Requirements Engineering

A very brief overview

Why do Software Projects Fail?

- Depending of project size, between 25% and 55% of projects fail because of cancellations or delays in schedule, **due to poor requirements management**

  Laker 1998

- "Analysts report that **as many as 71 percent of software projects that fail** do so because of **poor requirements management**, making it the **single biggest reason for project failure** - bigger than bad technology, missed deadlines or change management fiascoes"

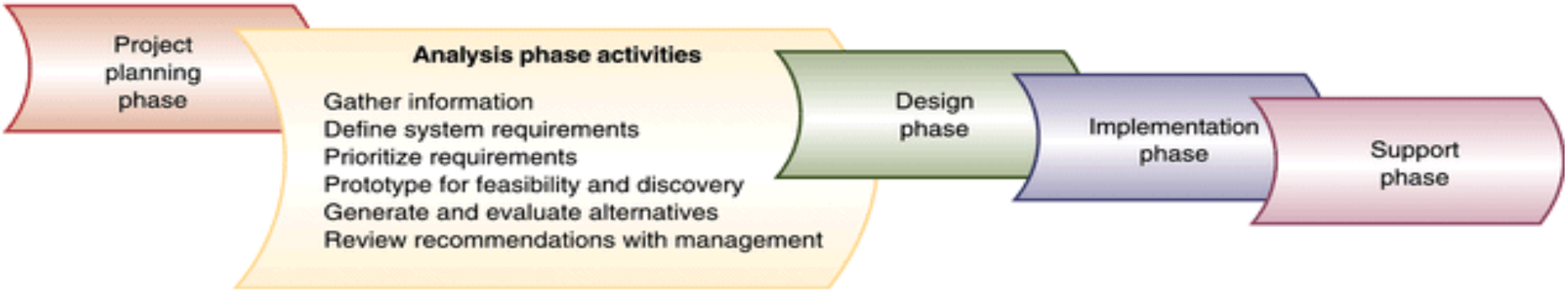
  CIO Magazine, November 2005

- " 80% new products fail, 70% of software projects fail **due to poor requirements"**

  Leveraging Business Architecture to Improve Business Requirements Analysis, 2014

Strategies for analysis:

- Problem Analysis - what are the problems?
- Root Cause Analysis - what is the source of the problem?
- Duration analysis - does process take longer than the sum of activities? (parallel work?)
- Activity based costing - which steps do not pay off?
- Informal Benchmarking - compare to others/competitors
- Technology analysis - any new interesting, useful technology available?



Requirement

- "*Requirement - statement which translates or expresses a need and its associated constraints and conditions*" – ISO/IEC/IEEE 29148 standard
- Functional
  - what the system should do
- Non-functional
  - How well/good/fast the system should do
  - a ***quality*** of the (functionality of the) product
  - It can represent the use case as a whole or one of the specific functional requirements
  - HOW does not mean technical requirement.

Example of requirements

- 9. Functional Requirements
  - *“The product shall record all the roads that have been treated”*

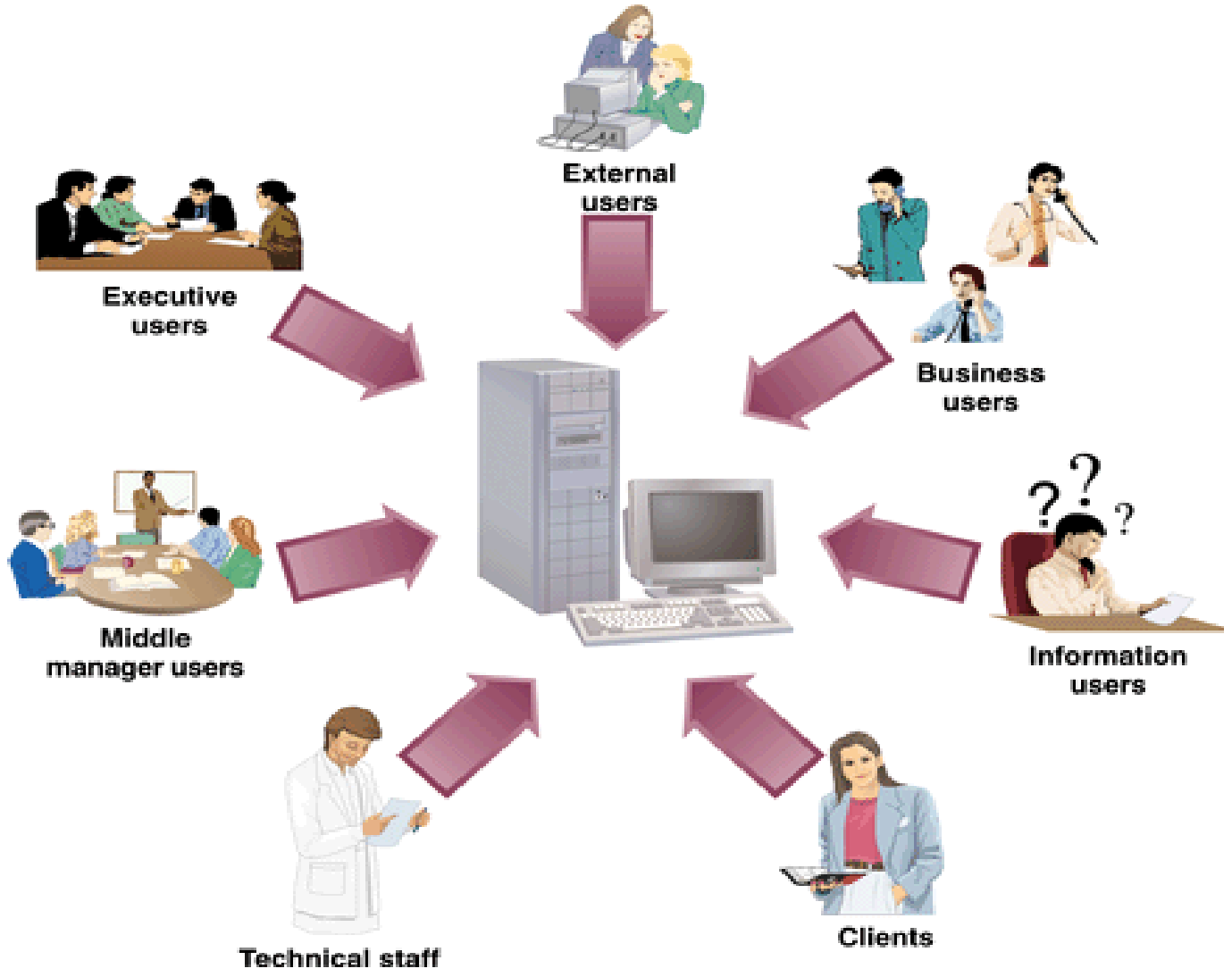
Nonfunctional Requirements

- 10. Look and Feel Requirements
  - *The product shall be attractive to a teenage audience.*
- 11. Usability and Humanity Requirements
  - *The product shall help the user to avoid making mistakes,*
  - *The product shall be usable by partially sighted users.*
- 12. Performance Requirements
  - *Any interface between a user and the automated system shall have a maximum response time of 2 seconds.*
- 13. Operational and Environmental Requirements
  - *The product shall be usable in dim light.*
- 14. Maintainability and Support Requirements
  - *A new weather station must be able to be added to the system overnight.*
- 15. Security Requirements
  - *Only direct managers can see the personnel records of their staff.*
- 16. Cultural and Political Requirements
  - *The product shall be able to distinguish between French, Italian, and British road-numbering systems*
- 17. Legal Requirements
  - *The product shall comply with insurance industry standards.*

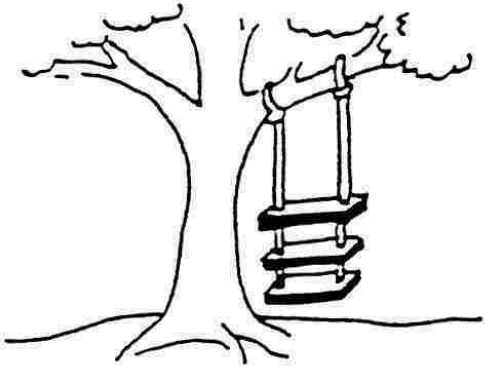
Requirements engineering activities (IEEE)

- Requirement Determination / Elicitation /Gathering /Discovery
  - The process of seeking, capturing and consolidating requirements from available requirements sources.
- Requirements Analysis
  - Analysis of elicited requirements in order to understand and document them.
- Requirements specification
  - A systematically represented collection of requirements, typically for a system or component, that satisfies given criteria.
- Systems modeling
  - deriving models of the system, often using a notation such as the Unified Modeling Language (UML)
  - Helps in better understanding and communication the requirements
- Requirements validation
  - checking that the documented requirements and models are consistent and meet stakeholder needs
- Requirements management
  - managing changes to the requirements as the system is developed and put into use

