#$%^@1*9a!#%^^&&&**

#$%^@1*9a!#%^^&&&**

Can I Get What I Want [Need] Without Knowing What I Want?

Can You Understand What I Want If I Don’t Tell You?

Can You Design X if You Don’t Know about X ?

The *Gizmo* of Worlds

- [Problem world]
- Is there one problem?
- What is it?

- [Expression of each world]|Cognitive

- [Solution world]
- [Design world]
- [Programmer's world] - [Expression of this world] |Intent, Syntax and Semantics

- [Machine world] - [Expression of this world] |Syntax & Semantics

The World of Stakeholders [Dilemma…]

Lead Software Engineer

Software Architect

Testing Engineer

Quality Analyst

Software Engineer

Domain Expert

Software Developer

Requirements Engineer

Usability Expert

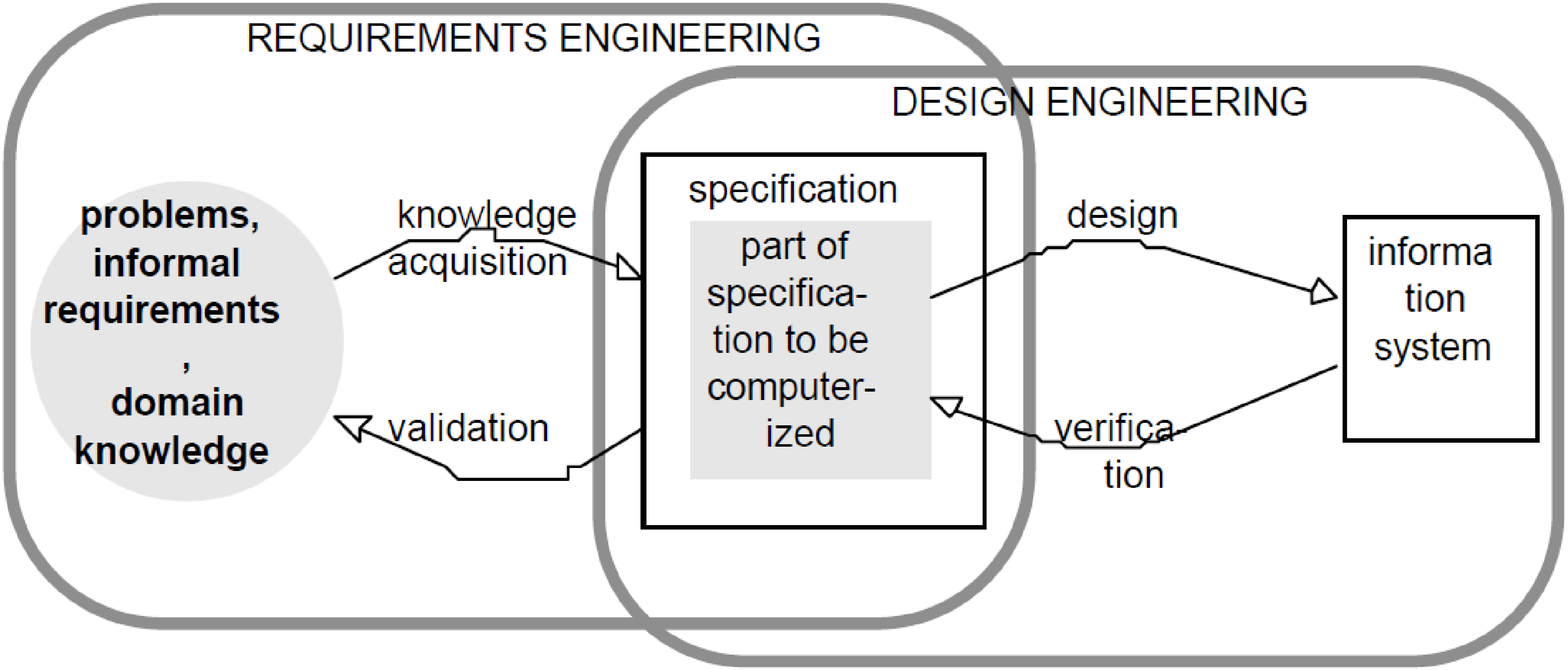
Support Engineer

Systems Analyst

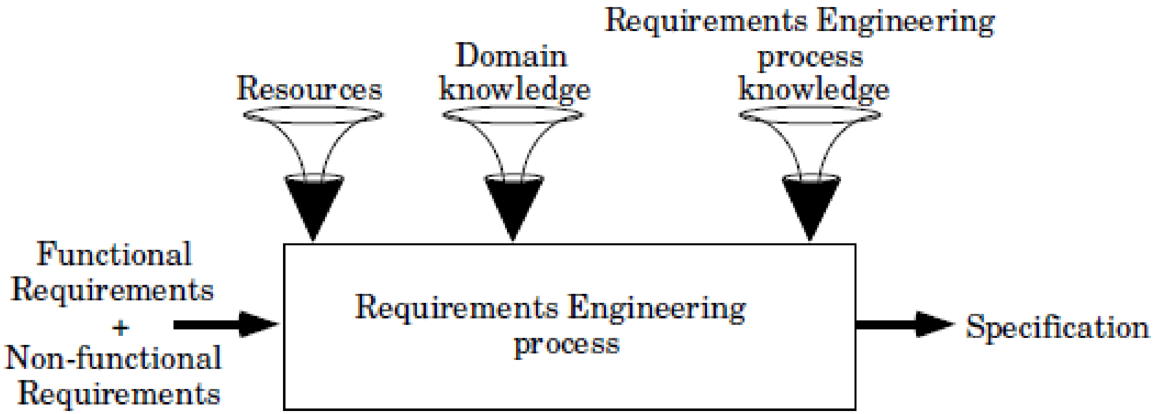
Business Analyst

Requirements Engineering – A Systems View

- It’s not software engineering alone!!!
- Multi-disciplinary
- Human-centered process
- Informal, semi-formal and formal
- functional and non-functional
- product and process-oriented
- …


REQUIREMENTS ENGINEERING
DESIGN ENGINEERING
problems, informal requirements , domain knowledge
knowledge acquisition
validation
specification
part of specifica- tion to be computer- ized
design
verifica- tion
informa tion system

A high-level process


Resources
Domain knowledge
Requirements Engineering process knowledge
Functional Requirements + Non-functional Requirements
Requirements Engineering process
Specification

Are Requirements Tough?

- "The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements . . . No other part of the work so cripples the resulting system if done wrong. No other part is as difficult to rectify later“ [Fred Brooks]

Requirements Engineering – Does it really work in practice?

