The Software Development Process

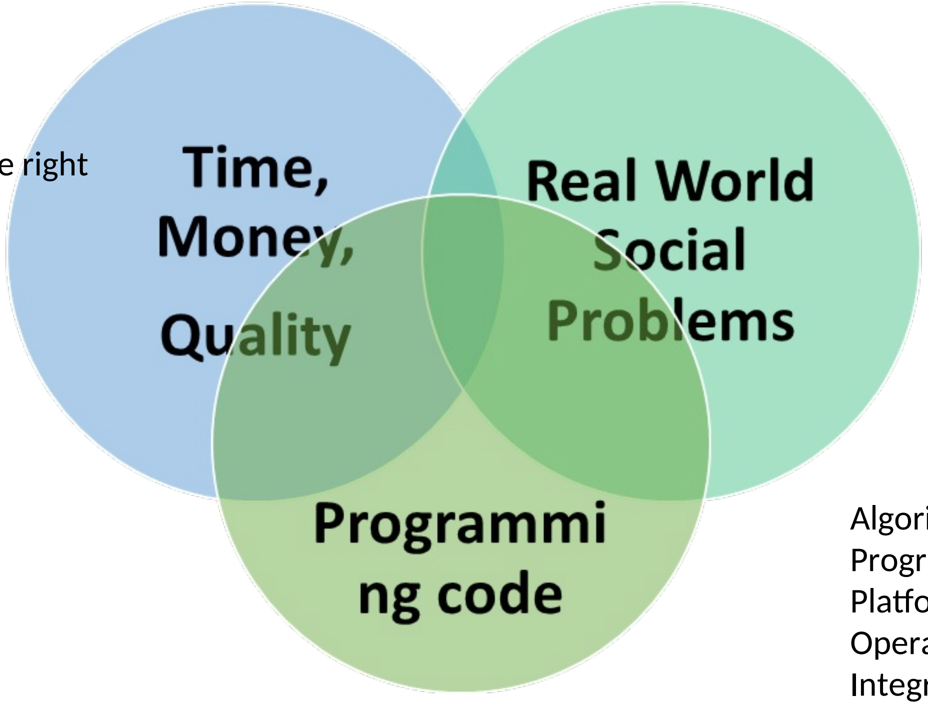
Kal Bugrara, Ph.D

Agenda

- Introduce the software development life-cycle (SDLC)
- Understand the SDLC roles and responsibilities
- Understand the different roles we have in the SDLC
- Understand why business requirements are important

Software Engineering
What happens when we build applications?

Can the customer afford the right Solution?
Where would one begin
Can I start small.
Can I deliver in small steps
What if we run out of time
Do we have enough people

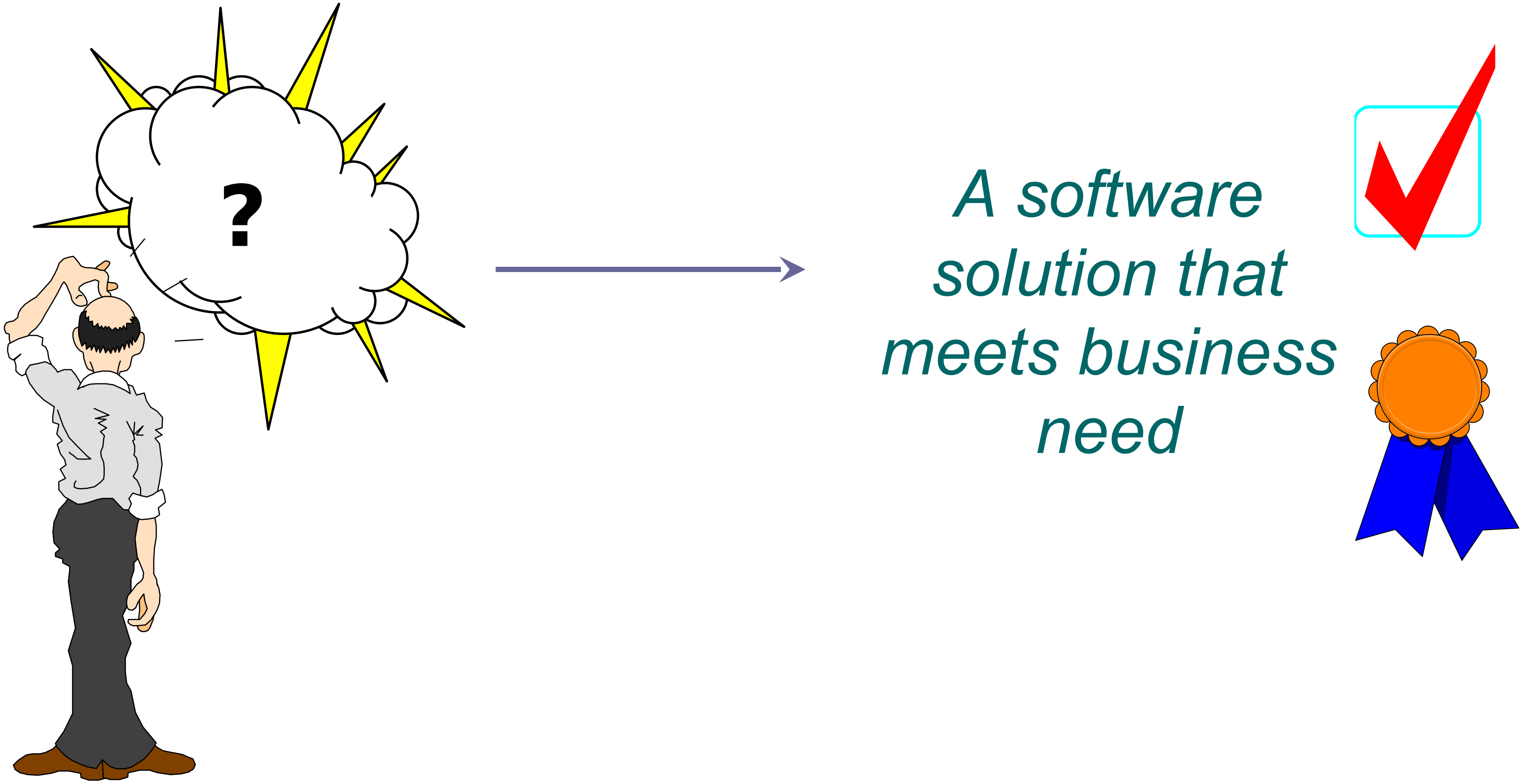


Why
What for
Who would use it
For whom
Scope: Social, group or individual

Algorithmic techniques
Programming Language
Platforms
Operations
Integration
Volume

Engineering and Design Matters

The SDLC objective



A software solution that meets business need

The SDLC Challenge

- **Build the right software solution**
- **Build it right**
- **Build it economically**

Meeting these challenges is a balancing act.

