

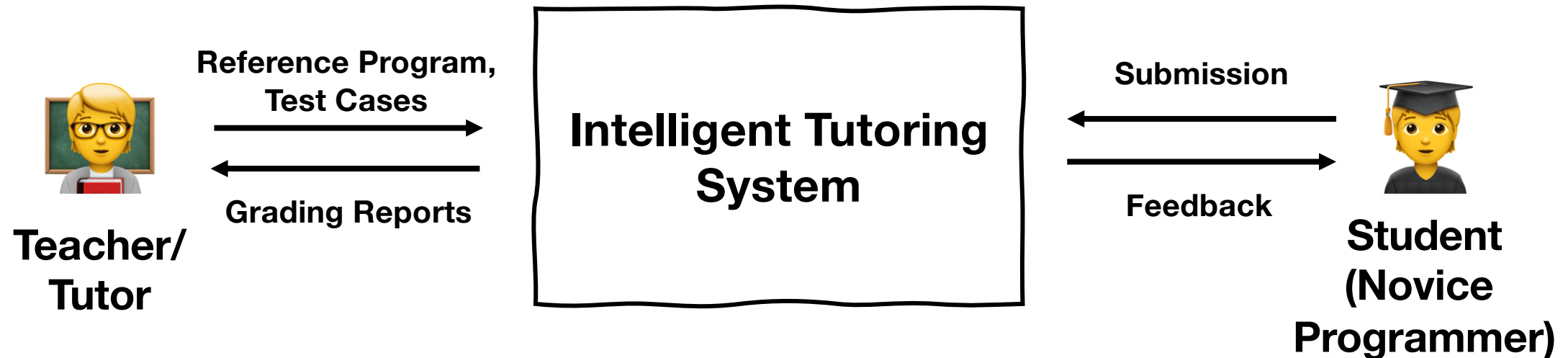
CS3213 Lab – Week 1

Introduction to Course Project | 12-01-2022

- ☐ Idea of Course Project
- ☐ Lab Logistics/Amenities
- ☐ Group Selection Process
- ☐ Assignment 1
- ☐ Requirement Elicitation in Week 2

Idea of the Course Project

- ❑ The **lecture** provides software engineering methods, theory, models, patterns, etc.
- ❑ The **lab** provides application of the software engineering principles.
- ❑ Not just any project but related to our **research** including **real customers** at NUS.



Team

Lecture



Abhik Roychoudhury

<https://www.comp.nus.edu.sg/~abhik/>

Lab and project coordination



Yannic Noller

<https://yannicnoller.github.io>

Tutors



Zhiyu Fan



Jon Chua



Guo Ai



Kishore R



Liu Yu

General Setup

- ❑ After the 2-hour lecture slot, there is a **1-hour slot** with a **general lab session** for all students.
- ❑ Additionally, there are dedicated **tutorial sessions** with tutors with a smaller set of students.
- ❑ Until week 5, there is a **weekly assignment** sheet, which needs to be submitted by each group.
- ❑ After week 6, the focus is on project implementation.

