SOFTWARE ENGINEERING

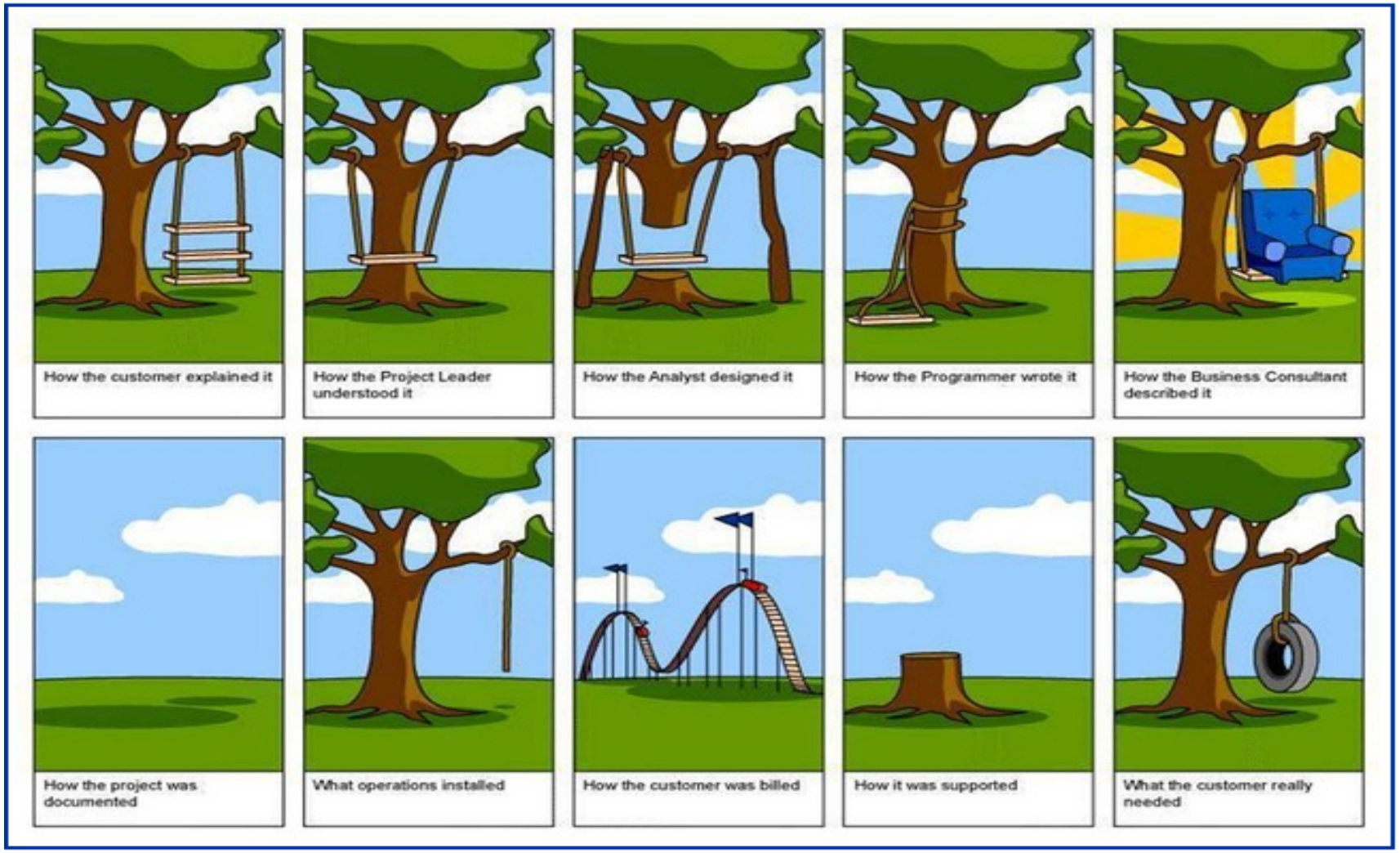
CHAPTER-5
UNDERSTANDING REQUIREMENTS

Software Engineering: A Practitioner's Approach, 7th edition

Originated by Roger S. Pressman

Software School, Fudan University
Spring Semester, 2016

Where is The Problem



Requirements!!!!

Requirements Engineering

It's your worst nightmare. A customer walks into your office, sits down, looks you straight in the eye, and says, "I know you think you understand what I said, but what you don't understand is what I said is not what I meant." Invariably, this happens late in the project, after deadline commitments have been made, reputations are on the line, and serious money is at stake.

