Lecture # 37-38

Review of Lectures 1-21

Introduction

- Requirements form the basis for all software products
- Requirements engineering is the process, which enables us to systematically determine the requirements for a software product

Software Requirements - 1

- Complete specification of the desired external behavior of the software system to be built
- What is external behavior?
- Software requirements may be:
  - Abstract statements of services and/or constraints
  - Detailed mathematical functions

Software Requirements - 2

- Software requirements may be:
  - Part of the bid of contract
  - The contract itself
  - Part of the technical document, which describes a product

Sources of Requirements

- Stakeholders
  - People affected in some way by the system
- Documents
- Existing system
- Domain/business area

Kinds of Software Requirements

- Functional requirements
- Non-functional requirements
- Domain requirements
- Inverse requirements
- Design and implementation constraints

Functional Requirements - 1

- Statements describing what the system does, i.e., functionality of the system
- Statements of services the system should provide
  - Reaction to particular inputs
  - Behavior in particular situations
- Sequencing and parallelism are also captured by functional requirements

Examples

- The system shall solve a quadratic equation using the following formula

$$x = (-b \pm \text{sqrt}(b^2 - 4*a*c))/2*a$$

- The user shall be able to search either the entire database of patients or select a subset from it (admitted patients, or patients with asthma, etc.)

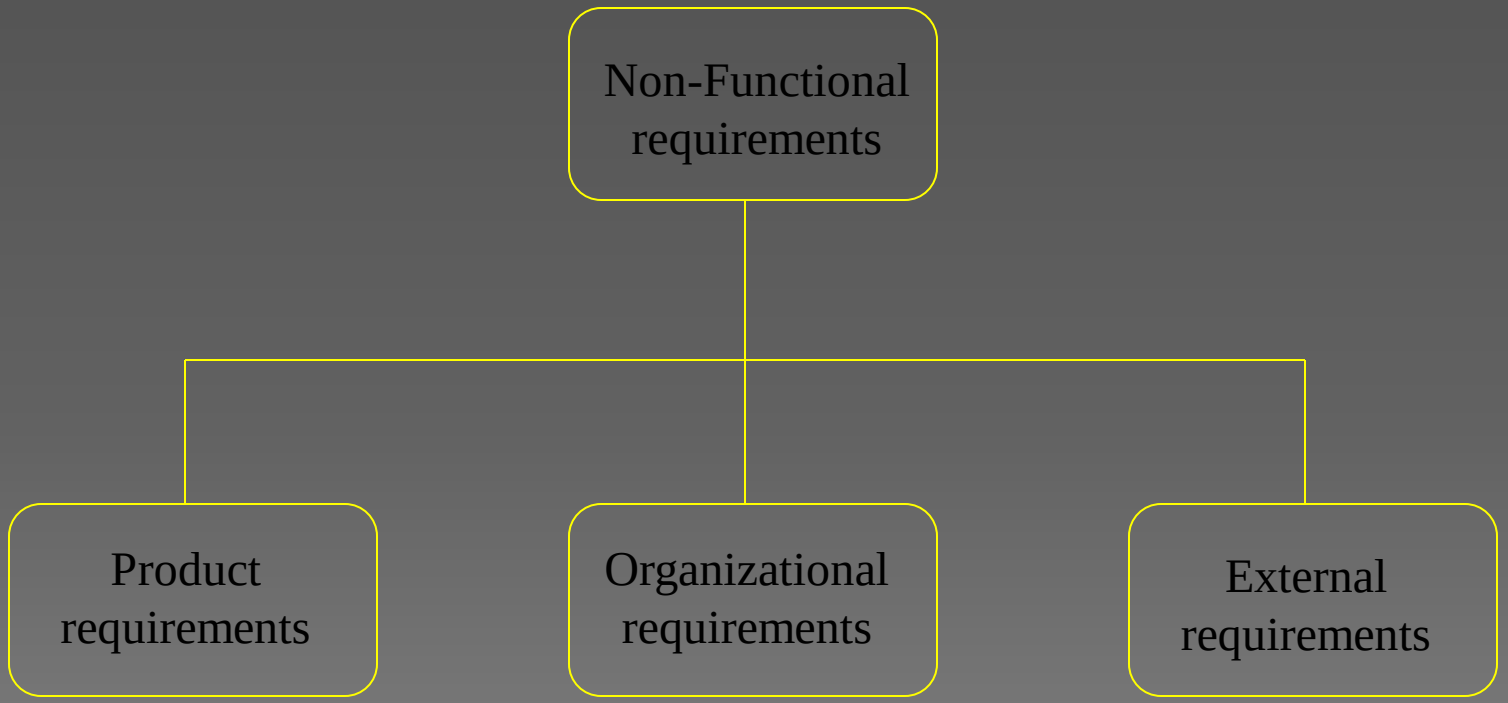
Non-Functional Requirements - 1

- Most non-functional requirements relate to the system as a whole. They include constraints on timing, performance, reliability, security, maintainability, accuracy, the development process, standards, etc., emergent behavior
- Often more critical than individual functional requirements

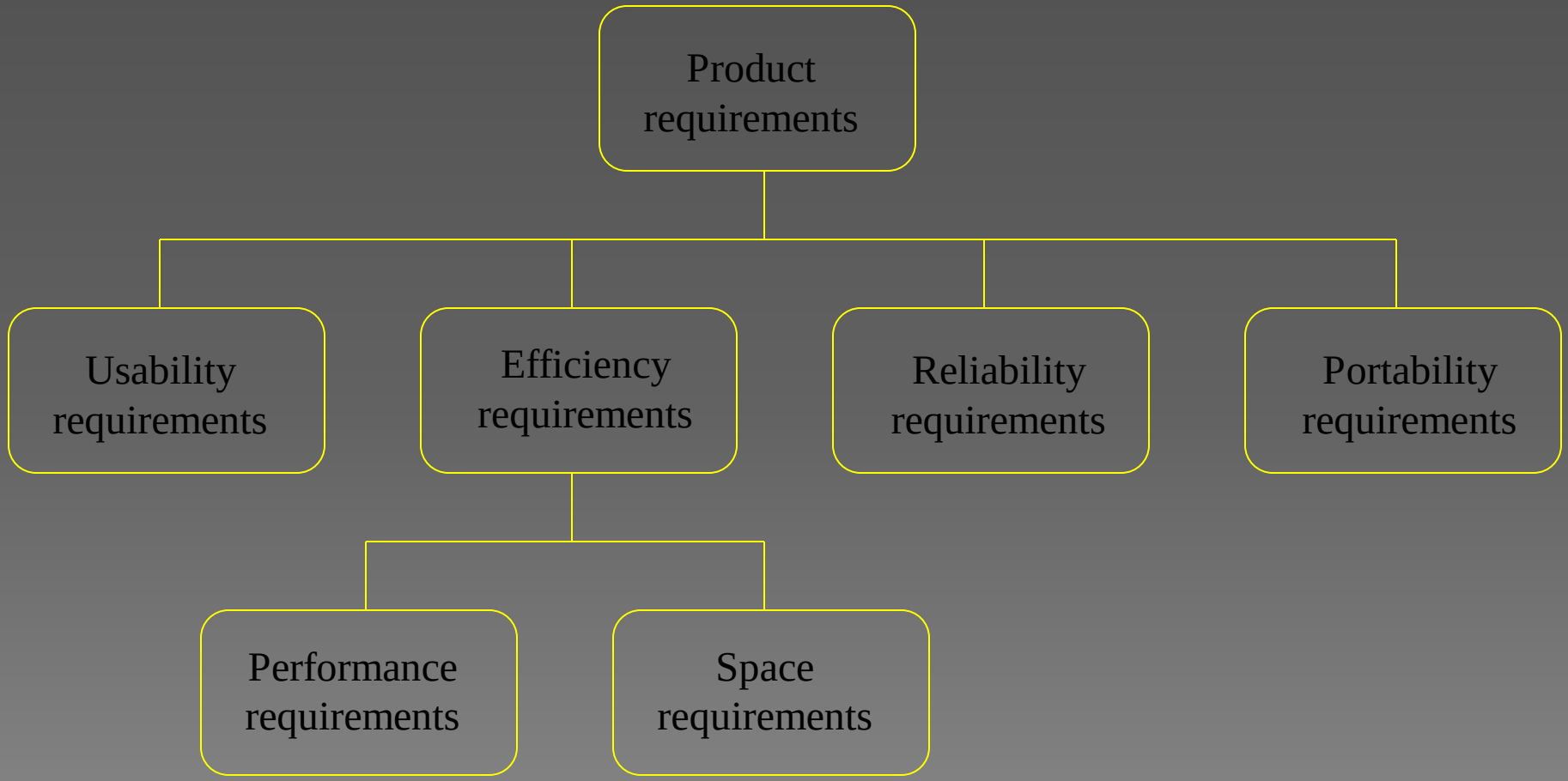
Non-Functional Requirements - 2

- Must be built into the framework of the software product
- Failure to meet a non-functional system requirement may make the whole system unusable