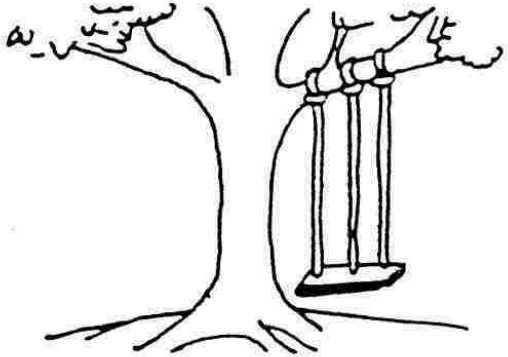
Stakeholders = a source of reqs



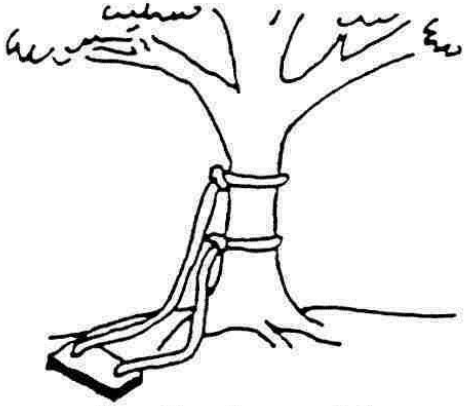
"Problem solving is an art form not fully appreciated by some"



