

COE691: Software Requirements and Specifications

Dr. Rasha Kashef
Dept. of Electrical, Computer Engineering, and Biomedical Engineering

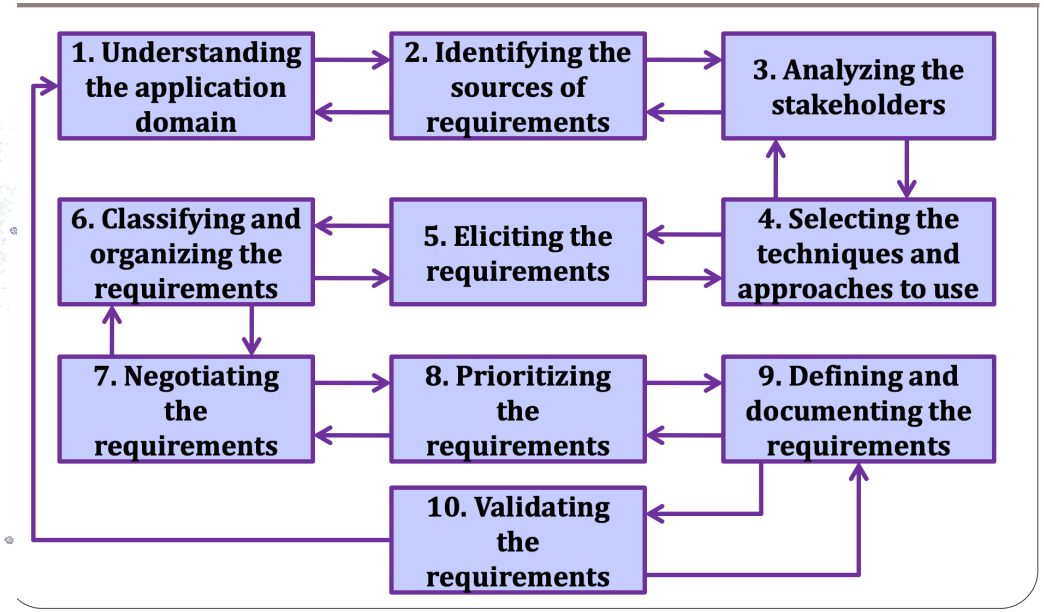
RYERSON UNIVERSITY

Everyone Makes a Mark

Last Week

- **Requirements Elicitation Techniques**
  - **Classical**
  - **Modern**

Week 5: Agenda



Requirements Classification and Organization

Requirements Classification and Organization

- **Objective**
  - **Understand how requirements may be organized in a software requirements document.**

- Unstructured requirements are organized into **multiple levels** of related **groups** or **clusters**.
  - Associated Functional and Non-Functional Requirements need to be **linked and defined thoroughly.**
  - Categorizing requirements with diagram models is a visually appealing way of viewing all requirements.

- Classified by subject and/or system area.
  - Using a System Architecture Model to identify sub-systems and associated requirements is an effective method of classification
- Classified by the source of data.
- Classified by the type of requirements : Functional/non-functional

Functional Requirements Classification

- In order to better understand and manage the large number of requirements, it is important to organize them in logical clusters
- It is possible to classify the requirements by the following categories (or any other clustering that appears to be convenient)
  - Features
  - Use cases
  - Mode of operation
  - User class
  - Responsible subsystem
- This makes it easier to understand the intended capabilities of the product
- And more effective to manage and prioritize large groups rather than single requirements

Requirements Classification – Features

- A Feature is
  - a set of logically related (functional) requirements that provides a capability to the user and enables the satisfaction of a business objective
- The description of a feature should include[1]
  - Name of feature (e.g. Spell check)
  - Description and Priority
  - Stimulus/response sequences
  - List of associated functional requirements

[1] Source: K.E. Wiegers

Requirements Classification – Feature Example (1)

- 3.1 Order Meals
  - 3.1.1 Description and Priority
    - A cafeteria Patron whose identity has been verified may order meals either to be delivered to a specified company location or to be picked up in the cafeteria. A Patron may cancel or change a meal order if it has not yet been prepared.
    - Priority = High.
  - 3.1.2 Stimulus/Response Sequences
    - Stimulus: Patron requests to place an order for one or more meals.
    - Response: System queries Patron for details of meal(s), payment, and delivery instructions.
    - Stimulus: Patron requests to change a meal order.
    - Response: If status is “Accepted,” system allows user to edit a previous meal order.

Source: K.E. Wiegers

Requirements Classification – Feature Example (2)

- Stimulus: Patron requests to cancel a meal order.
- Response: If status is “Accepted, ”system cancels a meal order.

– 3.1.3 Functional Requirements

- 3.1.3.1. The system shall let a Patron who is logged into the Cafeteria Ordering System place an order for one or more meals.
- 3.1.3.2. The system shall confirm that the Patron is registered for payroll deduction to place an order.
- ……

Source: K.E. Wiegers

Non-Functional Requirements: FURPS+ Model

- A possible model for categorizing the requirements is the FURPS+ Model:
- **F**unctional **U**sability **R**eliability **P**erformance **S**upportability +
  - Implementation
  - Interface
  - Operations
  - Packaging
  - Legal

FURPS+ Checklist

- Use the following checklist, based on FURPS+, as an aid when planning requirements elicitation events and during structured walkthroughs to verify whether all types of requirements have been covered in the requirements analysis and documentation. The requirements should cover the following FURPS+ categories:

FURPS+ Categories

- **Functionality:** Defines what the system must be able to do. Includes features, capabilities, security requirements, and user requirements (documented, for example, as system use cases).
- **Usability:** Requirements related to the user interface, such as user-friendliness, accessibility, look and feel, online help, and visual design guidelines.

FURPS+ Categories

- Reliability: The ability of the system to perform under specified routine and nonroutine conditions for a specified period of time. Includes the following:
  - **MTBF** (Mean Time Between Failures): Mean time between service failures of the same service.
  - **MTBSI** (Mean Time Between System/Service Incidents): Mean time between failures (of any service).
  - **MTTR** (Mean Time To Repair): Mean elapsed time to fix and restore a service from the time an incident occurs

FURPS+ Categories

- **Performance:** Describes how the system must behave with respect to time and resources. Includes the following:
  - Speed
  - Efficiency
  - Availability
  - Accuracy
  - Response time
  - Recovery time
  - Start-up time
  - Resource usage
  - Throughput (transactions per unit time)

FURPS+ Categories

- **Supportability:** Requirements related to the ability to monitor and maintain the system. Includes abilities to test, configure, install, and upgrade the system.
- Plus (+): Additional constraints on the system, including the following:
  - Design requirements.
  - Implementation requirements: Constraints on the coding and construction of the system. Includes constraints on platforms, coding languages, and standards.
  - Interface requirements: Capability to interact with specified external systems and the nature of those interactions (protocols, formats, and so on).
  - Physical requirements: Physical constraints on the hardware. Includes requirements related to size, temperature control, materials, and so on.
  - Legal, compliance, regulatory, and copyright requirements and constraints.