Types of Non-Functional Requirements



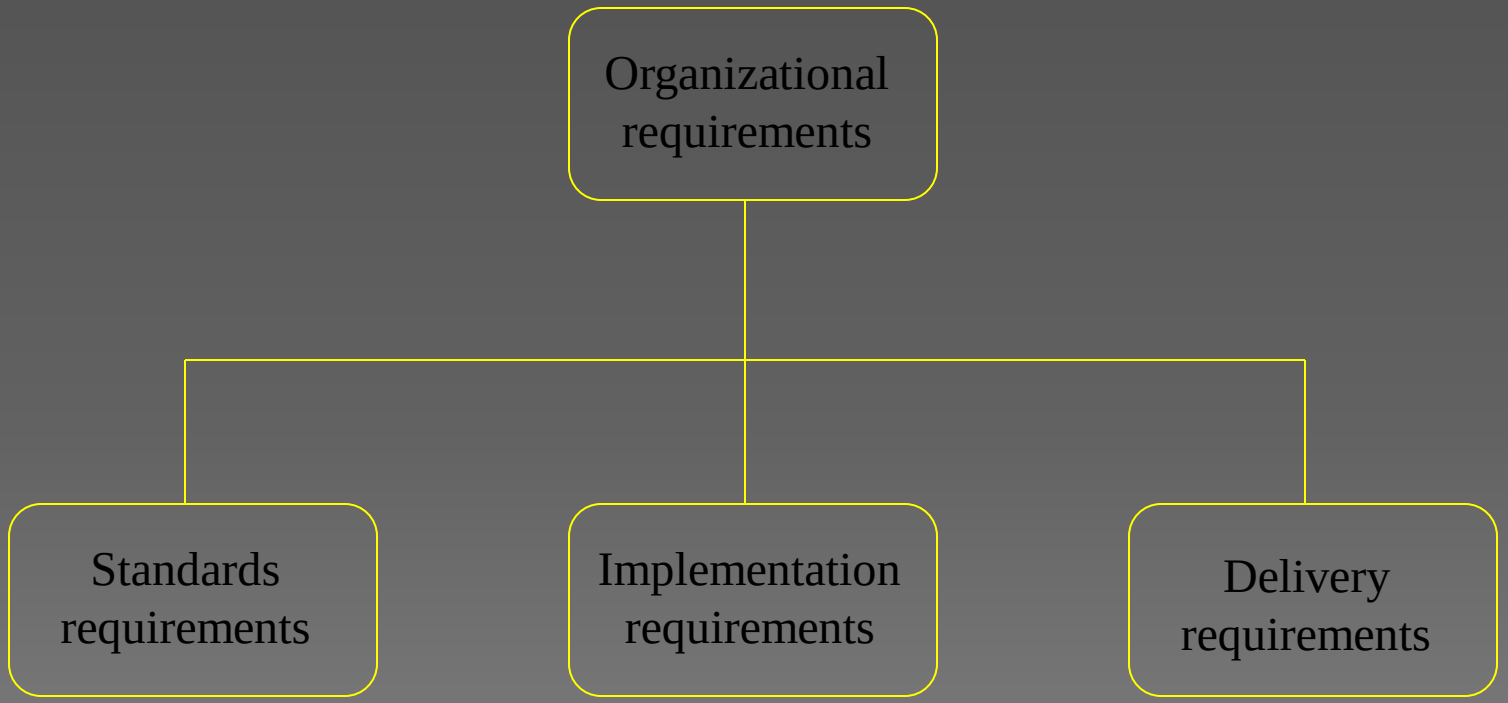
Product Requirements - 1



Product Requirements - 2

- The system shall allow one hundred thousand hits per minute on the website
- The system shall not have down time of more than one second for continuous execution of one thousand hours

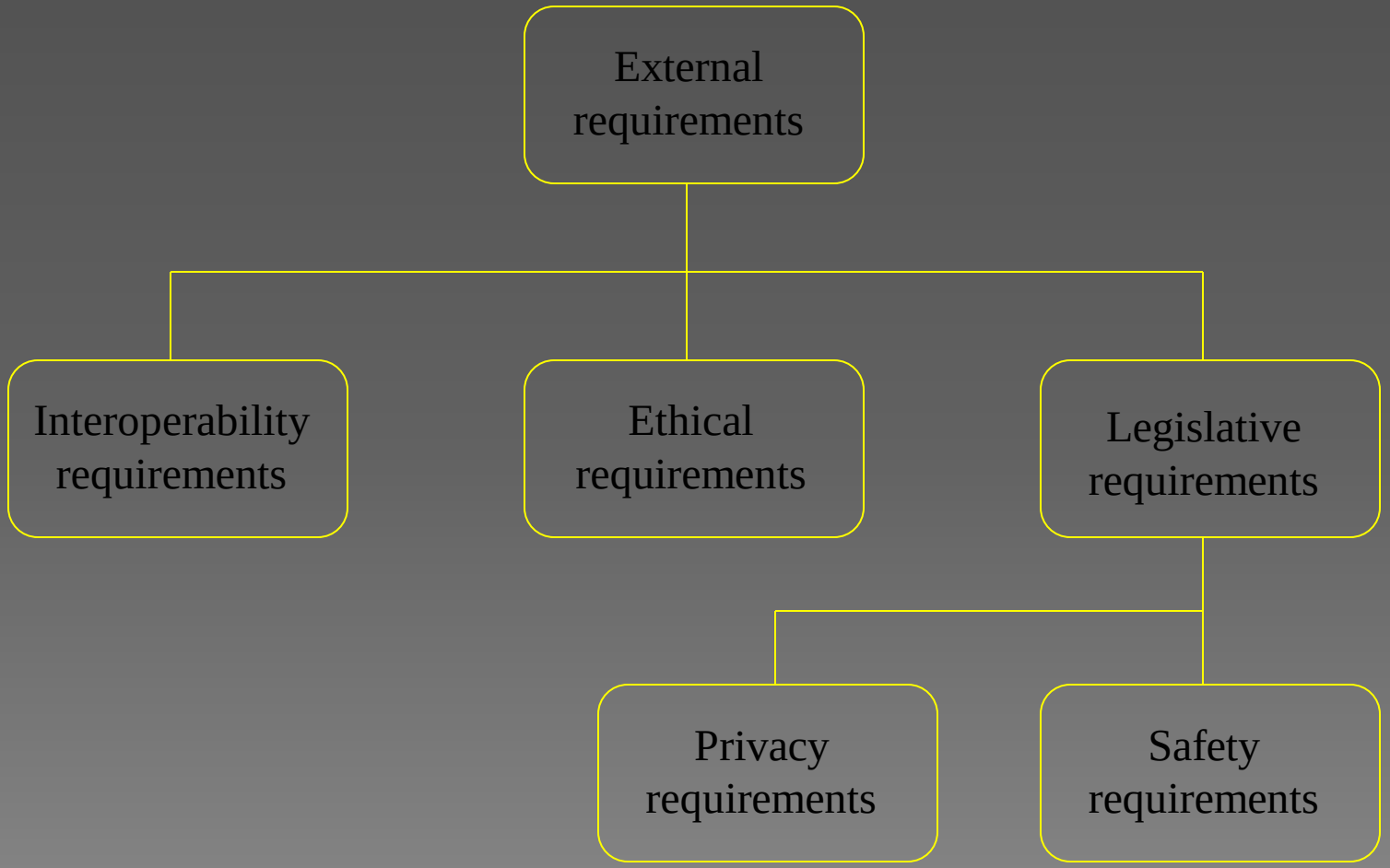
Organizational Requirements - 1



Organizational Requirements - 2

- The system development process and deliverable documents shall conform to the MIL-STD-2167A
- Any development work sub-contracted by the development organization shall be carried out in accordance with Capability Maturity Model

External Requirements - 1



External Requirements - 2

- The system shall not disclose any personal information about members of the library system to other members except system administrators
- The system shall comply with the local and national laws regarding the use of software tools

Metrics for Non-Functional Requirements

- Speed
- Size
- Ease of use
- Reliability
- Robustness
- Portability

Domain Requirements - 1

- Requirements that come from the application domain and reflect fundamental characteristics of that application domain. Can be functional or non-functional
- These requirements, sometimes, are not explicitly mentioned, as domain experts find it difficult to convey domain requirements

