



LUND
UNIVERSITY

ETSN15 Kravhantering

Lecture 2 (2025):

Decide on product idea, work on PM v1

Elicitation: Lau:8

Prioritization: [PRIO]

Björn Regnell

<http://www.cs.lth.se/krav>

Gästföreläsare

- L6: Dr. Johan Linåker om öppen källkod
- Fråga om ert intresse för **extra** sem. om RE+AI?
 - RE4AI
 - AI4RE
 - Doktorand Matthias Wagner om AI-reglering
 - Förslag: W4 eller W5? tisdag kl 15
- Förberedelse inför gästföreläsningar:
 - diskutera i projektgruppen intressanta frågor inför Q&A

Draft Product Names 2025

Alfa: squolgame

Beta: edubee

Gamma: easytrip

Delta: immune

Talk to me during the brake if you want to product idea.





You should bring PM v1 to tomorrow's exercise and start working on your context diagram.

reqtbox

a summary of important areas
in software requirements engineering

Requirements cheat sheet: reqtbox "kravboxen"

<https://github.com/lunduniversity/reqeng/tree/master/reqtbox>

context – who 	intentions – why 
requirements – what 	delivery – when 

reqtbox/who:context [spsi]

stakeholders incl. human users	our product
other systems	interfaces and protocols

reqtbox/why:intentions [gprc]

goals +-	priorities $\frac{1}{2}$
risks % * ☹️	commitments §\$

reqtbox/what:requirements [fdqt]

functionality	data
quality	tests

reqtbox/when:delivery [rrcr]

road-map and strategy	resources
constraints	release plan

reqtbox/{who,why,what,when} [cird/{spsi, gprc, fdqt, rrcr}]

context – who



stakeholders incl. human users	our product
other systems	interfaces and protocolls

intentions – why



goals	priorities
risks	commitments

requirements – what



functionality	data
quality	tests

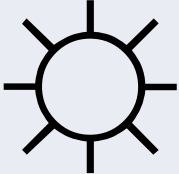



delivery – when



road-map and strategy	resources
constraints	release plan

reqt process box

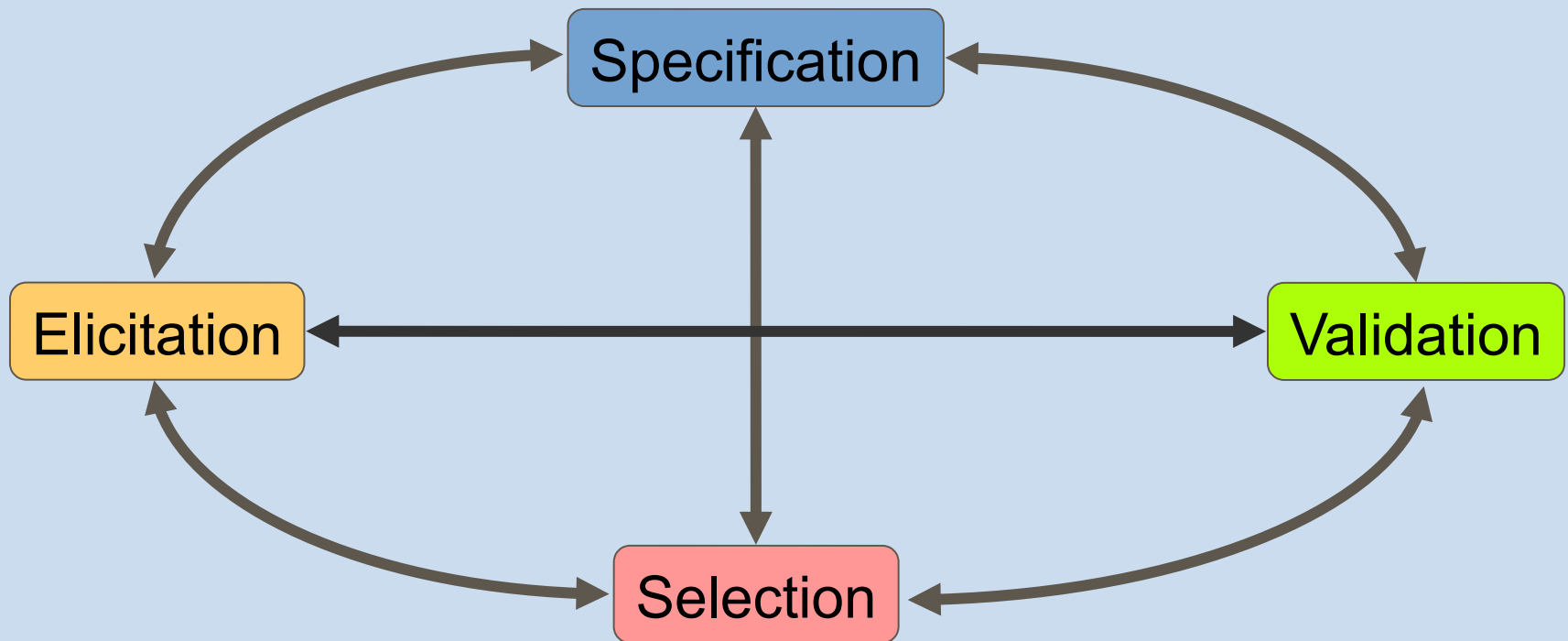
{learn, model, check, decide} [esvs]

elicitation – learn 	specification – model 
validation – check 	selection – decide 