Non-functional classifications

- Product requirements
  - Requirements which specify that the delivered product must behave in a particular way e.g. execution speed, reliability, etc.
- Organisational requirements
  - Requirements which are a consequence of organisational policies and procedures e.g. process standards used, implementation requirements, etc.
- External requirements
  - Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc.

Requirements Prioritization and Negotiation

Requirements Prioritization and Negotiation

- The requirements are prioritized, and conflicts are found and resolved through negotiation
- Two main stages:
  - Prioritize requirements and find/resolve conflicts through negotiation.
  - Stakeholders usually meet to resolve differences and agree on compromises.

Requirements Prioritization and Negotiation

- Requirements should be **prioritized** based on their **importance to the overall system**.
- During stakeholder negotiations:
  - An appropriate **priority system** needs to be defined in order to classify requirements and specifications.
    - Ex. Priorities: Critical, High, Baseline, Low, Utility, etc.
  - Requirement conflicts need to be resolved through **compromise**.
  - A **summary** of each **negotiation** meeting should be included within the requirements documentation.

Requirements Prioritization

- Why Prioritization is needed
  - Basic Trade-offs
- Cost-Value Approach
  - Sorting Requirements by cost/value
  - Estimating Relative Costs/Values using AHP
- What if stakeholders disagree?
  - Visualizing differences in priority
  - Resolving Disagreements

Basics of Prioritization

- Need to select what to implement
  - Customers (usually) ask for way too much
  - Balance time-to-market with amount of functionality
  - Decide which features go into the next release
- For each requirement/feature, ask:
  - How important is this to the customer?
  - How much will it cost to implement?
  - How risky will it be to attempt to build it?
- What to include/exclude?
  - Some requirements *must* be included
  - Some requirements should definitely be excluded
  - That leaves a pool of “nice-to-haves”, which we must select from.

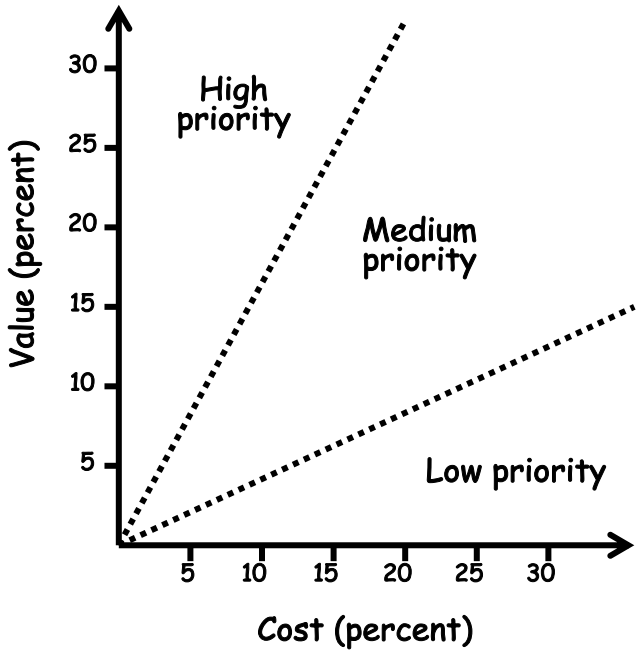
Prioritization Techniques

- Analytical Hierarchy Process (AHP)
  - N X (n-1)/2 at each hierarchy level
- Cumulative Voting, the 1—Dollar test
  - Given 100 Imaginary Units (money, hours, etc.) to distribute between the requirements
- Numerical Assignments (grouping)
  - Using three groups, critical, standard, and optional
- Ranking
  - The most important requirements are ranked from 1 to n
- Top-Ten Requirements

Source: Adapted from Karlsson & Ryan 1997

A Cost-Value Approach

- Calculate return on investment
  - Assess each requirement's importance to the project as a whole
  - Assess the relative cost of each requirement
  - Compute the cost-value trade-off:



Estimating Cost & Value

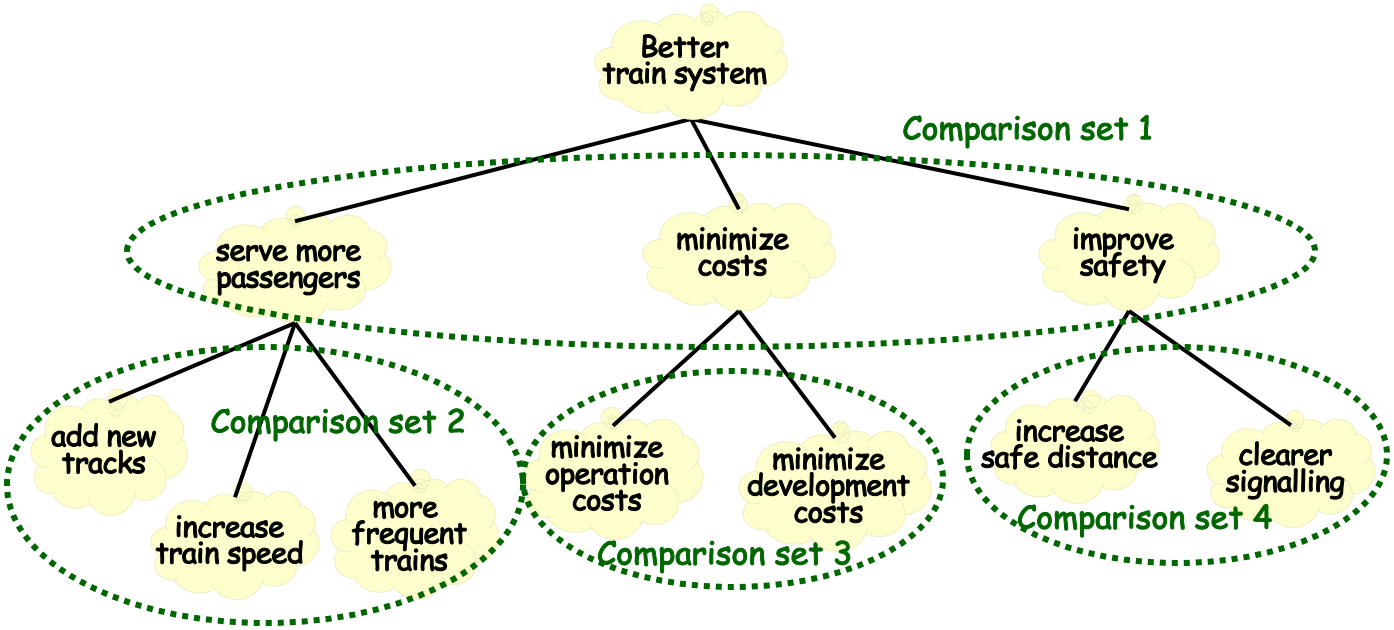
- Two approaches:
  - Absolute scale (e.g. dollar values)
    - Requires much domain experience
  - Relative values (e.g. less/more; a little, somewhat, very)
    - Much easier to elicit
    - Prioritization becomes a sorting problem