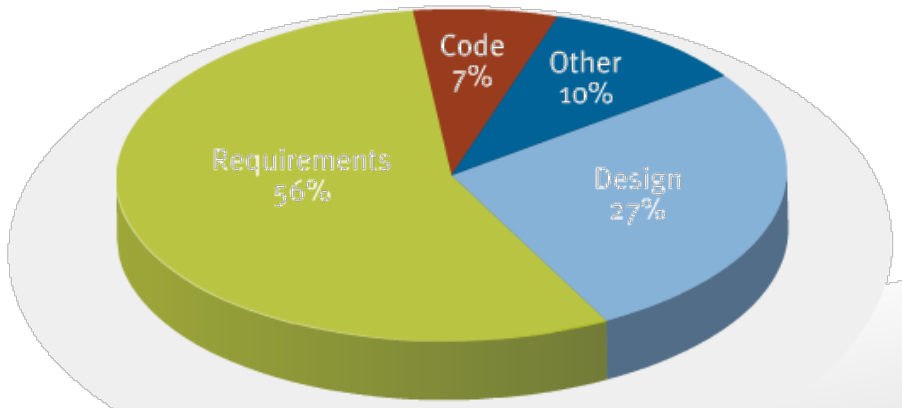
- Ralph Young

Requirements Errors

Requirements Errors

40% of effort in an average software project is rework

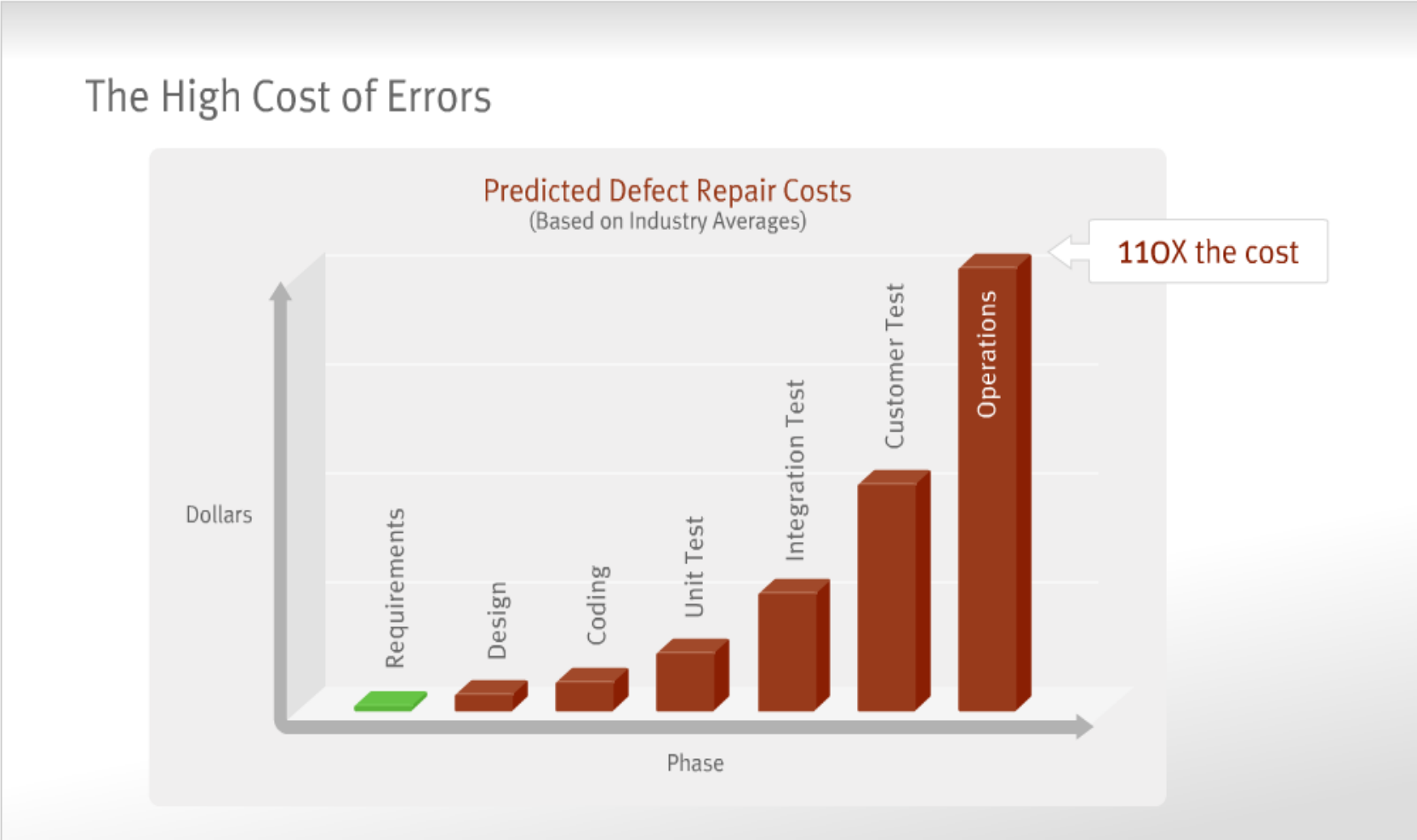
The largest contributor is requirements errors



Distribution of Defects

Butler Group, April 2005

Accumulation Effect of Requirements Errors



Requirements Defects — LAS: London Ambulance System

- System context
  - 网络环境: 未考虑到无线电通信盲点问题
- Stakeholder
  - 救护人员: 未充分让其参与需求工程过程—导致救护车上的系统操作接口上考虑的不足
- Scenario
  - 救护: 未考虑到救护人员情况临时更换救护车的情况
- Specification
  - 模糊性: 在最短的时间内将调度指令发送到救护车
  - 不一致: 调度最近的救护车以节省时间 Vs. 病人可以选择不同档次的救护车

Requirements Quality

- High-quality requirements should be
  - Correct
  - Complete
  - Unambiguous
  - Consistent
  - Feasible
  - Testable
  - …

Assess the Quality of the Following Requirements:

- 输入关键字后，系统应当快速返回查询结果
- 对于超过限制数量的图书借阅请求，正常情况下系统都应当拒绝
- 读取读者一卡通以及所借书籍条码后，显示其详细信息
- 系统必须永远不出错
- 借阅图书的师生需要在借阅期限内还书。对于超期不还的用户要按日收取罚款。

Requirements Engineering

- Tasks of Requirement Phase
  - Understand what the exact problem is
  - Specify the requirements with models and documents to enable following design and implementation

    Across communication, plan and modeling
- Requirement Engineering: conduct and manage the acquirement, analysis, specification and evolution of software requirements with an engineering process

Requirements Engineering

- Requirement Engineering provides the appropriate mechanism for
  - Understanding what the customer wants
  - Analyzing need
  - Assessing feasibility
  - Negotiating a reasonable solution
  - Specifying the solution unambiguously
  - Validating the specification
  - Managing the requirements as they are transformed into an operation system

