Fundamentals of RE

Chapter 2
Domain Understanding & Requirements Elicitation

Chap. 1: RE products and processes

Chap. 2: Elicitation techniques

alternative options

consolidated requirements

start

agreed requirements

documented requirements
A great deal of knowledge acquisition is involved: as introduced in Chapter 1 ...

- Studying the system-as-is
  - Business organization: structure, dependencies, strategic objectives, policies, workflows, operational procedures, ...
  - Application domain: concepts, objectives, tasks, constraints, regulations, ...
  - Analysis of problems with system-as-is: symptoms, causes, consequences
- Analyzing technology opportunities, new market conditions
- Identifying the system stakeholders
- Identifying improvement objectives: organizational & technical constraints on system-to-be; alternative options for satisfying objectives, for assigning responsibilities; scenarios of hypothetical software-environment interaction; requirements on software, assumptions on environment

Domain analysis & requirements elicitation: outline

- (1) Identifying stakeholders & interacting with them
- (2) Artifact-driven elicitation techniques
  - Background study
  - Data collection, questionnaires
  - Repertory grids, card sorts for concept acquisition
  - Scenarios, storyboards for problem world exploration
  - Prototypes, mock-ups for early feedback
  - Knowledge reuse: domain-independent, domain-specific
- (3) Stakeholder-driven elicitation techniques
  - Interviews
  - Observation and ethnographic studies
  - Group sessions
Stakeholder analysis

- Stakeholder cooperation is essential for successful RE
  - Elicitation = cooperative learning
- Representative sample must be selected to ensure adequate, comprehensive coverage of the problem world
  - dynamic selection as new knowledge is acquired
- Selection based on ...
  - relevant position in the organization
  - role in making decisions, reaching agreement
  - type of contributed knowledge, level of domain expertise
  - exposure to perceived problems
  - personal interests, potential conflicts
  - influence in system acceptance

Knowledge acquisition from stakeholders is difficult

- Distributed sources, conflicting viewpoints
- Difficult access to key people & data
- Different background, terminology, culture
- Tacit knowledge, hidden needs
- Irrelevant details
- Internal politics, competition, resistance to change, ...
- Personnel turnover, changes in organization, in priorities, ...

⇒ Needed:
  - Communication skills: for talking to, listening from diverse people
  - Trust relationship
  - Knowledge reformulation & restructuring (review meetings)
(2) Background study

- Collect, read, synthesize documents about...
  - the organization: organizational charts, business plans, financial reports, meeting minutes, etc
  - the domain: books, surveys, articles, regulations, reports on similar systems in the same domain
  - the system-as-is: documented workflows, procedures, business rules; exchanged documents; defect/complaint reports, change requests, etc.

- Provides basics for getting prepared before meeting stakeholders => prerequisite to other techniques

- Data mining problem: huge documentation, irrelevant details, outdated info

- Solution: use meta-knowledge to prune the doc space
  - know what you need to know & what you don’t need to know

Data collection

- Gather undocumented facts & figures
  - marketing data, usage statistics, performance figures, costs, ...
  - by designed experiments or selection of representative data sets from available sources (use of statistical sampling techniques)

- May complement background study

- Helpful for eliciting non-functional reqs on performance, usability, cost etc.

- Difficulties:
  - Getting reliable data may take time
  - Data must be correctly interpreted
Questionnaires

- Submit a list of questions to selected stakeholders, each with a list of possible answers (+ brief context if needed)
  - **Multiple choice** question: one answer to be selected from answer list
  - **Weighting** question: list of statements to be weighted...
    - qualitatively ('high', 'low', ...), or
    - quantitatively (percentages)
      to express perceived importance, preference, risk etc.
- Effective for acquiring subjective info quickly, cheaply, remotely from many people
- Helpful for preparing better focused interviews (see next)

Questionnaires should be carefully prepared

- Subject to ...
  - multiple **biases**: recipients, respondents, questions, answers
  - unreliable info: misinterpretation of questions, of answers, inconsistent answers, ....

⇒ **Guidelines for questionnaire design/validation:**
  - Select a representative, statistically significant sample of people; provide motivation for responding
  - Check coverage of questions, of possible answers
  - Make sure questions, answers, formulations are unbiased & unambiguous
  - Add implicitly redundant questions to detect inconsistent answers
  - Have your questionnaire checked by a third party
Card sorts & repertory grids

♦ **Goal:** acquire further info about concepts already elicited

♦ **Card sort:** ask stakeholders to partition a set of cards ...
  - Each card captures a concept textually or graphically
  - Cards grouped into subsets based on stakeholder's criteria
  - For each subset, ask...
    - implicit shared property used for grouping?
    - descriptive, prescriptive?
  - Iterate with same cards for new groupings/properties

