate technical competence?**
- **What is the suppressed object or suppressed process structure of the level 1 elements?**

Summary - Needs

- **Needs exist in the heart and mind of the beneficiary**
- **They exist outside the enterprise**
- **They are fuzzy, ambiguous and ill stated**
- **They must be identified and understood**

- **A beneficiary often has more than one need to be met by a product system**
- **There is often more than one beneficiary**

Summary - Goals

- **Goal is defined as**
  - **what it planned to be accomplished**
  - **what the designer hopes to achieve or obtain**
- **Expressed in the precise terms of Product Development**
- **Will include goals derived from beneficiary Needs (goals from beneficiaries) i.e. the functional goals**
- **Will also include goals from corporate strategy, regulations, competitive analysis, etc.**
- **Embodied in a statement of goals (requirements ?)**
- **Is defined (in part) by the architect**
- **Exist within, and under the control of the enterprise, and are traded against other attributes**

What is Good Architecture?

- **Answer #1**
- **That which meets the goals of the system**
- **Which are based on the important needs of key stakeholders**

- **Must start with all of this upstream ambiguity and resolve important needs and consistent goals**
- **Constrain the problem by a the mission and outcomes of the enterprise**

Complexity and Interfaces

- **Complexity, revisited**
- **Sources of complexity**
- **Interfaces**
- **Specifying Architecture**

Complexity

Defined:

- **Having many interrelated, interconnected or interwoven elements and interfaces**

Therefore

- **A complex system requires a *great deal of information* to specify (the really important feature of complexity)**
- **Complexity is an absolute and quantifiable system property (once a measure and atomic level are defined)**
- **Apparent complexity is the perception that something is complex. Complicated things have high apparent complexity**

It is the role of the architect to manage the evolution of complexity in such a way that a complex system does not appear complicated

What Makes a System Complex?

- Requiring more functionality
- Requiring more performance in the functionality
- Poor alignment of object structure and internal processes
- Reuse of legacy
- Supplier relationships
- Flexibility and platforming

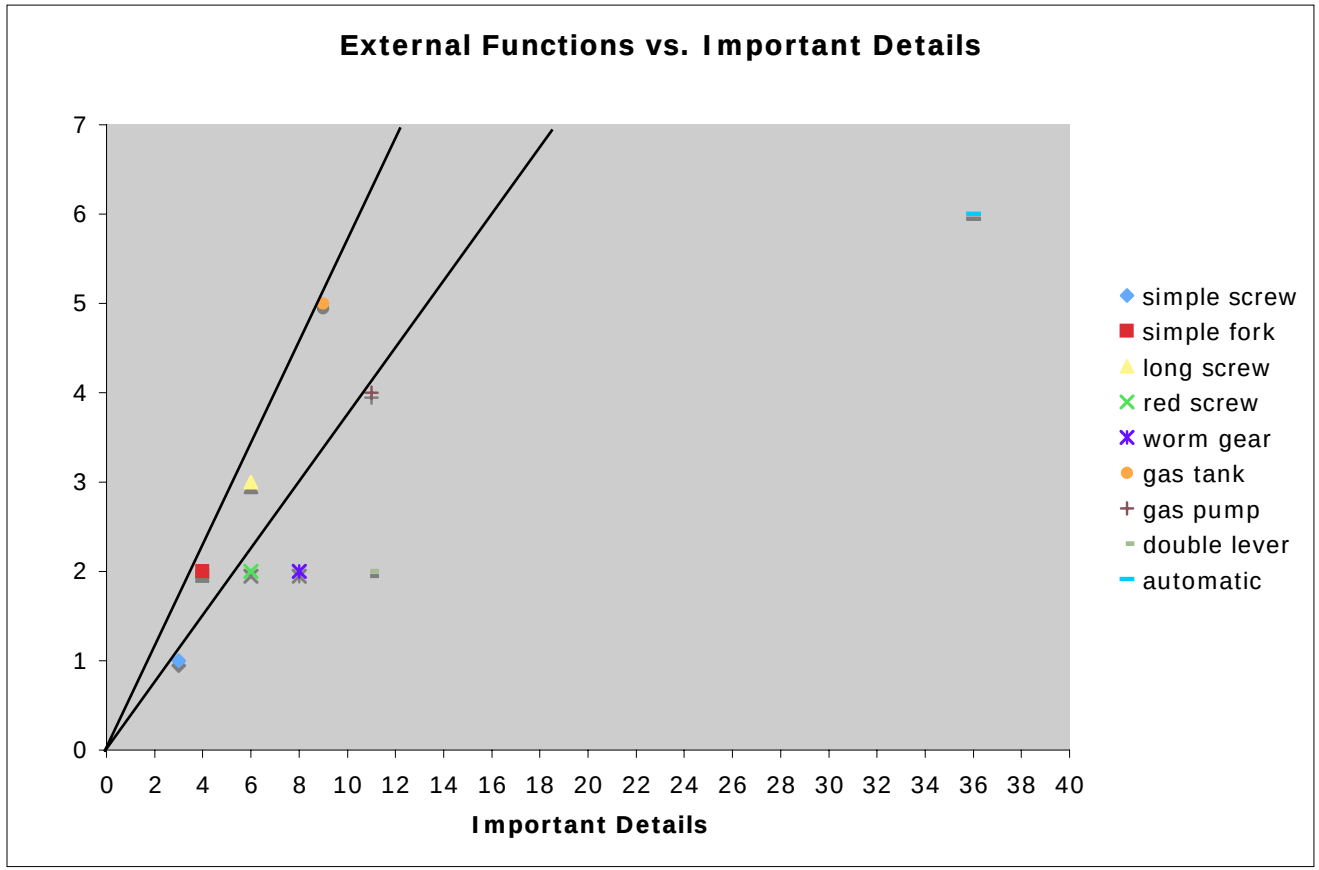
Dirt Simple Measure of Complexity

- **Number of things: N_{things}**
- **Number of types of things: $N_{types\_of\_things}$**
- **Number of connections among things: $N_{connections}$**
- **Number of types of connections: $N_{types\_of\_connections}$**

- **Simplest measure that captures all of these is the sum:**
- **$C = N_{things} + N_{types\_of\_things} + N_{connections} + N_{types\_of\_connections}$**

Delivered Functions Drive Complexity

- Delivered function for a “cork translater”
- Important elements as “atomic parts”, not apart parts
- Delivered functions include:
  - Engage cork
  - Magnify force
  - Create force
  - Aid engagement
  - Auto engagement
  - Aid disengagement
  - Auto disengagement



$$C = N_{\text{important details}}$$

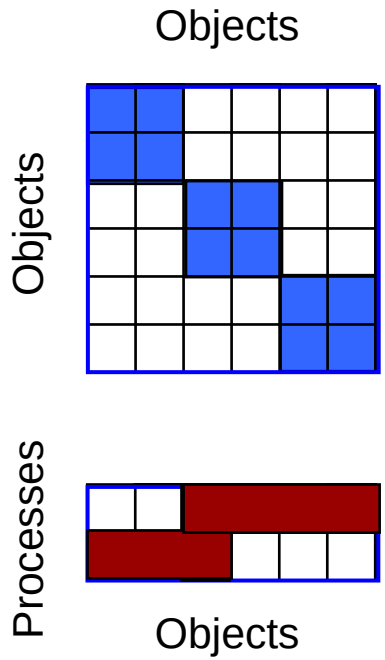
What is Good Architecture?

- **Answer #2**
- **That which uses appropriate and creative concepts (but not too creative) to meet the statements of solution neutral function while providing for minimum complexity**

- **Must explore the space creatively,completely and rationally**
- **Must seek combinations of concepts to meet various functional requirements**

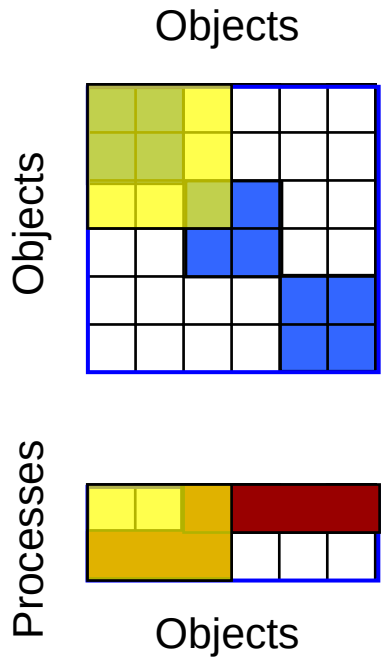
Poor Alignment of Object Structure and Processes

- **Do 2 down, 1 up on the object structure**
- **Map the level 2 objects to internal process**
- **What happens if they don't align?**



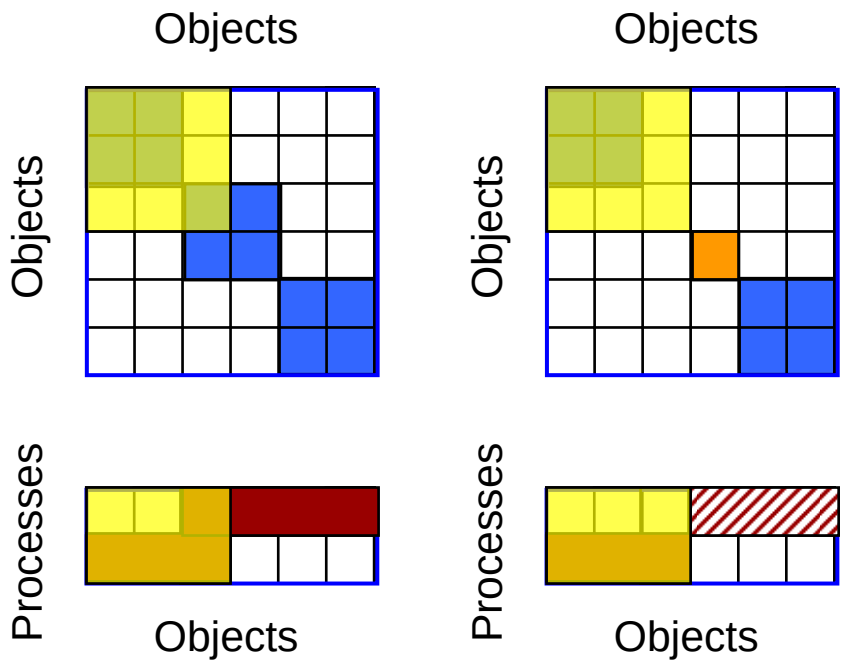
Reuse of Legacy

- **<u>Legacy</u> - inherited form, and associated function, from a previously built product/system**
- **Legacy elements have some structure and mapping to process**
- **First challenge is to identify this!**
- **Then how to deal with the fact that boundaries in the legacy elements may not be where you want them now**



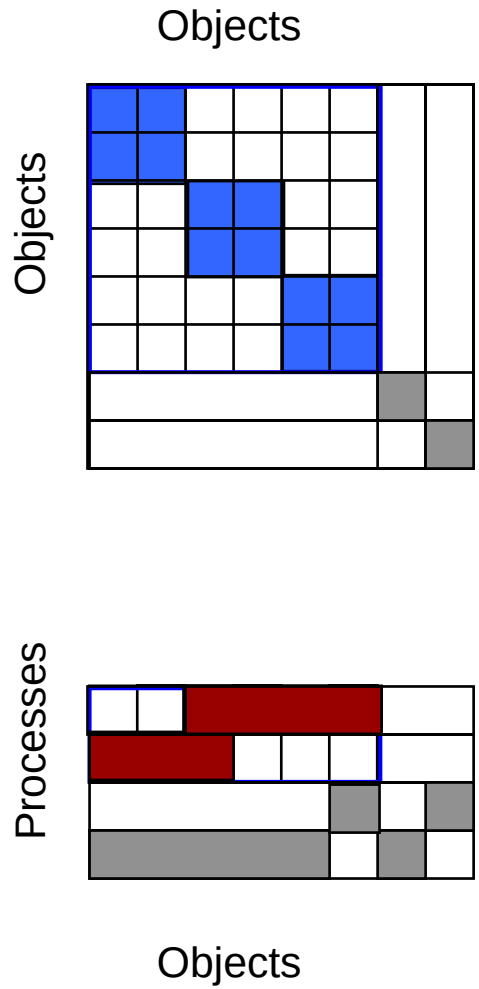
Platforming

- **<u>Platform</u> - common elements of form used in more than one product/ system**
- **Try to make the common elements of the platform common**
- **Try to fit the other elements that distinguish the individual products around the common elements**



Supplier Relations

- **Extend the object structure to include, for example, suppliers**
- **Extend the process object map to include the supplying process**
- **Now the supplier relations do not align with either the structural or object-process blocks!**



Modules and Parts

- A module is a collection of (1...n) parts which are defined by some intent to be a distinct system
- The intent of the module can be one or more of use:
  - For implementation, sometimes called assemblies
    - Ease of integration
    - Product flexibility - the basis of platforms
  - For routine user operation
  - For service/ evolution/ upgrade (can sometimes be at various levels - field or line replaceable, depot replaceable, ...)
- Part:
  - (An apart part) An element that you cannot take apart and then reconstitute in its original form, or
  - (A functional part) An element that you cannot take apart without destroying its ability to deliver function
- An “atomic part” can be a part or an important element within a part

Interfaces - Complexity

- **Complexity *arises* in a system as more is asked of it (performance, functionality, robustness, etc.)**
- **Complexity manifests itself as the interfaces between elements or modules are defined**
- **Complexity can be *exploited* to create platforms, etc.**
- **It is the role of the architect**
  - **To keep the actual complexity as low as possible**
  - **To keep the system from becoming too complicated**

Complexity is inherently neither good nor bad, but must be managed

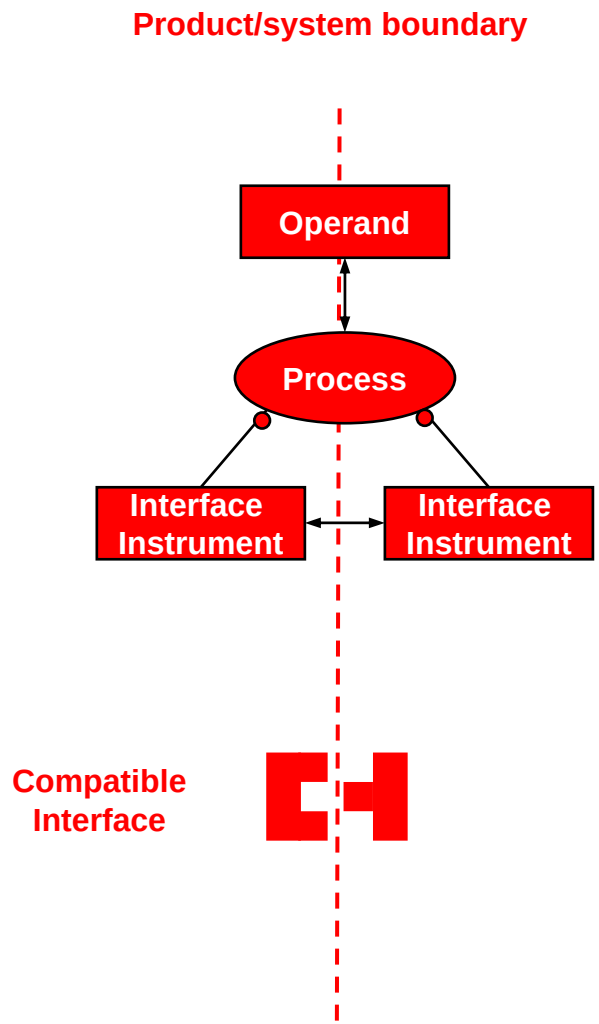
Interfaces

- **Interfaces are the points of contact between interacting elements which are modules and parts**
- **Interfaces are central to the definition of a system –**

