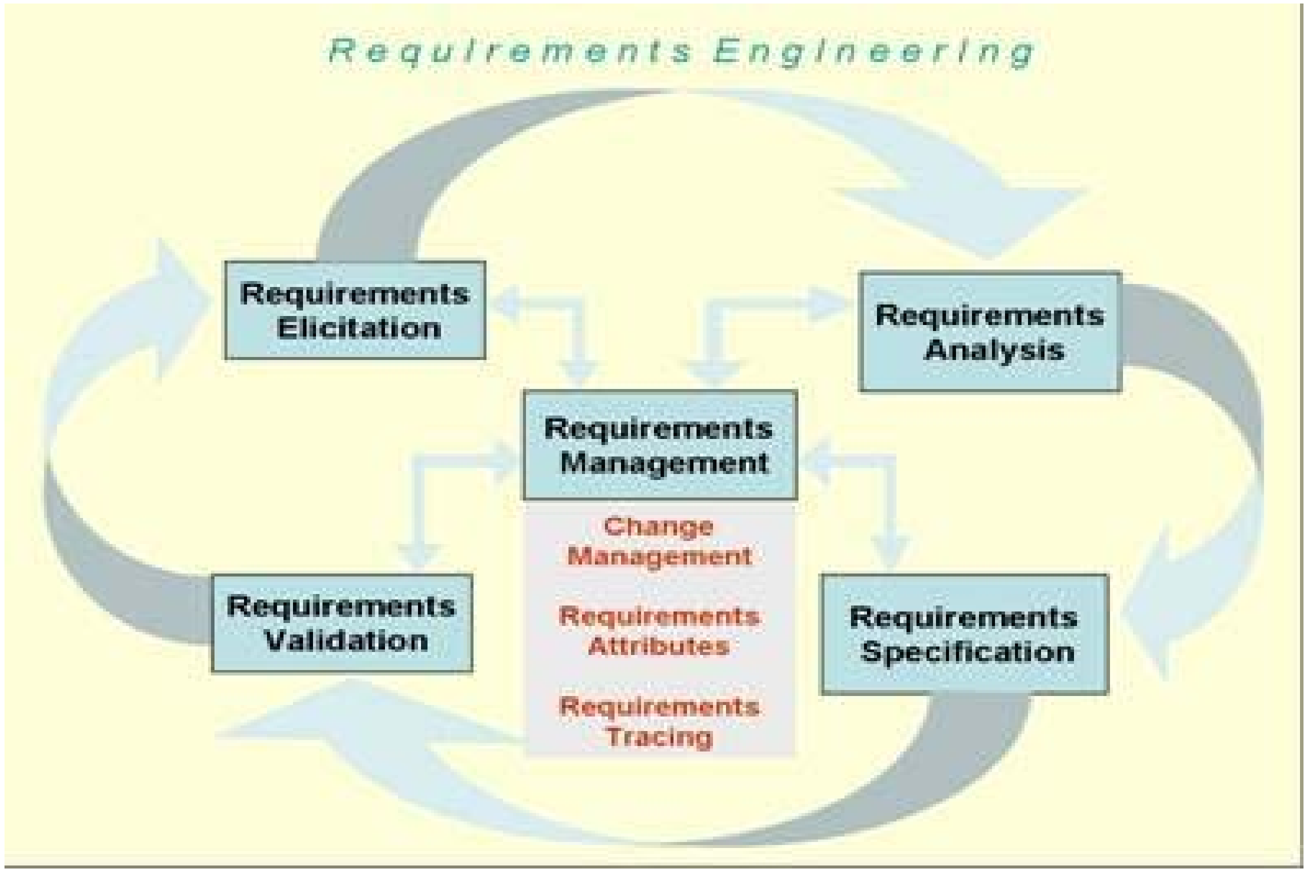
Overview of Requirements Engineering Activities