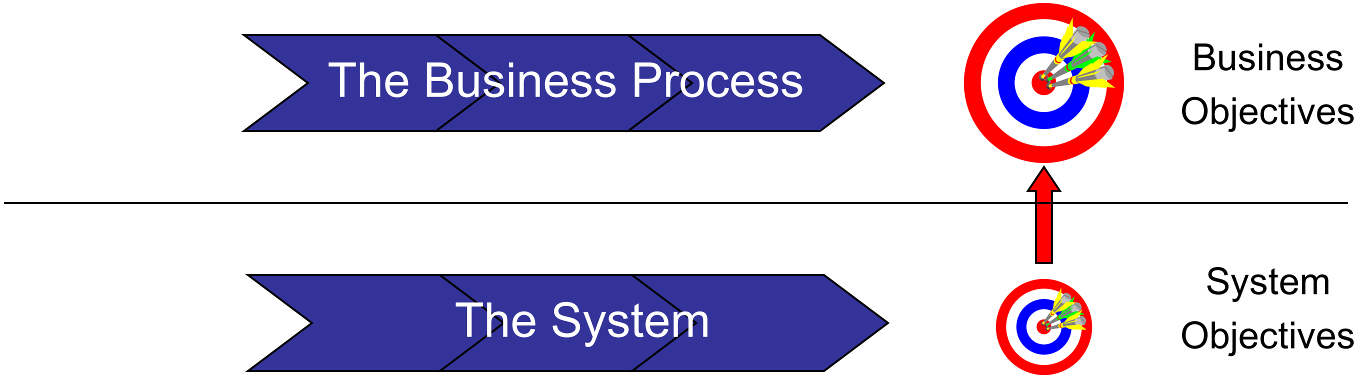
Two kinds of qualities

- **Business quality: a software solution that create business/social value**
  - E.g., The system helped the business realize an 70% reduction in customer complaints
  - The nursing tablet application reduces clinical errors
  - The nursing app eliminates the need to fill manual forms so a nurse would spend more time with the patient
  - The scanner on the “healthy diet” phone app allows you to fine-read the ingredient of a food package and alert you if toxic context is present (soy oil, or glutin, for example).
  - A web site allows you find last minute inexpensive tickets
  - ChatGPT helps you communicate effectively with business partners and friends
  - Design means coding can be an enjoyable experience for developers

Two kinds of qualities (contd.)

- **System Quality: the system meets the need for**
  - Improved response time
  - Scalability. E.g. can add thousands of new users at a reasonable cost.
  - Design means ease of change to the software overtime

Must link system objectives to business objectives

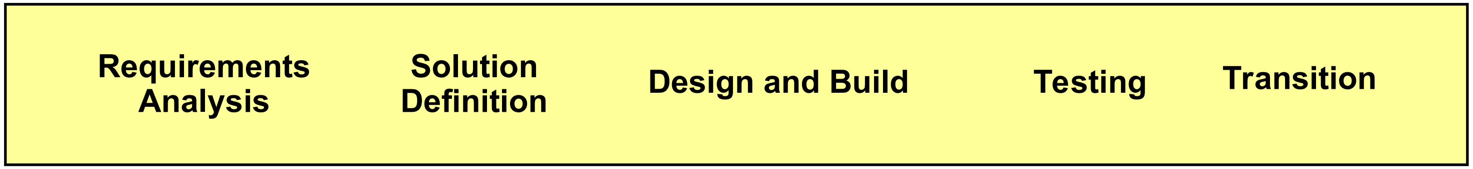


Business objective: Improve price performance of our products (want to make more profit)

System objective: Develop customer self-service app with simple user interface, and fast response time. The system must support simultaneous access to hundreds of thousands of user.

