Software Engineering

COEN 174

Ron Danielson

Requirements Engineering

Chapter 6

Objectives

- Enumerate the phases of requirements engineering
- Describe the details of each phase and know how to perform the tasks associated with each phase
- Understand some graphical representations associated with various phases

Requirements Specification

- A contract between the client and the system developers
- Describes **what** the system has to do
- Does **not** describe **how** the software is to be constructed
- If some requirements are **missing**, specification is **incomplete**
- If some requirements are **conflicting**, specification is **ambiguous** or **inconsistent**

Requirements Specification cont.

- Often divided into
  - **High-level** requirements
    - More readable overview
    - Outlines why the system is being built, benefits to be achieved
    - Should be written so they are unlikely to change
  - **Low-level** (or detailed) requirements
    - Highly detailed descriptions of work the system will perform
    - Often evolve as the requirements are analyzed or as the system is being developed
- Maintaining requirements specification over time is critical and difficult

Requirements

- Form the foundation for system design
- Provide a basis for acceptance testing
- Help limit scope creep
- Inform scheduling to lead to on-time completion
- Help define training and support services needed

Recall the Difference

- Functional requirements
  - Describe what the system will do
  - “The system will”
  - Are either met or not
- Nonfunctional requirements
  - How functional requirements are accomplished
  - “The system will be”
  - Can be achieved to varying degrees

Nonfunctional Requirement Categories

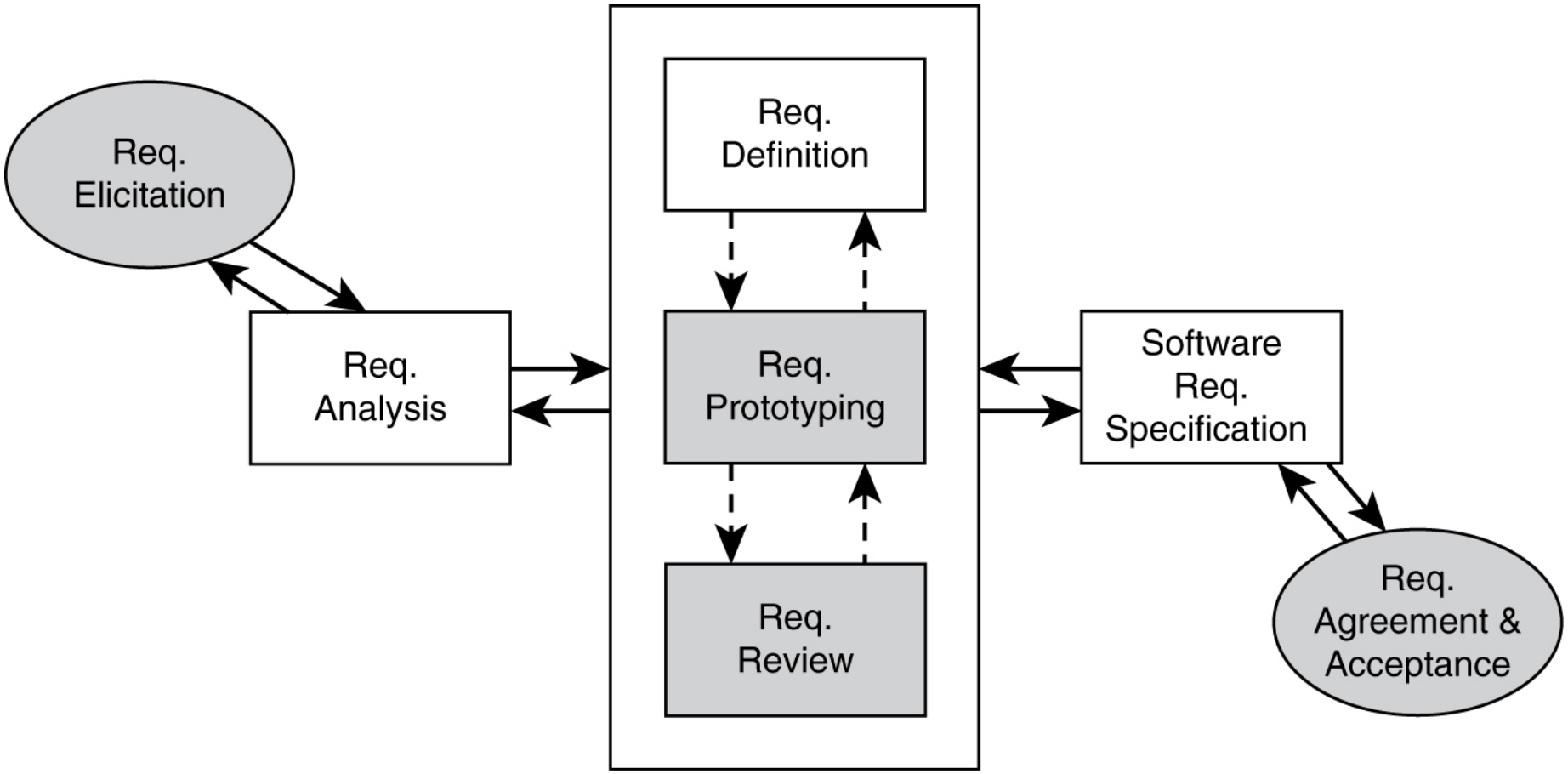
- Quality attributes
  - Reliability and availability
  - Performance (response, throughput, storage use)
  - Security
  - Maintainability
  - Portability
- Constraints
- External interfaces
  - Hardware
  - Other software
  - External agents
- User interfaces
- Error handling