Some complications

- Hard to *quantify* differences
  - easier to say “x is more important than y”…
  - …than to estimate by how much.
- Not all requirements comparable
  - E.g. different level of abstraction
  - E.g. core functionality vs. customer enhancements
- Requirements may not be independent
  - No point selecting between X and Y if they are mutually dependent
- Stakeholders may not be consistent
  - E.g. If X > Y, and Y > Z, then presumably X > Z?
- Stakeholders might not agree
  - Different cost/value assessments for different types of stakeholder

Hierarchical Prioritization

- Group Requirements into a hierarchy
  - E.g. A goal tree
  - E.g. A NFR tree
- Only make comparisons between branches of a single node:



Source: Adapted from Karlsson & Ryan 1997

Analytic Hierarchy Process (AHP)

- Create n x n matrix (for n requirements)
  - For element (x,y) in the matrix enter:
    - 1 - if x and y are of equal value
    - 3 - if x is slightly more preferred than y
    - 5 - if x is strongly more preferred than y
    - 7 - if x is very strongly more preferred than y
    - 9 - if x is extremely more preferred than y
    - (use the intermediate values, 2,4,6,8 if compromise needed)
  - …and for (y,x) enter the reciprocal.
- Estimate the eigenvalues:
  - E.g. “averaging over normalized columns”
    - Calculate the sum of each column
    - Divide each element in the matrix by the sum of it’s column
    - Calculate the sum of each row
    - Divide each row sum by the number of rows
- This gives a value for each requirement:
  - …giving the estimated percentage of total value of the project

Source: Adapted from Karlsson & Ryan 1997

AHP example - estimating costs

	Req 1	Req 2	Req 3	Req 4
Req 1	1	1/3	2	4
Req 2	3	1	5	3
Req 3	1/2	1/5	1	1/3
Req 4	1/4	1/3	3	1

Normalise columns

	Req 1	Req 2	Req 3	Req 4
Req 1	0.21	0.18	0.18	0.48
Req 2	0.63	0.54	0.45	0.36
Req 3	0.11	0.11	0.09	0.04
Req 4	0.05	0.18	0.27	0.12

Sum the rows

sum	sum/4
1.05	0.26
1.98	0.50
0.34	0.09
0.62	0.16

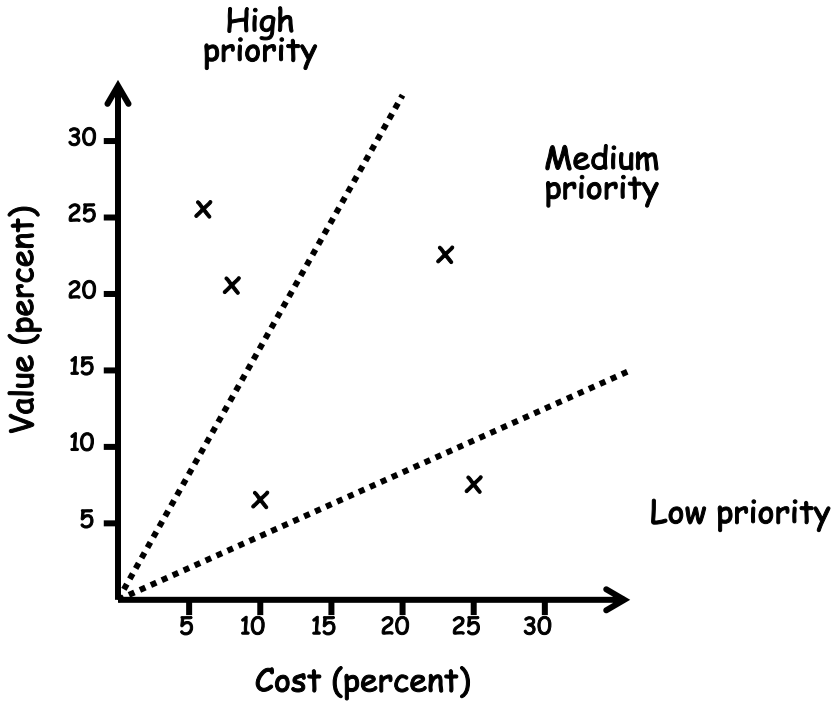
Result

Req1 - 26% of the cost
Req2 - 50% of the cost
Req3 - 9% of the cost
Req4 - 16% of the cost

Source: Adapted from Karlsson & Ryan 1997

Plot ROI graph

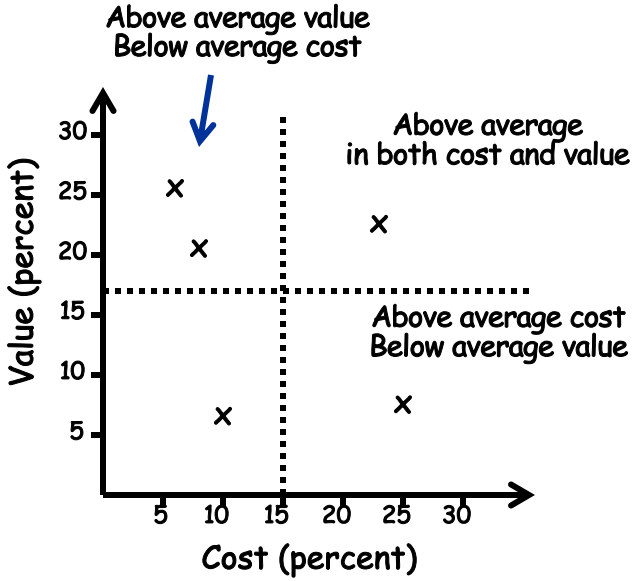
- Do AHP process twice:
  - Once to estimate relative value
  - Once to estimate relative cost
- Use results to calculate ROI ratio:

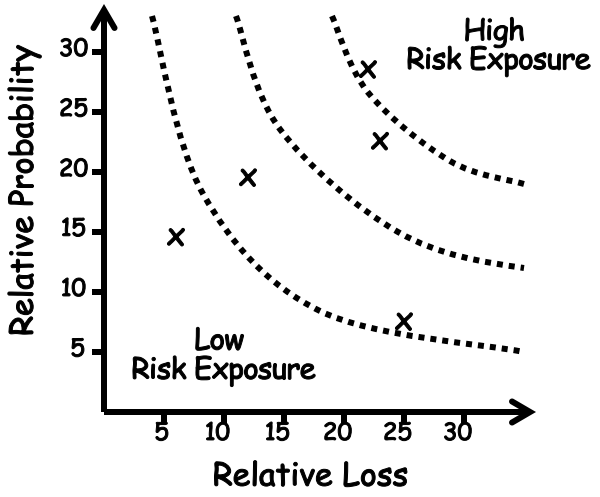


Source: Adapted from Park et al, 1999

Other selection criteria

- ROI ratio is not the only way to group requirements





Requirements Negotiation : Stakeholder conflicts

Requirement and Stakeholder conflicts

Interacting with stakeholders may be difficult because:

- **They have vague knowledge of what they want.**
  - Unrealistic Demands or Expectations.
  - Cannot Define what they want the system to do.

