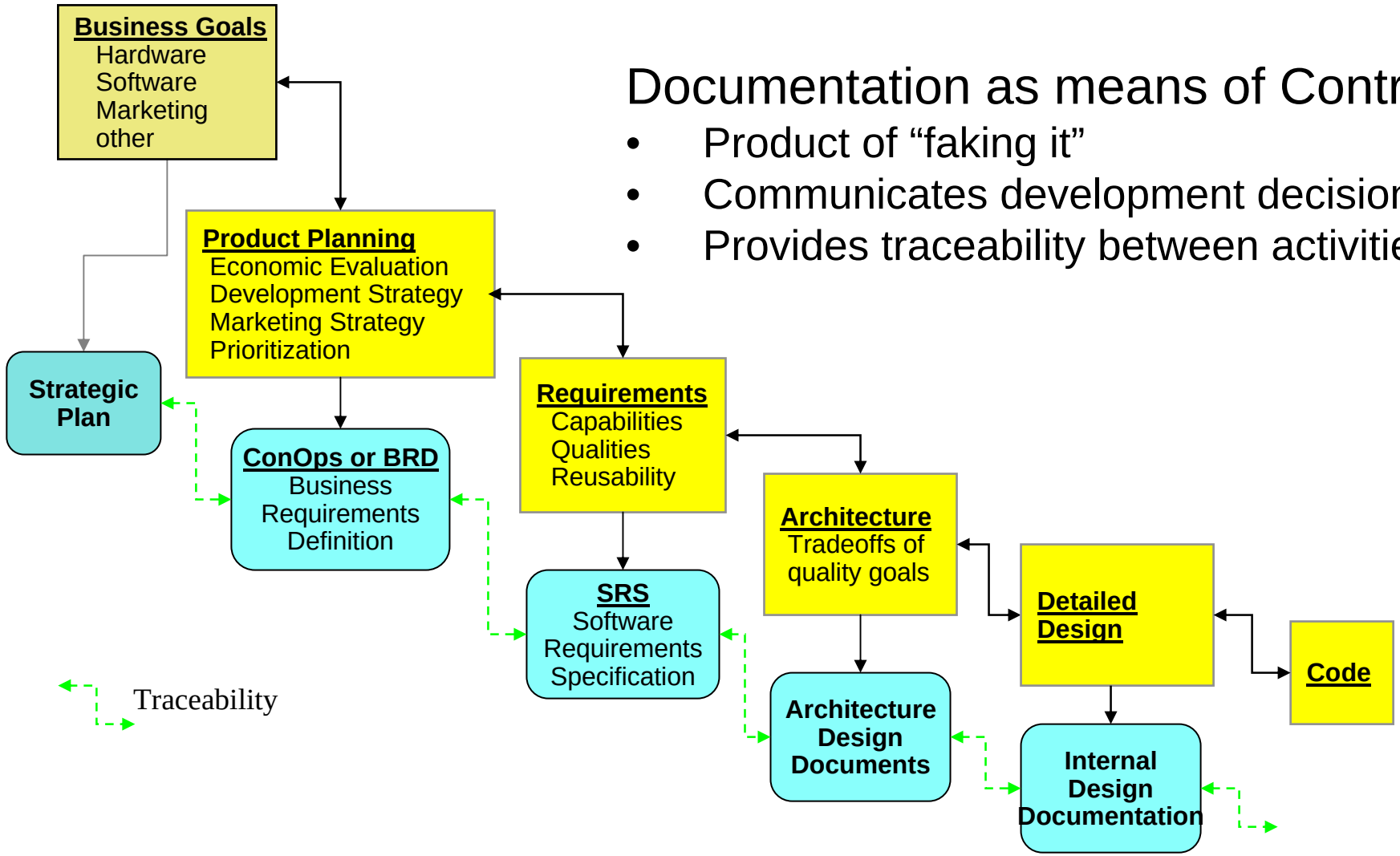
CIS 422/522

Product Specification

Product Development Cycle



Documentation as means of Control

- Product of "faking it"
- Communicates development decisions
- Provides traceability between activities

