

Software Engineering

Lecture 2.2: Rationale Management
(Adapted from slides by Bruegge & Dutoit)

Spring 2019 – Feb 12th

Thomas Troels Hildebrandt, Professor
Software, Data, People & Society Section
Department of Computer Science

UNIVERSITY OF COPENHAGEN



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



A330 & A340

- Long haul and ultra long haul
- 2x seats, 3x range
- Similar handling than 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

- 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 Management important for SE?

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 Management in SE

- 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

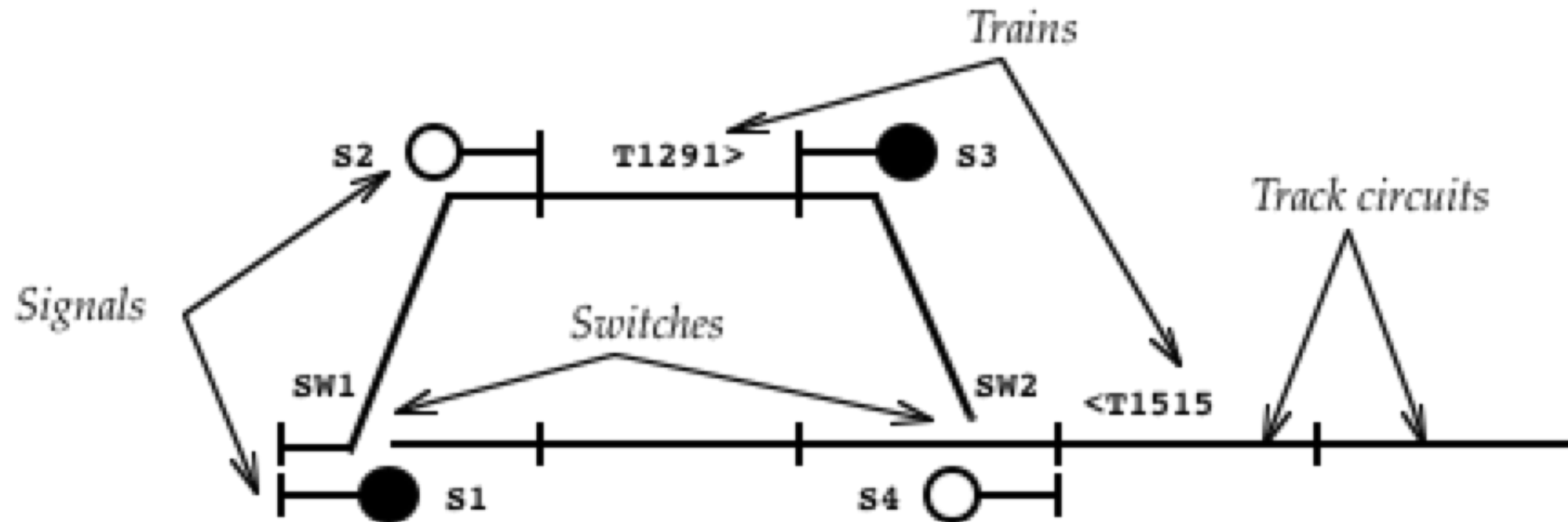
Question: Alternative Authentication Mechanisms?

References: Service: Authenticate

Decision: Smart Card + PIN

	Criteria 1: ATM Unit Cost	Criteria 2: Privacy
Option 1: Account number	+	-
Option 2: Finger print reader	-	+
Option 3: Smart Card + PIN	+	+

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

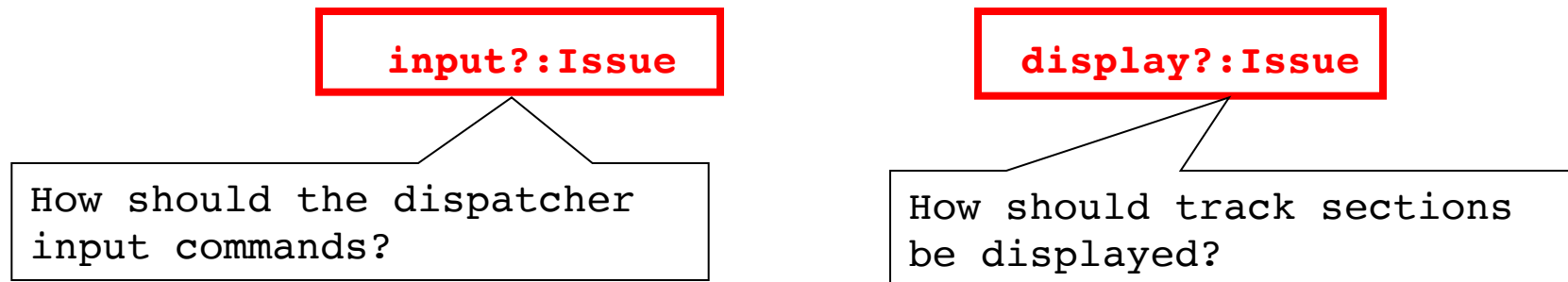
Centralized Traffic Control (2)