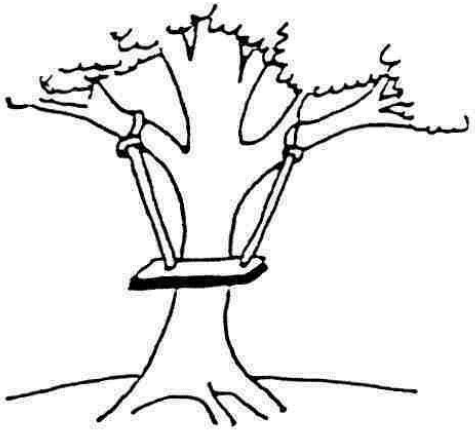
As proposed by the project sponsors



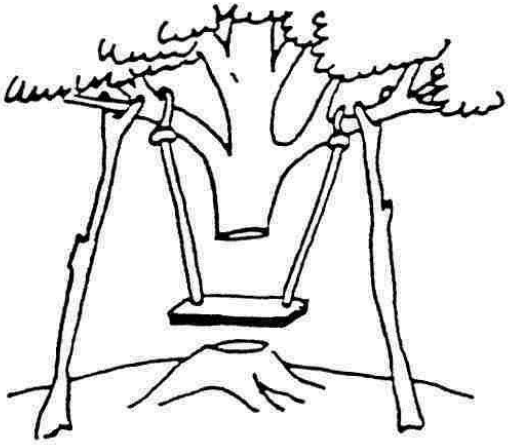
As specified in the project request



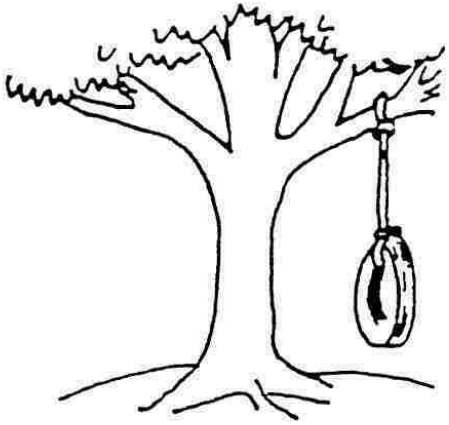
As designed by the senior analyst



As produced by the programmers



As installed at the user's site



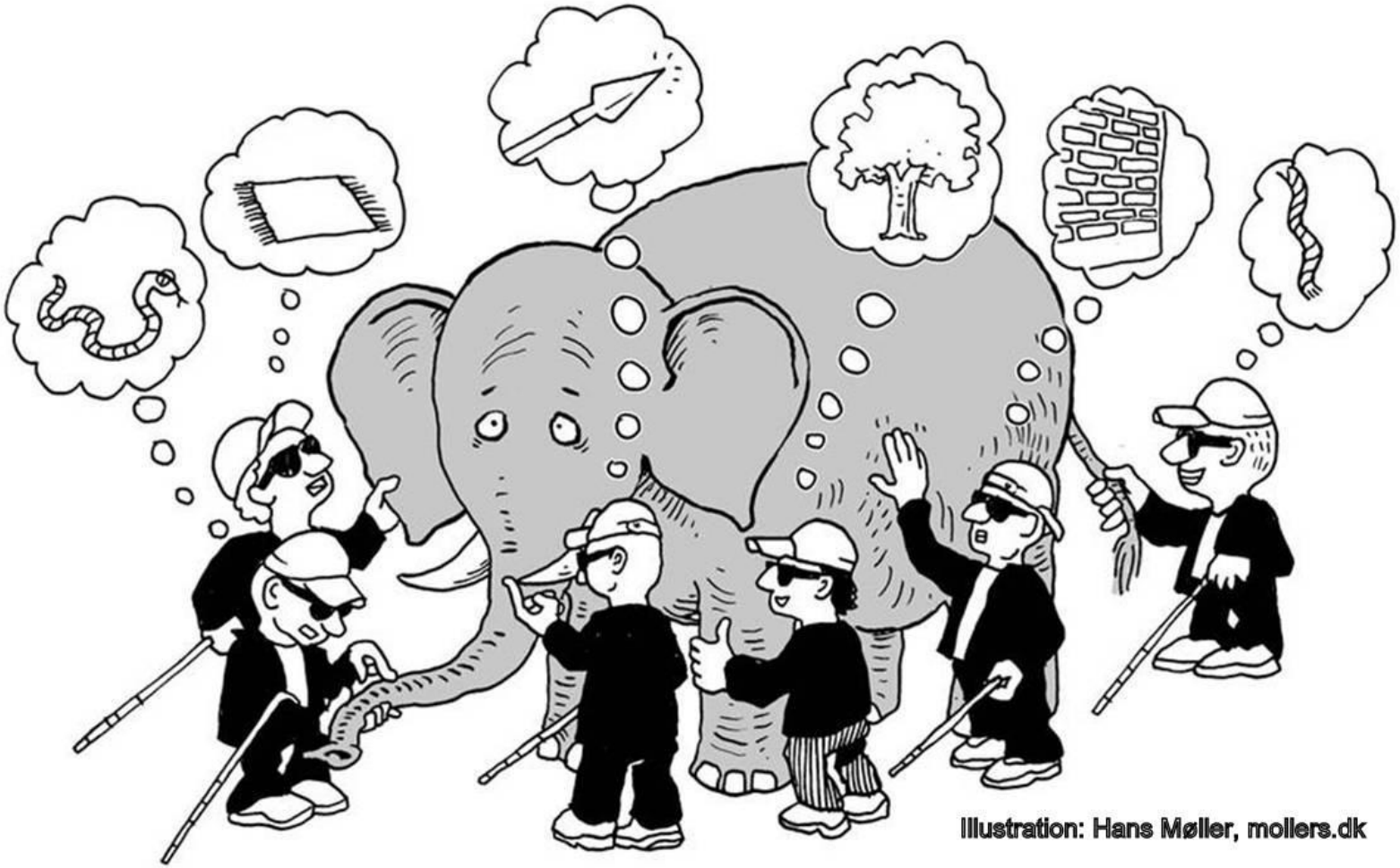
What the user wanted

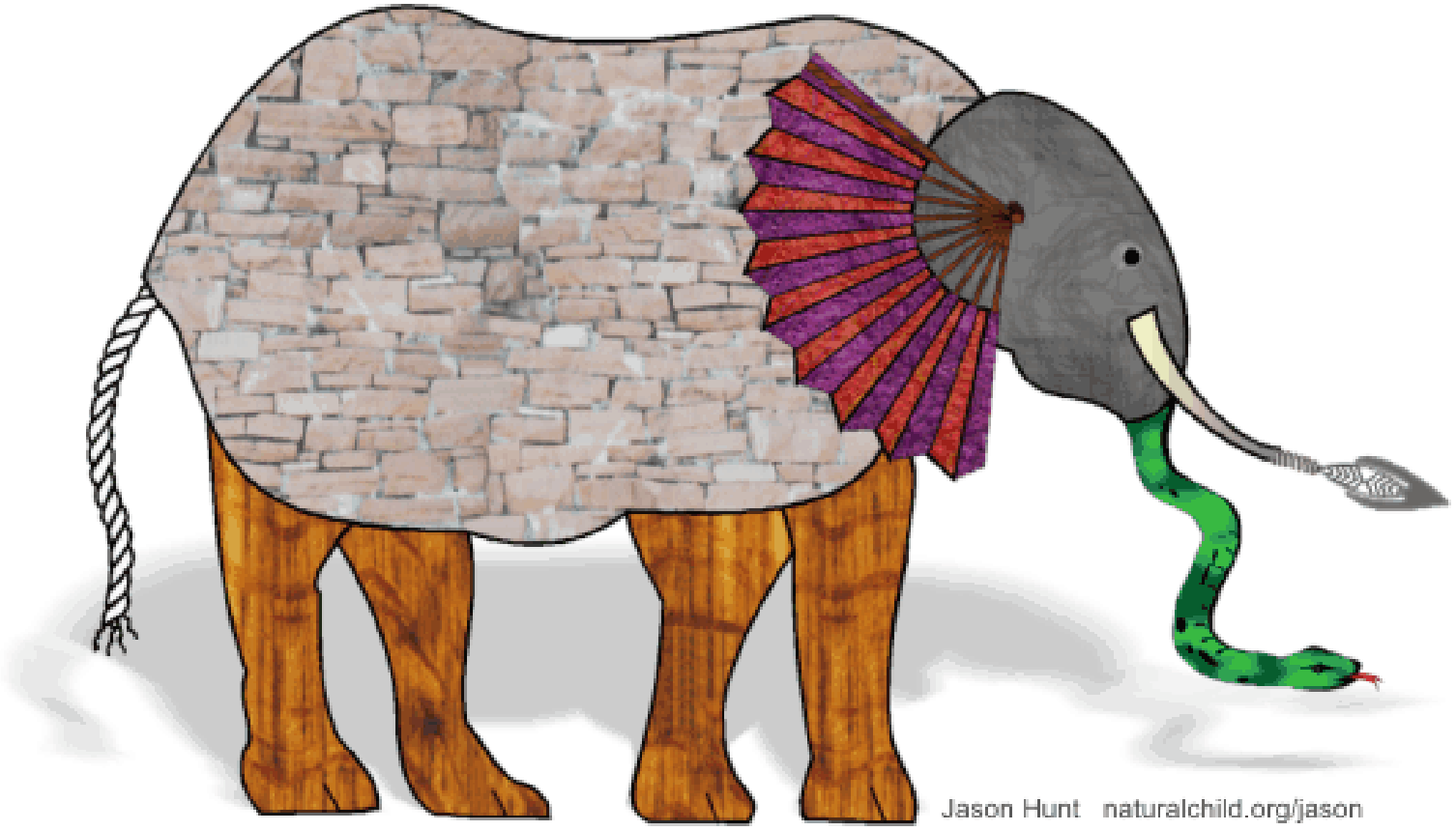
Tree Swing graphic by S Høgh 1993 - from Businessballs.com/treeswing.htm 2013

Stakeholder issues

- Users do not understand what they want
- All requirements are critical
- User change requirements after cost and schedule fixed
- Communication with users is slow
- Users are not technical experts
- They do not know about latest technology
- Users do not understand development process

Gathering requirements is difficult




Jason Hunt naturalchild.org/jason

Techniques for requirements elicitation

- Questionaires
- Interviews
- Document Analysis
- Observations
- Prototyping

Interviews: checklist

Checklist for Conducting an Interview

Before

- Establish the objective for the interview
- Determine correct user(s) to be involved
- Determine project team members to participate
- Build a list of questions and issues to be discussed
- Review related documents and materials
- Set the time and location
- Inform all participants of objective, time, and locations

During

- Dress appropriately
- Arrive on time
- Look for exception and error conditions
- Probe for details
- Take thorough notes
- Identify and document unanswered items or open questions

After

- Review notes for accuracy, completeness, and understanding
- Transfer information to appropriate models and documents
- Identify areas needing further clarification
- Send thank-you notes if appropriate

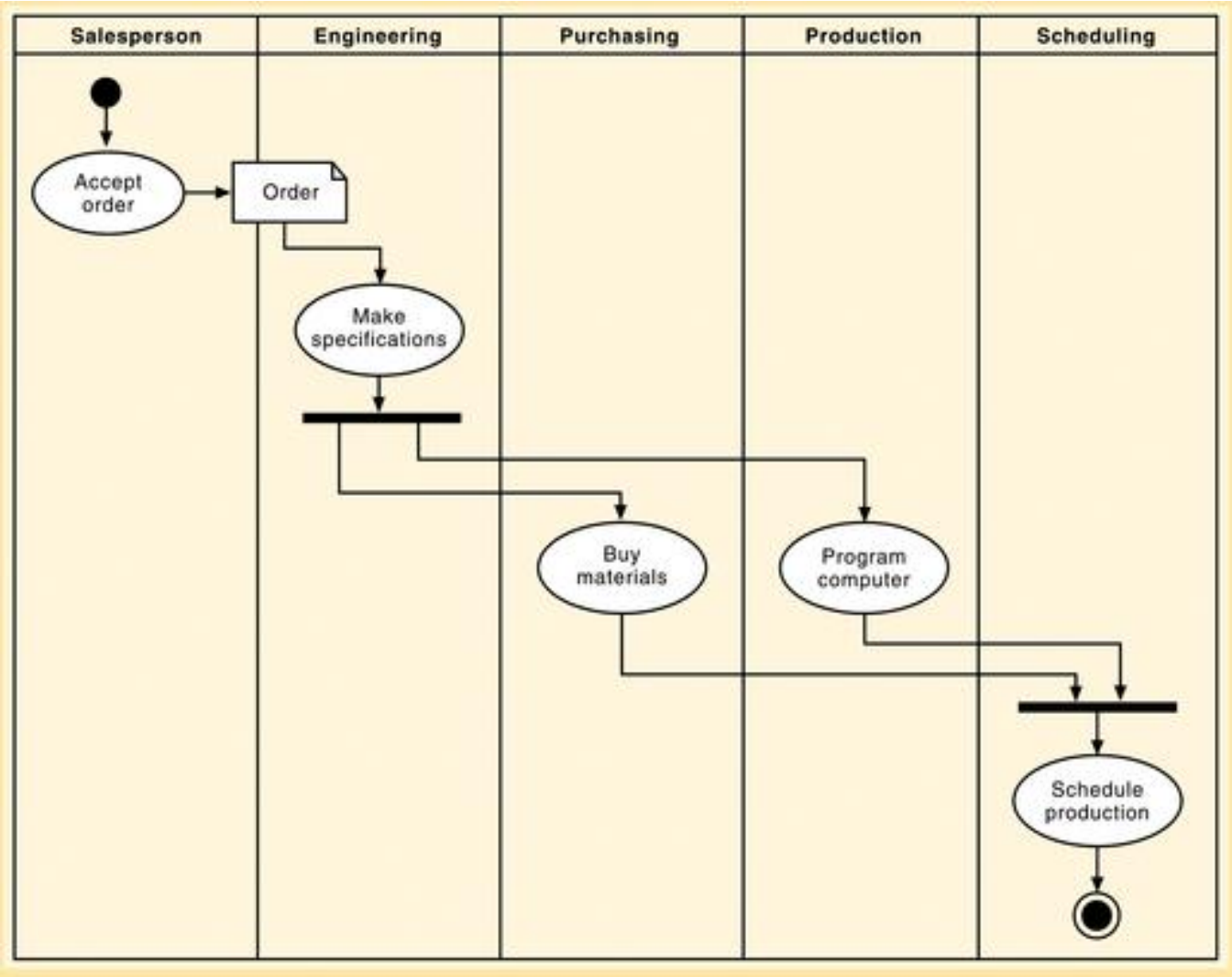
Types of questions - Examples

- Closed-Ended Questions
  - How many telephone orders are received per day?
  - How do customers place orders?
  - What information is missing from the monthly sales report?
- Open-Ended Questions
  - What do you think about the way invoices are currently processed?
  - What are some of the problems you face on a daily basis?
  - What are some of the improvements you would like to see in the way invoices are processed?
- Probing Questions
  - Why?
  - Can you give me an example?
  - Can you explain that in a bit more detail?

Questionaire design

- Begin with nonthreatening and interesting questions.
- Group items into logically coherent sections.
- Do not put important items at the very end of the questionnaire.
- Do not crowd a page with too many items.
- Avoid abbreviations.
- Avoid biased or suggestive items or terms.
- Number questions to avoid confusion.
- Pretest the questionnaire to identify confusing questions.
- Provide anonymity to respondents.