Key Activities of Requirements Engineering

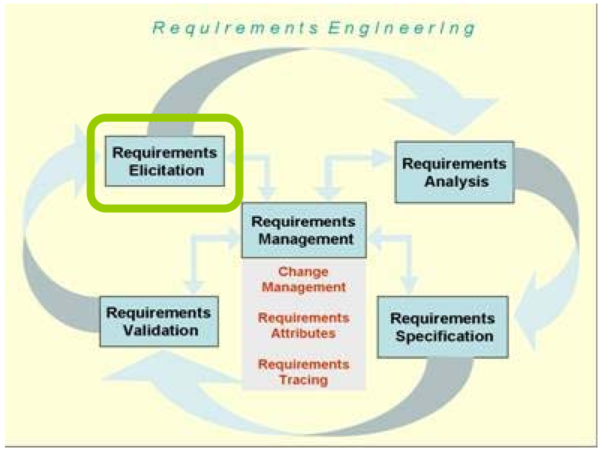


[DACS]

Requirements Process Activities

- Requirements Elicitation
  - Collect requirements from the stakeholders customers, users, developers, etc.
- Requirements Analysis
  - Study, analyze and model the problem to be solved
- Requirements Specification
  - Define and document the requirements in and organized and precise manner
- Requirements Validation
  - Ensure that the requirements are correct, complete, and consistent, and they satisfy the customer needs
- Requirements Management
  - Trace/track requirements and manage requirements change throughout the software life-cycle

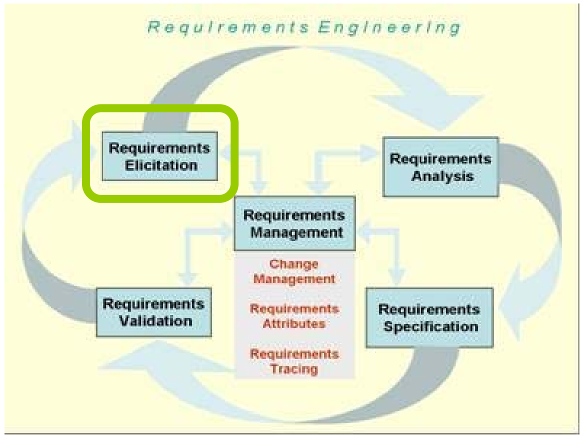
Elicitation



- Users know what they have, not what they need
  - They will better understand what they need after they see it, which is often too late

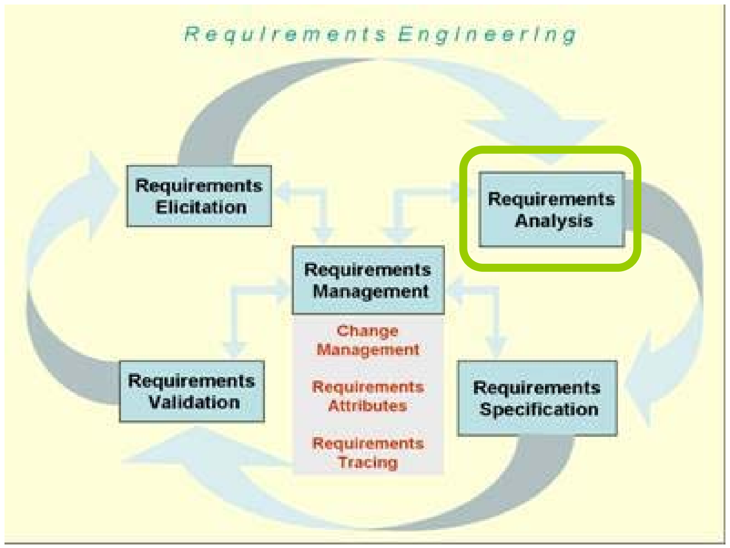
- User cannot always envision the possibilities enabled by technology

Elicitation



- Developer is rarely the user
- Diversity of users and stakeholders
- Support the *mission* of the user, not just the user
- User needs and missions are constantly changing
- The new system will impact the user's needs, resulting in new system needs (cycle)
- Hard to understand the constraints of legacy systems