Domain Requirements - 2

- Their absence can cause significant dissatisfaction
- Domain requirements can impose strict constraints on solutions. This is particularly true for scientific and engineering domains

Inverse Requirements

- They explain what the system shall **not** do. Many people find it convenient to describe their needs in this manner
- These requirements indicate the indecisive nature of customers about certain aspects of a new software product

Design and Implementation Constraints

- They are development guidelines within which the designer must work
- These requirements can seriously limit design and implementation options
- Can also have impact on human resources

Requirements Problems

- The requirements don't reflect the real needs of the customer for the system
- Requirements are inconsistent and/or incomplete
- There are misunderstandings between customers, those developing the system requirements, and software engineers developing or maintaining the system

Problems with Natural Languages - 1

- Lack of clarity
- Requirements confusion
- Requirements amalgamation

Problems with Natural Languages - 2

- Natural language understanding relies on the specification readers and writers using the same words for same concept
- A natural language requirements specification is over-flexible. You can say the same thing in completely different ways

Problems with Natural Languages - 3

- It is not possible to modularize natural language requirements. It may be difficult to find all related requirements
  - To discover the impact of a change, every requirement have to be examined

Process - 1

- A process is an organized set of activities, which transforms inputs to outputs
- Synonyms: procedure, method, course of action, etc.
- Processes are essential for dealing with complexity in real world

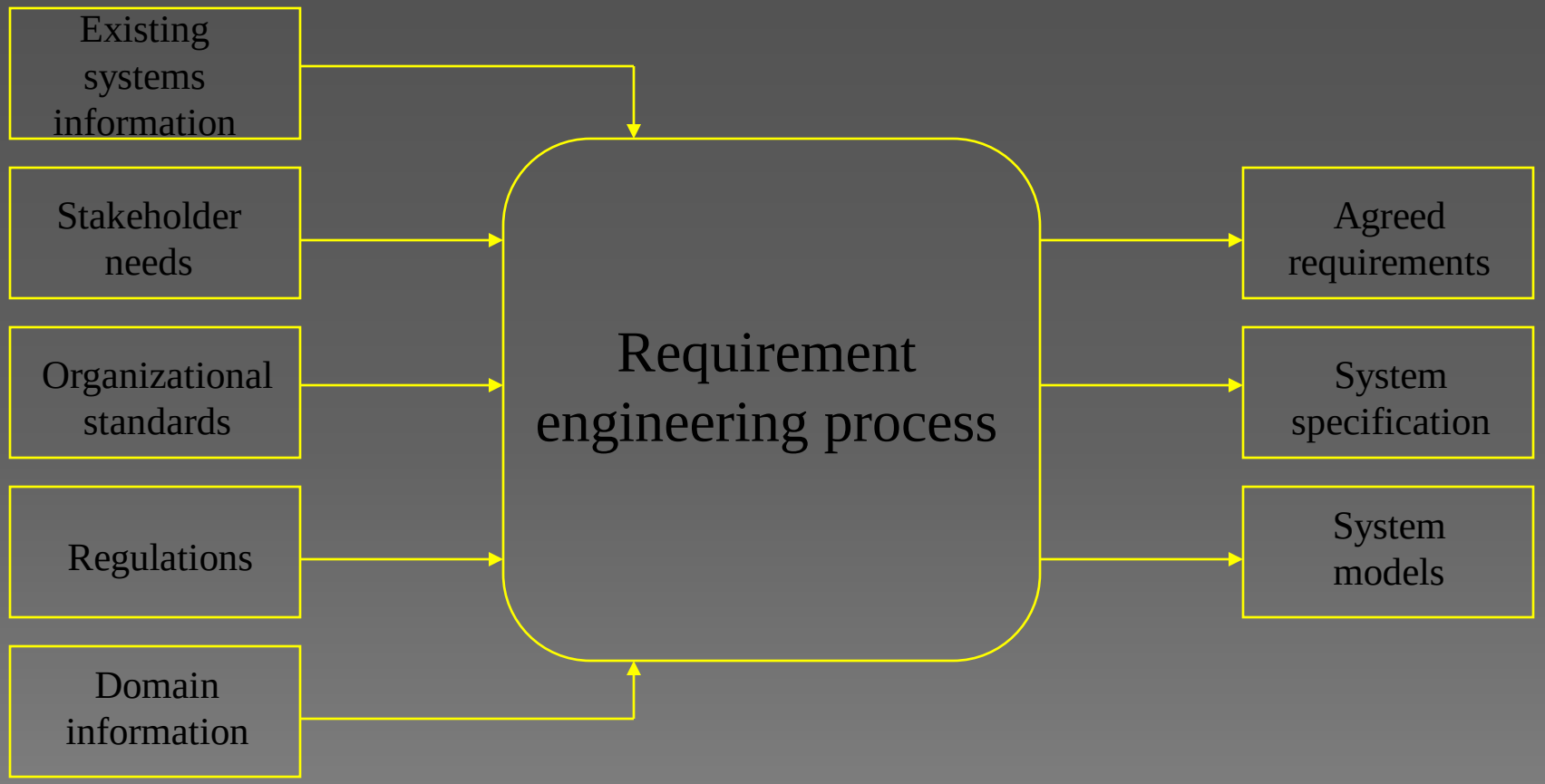
Process - 2

- Processes document the steps in solving a certain problem
- They allow knowledge to be reused
- Allows people to apply the process in their peculiar but similar problems

Requirements Engineering Process

The process(es) involved in developing system requirements

RE Process - Inputs and Outputs



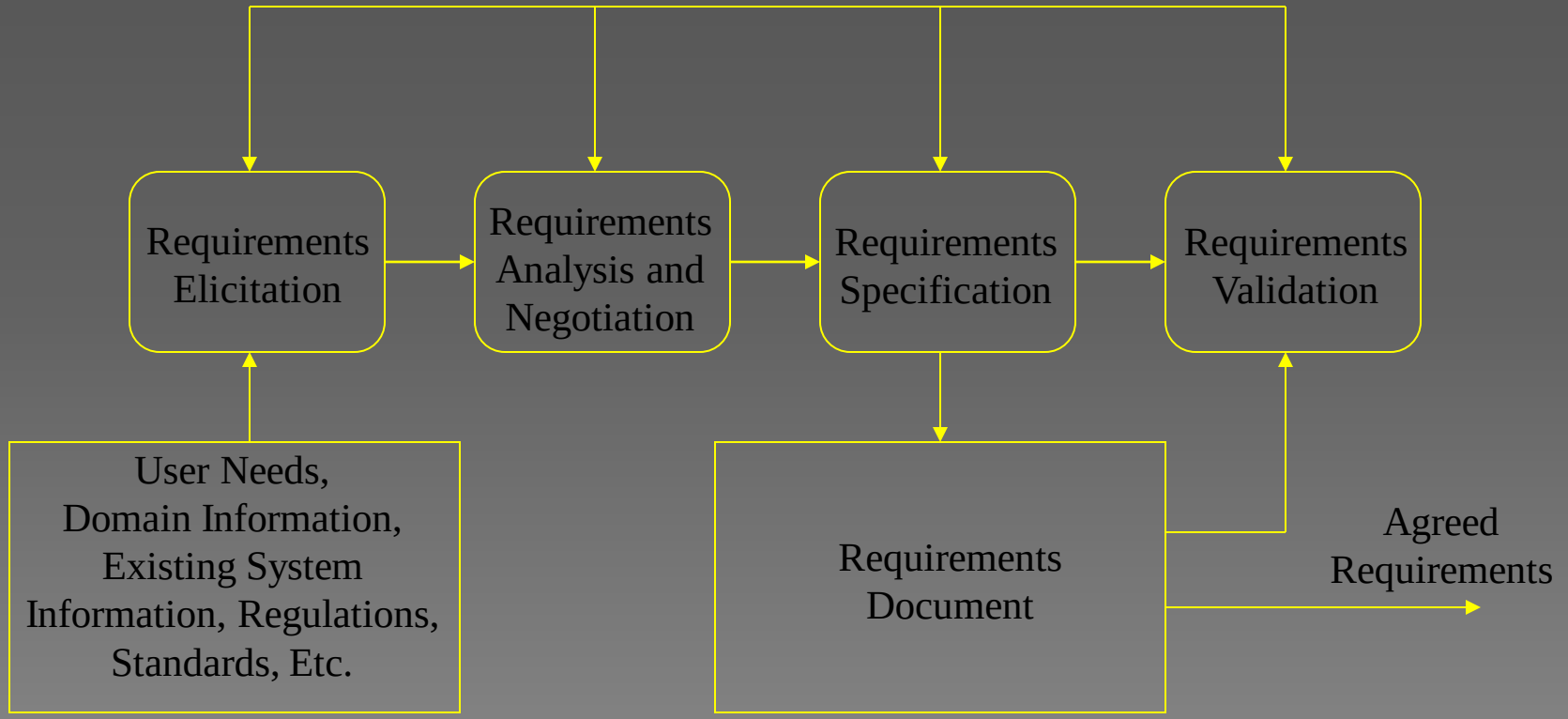
RE Process Variability

- RE processes vary radically from one organization to another, and even within an organization
- Unstructured process rely heavily on the experience of the people, while systematic processes are based on application of some analysis methodology , but still require human judgment