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

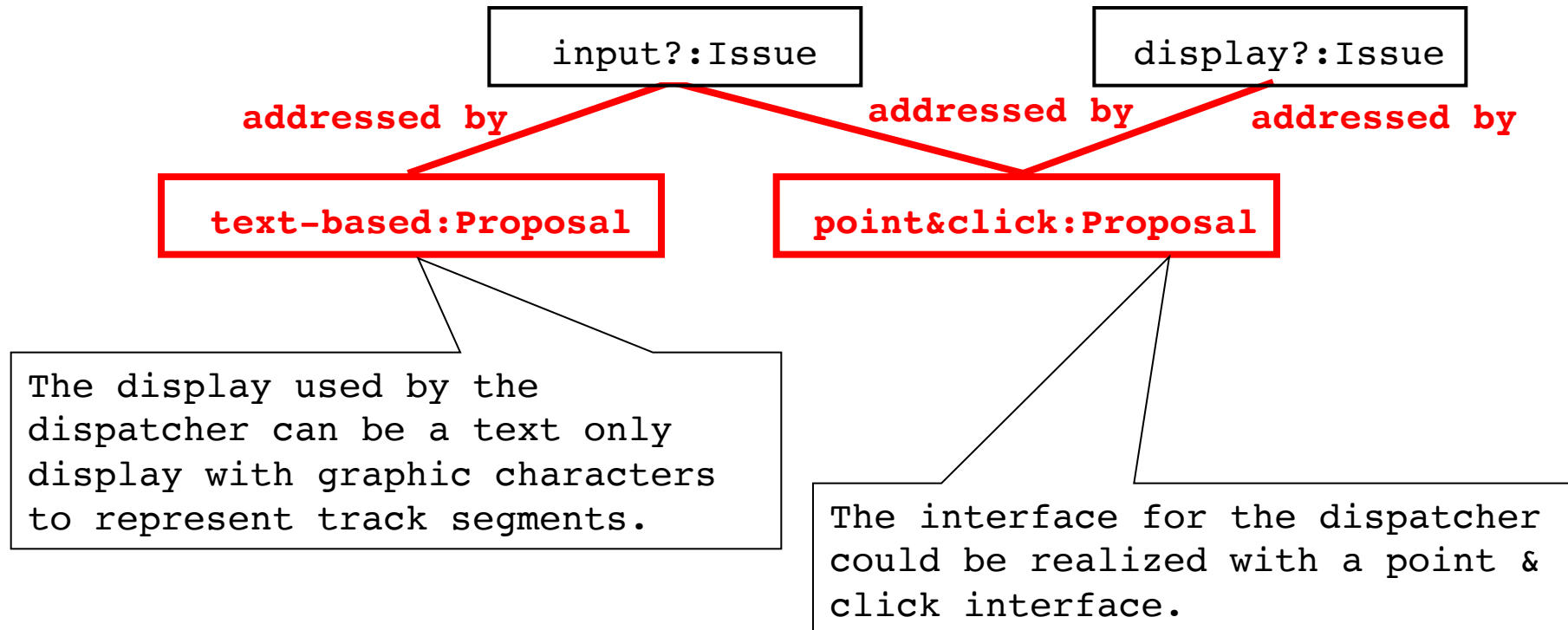
Issues

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



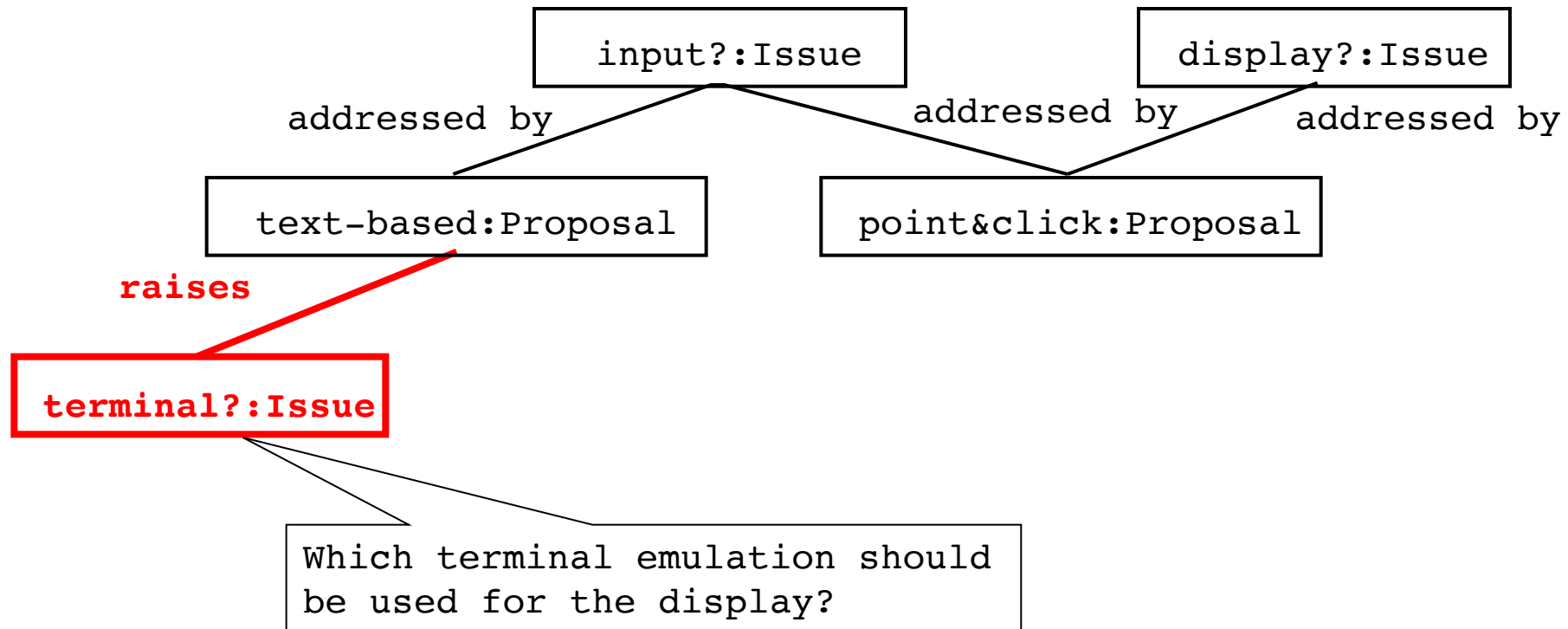