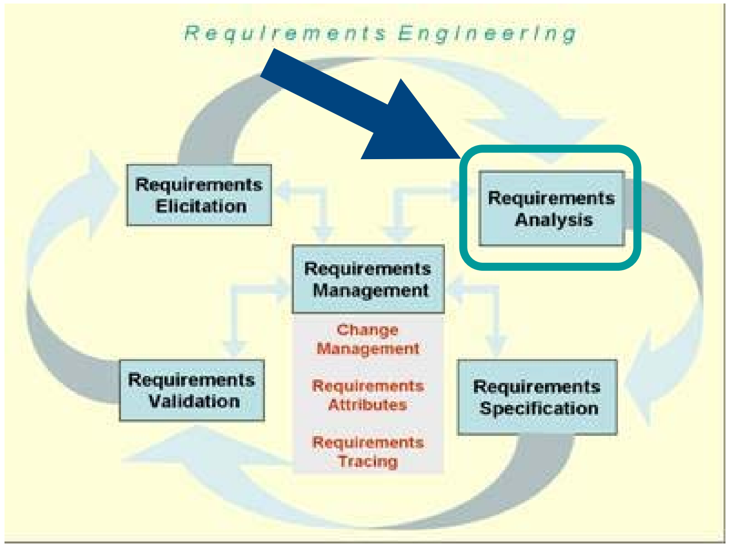
Requirements Analysis



1. The process of studying and refining system, hardware, or software requirements.

[IEEE-STD-610.12-1990 Software Engineering Glossary]

Requirements Analysis



- Requirements analysis
  1. The process of studying user needs to arrive at a definition of system, hardware, or software requirements.
  2. The process of studying and refining system, hardware, or software requirements.

  [IEEE-STD-610.12-1990 Software Engineering Glossary]

Usually results in a semi-formal or formal, rigorous, precise model of the system requirements

Requirements Analysis

- Objectives
  - Detect and resolve requirements conflicts
  - Discover the bounds of the system and how it interacts with its environment
- Activities
  - Classify requirements for planning, reporting and tracking
    - Source, type, priority, risk, scope, volatility/stability
  - Prioritize requirements based on relative importance and risk
  - Conceptual modeling of a solution to help understand the problem
  - Architectural design and requirements allocation
    - Functional and non-functional grouping
    - Allocation of requirements to subsystem components
  - Requirements negotiation and conflict resolution
    - Incompatible features, incompatible features and constraints, conflict between scope and resources, etc.

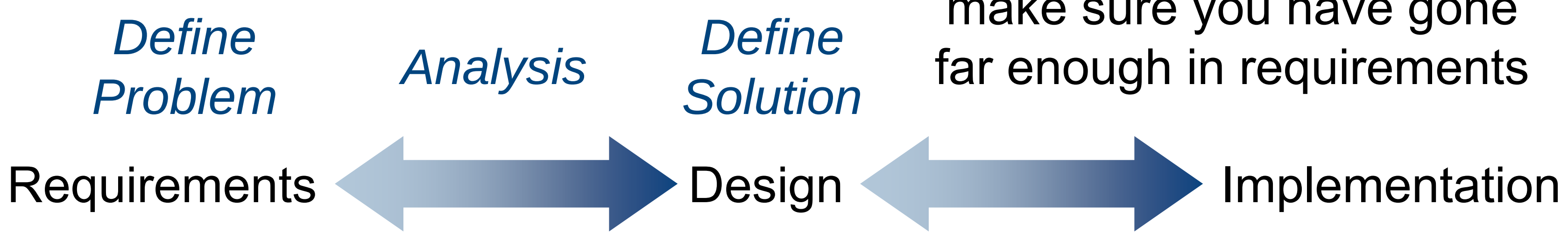
[DACS]

Is Analysis a Requirements Activity or a Design Activity?

- Goal: keep the requirements "pure" – free of design decisions
  - "What" not "How"
  - Allow freedom of design choices
- Analysis is part of the <u>process</u> of producing requirements and designs: of defining and solving the problem
  - Iterate between "What" and "How" (between reqts. and design activities)
  - The requirements that <u>result</u> from the analysis process can be "pure" but the process itself often iterates between requirements and design
  - Iteration is good!
- So, don't get hung up on "analysis as a requirements activity versus as a design activity"
  - The important point is that analysis gets done!
  - Design "removed from" the requirements can be saved for later design focus

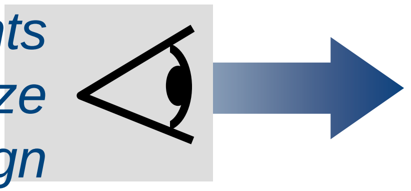
Iteratively Analyze the Problem and Synthesize a Solution

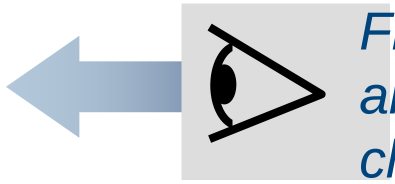
Go too far into design to make sure you have gone far enough in requirements



Classify and structure the requirements to provide a functional view of the architecture

From a requirements perspective, synthesize a feasible design





From a design perspective, analyze the requirements for clarity, completeness, etc.

