Ch. 3: Rs Elicitation & Analysis

to elicit means:
draw out
cause to come out
discover
reveal

criteria for the Requirements, once elicited:

- completeness
- consistency:
  without contradiction
- without ambiguity
- relevance:
  importance
- usefulness for practice:
  realistic

much negotiation involved

not only to **repair flaws**

by removing **contradiction**
by removing **ambiguity**

but also to **clarify**

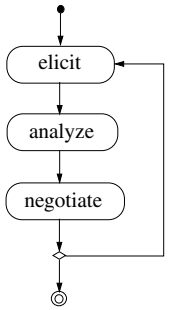
by removing **vagueness**

e.g. in expectation

by getting (better) **understanding** of
problem situation
kind of solution wanted

e.g.: what could be a solution
why does it work

therefore, this chapter is about the following



elicitation is **not**:

revealing by **asking** "them" **what they want**

instead it is:

revealing what "they" **would be wanting**
if they had a clear understanding

problem is: if we cannot ask **them**,
where can we find what we need to know
either the Rs
or the information to base
the clear understanding on

to that aim

elicitation addresses
4 dimensions / fields / main directions

4 dimensions of elicitation:

- application domain,
  general (background) knowledge

- problem situation
  specific knowledge, far more detailed
  than application domain knowledge

- business within which
  also concrete knowledge
  in this case of this one organization
  but on a rather general level
  business goal
  other systems contributing to business
  or this one system-to-be

- stakeholders'
  constraints, needs, work conditions
  concrete knowledge within business, e.g.
  kind of support expected (for concrete goals)

inherent problems

first of all with the 4 dimensions:

- application domain
  knowledge is distributed, inaccessible

- problem situation
  those who know are too busy

- business
  "unknown" politics, tactics, hidden agenda

- stakeholders
  imprecise, unrealistic, unstable,
  incomparable

second, there is an overall problem:

all kinds of change

so, for any structured Rs
elicitation, analysis, negotiation

- **flexibility**: to cope with change
- **sensitivity**: to have a good nose
  for what (probably) will change

are much needed

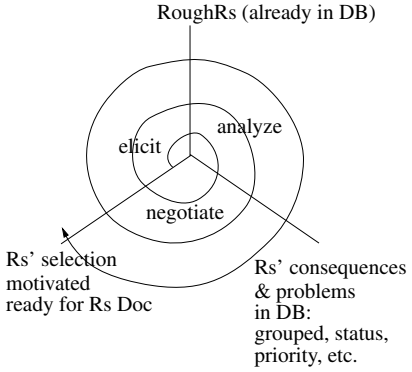
but again,
easy to say, hard to do

some research results concerning flexibility:
JIT techniques (Just In Time)

3.1. RE's Elicit-Analyse-Negotiate processes

characteristic is growth, incrementality

so process is strongly iterative, interleaved



during elicitation

4 main activities
("crossing over" the 4 dimensions)

- determine goals
  business
  problem outline
  budget
  schedule
  interoperability

- get general knowledge
  appl. domain
  organization
  existing system(s):
  those related
  those to be replaced

- organize knowledge
  - stakeholders
  - roles
  - prioritize goals / knowledge
  - group knowledge
  - filter knowledge
  - relate knowledge

- actual elicitation
  - stakeholders' Rs
  - problem domain Rs
  - domain Rs
  - organization Rs

  moreover:
  - explicit as well as implicit
  - global as well as detailed

during analysis

- check
  - necessity
  - consistency: overlap? --> contradiction?
  - completeness
  - feasibility

- specify model
  - declarative wrt system-to-be

  and less standard (IOPENER!)
  - operational wrt business-as-is
  - operational/declarative wrt business-to-be

among other things, it results in problematic Rs
superfluous, irrelevant, conflicting,
incomplete, missing, unfeasible, unrealistic,
ambiguous, vague, ... , too whatever

during negotiation

concerning the problematic Rs mainly:

- discuss
- clarify
- prioritize
- agree
- decide

NB wrt consistency there exists

- (in)consistency management
  e.g. delay of (enforcing) solution
  woven into lifecycle

is imaginable too
- for other kinds of problematic Rs

3.2. Elicitation Techniques

--> structuring mechanisms:

- partitioning / grouping / aggregation
- abstraction / generalization
- projection / view(point)s / perspectives

this is architecture-like (logical)
also related to OO (Object-Orientation)

inherent problems
- time: people have other priorities
- REngineers: not prepared for
  - domain / organization / humans
    in other words: (too technical) nerds
- stakeholders not willing
  - passive / active opposition
- appl domain knowledge
  - subtle / implicit / difficult to explain
- organization can be informal --> unclear

--> interviews

- closed vs open
  - standard questions
  - "just tell"
  - mix of these

- open-minded / good listener

- always a starting point

- check/improve by feedback

--> scenarios: like use cases

- global state descriptions
  - e.g. before / after
    - perhaps some intermediate