Before Beginning on Requirements

Requirements Engineering

- A process to get from an initial inquiry and discussion about a possible system to a contract for development and delivery
- Steps include
  - Requirement elicitation
  - Requirement analysis
  - Requirement definition
  - Requirement prototyping
  - Requirement review
  - Requirement specification
  - Requirement agreement

Requirement Engineering Process



- Grayed Boxes represents direct user/customer involvement

Requirements Elicitation

- Heavy involvement with client
- Strongly focused on business needs and issues
- **What** not **how**
- People who do this often have a business background
  - Need great communication skills
  - Need to understand the business domain and how the system will be used
  - Need to understand what's possible and lead the client discussion

High-Level Requirements

- Define sponsors
- Business case to justify system
- Scope definition
  - What software will do within the organization
- Major constraints on development and deployment
  - Cost, sites, schedule…
- Major functionality
- Success definition
- User characteristics

Low-Level Requirements

- Usually deal with technical concerns
  - Including standards compliance
- Often tradeoffs among issues
- Best practice is to write tests as each detailed requirement is written
- Each requirement needs to be **traceable**
  - Uniquely identifiable
  - Requirement to person or document that was the source, and requirement to elements of implementation
  - Initial documents to requirement
  - Elements of system to requirement

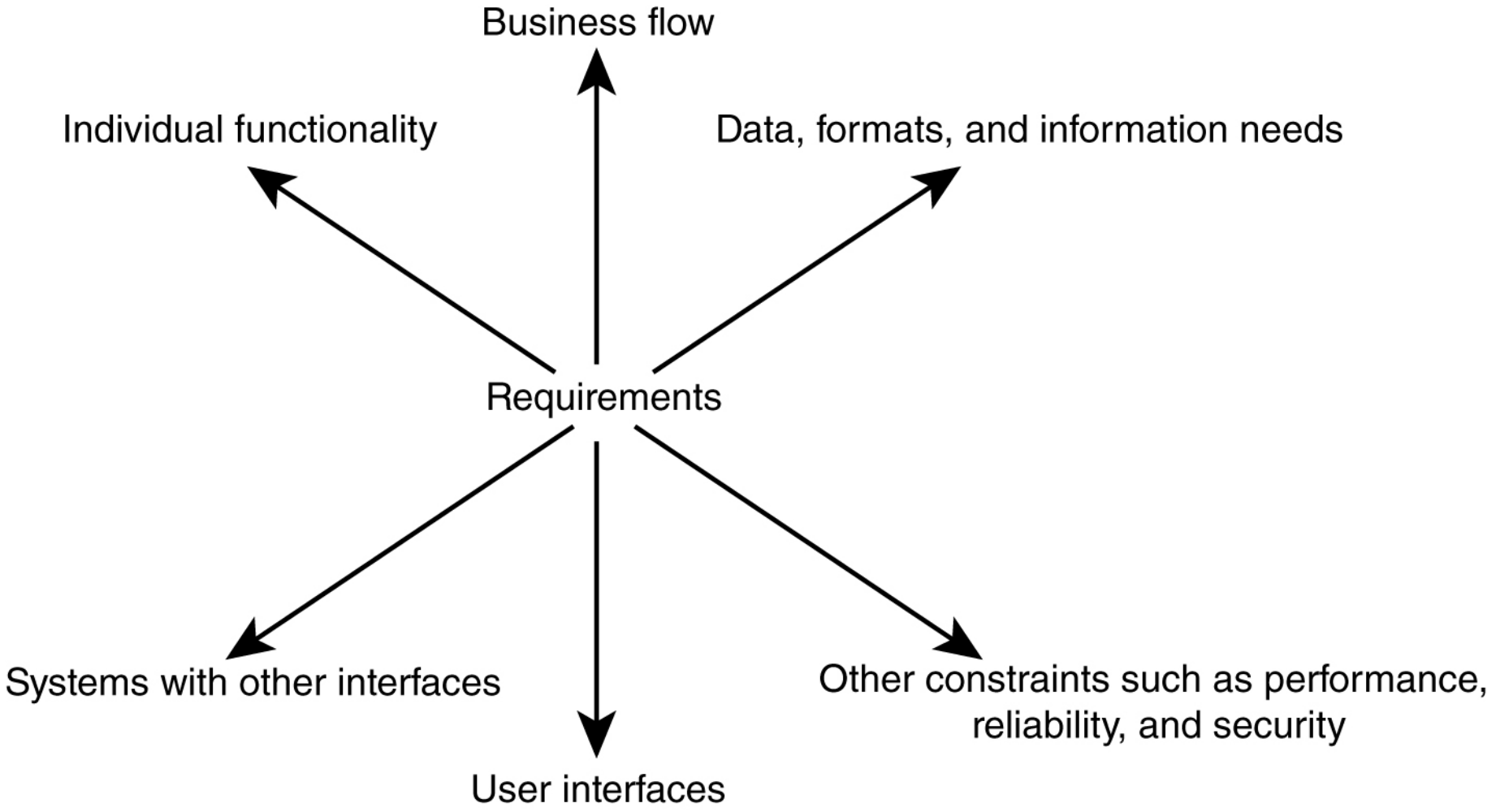
Sample Traceability Table

Requirement	Design	Code	Test	Other Related Requirements