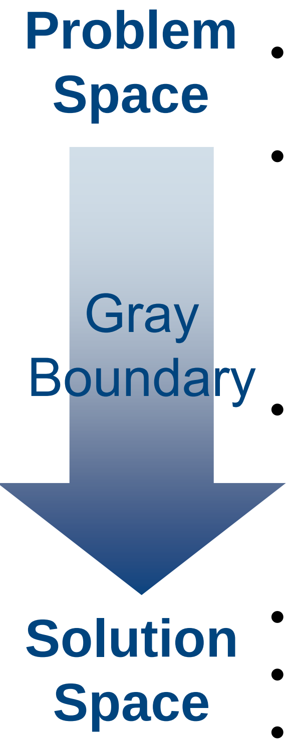
Essential Requirements

- Capture the essence of the need without biasing solution design options
  - The system shall beep and put a warning dialog box on the screen if ...
    - ► The system shall issue an alert if ...
  - The user clicks the "Submit" button
    - ► The system shall accept an indication that the user has completed ...
  - The user enters their ID and password
    - ► The system shall verify user identity

Hint for getting the essence: for every user action, ask "Why?"

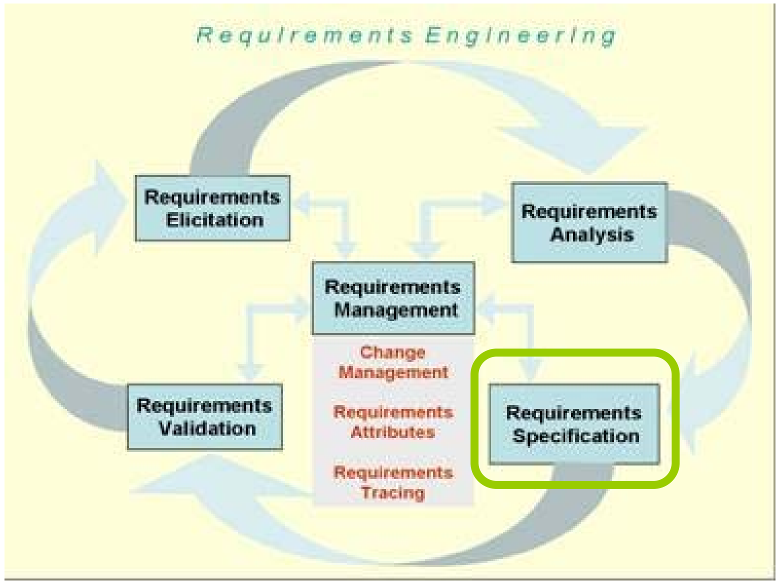
Requirements Model, Analysis Model, and Beyond

- The use-case model is a model of the requirements of a system, organized from a user perspective (external to the system)
- The analysis model is a model of the requirements of the system, organized from a system perspective (internal)



- Requirements modeling specifies the desired behavior of the system (model as use cases, activity diagrams, flow charts, etc.)
- Object-oriented analysis specifies classes (attributes and behavioral responsibilities), object interactions, object state machines, packages of classes and dependencies, etc.
  - Initial realization of the requirements in terms of interacting objects
- Object-oriented design specifies architectural mechanisms, design details, implementation packaging, etc.
  - Detailed realization of the requirements in terms of interacting objects
- Object-oriented implementation provides compilable specifications
- Run-time execution instantiates objects and their interactions
- Testing exercises (executes) the system to verify and validate its required behavior

What is a Requirements Specification?