♦ **Example:** meeting scheduling system
  - Iteration 1: “Meeting”, “Participant” grouped together
    => “participants shall be invited to the meeting”
  - Iteration 2: “Meeting”, “Participant” grouped together
    => “participant constraints for the meeting must be known”

Card sorts & repertory grids (2)

♦ **Repertory grid:** ask stakeholders to characterize target concept through attributes and value ranges
  => concept-attribute grid
  e.g. (Date, Mon-Fri), (Location, Europe) for grid characterizing Meeting concept

♦ **Conceptual laddering:** ask stakeholders to classify target concepts along class-subclass links
  e.g. subclasses RegularMeeting, OccasionalMeeting of Meeting

✿ Simple, cheap, easy-to-use techniques for prompt elicitation of missing info

✿ Results may be subjective, irrelevant, inaccurate
Scenarios & storyboards

- **Goal**: acquire or validate info from concrete examples through narratives ...
  - how things are running in the system-as-is
  - how things should be running in the system-to-be

- **Storyboard**: tells a story by a sequence of snapshots
  - Snapshot = sentence, sketch, slide, picture, etc.
  - Possibly structured with annotations:
    - WHO are the players, WHAT happens to them, WHY this happens, WHAT IF this does / does not happen, etc
  - **Passive** mode (for validation): stakeholders are told the story
  - **Active** mode (for joint exploration): stakeholders contribute

Scenarios

- Illustrate typical sequences of interaction among system components to meet an implicit objective

- Widely used for...
  - **explanation** of system-as-is
  - **exploration** of system-to-be + elicitation of further info ...
    - e.g. WHY this interaction sequence ?
    - WHY among these components ?
  - specification of acceptance test cases

- Represented by text or diagram (see Chap. 4)
Scenario example: meeting scheduling

1. The initiator asks the scheduler for planning a meeting within some date range. The request includes a list of desired participants.

2. The scheduler checks that the initiator is entitled to do so and that the request is valid. It confirms to the initiator that the requested meeting is initiated.

3. The scheduler asks all participants in the submitted list to send their date and location constraints back within the prescribed date range.

4. When a participant returns her constraints, the scheduler validates them (e.g., with respect to the prescribed date range). It confirms to the participant that the constraints have been safely received.

5. Once all valid constraints are received, the scheduler determines a meeting date and location that fit them.

6. The scheduler notifies the scheduled meeting date and location to the initiator and to all invited participants.

Types of scenario

- **Positive** scenario = one behavior the system should cover (example)
- **Negative** scenario = one behavior the system should exclude (counter-example), e.g.
  1. A participant returns a list of constraints covering all dates within the given date range
  2. The scheduler forwards this message to all participants asking them for alternative constraints within extended date range
- **Normal** scenario: everything proceeds as expected
- **Abnormal** scenario = a desired interaction sequence in exception situation (still positive)
  e.g. meeting initiator not authorized
  participant constraints not valid
**Scenarios: pros & cons**

- **😊 Concrete examples/counter-examples**
- **😊 Narrative style (appealing to stakeholders)**
- **😊 Yield animation sequences, acceptance test cases**
- **😊 Inherently partial (cf. test coverage problem)**
- **😊 Combinatorial explosion (cf. program traces)**
- **😊 Potential overspecification: unnecessary sequencing, premature software-environment boundary**
- **😊 May contain irrelevant details, incompatible granularities from different stakeholders**
- **😊 Keep requirements implicit**
  - cf. confidentiality req in negative scenario example

*Concrete scenarios naturally jump in anyway... invaluable as initial elicitation vehicles*

---

**Prototypes & mock-ups**

- **Goal**: check req adequacy from direct user feedback, by showing reduced sketch of software-to-be in action
  - focus on unclear, hard-to-formulate reqs to elicit further
- **Prototype** = quick implementation of some aspects...
  - **Functional proto**: focus on specific functional reqs
    - e.g. initiating meeting, gathering participant constraints
  - **User interface proto**: focus on usability by showing input-output forms, dialog patterns
    - e.g. static/dynamic interaction to get participant constraints
- **Quick implementation**: by use of very high-level programming language, executable spec language, generic services, ...
Mock-up: proto is thrown away (product = adequate reqs)

Evolutionary proto: transformed towards efficient code

Prototypes & mock-ups: pros & cons

Concrete flavor of what the software will look like
=> clarify reqs, elicit hidden ones, improve adequacy, understand implications, ...

Other uses: user training, stubb for integration testing, ...

Does not cover all aspects
- missing functionalities
- ignores important non-functional reqs (performance, cost, ...)