Observe and document processes



Prototyping

- A simplified potential product simulation of the requirements
- Can be done at early stages to clarify or discover new requirements
- Brige the terminology gap between stakeholders

Prototyping

- **IEEE** defines **prototyping** as "A type of development in which emphasis is placed on developing prototypes early in the development process to permit ***early feedback*** and ***analysis*** in support of the development process."
- A prototype is an initial/partial version of a system which is available early in the development phase
  - Some functionality may be left out
    - A prototype **does not** include all possible requirements
  - Non-functional requirements (performance) are less stringent
  - No complete documentation
- In other design fields a prototype is a small-scale model:
  - a miniature car, a miniature building or town
- In software design it can be (among other things):
  - a series of screen sketches, a storyboard, etc,
  - a Powerpoint slide show
  - a video simulating the use of a system
  - A piece of software with limited functionality written in the target language or in another language

Prototyping

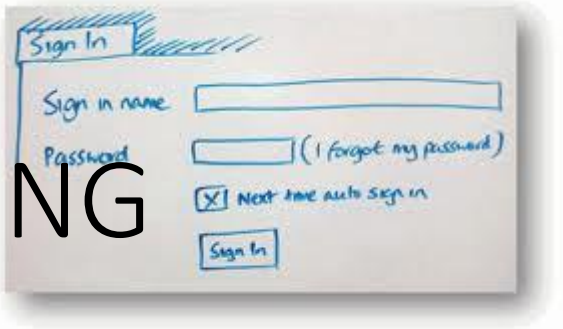
- Can be done at early stages to clarify or discover new requirements
- Brige the terminology gap between stakeholders
- Can even reduce the development costs
  - Forces a detailed study of the requirements which reveals inconsistencies and omissions
- Essential for developing the ‘look and feel’ of a user interface
- The use cases and scenarios are visualised with the prototypes in demonstration sessions with system stakeholders in order to elicit software requirements.
- Some tools: Powerpoint, Balsamiq, etc.

Types of prototypes

- horizontal prototypes
  - Behavioral prototypes that do not necessarily contain any real functionality
  - For refining unclear requirements
- vertical prototype
  - implements a specific part of system functionality in a quality manner
  - for example, in algorithm optimisation.

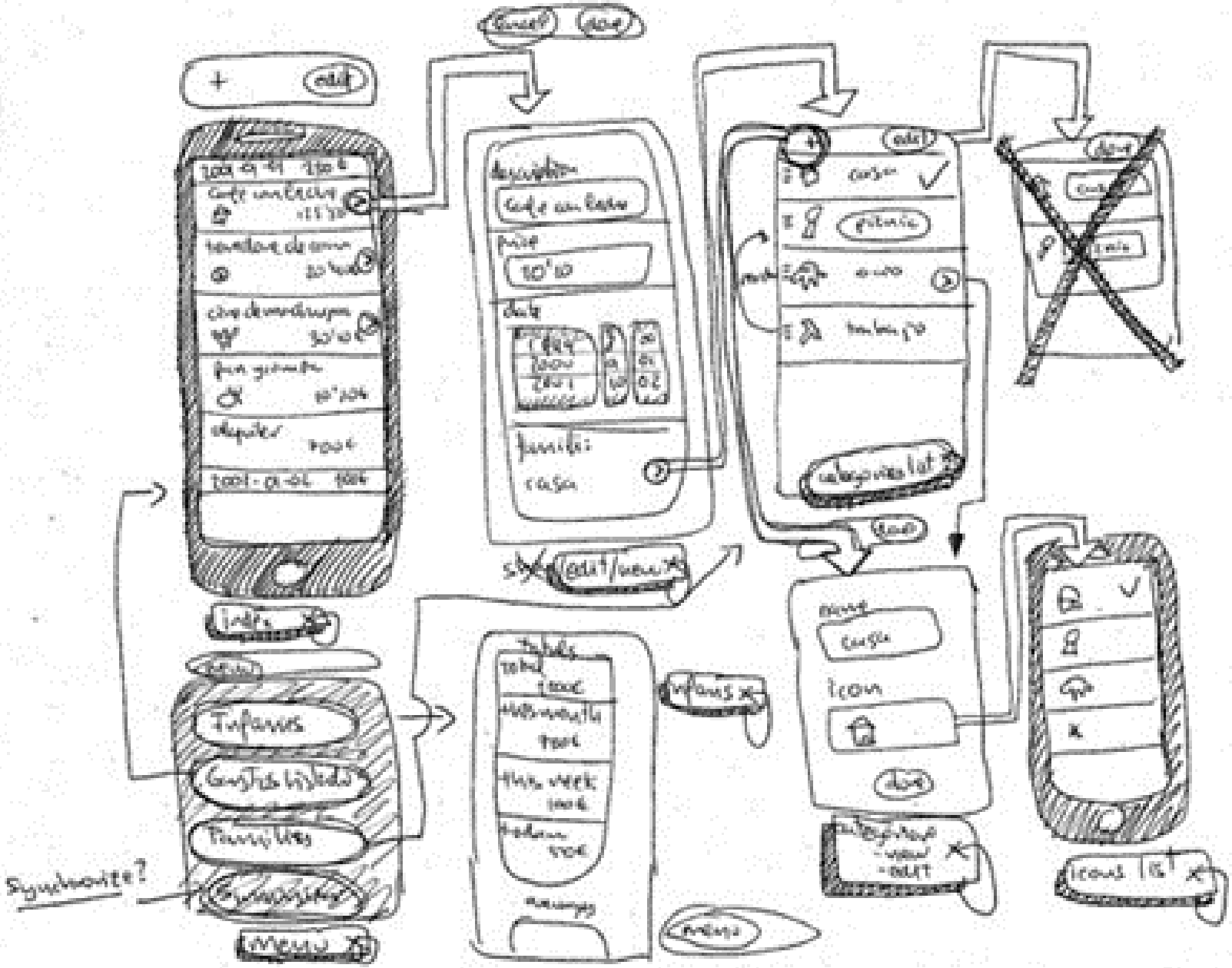
- Executable prototypes
  - as software constructed using a high-level programming language
  - a programming language or other rapid development environment is used to develop an executable prototype
- Non-executable prototypes
  - as paper prototypes and other mock-ups of the system.
  - a paper mock-up of the system is developed and used for system experiments

LOW-FIDELITY PROTOTYPING



- For refining unclear requirements, Uses a medium which is unlike the final medium, e.g. paper, cardboard
- Is informal, quick, cheap and easily changed (~10 sec)
- no one will mistake it for the finished product.
- Because it is easy to change, stakeholders are ... iterate, experiment, and investigate alternati...
- Created in order to prove, communicate and ...
- Examples:
  - sketches of screens,
  - task sequences, etc
  - 'Post-it' notes
  - storyboards

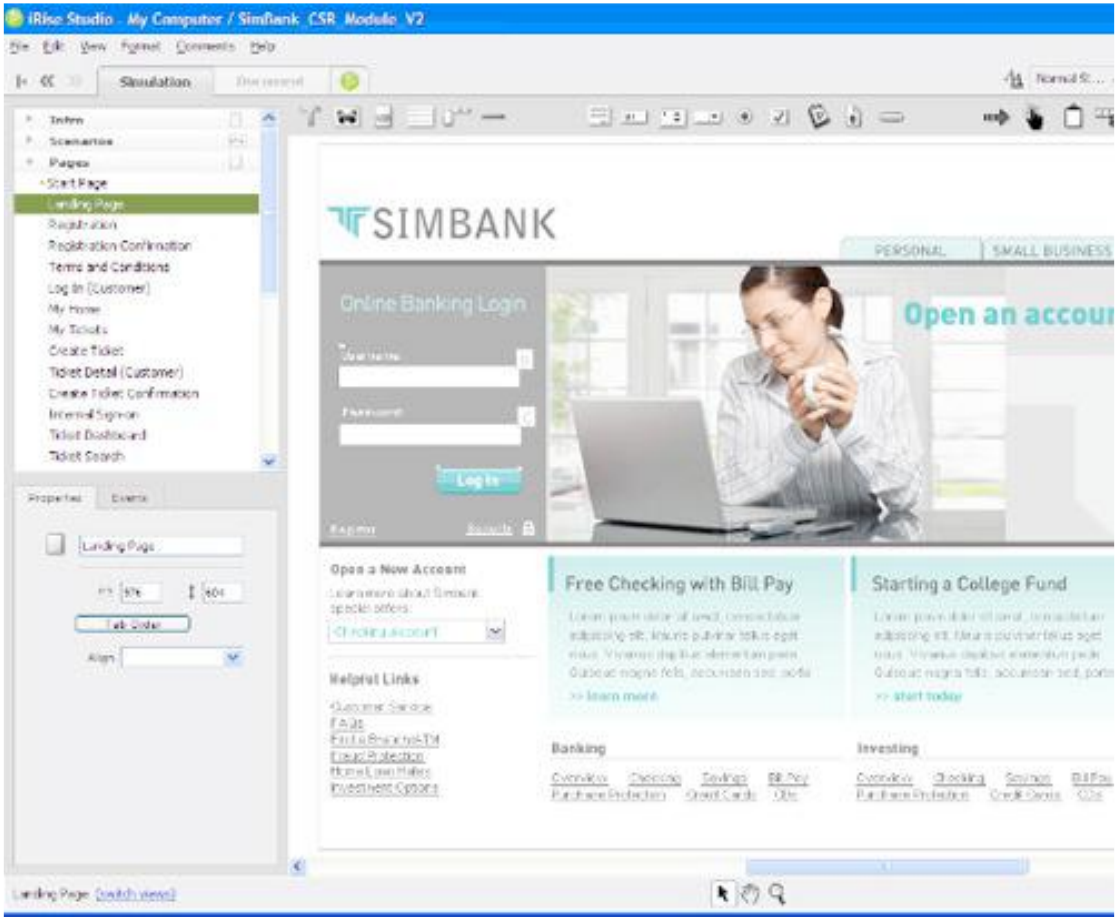
Low vs. High Fidelity Prototyping Debate, Rudd J., Stern K.,Isensee S., ACM Interactions, Jan. 1996