Phases and Products

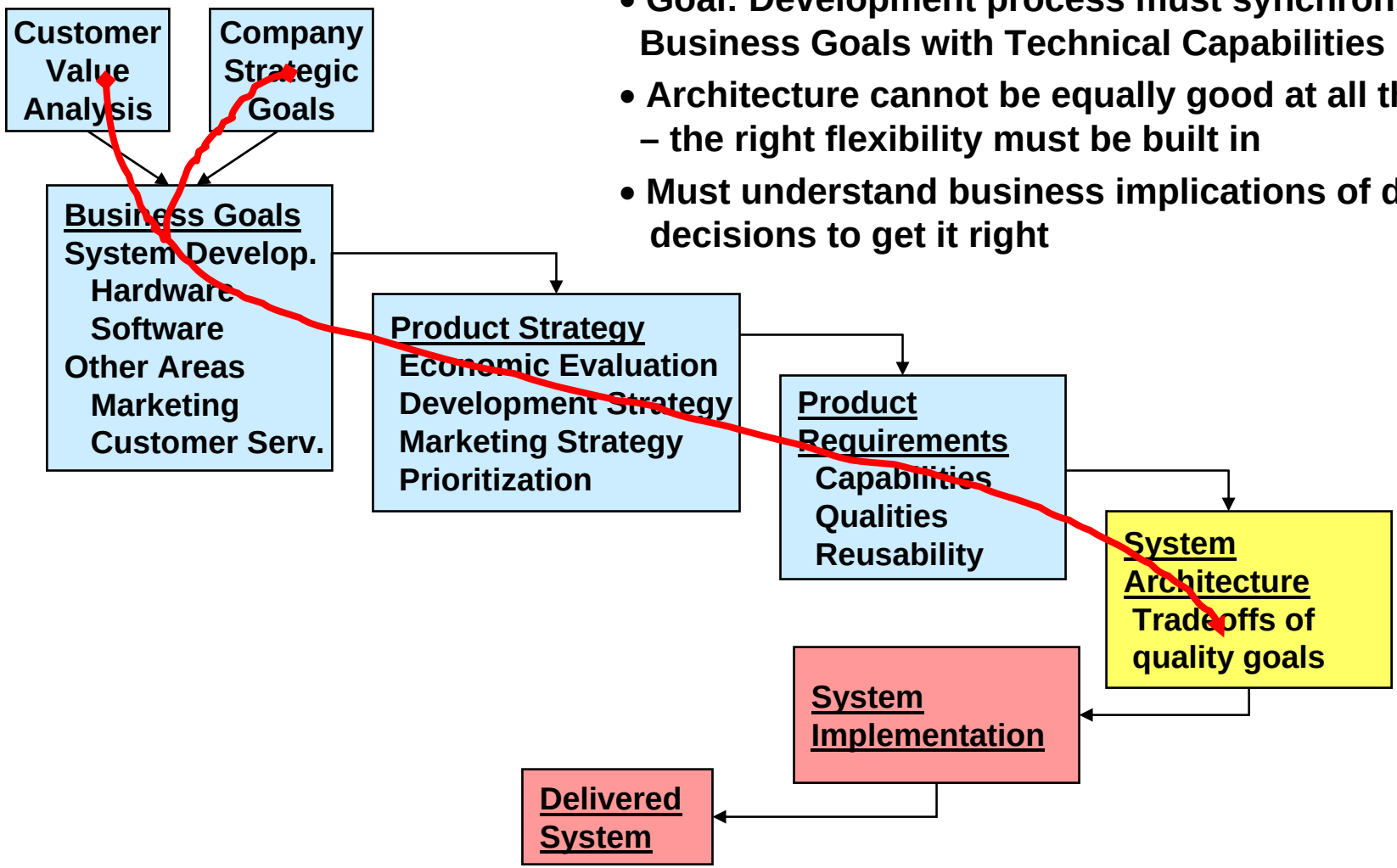
- Product Planning
  - Goal: Link organizational goals (why) with product features (what)
  - Product: Product spec. (ConOps, BRD, MRD, etc.)
- Requirements
  - Goal: technical specification of precisely what the software must do and any development constraints
  - Product: Software Requirements Specification (SRS)
- Architecture
  - Goal: decomposition of the problem into components that together satisfy the requirements and *quality goals* within the constraints
  - Products: specification of components, relations, interfaces (class structure, calls structure, task structure, etc.)
- Detail Design
  - Goal: internal design of components (e.g., objects) to identify appropriate algorithms and data structures supporting the interface
  - Products: design documentation, pseudo-code

Types of Information Needed

- Product Spec should link what will be developed with why we are developing it
  - Rationale: “What are the overall objectives and rationale (Purpose, Business case, Mission) for creating/changing this system?”
  - Solution Requirements: “What characteristics should the system have and capabilities should it provide to address the objectives and rationale?”

