Requirements Elicitation

- “The process of identifying needs and bridging the disparities among the involved communities for the purpose of defining and distilling requirements to meet the constraints of these communities.”

- “…requirements elicitation involves social, communicative issues as well as technical issues.”

- “…research efforts should be directed towards methods…providing more support to the elicitation of requirements.”
Elicitation process

- Identify relevant sources of requirements
- Ask appropriate questions to gain an understanding of the needs
- Analyze the gathered information, looking for implications, inconsistencies, or unresolved issues
- Synthesize appropriate statements of the requirements
- Confirm your understanding of the requirements with the users
- Loop back if conflicts occur

Actors in requirements elicitation

- Software requirements engineer & his staff
- Designers of the target system
- Potential users of the target system
- Application domain experts
- The users’ managers

- No ONE person knows everything about what a software system should do
- We have to find the right persons, then meet with them effectively by asking the right questions and use various techniques to gather as much information as early as possible
Outcomes of good elicitation

- Helps the users in the separation of what they want and what they need
- Enables all actors to share the same vision of the problem and the solutions that are feasible
- Enables trust between software developers and customers
- Minimizes the interactions between developers and customers

Outcomes poor elicitation

- The developers might solve the wrong problem
- Customers might be dissatisfied
  - Developers are enforcing their own view on the system
  - Customers are not participating enough
- Chaotic development process
  - Additional meetings are required with the customers
  - Developers make wrong decisions
  - Change in requirements can have unexpected wide-range impacts
- Loss of reputation, credibility and morale for the developers
- Loss of money, time and confidence for the customers
Difficulties of elicitation

- Requirements elicitation is a complex and imprecise process that varies greatly for different projects.
- Many technical problems are related to this process:
  - Scope problems
  - Problems related to the nature of computer science
  - Problems related to the process itself
- Many problems pertaining to human nature are related to this process:
  - Articulation problems
  - Communication barriers
  - Knowledge and cognitive limitations
  - Human behavior issues

Scope problems

- The boundaries of the system are ill-defined
- The context of the system is ill-defined
  - Hardware and software constraints
  - System’s role in a larger system
  - Maturity of the system’s domain
  - Attributes of the actors: management style, hierarchy, domain experience, computer experience, etc.
- Unnecessary or restrictive design information is given
The nature of computer science

- Problems are becoming increasingly complex
- Processes are evolving
- Software and hardware technologies are changing rapidly

Problems with the process itself

- Requirements change over time
- There are many sources of requirements
- The nature of the system imposes constraints on the elicitation process
- The cross-referencing and heterogeneity of information gathered complexifies its management
Articulation problems

- Users are aware of their needs but unable to express them
- Users are not aware of possible solutions available to them
- Users may be aware of some needs but afraid to articulate it
- Actors have different meanings for common terms
- Users are not aware of the consequences of their needs
- No actor has the complete picture of the system
- Developers might not be really listening to the user’s needs
- Developers fail to understand, appreciate or relate to users
- Developers overrule or dominate users

Communication barriers

- Actors have different vocabularies
- Users often have very grounded concerns
- All media of communication have their problems
- Actors have incompatible styles of interaction
- Actors have different personalities
Knowledge and cognitive limitations

- Developer must have adequate domain knowledge
- Actors have selective memories
- Actors use intuitive statistics to express ideas
- Actors have difficulties with scale and complexity
- Actors have preconceived ideas to the solution
- Actors focus on some narrow aspects of the problem
- Actors are not willing to undertake extensive exploration

Human behavior issues

- There are ambiguities in the roles that each actor has to play in the elicitation process
- Users fear that the introduction of the system will necessitate changes in their behavior
- Users fear the system will make them lose their jobs
Barriers to elicitation

- The “Yes, But” Syndrome
  - Two reactions when the users see the system implementation for the first time:
    - “Yes, this is so cool.”
    - “Yes, but what about this...?”....
  - It is human nature and is an integral and important part of application development
  - To address this problem, apply techniques that can get the “buts” out early
  - The later problems are found, the more costly they are to fix

- The “Undiscovered Ruins” Syndrome
  - “The more that are found, the more you know remain.”
  - It is impossible to know when requirements are complete
  - The validation phase will assess the completeness
  - To address this problem
    - Identify all of the stakeholders
    - Use the techniques we will discuss later
Barriers to elicitation

- The “User and the Developer” Syndrome
  - There is a communication gap between the user and the developer
    - They don’t use the same language
    - They don’t have the same background knowledge
    - They don’t have the same motivations and objectives
    - The gap has to be minimized
  - Most developers are not trained in elicitation techniques
  - To address this problem
    - Learn to communicate effectively
    - Table 7-1. P.84

Global problems

- 13% Lack of user input
- 12% Incomplete requirements and specifications
- Lack of understanding and communication (25%)

- Stakeholders never take the initiative of communicating their needs
- Initiative has to come from the developers
- Developers are responsible for choosing the techniques for requirements elicitation
Understanding user needs

- “Problem analysis” describes the process of analyzing the problem to be solved
- This process is based on input data gathered from the stakeholders
- How do we gather this data?