Proposals

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



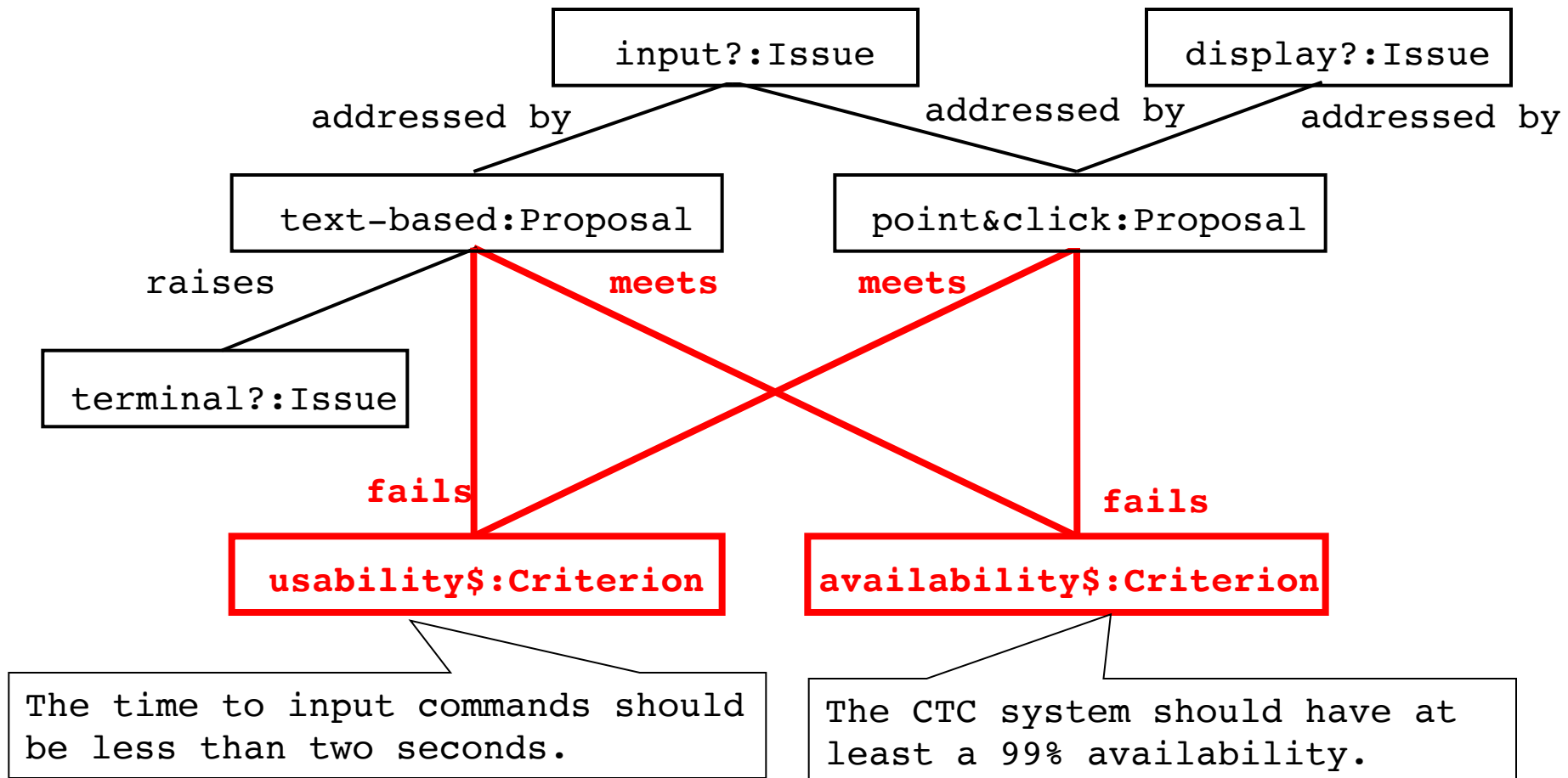
Consequent Issue

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



Criteria