The Goals of the Requirements Phase-1

- Decide what to build before starting to build it:
  - Make "what decisions" explicitly before design, not implicitly during design
  - Make sure you build what is needed
  - Allow future users, or representatives, to comment before the product is built – early feedback!!
- Provide an organized reference document for the software
  - Provide accurate, consistent information
  - Answer constraint questions only once
  - Relieve them of any need to decide what is best for the user
  - Compensate for their ignorance of the application environment
  - Give them the information needed to make good design decisions
  - Allow to make accurate estimates of time and resources needed

David Lorge Parnas

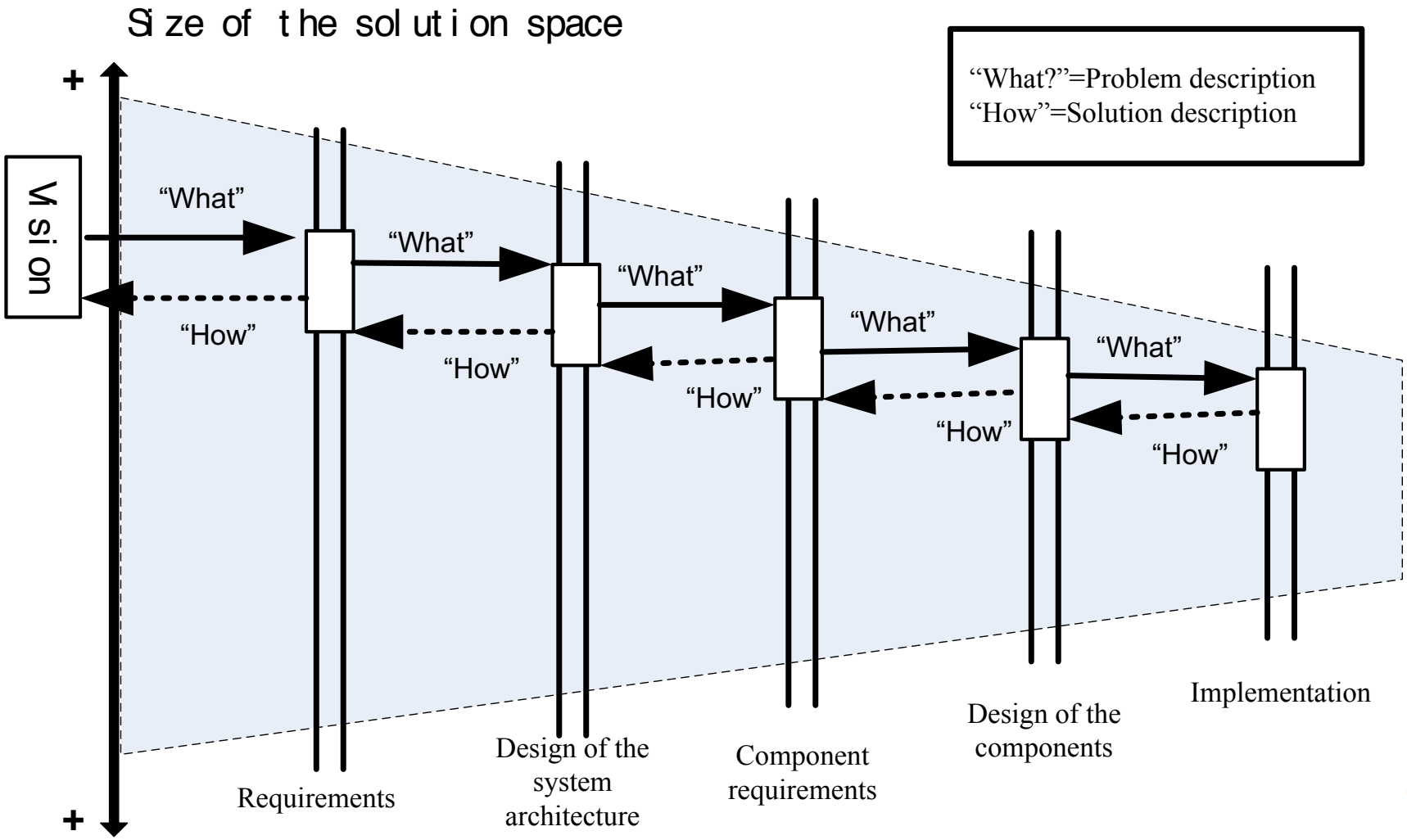
The Goals of the Requirements Phase-2

- Allow for personnel turnover
  - Record what has been learned for future replacements
- Provide a reference document for a Quality Assurance Group
  - Test design should not depend on program
  - Authority required:---Q.A. and programmer may disagree
- Specify all constraints on the implementation
  - Know what you're up against
  - Have some "protection" against customer changes
  - Be able to judge feasibility and cost.
- Specify constraints for future revisions

Note: A requirements document is not a "write, use (at most) once and discard" document. A good document, properly maintained, will be useful until the product is discarded and possibly for the next product.

David Lorge Parnas

"WHAT" AND "HOW"


Size of the solution space
"What?"=Problem description
"How"=Solution description
Vision
+
+
"What"
"How"
"What"
"How"
"What"
"How"
"What"
"How"
"What"
"How"
Requirements
Design of the system architecture
Component requirements
Design of the components
Implementation

"WHAT" AND "HOW": EXAMPLE

- What: （一卡通系统）多个校区之间实现互联互通
  - Solution 1: 其他校区一卡通办公室工作人员每天下班后导出数据通过邮件发送到主校区进行导入同步
  - Solution 2: 各个校区之间使用专线直连，使用统一的服务器
- What: 系统登录时验证用户身份
  - Solution 1: 输入用户名、密码
  - Solution 2: 用户插入随身携带的USB Key，然后输入软口令登录