  "A set of elements ..." must be connected at interfaces to perform "function greater than the sum of the parts"
- **Interfaces have important characteristics:**
  - **Object and process nature (discussed earlier)**
  - **Complexity of the interface**
  - **Stability of the interface**

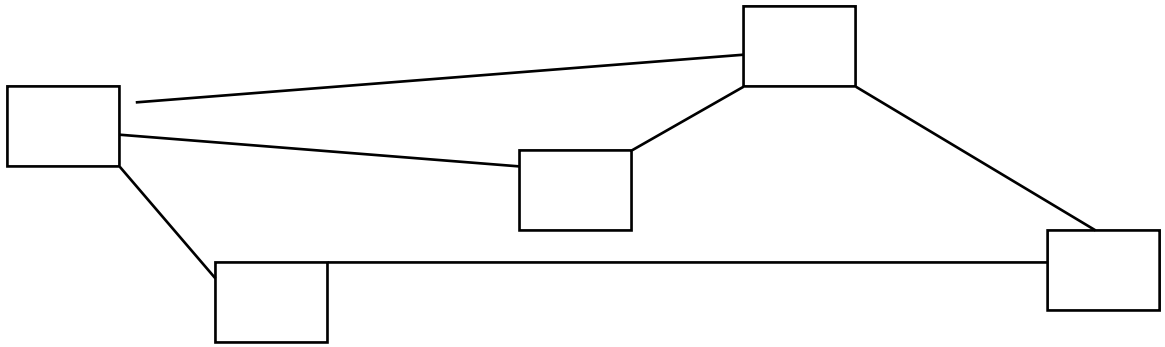
Interfaces Have Form and Function

- **The structure usually indicates the *existence* of an interface (more about this next time)**
- **At the interface:**
  - **Form has some structural relationship - usually compatible**
  - **A function is performed - usually the process is the same or the complement**
  - **The operand is the same**

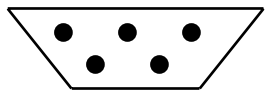


Complexity in Interfaces

Number

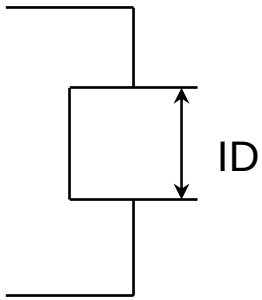


Sophistication



0 V = off

2 V = on



heapstatus =
heapcheck();

Sensitivity/Robustness

0 - 0.5 V = off

1.5 - 3.0 V = on

ID = XXX

+/- .YYY

Is there any?

Important pattern here is increasing information that must be specified

Stability of the Interface

- Some interfaces are created just for internal use, and are shorter lived or ad hoc in nature
- Other interfaces are created for internal use, but are intended to be long term and stable in nature
- Other interfaces are created for external use, which define the interface of the product/system with supplier modules or other elements of the whole product system - these are very stable
- Other interfaces are defined to be standards - the most stable

Define stable interfaces very carefully, they will be with the system for a long time, and are hard to change!

Why are Interfaces Important?

- Define the “system nature” of the product
- Are an area of great leverage
- Are an area of great uncertainty
- Are key to
  - assembly/integration/checkout
  - maintainability
  - product evolution and adaptability

Interface definition and control are an area of prime importance to the system architect

What Informs the Definition of Interfaces?

- **Vision of:**
  - **assembly/integration/checkout**
  - **maintainability**
  - **product evolution and adaptability**
- **Need to:**
  - **Build around legacy elements**
  - **Incorporate supply chain elements**
  - **Fit into product platform constraints**
  - **Accommodate standards**
- **The different classes of structural links inform the definition of different types of modules:**
  - **Implementation links - implementation modules**
  - **Operational links - operator modules**
  - **Spatial/topological links - service/upgrade modules**

Much more about this in the fall

Specifying Elements

- **Three possible modules of how you specify an element to a "supplier"**
  - **interfaces and functions**
  - **interfaces, functions and concept**
  - **interfaces, form [build to print]**
- **Sometimes when decomposing is is desirable to decompose function and/or form to get to supplier product/capability - *supply chain* impact on architecture**

Summary: Complexity and Interfaces

- **Complexity accumulates in a system is more is asked of it, and for a number of other reasons**
- **Modularity is a key way of dealing with complexity, but requires the definition and control of interfaces**
- **Interfaces occur between parts and modules, are essential to the nature of a system, and are an important area of focus for the architect - especially the stable ones**
- **The apparent complexity of a system at any level of decomposition is dependent on the definition of the interfaces**
  - **It establishes the number and type of modules**
  - **It establishes the number and type of interfaces**
- **The definition of modules and interfaces is informed by function, structure, suppliers, platforming, and many other factors**

What is Good Architecture?

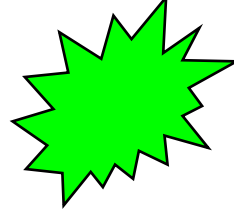
- **Answer #3**
- **That which has a set of internal processes, and a well structured set of instrument objects, so that the external value related function emerges**
- **That which resolves all of these possible inconsistencies in the modularization**

- **Must carefully decompose the system, and ensure soundness of architecture**
- **Must seek combinations of objects and processes to meet various functional requirements**

PDP Synthesis

- Reference PDP
- Alternative models
- Key distinctions
- Underlying Principles

Reflections on PDP

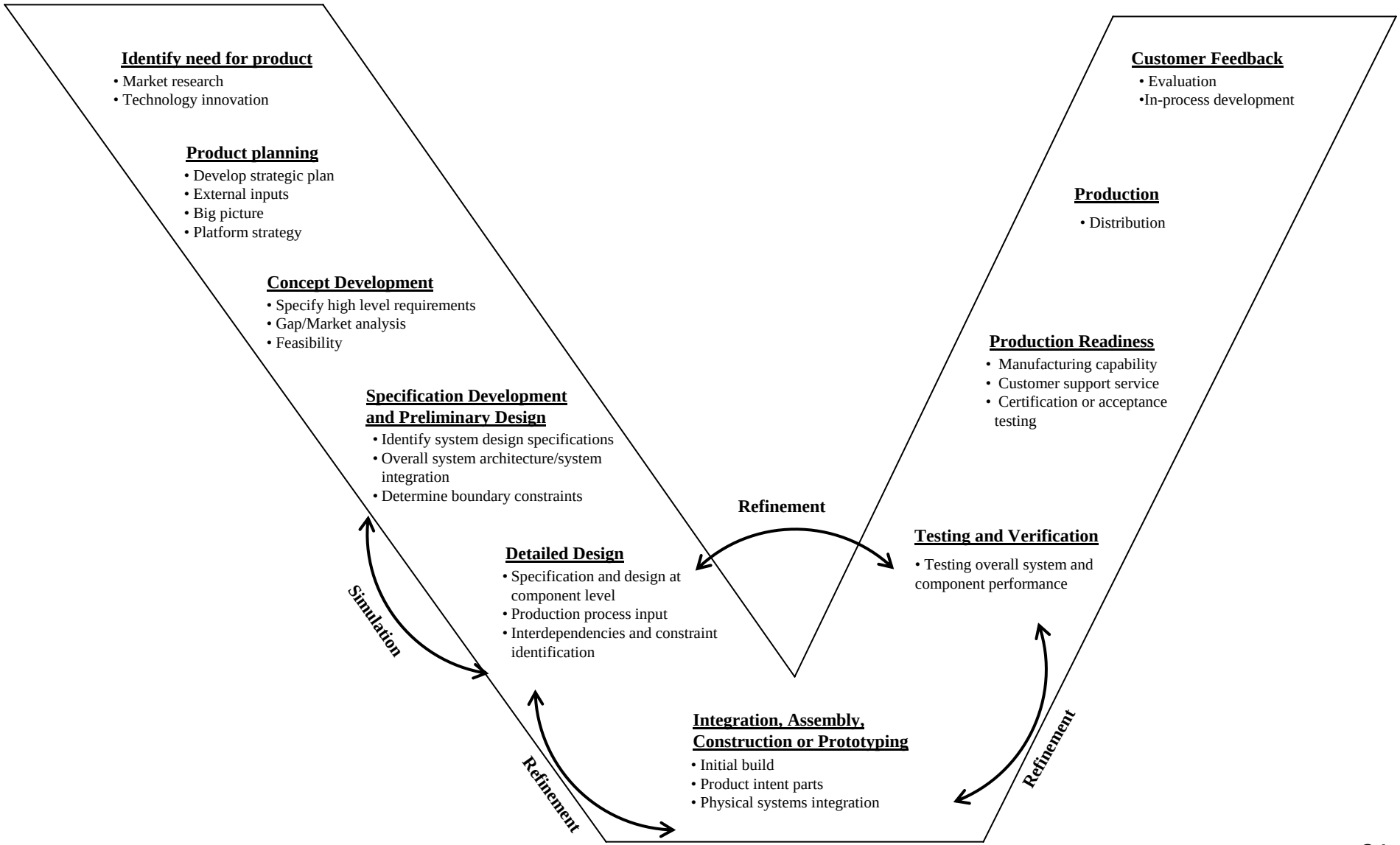


- How common are the various PDP's?
- What are the prime distinguishing features?
- What is the origin of the distinguishing features?

Case Study Examples

[Several slides with examples removed due to copyrighted and company-proprietary information.]

Generic Product Development Process



Key Differences in PDP's

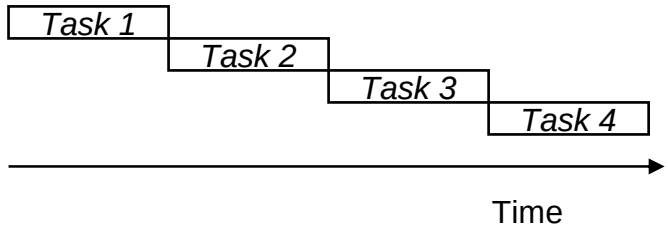
- Number of phases (often a superficial difference)
- Phase exit criteria (and degree of formality)
- Requirement "enforcement"
- Reviews
- Prototyping
- Testing and Validation
- Timing for committing capital
- Degree of "customer" selling and interference
- Degree of explicit/implicit iteration (waterfall or not)
- Timing of supplier involvement

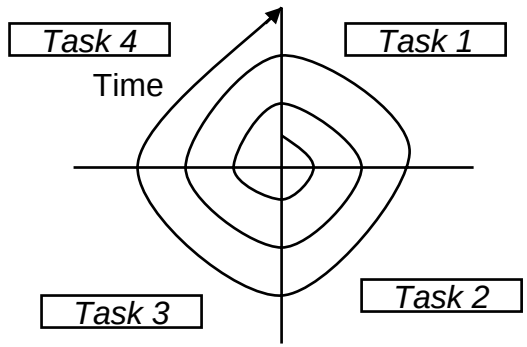
Key Drivers of Differences

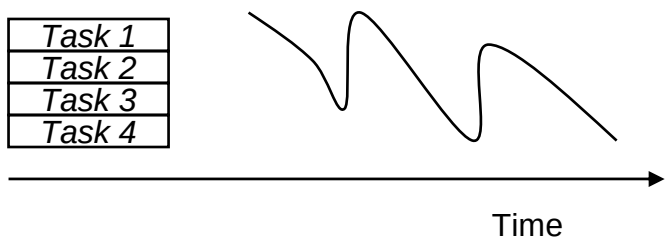
- **Product**
  - **HW vs SW**
  - **Stand alone vs. platform**
  - **Requirements for Generic or Specialized Manufacturing**
  - **Production volume 1 - many**
  - **Capital Intensity**
  - **Technology Push/Market Pull/Process Intensive**
  - **Complexity/Size/Dynamic Nature**
  - **Regulatory certification or standards compliance**
- **Market:**
  - **True customer, OEM, Small Market, Government**
  - **Duration of Life Cycle**
  - **After sale support and service, revision/upgrade process**
- **Culture/Strategy**
  - **Management control**
  - **Resource allocation process**

Waterfall and Spiral

- **Waterfall model has sequential (overlapped?) tasks**
- **Spiral model has iteration on tasks**
- **In fact all processes have some combination of sequence, overlap and iteration, this is "much ado about nothing"**

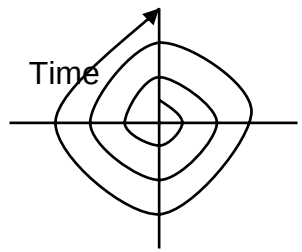
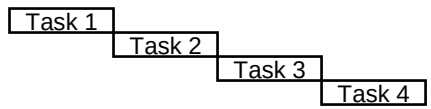