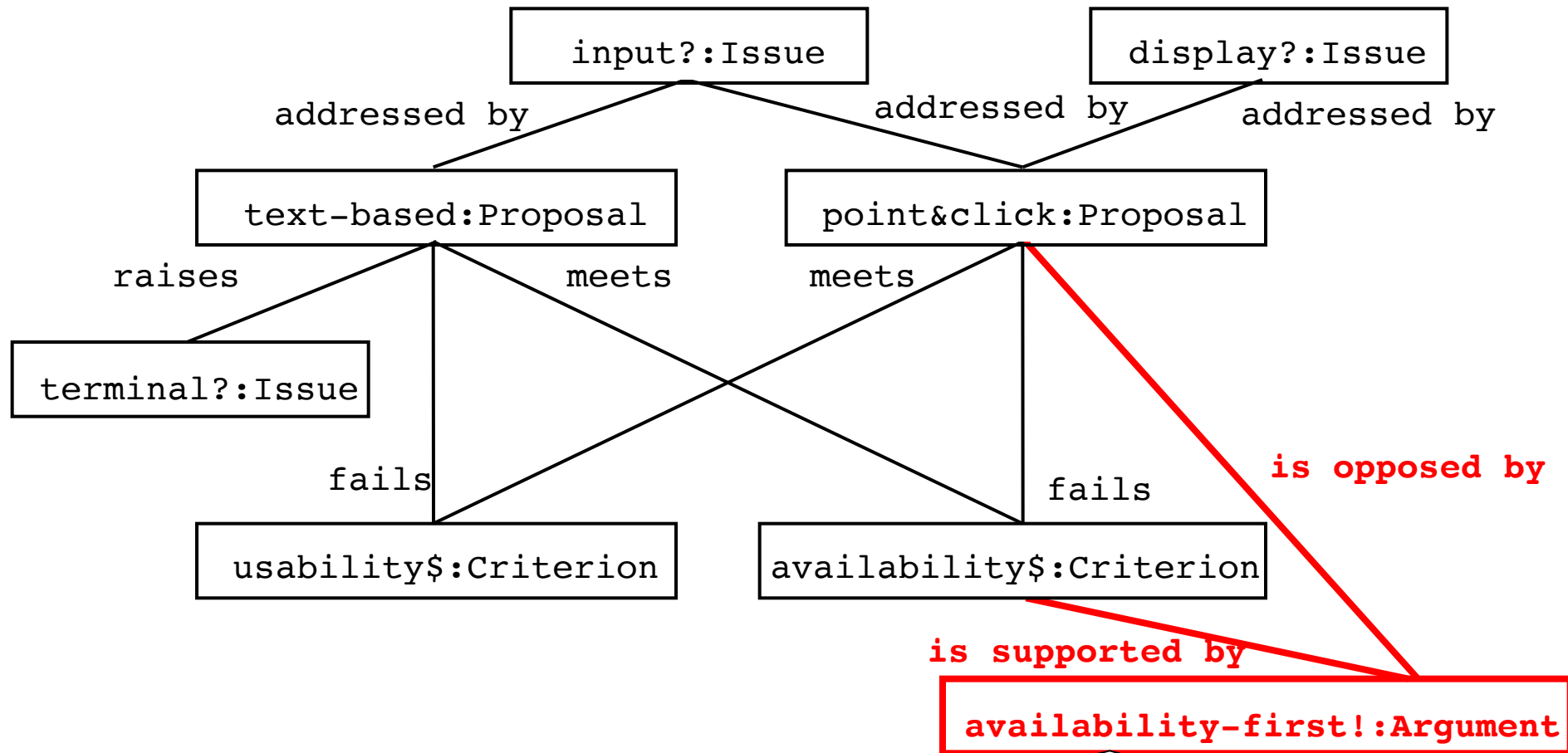
- A criteria represent a goodness measure.
- Criteria are often design goals or nonfunctional requirements.



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.

Arguments (2)