Requirement elicitation techniques

- To address the three problems, we will discuss the following techniques:
  - Interviewing and questionnaires
  - Requirements workshop
  - Brainstorming and idea reduction
  - Storyboards
  - Use cases
  - Role playing
  - Prototyping
- How do we choose the right technique for elicitation?
Managing the complexity of acquired data

Stakeholder needs

- A stakeholder need is:
  - a reflection of the business, personal, or operational problem.
  - must be addressed in order to justify consideration, purchase, or use of a new system.

- Most needs are very abstract and require analysis:
  - “I need easier ways to understand the status of my inventory”
  - “I’d like to see a big increase in the productivity of sales order entry”
  - “The weapon system must score a hit in 95% of missile firing”
  - “The vehicle must be able to slow down as quickly as possible”

- Others are very specific:
  - “The ATM machine must provide features such as withdraw, deposit, and transfer money”

- We have to understand the true stakeholders needs
System features

- The features of a product or system are high-level expressions of desired system behavior
- These features are often not well defined and may even be in conflict
- Features are a convenient way to describe functionality without getting down in detail
- The features are easily expressed in natural language and consist of a short phrase
- Features and needs are sometimes hard to categorize

Needs vs. features

- If needs are not met, project is a failure
- Features should satisfy needs
- Stakeholders will express either needs or features
- Needs generate features
- Needs and features will be analyzed to form software requirements
- Requirements can be modeled into software specifications
- Each level is adding details and coming closer to implementation considerations
Managing complexity

- Complexity of system analysis comes mainly from the amount of data acquired during elicitation
  - Introduce abstraction levels to reduce the local complexity
  - Any system can be defined in a list of 25-99 features
  - The small amount of information provides a comprehensive and complete basis for
    - product definition
    - communication with the stakeholders
    - scope management
    - project management

Managing complexity

- Features can be described and categorized using attributes
- They are used to provide additional information about the features
- They help us to manage the complexity of the information
- Helps in project scoping and planning
- Table 8-2, p92
Interviewing

- Interviewing is a simple and direct technique that can be used in every situation
- Make sure that any preconceived ideas do not interfere
  - Normalize first impression effects by making the context clear
  - Be open to all input and expect conflicts of all sorts
  - Do not impose your views
- Nothing substitutes for an interview
- Good technique in early stages
- Can be structured or unstructured
Interviewing

- Context-free questions
  - Question about the nature of the user’s problem without any context for a potential solution
  - Aims at coming to an understanding of the problem
  - Force you to listen before attempting to invent or to describe a potential solution
  - Listening gives us a better understanding of the user's problem and any problems behind the problem
- Context, such as solution hints, will later give more information to stimulate the user’s imagination

Interviewing

- Questionnaires
  - Do the interview first
  - The questionnaire is not a substitute for an interview
  - Can be applied with good effect as a corroborating technique after the initial interviewing and analysis activity
  - Be careful when designing questions
  - Answers are often unclear
Requirements workshop

- It gathers all key stakeholders together for a short but intensely focused period (1-2 days)
- One of the most powerful technique for eliciting requirements
- Brainstorming is the most important part of the workshop
Requirements workshop

- Preparing for the workshop
  - Proper preparation for the workshop is critical to success.
  - The steps of preparing
    - Selling the concept
    - Identifying stakeholders
    - Logistics
    - Sending materials out in advance
  - Ask a facilitator to run the workshop

Requirements workshop

- Running the workshop
  - Facilitator:
    - Outsider, if possible
    - Trained and/or experienced in workshops
    - Great team worker and mediator
    - Respected person
  - Brainstorming and idea reduction
  - Workshop tickets (figure 10-2, p110)
  - Output of the workshop
Requirements workshop

- A properly run requirements workshop has many benefits.
  - It assists in building an effective team.
  - All stakeholders get their say.
  - It forges an agreement between the stakeholders and the development team.
  - It can expose and resolve political issues.
  - The output, a preliminary system definition at the features level, is available immediately.

