Integrative Systems Engineering

Requirements Engineering in the Problem Domain

Prof. Michael Zink
Overview

- What is the problem domain?
- Instantiating the generic process
- Agree requirements with customers
- Analyze and model
- Identify stakeholders
- Create use scenarios
- Scoping the system
- Define structure
- Capture requirements
- Define acceptance criteria
What is the Problem Domain?

- Domain in which system is going to be used
- Look at requirements from an operational point of view
  - *Enabling aspect* is at the heart of requirements engineering in the problem domain
- Tempted to ask the question: *What do you want the system to do?*
- Hard to answer if no existing system is available
- To avoid premature jump into solutions domain ask: *What is the purpose of the system you want?*
What is the Problem Domain?

- This way customers will think about what they want to be able to do with the system, rather than how they will do it
- Leaves solution space open to Systems Engineers and Architects
- Even mentioning “the system” could be misleading: 
  *What do you want to be able to do?*
- Answer
  *I want to be able to ...*
  is known as a capability requirement
What is the Problem Domain?

- Engineering in the problem domain is primarily about eliciting capabilities.

- This leads to the identification of stakeholders: *People or organizations that have some direct or indirect interest in the intended system*.

- What sort of models are relevant to the problem domain?
  - Must be understandable to stakeholders, because they will validate them.
  - Use scenarios are good models.

- Overriding constraints:
  - Budget
  - Time
Instantiating the Generic Process

- Starting point is statement of needs
  - Could be as simple as an email from CEO or CTO stating that a new product is required to get one step ahead of the competition
  - Alternatively, there may already have been a study performed to look at possible options
  - Concept of operations document produced that identifies some use scenarios

- Following figure identifies that analyze and model process creates a set of use scenarios plus a list of stakeholders

- Derived requirements will be stakeholder requirements
Instantiating the Generic Process

1. Statement of need
2. Agree Requirements
3. Qualification Strategy
4. Derive Stakeholder Requirement & Qualification Strategy
5. Analyze & Model
6. Use Scenarios
7. Stakeholders
8. Change Request
9. Agree Requirements
10. Qualification Strategy
11. Change Request
12. Stakeholder Requirements
13. Change Request
14. Qualification Strategy
Agree Requirements with Customer

- Agreement process is usually very informal
- Likely the statement of needs document is a simple document
  - not been engineered from requirements point of view
  - Not containing atomic requirements
- In this respect stakeholder requirements process is different to other requirements processes – because it starts from rather vague position
- One of key elements in eliciting stakeholder requirements is to establish scope of intended system
  - Usually done after set of use scenarios has been established
Analyze and Model

Statement of need

Identify Stakeholders

Create Models

Change Request

Stakeholders

Use Scenarios
Identify Stakeholders

- Can be any person or organization that has an opinion, a responsibility for, or who may be influenced or effected by the proposed system
- Types of stakeholders vary according to the nature of the system
- People who have an opinion about proposed system include people who will use system directly
- People with responsibility for system will be managers in charge of operating or safety authorities
- Following list contains potential stakeholders (no claim to be complete)
Identify Stakeholders

- Managers
- Investors
- System users
- Maintenance and service staff
- Product disposer
- Training personnel
- System buyer
- Sales and marketing
- Usability and efficiency experts
- Operational environment experts
- Government
- Standard bodies
- Public opinion and opinion leaders
- Regulatory authorities
Create Use Scenarios

- Basic structuring mechanism for discussing capability requirements
- Produces hierarchically (by time) organized structure
- Scenarios are used as a means of establishing framework for meaningful dialogue
- Encourages stakeholders to think about job they are doing and how they would like to do it
- Once scenario has been agreed individual requirements can be generated
- Use scenarios may be represented as a hierarchy of goals - capability hierarchy
Create Use Scenarios

Final Goal

Sub-goal

Sub-goal

Sub-goal

Sub-goal

Sub-goal

Sub-goal

Sub-goal

Sub-goal

Operational sequence
Create Use Scenarios

- Clearly defined approach to follow when creating use case scenarios:
  *What do you want to achieve?*

- Process:
  - Start with the end goal
  - Derive the necessary capabilities to get to that point
  - Break large steps into smaller steps
  - Keep the set hierarchical
  - Review informally at each stage
  - Be wary of defining solutions
Create Use Scenarios

- Scenarios represent the capabilities to be provided by the system organized into a hierarchy – without saying how to provide them.