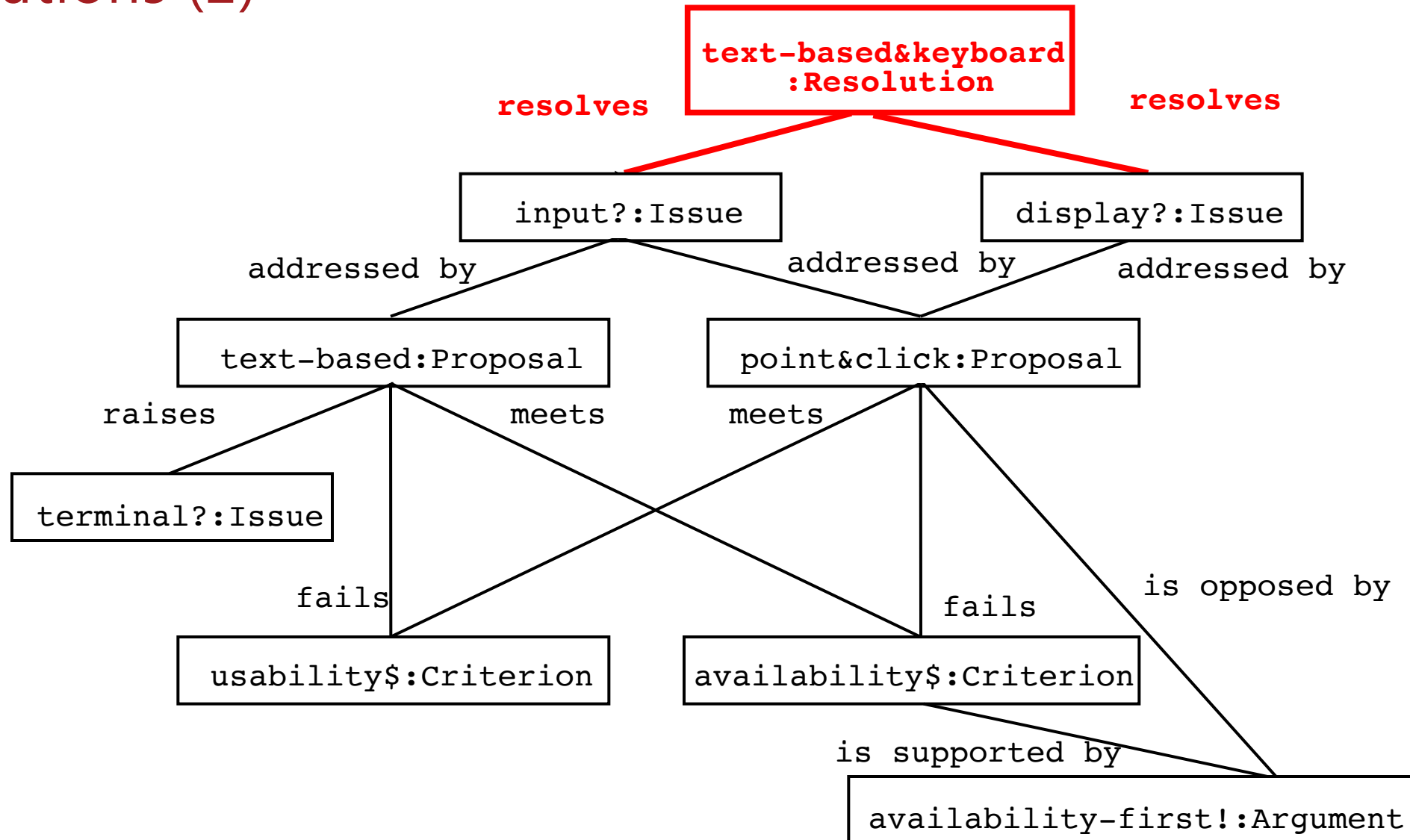


Point&click interfaces are more complex to implement than text-based interfaces. Hence, they are also more difficult to test. The point&click interface risks introducing fatal errors in the system that would offset any usability benefit the interface would provide.

Resolutions

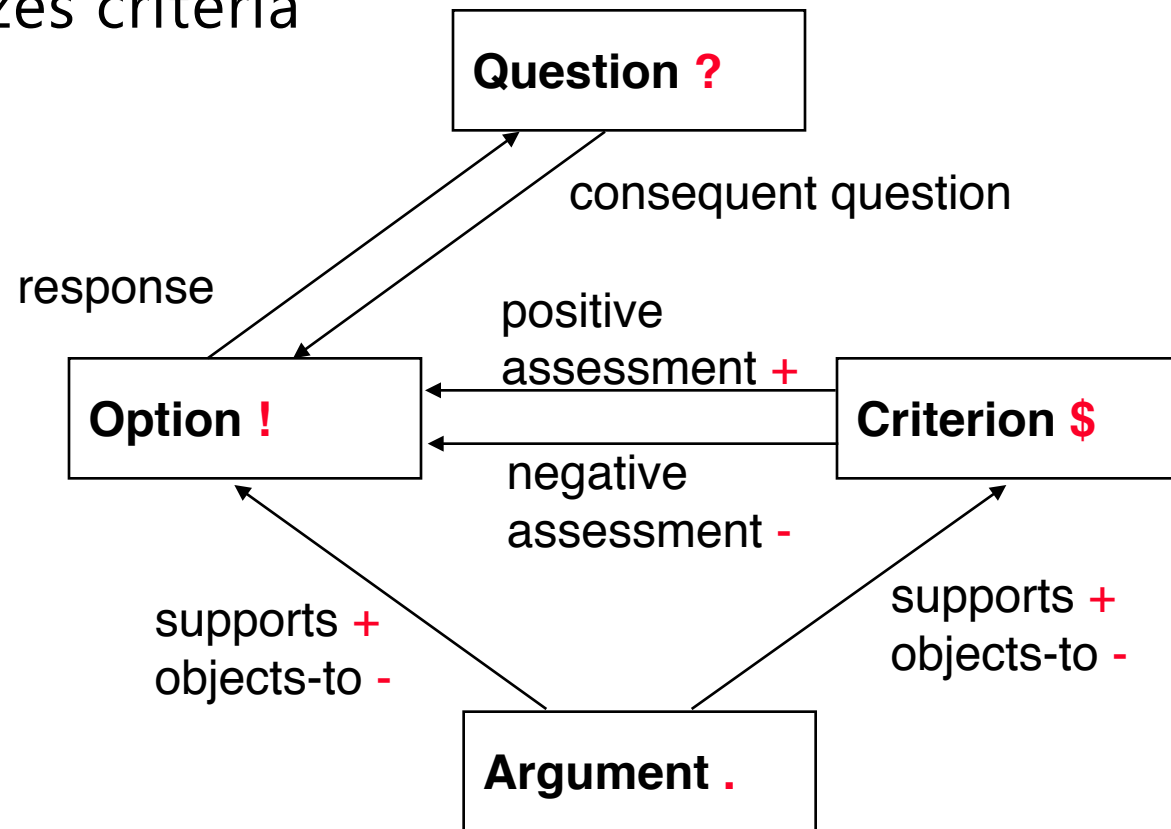
- Resolutions represent decisions.
- A resolution summarizes the chosen alternative and the argument supporting it.
- A resolved issue is said to be closed.
- A resolved issue can be re-opened if necessary, in which case the resolution is demoted.