The SDLC Phases

The steps



Requirements Analysis

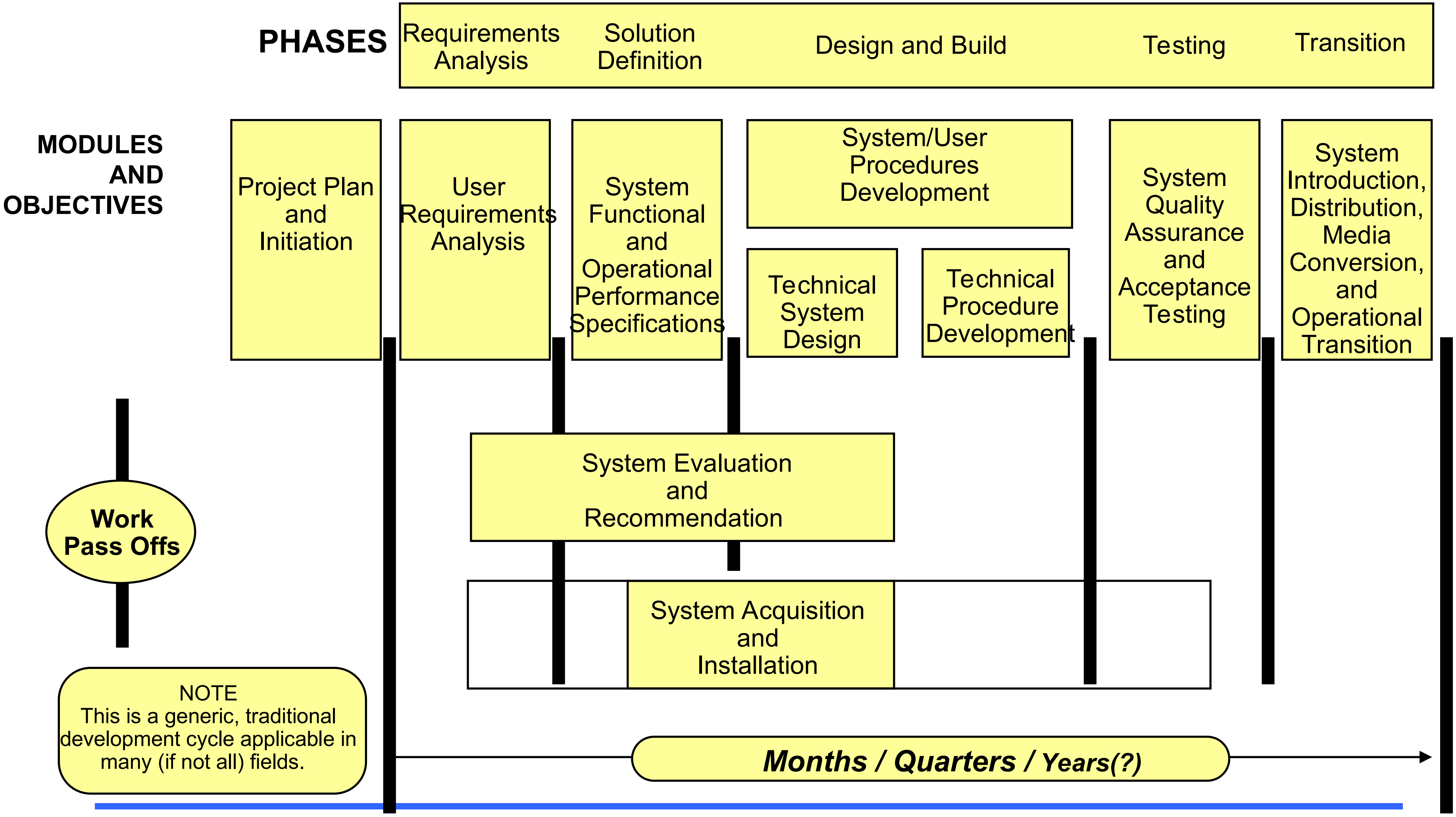
Solution Definition

Design and Build

Testing

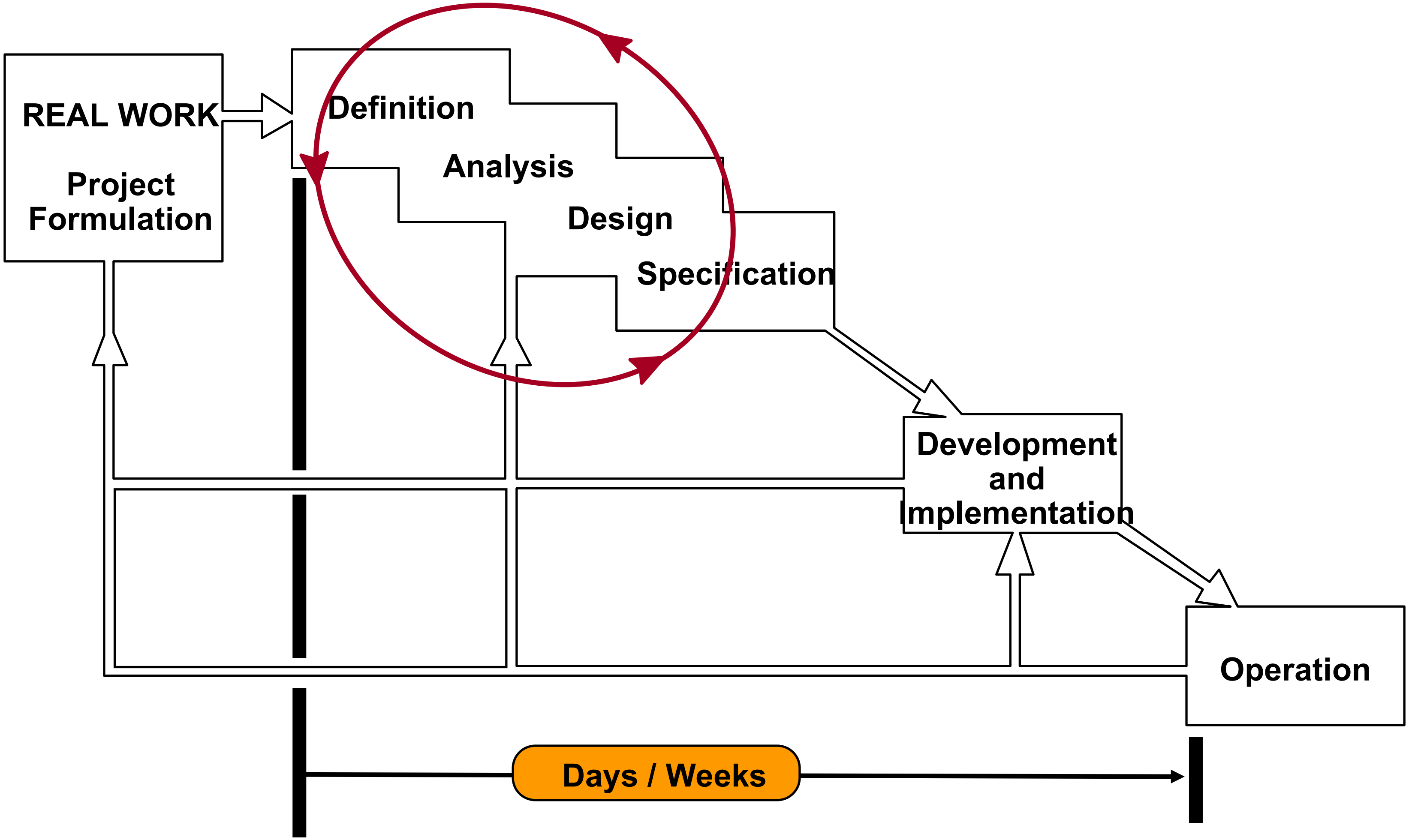
Transition

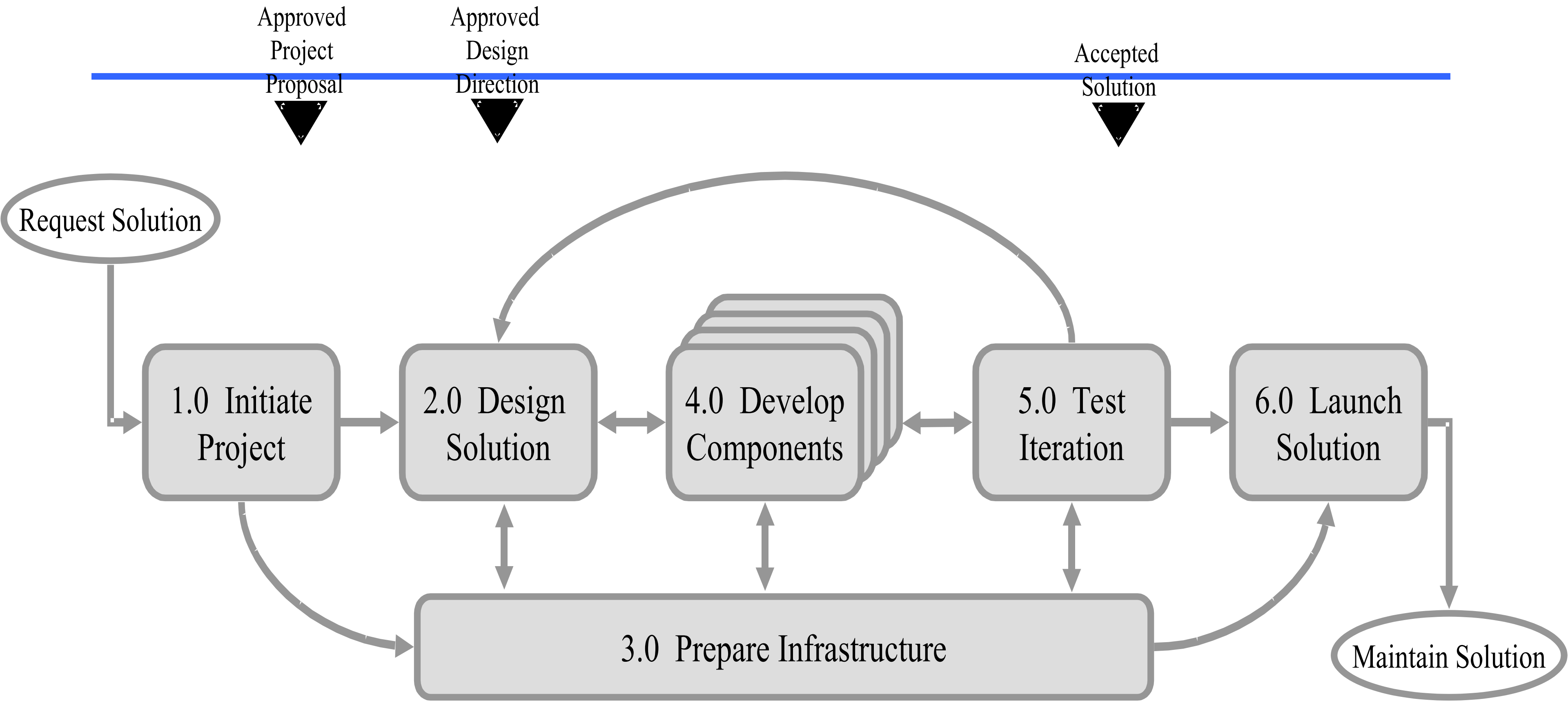
The software development life-cycle


PHASES
Requirements Analysis
Solution Definition
Design and Build
Testing
Transition
MODULES AND OBJECTIVES
Project Plan and Initiation
User Requirements Analysis
System Functional and Operational Performance Specifications
System/User Procedures Development
Technical System Design
Technical Procedure Development
System Quality Assurance and Acceptance Testing
System Introduction, Distribution, Media Conversion, and Operational Transition
Work Pass Offs
System Evaluation and Recommendation
System Acquisition and Installation
Months / Quarters / Years(?)

NOTE
This is a generic, traditional development cycle applicable in many (if not all) fields.

The software development life-cycle