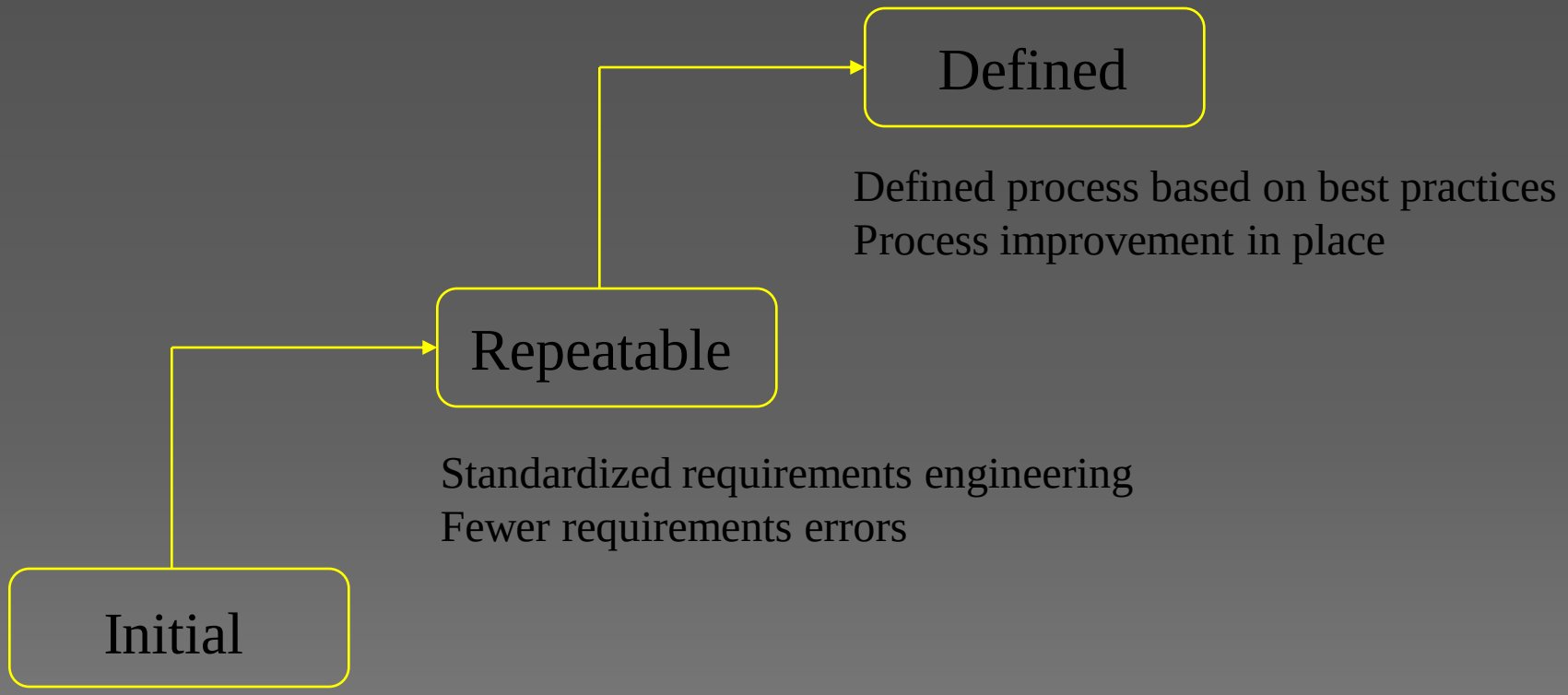
Variability Factors - 1

- Technical maturity
- Disciplinary involvement
- Organizational culture
- Application domain

Requirements Engineering Activities



RE Process Maturity Model



Social and Cultural Issues in RE

- Some aspects of the requirements engineering process deal with social and cultural issues
- What is the best way to deal with these issues?

Six Areas of Social Issues - 1

- Within the client organization
- Within the requirements team
- Between the client and the requirements team

Six Areas of Social Issues - 2

- Between the development and requirements teams
- Within the development team
- Between the development team and the client

Cultural Issues in RE

- Time zones differences
- Language and terminology differences
- Religious and racial differences
- Ethical issues
- Political differences
- Differences in business environment