Resolutions (2)



Questions, Options, Criteria

- Designed for capturing rationale after the fact (e.g., quality assessment).
- QOC emphasizes criteria



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

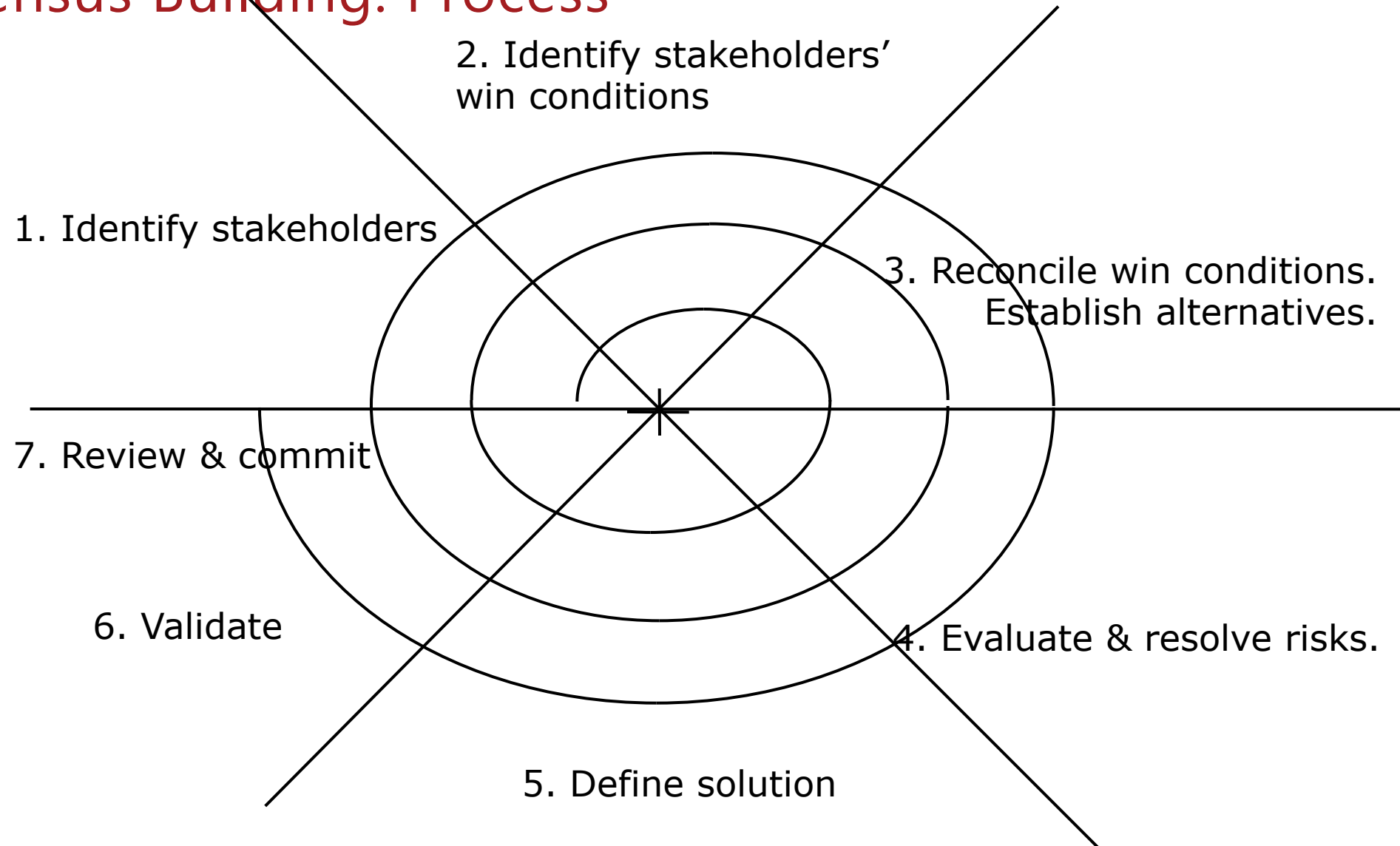
Example