Requirements engineering core activities

you need to iterate & work in parallel



Evolving mix of levels of detail & quality in continuous requirements engineering

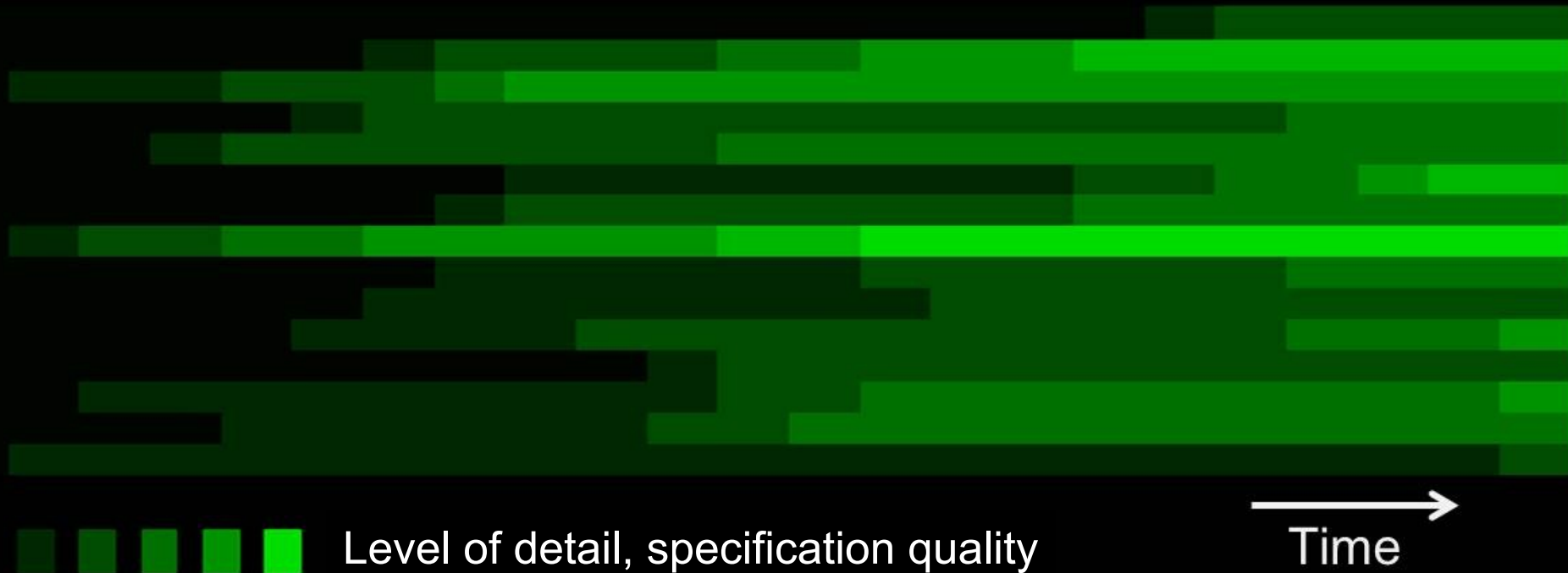
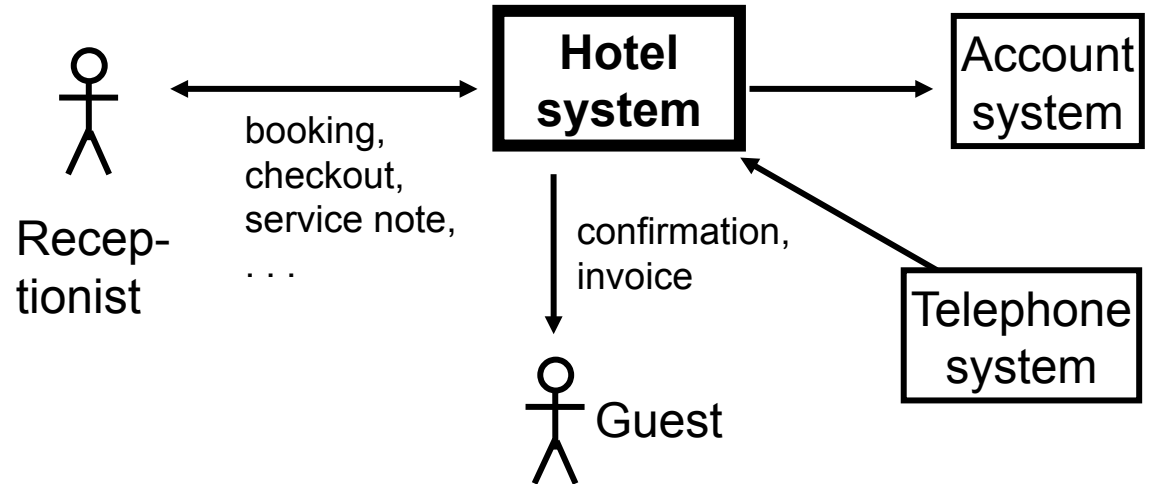