Another View of PDP Design

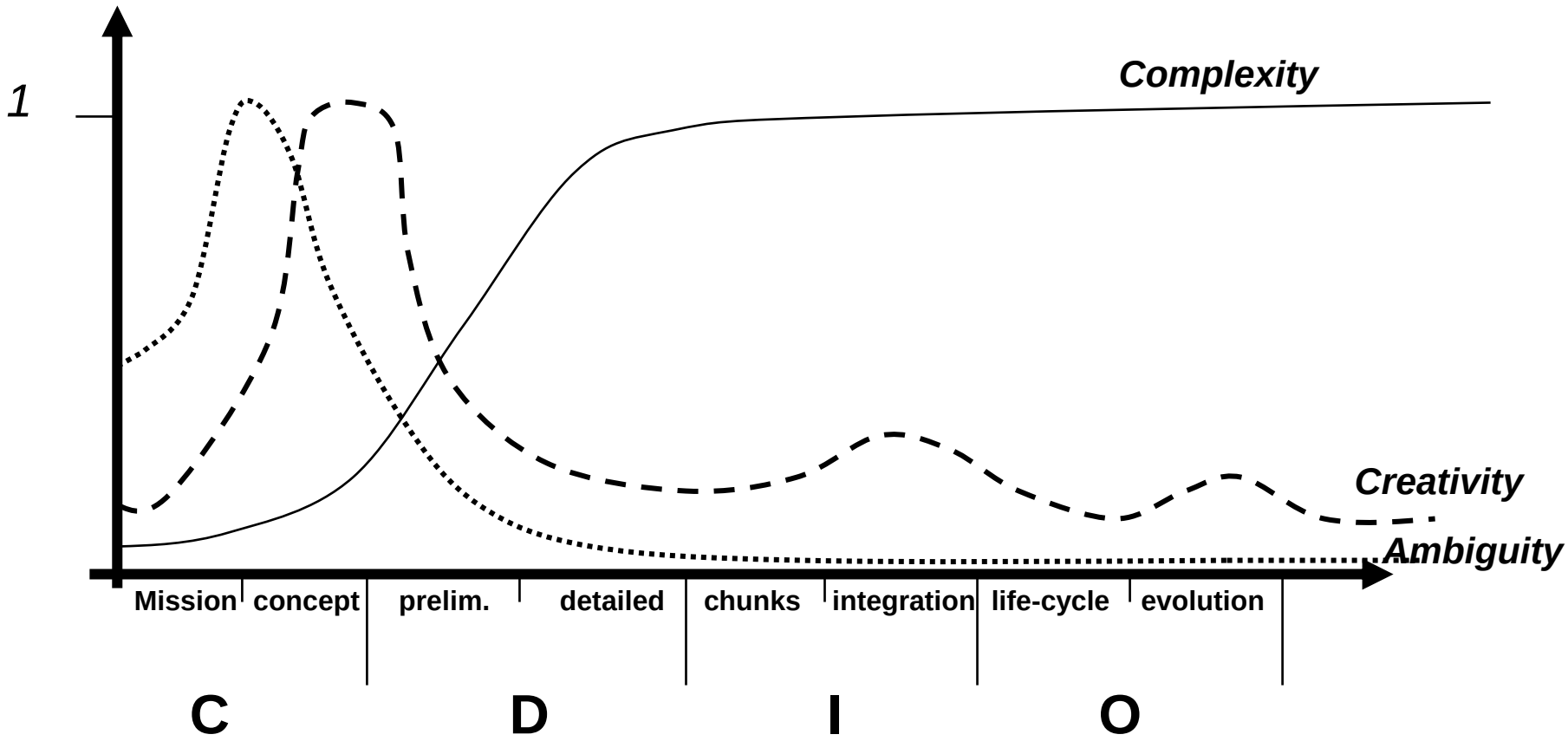
- **Nature of PDP design depends on degree of system complexity and precedent**

	Complexity: Less complex	More complex
Precedent: None		?
Some:		V

Generic PDP

Conceive	Design	Implement	Operate
Mission Conceptual Design	Preliminary Design Detailed Design	Element Creation Integration, System Test	Life Cycle Support Evolution
• Business Strategy • Functional Strategy • Customer Needs • Competitors • Program plan • Business case • Goals • Function • Concepts • Regulation • Technology • Platform plan • Supplier plan • Architecture • Commitment	• Requirements definition • Model development • Requirements flowdown • Detail decomposition • Interface control • Design elaboration • Goal verification • Failure & contingency analysis • Validated design	• Sourcing • Implementation ramp-up • Element implementation • Element • testing • Element refinement • Product integration • Product testing • System testing • Refinement • Certification • Market positioning • Delivery	• Sales, Distribution • Operations • Logistics • Customer support • Maintenance, repair, overhaul • Upgrades • Product improve-ment • Family expansion • Retirement
Envision	*Design*	*Develop*	*Deploy*

Themes from the Perspective of the Architect

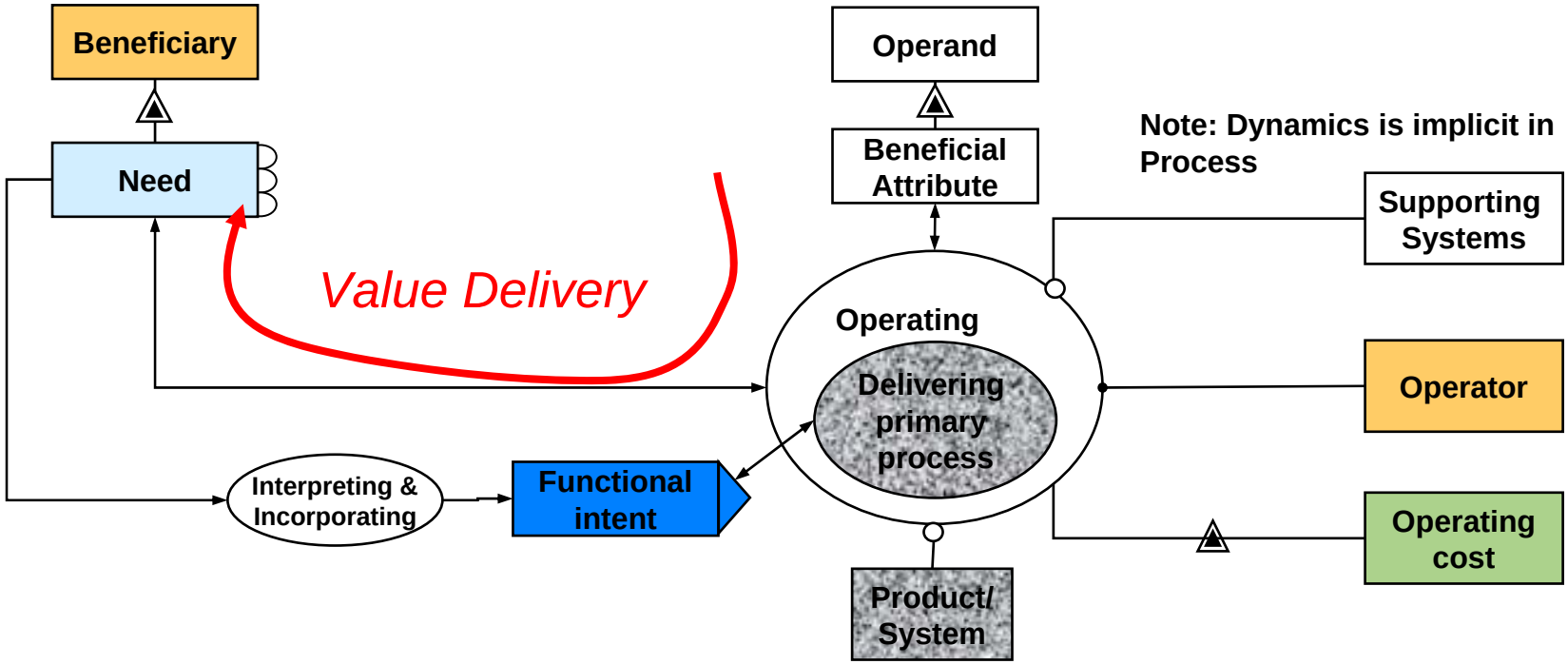


What is a PDP?

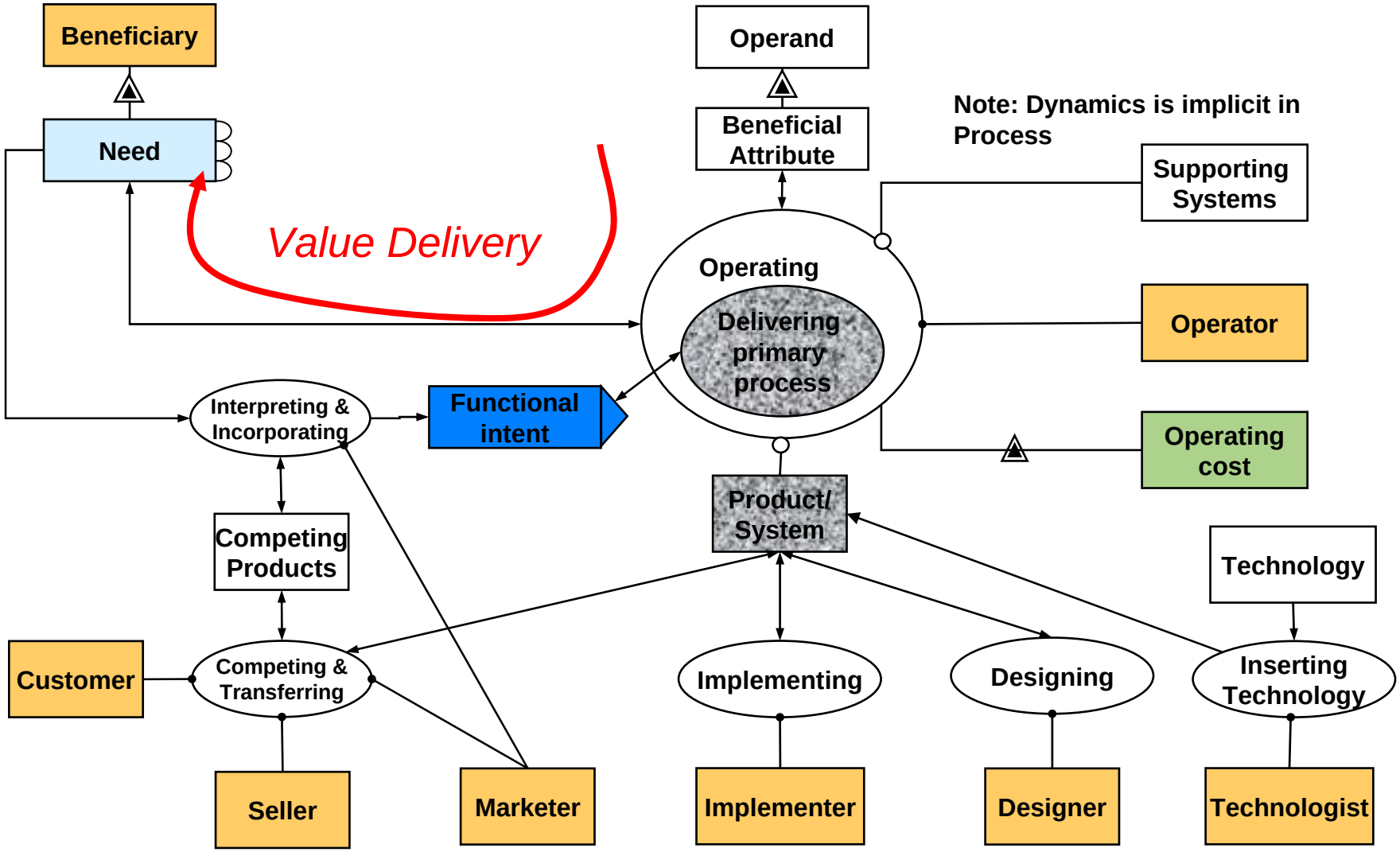
- **Principle?**
- **Process?**
- **Tool?**

So what are underlying principles?

Product Attributes



Product Attributes, with PDP



Costs

- Costs are traditionally divided into:
- Operational Costs, born by the operator

Operating cost

- Development Costs, born by the developer:
  - Cost of developing or acquiring the technology
  - Cost of designing the product (non-recurring)

R&D Costs

  - Costs of implementing (non-recurring and recurring)