Basics of Knowledge Acquisition

- Reading
- Listening
- Asking
- Observing

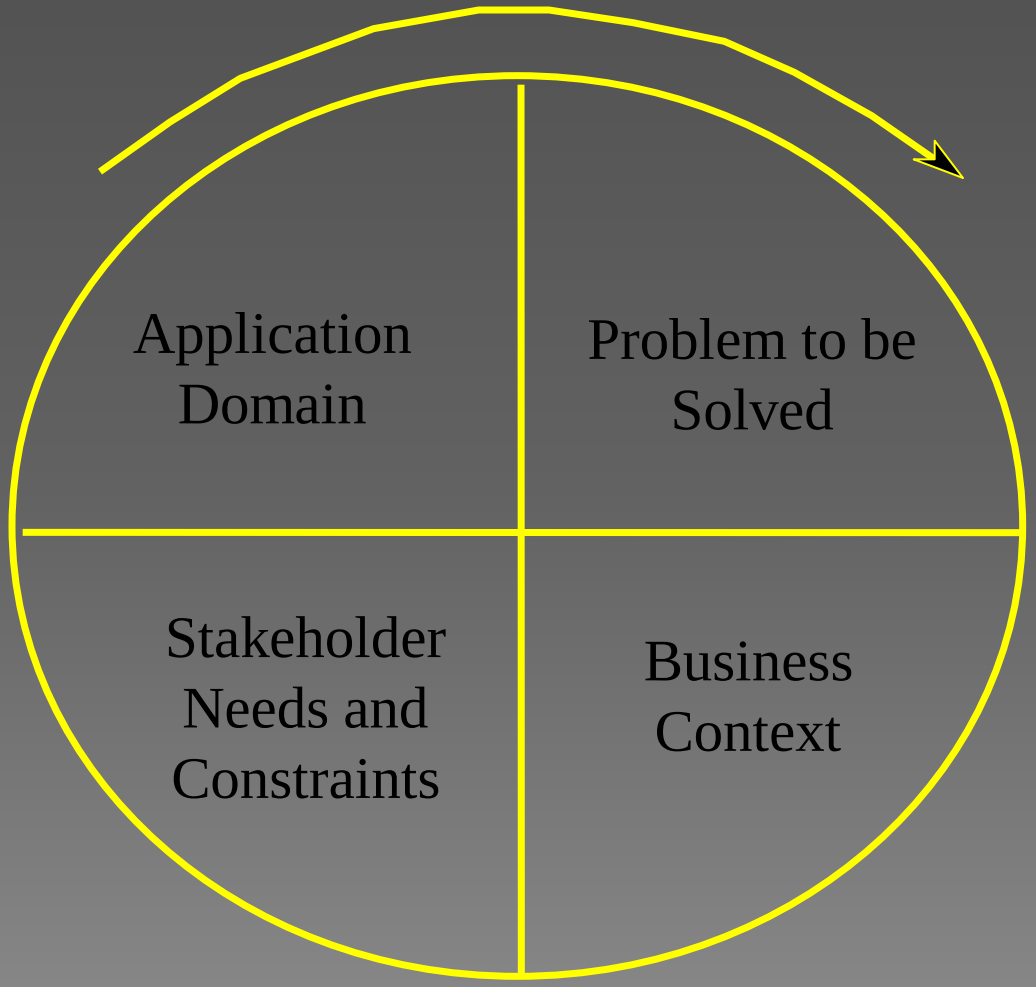
Requirements Elicitation Techniques

- Individual
- Group
- Modeling
- Cognitive

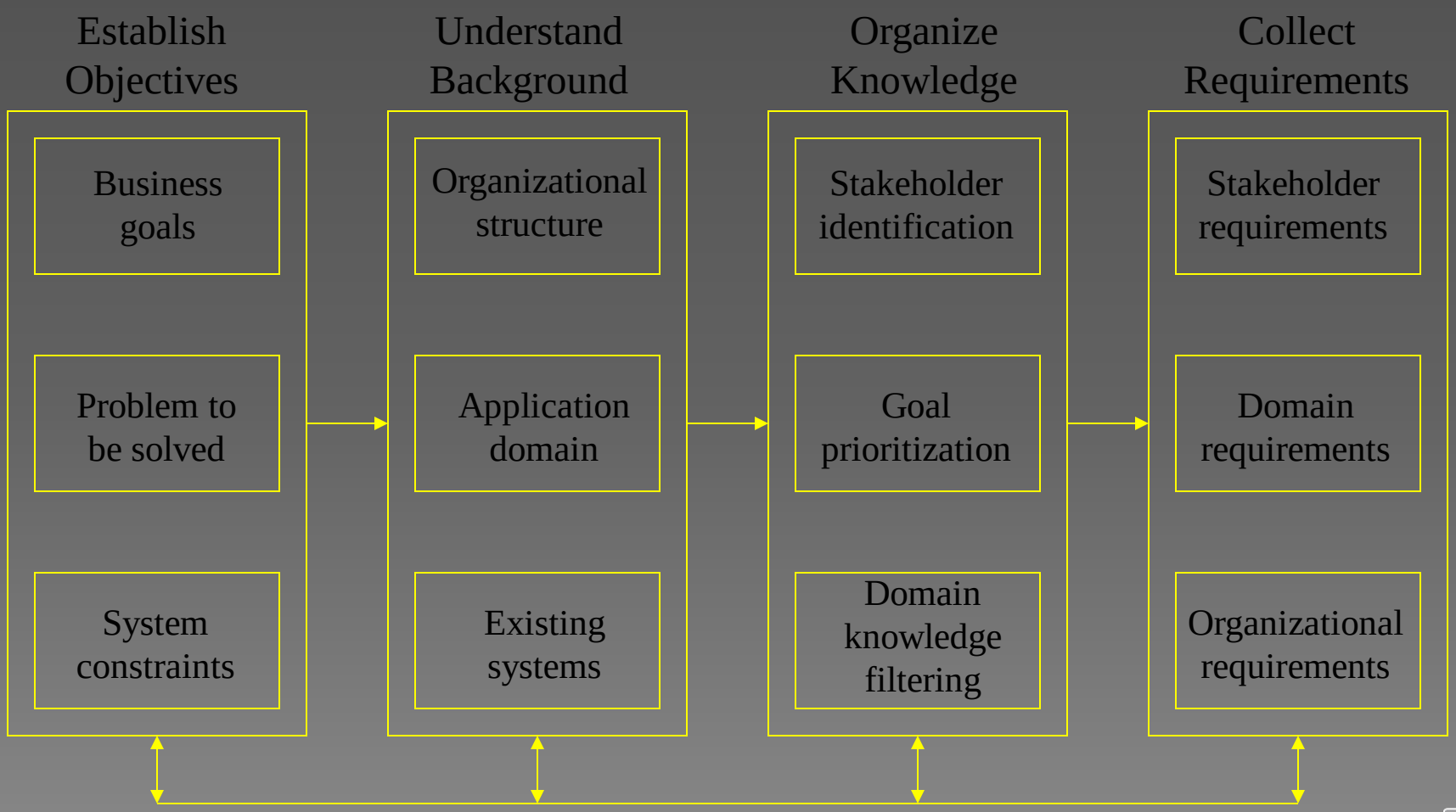
Problems in Requirements Elicitation

- Problems of scope
- Problems of understanding
- Problems of volatility

Components of Requirements Elicitation



A General Requirements Elicitation Process



Knowledge Structuring Techniques

- Partitioning
- Abstraction
- Projection

Specific Elicitation Techniques

- Interviews
- Scenarios
- Observations and social analysis
- Requirements reuse

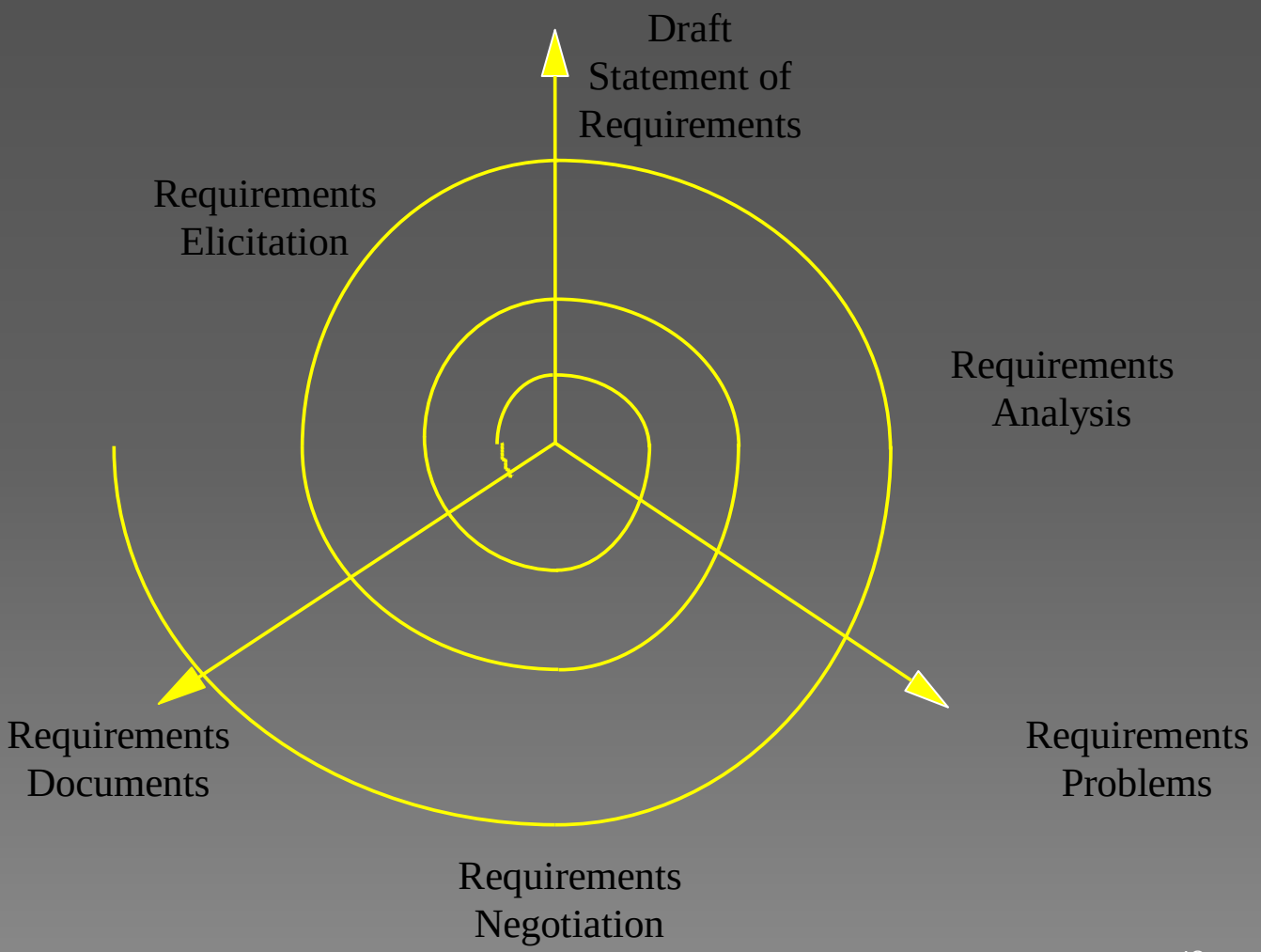
Interview Steps

- Prepare
- Conduct
  - Opening
  - Body
  - Closing
- Follow through

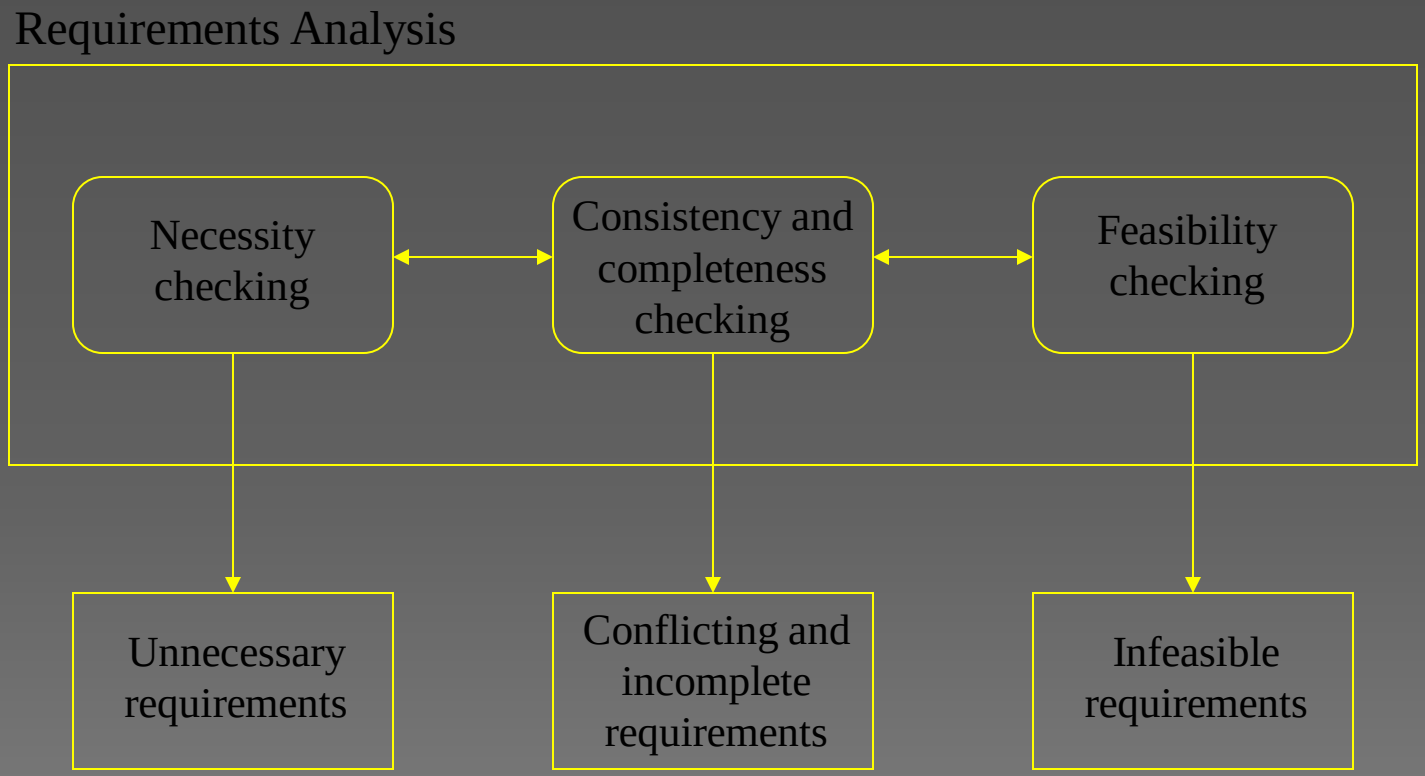