Approved Project Proposal

Approved Design Direction

Accepted Solution

Request Solution

1.0 Initiate Project

2.0 Design Solution

4.0 Develop Components

5.0 Test Iteration

6.0 Launch Solution

3.0 Prepare Infrastructure

Maintain Solution

Notes:

"**Prepare Infrastructure**" includes planning and process development for change control, content workflow, testing, piloting, business transition, user training, user support, marketing, maintenance, and creating technical infrastructure.

"**Component**" includes all types of content (e.g. features, functions, modules, web pages, applications, calculators, etc.)

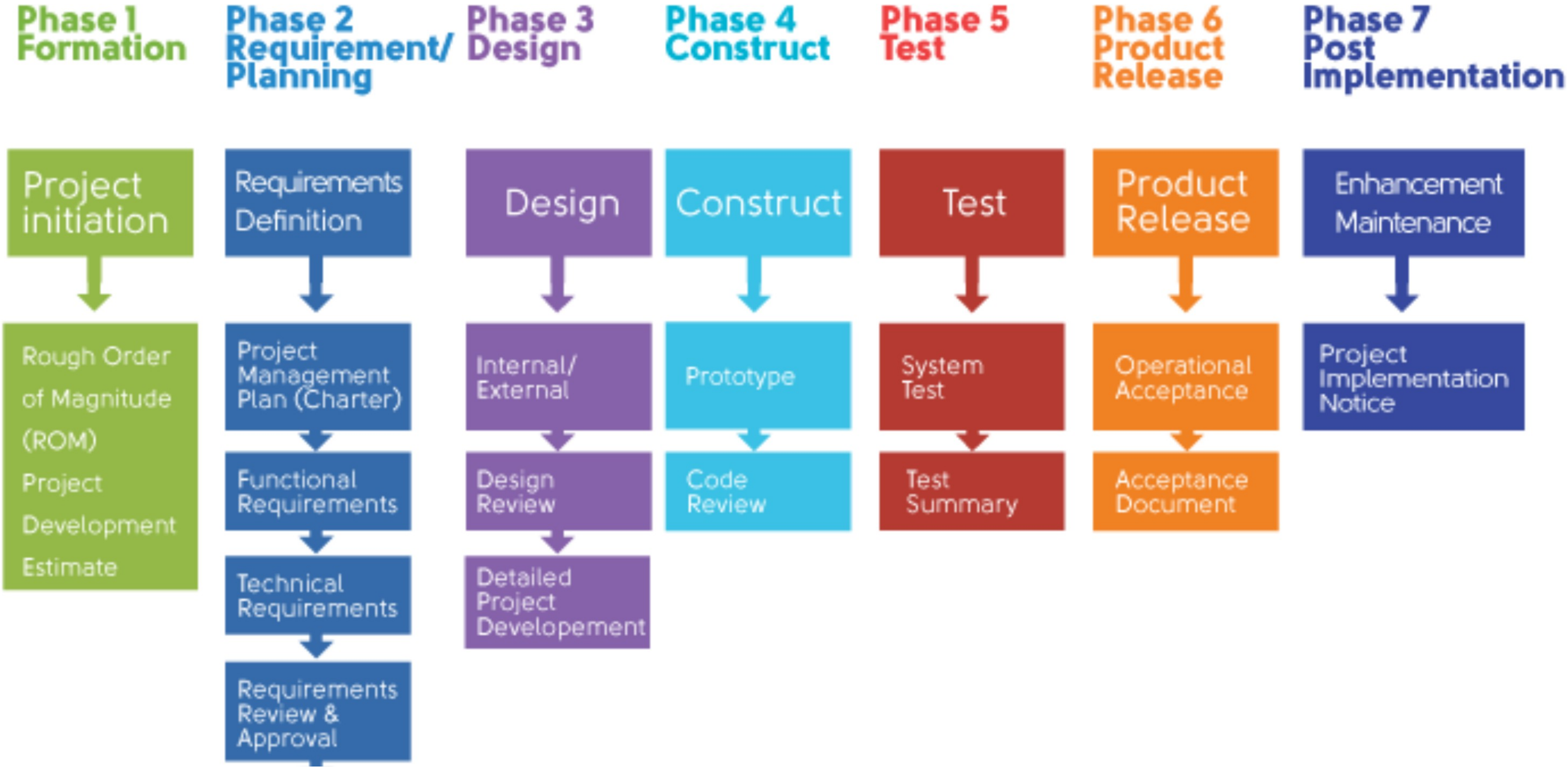
Legend:

▼ Milestone

▢ Project Process

▢ Concurrent Development

○ Preceding or Succeeding Process



Phase 1
Formation

Project initiation

Rough Order of Magnitude (ROM) Project Development Estimate

Phase 2
Requirement/
Planning

Requirements Definition

Project Management Plan (Charter)