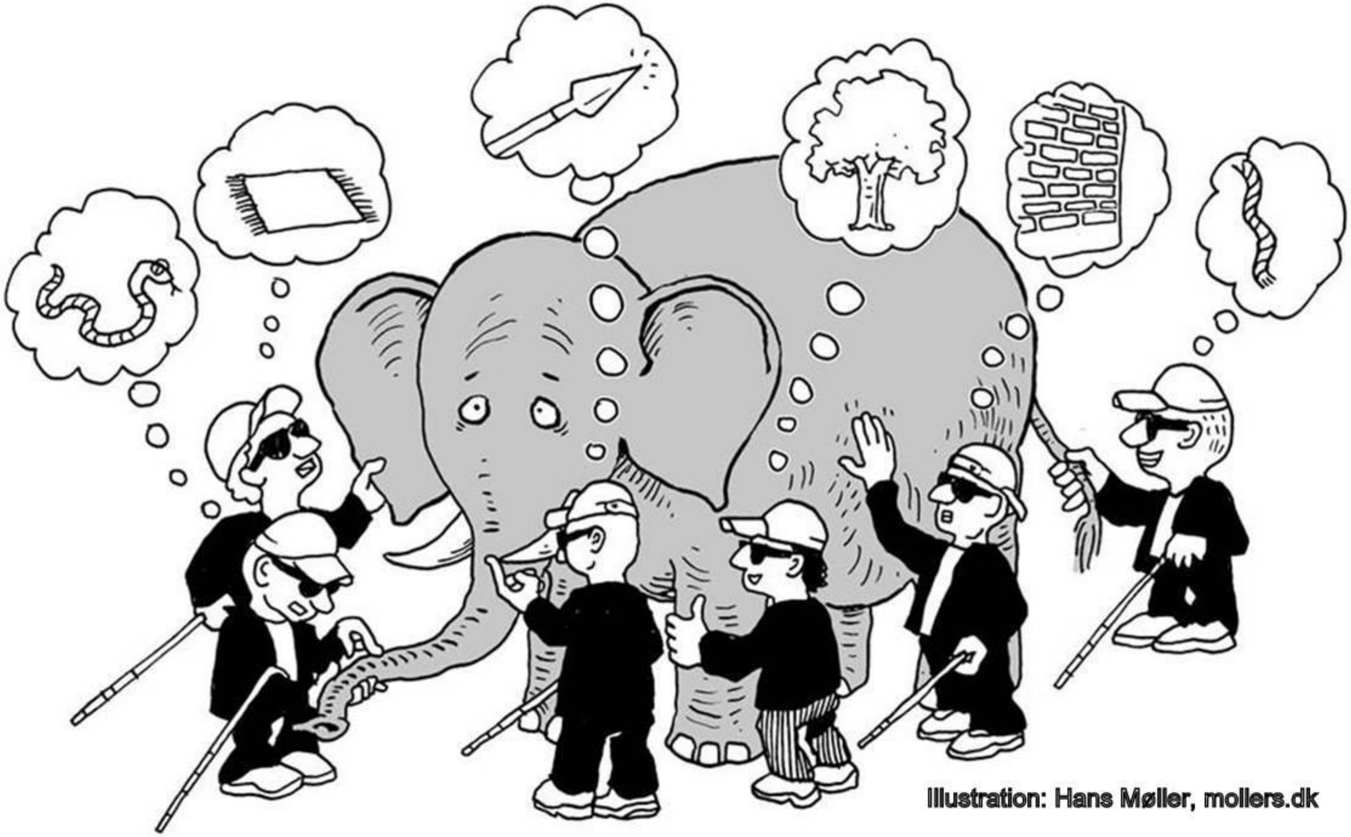
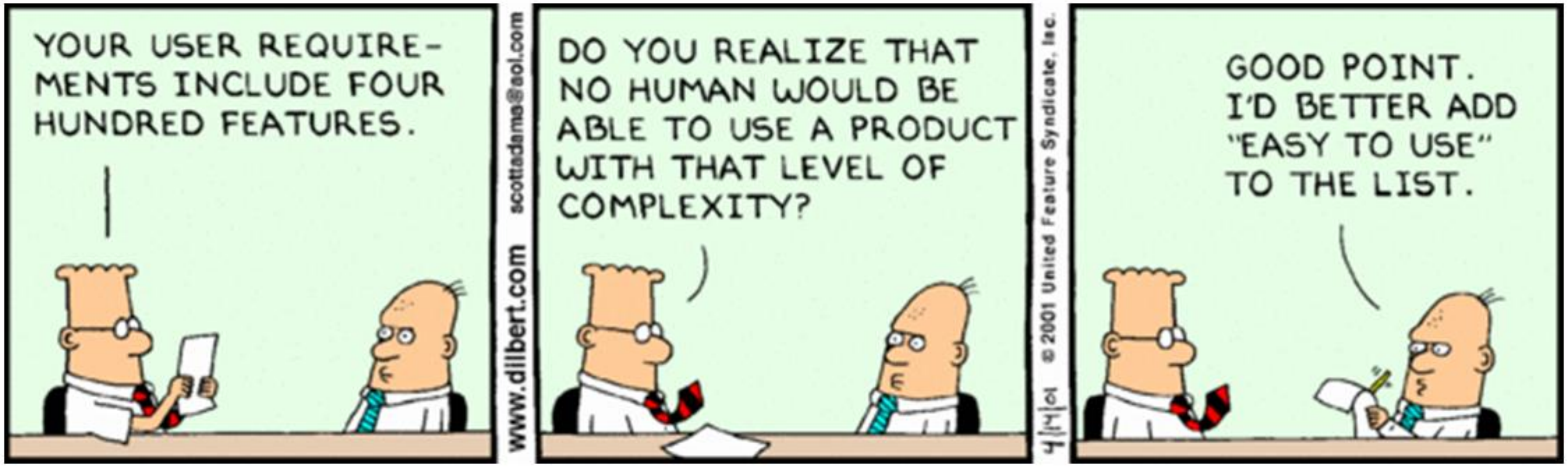