- A document that specifies the requirements for a system or component. Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards.
  - [IEEE-STD-610.12-1990 Software Engineering Glossary]

Contrast With: Design Description

- Design description: A document that describes the design of a system or component. Typical contents include system or component architecture, control logic, data structures, input/output formats, interface descriptions, and algorithms. *Synonyms:* design document; design specification. *See also:* product specification. *Contrast with:* requirements specification
  - [IEEE-STD-610.12-1990 Software Engineering Glossary]

Specify?

- Webster's II New Riverside Dictionary (1984)
  - Specify:
  1. To state, name, or mention explicitly, exactly, and unambiguously.
- Merriam Webster Online Dictionary (http://www.m-w.com, accessed 2004-09-06)
  - Specify:
  1. to name or state explicitly or in detail
  2. to include as an item in a specification

So, where does this level of precision come from?

What is a “Good” Requirements Specification?

- Correct
- Unambiguous
- Complete
- Consistent
- Characterized with attributes (priority, stability, timing, etc.)
- Verifiable (testable)
- Modifiable
- Traceable (clear identifier, traced to source, design, implementation, and test, and to related requirements)
- Free of design and implementation details (unless they are constraints)

What is a Software Requirements Specification (SRS)?

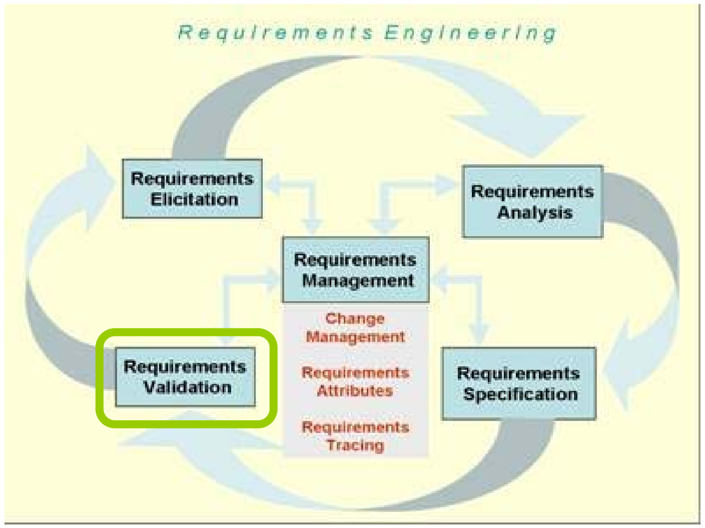
- Software requirements specification: Documentation of the essential requirements (functions, performance, design constraints, and attributes) of the software and its external interfaces. *See:* IEEE-STD-1012 [*sic:* IEEE-STD-830?]

- Specification: A document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system or component, and, often, the procedures for determining whether these provisions have been satisfied. *See also:* formal specification; product specification; requirements specification.

[IEEE-STD-610.12-1990 Software Engineering Glossary]

How do you know you have the right requirements?

Requirements Validation



- Requirements validation:
  - Certify that the requirements meet the stakeholders' intentions
  - Ensure that the requirements define the right system
  - Ensure a common understanding across the project team and among the stakeholders

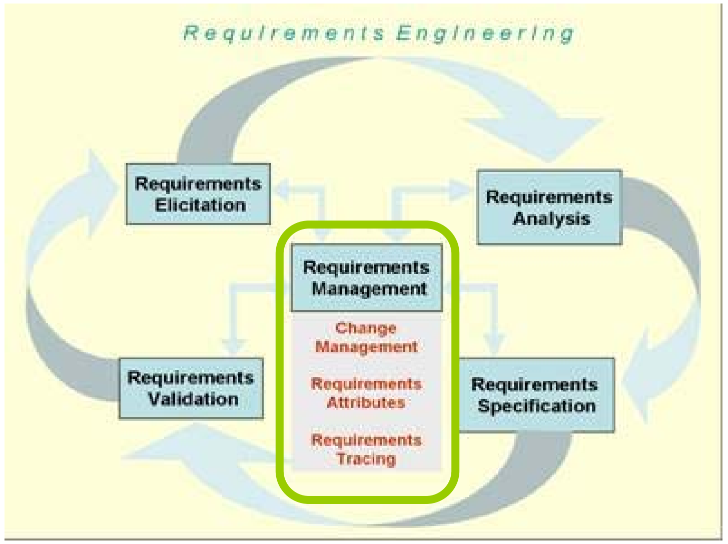
- Validation should not be confused with verification.
  - Validation: are we building the right system?
  - Verification: have we built the system right?