HIGH-FIDELITY PROTOTYPING

- Uses materials that you would expect to be in the final product (eg, software and hardware)
- Prototype looks more like the final system than a low-fidelity version.
- Danger that users think they have a full system…….

HI-FI PROTOTYPE



Requirements specification

- Detailing the characteristics of a requirements
  - To which use case /scenario belongs
  - Description
  - Dependencies on other requirements
  - Priority
  - Acceptance criteria
  - Which stackholder(s) provided the requirements
  - Which actor(s) use the requirement
  - Who specified the requirement
  - When last updated/added/deleted
  - Etc
- Lots of different templates available

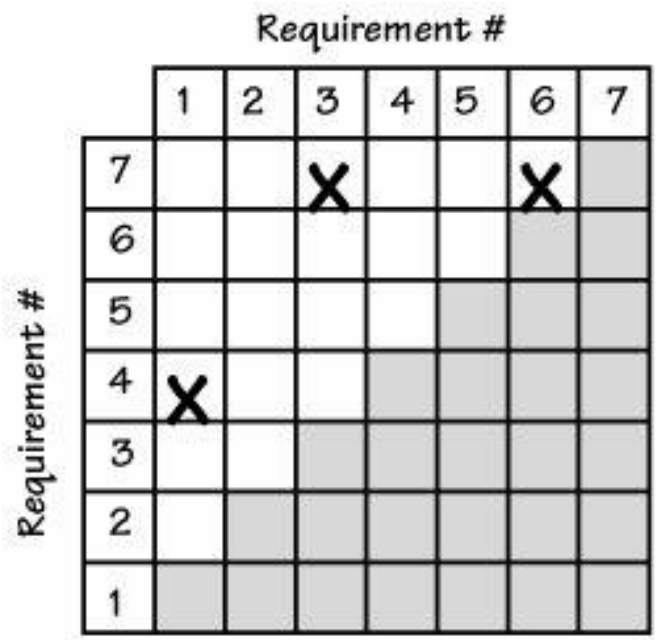
Priority

- From different perspectives
- How important for different stakeholders
- How difficult to implement (cost, time, etc)

Volere Prioritisation Spreadsheet											
Copyright c The Atlantic Systems Guild 2002											
Requirement/Product Use Case/Feature	Number	Factor - score out of 10	%Weight applied	Factor - score out of 10	%Weight applied	Factor - score out of 10	%Weight applied	Factor - score out of 10	%Weight applied		Total Weight
		Value to Customer	40	Value to Business	20	Minimise Implementation Cost	10	Ease of Implementation	30	Priority Rating	100
Requirement 1	1	2	0.8	7	1.4	3	0.3	8	2.4	4.9	
Requirement 2	2	2	0.8	8	1.6	5	0.5	7	2.1	5	
Requirement 3	3	7	2.8	3	0.6	7	0.7	4	1.2	5.3	
Requirement 4	4	6	2.4	8	1.6	3	0.3	5	1.5	5.8	
Requirement 5	5	5	2	5	1	1	0.1	3	0.9	4	
Requirement 6	6	9	4	6	1.2	6	0.6	5	1.5	6.9	
Requirement 7	7	4	2	3	0.6	6	0.6	7	2.1	4.9	
Requirement											

Conflicting requirements

- Identify and keep track
- Try to understand and discuss with stakeholders
- A conflict matrix may help with the bigger picture

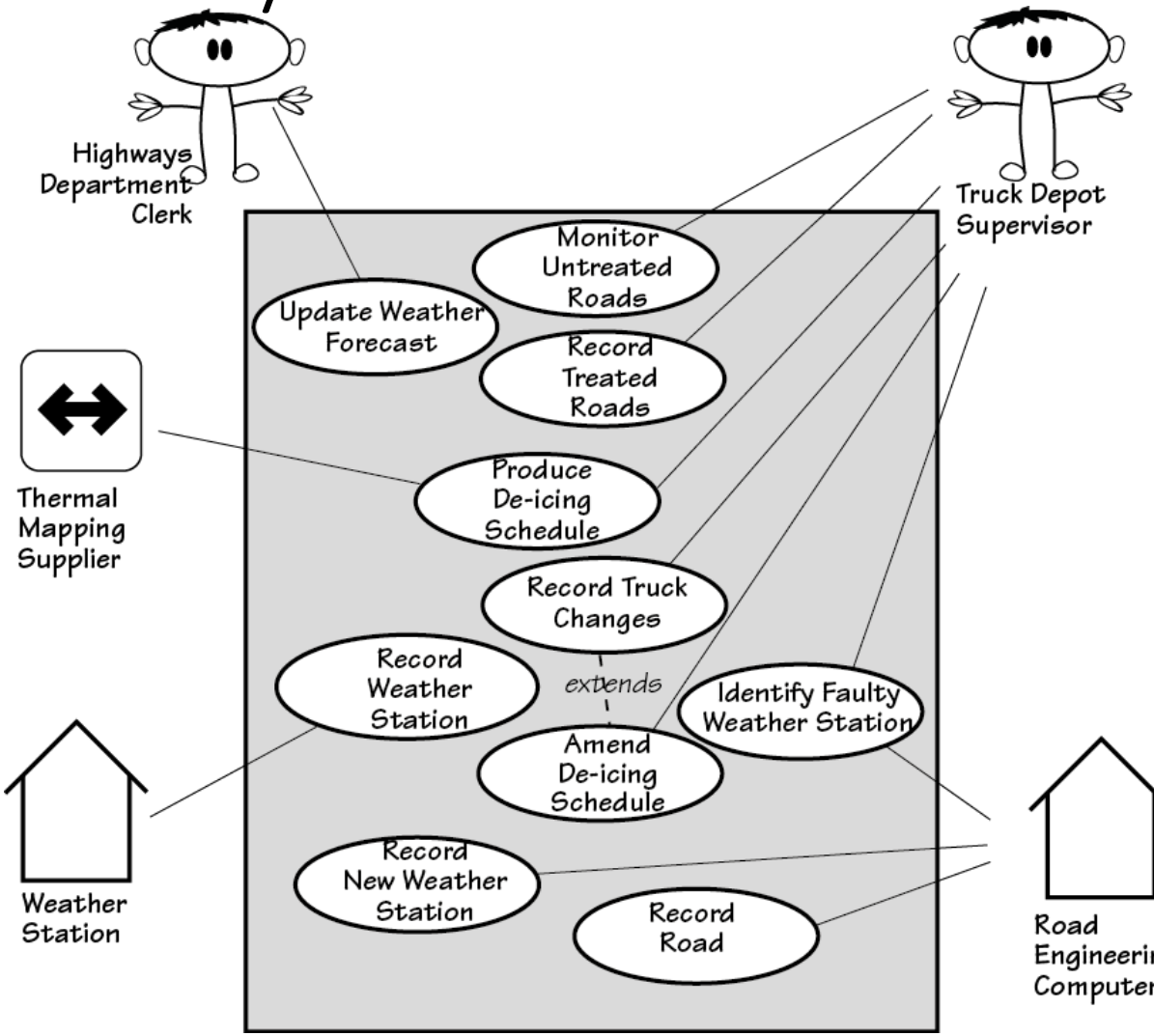


Requirements modeling

- An alternative (graphical) way do describe requirements
- *”All models are wrong, but some are useful”* (George Box)
- *”A picture is worth a 1000 words”* (Ch.Ph., more recently ….Mats Neovius)