- They are seen to be beneficial for the following reasons:
  - Enables stakeholders to step through operational use
  - Missing steps can be found
  - Different stakeholders can have different scenarios
  - Time construct can be identified
Characteristics of Use Scenarios

- Following scenario looks at a simple example on a day out on a sailing boat that can be transported on a car
- Covers all aspects of trip:
  - Loading boat on car
  - Getting ready to sail
  - Sailing and returning home
- Illustrates also other points:
  - Generally, it follows time sequence
  - Its nodes are high-level capabilities
  - It shows alternatives
  - It shows periodic repeated behavior
  - It shows where sequence is not important
  - It shows exceptions
Characteristics of Use Scenarios

**Sequential:**
- Able to go sailing
- Able to get ready to sail
- Able to load the boat
- Boat lifted
- Boat on car
- Able to unload the boat
- Able to rig mast
- Able to rig rudder
- Able to rig center-plate

**Periodic:**
- Able to sail
- Able to maneuver
- Able to sail normally
- Able to go ashore

**Alternate:**
- Able to return home
- Able to rig boat
- Able to rig rudder
- Able to survive capsize
Scoping the System

- Initially, set boundary slightly wider than anticipated system boundary
- At some point it is important to determine where boundary of system is to be placed
- Once complete set of scenarios has been assembled, scope of system can be finalized
- May change once after cost of system has been estimated
- Estimates based on scenarios are very coarse and have a high degree of uncertainty
Derive Requirements

1. Stakeholders
2. Statement of Need
3. Use Scenarios

- Define Structure
- Capture Requirements

- Structured Requirements
- Refine Requirements

- Generate Stakeholder Requirements
- Structure
- Candidate Requirements

Stakeholder Requirement
Define Structure

- Structure is critical for handling all elements in life cycle
- Stakeholder requirements:
  - Captured one by one
  - Cleaned up
  - Attached into structure
- Some approached assume that:
  - Stakeholder requirements are inherently unstructured
  - Traceability to design is enough
  - Never see complete requirements model – requirements need be viewed only one at a time
Define Structure

- Main structuring concept for stakeholder is use scenario
- However, there can be many
- Try to merge scenarios – but will not always work
- Example:
  - Overall life of restaurant – owner’s scenario
  - A day in the life of restaurant – manager’s scenario
  - A meet at the restaurant – customer’s scenario

```
Restaurant Life
  1 Acquired
  2 Operating
  3 Sold
```
Define Structure

Restaurant Day

1 Replenished

2 Open

3 Closed

1.1 Food delivered

1.1.1 Meat delivered
1.1.2 Fish delivered
1.1.3 Veggies delivered
1.1.4 Bread delivered

1.1 Drinks delivered

3.1 Tables cleared
3.2 Washing-up complete
3.3 Waste bins ready
3.4 Replenishments listed
Define Structure

Customer Meal

1. Table booked
2. Arrived and seated
3. Food served
4. Food eaten
5. Bill received
6. Bill paid
Define Structure

Restaurant Life

1 Acquired

2 Operating

2.1 Restaurant Day

2.1.1 Replenished

1.1 Meat delivered

1.1.1 Meat delivered

1.1.2 Fish delivered

1.1.3 Veggies delivered

1.1.4 Bread delivered

1.1.5 Replenishments listed

2.1.2.1 Table booked

2.1.2.1.1 Table booked

2.1.2.1.2 Arrived and seated

2.1.2.1.3 Food served

2.1.2.1.4 Food eaten

2.1.2.1.5 Bill received

2.1.2.1.6 Bill paid

2.1.2.1 Customer Meal

2.1.3 Closed

2.1.3.1 Tables cleared

2.1.3.2 Washing-up complete

2.1.3.3 Waste bins ready

2.1.3.4 Replenishments listed

1.1.6 Drinks delivered

2 ECE Department

24
Capture Requirements

- Sources of stakeholder requirements
- Stakeholder interviews
- Extracting requirements from informal documents
- Identifying capability requirements from scenarios
- Requirements workshops
- Requirements learnt from experience
- Requirements from prototypes
- Constraints in stakeholder requirements
- Refine requirements
- Derive qualification strategy
Sources of Stakeholder Requirements