COGS

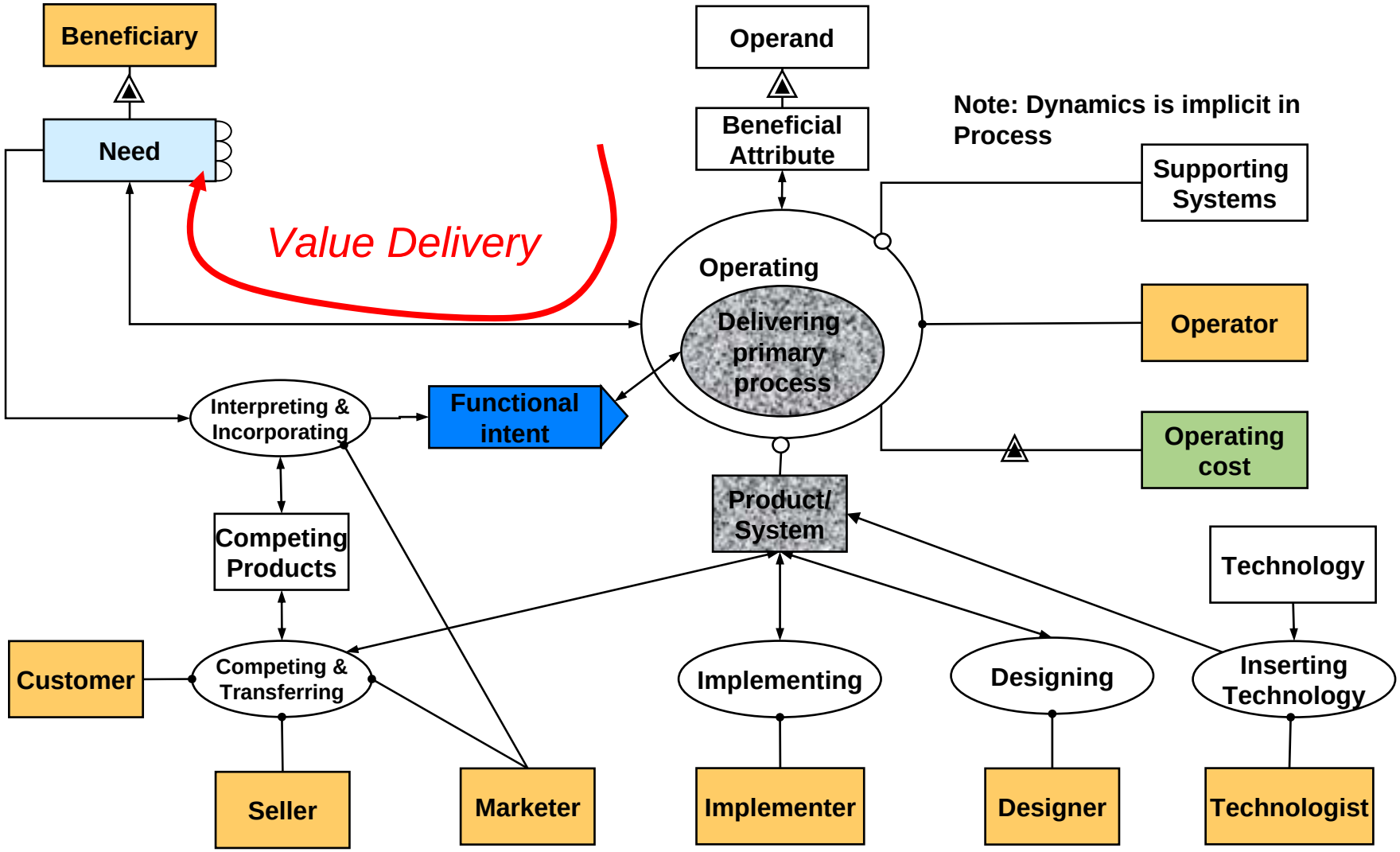
  - Costs of marketing, selling

SG&A Costs

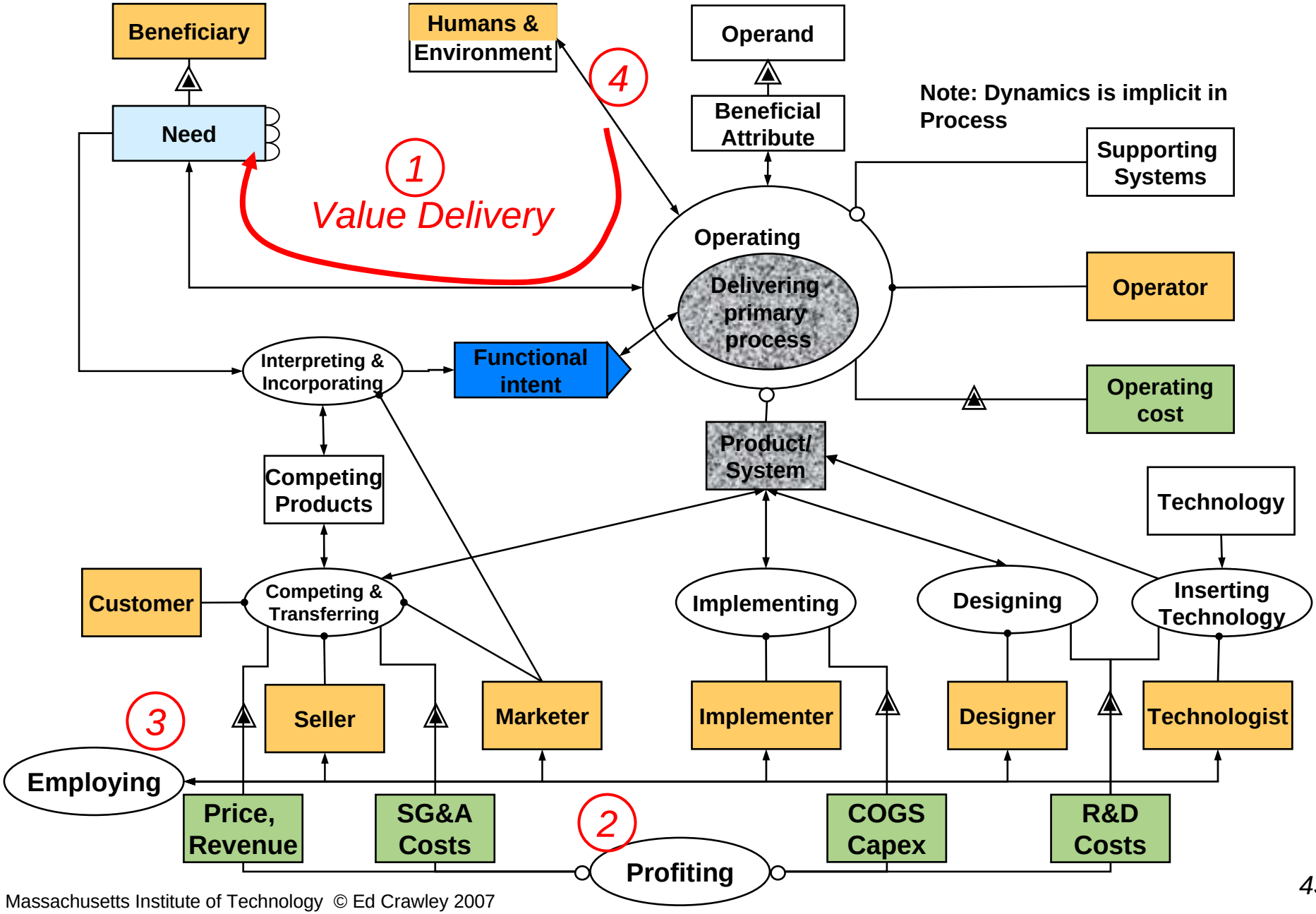
- An of course price of sales

Price, Revenue

Product Attributes, with PDP



Product Attributes, PDP and Costs



4 Main Goals of a (for profit) Enterprise

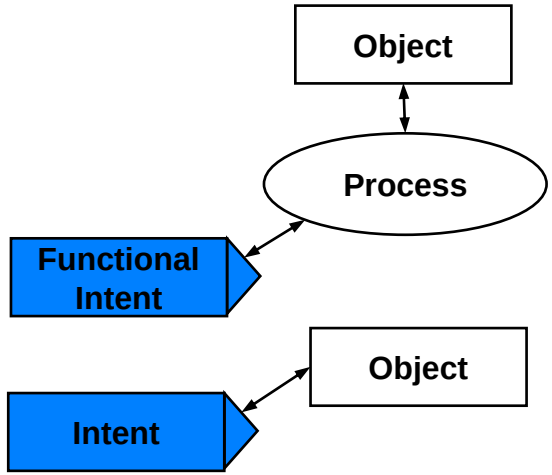
1. Delivering value to the beneficiary
2. Profiting
3. Providing meaningful and stable employment
4. Creating no unintended consequences for society or the environment

Types of Goals

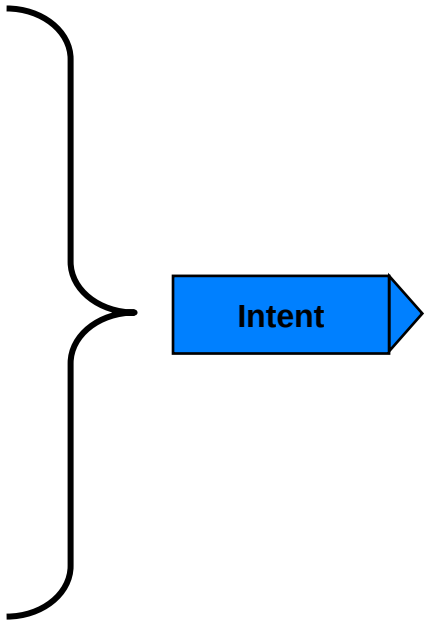
Goals are a mixture of several types:

- **Intent on Function**

- **Intent on instrument Object**

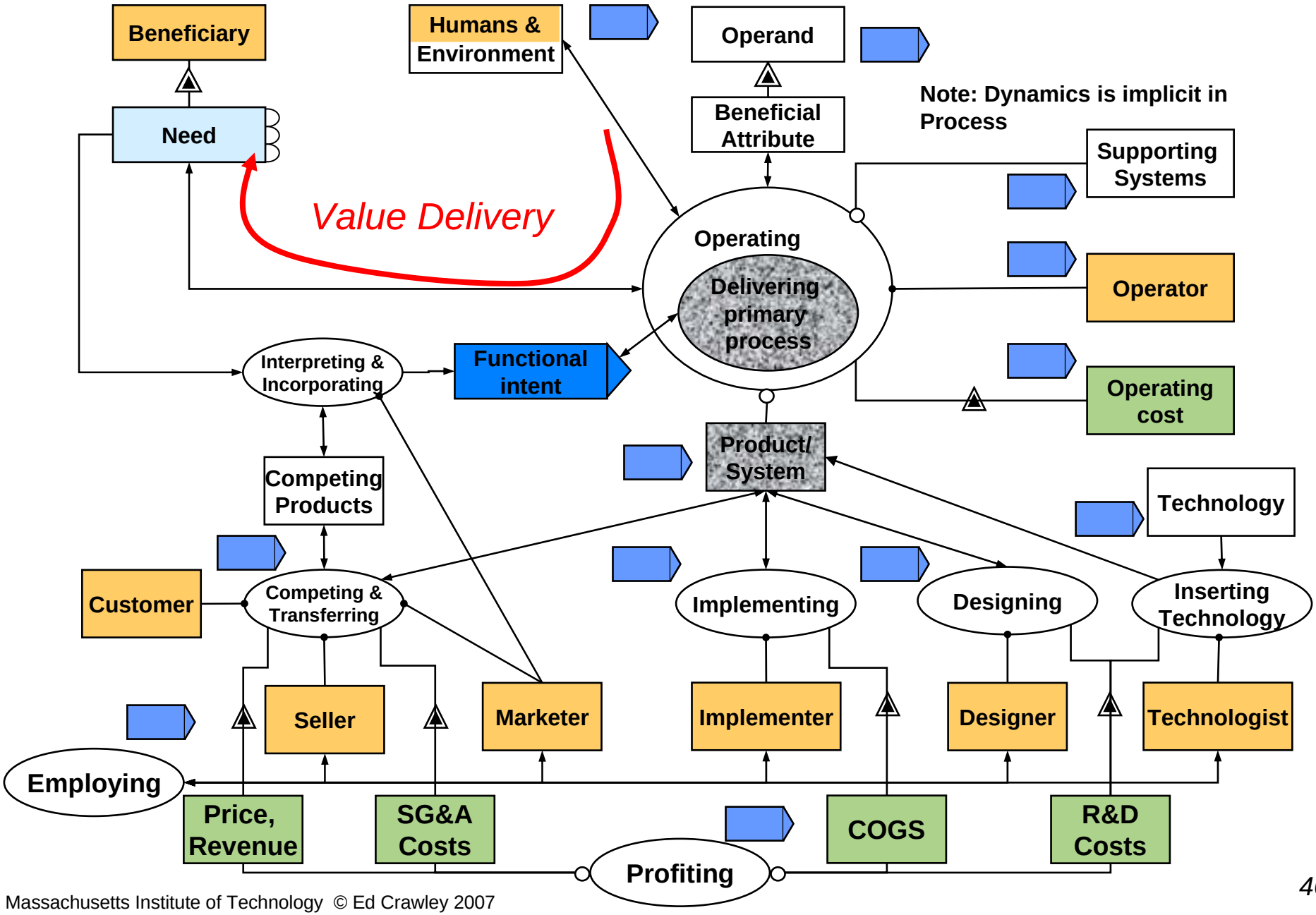


- **Intent on the design process**
- **Intent on the implementation process**
- **Intent on the operation process**
- **Intent on the strategic process**
- **Intent on the market process**
- **Intent on the regulatory process**
- **Intent on the technology process**



See the pattern???

Product Attributes, PDP, Costs and Intents



Summary: PDP

A structured PDP

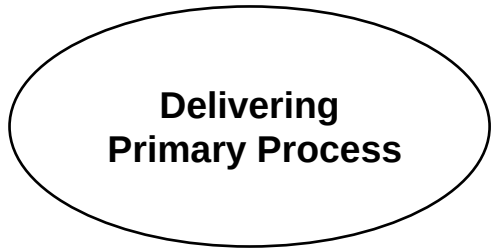
- **Increases value added, efficiency and competitiveness (e.g. time to market) of the process**
- **Provides something that can be learned and improved**
- **Should be customized to product/market/culture**
- **Should be based on underlying principles**

A Summary by “Zooming Out”

- Value related externally delivered process
- + other processes
- + form = architecture
- + other product attributes (upstream and downstream)
- + other design and implementation attributes
- + enterprise
- + society
- = a holistic view

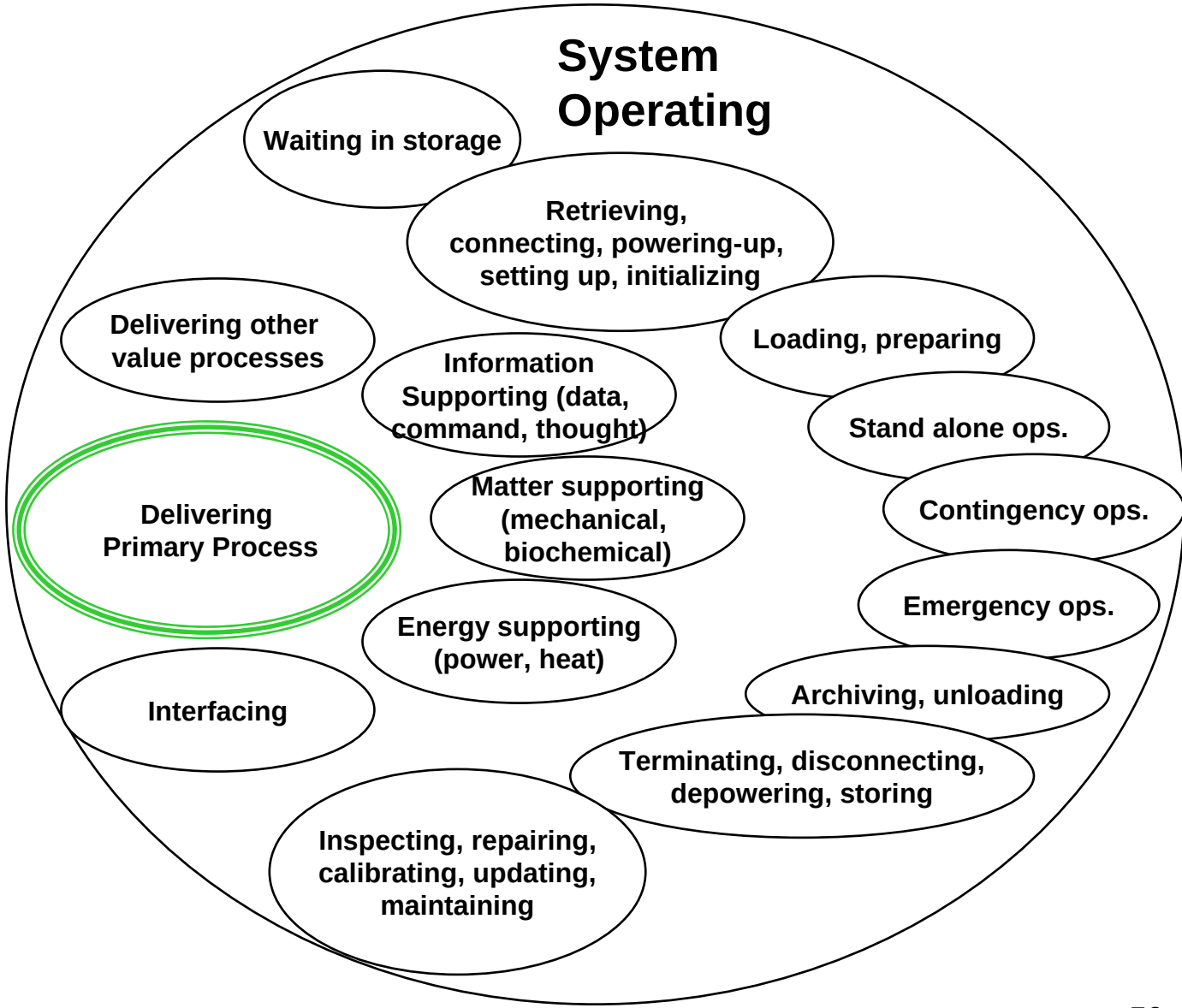
Focus First on the Value Process

- **Process associated with value**
- **Delivered to the value related operand**
- **With utility to the beneficiary**