- **Ideas are expressed with implicit knowledge.**
  - Without prior experience of the end-user's environment, an engineer may not understand the stakeholder's meaning.

- **Different Stakeholders input different requirements that are expressed in different ways.**
  - **Ambiguity** between personnel may lead to **conflicting** or **redundant** requirements.

Requirement and Stakeholder conflicts

Interacting with stakeholders may also be difficult because:

- **Political factors may influence system requirements**
  - A business stakeholder could demand certain functionality to increase their power in the organization.
- **Dynamic Economic and Business Conditions**
  - Particular requirements may change throughout the process due to factors outside of the engineers' control.
  - I.E. Company is restructured, bought out, closes.
- Key Point:
  - It is very common that different **stakeholders** might be requesting **requirements** to be included that **conflict** with each other. This is simply a result of **conflicting** priorities between **stakeholder** groups and they may not realize they are requesting **conflicting requirements**

Requirement and Stakeholder conflicts:

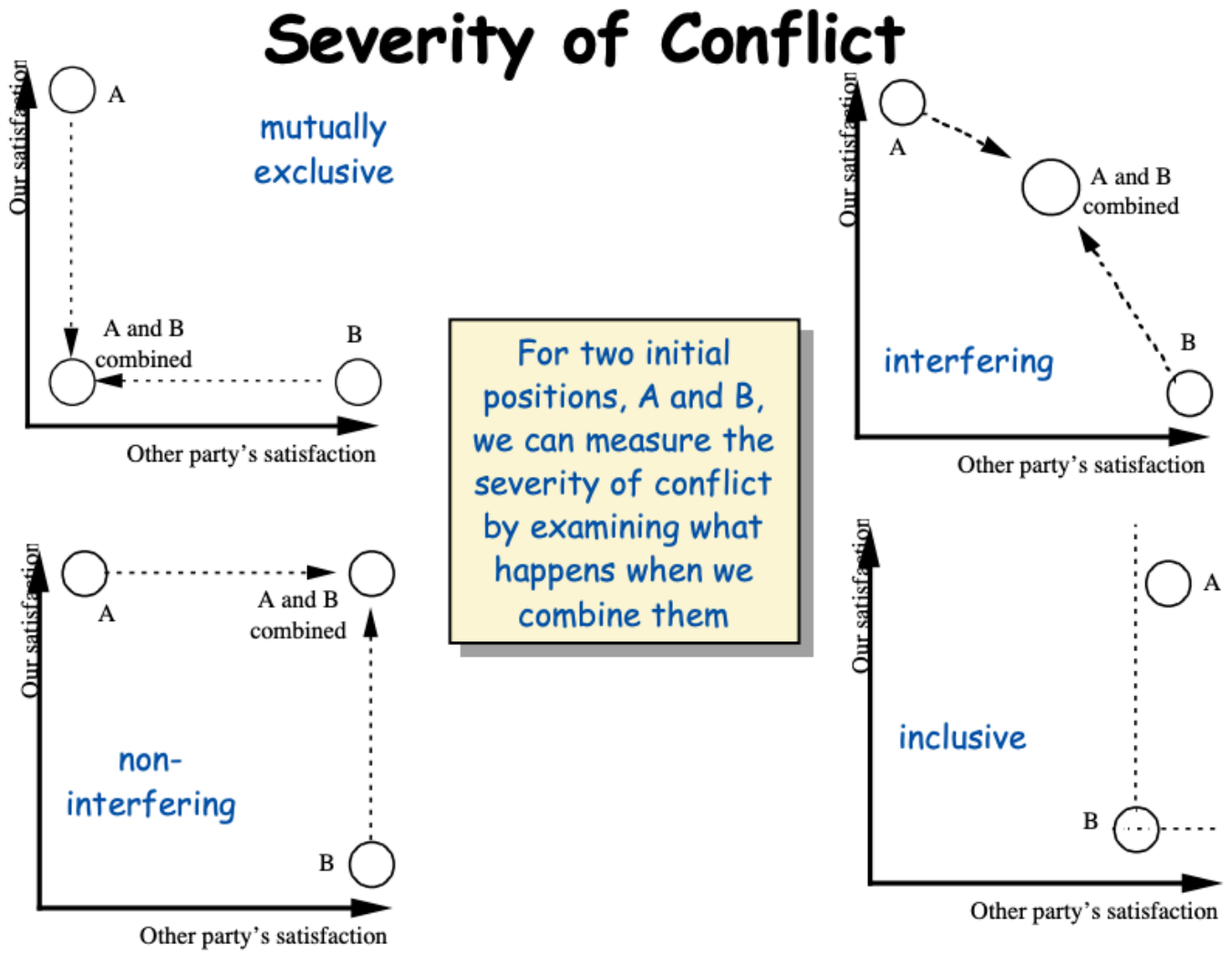
- **Functional and Non-Functional Requirements are bound to conflict.**
  - Stakeholders will **adamantly defend** their own requirements and try to **repress other** conflicting requirements.
- **Regular negotiations should be held to come to a compromise.**
  - A stakeholder whose ideas are voted down may try to undermine the rest of the process due to spite.
- **All conflicts and decisions must be recorded in the documentation for archiving purposes.**

Conflicting Requirements : Example

- Two **requirements** are **conflicting** if you cannot implement them both the solution to one **requirement** prohibits implementing the other. For **example**, if one **requirement** asks for the product to "be available to all" and another says it shall be "fully secure," then both **requirements** cannot be implemented as specified.

- An **example** of a **conflicting requirement** might be that the Human Resources stakeholder group explicitly requests to capture the age of an employee, but the Data Privacy team is saying that the age of the employee may not be captured or used in reporting

Severity of Conflicts



Causes of Conflict

- **Deutsch (1973):**
  - control over resources
  - preferences and nuisances (tastes or activities of one party impinge upon another)
  - values (a claim that a value or set of values should dominate)
  - beliefs (dispute over facts, information, reality, etc.)
  - the nature of the relationship between the parties.
- **Robbins (1989):**
  - communicational (insufficient exchange of information, noise, selective perception)
  - structural (goal compatibility, jurisdictional clarity, leadership style)
  - personal factors, (individual value systems, personality characteristics.

Interesting Results

- Deviant behaviour & conflict are normal in small group decision making
- More aggression and less co-operation when communication is restricted
  - a decrease in communication tends to intensify a conflict (the contact hypothesis)
- Heterogeneous teams experience more conflict;
- Homogeneous groups are more likely to make high risk decisions (groupthink)
- Effect of personality is overshadowed by situational and perceptual factors

Basic Approaches in Conflict Resolutions

- **Negotiation: is a collaborative exploration**
  - Participants attempt to find a settlement that satisfies all parties as much as possible
  - also known as:
    - integrative behaviour
    - constructive negotiation
  - Distinct from:
    - distributive/competitive negotiation
- **Competition is maximizing your own gain**
  - no regard for the degree of satisfaction of other parties.