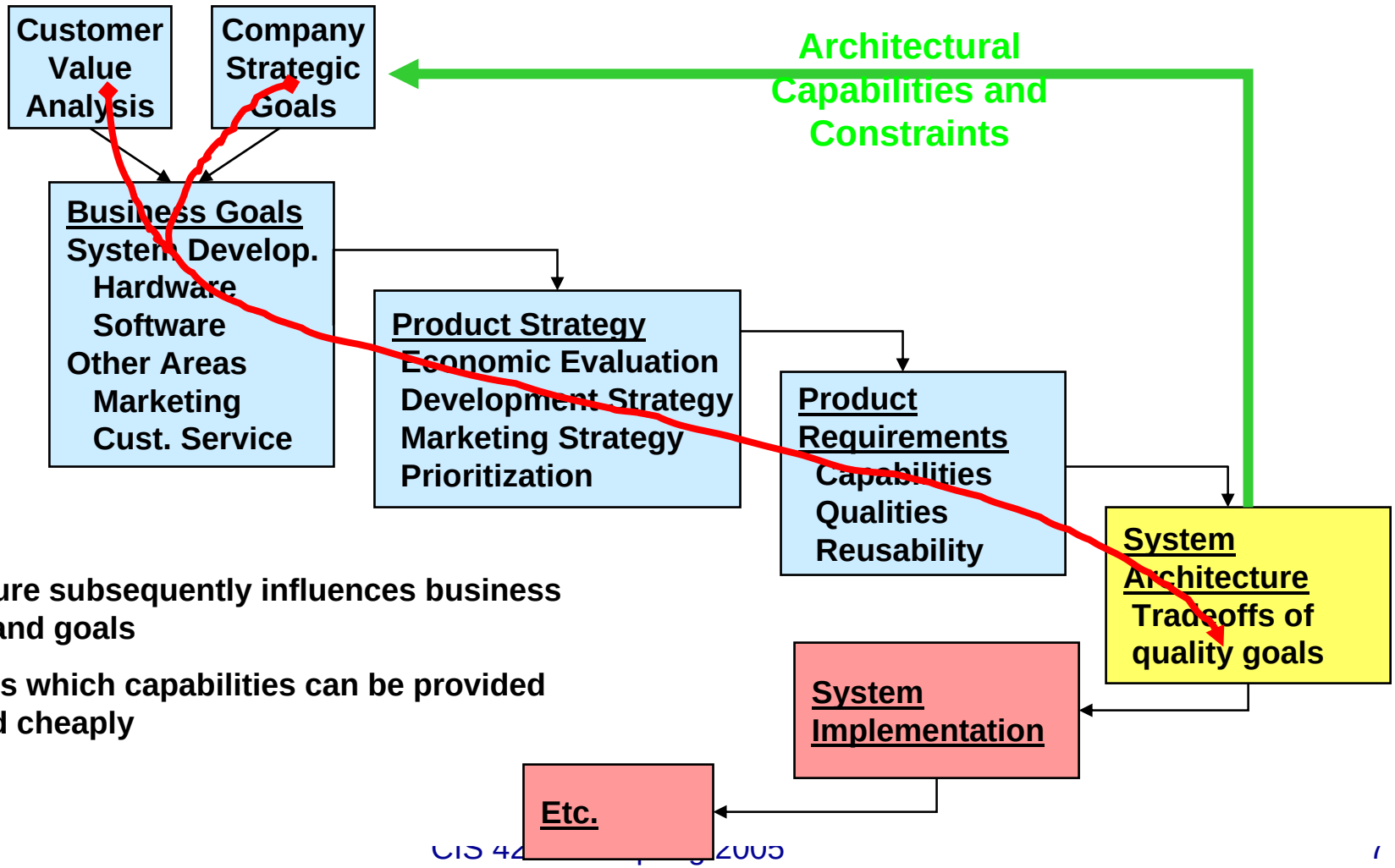
The ConOps and Related Docs

Product Development Cycle



- **Goal: Development process must synchronize Business Goals with Technical Capabilities**
- **Architecture cannot be equally good at all things – the right flexibility must be built in**
- **Must understand business implications of design decisions to get it right**

Closing the Business Cycle



- Architecture subsequently influences business decisions and goals
- Determines which capabilities can be provided quickly and cheaply