Fig 3.2 Context diagram

R1:

The product shall have the following interfaces:



R2 ??:

The reception domain communicates with the surroundings in this way:

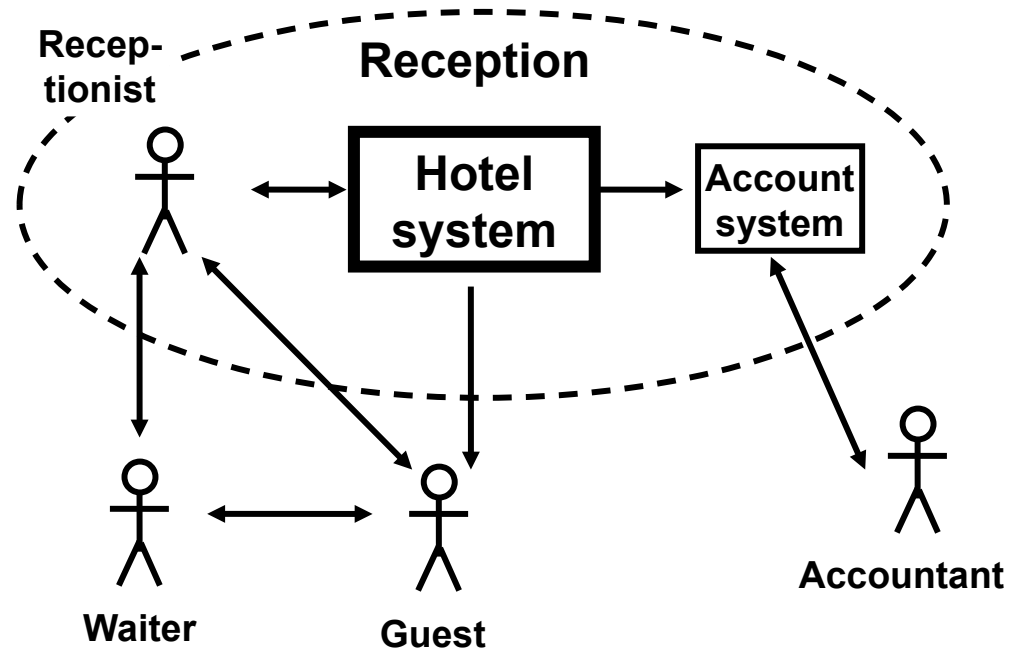
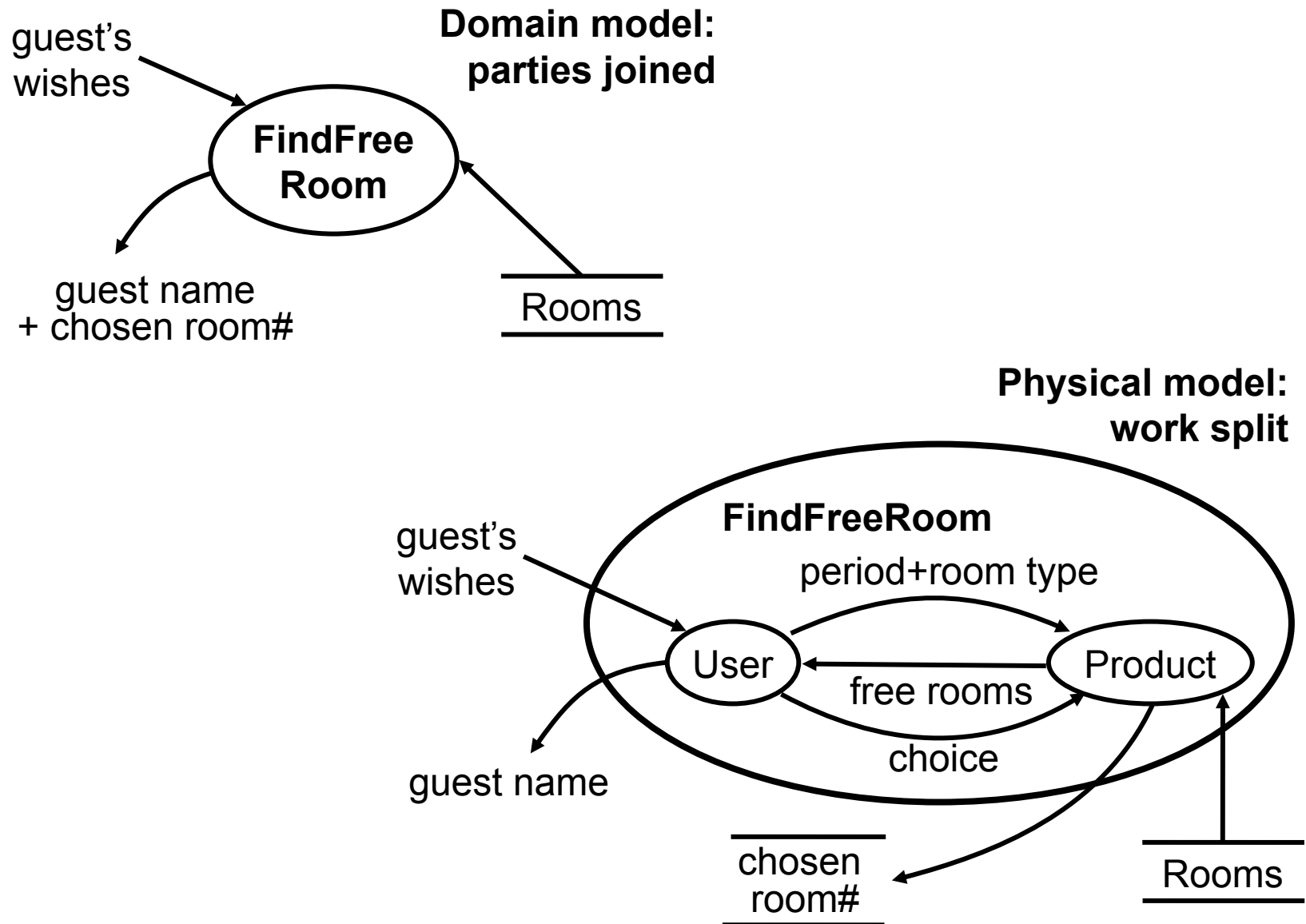