Requirements Engineering-1

- Inception—ask a set of questions that establish …
  - basic understanding of the problem
  - the people who want a solution
  - the nature of the solution that is desired, and
  - the effectiveness of preliminary communication and collaboration between the customer and the developer
- Elicitation—elicit requirements from all stakeholders
- Elaboration—create an analysis model that identifies data, function and behavioral requirements
- Negotiation—agree on a deliverable system that is realistic for developers and customers

Requirements Engineering-2

- Specification—can be any one (or more) of the following:
  - A written document
  - A set of models
  - A formal mathematical
  - A collection of user scenarios (use-cases)
  - A prototype
- Validation—a review mechanism that looks for
  - errors in content or interpretation
  - areas where clarification may be required
  - missing information
  - inconsistencies (a major problem when large products or systems are engineered)
  - conflicting or unrealistic (unachievable) requirements.
- Requirements management

RE: Reality

- Ideal setting
  - Stakeholders work together with engineers
  - Customers have clear thought about the desired system and can clearly express it
  - RE activities: conducting meaningful conversations with colleagues
- Reality
  - customers or end users located in different places
  - may have only a vague idea of what required
  - may have conflicting opinions about the system
  - may have limited technical knowledge
  - may have limited time to interact
  - …

Getting Requirements Right

- "The hardest single part of building a software system is deciding what to build. No part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later."

  —Fred Brooks

- "The seeds of major software disasters are usually sown within the first three months of commencing the software project."

  —Capers Jones

- "We spend a lot of time—the majority of project effort—not implementing or testing, but trying to decide what to build."

  —Brian Lawrence

INCEPTION

- Identify stakeholders
  - "who else do you think I should talk to?"
- Recognize multiple points of view
- Work toward collaboration
- The first questions
  - Who is behind the request for this work?
  - Who will use the solution?
  - What will be the economic benefit of a successful solution
  - Is there another source for the solution that you need?

STAKEHOLDER

- Stakeholder: anyone who has a direct interest in or benefits from the system that is to be developed
- Identify Stakeholders
  - Typical stakeholders: business operations managers, product managers, marketing people, internal and external customers, end users, consultants, product engineers, software engineers, support and maintenance engineers, …
  - Each stakeholder has a different view of the system, achieves different benefits if success, is open to different risks if fail
- At inception, a initial stakeholder list should be created and grow with more communication

ELICITING REQUIREMENTS

- meetings are conducted and attended by both software engineers and customers
- rules for preparation and participation are established
- an agenda is suggested
- a "facilitator" (can be a customer, a developer, or an outsider) controls the meeting
- a "definition mechanism" (can be work sheets, flip charts, or wall stickers or an electronic bulletin board, chat room or virtual forum) is used
- the goal is
  - to identify the problem
  - propose elements of the solution
  - negotiate different approaches, and
  - specify a preliminary set of solution requirements

Eliciting Requirements-Difficulties

- Why is it so difficult to clearly understand what the customer wants?
  - Scope
    - The boundary of the system is ill-defined
      - boundaries between software behaviors and human behaviors
      - boundaries between targeted-system and other interacting systems
      - boundaries between hardware and software
      - boundaries between targeted software and DB/OS/middleware (and other underlying software)
      - ……
    - Customers/users specify unnecessary technical detail that may confuse rather than clarify objectives

Customer Specified Requirements: Example

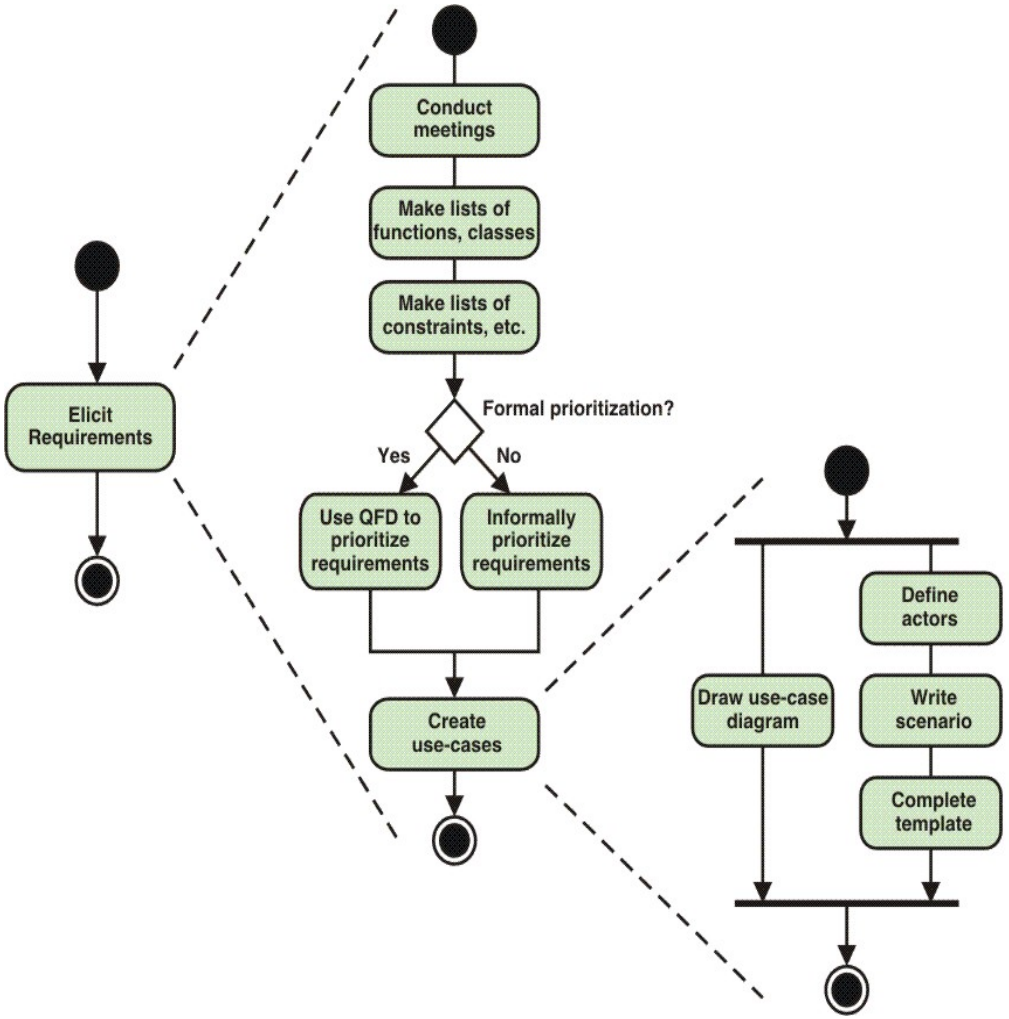
- 笔试考试计算机辅助阅卷系统
  - 客户表述的需求：通过手写识别技术自动识别并记录考生在答卷上的方格内所书写的考号
  - 分析客户的最终目的（why）：准确、自动地识别每一份答卷的考生身份
  - 建议的需求方案（how）
    - 方案1：通过铅笔涂黑的方式确定考号
    - 方案2：为每位考生发放一张与考号对应的条形码并要求其贴在考卷指定位置上

