CS 409

Types of Requirement

capability functionlity behavior

1. A functional requirement is something the product must do (p.10)
   - Ex: "The product shall produce an updated schedule when a truck's status changes

2. A nonfunctional requirement is a quality (property) that the product must have (p.10)
   - Ex: "The product shall produce this amended schedule in less than 0.25 s

   quality attribute

3. A Constant is a restriction placed on the project or eventual design of the product (P.10)

   Constrains the design space.
   - Ex: "The product shall be available at the beginning of the new tax year."
     "The product shall operate on an iPad."

Adjacent system: are automated or human system that recieve data or services from your work acitivity &/or supply data or services to your work

Stakeholders

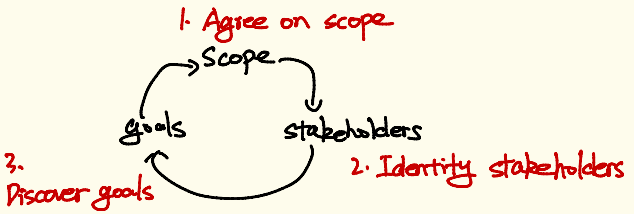
Anyone with interest in, or an effect on, the outcome of the product

IceBreaker stakeholders: road engineers, truck depot supervisors, road saftey experts, whether forecaster...

Business Goals of IceBreaker project:

more accurate prediction & de-icing schedule
safer roads
reduced waste of de-icing supplies

Software product is there to solve a business problem.



Risk: You must come to the correct understanding of the requirements and have your client agree with them, or your product will be deficient

Risk Reduction:

1. scope
2. Identify stakeholders with requirements/constrains for the product
3. understand the bussiness goals.

CHAPTER 4

Product Use Case Diagram

- Product use case
  the part of the business use case handled by the automated system (the software product)
- Product use case diagram shows
  - boundary between actors & software product to be built (rectangle)
  - Product use cases (ellipses)
  - usage (lines)
  - actors: people or adjacent system that interact with software product (P190-194)
- Advantages:
  - High level & informal model helps discovering & reaching agreement on clients' needs
  - Popular & easy to prompt discussion of requirements with stakeholds
  - Some regulators require them
  - Helps in planning: builds, tests, prototypes, changes
- Limitations:
  - Some functional requirement will be missing
  - From the users' perspective, so internal processing requirements not included
  - Most non-functional requirements & constraints will be missing.
- Solution: [Wiegers & Beatty]
  Use simple use case diagram together with Functional Requirements

Chapter 6: Product Use Case → Scenario

Fast way to find/record/revise functional requirements in meetings with stakeholders.