[DACS]

Requirements Validation Techniques

- Requirements reviews
- Prototyping to elicit and validate requirements
- Review requirements analysis models
- Prove properties of formal analysis models
- Plan how each requirement will be verified in acceptance tests

What is Req. Management?



- The process of eliciting, documenting, organizing, and tracking changing requirements and communicating this information across the project team to ensure that iterative and unanticipated changes are maintained throughout the project lifecycle [DACS].

Problems of Requirements Management

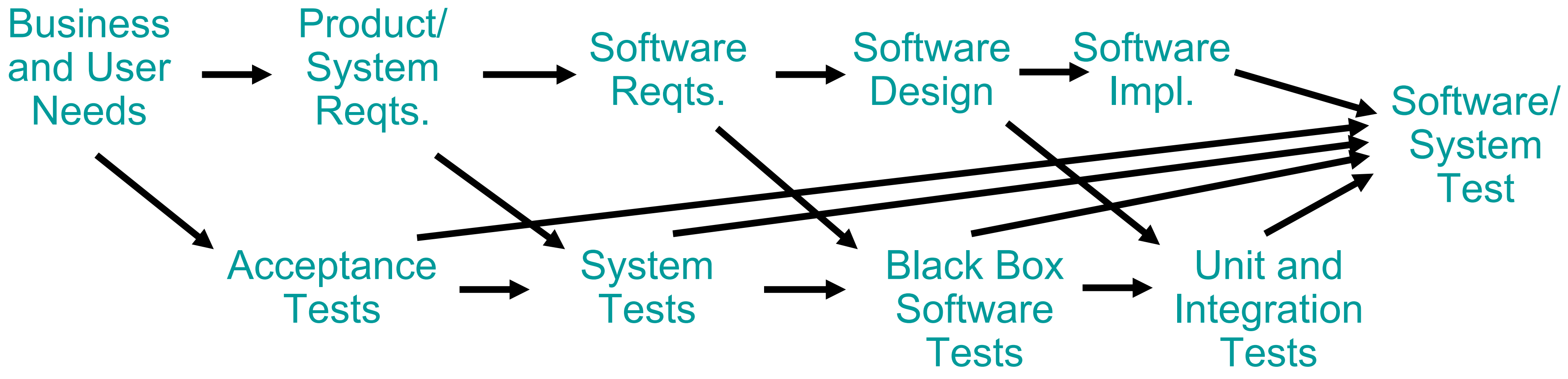
- Requirements are not always obvious and have many sources
- Requirements are not always easy to express in words
- There are many different types of requirements at different levels of detail
- The number of requirements can become unmanageable if not controlled
- Requirements are related to one another and to other deliverables of the process in a number of ways
- Requirements have unique properties or property values
- There are many interested and responsible parties
- Requirements change
- Requirements can be time-sensitive