Basic Approaches in Conflict Resolutions

- **Third Part Resolution**
  - Participants appeal to outside source
  - the rule-book, a figure of authority, or the toss of a coin.
  - can occur with the breakdown of either negotiation or competition as resolution methods.
- **Types of third-party resolution**
  - Judicial: cases presented by each participant are taken into account
  - Extra-judicial: a decision is determined by factors other than the cases presented (e.g. relative status of participants).
  - Arbitrary: e.g. toss of a coin

Basic Approaches in Conflict Resolutions

- Bidding and Bargaining
  - Bidding:
    - participants state their desired terms
  - Bargaining:
    - participants search for a satisfactory integration of bids.


’ Goals Analysis
and Modeling

What is a goal?

- A stakeholder objective for the system
  - The system includes the software and its environment
- Who are stakeholders?
  - Anyone who has an interest in the system
  - Customers, end users, system developers, system maintainers...
- What is a goal model?
  - A hierarchy of goals
  - Relates the high-level goals to low-level system requirements

Where do goals come from?

- Conveyed by stakeholders
- Disclosed in requirements documents
- Analysis of similar or current system
- Elaborating other goals

Identifying Stakeholders' Goals

- Advantages
  - Reasonably intuitive
  - Explicit declaration of goals provides sound basis for conflict resolution
- Disadvantages
  - Captures a static picture - what if goals change over time?
  - Can regress forever up (or down) the goal hierarchy

Goal or Not a Goal?

- Goal Examples:
  - Credit card information is kept private
  - Credit card information is accurate
  - Safe transportation
  - Highly reliability
- Non-goal Examples:
  - The system will be implemented in C++
  - The paint colors for the cars will be yellow, orange, and red

Goal Exercise

- **Order the list of goals from high-level concern to low-level concern**
  - User receives a request for a timetable from a system
  - Collect timetables by system
  - Schedule meeting
  - System collect timetables from user
  - Collect timetables

1– Schedule meeting
2– Collect timetables
3– Collect timetables by system
4– System collect timetables from user
5– User receives a request for a timetable

Types of Goals

- **Functional (Hard):** Describe functions that must be carried out
  - Describe functions the system will perform
  - Well defined criteria for satisfaction
  - E.g., System collects timetables from user
  - Satisfaction goals
  - Information goals

- **Non-functional** (Soft or fuzzy) : Cannot really be fully satisfied
  - Describe desired system qualities
  - Hard to define; satisficed rather than satisfied
  - Reliability
    - E.g., System should be reliable
  - Quality
    - E.g., System should be high quality.
  - Accuracy
  - Performance
  - Security

Different categories of Goals (Requirements)

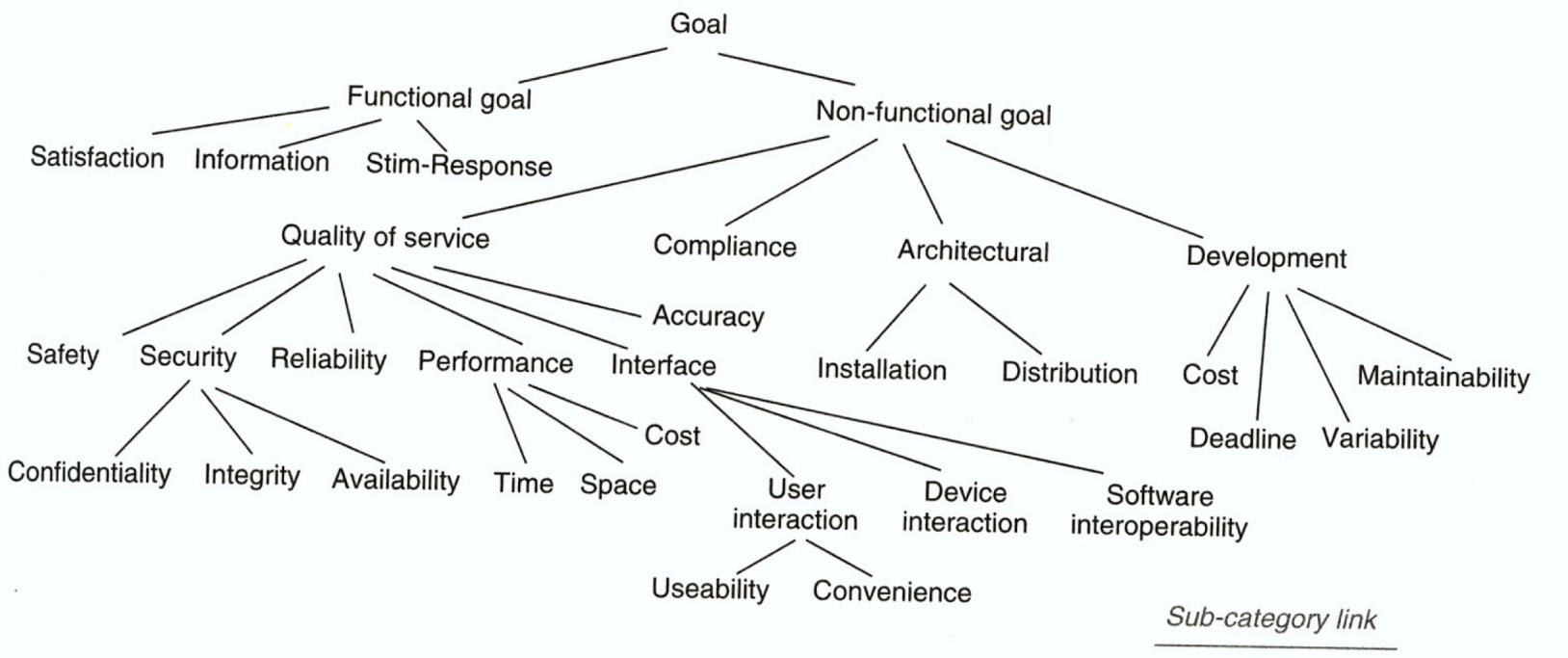


Figure 7.5 *Goal categories*

This is the same as the classification of requirements into functional and non-functional (with all its sub-categories)

When and why to use goal models

- Early requirements elicitations “Why”
  - Focus on identifying problems
  - Exploring system solutions and alternatives
  - Done before UML modeling

- Why use goal models?
  - Give rationale for requirements
  - Identify stable information
  - Guide requirement elaboration