- NOTE: we are representing the concepts of the system (domain models) **not** providing design models

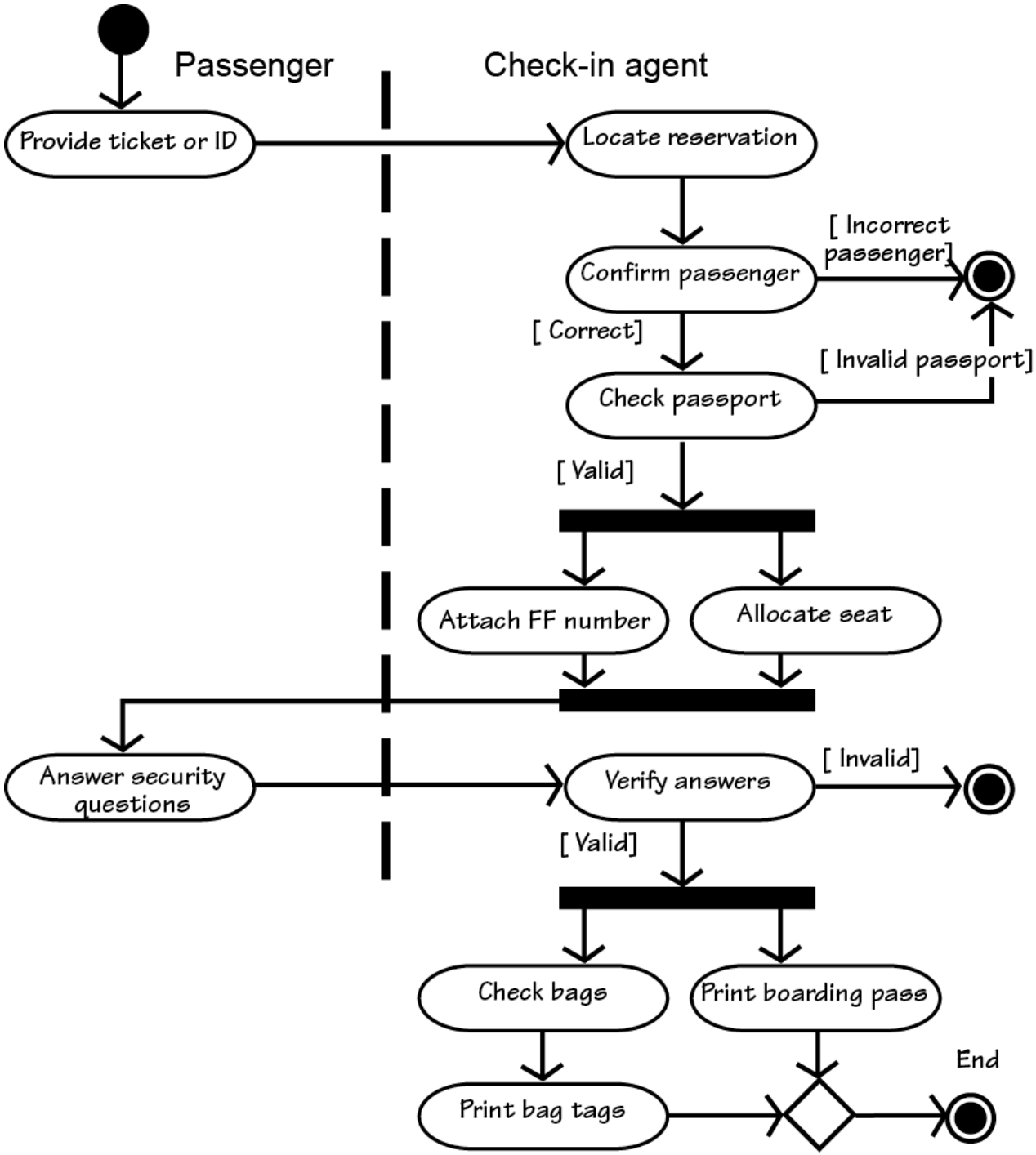
Capture use cases/scenarios



Specify Scenarios from User Stories

- A scenario is a story describing a business use case
- Typically as a number of steps (3-13)
- In a language and detail level acceptable to stakeholders
- Example: Customer:“I make sure I have **the right passenger and the right flight.** It would be pretty embarrassing to give away someone else's seat or to send a passenger to the wrong destination. Anyway, somehow I locate the passenger's flight record in the computer. If he has not already given it to me**, I ask for the passenger's passport.** I **check that the picture looks like the passenger and that the passport is still valid.**”
- ….

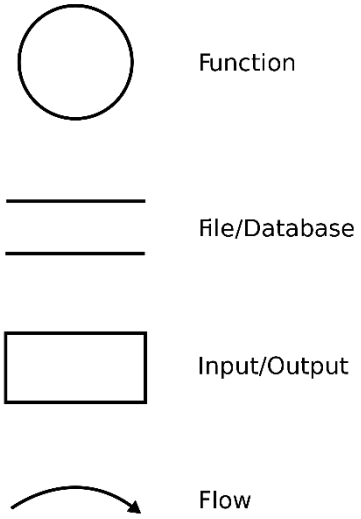
1. **Get the passenger's ticket or record locator.**
2. **Is this the right passenger, flight, and destination?**
3. **Check the passport is valid and belongs to the passenger.**
4. Record the frequent-flyer number.
5. Find a seat.
6. Ask security questions.
7. Check the baggage onto the flight.
8. Print and hand over the boarding pass and bag tags.
9. "Have a nice flight."

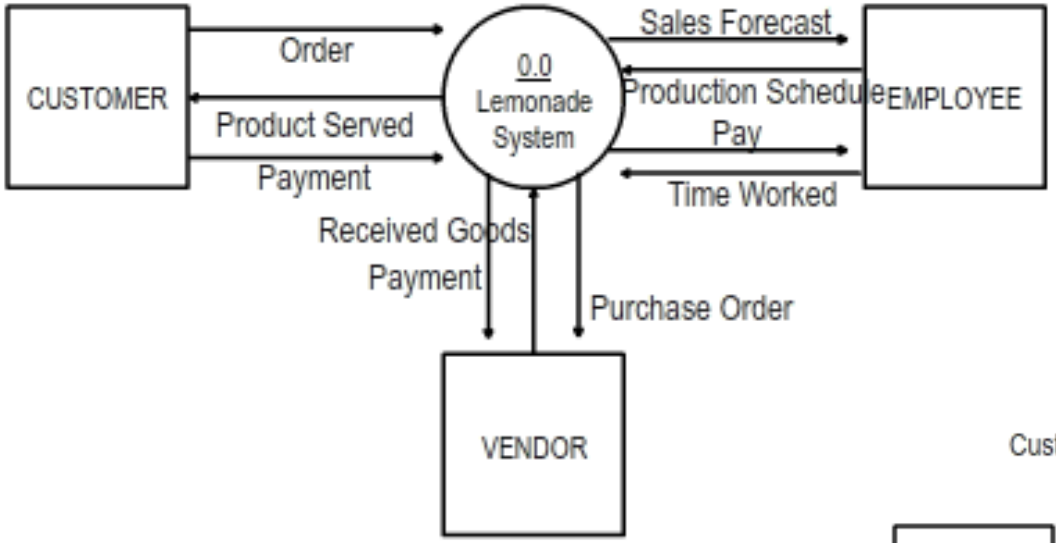

Passenger
Check-in agent
Provide ticket or ID
Locate reservation
Confirm passenger
[Incorrect passenger]
[Correct]
[Invalid passport]
Check passport
[Valid]
Attach FF number
Allocate seat
Answer security questions
Verify answers
[Invalid]
[Valid]
Check bags
Print boarding pass
Print bag tags
End