      在此基础上分析利弊（成本、准确性等）引导客户做出选择

Eliciting Requirements-Difficulties (Cont.)

- Understanding
  - Customers are not completely sure of what is needed
  - Customers have a poor understanding of the capabilities and limitations of the computing environment
  - Customers don't have a full understanding of their problem domain
  - Customers have trouble communicating needs to the system engineer
  - Customers omit detail that is believed to be obvious
  - Customers specify requirements that conflict with other requirements
  - Customers specify requirements that are ambiguous or untestable
- Volatility
  - Requirements change over time

Eliciting Requirements



Quality Function Deployment

- Function deployment determines the "value" (as perceived by the customer) of each function required of the system
- Information deployment identifies data objects and events
- Task deployment examines the behavior of the system
- Value analysis determines the relative priority of requirements

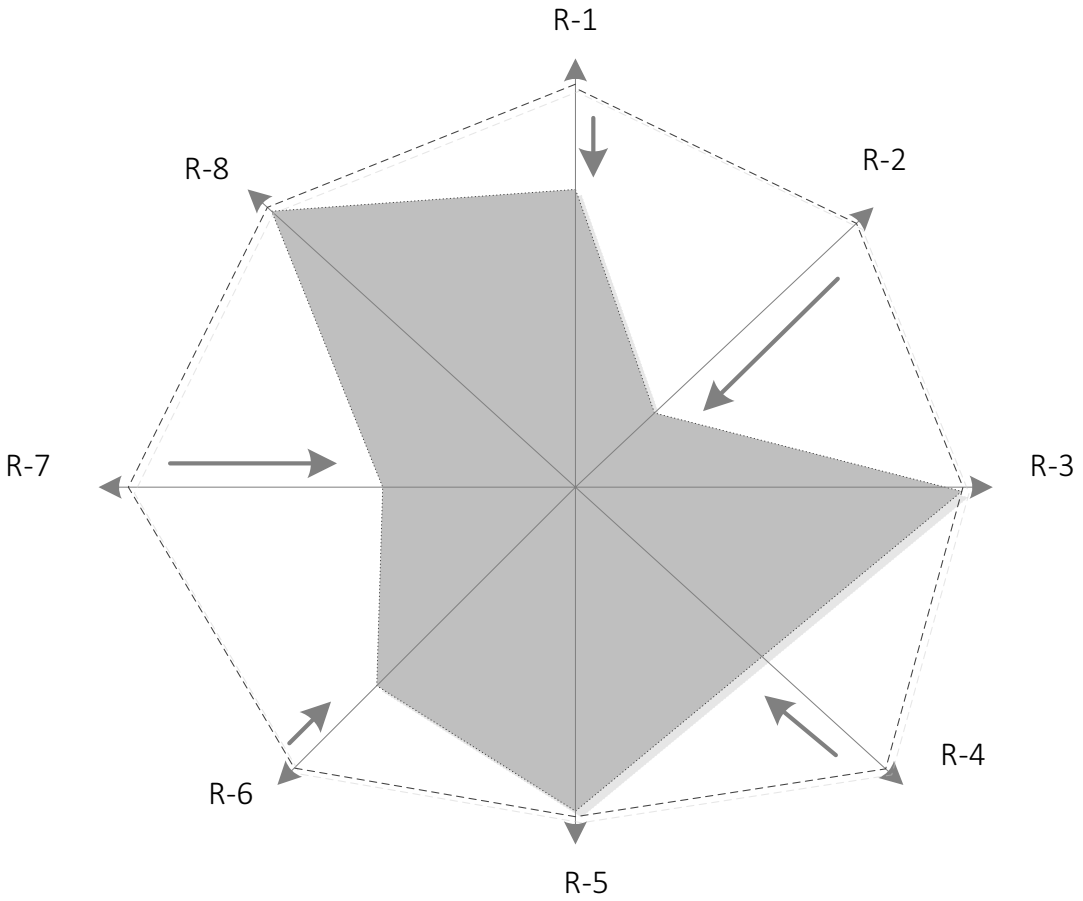
Quality Function Deployment (Cont.)

- QFD spans the entire engineering process, and many QFD concepts are applicable to the requirements elicitation activity
  - Function deployment determines the "value" (to the customer) of each function required of the system
  - Information deployment identifies data objects and events
  - Task deployment examines the behavior of the system
  - Value analysis determines the priority of requirements

Elicitation Work Products

- a statement of need and feasibility.
- a bounded statement of scope for the system or product.
- a list of customers, users, and other stakeholders who participated in requirements elicitation
- a description of the system's technical environment.
- a list of requirements (preferably organized by function) and the domain constraints that apply to each.
- a set of usage scenarios that provide insight into the use of the system or product under different operating conditions.
- any prototypes developed to better define requirements.