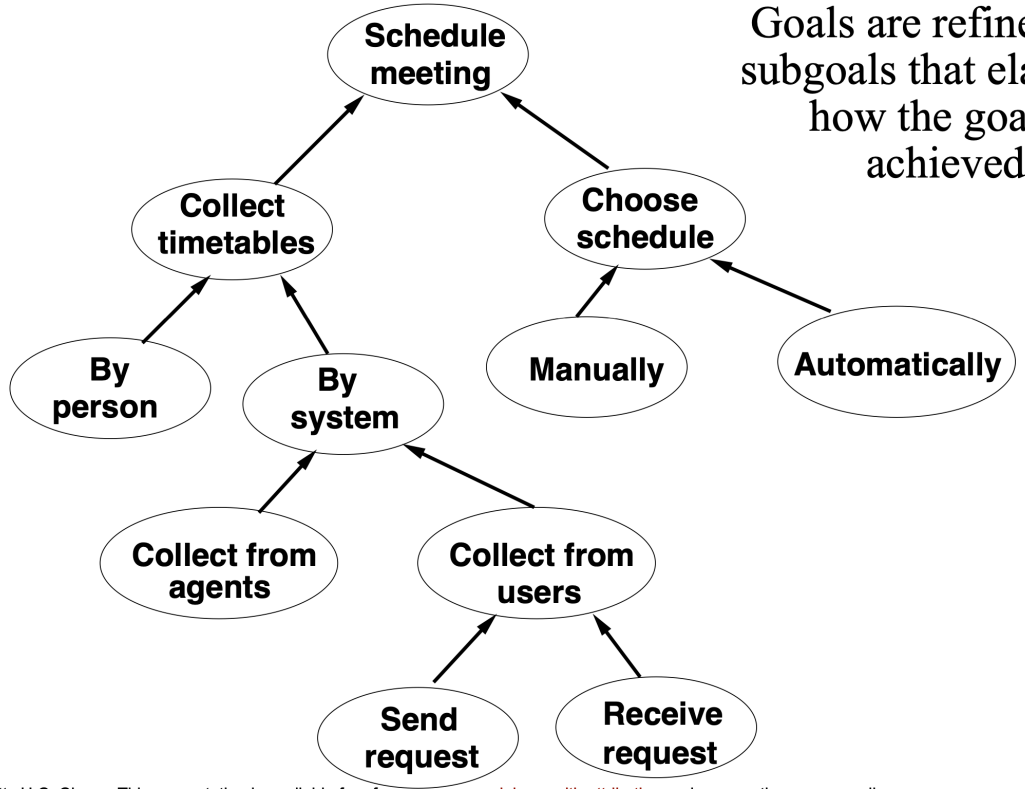
Source: *Adapted from Anton, 1996.*

Identifying Stakeholders' Goals

- **Approach**
  - Focus on *why* a system is required
  - Express the 'why' as a set of stakeholder goals
  - Use goal refinement to arrive at specific requirements
  - Goal analysis
    - document, organize and classify goals
  - Goal evolution
    - refine, elaborate, and operationalize goals
  - Goal hierarchies show refinements and alternatives

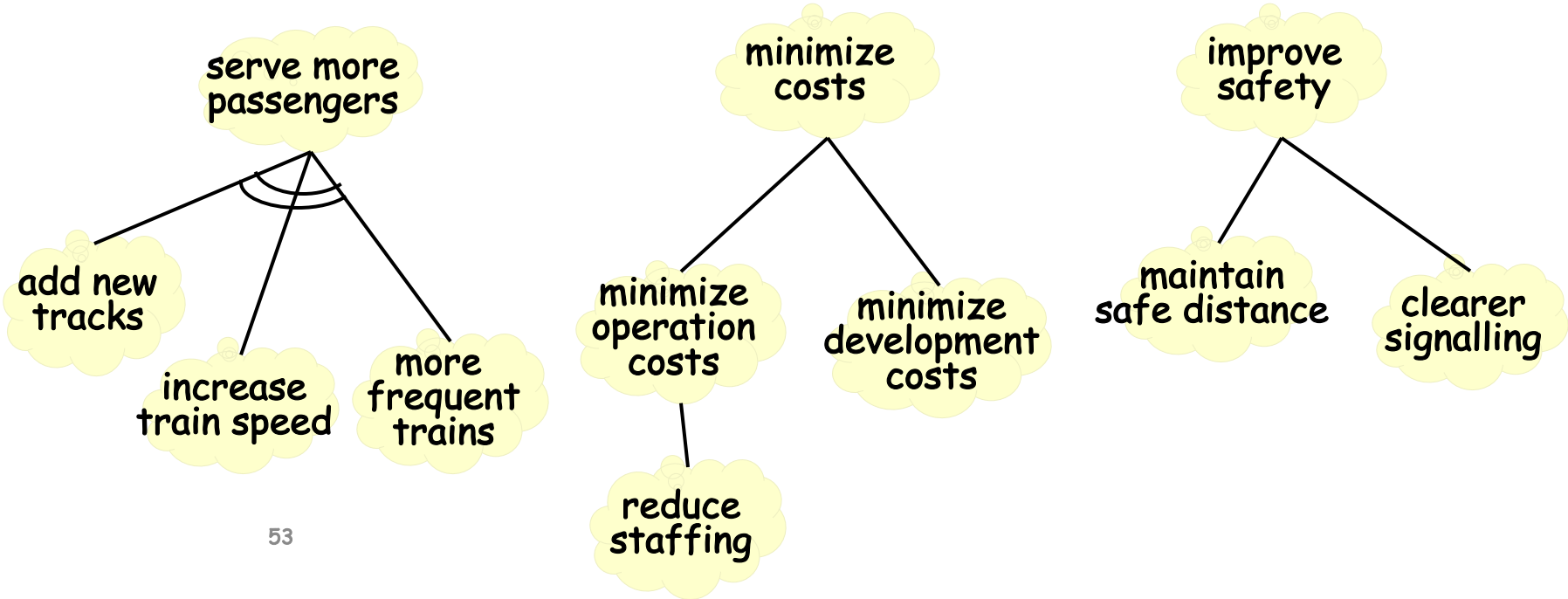
Running Example

- Meeting Scheduler
  - Assists the initiator in scheduling a meeting
  - Meeting should be convenient for participants
    - Participants should be available
  - Modeled using the i* goal notation