- Interviews with stakeholders
- Scenario exploration (generally through interviews)
- Descriptive documentation (studies/market research)
- Existing systems which are being upgraded
- Problems and change suggestions from existing systems
- Analogous systems
- Prototyping (partial systems, mock-ups or even simple sketches)
- Opportunities from new technology
- Studies
- Questionnaires
- Anthropomorphic studies or analysis of videos
Stakeholder Interviews

- Interview every type of stakeholder
- Take them seriously
- Document the interview and have stakeholders sign document
- Identify relevant scenarios
- If necessary create new scenarios and develop requirements from them
- Discover importance of each requirement to stakeholder
- If stakeholder is vague about requirement ask how it could be demonstrated
- Enquire about known constraints
- Make stakeholders aware that their requirements will shape system
- Do not be judgemental
Extracting Requirements from Informal Docs

- Informal documents:
  - Letter
  - Studies
  - Action lists
- Should be brought into the open
- Record where stakeholder requirements have come from
- Requirements extracted this way must be substantiated by stakeholders
Identifying Capability Requirements from Scenarios

- Possible to derive requirements capability directly from outline scenario
  - E.g., ready to sail
    the user shall be able to lift the boat on
    the roof of an average sedan

- Raises questions:
  - How strong are the people?
  - What is an “average sedan”?

- Important to document requirements
- Critical issue is NOT to loose an idea
Requirements Workshops

- Gather stakeholders in conducive environment
- Structure meeting teach subject area
- Present stakeholders with a requirements document or a set of use scenarios
- Encourage criticism and interaction amongst stakeholders
- Rapidly process amendments
- Produce a new version
Requirements Learnt from Experience

- Problems reported by real user of systems – priceless
- Negative attitude because it is associated with a problem
- The earlier problem is detected the less will be the cost of change
- In iterative development these changes can be taken into account
Requirements from Prototypes

- Prototypes can be invaluable when creating unprecedented systems
- Important in software systems where user interface is difficult to imagine
- Problem: Developers get carried away and spend too much time and effort
- Should be treated as a small sub-project
- Problems with prototyping:
  - Developers get carried away
  - Cause stakeholders to stray into implementation
  - Stakeholders may be impressed with prototype and want to use it operationally
Constraint in Stakeholder Requirements

- Requirement that does not add any capability to system
- Constraint controls way in which capabilities are to be delivered
- Quantifies service to be provided
- Caution: Mass of constraints can make development impossible

When design is known each constraint should be analyzed for cost-benefit value or impact:
  - It may bring function into existence (e.g., caution and warning system or backup)
  - Cost of a constraint can only be guessed before design is known
Refine Requirements

- Refine each requirement in its context and ensure that:
  - It belongs in the place it is in
  - It conforms to the criteria for well-written requirements as explained in last lecture
Derive Qualification Strategy

- Stakeholders
- Use Scenarios

Derive qualification strategy

Define acceptance criteria

Define qualification strategy

Acceptance criteria

Stakeholder requirement

Qualification strategy
Define Acceptance Criteria

- *What will convince you that this requirement has been satisfied?*

- Question can be answered in two ways:
  - Stakeholders may define operational situation in which the requirement can be demonstrated
  - Stakeholders may define a numerical value for a level of achievement that must be demonstrated

- Acceptance criteria for restaurant:
  - Profitability
  - Return on investment
  - Reputation
  - Forward load (how far ahead is restaurant booked?)
Define Qualification Strategy

- Air traffic controls system:
  - Necessary to show that all functionality has been properly provided
  - Controllers are satisfied
  - Requires mixture of tests:
    - System under light load must be demonstrated
    - If capability is not acceptable, then no point in progressing

- Cost of qualification strategy
  - Trade-off between cost and failing to detect a flaw