Functional Requirements

Technical Requirements

Requirements Review & Approval

Phase 3
Design

Design

Internal/ External

Design Review

Detailed Project Developement

Phase 4
Construct

Construct

Prototype

Code Review

Phase 5
Test

Test

System Test

Test Summary

Phase 6
Product
Release

Product Release

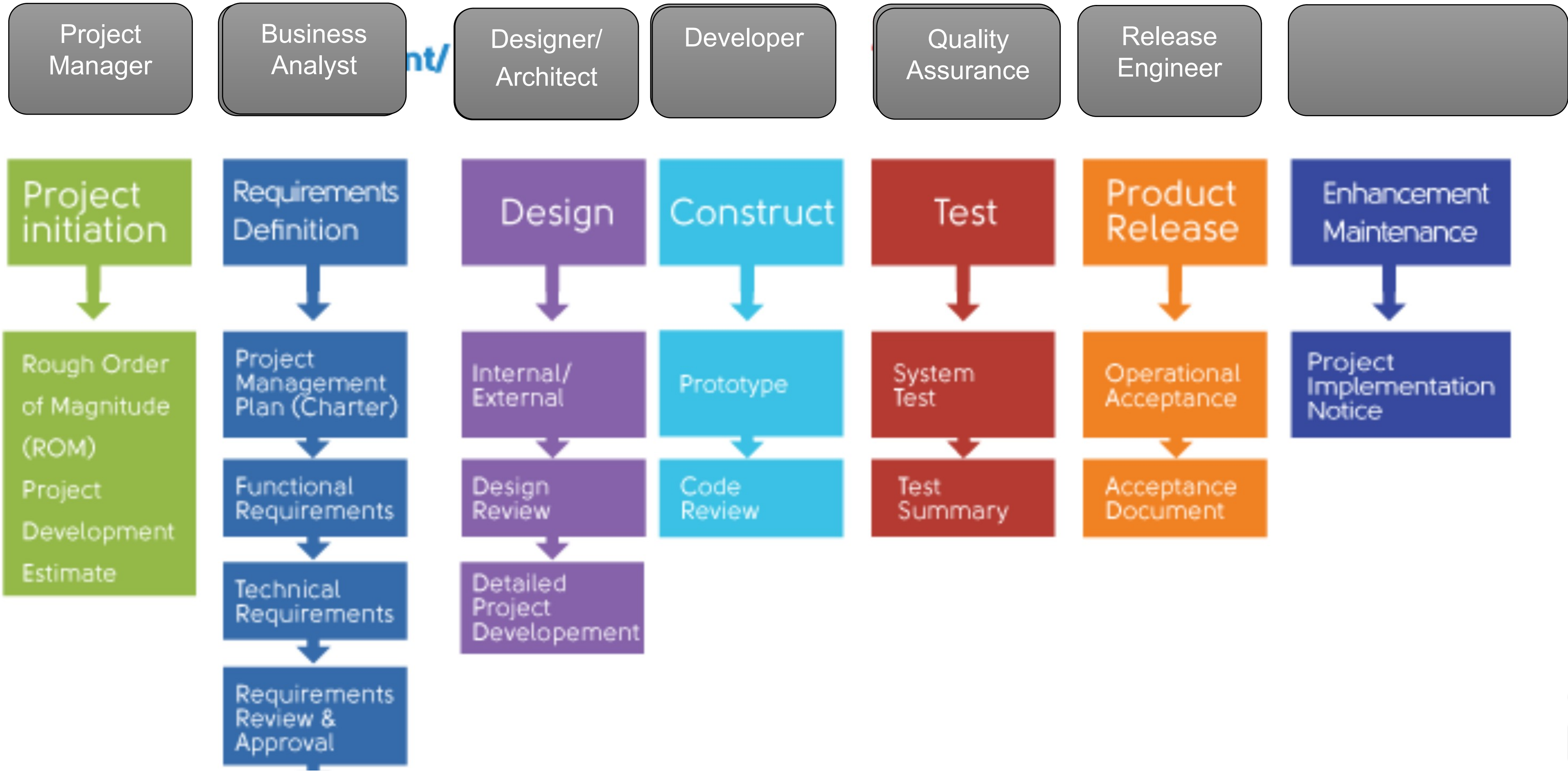
Operational Acceptance

Acceptance Document

Phase 7
Post
Implementation

Enhancement Maintenance

Project Implementation Notice



Project Manager

Project initiation

Rough Order of Magnitude (ROM)

Project Development Estimate

Business Analyst

Requirements Definition

Project Management Plan (Charter)

Functional Requirements

Technical Requirements

Requirements Review & Approval

Designer/ Architect

Design

Internal/ External

Design Review

Detailed Project Developement

Developer

Construct

Prototype

Code Review

Quality Assurance

Test

System Test

Test Summary

Release Engineer

Product Release

Operational Acceptance

Acceptance Document

Enhancement Maintenance

Project Implementation Notice

Business Requirements

- **What is the business need?**
- **What are the business issues underlying the need**
- **What is working? What is not working?**
- **What is the business' vision for the new system? What benefits and qualities that will result?**
- **Who are the stakeholder of the system? What are their needs and preferences?**
  - Users
  - Sponsors, etc.
  - All stakeholders are customers

Business Requirements (Contd.)

- **What is our offer to these customers?**
  - What are the features and benefits that will satisfy customers needs and preferences.
- **How are we going to deliver the offer?**
- **What is the business' "glossary of terms"?**
- **How users are going to interface to the new system? How are they going to interact with the new system? What business tasks they perform? What are the data elements they need? What tools are needed to help facilitate their work.**

Our Methodology: Problem-->Design-->Spec--Implement

Problem Definition

- □ Challenges
- □ Inefficiency
- □ High cost
- □ Miscommunication
- □ Waste
- □ Competitiveness

Design

- □ Model Components
- □ Modular design
- □ Components specialize
- □ Attributes
- □ Component interaction
- □ Use cases/functions
- □ User responsibility to solution