Fig 3.1 Human-computer - who does what?



Some different project types



- **Product Development** – **Produktutveckling** för öppen marknad, t.ex. inbyggda system, generella appar för en marknad (COTS development), etc.
- **Commercial Off-The-Shelf software purchase** (COTS purchase)
 - **Produktinköp** av generisk (hyll-) programvara
- **Time & Materials** – Utveckling på **löpande räkning**, rörligt pris
- **Customization** – **Kundspecifik anpassning** av generisk programvara
- **(Request for) Tender** – **Anbudsförfrågan**
 - ◆ Customer specific: för upphandling av kundspecifik utveckling
 - ◆ Generic (COTS): för upphandling av generisk programvara
- **Contract development** – **Kontraktsbaserad utveckling** med fast/rörligt pris
- **Sub-contracting** – **Underleverantörskontrakt** med fast/rörligt pris
- **In-house** – **Internutveckling** för egna behov
- **Unknown, pre-study** – Okänd, förstudie för att utreda lämplig projekttyp
- **Hybrid** – kombinationer av ovanstående
- ... ?

The **context is critical** to how you do requirements engineering!



The goal-design scale:

Goal→Domain→Product→Design



Overview of techniques for functional requirements (Swedish terms)

Datakravstilar:

- Datamodell
(=E/R-diagr.)
- Dataordlista
- Reguljära uttryck
- Virtuella fönster

Funktionella kravstilar:

- Kontextdiagram
- Händelse- & Funktionslistor
- Produktgenskapskrav
- Skärmbilder & Prototyper
- Uppgiftsbeskrivningar
- Egenskaper från uppgifter
- Uppgifter och stöd
- (Levande) Scenarier
- Högnivåuppgifter
- Användningsfall
- Uppgifter med data
- Dataflödesdiagram
- Standardkrav
- Krav på utvecklingsprocessen

Funktionella detaljer:

- Enkla och sammansatta funktioner
- Tabeller & Beslutstabeller
- Textuella processbeskrivningar
- Tillståndsdigram
- Övergångsmatriser
- Aktivitetsdiagram
- Klassdiagram
- Samarbetsdiagram
- Sekvensdiagram

Speciella gränssnitt

- Rapporter
- Plattformskrav
- Produktintegration
- Tekniska gränssnitt

First read the "gray box" of all styles so that you understand what they are about and their pros and cons. Then read in depth as needed.

Elicitation: [Lau:8]

Get out there and dig up reqts!