CONSTRAINTS



Range of realisation alternatives for requirements **without** considering constraints

Range of possible realisation alternatives for requirements **with** the consideration of constraints

Restricting effect of constraints on a requirement

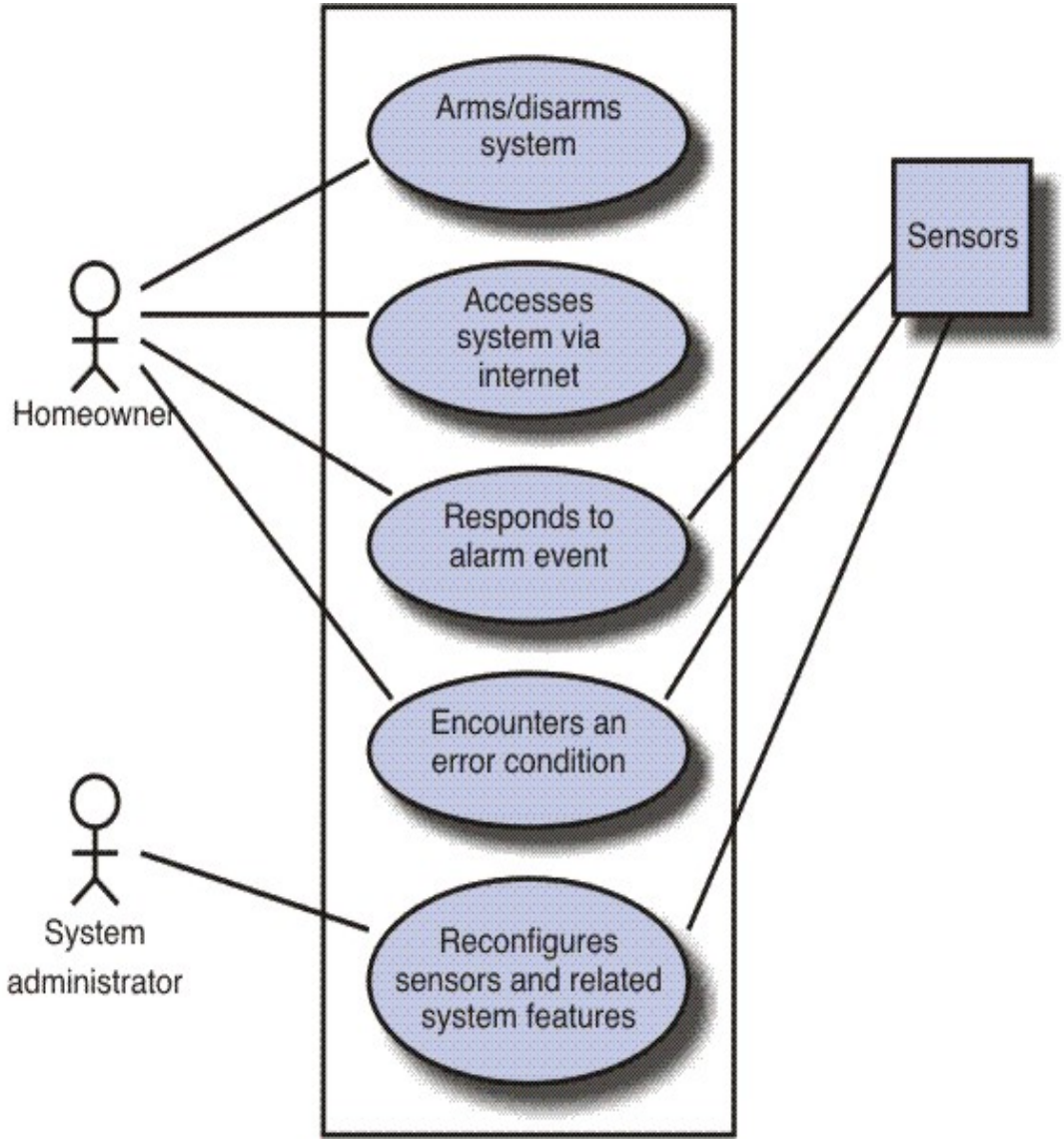
Elaboration: Building an Analysis Model

- Scenario-based elements
  - Use-case—How external actors interact with the system (use-case diagrams; detailed templates)
  - Functional—How software functions are processed in the system (flow charts; activity diagrams)
- Class-based elements
  - The various system objects (obtained from scenarios) including their attributes and functions (class diagram)
- Behavioral elements
  - How the system behaves in response to different events (state diagram)
- Flow-oriented elements
  - How information is transformed as if flows through the system (data flow diagram)