Brainstorming and idea reduction
Brainstorming and idea reduction

1- idea generation
   - Generate as many abstract ideas as possible

2- idea reduction
   - Sort out ideas by ranking, expanding, grouping, eliminating

Rules:
   - Do not allow criticism or debate
   - Let your imagination soar
   - Generate as many ideas as possible
   - Mutate and merge ideas

Set up a goal (e.g. What features are needed?)

Facilitator writes down ideas clearly, keeps the participants on focus, and decides when to end the meeting

Brainstorming and Idea Reduction

The benefits of this elicitation technique:
   - It encourages participation by all parties present.
   - It allows participants to “piggyback” on one another’s ideas.
   - The facilitator maintains a written trail of everything discussed.
   - Large scope is discussed.
   - Typically, it results in a broad set of possible solutions.
   - It encourages free thinking without being limited by normal constraints.
Storyboarding

- Goal: elicit early “yes, but” reactions
- Put the stakeholders in a situation and ask them to comment
- Passive: tell a story using sketches, pictures, screenshots, sample output
- Active: Show an animated view of system behavior
- Interactive: have the users interact with a throw-away prototype
Storyboarding

- Tips
  - Don’t invest too much
  - Make the story board easy to change
  - Don’t make the storyboard too good
  - The more interaction, the better
- When?
  - Early
  - Often
  - On any project with innovative content
Use cases

- As in the use case driven approach
- Defines the who, what and how of the system
- Defines only the functional behavior
  - Input only from users
  - Does not address non-functional issues
- Meshes perfectly with OO design
- Very powerful to elicitate GUI-intensive applications

Define Use Cases

- Use case: A coherent unit of externally visible functionality provided by a system unit.
- Used to define a behavior without revealing the internal details.
- A use case describes what the system does, not how it does it.
- Scenario: flow of events describing how a use case is realized.
- Each use case has a primary scenario.
- Eventually also has a set of alternate scenarios.
- Pre-conditions and post-conditions are stated.
Define Use Cases

Place Order
Pre-conditions:
A valid user has logged into the system
Primary Flow of Events:
1. (start) The customer selects Place Order
2. The customer enters its address
3. The customer enters the product codes it wants to order
4. The system provides the items description and prices, and a running total
5. The customer enters its credit card number
6. The customer clicks on submit
7. The system validates the information, saves the order and forwards the transaction request to the accounting system
8. (end) When the payment is confirmed, the order is marked as paid
Alternate Flow of Events 1:
In step 7, the system prompts the user to correct any incorrect information
Alternate Flow of Events 2:
In step 8, if the transaction is refused by the bank, the order is marked as pending
Post-conditions:
The order has been saved in the database

Scenarios: Diagrams

- Complex scenarios are better expressed using diagrams.
- The UML provides two kinds of diagrams:
  - Activity diagrams for a high-level description.
  - Sequence diagrams for more in-depth analysis.
Use Case Diagrams

- Roles
  - Model the context of the system. Define what are the actors that are external to the system
  - Model the requirements of the system. Define what the system should do from an external point of view

Order-Processing Use Case Diagram
Role playing

- The problem:
  - We have talked about it (interview)
  - We have discussed it (workshops)
  - We have presented our view of it (storyboarding)
  - We haven't experienced it

- Sometimes, experience is needed:
  - Some situations are hard to articulate (e.g. tie your shoes)
  - Some situations are hard to admit (e.g. workarounds)
  - The context is sometimes important (e.g. working environment)
Goal: validate our understanding of the problem by implementing a partial solution that the user can use.

Addresses two syndromes:
- Yes but: “That is not exactly what I meant”
- Undiscovered ruins: “Now that I see it, I have another requirement to add”

Two types of prototypes:
- Throwaway: tool for requirements validation
- Evolutionary: Early version of the system to be delivered
Prototyping

- Emphasis on the interface, not on internal mechanics of the system
- Cannot validate non-functional requirements effectively
- Must be cost-effective
- Must be exercised in the deployment environment

Prototyping

- What to prototype?
  - Well-understood requirements are not needed, unless they are necessary for the prototype to function
  - Unknown or undefined needs cannot be prototyped
  - Only the “fuzzy” part in between is prototyped
- Result
  - Validation of understanding of the problem
  - Refinement of understanding
  - Reduces risks in the project
- Must look as closely as possible as the final product
Summary

- Elicitation is the input to all system development
- Elicitation technique varies greatly across projects
- No one technique is universal
- Missing requirements is extremely costly to fix
- Completeness can never be assured
- Lots of human nature and communication problems
- Elicitation skills, techniques and processes are minimal in too many SE companies