”You *cannot* sit in your office and produce requirements based on intuition and logic.

You have to **discover** the non-trivial requirements from users and other stakeholders.”

[Lausen, page 42]

⇒ PROVOKING AN
UNDERSTANDING

Fig 1.6B Ask “why”

Neural diagnostics

System shall have mini keyboard with start/stop button, . . .

Why?

Possible to operate it with “left hand”.

Why?

Both hands must be at the patient.

Why?

Electrodes, bandages, painful . . .

Deep domain knowledge is critical to successful requirements engineering!!





Why+How+Example

- * **Feature:** navigate **has**
 - * **Why:** Measuring neural response is a bit painful to the patient. Electrodes must be kept in place ... So both hands should be at the patient during a measurement.
 - * **Spec:** It shall be possible to perform the commands start, stop, ... with both hands at the patient.
 - * **Example:** Might be done with mini keyboard (wrist keys), foot pedal, voice recognition, etc.

Why is elicitation so challenging in real projects?



Fig 8.1 Elicitation issues



**Should be simple.
Ask stakeholders what they need!**

Barriers:

Cannot express what they need
Cannot explain what they do and why
May ask for specific solutions
Lack of imagination - new ways
Lack of imagination - consequences

Conflicting demands
Resistance to change
Luxury demands
New demands once others
are met

Things to elicit - intermediate work products:

Present work, Present problems
Goals and critical issues
Future system ideas
Realistic possibilities

Consequences and risks
Commitment, Conflict resolution
Requirements, Priorities
Completeness

Fig 8.2 Elicitation techniques

	Present work Present problems Goals & key issues	Future system ideas Realistic possibilities Consequences &	Commitment Conflict resolution	Requirements Priorities Completeness
Stakeholder analysis (Group) interview Observation Task demo Document studies Questionnaires				
Brainstorm Focus groups Domain workshops Design workshops				
Prototyping Pilot experiments				
Similar companies Ask suppliers				
Negotiation Risk analysis Cost / benefit Goal-domain analysis Domain-reqs analysis				

Nuläge
Framtid
Åtagande&samsyn
Krav & deras värde

**Studier av & med (enskilda)
intressenter eller dokument**

Förberedda gruppaktiviteter

Exekvering av system

Omvärldsanalys

**Avvägningar, risker, analys
av kopplingar mellan nivåer**

Stakeholder analysis (intressentanalys)

Example: In-house (internprojekt)

- Sponsors – want value for their money
- Users at different departments
- Managers at different departments
- Authorities, security managers, accountants etc.
- System management and support,
- Other indirect stakeholders that may provide valuable input

Example: product development:

- Distribution channels and retailers
- Solution providers building on your product
- Competitors

Interviews in requirements elicitation

On or more stakeholders are interviewed by a requirements engineer (aka analyst)

Probably the most common elicitation method.

Reflect on pros and cons with ...

- Individual or group interviews?
- **Structured:** prepared questions and perhaps also response alternatives
- **Semi-structured:** some Q prepared, but freedom in order and depth
- **Unstructured:** no preconceived closed questions; open questions to start off: "What is your view on the system?"