Can be misleading, set expectations too high

'Quick-and-dirty' code, hard to reuse for sw development

Potential inconsistencies between modified code and documented reqs
Knowledge reuse

- **Goal**: speed up elicitation by reuse of knowledge from experience with related systems
  - knowledge about similar organization, domain, problem world: requirements, assumptions, dom props, ...

- General reuse process:
  1. **RETRIEVE** relevant knowledge from other systems
  2. **TRANSPOSE** it to the target system
  3. **VALIDATE** the result, **ADAPT** it if necessary & **INTEGRATE** it with the system knowledge already acquired

- Transposition mechanisms:
  - instantiation (memberOf)
  - specialization (subClassOf) + feature inheritance
  - reformulation in vocabulary of target system

Reuse of domain-independent knowledge: requirements taxonomies

- For each leaf node in available req taxonomies:
  "Is there any system-specific req instance from this class?"

- More specific taxonomy => more focussed search

Performance Requirement

Space
- Main Storage
- Secondary Storage

Time
- ResponseTime
- Throughput
- OffPeakThroughput
- PeakThroughput
- PeakMeanThroughput
- PeakUniformThroughput

- mean number of meetings to be scheduled at peak times?

response time for ...
- participant constraints?
- meeting scheduling?
- meeting notification?
Reuse of domain-independent knowledge: RD meta-model

- **RD meta-model** = concepts & relationships in terms of which RD items are captured
- Elicitation by meta-model traversal
- RD items are acquired as **instantiations** of meta-model items

Reuse of domain-specific knowledge

- **Abstract domain** = concepts, tasks, actors, objectives, reqs, dom props abstracting from a class of domains
- RD items acquired as **specializations** of abstract items to target system (feature inheritance + system-specific renaming)
Reuse of domain-specific knowledge (2)

- Same abstract domain may have multiple specializations
  e.g. resource management ← library loan management, video store management, flight or concert seat allocation, ...

- Same concrete domain may specialize multiple abstract domains
  e.g. library management:
  
  - loan management → resource management
  - book acquisition → e-shopping
  - patron registration → group membership management

- More adequate RD items elicited by reuse of more structured, more accurate abstract domains
  e.g. resource management: returnable vs. consumable resource
  sharable vs. non-sharable resource
  ⇒ "A book copy can be borrowed by one patron at a time" (dom prop for non-sharable, returnable resource)

Knowledge reuse: pros & cons

- Expert analysts naturally reuse from past experience
- Significant guidance and reduction of elicitation efforts
- Inheritance of structure & quality of abstract domain spec
- Effective for completing RD with overlooked aspects
- Effective only if abstract domain sufficiently "close", accurate
- Defining abstract domains for significant reusability is hard
- Validation & integration efforts
- Near-matches may require tricky adaptations
Domain analysis & requirements elicitation: outline

- Identifying stakeholders & interacting with them
- Artifact-driven elicitation techniques
  - Background study
  - Data collection, questionnaires
  - Repertory grids, card sorts for concept acquisition
  - Scenarios, storyboards for problem world exploration
  - Prototypes, mock-ups for early feedback
  - Knowledge reuse: domain-independent, domain-specific
- Stakeholder-driven elicitation techniques
  - Interviews
  - Observation and ethnographic studies
  - Group sessions

(3) Interviews

- Primary technique for knowledge elicitation
  1. Select stakeholder specifically for info to be acquired
     (domain expert, manager, salesperson, end-user, consultant, ...)
  2. Organize meeting with interviewee, ask questions, record answers
  3. Write report from interview transcripts
  4. Submit report to interviewee for validation & refinement
- Single interview may involve multiple stakeholders
  - saves times
  - weaker contact; individuals less involved, speak less freely
- Interview effectiveness:
  \[
  \text{utility} \times \text{coverage of acquired info} / \text{acquisition time}
  \]
Types of interview

- **Structured interview**: predetermined set of questions
  - specific to purpose of interview
  - some open-ended, others with pre-determined answer set
  => more focused discussion, no rambling among topics

- **Unstructured interview**: no predetermined set of questions
  - free discussion about system-as-is, perceived problems, proposed solutions
  => exploration of possibly overlooked issues

=> Effective interviews should mix both modes ...
  - start with structured parts
  - shift to unstructured parts as felt necessary

Interviews: strengths & difficulties

😊 May reveal info not acquired through other techniques
  - how things are running *really*, personal complaints, suggestions for improvement, ...