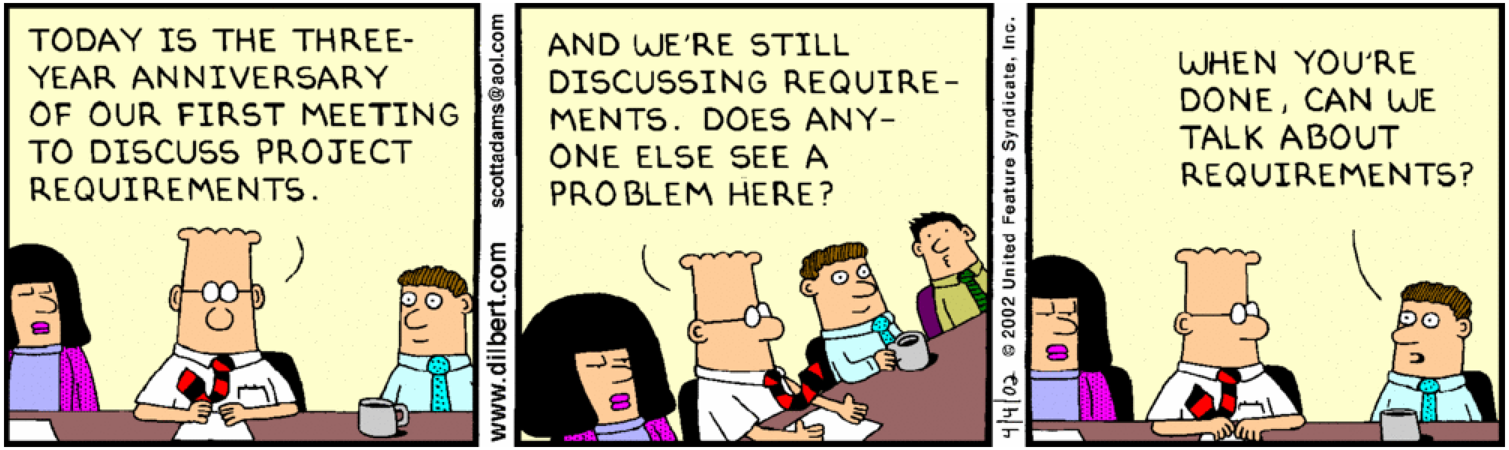
Illustration: Hans Møller, mollers.dk

Tip: Always freeze things after a certain point of time… not later…


YOUR USER REQUIREMENTS INCLUDE FOUR HUNDRED FEATURES.
DO YOU REALIZE THAT NO HUMAN WOULD BE ABLE TO USE A PRODUCT WITH THAT LEVEL OF COMPLEXITY?
GOOD POINT. I'D BETTER ADD "EASY TO USE" TO THE LIST.