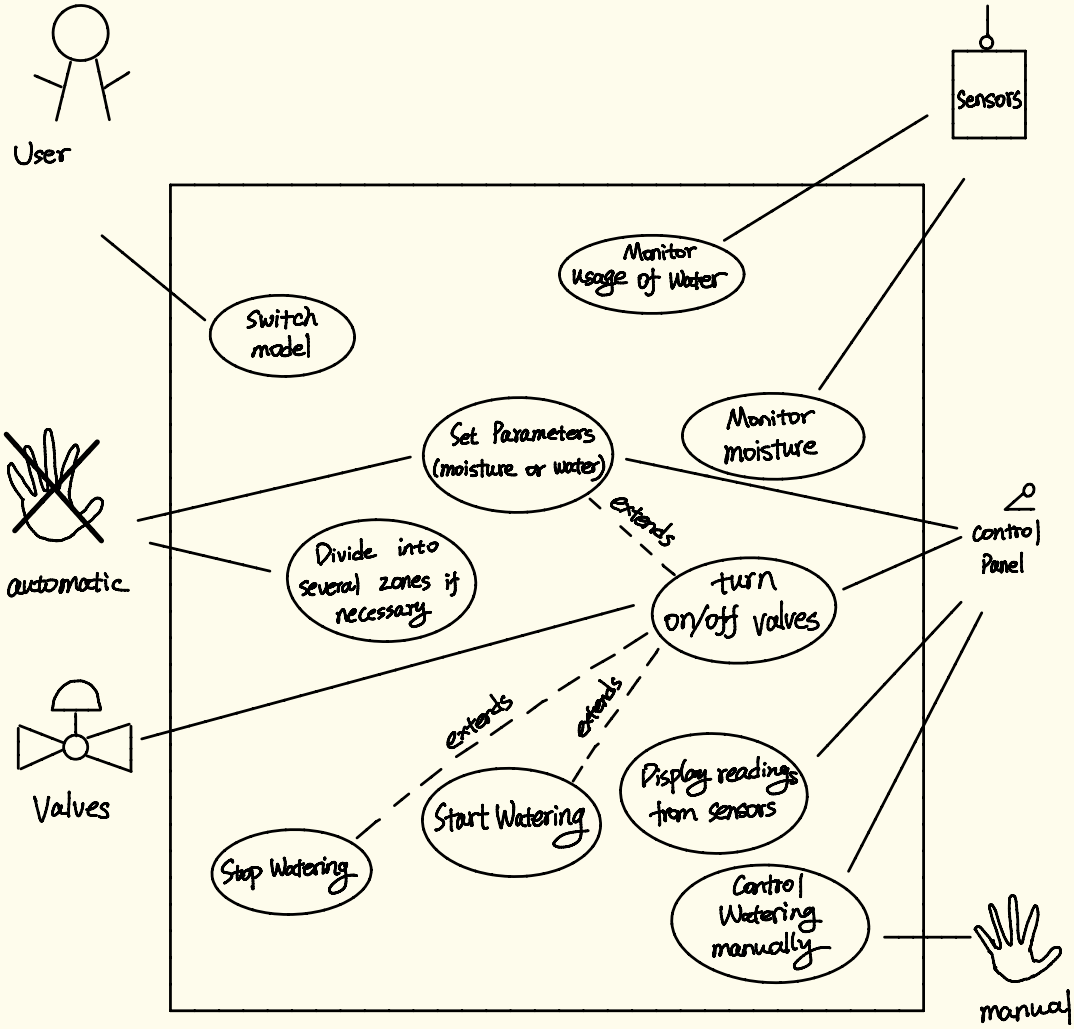
Scenario template

*1 Product use case name: passenger checks into airline flight.
*2 Trigger: passenger activates the machine
3 Preconditions:
4 Interested stakeholders:
*5 Actor: passenger
**6 Step: (1) Software asks for passenger's name...
⋮
(9) Software prints baggage tag
7. Outcome:

each product use case is an oval in PUCD
is expanded into a scenario

Beyond normal scenarios → off normals

- Alternative path
  actor has choices, so subsequent steps branch based on that choice
  Ex: buy now OR save cart for later?
- failure: "exception" al/misuse/negative/what-if scenarios
  unwanted deviations from normal
  Ex: buyer forgets password; malicious user enters wrong data; user is distracted
  at each step, ask "what can go wrong"?
  don't get carried away thinking of too many of these too early
  for critical systems use jackson's principle of commensurate care:
  Required care = probability of an exception × seriousness of dmg if it occurs
- storyboard: shows the steps of a scenario as a sequence of pictures
  Good for user interfaces, displays, dashboard


User
Sensors
Monitor usage of Water
Switch model
Set Parameters (moisture or water)
Monitor moisture
extends
automatic
Divide into several zones if necessary
turn on/off valves
Control Panel
Valves
extends
extends
Display readings from sensors
Start Watering
Stop Watering
Control Watering manually
manual

Scenarios → Functional Requirement (FRs)

· Functional Requirement : What the software product must do. ※

Need to contain enough detail for the developer to build right product.

Where to get them? (Robertson's rule of thumb: a step in a scenario maps to ~1-5 FRs)

From the steps of the product use case scenarios

EARS: Easy Approach to Requirements Syntax

5 templates:

1. Ubiquitous
   ~ requirement is something the system must always do (unconditional & continuously active)
   Ex: "The <system name> shall comply with regulation XXX."
   Template : The <system name> shall <response> ※
2. Event-driven
   keyword : <u>when</u>
   Template: WHEN <trigger> the <system name> shall respond
3. State-driven
   Keyword : While
   Template: WHILE <in specific state>, the <system name> shall <response>
4. Option
   Keyword: WHERE
   Template: WHERE <feature is included>, the <system nam> shall <response>
5. Unwanted behavior
   keyword: IF/THEN
   IF <unwanted trigger>, THEN the <system name> shall <response>

Combining 5 templates.

Can state more complicated requirements

Lecture 7

Nonfunctional requirements (NFRs)

functional: What does the software have to do?
Nonfunctional: How well does it have to do it?

Why important?
NFRs determine whether the software works well enough to be used (fast, can handle volume of requests, reliable, familiar interface, secure, etc.)

How to find nonfunctional requirements (NFRs)
- Elicit from stakeholders
- Prototype get feedback from stakeholders
- Apply checklist of 8 major categories of NFRs to each product use case

Ex: Table 11.1 (P.232)

8 categories of nonfunctional requirements
1. Look & feel
2. Usability
3. performance ✗
   timing/accuracy/capacity/throughput/availability/fault-tolerance/reliability also: safety
4. Operational
5. Maintainability
6. Security ✗
7. Cultural and political
8. legal

3 Notes on NFRs
Rationale: If it's not clear why a requirement is needed, add the reason or justification for it.
Fit Criteria
Cross-reference:

Lecture 8

training — choose the best technique(s) for the project
Who are stakeholders?
Any person who has interest in the product & who therefore has requirements for it.
client/user/developer/some are remote: auditor, safety inspector, company lawyer

Interviewing stakeholders
Ask (using their terminology)/listen/feedback your understanding/Get agreement

Exam 1

3 Notes on NFRs Lecture 6

a) Rational (基本原理) if it's not clear why a requirement is needed, add the reason/justification for it

b) Fit Criteria

c) Cross-reference (交叉引用/互相参照) overlaps between FRs & NFRs (Ex: product shall provide a firewall)
overlaps between NFR categories

Requirements hard to write.
figure out what's needed
clear way to express it. } Hard

EARS: Easy Approach to Requirements Syntax (轻松接近 Requirements 句法)
classifies all requirements into 5 basic templates.