- 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: Process



Consistency with Goals

Problem

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

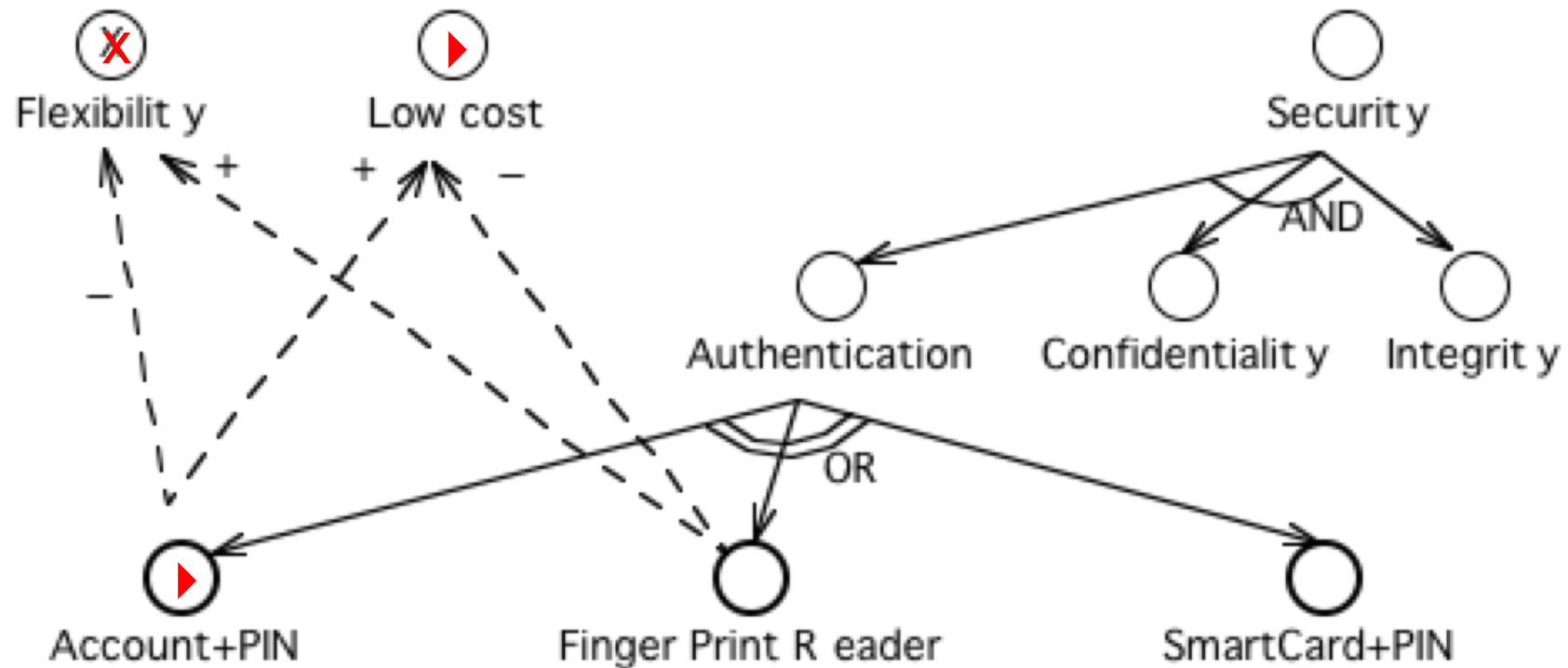
Example

- Authentication should be ***secure, flexible*** for the user, and ***low cost***.