Listening Steps

- Hear
- Interpret
- Respond
- Evaluate

Iterative Aspects of Elicitation, Analysis, and Negotiation



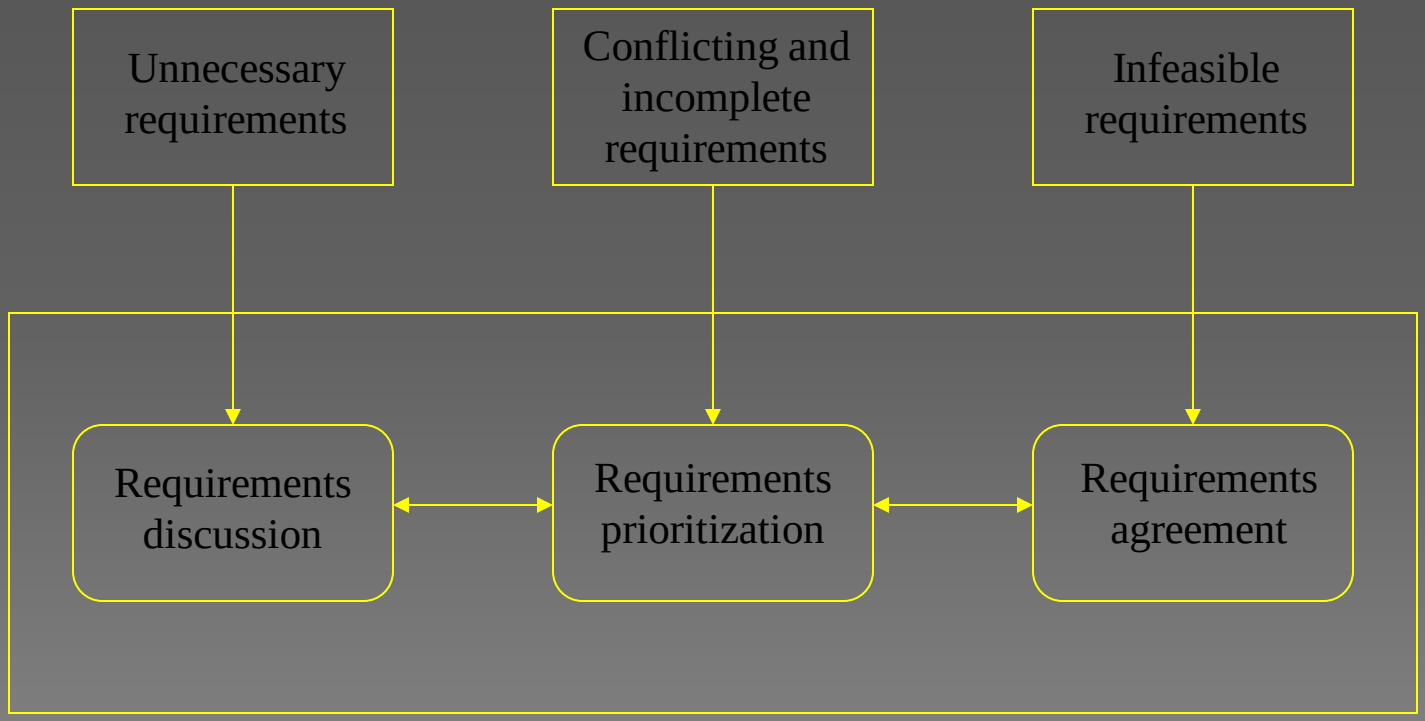
Requirements Analysis Process



Analysis Techniques

- Analysis checklists
  - A checklist is a list of questions which analysts may use to assess each requirement
- Interaction matrices
  - Interaction matrices are used to discover interactions between requirements and to highlight conflicts and overlaps

Requirements Negotiation Process



Requirements Negotiation

Stages of Negotiation Meetings

- Information stage
- Discussion stage
- Resolution stage

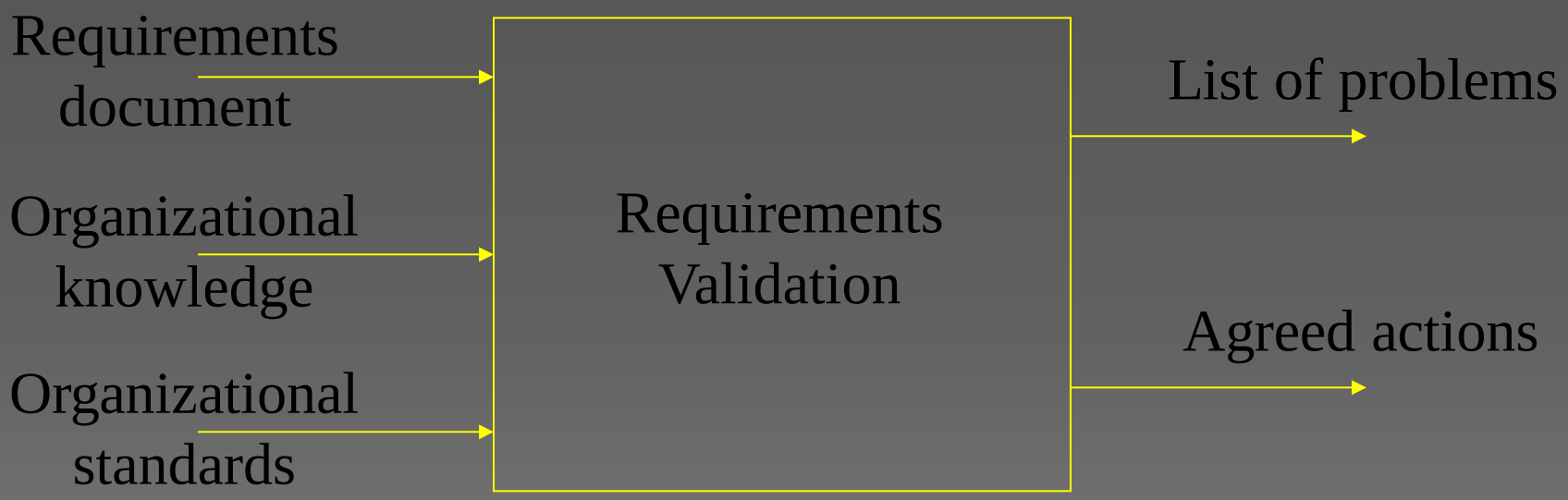
Types of Requirements Errors

- Errors of omission
- Errors of commission
- Errors of clarity and ambiguity
- Errors of speed and capacity