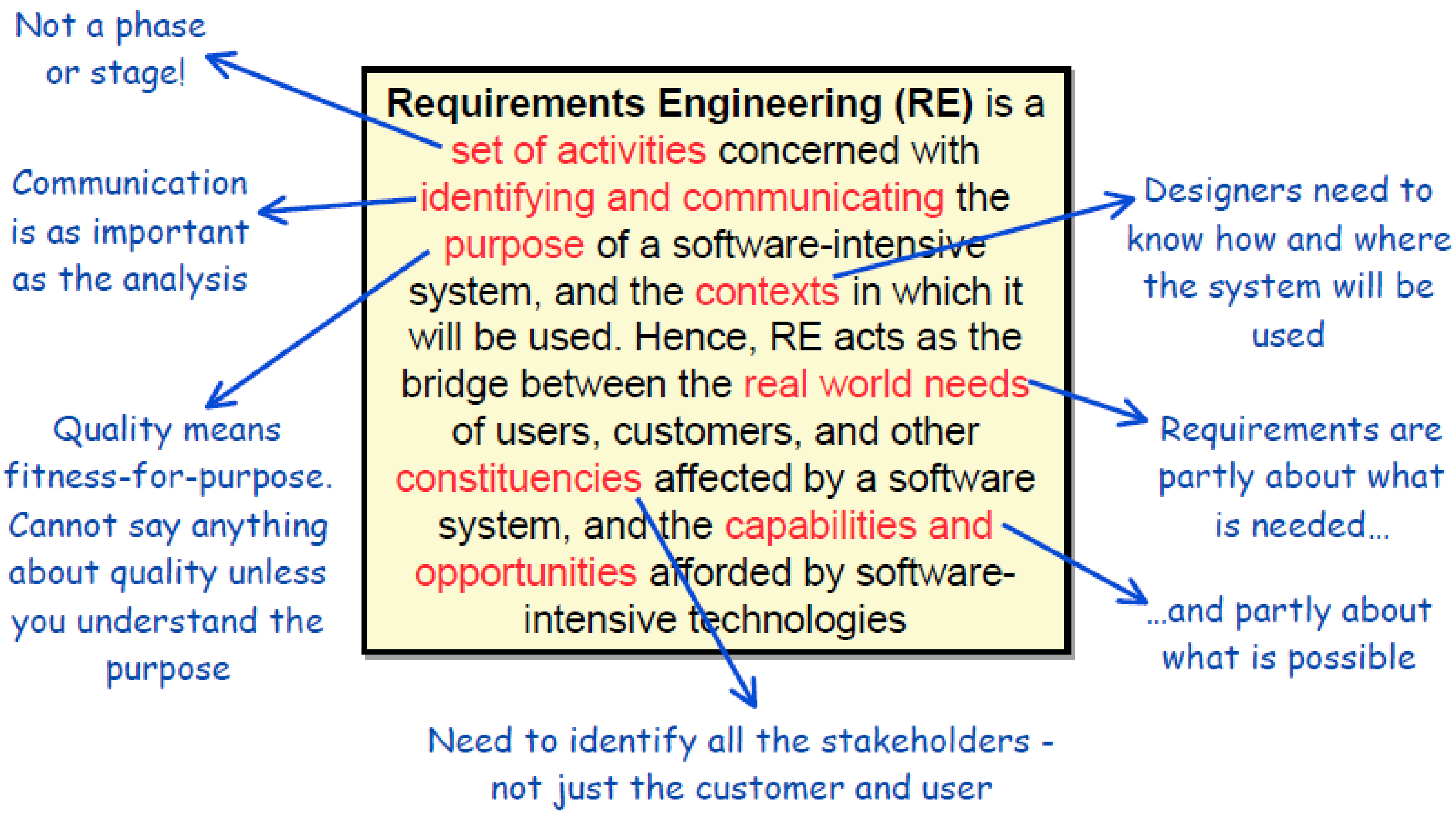
scottadams@aol.com www.dilbert.com

© 2001 United Feature Syndicate, Inc. 4/14/01


TODAY IS THE THREE-YEAR ANNIVERSARY OF OUR FIRST MEETING TO DISCUSS PROJECT REQUIREMENTS.
scottadams@aol.com
www.dilbert.com
AND WE'RE STILL DISCUSSING REQUIREMENTS. DOES ANYONE ELSE SEE A PROBLEM HERE?
© 2002 United Feature Syndicate, Inc.
4/4/02
WHEN YOU'RE DONE, CAN WE TALK ABOUT REQUIREMENTS?