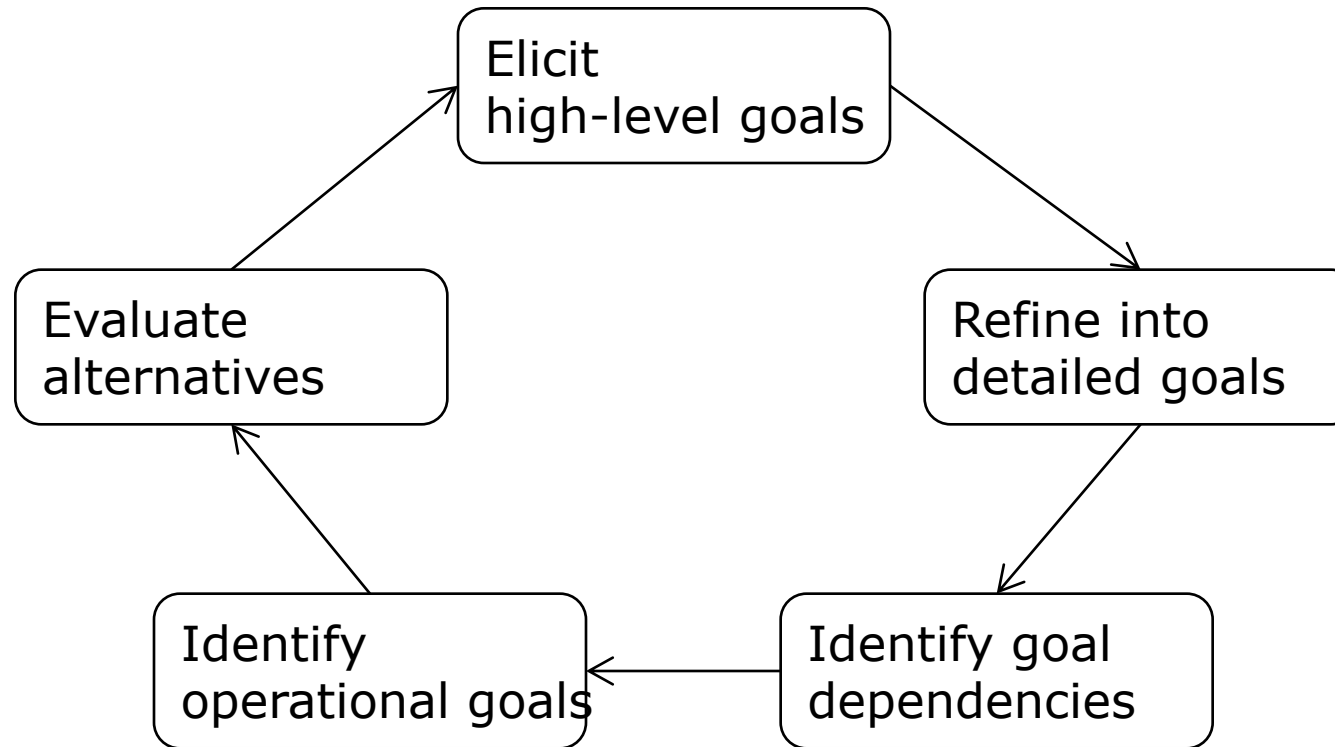
Consistency with Goals: NFR Framework

- 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



Consistency with Goals: Process



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

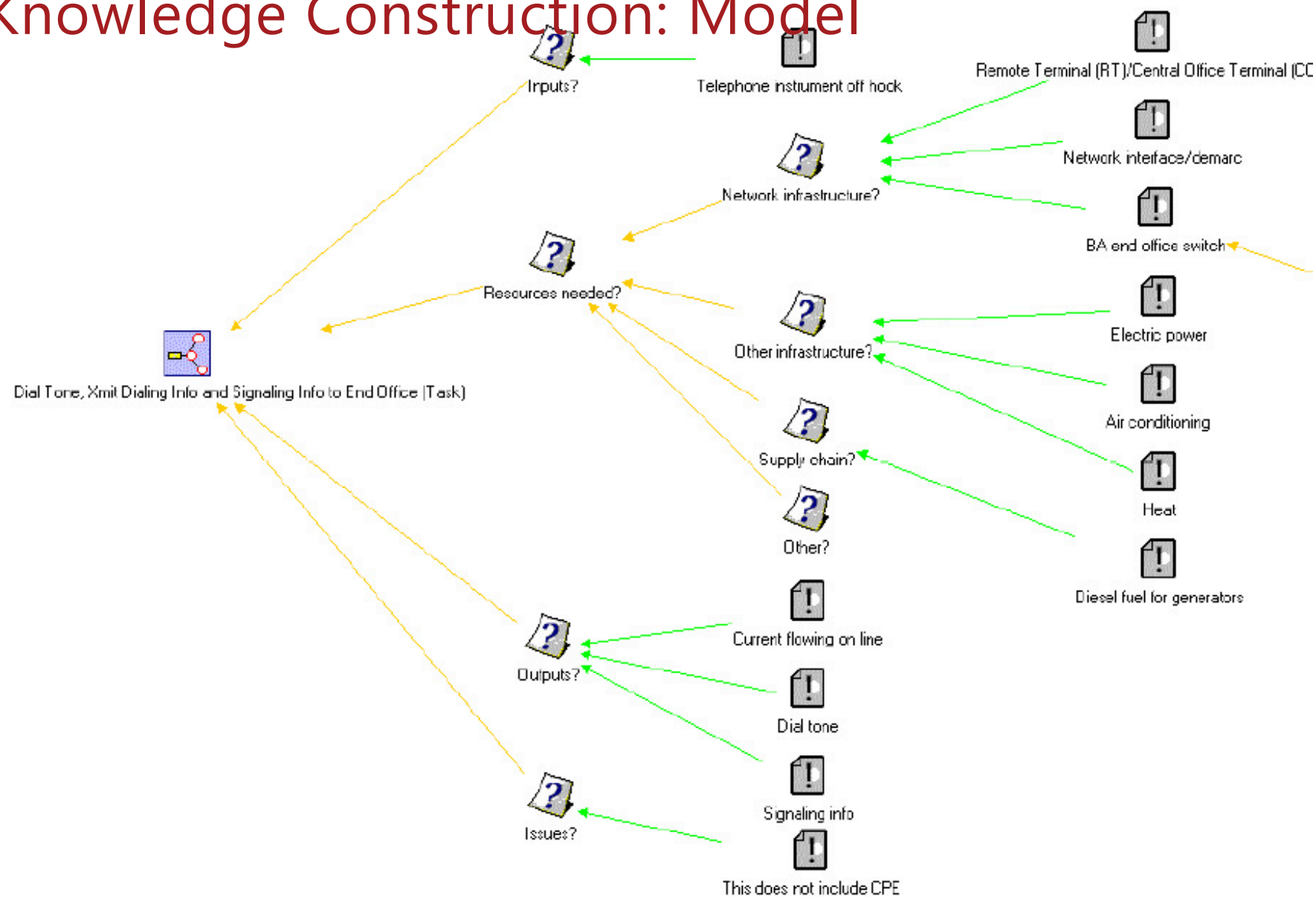
Example

- 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



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

Exercises

- 12-3 : Model the rationale in exercise 12-3 (page 522) with issues, proposals, arguments, criteria, and resolutions, as defined in Section 12.3