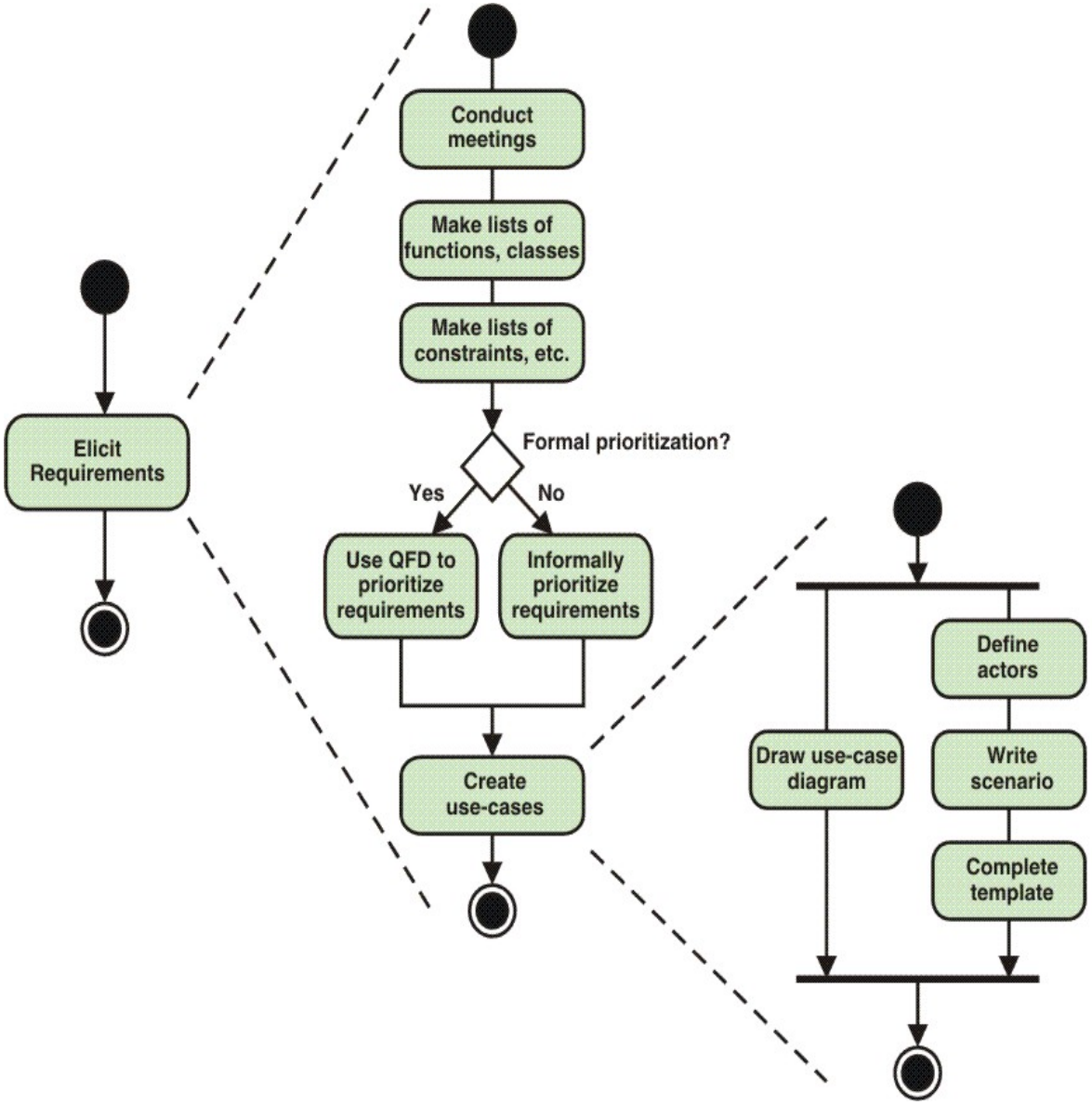
USE-CASES

- A use-case scenario is a story about how someone or something external to the software (known as an actor) interacts with the system
- Basic use cases present high-level stories
  - often use informal text-based descriptions to "tell the stories"
- After that, use cases are further elaborated to provide considerably more details
  - often use more formal templates
  - also use UML diagrams like use case diagram, activity diagram, etc.