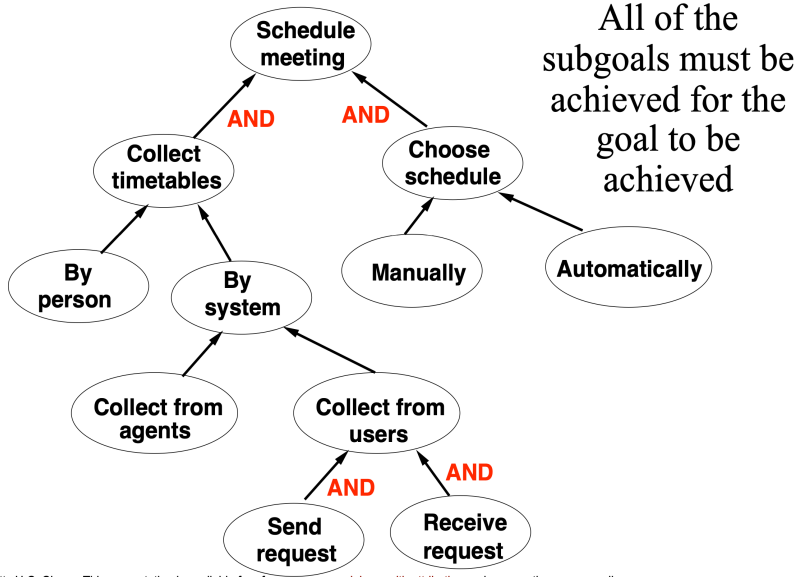
Softgoals

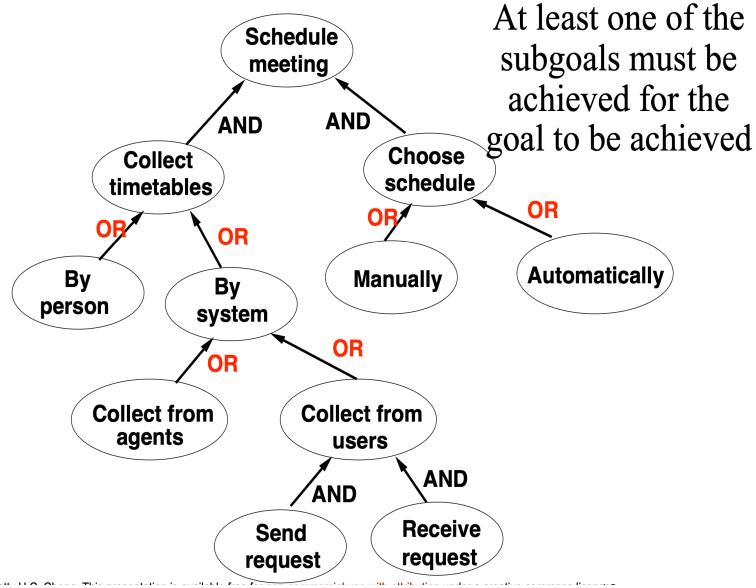
- Some goals can never be fully satisfied
  - Treat these as softgoals
    - E.g. “system be easy to use”; “access be secure”
    - Also known as ‘non-functional requirements’; ‘quality requirements’
  - Will look for things that contribute to satisficing the softgoals
  - E.g. for a train system:



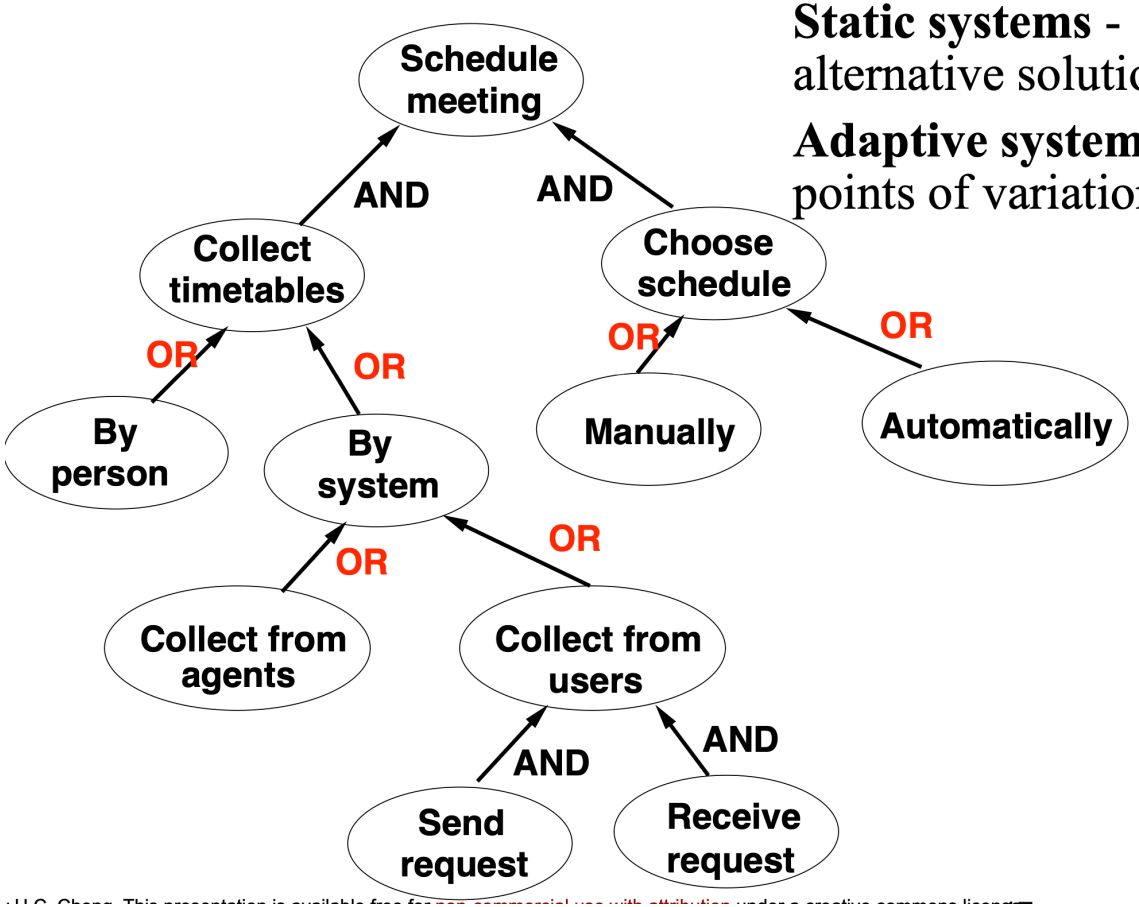
Goal refinement

- Goal refinement: expressing how a more abstract goal can be established by a set of more low-level goals
  - AND refinement
  - OR refinement