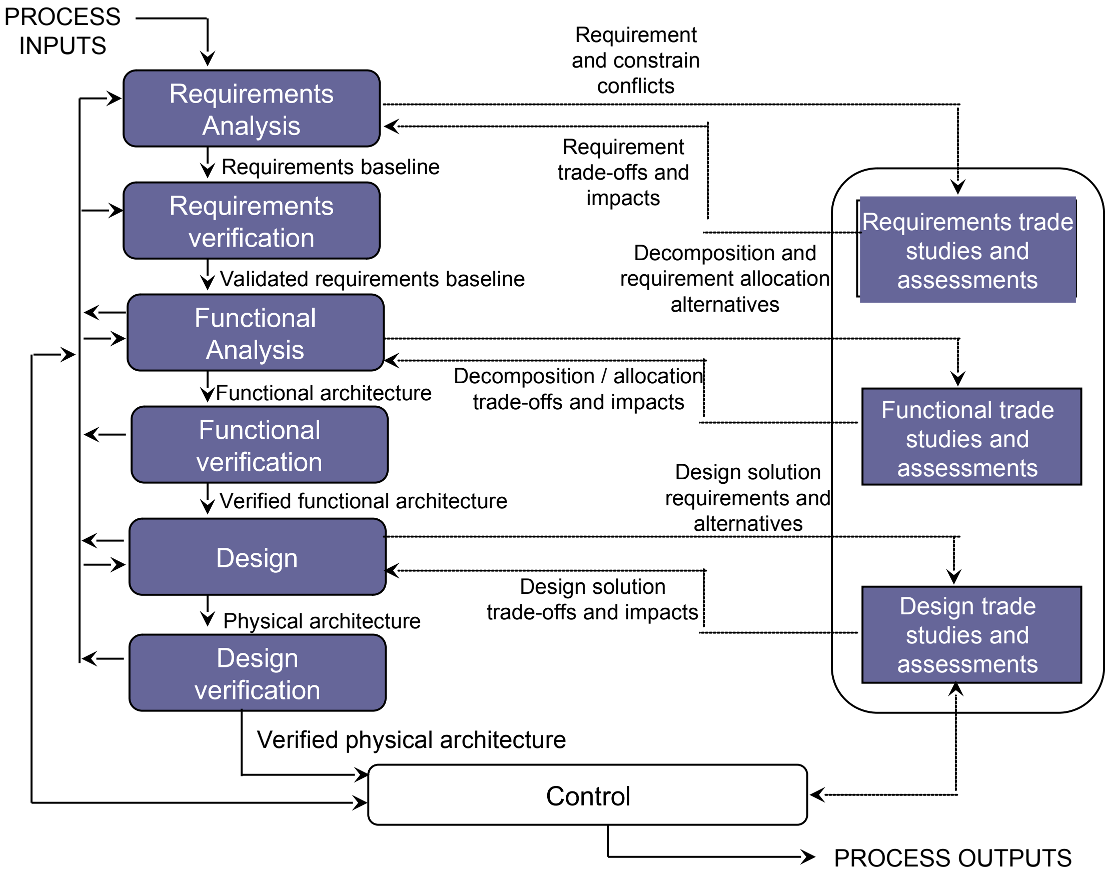
Specification

- □ Interfaces
- □ Interaction between methods
- □ Methods help each other
- □ Method Input specified
- □ Method output specified

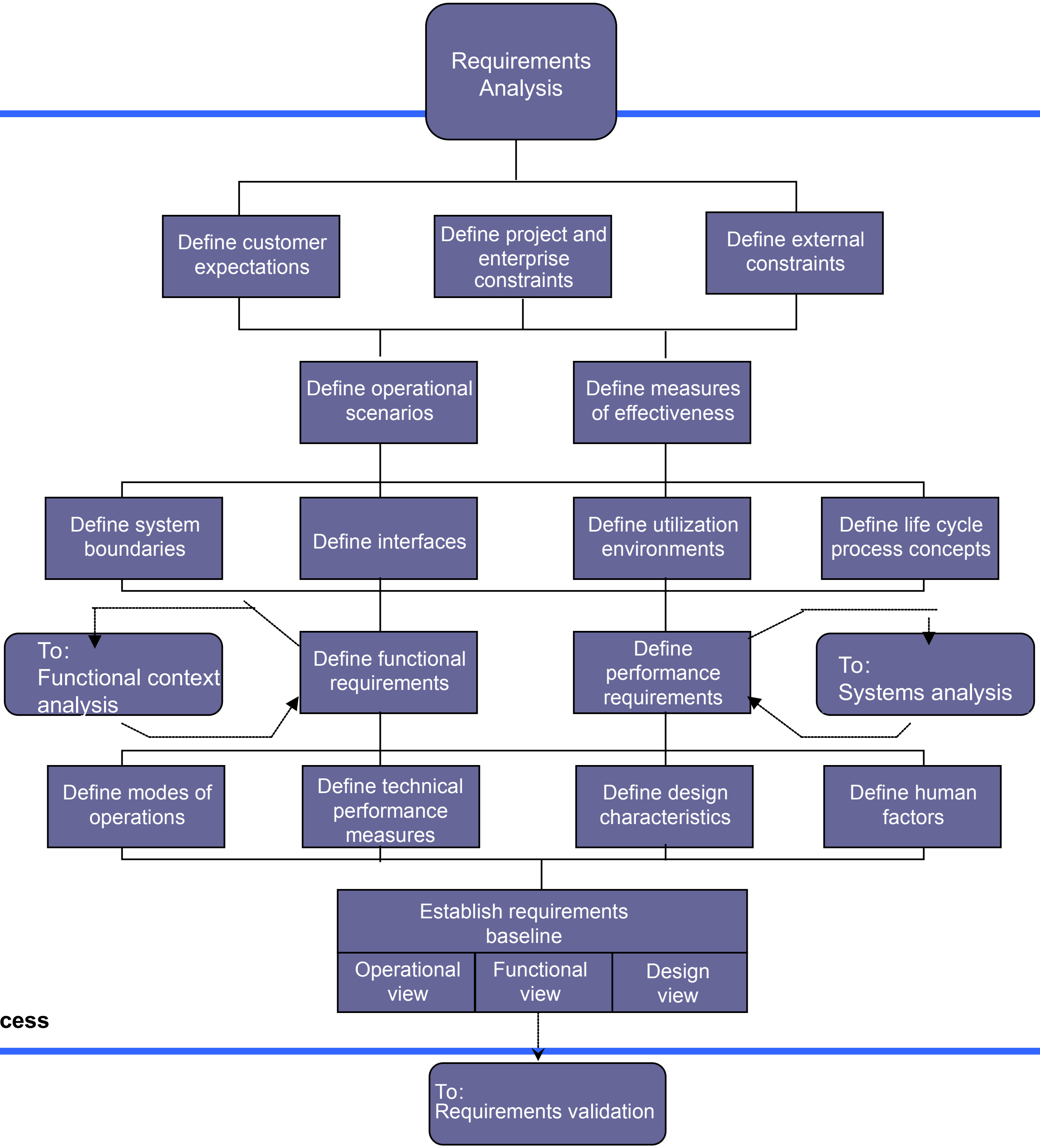
Implementation

- □ Java classes
- □ Java attributes and types
- □ Arraylist for group operations
- □ User interact components
- □ Interaction between swing components and java classes on the backend
- □ Front-end (UI)
- □ Backend (java classes)