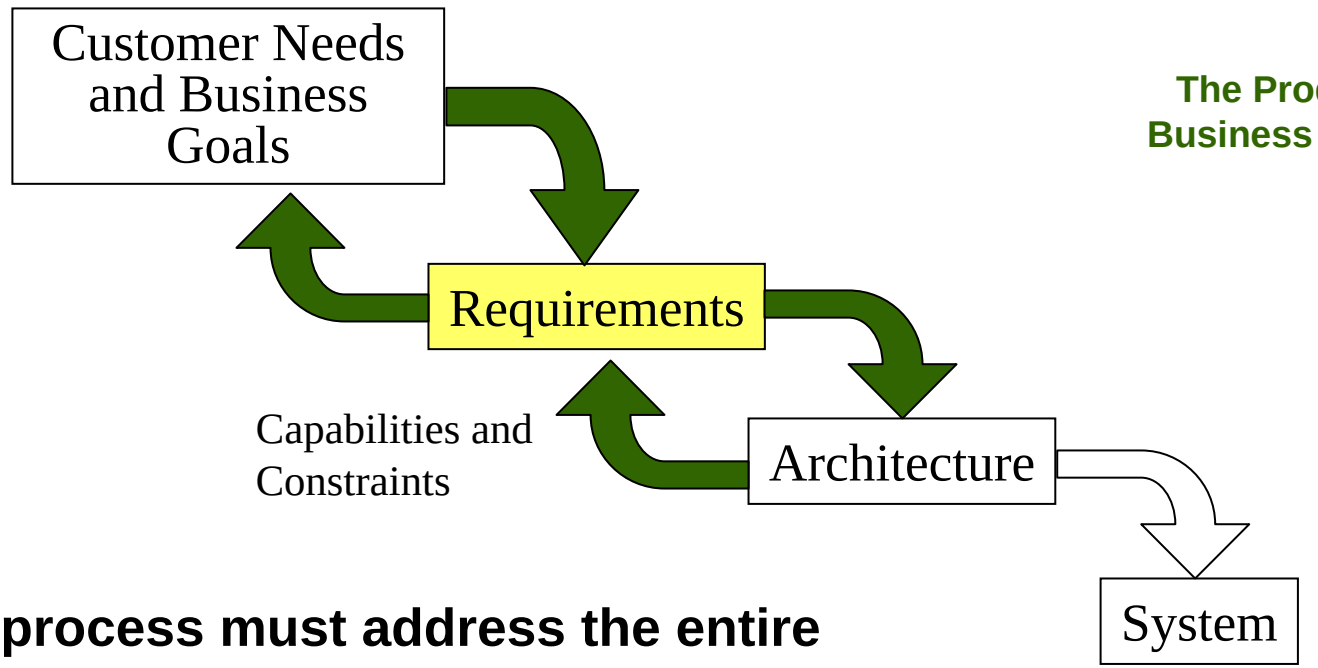
Implications

- Making sound business decisions requires understanding the software engineering implications of those decisions
  - How will adding feature F affect my ability to change the system to add feature G next?
- Making sound software engineering decisions requires understanding the business implications of those decisions.
  - Will design A accommodate desired new features more easily than design B?
- There is a fundamental need for effective two-way communication between business and engineering organizations in a product company
- Otherwise, what happens over time?

Effects of Disconnects

- Result is often a mismatch between *technical capabilities* and *business goals* that cannot easily be corrected, e.g.
  - "Simple" changes are difficult
  - System must be redesigned to create similar product
- Symptoms:
  - "We had the best product by the time it shipped, we'd missed the market window."
  - "Our customers wanted us to add a couple of features but we couldn't do it without rewriting much of the code."
  - "We planned to develop a whole line of products based on the original design but found we had to start over again."