1	Component .X	Module 3	Test Case 32	2, 3
2	Component .A	Module 5	Test Case 16	1
3	Component .P	Module 2	Test Case 27	1
4				

Traceability

- Tables make it easy to follow how each requirement is satisfied
  - Good for documentation and verification, but not necessarily for day-to-day use
- Many software organizations use hypertext to facilitate tracing through the various artifacts that are created during the development process

Low-Level Requirements (cont.)



Low-Level Requirements (cont.)

- Requirements are often in the context of existing systems
- Functionality in the context of business flow
- Interfaces
  - Error reporting
  - With existing systems
    - Control and data transfer to and from, retry on error
- Other constraints are usually non-functional

System Characteristics

- Physical environment and locations
- Interfaces in and out
- Users
  - Who, how many, characteristics, range of expertise
- Functionality and constraints on time and space
- What documentation is needed and for whom
- Data format, volume, persistence, accuracy
- Hardware and personnel resources
- Security, access control, backup

Categorizing Detailed Requirements

- By feature or function
  - + Maps to application easily
  - - Doesn't map well to OO implementation
- By use case
  - + Easy to understand
  - - Hard to trace to design and code
- By GUI
  - + Easy to understand
  - - Not every function is in a GUI
  - - Hard to trace to design and code

Categorizing Detailed Requirements (cont.)

- By state
  - + Easy to understand
  - - Can be hard to allocate requirements to a single state
- By class
  - + Easy to trace to code
  - - Can be hard for client to understand