Intervjuer för att elicitera krav

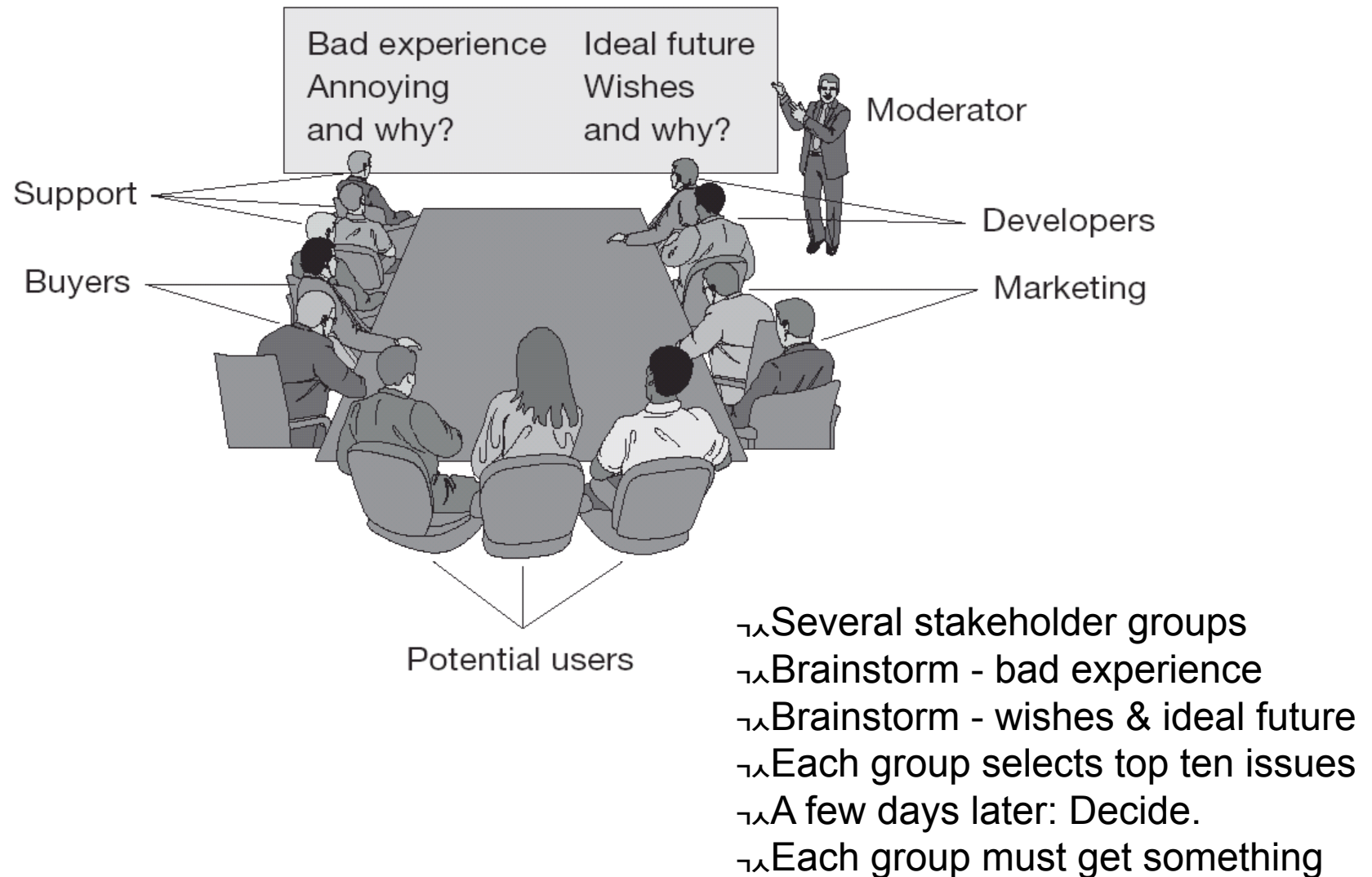
En eller flera intressenter tillfrågas av en kravingenjör.

Förmodligen den vanligaste metoden.

För och nackdelar med...

- Enskilda eller gruppvisa intervjuer?
- **Strukturerade:** förbestämda frågor, ev förbestämda svarsalternativ
- **Semi-strukturerade:** vissa frågor är förberett men frihet i ordning och djup
- **Ostrukturerade:** inga förberedda frågor alt. några få öppna frågor
"Berätta om din syn på systemet?"

Fig 8.4 Focus groups



Requirements Prioritization



Book chapter [PRIO]

"Requirements Prioritization",
Patrik Berander and Anneliese Andrews,
*Engineering and Managing Software
Requirements*,
Eds. A. Aurum and C. Wohlin, Springer,
ISBN 3-540-25043-3, 2005

Why prioritize?

- To focus on the most important issues
- To find high and low priority requirements
- To implement requirements in a good order
- To save time and money

Steps we need to do...

- Select prioritization aspects
- Select prioritization objects (e.g., features)
 - Example: Define features at high level that can be selected or de-selected independently
- Structure and grouping
- Do the actual prioritization
 - Decide priorities for each aspect and object
- Visualize, discuss, iterate...