I'LL DESIGN THE SYSTEM AS SOON AS YOU GIVE ME THE USER REQUIREMENTS.
BETTER YET, YOU COULD BUILD THE SYSTEM, THEN I'LL TELL YOUR BOSS THAT IT DOESN'T MEET MY NEEDS.
I DON'T MEAN TO FRIGHTEN YOU, BUT YOU'LL HAVE TO DO SOME ACTUAL WORK.
THAT'S CRAZY TALK.
www.dilbert.com
scottadams@aol.com
3/21/03
© 2003 United Feature Syndicate, Inc.

Definition of RE



Requirements Engineering (RE) is a set of activities concerned with identifying and communicating the purpose of a software-intensive system, and the contexts in which it will be used. Hence, RE acts as the bridge between the real world needs of users, customers, and other constituencies affected by a software system, and the capabilities and opportunities afforded by software-intensive technologies

Not a phase or stage!

Communication is as important as the analysis

Quality means fitness-for-purpose. Cannot say anything about quality unless you understand the purpose

Designers need to know how and where the system will be used

Requirements are partly about what is needed…

…and partly about what is possible

Need to identify all the stakeholders - not just the customer and user

Requirements Analysis

Some content adapted from Rajib Mall's book and Craig Larman's book.

Requirements Phase

- Many projects fail:
  - Because they start implementing the system.
  - Without determining whether they are building what the customer really wants.

Why Requirements analysis and specification?

- Factors that cause projects to fail:
  - Lack of User Input 12.8%
  - Incomplete Requirements & Specifications 12.3%
  - Changing Requirements & Specifications 11.8%
  - Lack of Executive Support 7.5%
  - Technology Incompetence 7.0%
  - Lack of Resources 6.4%
  - Unrealistic Expectations 5.9%
  - Unclear Objectives 5.3%
  - Unrealistic Time Frames 4.3%
  - New Technology 3.7%
  - Other 23.0%

- Standish Group, *The Standish Group Report: Chaos*, 1995, http://www.scs.carleton.ca/~beau/PM/Standish-Report.html

Domain Analysis

- With in the context of SDLC
  - Feasibility Study
  - Requirements
  - Design
  - Implementation
  - Testing
  - Maintenance

Domain Analysis

- The process by which a software engineer learns about the domain to better understand the problem:
  - The *domain* is the general field of business or technology in which the clients will use the software
  - A *domain expert* is a person who has a deep knowledge of the domain

- Benefits of performing domain analysis:
  - Faster development
  - Better system
  - Anticipation of extensions

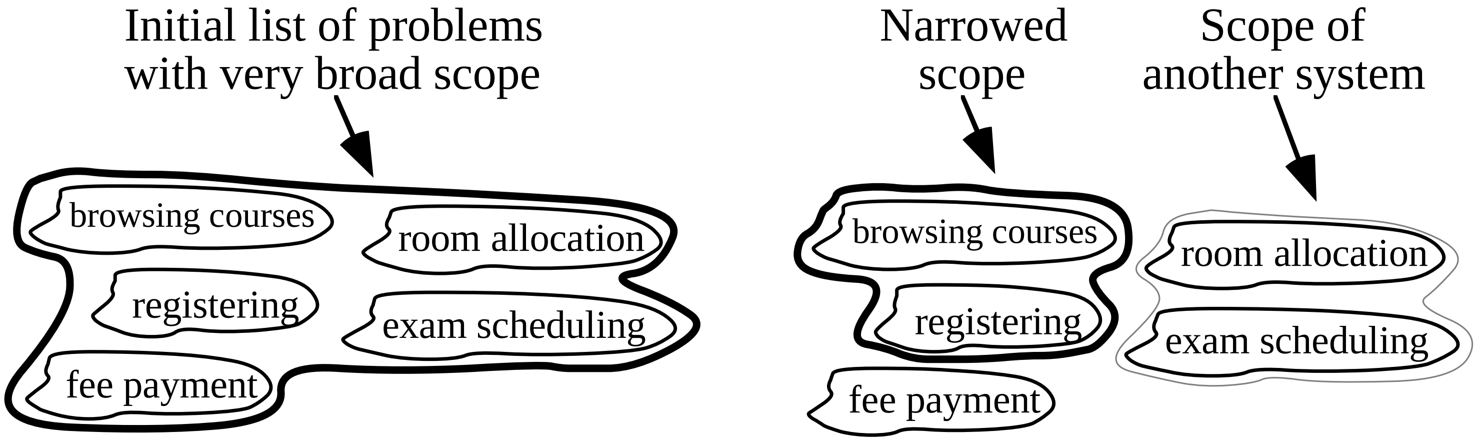
The Starting Point for Software Projects