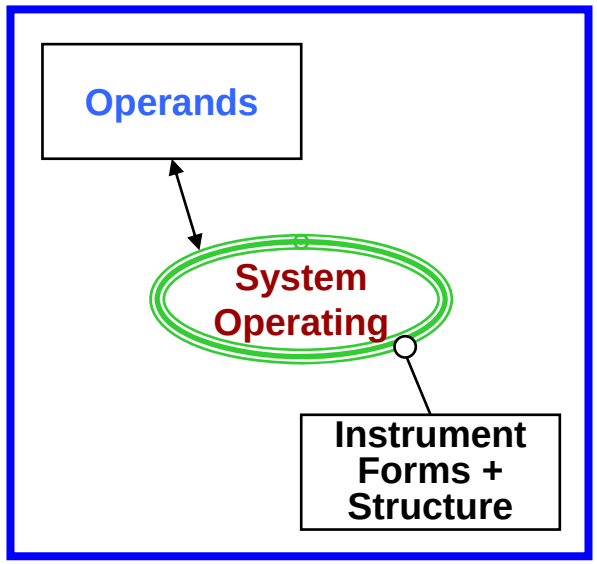


System Operating

- Identify other important processes
- Based on:
  - Supporting processes (5 fundamental process types)
  - Operational sequence

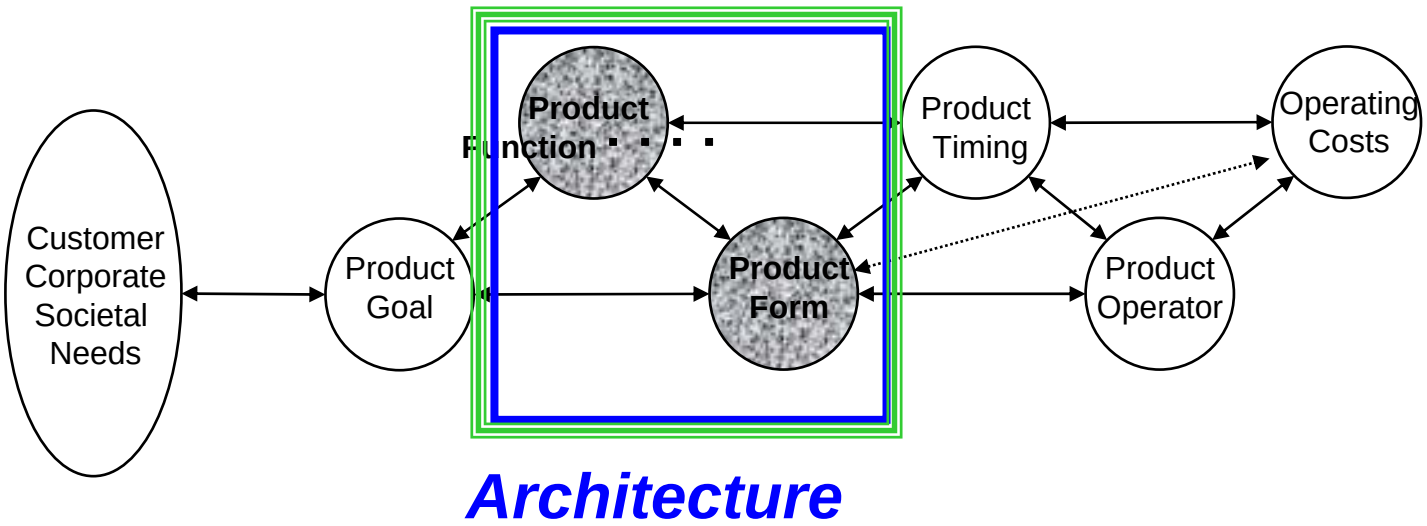


The Product Architecture

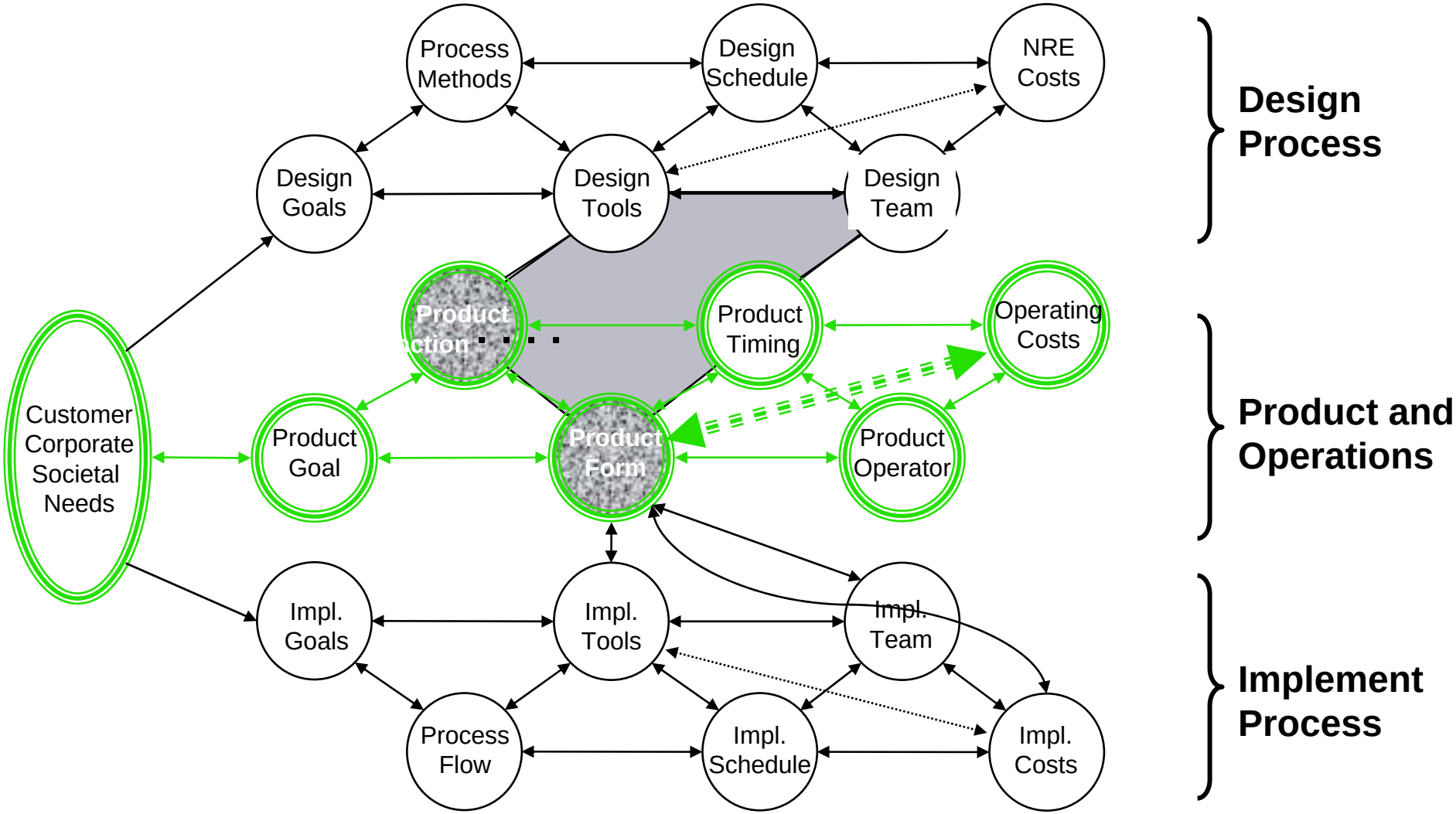


Architecture

Summary - Product Attributes

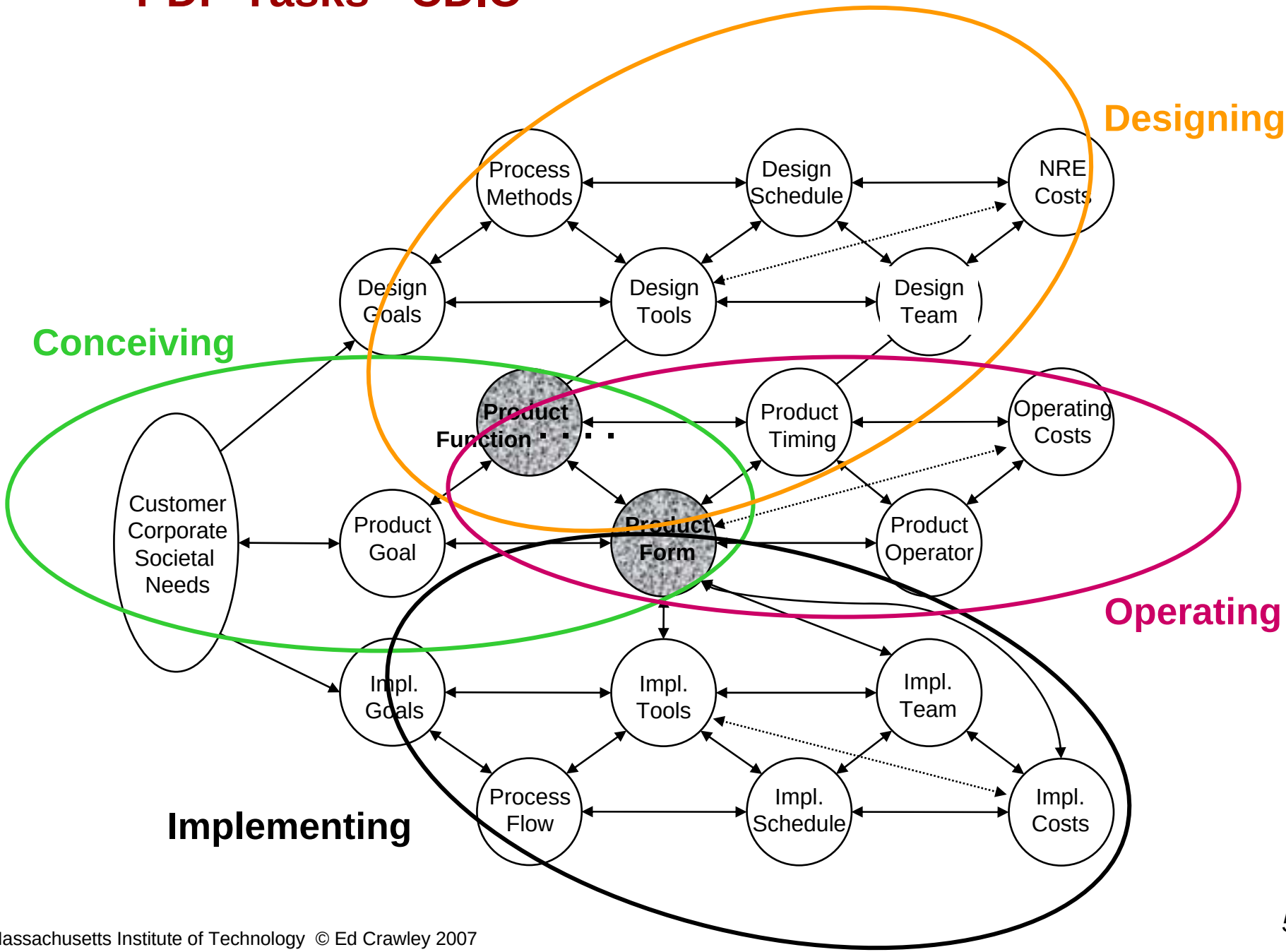


Holistic Framework for Product, Design & Implementation Processes



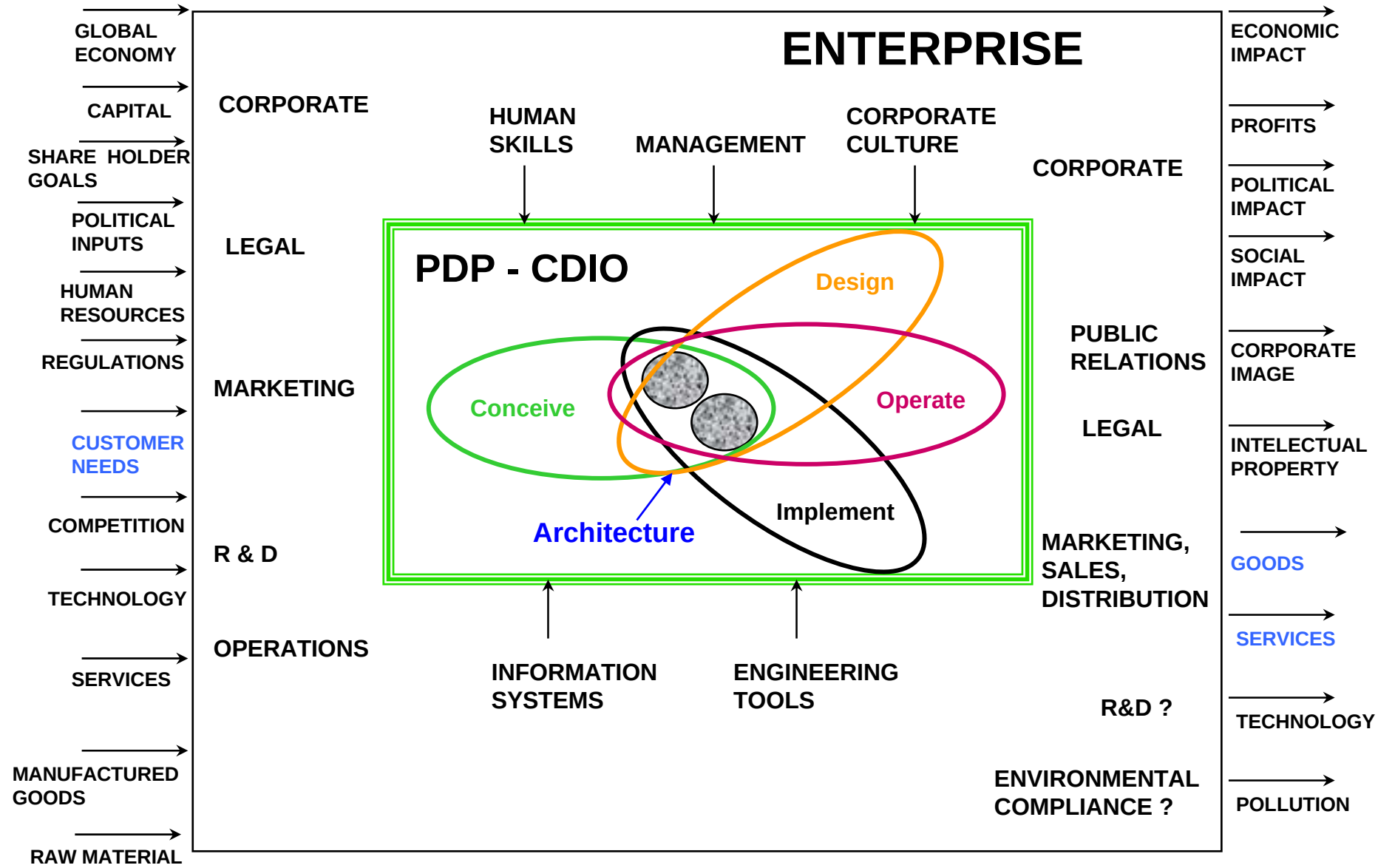
Why **What** **How** **Where** **When** **Who** **How Much**