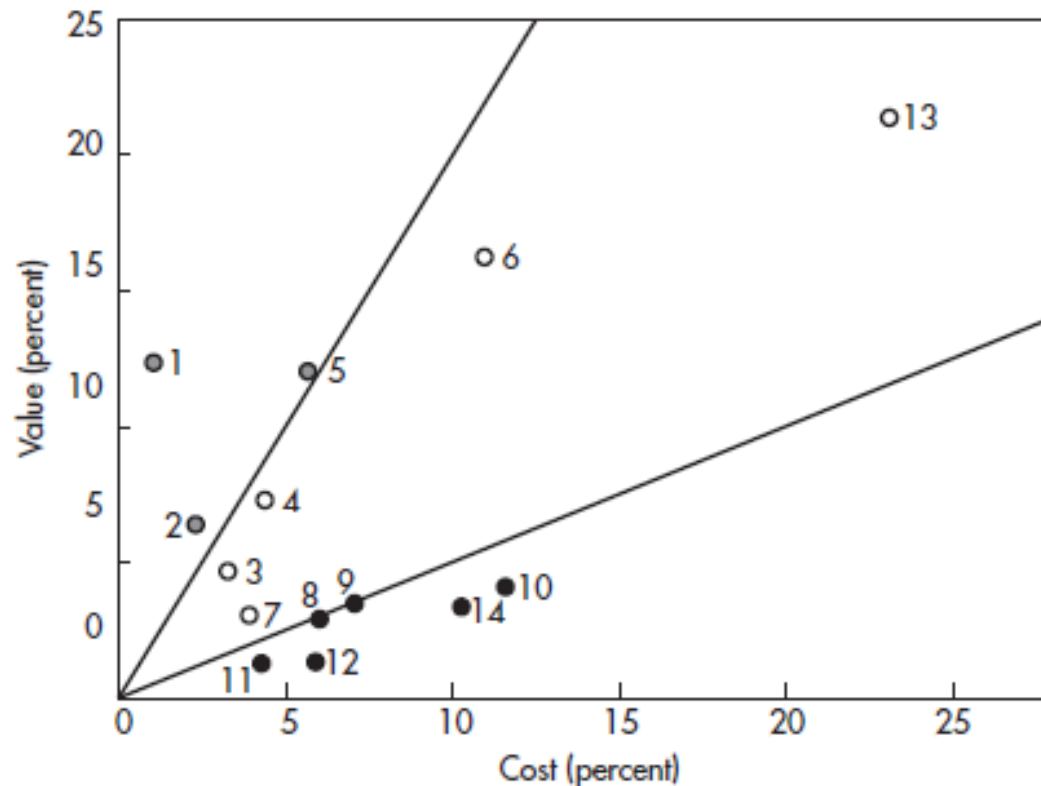
Prioritization challenges

- Finding a good abstraction level
- Combinatorial explosion
- Inter-dependencies
- Not easy to predict the future
- Power and politics

Prioritization aspects

- Example aspects:
 - Importance (e.g. financial benefit, urgency, strategic value, market share...)
 - Penalty (e.g. bad-will if requirement not included)
 - Cost (e.g., staff effort)
 - Time (e.g., lead time)
 - Risk (e.g., technical risk, business risk)
 - Volatility (e.g. scope instability, probability of change)
- Other aspects: competitors, brand fitness, competence, release theme...
- *Combination* of aspects: cost vs. benefit, cost vs. risk, importance vs. volatility
- **Optimize**: minimize or maximize some combinations, e.g., cost vs. benefit

Combining two criteria: Example Cost vs Benefit



Karlsson, Joachim, and Kevin Ryan. "A cost-value approach for prioritizing requirements." *IEEE software* 14.5 (1997): 67-74.

When to prioritize?

- At decision points (toll gates), e.g.,
 - Project start
 - Start of construction
 - Release planning
 - Increment planning
- When big changes occur
- Regularly with **lagom** intervals

Who should prioritize?

- Find the right competence for the right aspect
 - **Developers** know about e.g., development effort and engineering risk
 - **Support organization** knows about e.g., customer value and penalty
 - **Marketing organization** knows e.g., about competitors
 - etc...

Prioritization techniques

- **Direct numerical assignment (grading)** [Lau 7.4]
 - Can be done using any scale (categorical, ordinal, ratio) depending on what the number actually means.
 - Quick & easy; **but**
 - a risk is that all reqs are deemed highly important as they are not challenged against each other
 - may be misinterpreted as ratio scale (even if "4" not necessarily is "twice as much" as "2" when using an ordinal scale).
- **Ratio scale 100\$-test**
 - Ratio scale, quick and easy, risk of shrewd tactics (listigt taktikspel)
- **Ordinal scale Ranking**
 - sorting, pairwise comparison, easy and rather quick
- **Top-ten (or Top-n)**
 - *Ordinal scale* if the top list is ranked or *Categorical scale* if grouping is not ranked;
 - very quick and simple, gives a rough estimate on a limited set of req
- Other methods, e.g. ratio scale Analytical Hierarchy Process (AHP)
 - Priority matrix calculation based on pairwise comparison, tool is needed, includes redundancy that *gives estimate of consistency*

[PRIO]

Typical industry praxis

- **Numerical assignment**, e.g., 1-5
 - See Lauesen, chapter 7.4
- **Problems**
 - Requirements are not confronted to each other
 - What should go out when new requirements wants in?
 - Everything tends to be important
 - decision avoidance



[PRIO]