	Requirements must be determined	Clients have produced requirements
New development	A	B
Evolution of existing system	C	D

Defining the Scope

- Narrow the *scope* by defining a more precise problem
  - List all the things you might imagine the system doing
    - Exclude some of these things if too broad
    - Determine high-level goals if too narrow

- Example: A university registration system



What is a Requirement?

- Requirement: A statement about the proposed system that all stakeholders agree must be made true in order for the customer's problem to be adequately solved.
  - Short and concise piece of information
  - Says something about the system
  - All the stakeholders have agreed that it is valid
  - It helps solve the customer's problem

- A collection of requirements is a *requirements document*.

Types of Requirements

- Business requirements
  - High-level objectives of the organization or customer who requests the system.

- Functional requirements
  - Describe *what* the system should do

  For example, features (use cases)

- Non-functional requirements
  - *Constraints* that must be adhered to during development

  For example, quality constraints, technology constraints, process constraints, etc.

Requirements Phase

- Goals of requirements phase:
  - Fully understand the user requirements.
  - Remove inconsistencies, anomalies, etc. from requirements.
  - Document requirements properly in an SRS document.

Requirements Phase

- Consists of two distinct activities:
    - Requirements Gathering and Analysis
    - Requirements Specification

Who Carries Out Requirements Analysis and Specification?

- The person who undertakes requirements analysis and specification:
  - Known as requirements analyst or systems analyst
  - Collects data pertaining to the product
  - Analyzes collected data:
    - To understand what exactly needs to be done.
  - Writes the Software Requirements Specification (SRS) document.

Requirements Phase

- Final output of this phase:
  - Software Requirements Specification (SRS) Document.
- The SRS document is reviewed by the customer.
  - Reviewed SRS document forms the basis of all future development activities.

Requirements Analysis

- Requirements analysis consists of two main activities:
  - Requirements gathering
  - Analysis of the gathered requirements

Requirements Gathering

- Also known as requirements elicitation.
- If the project is to automate some existing procedures
  - e.g., automating existing manual accounting activities,
  - The task of the system analyst is a little easier
  - Analyst can immediately obtain:
    - input and output formats
    - accurate details of the operational procedures

Requirements Gathering (CONT.)

- In the absence of a working system,
  - Lot of imagination and creativity are required.
- Interacting with the customer to gather relevant data:
  - Requires a lot of experience.

Case Study: Automation of Office Work at CSE Dept.

- The academic, inventory, and financial information at the CSE department:
  - Being carried though manual processing by two office clerks, a store keeper, and two attendants.
- Considering the low budget he had at his
- Disposal:
  - The HoD entrusted the work to a team of student volunteers.

Case Study: Automation of Office Work at CSE Dept.

- The team was first briefed by the HoD about the specific activities to be automated.
- The analyst first discussed with the two clerks:
  - Regarding their specific responsibilities (tasks) that were to be automated.
- The analyst also interviewed student and faculty representatives who would also use the software.

Case Study: Automation of Office Work at CSE Dept.

- For each task, they asked:
  - About the steps through which these are performed.
  - They also discussed various scenarios that might arise for each task.
  - The analyst collected all types of forms that were being used.