- normal flow of events
  - variants
  - exceptions
  - other ongoing activities / scenarios
    - --> interaction !!

overall inherent problem:
- integration of the parts / partial insights

also this is consistency

--> soft system methods

NB
some relation with Integration-Orientation,
see *

it is a process-like description:

* - assess problem situation
* - describe problem situation
* - give a system-to-be description; viewpoints
* - integrate viewpoint descriptions
* - compare as-is vs to-be
* - evaluate differences
* - identify changes
- recommend migration activities

often there is
some architectural pattern for organizations

workflow: pipe and filter pattern
- for any business activity

examples:

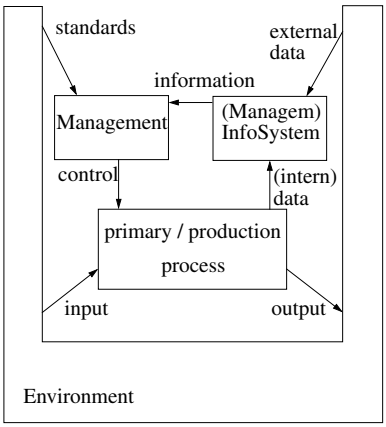
all waterfall-like process descriptions, e.g.

- complete lifecycle process
  - of software engineering

- complete RE process as in chapter 2

- complete elicitation&analysis process
  - as in beginning of this chapter

general, managed organizations:
embedded feedback loop pattern
(Dutch: besturingsparadigma)



some remarks:

- this is not a UML diagram

- information and control are both data too

- standards and external data are optional

- pattern is recursive: it can re-occur
  inside primary process
  inside information system (IS)
  inside management

- ICT can have overlap with Management and with primary process (not only with IS)

wrt above embedded feedback loop pattern:

ASSIGNMENT1:
1. give a class diagram for it
2. give a collaboration diagram for it
3. give an activity diagram for it
4. give a sequence diagram for it
(NB: in UML 2.0)

ASSIGNMENT2:
specialize / refine the above four UML diagrams towards the organization you work for (or have worked for most recently);
in the fourth case you might involve an interaction overview diagram

ASSIGNMENT3:
make sure you have incorporated details in order to express a soft system-like / Integration-Oriented approach

always: EXPLAIN

remarks wrt soft systems:

- take people serious

- invest in being trusted

- details

- non-standard approaches

- choose views (location, role, activity, ...)

- integrate

- confront / evaluate
  internal: stakeholder
  external: outsider

--> Reuse

- problem domain

- look&feel

- organization policy
  aspects like:
  security
  auditibility
  authorization
  back-up
  keeping informed
  process support
  ...

but always: new Rs, how do they fit in above

3.3. Prototyping

2 kinds:

- throw away
  the difficult Rs mainly

- evolutionary
  starting with the easy Rs
  gradually more complex Rs

good for (overall) Rs like
- feasibility
- usefulness
  both wrt cost-profit balance

- interfaces
- acceptance test planning
- consistency
- completeness (although being incomplete)

no (so) good for
- performance
- reliability
- ...
  "aspects":
  general, "cross-cutting" / "entangled" lines of interest

additional drawback:
  additional costs
  - training in using prototype
  - development
  - misleading because of
    being inherently incomplete

implementation of prototype:

- on paper
  parts of Rdoc, e.g. the windows

- wizard of Oz
  human simulating software

- automated
  - 4GL
  - visual
  - internet (Java + JBeans +)
  - in general, all new
    high-level programming approaches
    (scripting, mark-up, ...)
    e.g. where COTS can be integrated
    (Components Off The Shelf)

4.4. Rs Analysis and Negotiation

important observation:

analysis **versus** validation:

analysis:
- REngineer herhimself
- Rs still unfinished

validation:
- other reviewers
- Rs ready

on the other hand:

during validation much analysis is to be redone

checklist:
  restricted as well as **global**

(e.g.)

- ambiguity
  can be very expensive

- testable
  otherwise one can never know

- more than 1 (in "one R")
  a hidden R is risky

- design-like
  premature: unnecessarily restricting
  the possible solutions

between the various Rs:

interaction / dependency matrix:

conflict (guessed): 1
overlap: 1000
independent: 0

adding a row:
still discriminates between the 3 categories

negotiation is needed
if two Rs conflict

it DOES NOT MEAN two stakeholders have conflict

BUT two stakeholders have different responsibilities (even: 1 stakeholder with 2 roles)

meeting in order to:

- explain
  from all relevant sides

- discuss
  by all parties

- prioritize
  manager is NOT the overall boss here

- decide
  register the arguments

after that:

- remove or change / reformulate
- re-elicit
- re-analyze

so we see a cycle

still missing as important part of analysis
(and elicit and negotiation)

modelling

- architectural: as bridge towards design

- specifying a certain problem situation

and conform Integration-Orientation:
what is known about organization & end
can and should be specified

so

- IT's outer world
  business: structure, goal, activity, interaction
  environment (new forms of CRM):
  client: structure, interest, activity, interaction