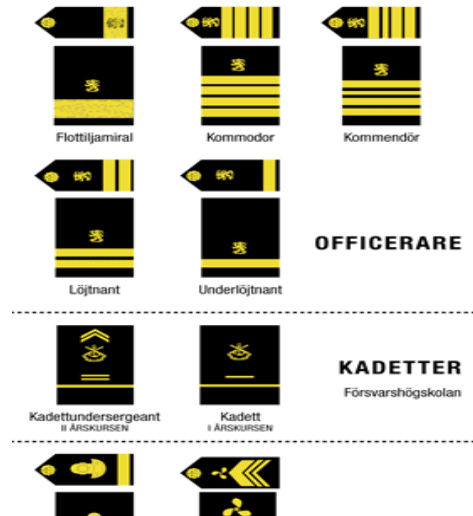
Prioritization scales



Categorization

e.g.: must,
ambiguous, volatile

Partition in groups
without greater-less
relations



Ordinal scale

e.g.: more
expensive,
higher risk,
higher value

Ranked list
 $A > B$



Ratio scale

ex: \$, h,
% (relative)

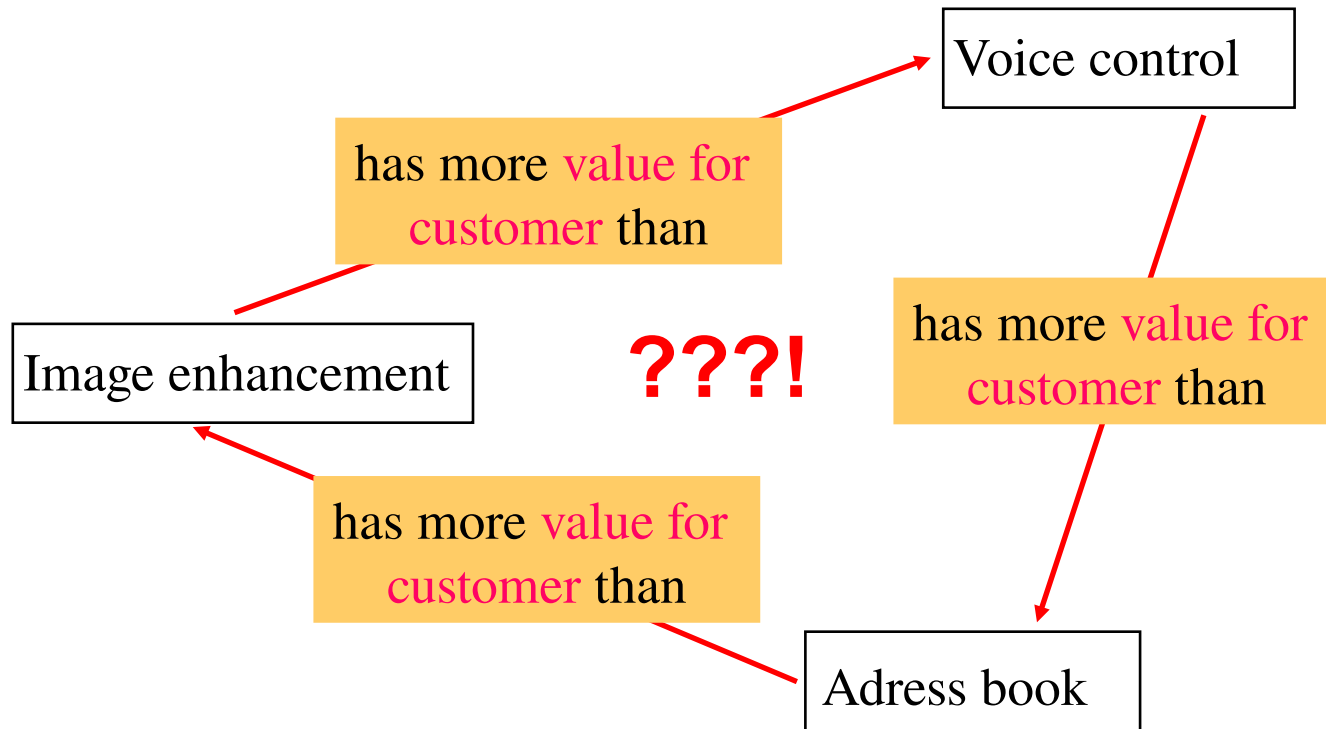
Numeric relations:
 $A = 2 * B$

[PRIO]

Combination of prioritization techniques

- Example of how prioritization techniques can be combined:
 - 1) Do a high-level **grouping**,
 - 2) Do **sorting** or **top-5** per group
 - 3) For some selected groups:
define sub-groups and use **100\$-method**

Tools can help find inconsistencies



This will be illustrated at Lab 1.

To do...

- Project Mission v1, bring to exercise E1
- Assign project roles according to project description
- Read Lau:1 on intro, Lau: 8 on elicitation, paper: PRIO
- Take a peek into Lau: 3.2 on context diagrams
- Sign up for labs in Canvas
- Exercise 1: work in your project group
 - Lauesen's book needed. esp. Chapter 1, context diagram
READ CHAPTER ONE BEFORE E1!
 - Bring your laptop.
- Book meeting with project supervisor W2, see Canvas
- Hand in PM v2 on Tuesday W2 (default deadline 23:59)