Review of the IEEE standard for SDLC

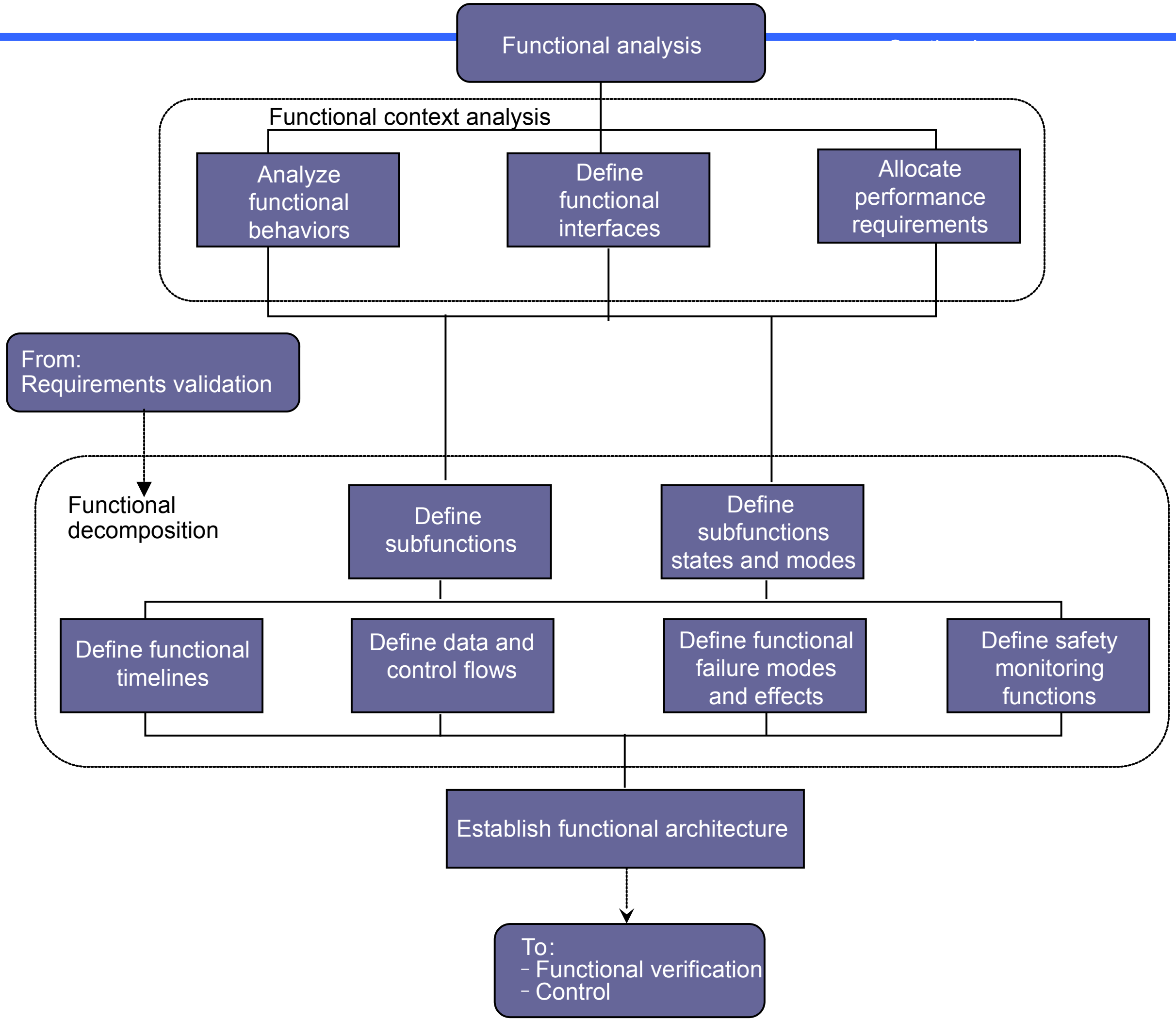

PROCESS INPUTS
Requirement and constrain conflicts
Requirements Analysis
Requirements baseline
Requirement trade-offs and impacts
Requirements verification
Requirements trade studies and assessments
Validated requirements baseline
Decomposition and requirement allocation alternatives
Functional Analysis
Functional architecture
Decomposition / allocation trade-offs and impacts
Functional verification
Functional trade studies and assessments
Verified functional architecture
Design solution requirements and alternatives
Design
Design solution trade-offs and impacts
Physical architecture
Design verification
Design trade studies and assessments
Verified physical architecture
Control
PROCESS OUTPUTS

Systems engineering process (SEP)



Requirements Analysis
Define customer expectations
Define project and enterprise constraints
Define external constraints
Define operational scenarios
Define measures of effectiveness
Define system boundaries
Define interfaces
Define utilization environments
Define life cycle process concepts
To: Functional context analysis
Define functional requirements
Define performance requirements
To: Systems analysis
Define modes of operations
Define technical performance measures
Define design characteristics
Define human factors
Establish requirements baseline
Operational view
Functional view
Design view
To: Requirements validation