PDP Tasks - CDIO



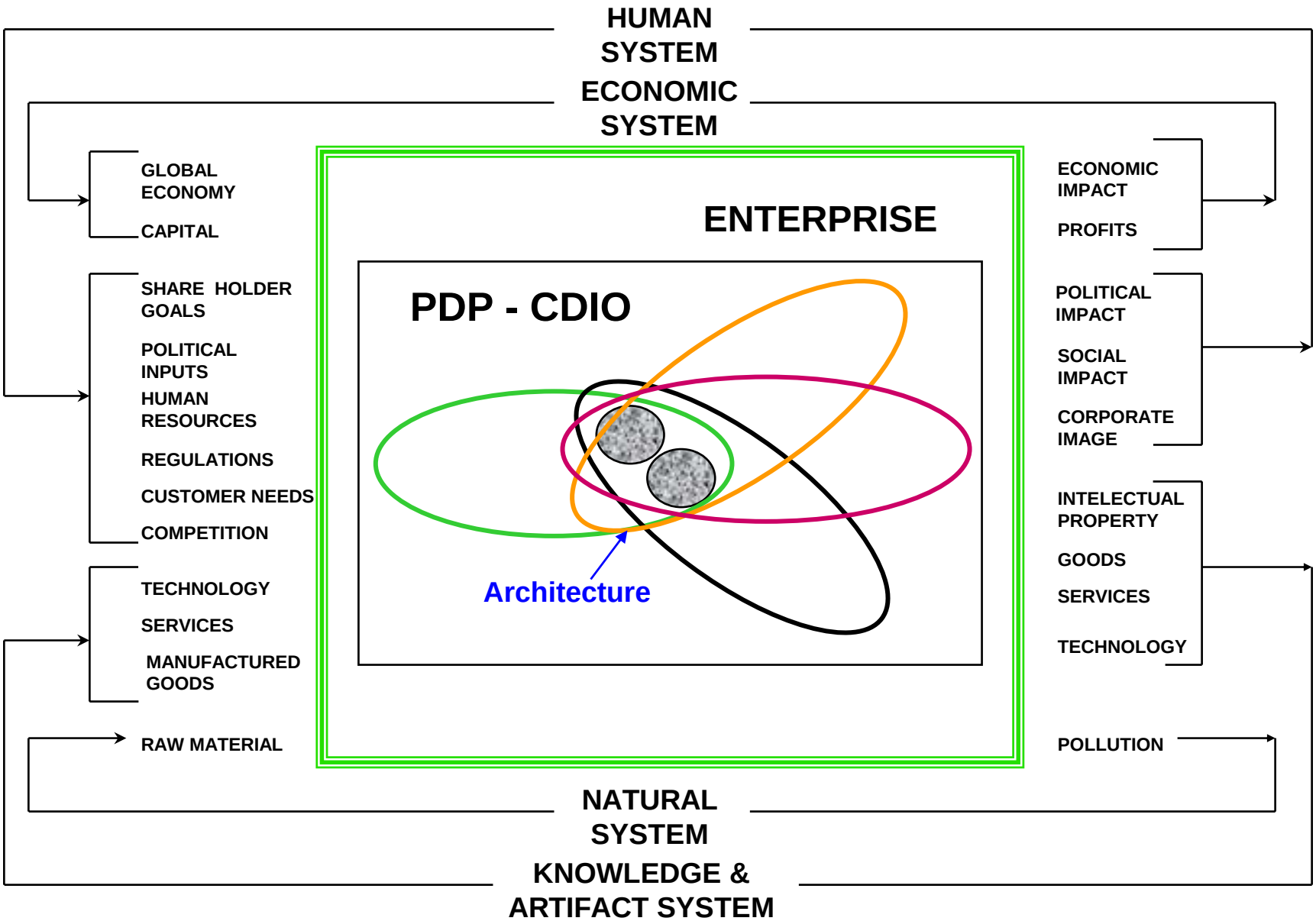
Enterprise Context

- **The PDP sits within an enterprise**
- **The enterprise brings many internal strengths (and weaknesses) to bear on the PDP**
- **The enterprise "helps" interface with external influences**
- **Knowledge of how the enterprise operates is vital to a successful architect**



Societal Context

- **Many (most) of the influences on and of the architecture also come and go across the enterprise boundary with society**
- **The elements beyond the enterprise are often critical to the architectural success**
- **These interactions are with four super-systems:economic, human, knowledge/artifact, and natural**



Holism and Focus

- **These and many things are in the holistic view of a good architect**
- **Of all of the things in the view, on any given day or week, one must FOCUS**
- **Focus on the 5-7 things that are the most important or urgent at any one time**
- **Reevaluate the focus list routinely**

The Role of the Architect

- **The role of the architect**
- **Home of the architect**
- **The deliverables of the architect**
- **The goal of the architect - Good Architecture**

The Role of the Architect

- **Defines the boundaries, goals, and functions**
- **Creates the Concept**
- **Allocates functionality and defines interfaces and abstractions**

The architect is not a generalist, but a specialist in simplifying complexity, resolving ambiguity and focusing creativity

Defines the Boundary and Functions

The architect defines the boundary of the "closed system" which constitutes the design of the system and its implementation process.

Specifically, the architect defines the goal(s) and function(s) and:

- **Interprets corporate strategy**
- **Interprets corporate marketing strategy and competitive analysis**
- **Listens to "customers" or their representative**
- **Infuses technology available**
  - **in platforms,**
  - **at the company, and**
  - **from other sources,**
- **Interprets regulatory and pre-regulatory influences**
- **Is sensitive to product liability and intellectual property issues.**

Creates the Concept

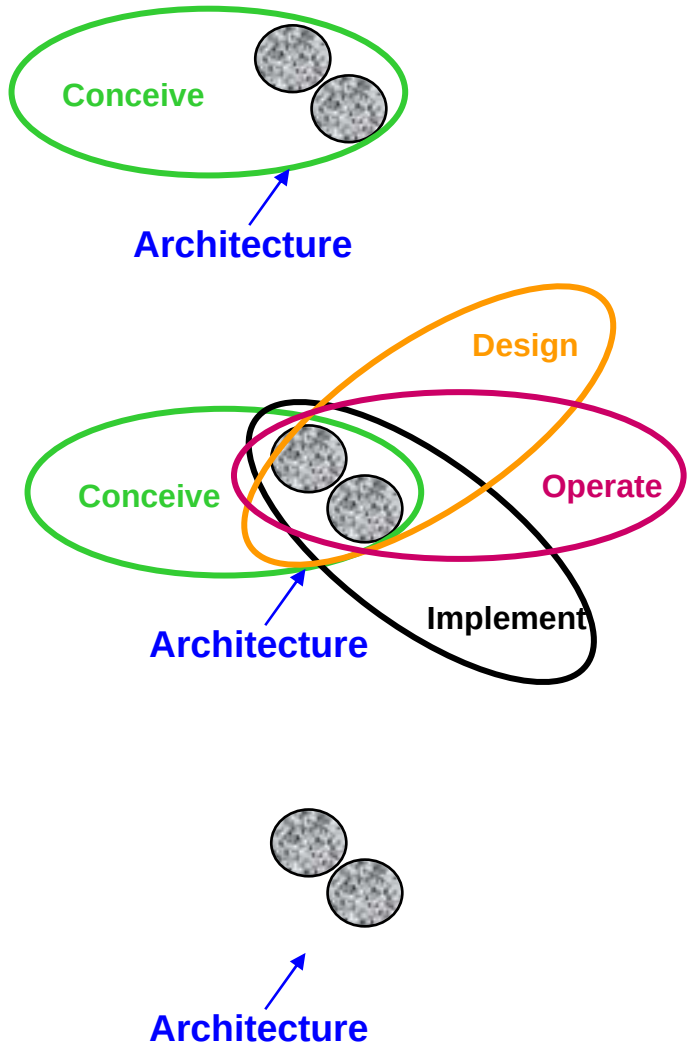
- Proposes and develops options
- Identifies key metrics and drivers
- Conducts highest level trades, and optimization
- Thinks holistically about the entire product life cycle in terms of
  - design
  - implementation (sourcing and manufacturing)
  - operation
  - product and process
  - risk management
  - sustainability
- Anticipates failure modes and plans for mitigation and recovery

Allocates Functionality and Defines Interfaces and Abstractions

- Decomposes form and function
- Allocates functionality to elements
- Defines interfaces between subsystems,
- Configures the subsystems - creates the structure of the system while considering:
  - Flexibility vs. optimality
  - Modularity vs. platform
  - Vertical vs. horizontal strategies, and
  - In-house vs. outsourcing design and manufacturing

Specific Role of the Architect

- **The reference case in the architecture of a new product or system. The architect deals with an external customer, and architects a product, mindful of its design, implementation and operations**
- **A higher level view is the architecture of the product, its platform, its design process, and its implementation process (of a maturing system)**
- **A lower level view is the architecting of a major element/module (which often allows architectural innovation), and often does not involve interacting with an external customer**

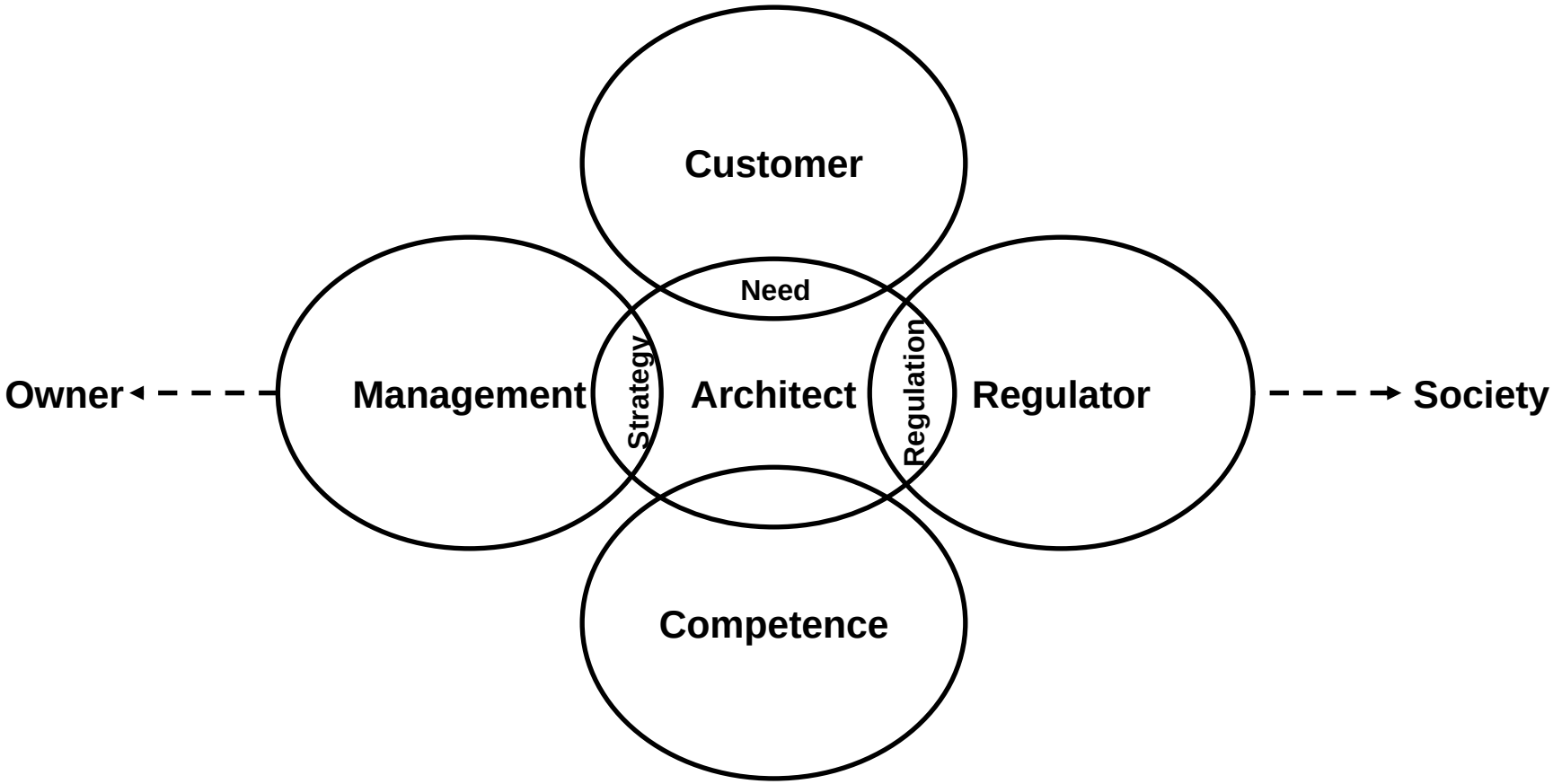


Architecting is Recursive

- **Takes place at the whole product system level (sometimes called “system of systems”)**
- **At the product or system level**
- **At the level of elements which are modules**
- **At the level of elements which are parts**

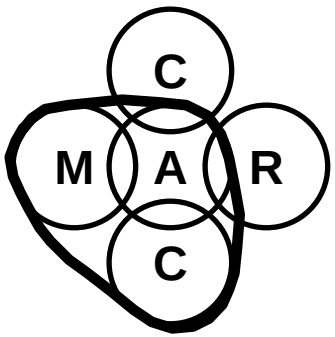
Home of the Architect

- **It is the responsibility of the architect to interact with the "manager", "customer", "regulator" and "competence" to get the system built.**

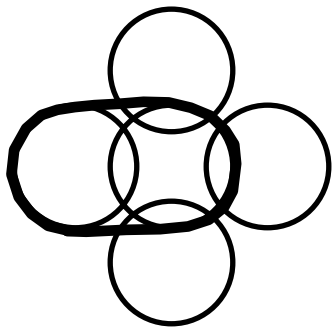


Organizational "Homes" of the Architect

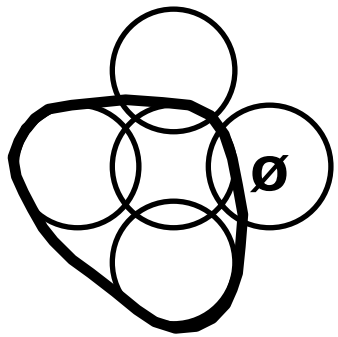
The Architect may be in one of several organizational forms



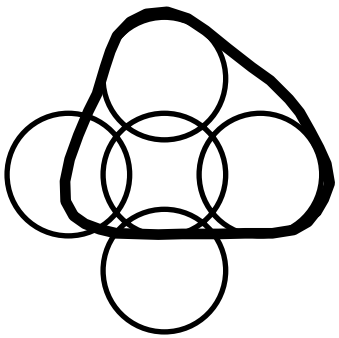
Traditional Co.