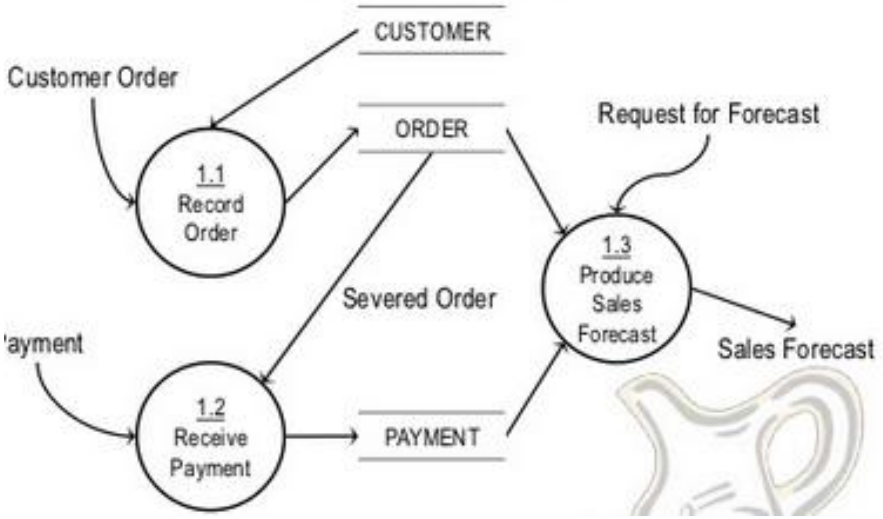
Other models can be used

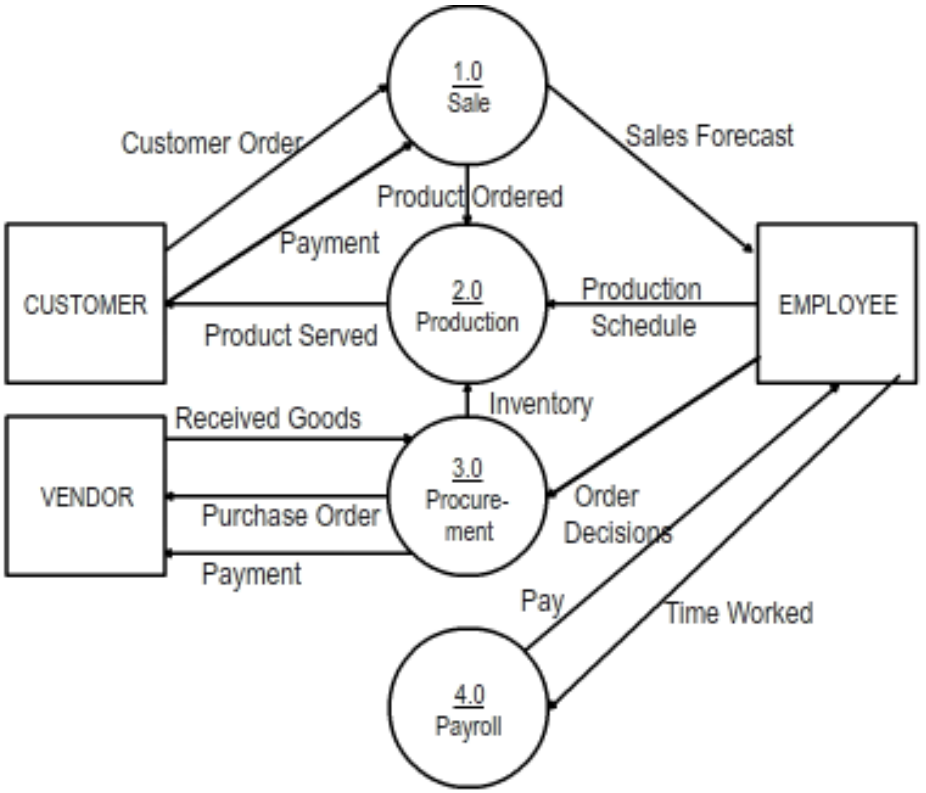
- Data flow diagrams
- Sequence diagrams
- Timming diagrams
- Class and object diagrams
- Etc.
-