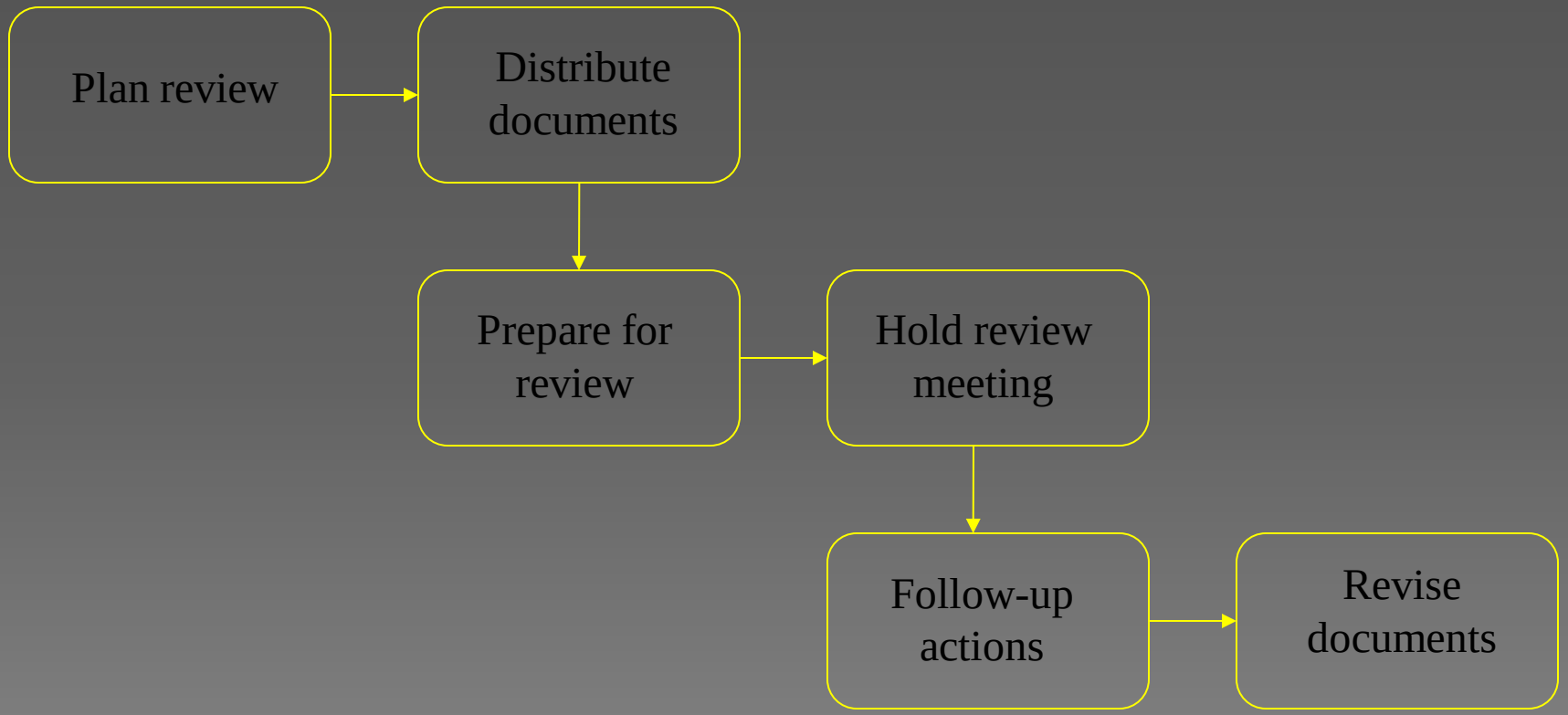
Prevention vs. Removal

- For requirements errors, prevention is usually more effective than removal
- Joint application development (JAD), quality function deployment (QFD), and prototyping are more effective in defect prevention
- Requirements inspections and prototyping play an important role in defect removal. Discussed in detail perspective-based reading technique

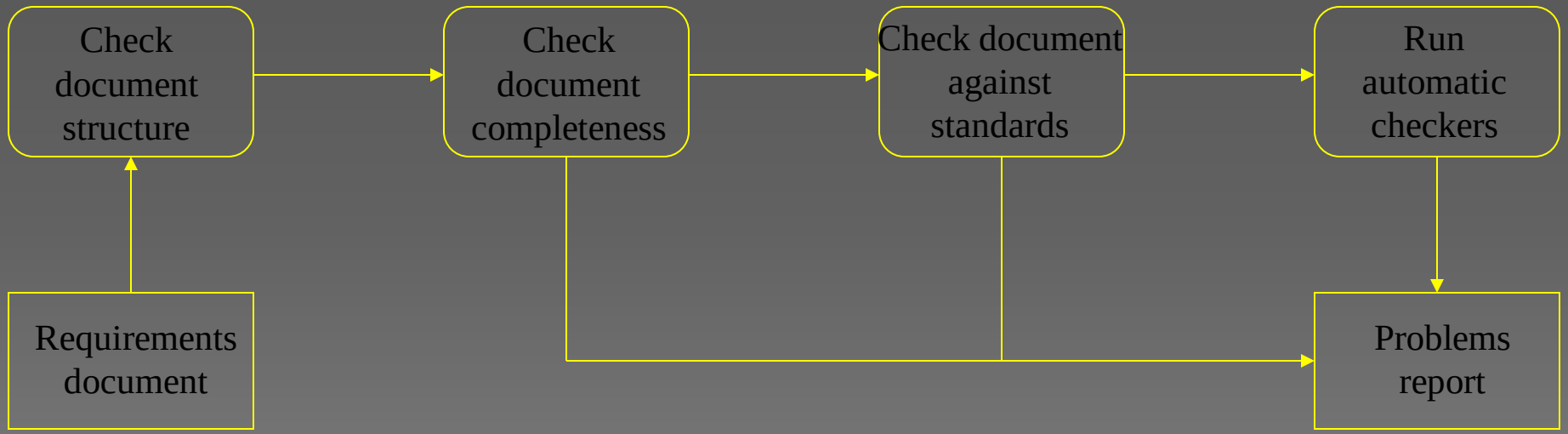
Validation Inputs and Outputs



Requirements Review Process



Pre-review Checking Stages



Hard-to-Test Requirements

- System requirements
- Exclusive requirements
- Some non-functional requirements

Requirements Management

- The process of managing change to the requirements for a system
- In this lecture, we'll talk about the reasons for changes in requirements and how to manage them

Sources of Change - 1

- New business or market conditions dictate changes in product requirements or business rules
- New customer needs demand modification of data produced by information systems, functionality delivered by products, or services delivered by computer-based system