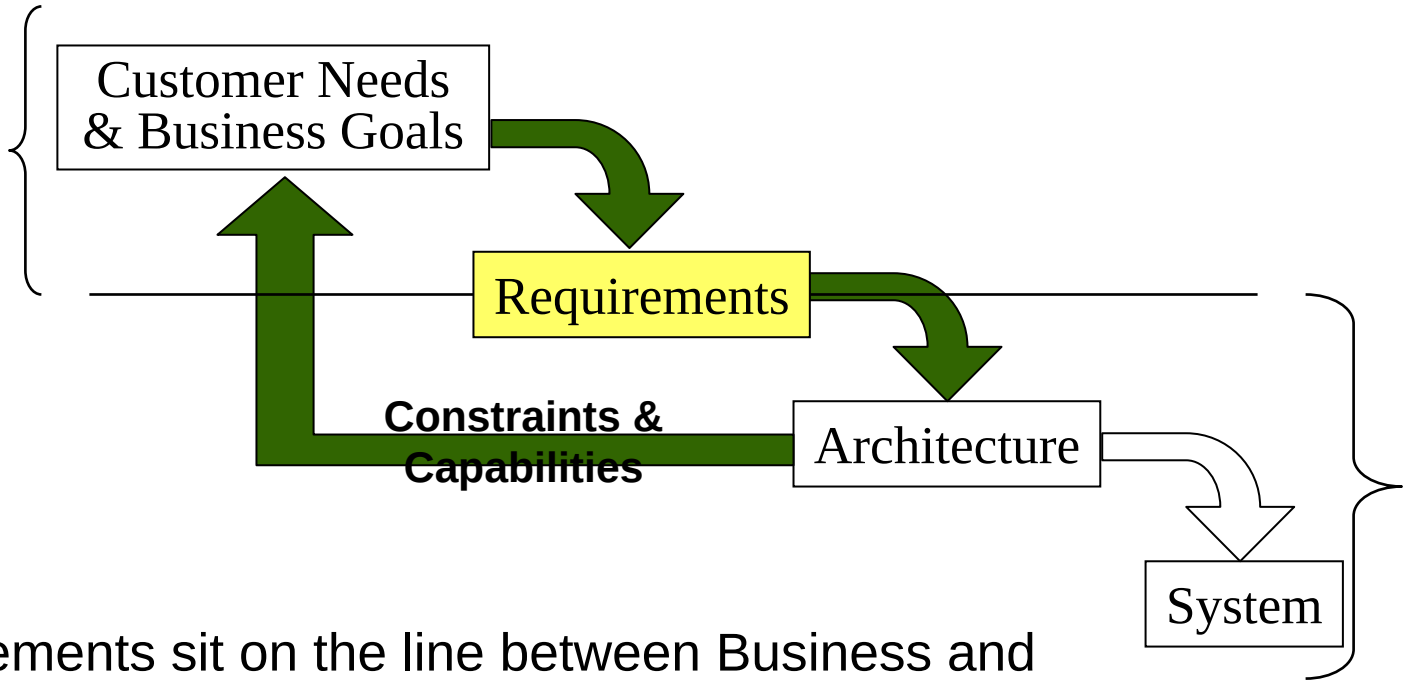
Role of Requirements



The Product Business Cycle

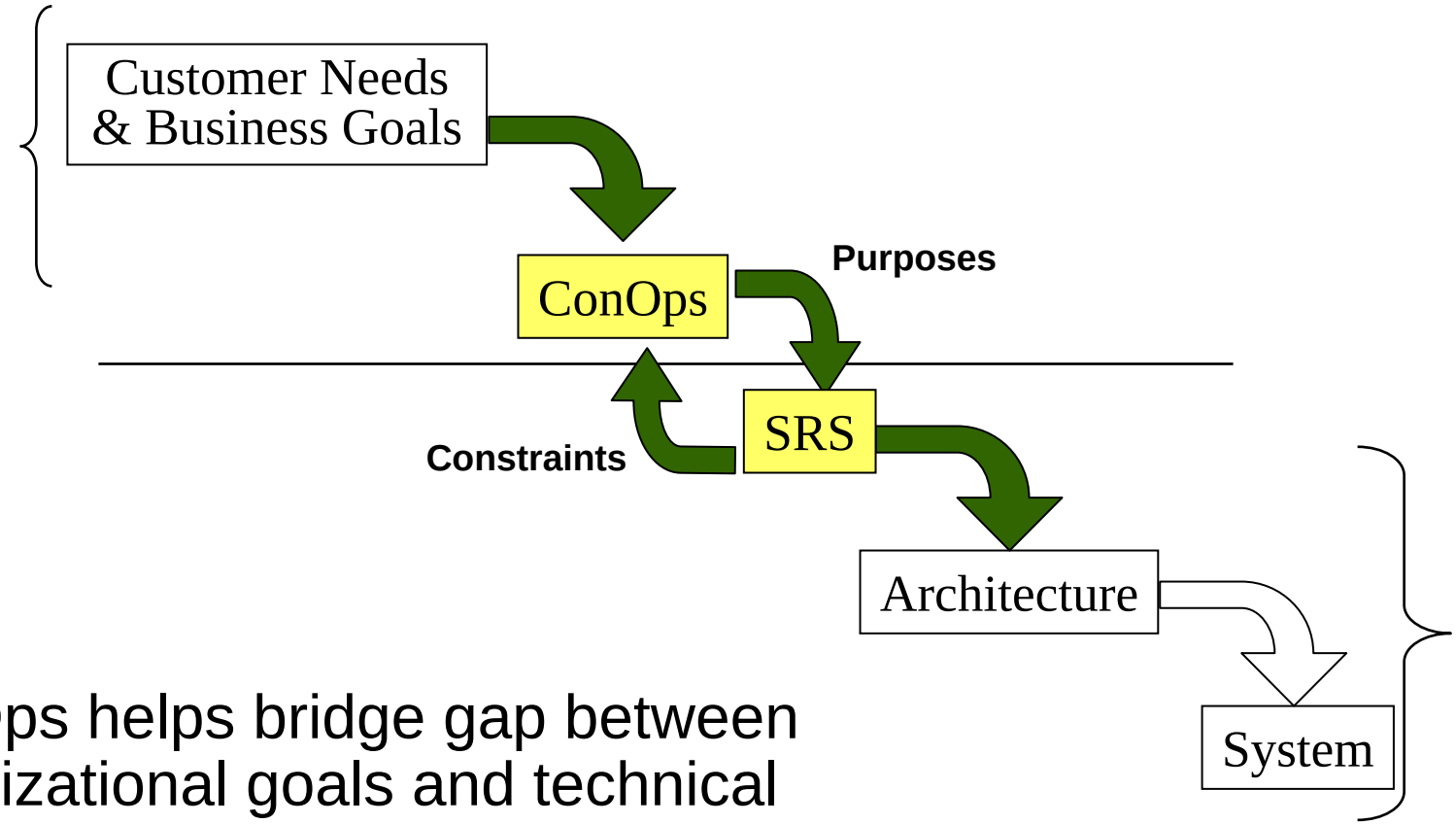
- **An effective process must address the entire Product Business Cycle**
  - Communicate business/customer needs down
  - Communicate technical constraints up
- **Requirements are key**

A Split Personality …



- Requirements sit on the line between Business and Technical sides of the house – two audiences
- May have two specifications, one owned by the business side, other by development
- Goal is a clear allocation of purpose, content, ownership

A Split Personality …



- ConOps helps bridge gap between organizational goals and technical requirements
  - ConOps belongs to business side
  - SRS belongs to development

Product Requirements

- Input: Customer needs, business goals, strategic goals
- Goal: Capture user's view of product
- Product: Business Requirement Document
  - Equivalently: Concept of Operations, Market Requirements Document, Mission Requirements, etc.
  - Distinguished by capturing the Development Context and User's View of product

ConOps Contents

- Current system or situation
  - What is the anticipated context for the system?
- Needs motivating new or modified system
  - What are the current desires or needs not being met in the problem context?
- Concepts for the proposed system
  - What are the proposed system capabilities and constraints?
- Operation modes
  - What are the anticipated modes of operation (e.g., test, maintenance, operation, etc.)
- User classes and characteristics
  - Who are the anticipated classes of users, what do we assume about them, and what will they use it for?

ConOps Contents

- Desired features with priorities (by mode and user class)
- Operational scenarios for each user class and mode
  - What are examples of the expected use of the system?
- Analysis of system concepts
  - What are the anticipated advantages, disadvantages or limitations, and tradeoffs?
- Summary of impacts
  - What are the anticipated impacts on the organization or its operation?

ConOps Characteristics

- Focus on capture and communication of user needs (desires)
- Typically a prose description
- Organized to “tell a story”

Review

- Supports separation of concerns
- Product Planning
  - Goal: Link organizational (e.g., business) goals (why) with product features (what)
  - Product: Product specification (ConOps, BRD, etc.)
  - Written in customer's language
  - Owned and produced by business side of the house
- Requirements
  - Goal: technical specification of what the software must do and any constraints on its development
  - Product: Software Requirements Specification (SRS)
  - Written in developer's language
  - Owned by technical side of the house

Assignment

- Reading Chapter 9
- Tutorial