Chapter 12, Rationale Management
An aircraft example

A320
- First fly-by-wire passenger aircraft
- 150 seats, short to medium haul

A319 & A321
- Derivatives of A320
- Same handling as A320

Design rationale
- Reduce pilot training & maintenance costs
- Increase flexibility for airline
An aircraft example (2)

A330 & A340
♦ Long haul and ultra long haul
♦ 2x seats, 3x range
♦ Similar handling as A320 family

Design rationale
♦ With minimum cross training, A320 pilots can be certified to fly A330 and A340 airplanes

Consequence
♦ Any change in these five airplanes must maintain this similarity
Overview: rationale

♦ What is rationale?
♦ Why is it critical in software engineering?
♦ Centralized traffic control example
♦ Rationale in project management
  ♦ Consensus building
  ♦ Consistency with goals
  ♦ Rapid knowledge construction
♦ Summary
What is rationale?

Rationale is the reasoning that lead to the system.

Rationale includes:

♦ the *issues* that were addressed,
♦ the *alternatives* that were considered,
♦ the *decisions* that were made to resolve the issues,
♦ the *criteria* that were used to guide decisions, and
♦ the *debate* developers went through to reach a decision.
Why is rationale important in software engineering?

Many software systems are like aircraft:

They result from a large number of decisions taken over an extended period of time.

♦ Evolving assumptions
♦ Legacy decisions
♦ Conflicting criteria

-> high maintenance cost
-> loss & rediscovery of information
Uses of rationale in software engineering

♦ Improve design support
  • Avoid duplicate evaluation of poor alternatives
  • Make consistent and explicit trade-offs

♦ Improve documentation support
  • Makes it easier for non developers (e.g., managers, lawyers, technical writers) to review the design

♦ Improve maintenance support
  • Provide maintainers with design context

♦ Improve learning
  • New staff can learn the design by replaying the decisions that produced it
Representing rationale: issue models

Argumentation is the most promising approach so far:

♦ More information than document: captures trade-offs and discarded alternatives that design documents do not.
♦ Less messy than communication records: communication records contain everything.

Issue models represent arguments in a semi-structure form:

♦ Nodes represent argument steps
♦ Links represent their relationships
ATM Example

Question: Alternative Authentication Mechanisms?

References: Service: Authenticate

Decision: Smart Card + PIN

<table>
<thead>
<tr>
<th></th>
<th>Criteria 1: ATM Unit Cost</th>
<th>Criteria 2: Privacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option 1: Account number</td>
<td>+</td>
<td>−</td>
</tr>
<tr>
<td>Option 2: Finger print reader</td>
<td>−</td>
<td>+</td>
</tr>
<tr>
<td>Option 3: Smart Card + PIN</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
Centralized traffic control

♦ CTC systems enable dispatchers to monitor and control trains remotely
♦ CTC allows the planning of routes and replanning in case of problems
Centralized traffic control (2)

CTC systems are ideal examples of rationale capture:

♦ Long lived systems (some systems include relays installed last century)
  ◆ Extended maintenance life cycle

♦ Although not life critical, downtime is expensive
  ◆ Low tolerance for bugs
  ◆ Transition to mature technology
Issues

- Issues are concrete problems which usually do not have a unique, correct solution.
- Issues are phrased as questions.

input?:Issue

How should the dispatcher input commands?

display?:Issue

How should track sections be displayed?
Proposals

- Proposals are possible alternatives to issues.
- One proposal can be shared across multiple issues.

The display used by the dispatcher can be a text only display with graphic characters to represent track segments.

The interface for the dispatcher could be realized with a point & click interface.
Consequent issue

♦ Consequent issues are issues raised by the introduction of a proposal.

Which terminal emulation should be used for the display?
Criteria

- A criteria represent a goodness measure.
- Criteria are often design goals or nonfunctional requirements.

The CTC system should have at least a 99% availability.

The time to input commands should be less than two seconds.
Arguments

♦ Arguments represent the debate developers went through to arrive to resolve the issue.

♦ Arguments can support or oppose any other part of the rationale.

♦ Arguments constitute the most part of rationale.
Point & click interfaces are more complex to implement than text-based interfaces. Hence, they are also more difficult to test. The point & click interface risks introducing fatal errors in the system that would offset any usability benefit the interface would provide.
Resolutions

♦ Resolutions represent decisions.
♦ A resolution summarizes the chosen alternative and the argument supporting it.
♦ A resolved issue is said to be closed.
♦ A resolved issue can be re-opened if necessary, in which case the resolution is demoted.
Resolutions (2)

- **text-based & keyboard**: Addressed
  - input?: Issue
    - text-based: Proposal
      - terminal?: Issue
        - usability?: Criterion
      - raises
    - usability?: Criterion
  - fails

- **display?**: Addressed
  - point & click: Proposal
    - meets
  - availability?: Criterion
    - availability-first!: Argument
    - is opposed by
Questions, Options, Criteria