Requirements Management Skills

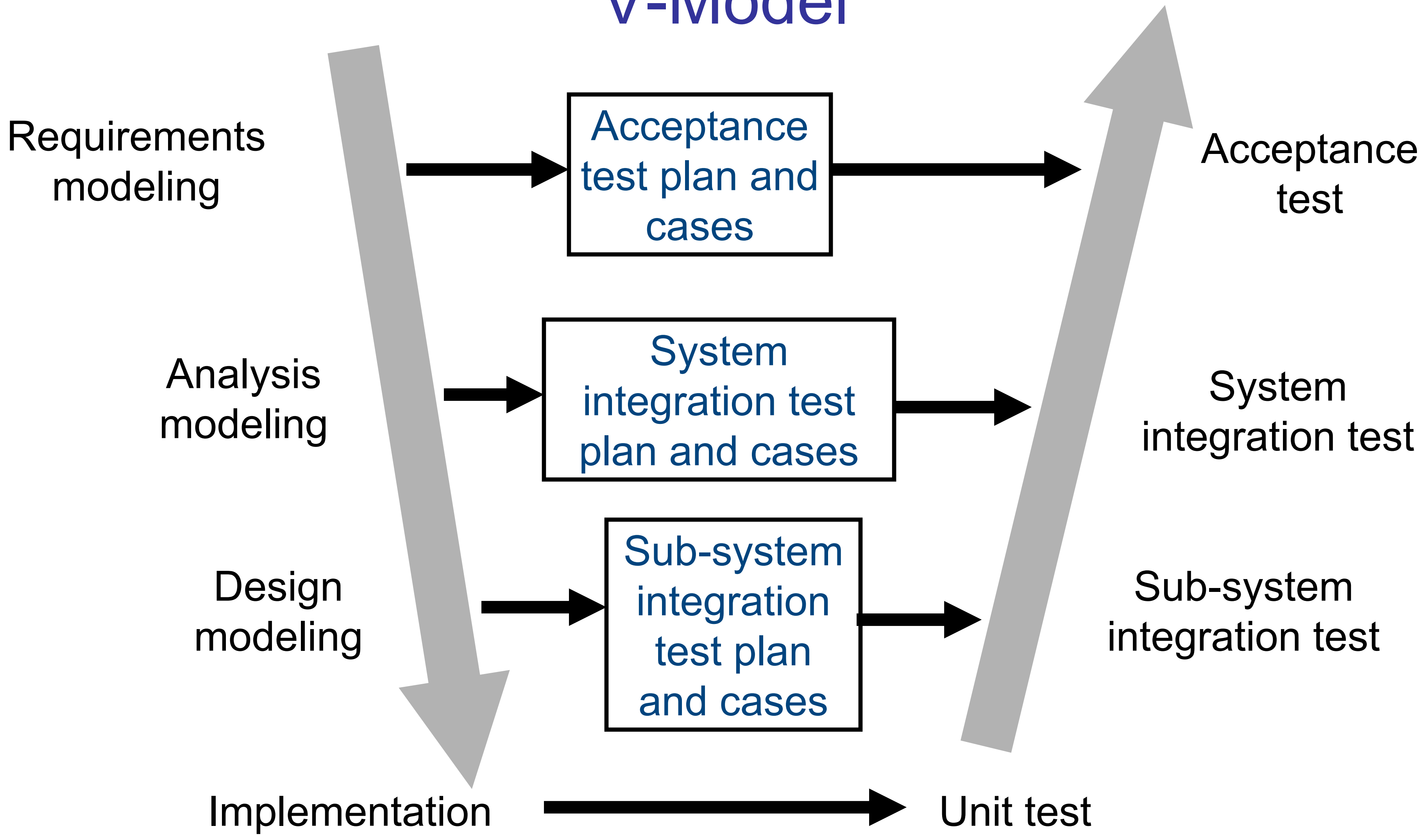
- Application domain understanding
- Problem analysis
- Understanding stakeholder needs
- Defining the system
- Managing the scope of the project
  - Fighting scope creep
- Refining and detailing the system definition
- Managing changing requirements
  - The actual business requirements do not often change
  - Our understanding of them changes often
  - Our inclusion of them in our product scope changes often

How do you know a system meets its requirements?

Requirements Define Tests



V-Model


Requirements modeling
Acceptance test plan and cases
Acceptance test
Analysis modeling
System integration test plan and cases
System integration test
Design modeling
Sub-system integration test plan and cases
Sub-system integration test
Implementation
Unit test

Sources

- Karl Wiegers, *Software Requirements*, 2nd Edition, Microsoft Press, 2003.
- Rational Unified Process (RUP)
- Unified Process for Education (UPEDU –Yoopeedoo) (http://www.upedu.org/)
- Data Analysis Center for Software (DACS) (http://www.dacs.dtic.mil/) Requirements Management Gold Practice (http://www.goldpractices.com/practices/mr/index.php)
- IEEE-STD-830-1998 Recommended Practice for Software Requirements Specifications
- Software Engineering Body of Knowledge (http://www.swebok.org/)
- Brooks, Fred P., *The Mythical Man-Month*, 20th Anniversary Edition, Addison Wesley, 1995.
- [Standish 1995] Standish Group, *The Standish Group Report: Chaos*, 1995, http://www.scs.carleton.ca/~beau/PM/Standish-Report.html.
- Based on material from Professors Ludi and Hawker, RIT SE Department