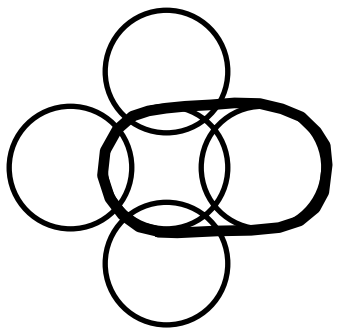
Virtual Co.
Civil Architect Firm



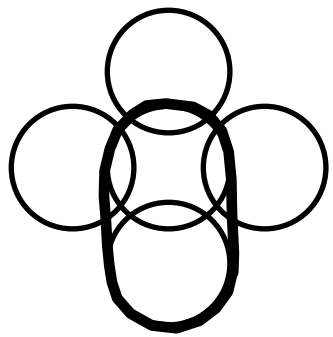
New Product Sector



Government Agency



Regulator/Arch.
(eg. Aerospace Co.)



Deliverables of the Architect

- **A clear, complete, consistent and attainable (with 80% - 90% confidence) set of goals (with emphasis on functional goals)**
- **A description of the broader usage context in which the system sits, and the whole product context**
- **A functional description of the system, with at least two layers of decomposition**
- **A concept for the system**
- **A design for the form of the system, with at least two layers of decomposition**
- **A notion of the timing, operator attributes, and the implementation, operation and evolution plans**
- **A document or process which ensures functional decomposition is followed, and the form and function at the interfaces are controlled**

The Architect Creates Good Architecture

- **Satisfies customer needs**
- **Can compete effectively in the marketplace**
- **Incorporates appropriate technology**
- **Meets strategic business goals**
- **Meets or exceeds present and future regulations**
  - **Is operable, maintainable, sustainable, reliable**
  - **Can be evolved/modified as appropriate**
  - **Can be designed and implemented by envisioned team**
  - **Can be implemented with existing/planned capabilities**

AND IS ELEGANT

Architecting is Universal

- **Political, Organizational, Technical and Artistic/Building systems are explicitly architected**
- **Social, Urban systems are implicitly architected**
- **Natural systems?**

Summary - Role of the Architect

- **The architect performs the most abstract, high level function in product development**
- **The architect has the greatest leverage on the eventual success of the product**
- **The architect must employ system thinking**
- **The architect**
  - **defines the boundaries and function**
  - **creates the concept**
  - **defines the elements, interfaces and abstractions**
- **The architect delivers the deliverables, and creates elegant and successful architecture**

Summary - Learning Objectives

Students will be able to:

Apply the principles, processes and tools of system architecture to:

- Structure and lead the early, conceptual phases of the product development process

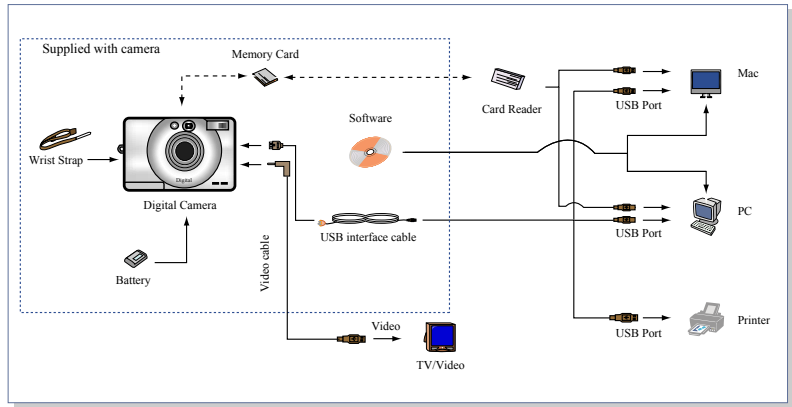
- Support the process through development, deployment, operation and evolution

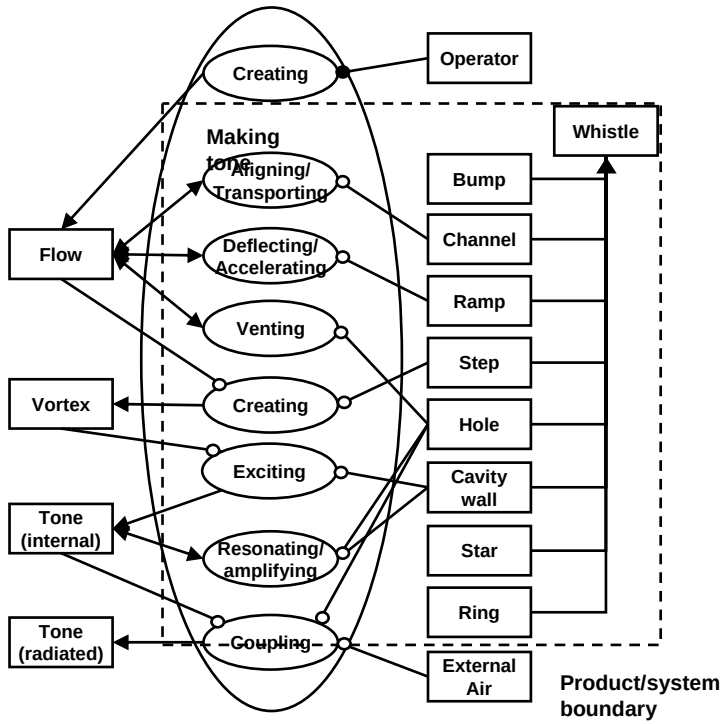
Learning Objectives

- **Discuss systems, systems thinking, products (value and competitive advantage), the PDP and the "role" of architecting in the PDP.**

- **Analyze and critique the architecture of existing systems, create the architecture of new or improved systems, and produce the deliverables of the architect.**

Figure by MIT OCW.





Learning Objectives

- **Drive the ambiguity from the upstream process by defining the context and boundaries of the system, interpreting needs, setting goals and defining the externally delivered functions.**

- **Create the concept for the system, consisting of internal function and form, while thinking "holistically and out of the box" when necessary.**

- **Manage the evolution of complexity in the system so that goals are met and function is delivered, while the system is comprehendible to all during its design, implementation, operation and evolution.**

Learning Objectives

- **Challenge and critically evaluate current modes of architecting, and create new synthesized modes.**
- **Develop for themselves the guiding principles for successful architecting.**

To prepare students for their first, second, and third jobs after SDM.

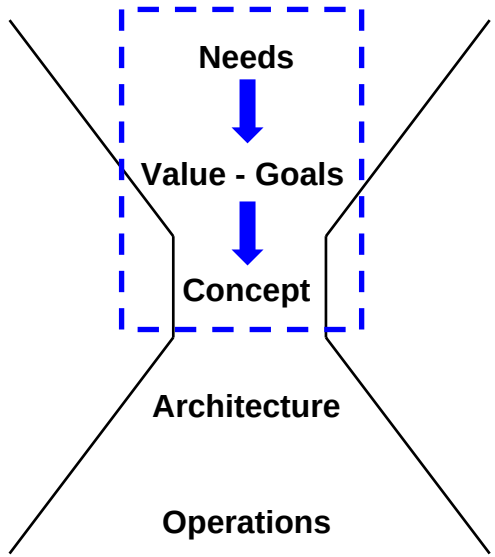
This is a course in how to think, not what to think

A Guide to Guiding Others

- On the next four charts are a set of questions that could be used to help others you work with understand a process about architecture
- Note they are not far from the questions that were on OS 4, which led to the deliverables of an architect
- But they try not to rely on the details and formalism we have discussed, but the main ideas
- Try them and see how they work
- I would appreciate feedback

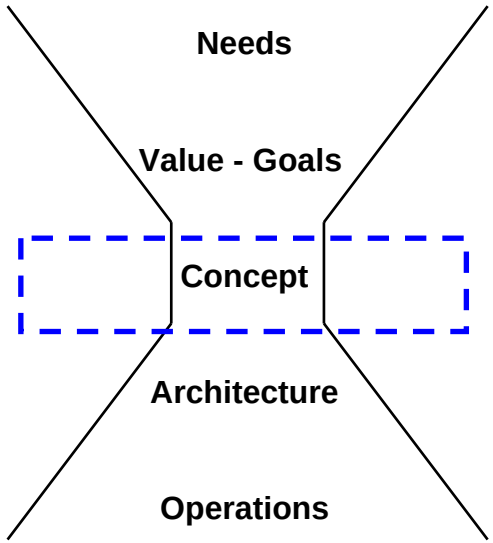
Framework - Needs to Value to Concept

- Who are the beneficiaries? What are their needs?
- What is the value related operand and its states?
- What is a solution neutral statement of the value related transformation - the externally delivered value related function? (solution neutral function)
- What are the solution specific functions which will achieve this transformation? (the function part of concept)
- What are the solution specific abstractions of form that can execute this process?(the form part of concept)
- What are potential multi-functional aspects of the concept process?
- How does value trace to the beneficiaries?



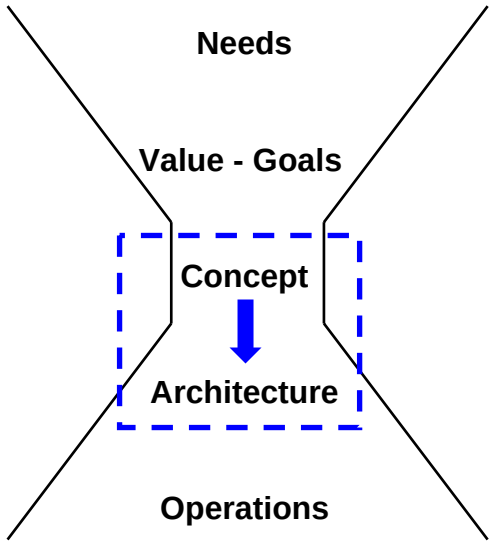
Framework - Concept In Context

- What is the product system?
- What are the supporting systems?
- What is the whole product system?
- What is the use context?
- What are the boundaries?
- What are the interfaces? What are the operands that are passed or shared? Interface process? Interface instrument objects?



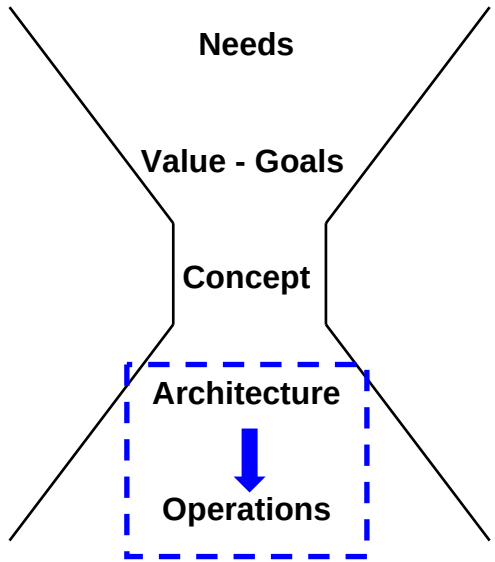
Framework - Concept to Architecture

- What are the principal internal functions? Operands along the way?
- How is the form decomposed into elements? (decomposition of form)
- What is the structure of the elements? How are the internal functions mapped to elements of form?
- How do these combine to produce the emergent externally delivered value related function?
- What other value related external functions are delivered?
- Are there supporting processes and objects?



Framework - Architecture to Operations

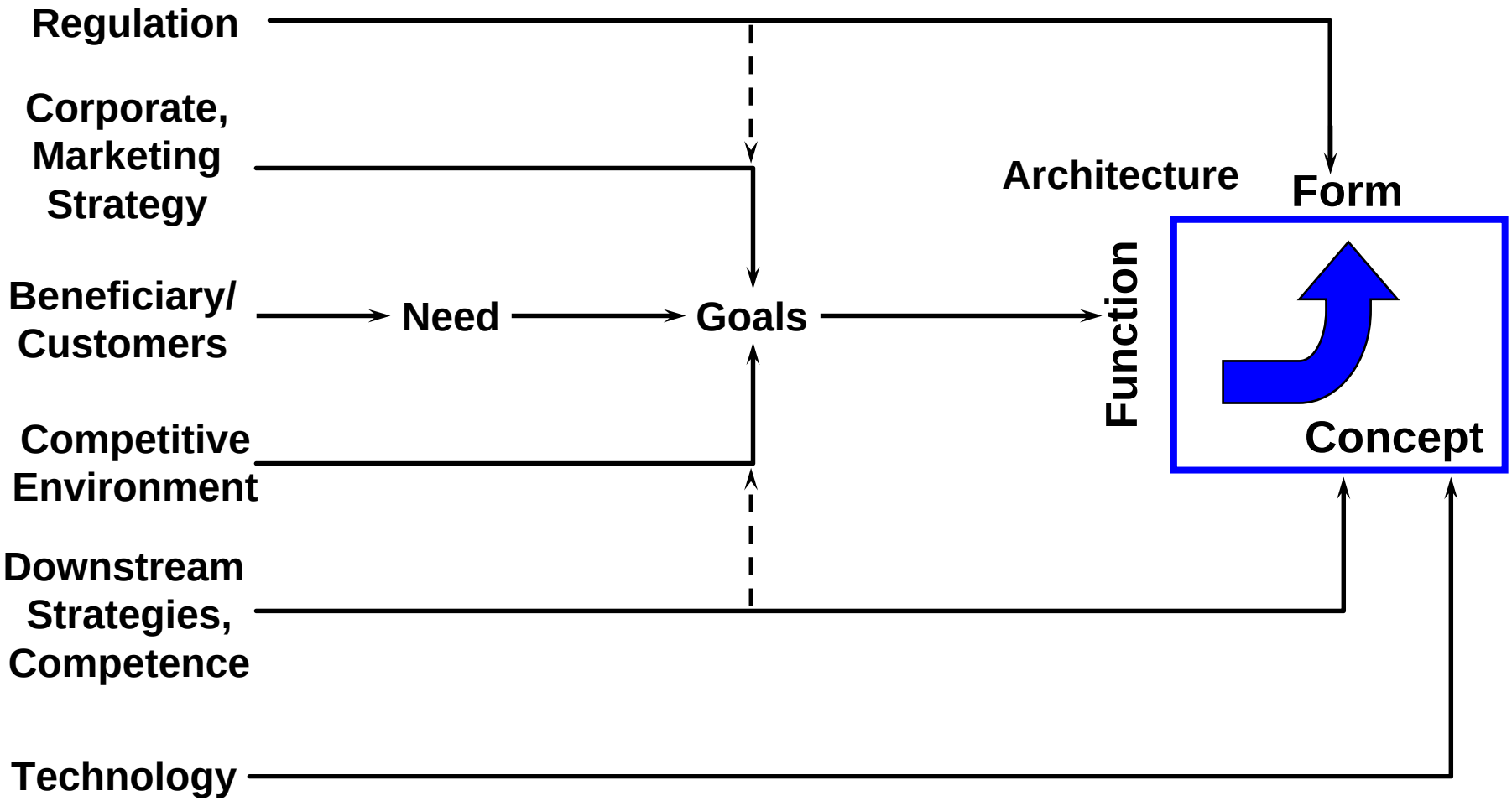
- What is the sequence in the process of delivering primary function?
- Are their contingency, emergency or stand alone processes?
- Are there commissioning, decommissioning and maintaining processes?
- Is clock time important to understand in the operations?