Requirements analysis process



Functional analysis

Functional context analysis

Analyze functional behaviors

Define functional interfaces

Allocate performance requirements

From:
Requirements validation

Functional decomposition

Define subfunctions

Define subfunctions states and modes

Define functional timelines

Define data and control flows

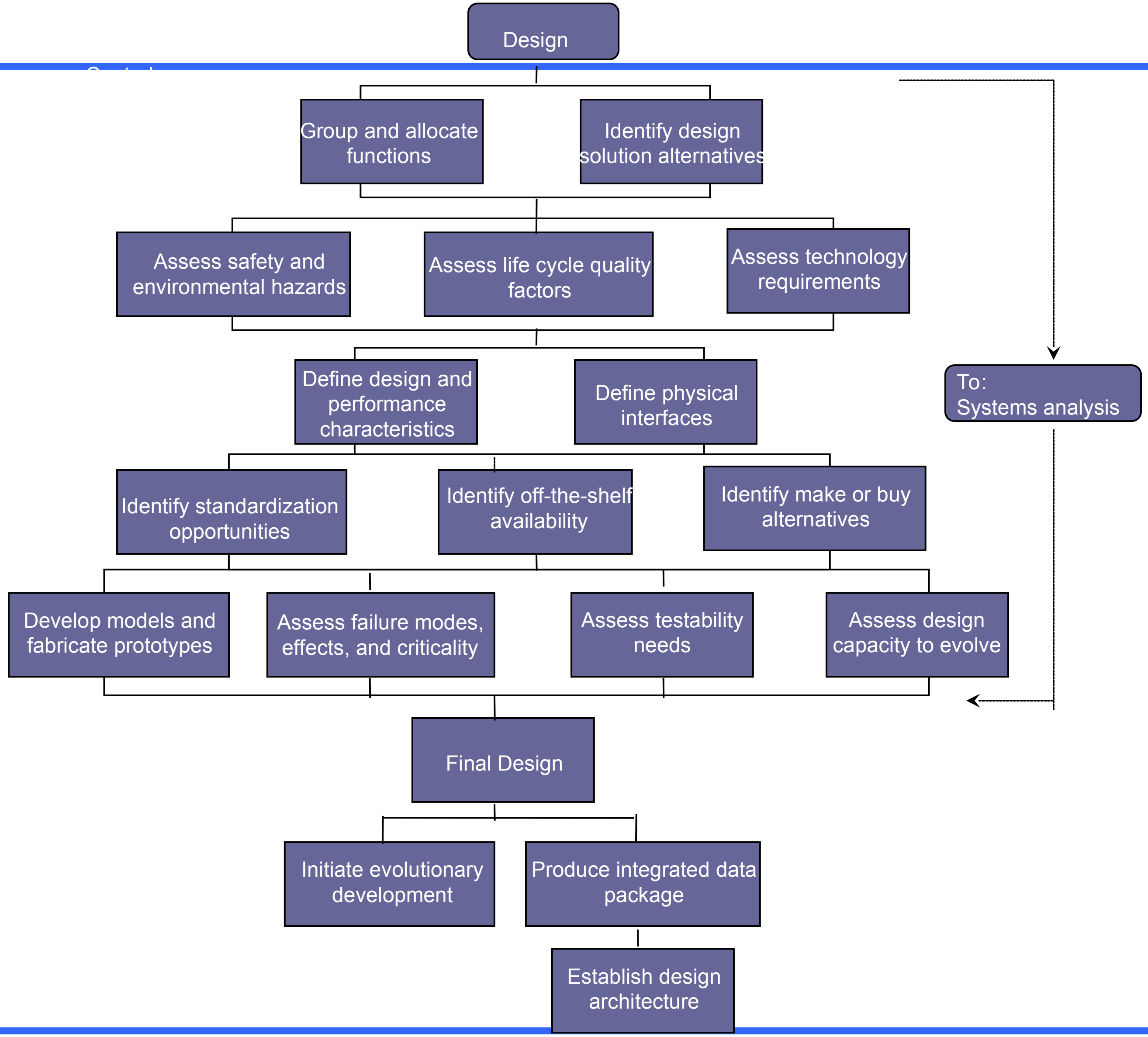
Define functional failure modes and effects

Define safety monitoring functions

Establish functional architecture

To:
- Functional verification
- Control

Functional analysis process



Design

Group and allocate functions

Identify design solution alternatives

Assess safety and environmental hazards

Assess life cycle quality factors

Assess technology requirements

Define design and performance characteristics

Define physical interfaces

To: Systems analysis

Identify standardization opportunities

Identify off-the-shelf availability

Identify make or buy alternatives

Develop models and fabricate prototypes

Assess failure modes, effects, and criticality

Assess testability needs

Assess design capacity to evolve

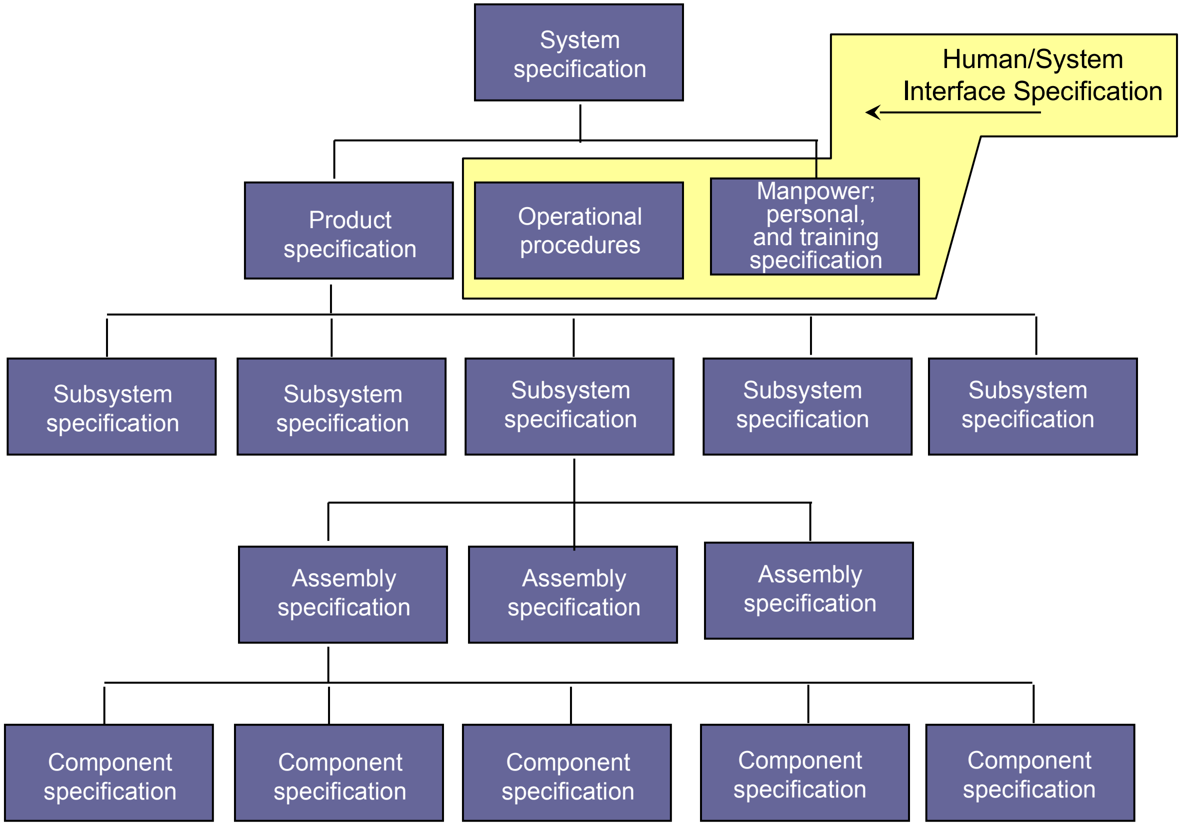
Final Design

Initiate evolutionary development

Produce integrated data package

Establish design architecture

Synthesis process



The specification tree

Key Take aways

- Companies survival depend on its digital strategy and applications.
- Software applications carry a lot of risk to the organization. Failure of deliver could have serious consequences on companies' financial health and growth strategies.
- The SDLC is a social group with so many people of different skills sets, and personalities, of course. Except in the simplest of software projects, the SDLC is vulnerable to confusion, miscommunication, overlap in responsibilities, delays, etc. So being organized and **deliberate** with good people and communication skills are critical for your success as a developer. (See Slide 3)
- Developers must understand the broader context of their programming job. For example, why you are programming this? what is the business value of the programs you are developing? That way, you will have a strong sense of your value to the organization –a necessary condition for your growth and success as a future IT Leader.