- Designed for capturing rationale after the fact (e.g., quality assessment).
- QOC emphasizes criteria
Other issue models: Decision Representation Language

Alternative

AchievesLink

Claim

Procedure

Decision Problem

Goal

Claim

Question

is a good alternative for

achieves

denies

is a result of

supports

denies

supports presupposes

is an answering procedure for

answers

raises
Overview: rationale

♦ What is rationale?
♦ Why is it critical in software engineering?
♦ Centralized traffic control example
♦ Rationale in project management
  ♦ Consensus building (WinWin)
  ♦ Consistency with goals (NFR Framework)
  ♦ Rapid knowledge construction (Compendium)
♦ Summary
Consensus building

Problem

♦ Any realistic project suffers the tension of conflicting goals
  ♦ Stakeholders come from different background
  ♦ Stakeholders have different criteria

Example

♦ Requirements engineering
  ♦ Client: business process (cost and schedule)
  ♦ User: functionality
  ♦ Developer: architecture
  ♦ Manager: development process (cost and schedule)
Consensus building: WinWin

- Incremental, risk-driven spiral process
  - Identification of stakeholders
  - Identification of win conditions
  - Conflict resolution

- Asynchronous groupware tool
  - Stakeholders post win conditions
  - Facilitator detects conflict
  - Stakeholders discuss alternatives
  - Stakeholders make agreements
Consensus building: Model

Win Condition involves Taxonomy Category

issues addresses adopts

Issue Option Agreement

covers
**Consensus building: Process**

1. Identify stakeholders
2. Identify stakeholders’ win conditions
4. Evaluate & resolve risks.
5. Define solution
6. Validate
7. Review & commit
Consensus building: WinWin tool
Consensus building: Experiences

Context
♦ Initial case studies used project courses with real customers
♦ Used in industry

Results
+ Risk management focus
+ Trust building between developers and clients
+ Discipline
− Inadequate tool support
Consistency with goals

Problem
♦ Once multiple criteria have been acknowledged
  ♦ Find solutions that satisfy all of them
  ♦ Document the trade-offs that were made

Example
♦ Authentication should be secure, flexible for the user, and low cost.
Consistency with goals: NFR Framework

♦ NFR goal refinement
  ♦ NFRs are represented as goals in a graph
  ♦ Leaf nodes of the graph are operational requirements
  ♦ Relationships represent “help” “hurt” relationships
  ♦ One graph can represent many alternatives

♦ NFR evaluation
  ♦ Make and break values are propagated through the graph automatically
  ♦ Developer can evaluate different alternatives and compare them
Consistency with goals: Model

Flexibility ×
Low cost +
Account+PIN
Finger Print Reader
SmartCard+PIN

Security
Authentication
Confidentiality
Integrity

AND
OR
Consistency with goals: Process

- Elicit high-level goals
- Evaluate alternatives
- Refine into detailed goals
- Identify operational goals
- Identify goal dependencies
Consistency with goals: Experiences

+ Case studies on existing systems lead to clearer trade-offs
+ Research into integrating NFR framework and design patterns
  - Match NFRs to design pattern “Forces”
  - Link NFRs, design patterns, and functional requirements

– Tool support important
Rapid knowledge construction

Problem
♦ When a company is large enough, it doesn’t know what it does.
  ♦ Knowledge rarely crosses organizational boundaries
  ♦ Knowledge rarely crosses physical boundaries

Example
♦ Identify resources at risk for Y2K and prioritize responses.
Rapid knowledge construction: Compendium

♦ Meeting facilitation
  - Stakeholders from different business units
  - External facilitator

♦ Real-time construction of knowledge maps
  - The focus of the meeting is a concept map under construction
  - Map includes the issue model nodes and custom nodes
    (e.g., process, resource, etc.)

♦ Knowledge structuring for long term use
  - Concept map exported as document outline, process model, memos, etc.
Rapid knowledge construction: Model

Dial Tone, Xmit Dialing Info and Signaling Info to End Office (Task)

- Inputs?
- Telephone instrument off hook
- Remote Terminal (RT)/Central Office Terminal (CO)
- Network interface/demarc
- BA end office switch
- Electric power
- Air conditioning
- Heat
- Diesel fuel for generators

- Network infrastructure?
- Other infrastructure?
- Supply chain?
- Other?

- Current flowing on line
- Dial tone
- Signaling info

- Resources needed?

- Outputs?

- Issues?

This does not include CPE
Rapid knowledge construction: Process example
Rapid knowledge Construction: Experiences

Context
♦ Several industrial case studies, including Y2K contingency planning at Bell Atlantic

Results
♦ Increased meeting efficiency (templates are reused)
♦ Knowledge reused for other tasks
Summary

♦ Rationale can be used in project management
  ♦ To build consensus (WinWin)
  ♦ To ensure quality (NFR Framework)
  ♦ To elicit knowledge (Compendium)

♦ Other applications include
  ♦ Risk management
  ♦ Change management
  ♦ Process improvement

♦ Open issues
  ♦ Tool support
  ♦ User acceptance