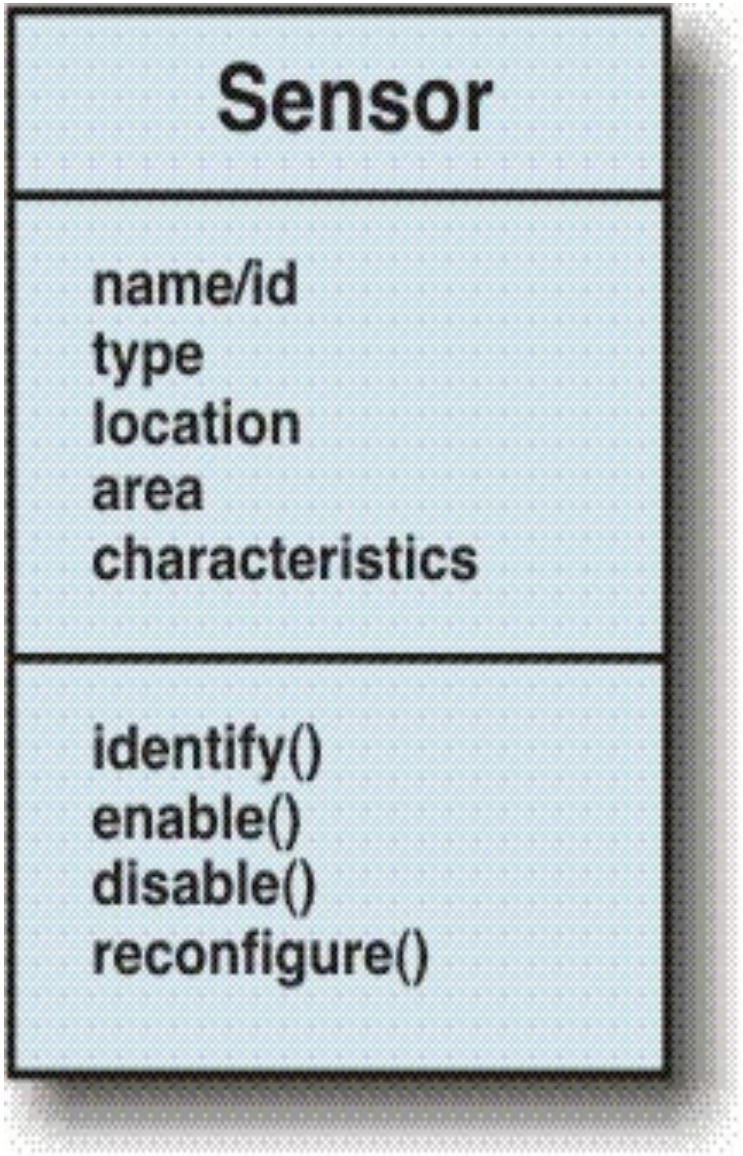
USE-CASE DIAGRAM



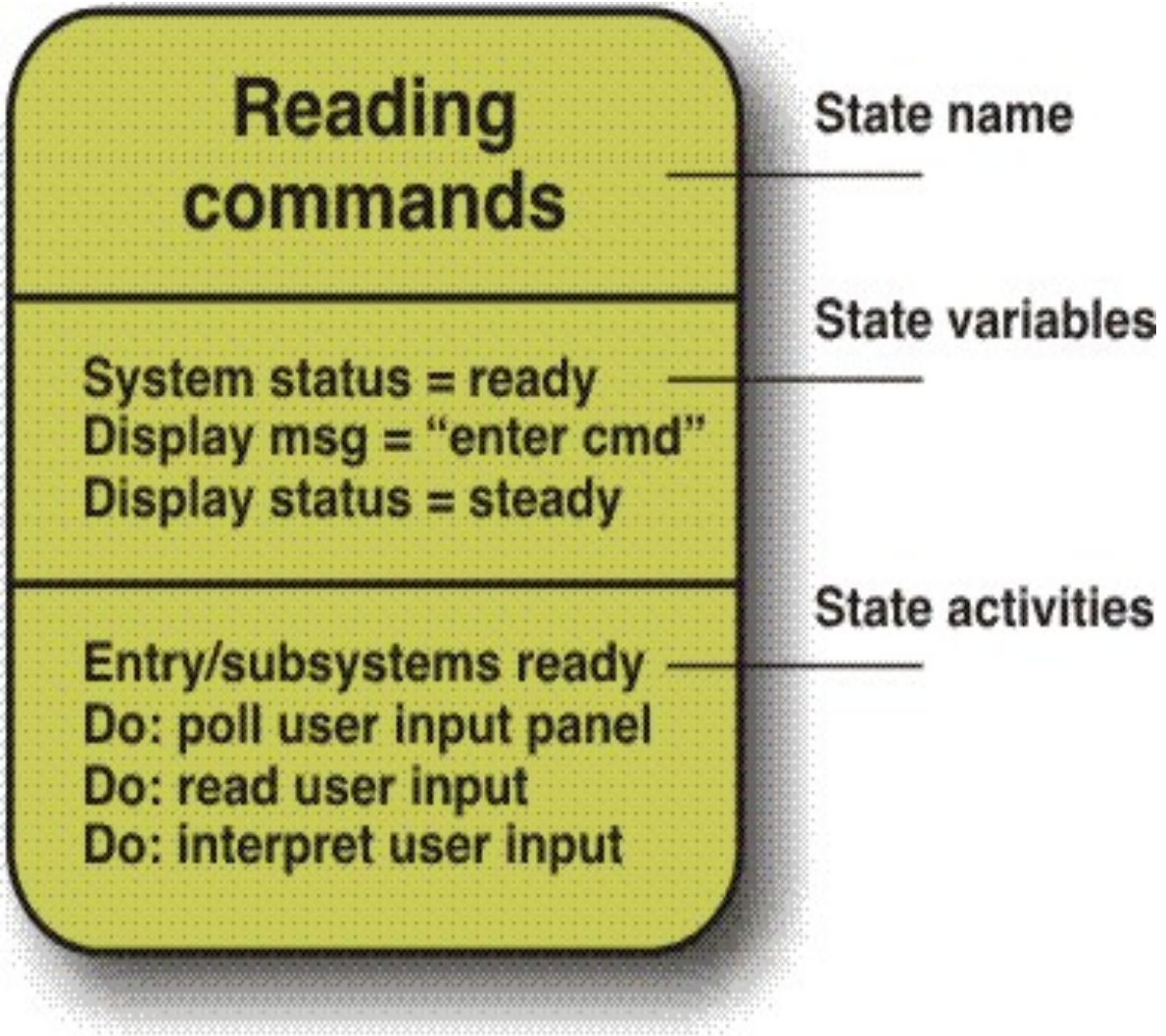
Activity Diagram for RE



Class Diagram



State Diagram



Analysis Patterns

Pattern name: A descriptor that captures the essence of the pattern.

Intent: Describes what the pattern accomplishes or represents

Motivation: A scenario that illustrates how the pattern can be used to address the problem.

Forces and context: A description of external issues (forces) that can affect how the pattern is used and also the external issues that will be resolved when the pattern is applied.

Solution: A description of how the pattern is applied to solve the problem with an emphasis on structural and behavioral issues.

Consequences: Addresses what happens when the pattern is applied and what trade-offs exist during its application.

Design: Discusses how the analysis pattern can be achieved through the use of known design patterns.

Known uses: Examples of uses within actual systems.

Related patterns: On e or more analysis patterns that are related to the named pattern because (1) it is commonly used with the named pattern; (2) it is structurally similar to the named pattern; (3) it is a variation of the named pattern.

Negotiating Requirements

- Identify the key stakeholders
  - These are the people who will be involved in the negotiation
- Determine each of the stakeholders "win conditions"
  - Win conditions are not always obvious
- Negotiate
  - Work toward a set of requirements that lead to "win-win"

VALIDATING REQUIREMENTS

- Is each requirement consistent with the objective of the system?
- Have all requirements been specified at the proper level of abstraction?
- Is the requirement really necessary?
- Is each requirement bounded and unambiguous?
- Does each requirement have attribution?
- Do any requirements conflict with other requirements?
- Is each requirement achievable in the system's technical environment?
- Is each requirement testable, once implemented?
- Does the model reflect the system's information, function and behavior?
- Has the model been appropriately "partitioned"?
- Have appropriate requirements patterns been used?

End of Chapter 5