An alternative:
Data Flow Diagram

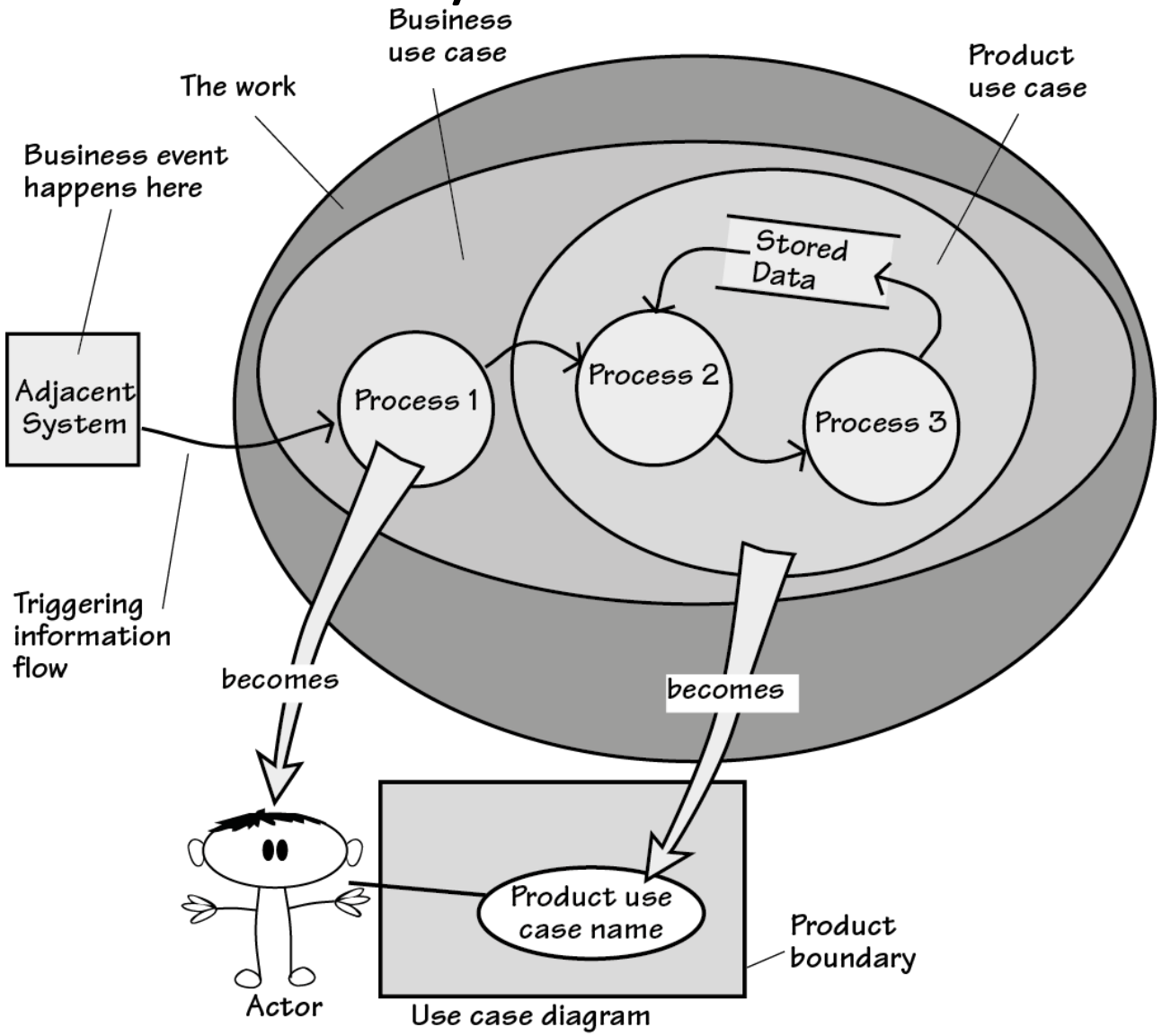




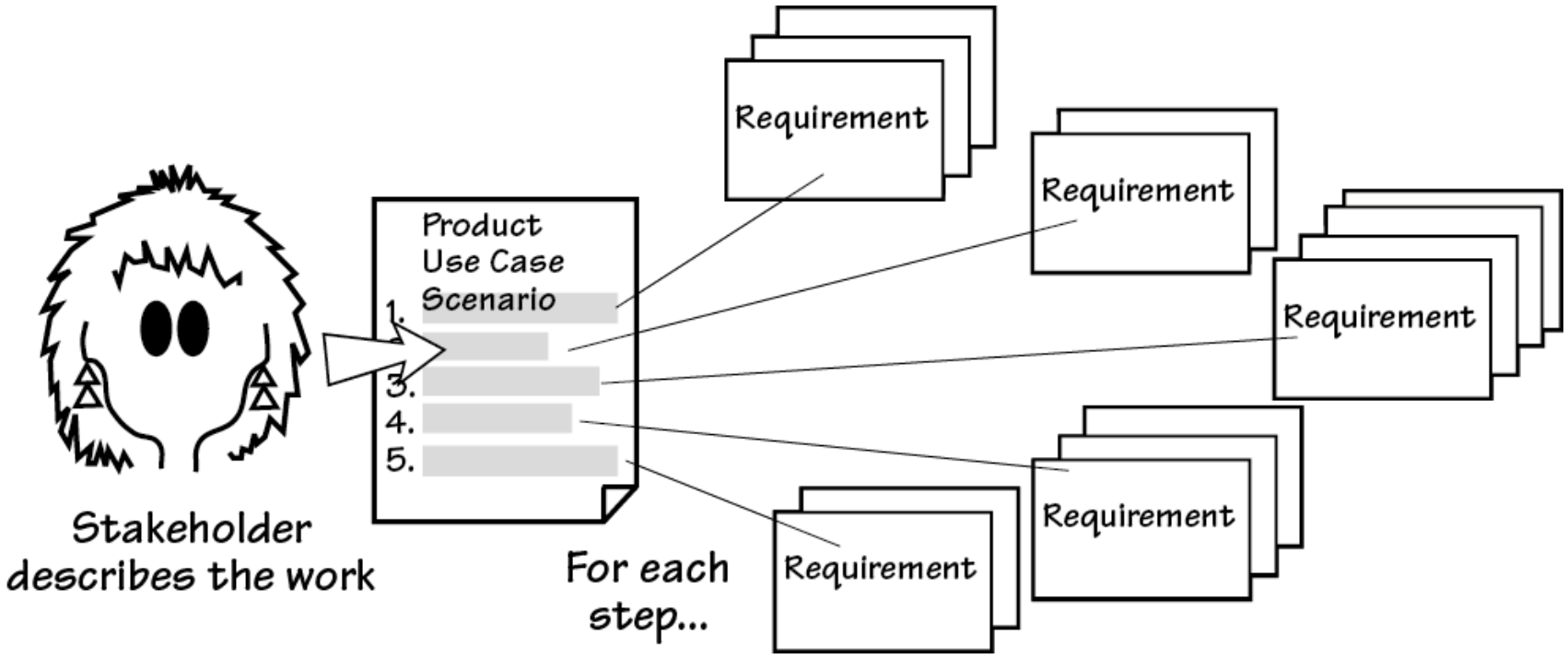




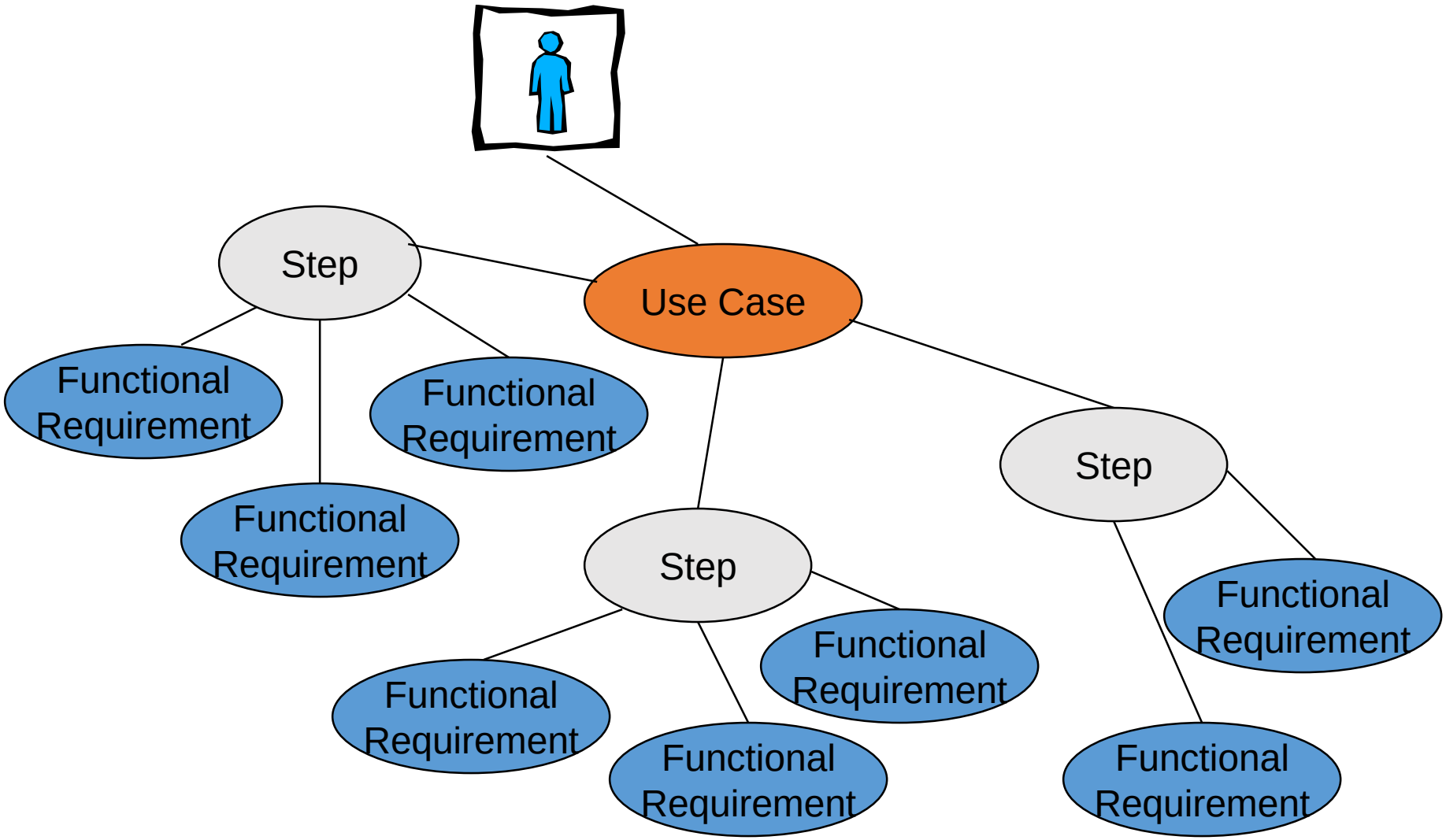
Border of the system



Functional Requirements



How to find functional requirements



Example

- For each step, ask, ***"What does the product have to do to complete this step?"***

- For example, the first step in the scenario is
  - ***Engineer provides a scheduling date and district identifier.***

- The first functional requirement to come from this step is fairly obvious:
  - ***The product shall accept a scheduling date.***

- Another requirement from the first step is
  - ***The product shall accept a valid district identifier.***

Requirements, Not Solutions

- There is a ***difference between a requirement and its solution***.
- It is important to your requirements discovery that you do not write solutions instead of requirements.

- Example:
  - ***"The product shall display pictures of goods for the customer to click on."***
  - !!! The requirements analyst **has assumed a screen, a picture, and ordering by clicking.**

- Here's the **correct way** to write this requirement:
  - ***"The product shall enable the customer to select the goods he wishes to order."***

Acceptance/Fit criterion

- Description: ***The product shall record the weather station readings.***

- Fit criterion: ***The recorded weather station readings shall match the readings sent by the weather station***

- This can be use later on for (acceptance) testing

NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirement Fit Criteria: Usability Requirements Example

Description: *The product shall be user friendly.*

Fit criterion: *New users shall be able to add, change and delete roads within 30 minutes of their first attempt at using the product.*

Non-functional Requirement Fit Criteria: Usability Requirements Example

Description: *The product shall be clear.*

Fit criterion: *Nine out of ten road engineers shall be able to successfully complete [list of selected tasks] after one day's training.*

Non-functional Requirement Fit Criteria: Usability Requirements Example

Description: *The product shall be clear.*

Fit criterion: *Nine out of ten road engineers shall be able to successfully complete [list of selected tasks] after one day's training.*

Result of your analysis work

- System proposal (Technical Document v1)
  - Requirements
  - Optionally models
- After that
  - Project management - evaluate and assign resources
  - Design
    - Choose different architectures to satisfy NFRs
    - Design GUI
  - Implement
    - Evaluate different techologies

Use the right method, technology, tool!!

Borrowed from M. Berglund, Ericsson, T3UC2007

Use the right method, technology, tool!!

Borrowed from M. Berglund, Ericsson, T3UC2007

Use the right method, technology, tool!!

Borrowed from M. Berglund, Ericsson, T3UC2007

Use the right method, technology, tool!!

Borrowed from M. Berglund, Ericsson, T3UC2007

Use the right method, technology, tool!!

Borrowed from M. Berglund, Ericsson, T3UC2007

Use the right method, technology, tool!!

Borrowed from M. Berglund, Ericsson, T3UC2007

Use the right method, technology, tool!!

Borrowed from M. Berglund, Ericsson, T3UC2007

Use the right method, technology, tool!!

Borrowed from M. Berglund, Ericsson, T3UC2007