1. Ubiquitous 普遍存在的
A ubiquitous requirement is something system must always do. (unconditional & continuously active)
无条件 连续的
Template: The <system name> shall <response>

2. Event-driven 事件驱动. keyword: WHEN
System response is initiated by a triggering event the system detects at the system boundary.
系统反应是由系统在系统边界检测到的触发事件.
Template: WHEN <trigger> the <system name> shall <respone>

3. State-driven 状态驱动的. keyword: WHILE
Template: WHILE <in specific state>, the <system name> shall <response>

4. Option 选择 keyword: WHERE
Template: WHERE <feature is included>, the <system name> shall <response>

5. Unwanted Behavior keyword: IF/THEN
Template: IF <unwanted trigger>, THEN the <system name> shall <response>

Product Use Case Diagram

Adjacent systems 情节, 情景说明

Context Diagram → Product Use Case → Scenarios → Functional Requirements

Trigger: Precondition: steps: .. Lecture 5. Product use case name: Actor: Outcomes.
先决条件. 参与者

Lecture 11

Fit criteria: Chapter 12 适合标准

A requirement needs to be testable

Not: "The interface shall be user-friendly."

To be testable, it needs to be measurable

The test then can show whether the requirement is met

Def: A fit criterion is the measurement of a requirement.

Ex:

New users shall be able to add a road, change & delet a road within 30 mins of their first attempt at using IceBreaker."

- measurable
- units: minutes, new users.
- Testable!

Fit criteria are most commoly needed for nonfunctional requirements.

Lecture 12

Communicating the Requirements: Rationale

Rationale answers: "Why this requirement??"

Def: The rationale is the reason, or justification, for a requirement."

Remove it if some requirements are inconsistent with project's budget and schedule

Lecture 13

SRS: software Requirements Specification

Rabbit	Horse	Elephant
~ project is agile with small team & small increments less time writing requirements	longer release cycle & many stakeholders. Needs some formal documentation since requirements are passed between distributed groups	larger and longer project Many developers.

SRS template

Template has 5 parts, divided into 27 sections.

Project Drivers: Section 1-2

Project Constraints: 3-5

Functional Requirements: 6-9

Non-functional Requirements: 10-17

Project Issues: 18-27

Lecture 14 Chap 16 Appendix A

Part I. **Project Drivers:**

What is the need for the product & who wants it?

1. Purpose of project
   business problem & goal -- how this solves problem
2. Stakeholders

Part II. **Project Constraints**

Restrictions placed on the requirements & design space

3. Mandated constraints
4. Naming conventions & terminology
5. Relevant facts & assumptions *

Part III **Functional Requirements**

6. Scope of the work
   context diagram
7. Business data model & data dictionary
8. Scope of the product
   product use case diagram & scenarios

* 9. Functional requirements
   & their fit criteria, rationale, priority, trace links backward to use case & scenarios
   testable why? ↑ public scenarios ← ER

Part IV **Nonfunctional Requirements**

& the fit criteria, rationale, priority, trace links backward . .

10. look & feel
11. usability & humanity
12. Performance
13. Operational & environmental
14. Maintainability & support
15. Security
16. Cultural
17. legal

Part V. **Project Issues.**

18. Open issues
19. Off-the-shelf solutions
20. New problems
21. Tasks
22. Migration to the new product
23. Risks
24. Costs
25. User documentation & training
26. Waiting room
27. Ideas for solutions

Lecture 15 Validating the Requirements through the Quality Gateway.

Quality Gateway : Chapter 13

Problem: How to keep bad requirements out of our SRS (Software Requirements Specification)?

Solution: "Fly" each potential requirement through our Quality Gateway

Goal: prevent incorrect requirements from going into the SRS

V&V

Verification: build the product right (good RE process)

Validation : build the right product (accurate requirement specs in SRS)

What's a good requirements specification (cont)?

Complete — any missing requirements?
exceptional/failure & alternative scenarios: look for NFR types

Clear — can it be misunderstood?" (use EARS)

Consistent — are descriptions and terms unambiguous?

Conflict-free — does satisfying it mean another requirement can't be satisfied

What's a good requirements specification (cont.)

Checking the value

Needed

Valued

Testable

Feasible

Traceable

What's a . - - - - .

sidebar: Value ratings