😊 On-the-fly acquisition of info appearing relevant
  - new questions triggered from previous answers

😊 Acquired info might be subjective (hard to assess)

😊 Potential inconsistencies between different interviewees

😊 Effectiveness critically relies on interviewer's attitude, appropriateness of questions

=> Interviewing guidelines
Guidelines for effective interviews

- Identify the right interviewee sample for full coverage of issues
  - different responsibilities, expertise, tasks, exposure to problems
- Come prepared, to focus on right issue at right time
  - background study first
  - predesign a sequence of questions for this interviewee
- Centre the interview on the interviewee’s work & concerns
- Keep control over the interview
- Make the interviewee feel comfortable
  - Start: break ice, provide motivation, ask easy questions
  - Consider the person too, not only the role
  - Do always appear as a trustworthy partner

Guidelines for effective interviews (2)

- Be focused, keep open-ended questions for the end
- Be open-minded, flexible in case of unexpected answers
- Ask why-questions without being offending
- Avoid certain types of questions ...
  - opinionated or biased
  - affirmative
  - obvious or impossible answer for this interviewee
- Edit & structure interview transcripts while still fresh in mind
  - including personal reactions, attitudes, etc
- Keep interviewee in the loop
  - co-review interview transcript for validation & refinement

Model-driven interviews may help structure them
(see Part 2 of the book)
Observation & ethnographic studies

- Focus on task elicitation in the system-as-is
- Understanding a task is often easier by observing people performing it (rather than verbal or textual explanation)
  - cf. tying shoelaces
- Passive observation: no interference with task performers
  - Watch from outside, record (notes, video), edit transcripts, interpret
  - Protocol analysis: task performers concurrently explain it
- Ethnographic studies: over long periods of time, try to discover emergent properties of social group involved about task performance + attitudes, reactions, gestures, ...
- Active observation: you get involved in the task, even become a team member

Observation & ethnographic studies: pros & cons

😊 May reveal ...
- tacit knowledge that would not emerge otherwise
  - e.g. ethnographic study of air traffic control => implicit mental model of air traffic to be preserved in system-to-be
  - hidden problems through tricky ways of doing things
  - culture-specific aspects to be taken into account

😊 Contextualization of acquired info

😢 Slow & expensive: to be done over long periods of time, at different times, under different workload conditions

😢 Potentially inaccurate (people behave differently when observed)

😢 Data mining problem, interpretation problem

😢 Focus on system-as-is

Some of the interviewing guidelines are relevant
Group sessions

- More perception, judgement, invention from interactions within group of diverse people
- Elicitation takes place in series of group workshops (a few days each) + follow-up actions audiovisuals, wall charts to foster discussion, record outcome
- **Structured** group sessions:
  - Each participant has a clearly defined role (leader, moderator, manager, user, developer, ...)
  - Contributes to req elaboration according to his/her role, towards reaching synergies
  - Generally focused on high-level reqs
  - Variants: focus groups, JAD, QFD, ...

Group sessions (2)

- **Unstructured** group sessions (brainstorming):
  - Participants have a less clearly defined role
  - Two separate stages ...
    1. **Idea generation** to address a problem:
       as many ideas as possible
       from each participant without censorship/criticism
    2. **Idea evaluation**:
       by all participants together according to agreed criteria (e.g. value, cost, feasibility) to prioritize ideas
Group sessions: pros & cons

 Less formal interactions than interviews
  => may reveal hidden aspects of the system (as-is or to-be)

 Potentially ...
  - wider exploration of issues & ideas
  - more inventive ways of addressing problems

 Synergies => agreed conflict resolutions

 Group composition is critical ...
  - time consuming for key, busy people
  - heavily relying on leader expertise & skills
  - group dynamics, dominant persons => biases, inadequacies

 Risk of ...
  - missing focus & structure => rambling discussions, little
    concrete outcome, waste of time
  - superficial coverage of more technical issues

Combining techniques

◆ Elicitation techniques have complementary strengths & limitations

◆ Strength-based combinations are more effective for full, adequate coverage
  - artifact-driven + stakeholder-driven

◆ Examples
  - Contextual Inquiry: workplace observation + open-ended interviews + prototyping
  - RAD: JAD group sessions + evolutionary prototyping (with code generation tools)

◆ Techniques from other RE phases support elicitation too
  - Resolution of conflicts, risks, omissions, etc.
Domain analysis & requirements elicitation: summary

- Identifying the right stakeholders, interacting the right way
- Artifact-driven elicitation techniques
  - Background study as a prerequisite
  - Data collection, questionnaires for preparing interviews
  - Repertory grids, card sorts for concept characterization
  - Scenarios, storyboards for concrete exploration
  - Prototypes, mock-ups for early feedback & adequacy check
  - Knowledge reuse brings a lot: domain-independent, domain-specific
- Stakeholder-driven elicitation techniques
  - Interviews are essential - structured, unstructured, cf. guidelines
  - Observation, ethnographic studies for hidden knowledge
  - Group sessions for broader, more inventive acquisition & agreement

Model-driven elicitation provides focus & structure for what needs to be elicited (see Part 2 of the book)