Interpretations of OR Refinements

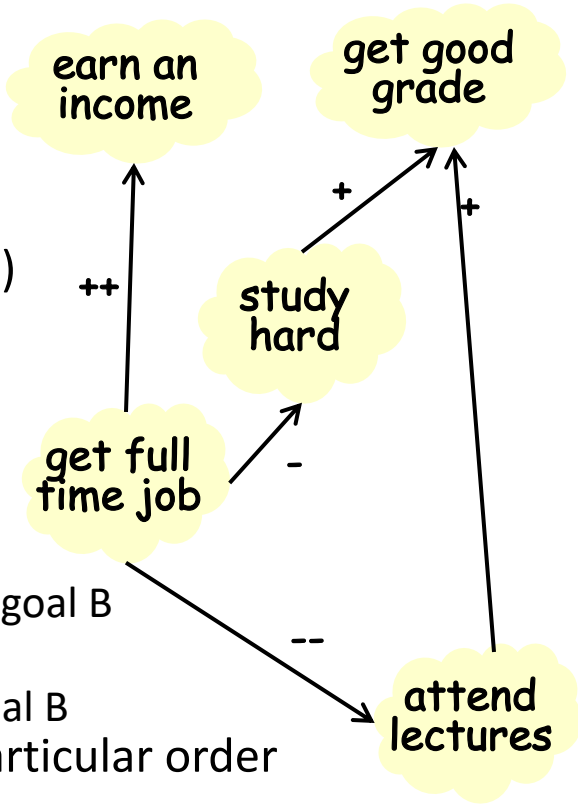


Static systems - alternative solutions

Adaptive systems – points of variation

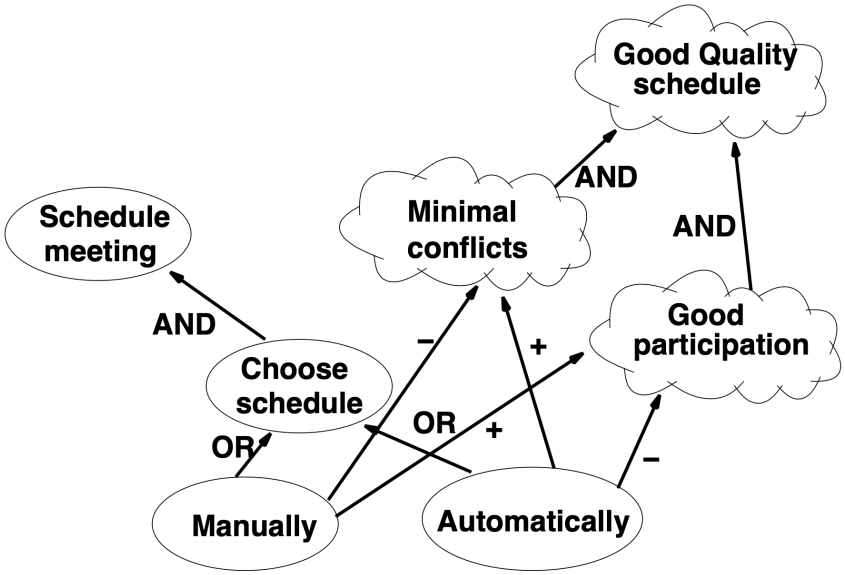
Goal Analysis

- Goal Elaboration:
  - "Why" questions explore higher goals (context)
  - "How" questions explore lower goals (operations)
  - "How else" questions explore alternatives
- Relationships between goals:
  - One goal helps achieve another (+)
  - One goal hurts achievement of another (-)
  - One goal makes another (++)
    - Achievement of goal A guarantees achievement of goal B
  - One goal breaks another (--)
    - Achievement of goal A prevents achievement of goal B
  - Precedence ordering – must achieve goals in a particular order
- Obstacle Analysis:
  - Can this goal be obstructed, if so how?
  - What are the consequences of obstructing it?

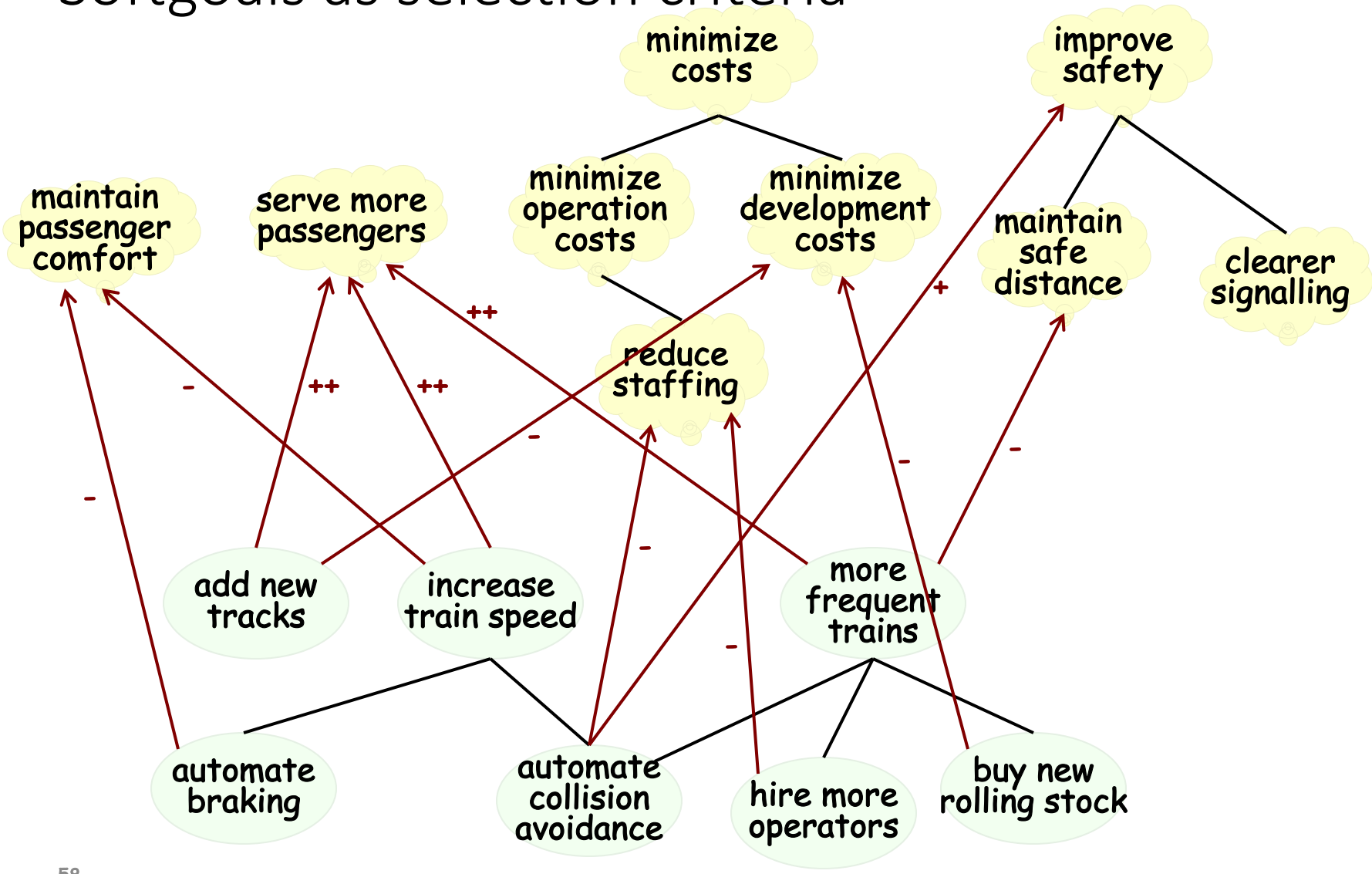


Goal Analysis process

1. Define Soft goals
2. Define hard goals
   - Step 1 and 2 may be alternative
3. Evaluate Hard goal contribution to soft goals



Softgoals as selection criteria



Summary

- Requirements Classifications
- Requirements Prioritization
  - Conflicts
- Requirements Negotiation
- Stakeholder Goals
  - Goal Analysis

- Next Week
  - Goal Modeling