

Vragen

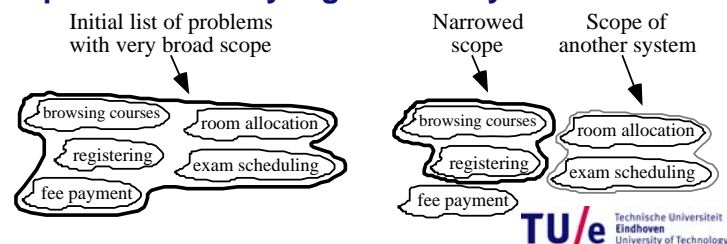
- Noem 3 software proces modellen
- Wat de overeenkomst tussen de moderne proces modellen?
- Wat is het nut van domein analyse?

Defining Problem and Scope

- A problem can be expressed as:
 - A *difficulty* the users or customers are facing,
 - Or as an *opportunity* that will result in some benefit such as improved productivity or sales.
- The solution to the problem normally will entail developing software
- A good problem statement is short and succinct

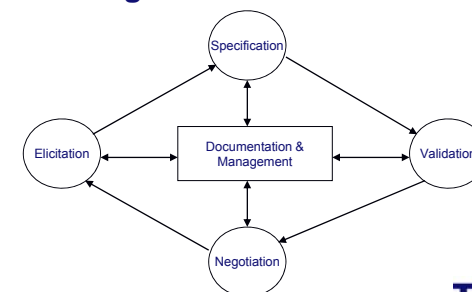
Defining the Scope

- 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



Processes in requirements engineering

- Requirements elicitation
- Requirements specification
- Requirements validation and verification
- Requirements negotiation



What is a Requirement ?

- It is a statement describing either
 - 1) an aspect of what the proposed system must do,
 - or 2) a constraint on the system's development.
- In either case it must contribute in some way towards adequately solving the customer's problem;
- the set of requirements as a whole represents a negotiated agreement among the stakeholders.

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

Types of Requirements

- Functional requirements
 - Describe *what* the system should do
- Quality requirements
 - *Constraints* on the design to meet specified levels of quality
- Platform requirements
 - *Constraints* on the environment and technology of the system
- Process requirements
 - *Constraints* on the project plan and development methods

Functional Requirements

- What *inputs* the system should accept
- What *outputs* the system should produce
- What data the system should *store* that other systems might use
- What *computations* the system should perform
- The *timing and synchronization* of the above

Quality Requirements

- All must be verifiable
- Examples: Constraints on
 - Response time
 - Throughput
 - Resource usage
 - Reliability
 - Availability
 - Recovery from failure
 - Allowances for maintainability and enhancement
 - Allowances for reusability

Elicitation techniques

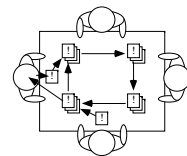
- **Asking:**
 - interview
 - Delphi technique
 - brainstorming session
- **Observing**
 - task analysis
 - scenario analysis
 - ethnography
 - form analysis
 - synthesis from existing system
- **Others:**
 - analysis of natural language descriptions
 - domain analysis
 - Business Process Redesign (BPR)
 - prototyping

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

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

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 video taping**

Task Analysis

- Task analysis is the process of analyzing the way people perform their jobs: the things they do, the things they act on and the things they need to know.
- The relation between tasks and goals: a task is performed in order to achieve a goal.
- Task analysis has a broad scope.

Task Analysis

- Task analysis concentrates on the current situation. However, it can be used as a starting point for a new system:
 - users will refer to new elements of a system and its functionality
 - scenario-based analysis can be used to exploit new possibilities