Architecture of SDM

- **The SDM program was developed using the technique of system development**
- **About a 3 year process from the “moral” commitment to the beginning of the actual program**
- **Review this History in order to**
  - **Give an example of architecting a complex system**
  - **Allow you to understand the context of your experience**

Dominant Upstream Influence on Architecture



MIT's "Corporate" Strategy

- **Leadership through developing and delivering the best education polarized around science and technology**
- **True interdepartmental/interschool efforts**
- **Research results transferred quickly to education**
- **Close relationship to industry**
- **Innovation in graduate programs which migrates to undergraduates**

Market Strategy

- **Develop a “first-in”, advanced professional degree for engineering leadership (the “MBA for Engineers”)**
- **Focus program on large producers of complex electro-mechanical, information systems**
  (auto, aero, comm, computer)
- **Make available to smaller companies through individual participation**
- **Slowly encourage government, regulators, etc. to participate**
- **Eventually broaden to other industrial sectors**

Market Characteristics

- **Segmentation**
  - **on campus, 1 year**
  - **on campus, RA**
  - **on-off campus**
- **Competition**
- **Size**
- **Price Point**

Regulation

- Rules and Regulations of the MIT Faculty
- 6 subjects = MS
- 1 semester of residency
- No previous distance education program for credit

Technology

- Picturetel is not sufficient to replace all personal contact

Customer Need

- **Research and Scholarship to codify and improve best practice in product development**
- **Educational programs for key people**
- **Corporate Impact**

derived from –
~ 30 interviews/presentations
~ 5 focus groups
prototype year (95-96)

Goals

To educate future technical leaders in system engineering/architecture, and the conception and design of complex products and systems. Students completing the program become prepared for careers as the technically-grounded senior managers of their enterprises

Plus: internally driven goals

- **jumpstart/leapfrog distance education**
- **bring about closer relations with industry**

Functions

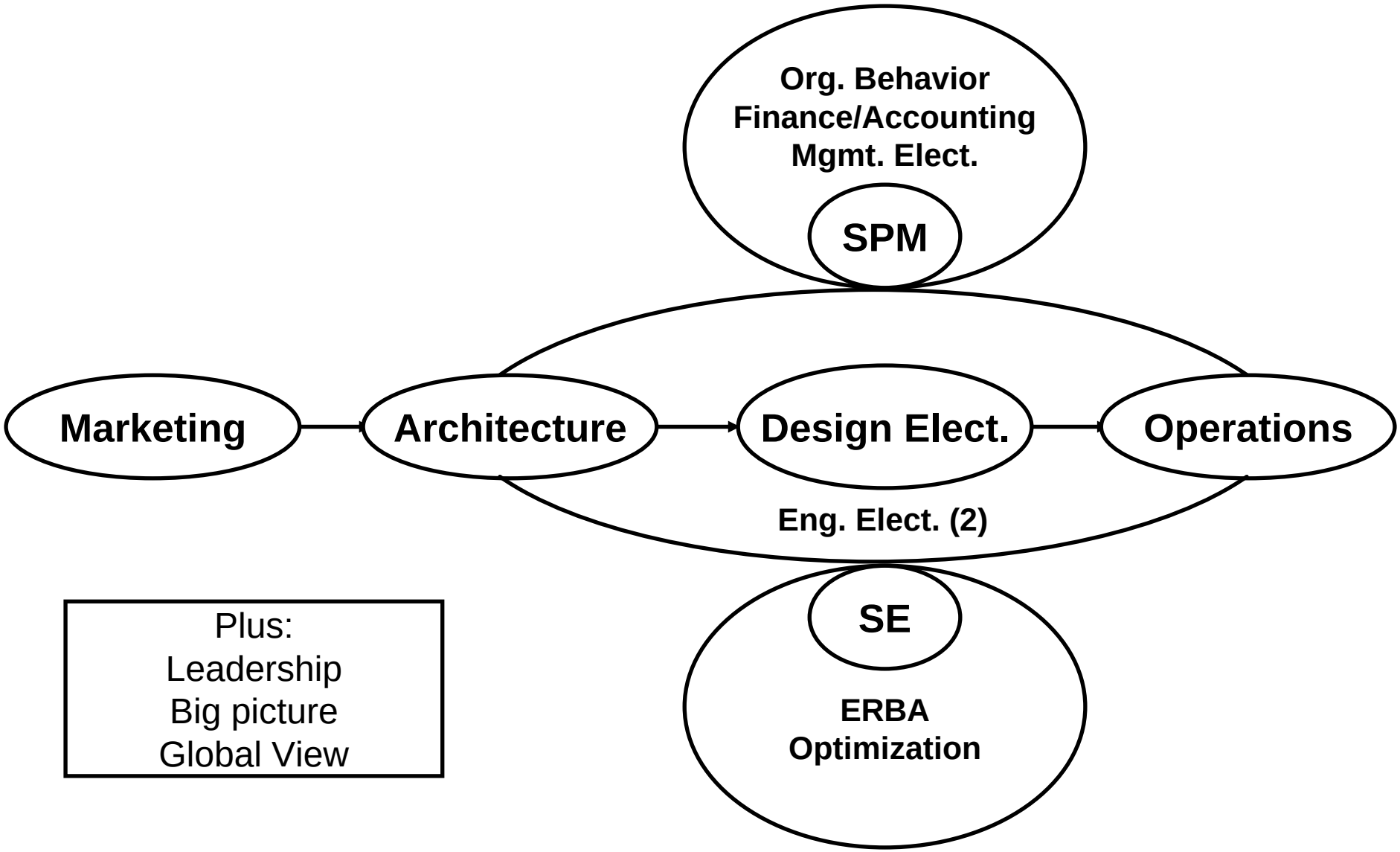
To increase the knowledge, skills, and attitudes of the students in:

System Thinking	System Engineering System Integration Risk Benefit Analysis System Optimization Modeling and Simulation	System Architecture System Design Product Development Technical Planning
Management Skills	Life Cycle Planning Operations Program Management Program Control	Marketing Strategy Accounting/Finance Law Technology Assessment
Leadership Skills	Technical Teams Organizations Leadership	
Technical Skills	Disciplinary Design Disciplinary Expertise	
Big Picture Thinking	Big Picture Alternative Thinking Global Perspective	

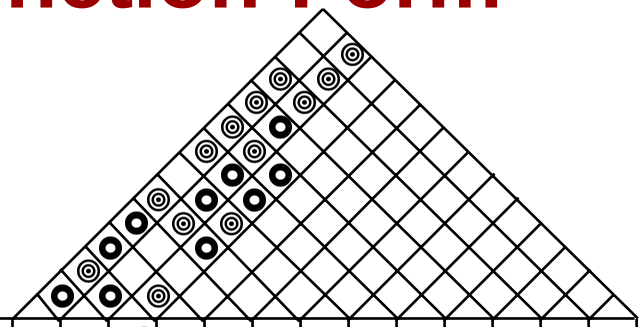
Concept

- **~ 13 subject platform program**
- **On and On-off campus variants**
- **Partnership with industry to develop content**
- **New distance education technology to replicate campus experience at a distance**

Form of Program



Matrix of Function-Form

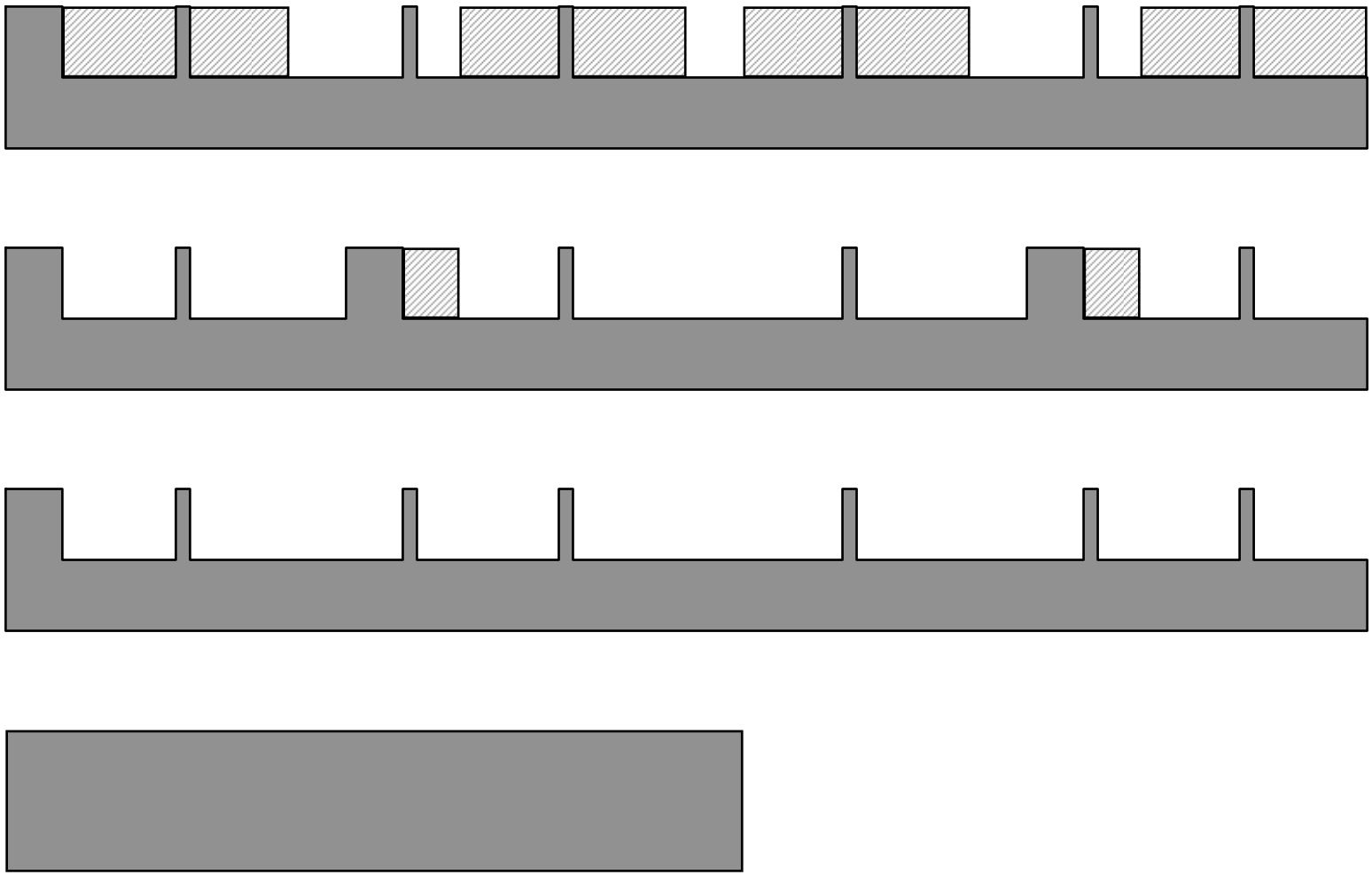


Matrix Key
- ◎ Strong
- ◉ Medium
- ○ Weak

Roof Key
- ◎ Strong Positive
- ◉ Positive

		IMPORTANCE	Sys. Eng.	Sys. Arch.	Sys. Managem.	ERBA	Sys. Opt.	Operations	Marketing	Acct/Finance	Organizations	Design Elect.	Eng. Elect. 1	Eng. Elect. 2	Mgmt. Elect.
System Thinking	Systems Eng.	5	◎												
	Sys. Integration	5	◎												
	System Opt.	5	◉				◎								
	Risk-Benefit	5	◉			◎									
	Modelling/Sim	5	○				◉					◎			
	System Arch.	5		◎											
	System Design	5		◉								◎			
	Prod. Develop	5		◉								○			
	Tech Planning	5		◎											
Management Skills	Prog. Management	5			◎										
	Prog. Control	5			◎										
	L.C. Planning	5			◎										
	Operations	5			◉			◎							
	Marketing	5							◎						
	Acct/Finance	5								◎					
	Strategy	2							○						◎
	Law	2		◉											◎
	Tech. Assess			◉											◎
Leadership Skills	Tech. Teams	5	○		◉						○				
	Organizations	5									◎				
	Leadership	3													
Technical Skills	Disc. Design	5										◎			
	Disc. Expertise	3											◎	◎	
Big Picture Thinking	Big Picture	5	◉	◉	◉										
	Alt. Thinking			○											
	Global Perspective	3		○											

Timing: Schedule Variants



Summary of Assignments
Until we meet again

- **Think about the architecture of your products. Is it good? What *is* good architecture?**
- **Read about ambiguity, complexity and creativity. How is it handled by architects in your enterprise?**
- **How are product goals set in your enterprise? How are upstream influences identified and incorporated?**
- **How are the downstream influences factored into the architecting process?**
- **Get to know an architect - try to develop a mentor/mentee relationship**
- **Search for and record architecting principles in life, work, school**
- **Be increasingly Globally Aware!**