Analysis of the gathered requirements

- Main purpose of requirements analysis:
    - Clearly understand the user requirements,
    - Detect inconsistencies, ambiguities, and incompleteness.
- Incompleteness and inconsistencies:
    - Resolved through further discussions with the end-users and the customers.

Inconsistent Requirement

- Some part of the requirement:
  - contradicts with some other part.
- Example:
  - One customer says turn off heater and open water shower when temperature > 100 C
  - Another customer says turn off heater and turn ON cooler when temperature > 100 C

Incomplete Requirement

- Some requirements have been omitted:
  - Possibly due to oversight.
- Example:
  - The analyst has not recorded:
    when temperature falls below 90 C
    - heater should be turned ON
    - water shower turned OFF.

Analysis of the gathered requirements (contd.)

- Requirements analysis involves:
  - Obtaining a clear, in-depth understanding of the product to be developed,
  - Remove all ambiguities and inconsistencies from the initial customer perception of the problem.

Analysis of gathered requirements (contd.)

- Experienced analysts take considerable time:
  - To understand the exact requirements the customer has in his mind.
- Experienced systems analysts know - often as a result of past (painful) experiences

Analysis of gathered requirements (contd.)

- Several things about the project should be clearly understood by the analyst:
  - What is the problem?
  - Why is it important to solve the problem?
  - What are the possible solutions to the problem?
  - What complexities might arise while solving the problem?

Analysis of gathered requirements (contd.)

- Some anomalies and inconsistencies can be very subtle:
  - Escape even most experienced eyes.
  - If a **formal model** of the system is constructed,
    - Many of the subtle anomalies and inconsistencies get detected.

Analysis of gathered requirements (contd.)

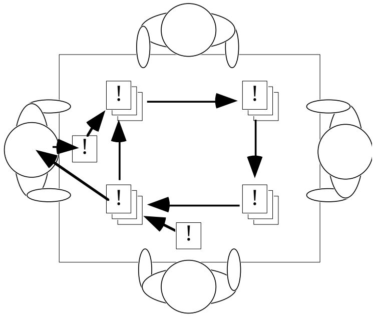
- After collecting all data regarding the system to be developed,
  - Remove all inconsistencies and anomalies from the requirements,
  - Systematically organize requirements into a Software Requirements Specification (SRS) document.

Techniques - Gathering and Analyzing Requirements

- Observation
  - Read documents and discuss requirements with users
  - Shadowing important potential users as they do their work
    - ask the user to explain everything he or she is doing
  - Session videotaping
- Interviewing
  - Conduct a series of interviews
    - Ask about specific details
    - Ask about the stakeholder’s vision for the future
    - Ask if they have alternative ideas
    - Ask for other sources of information
    - Ask them to draw diagrams

Gathering and Analyzing Requirements

- Brainstorming
  - Appoint an experienced moderator
  - Arrange the attendees around a table
  - Decide on a ‘trigger question’
  - Ask each participant to write an answer and pass the paper to its neighbour



- ***Joint Application Development (JAD)* is a technique based on intensive brainstorming sessions**

Gathering and Analyzing Requirements

- Prototyping
  - The simplest kind: *paper prototype*.
    - a set of pictures of the system that are shown to users in sequence to explain what would happen
  - The most common: a mock-up of the system's UI
    - Written in a rapid prototyping language
    - Does *not* normally perform any computations, access any databases or interact with any other systems
    - May prototype a particular aspect of the system

Difficulties and Risks in Domain and Requirements analysis

- Lack of understanding of the domain or the real problem
  - *Do domain analysis and prototyping*
- Requirements change rapidly
  - *Perform incremental development, build flexibility into the design, do regular reviews*
- Attempting to do too much
  - *Document the problem boundaries at an early stage, carefully estimate the time*
- It may be hard to reconcile conflicting sets of requirements
  - *Brainstorming, JAD sessions, competing prototypes*
- It is hard to state requirements precisely
  - *Break requirements down into simple sentences and review them carefully, look for potential ambiguity, make early prototypes*