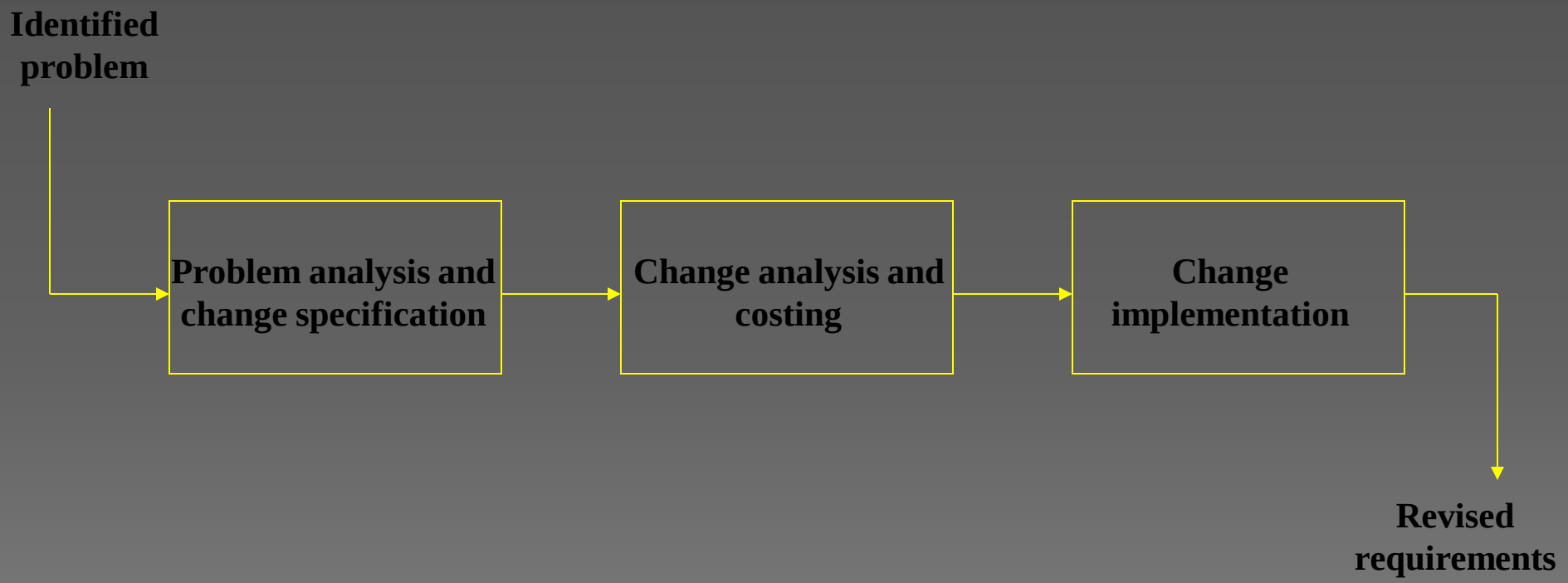
Sources of Change - 2

- Reorganization or business growth/downsizing causes changes in project priorities or software engineering team structure
- Budgetary or scheduling constraints cause a redefinition of the system or product

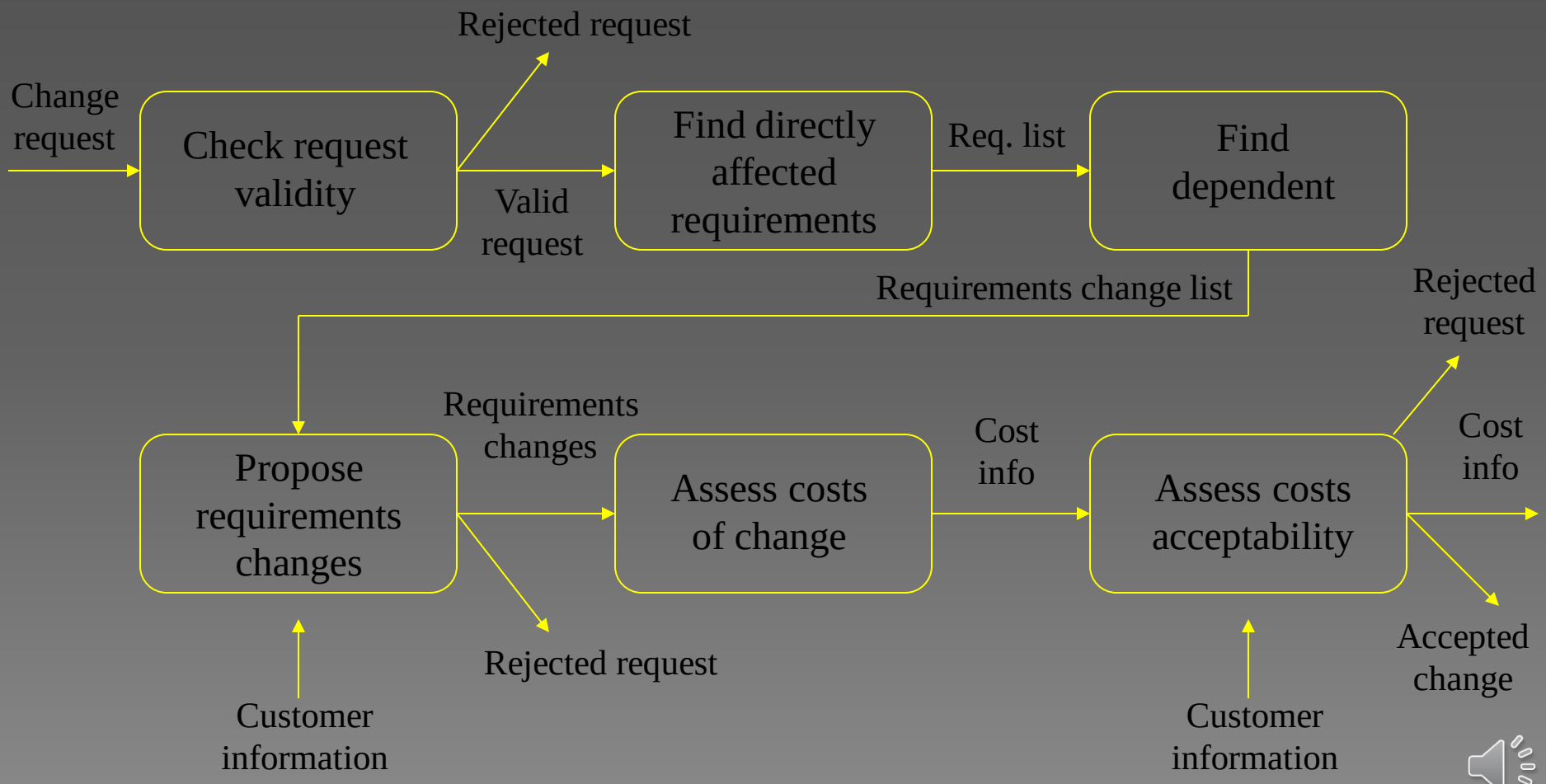
Main Concerns in Requirements Management

- Managing changes to agreed requirements
- Managing the relationships between requirements
- Managing the dependencies between the requirements document and other documents produced in the systems engineering process

Change Management Stages



Change Analysis and Costing Process



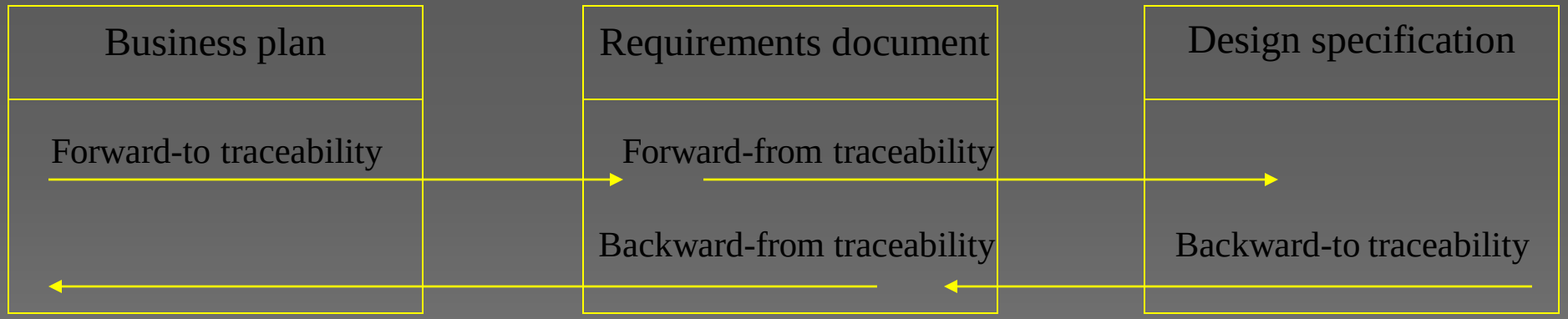
Requirements Traceability

- Refers to ability to describe and follow the life of a requirement, in both a forwards and backwards direction
- That is from its origins, through its development and specification, to its subsequent deployment and use, and through all periods of on-going refinement and iteration in any of these phases

Classifications of Requirements Traceability

- Backward-from traceability
- Forward-from traceability
- Backward-to traceability
- Forward-to traceability

Backwards and Forwards Traceability



Categories of Traceability

- Requirements-sources traceability
- Requirements-rationale traceability
- Requirements-requirements traceability
- Requirements-architecture traceability
- Requirements-design traceability
- Requirements-interface traceability

Types of Prototyping

- Throw-away prototyping
- Evolutionary prototyping

Approaches to Prototyping

- Paper prototyping
- 'Wizard of Oz' prototyping
- Executable prototyping