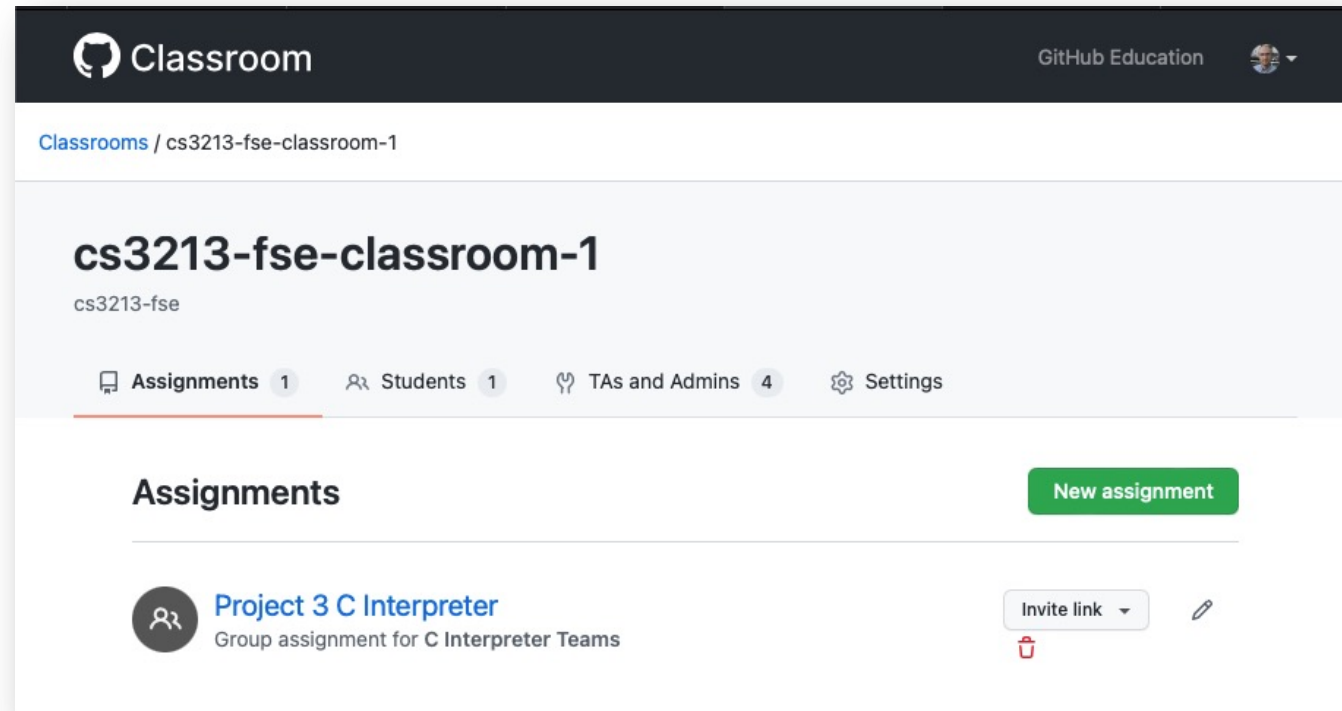
LumiNUS Submissions



- ❑ The assignment sheets and lab slides are available in LumiNUS.
- ❑ Each submission will be made by LumiNUS, except for the project code (see next slide).

Project Code Submissions

- ❑ GitHub Classroom with project templates/skeletons and integration and test pipelines.



⇒ *More details will follow in the upcoming weeks...*

Lab Assignments and Timeline

Week	Lab Session	Assignment	Assignment Due
1	Introduction	A1 – Requirements Analysis & Elicitation	Week 2
2	Requirements Elicitation with Customer	A2 – Requirements Modeling	Week 3
3	Requirements Modeling	A3 – Software Architecture	Week 4
4	ITS Architecture & Projects	A4 – Module Design	Week 5
5	Module Structure and Behavior Design	A5 – Project Planning	Week 6
6	Implementation	A6 – Intermediate Deliverable	Week 7
<i>R</i>	<i>Recess Week</i>	-	-
7	Recap – Requirements & Design	-	-
8	Static Code Analysis	-	-
9	Unit Testing	A7 – Test Case Design	Week 10
10	Fault Localization and Debugging	A8 – Presentation + Final Code	Week 13
11	Integration testing	-	-
12	Lab Closing Session	A9 – Final Report	Week RD
13	Student Presentations	-	-
<i>RD</i>	<i>Reading Week</i>	-	-
<i>E</i>	<i>Examination</i>	-	-

Group Selection

- ❑ Each assignment (except for the first one) and the project work need to be done in submission **groups**.
- ❑ Each group will have 3-4 students.
- ❑ Coordinate with your fellow students and register in LumiNUS. We provided a Google Sheet¹ for a preliminary selection of groups and projects.
- ❑ Make sure that you and your teammates are in the same tutoring session!

¹ https://docs.google.com/spreadsheets/d/15sk6WnvQHTjClhMi_TUuyDkylow6lOmMHVhD5n3Qu-l/edit?usp=sharing

Contacts and Consultation

- ❑ General and assignment specific questions should be always submitted via the **LumiNUS forum**. This way you will get the fastest answer and all students can benefit.
- ❑ For question to your tutorial group you can ask your **tutor**.
- ❑ Otherwise, feel free to contact **Yannic** with all other questions.
- ❑ Lab Consultation Times are added to the LumiNUS course.

Course Grading

Lab Grading consists of grading for:	Marks:
6 extra Assignment Sheets (A1, A2, A3, A4, A5, A7)	12
Intermediate Submission of project (A6)	6
Final Report and Presentation (A8+A9)	12
Final Code for Project (A8)	15
Total:	45

Aspects for the code submission: *correctness, completeness, maintainability*

Blue Slide



Do you have any questions so far?

Requirements

“The hardest single part of building a software system is **deciding** precisely **what** to build. No other part of the conceptual work is as difficult as establishing the detailed **technical requirements** ... No other part of the work so cripples the resulting system if done wrong. **No other part is as difficult to rectify later.**”

 **Requirements Analysis & Elicitation**

Brooks, F. P., “No silver bullet – essence and accidents of software engineering” in IEEE Computer, Vol. 20 (4), 10-19, 1987.



How to identify requirements?

What needs to be done?

- ❑ **“as-is”** analysis
- ❑ **”to-be”** analysis

- ❑ **Terminology**
 - create a dictionary for special terms?
- ❑ **Documentation:** Traceability of Requirements?

“as-is” analysis

- ❑ Identification of the existing state:
 - ❑ What is the current state?
 - ❑ Why? How did the current state emerge?
 - ❑ What is good about the current state?
 - ❑ What is bad about the current state?

What is the benefit of the “as-is” state?

It is “real”.

“as-is” analysis – challenges

- ❑ There are **implicit expectations** by the customer for the new system: everything which was acceptable or good in the current system, should remain unchanged or be improved.

Software developers often do not realize that the customer does not primarily want a **change** but an **improvement**.
→ negligence of the as-is analysis

- ❑ thankless task

You open the **door** at the **main entrance**?

Yes, this is what I told you.

Every **morning**?

Certainly.

Also at the **weekend**?

No, the entrance remains closed at the weekend.

Also at the **public holidays**?

No, obviously it remains closed.

What about if you are **sick** or on **leave**?

Then this is done by Mr. X.

And what about if Mr. X is not **available**?

Then at some point during the morning, a client will knock at the window because he or she cannot enter the store.

What means **morning**? ...

“to-be” analysis

- ❑ Collection of **new** requirements
- ❑ Problems?
 - **unrealistic** expectations by client
 - **formulation/communication** by customers is incomprehensible by software people
 - there are multiple stakeholders representing the customer, but **not** every stakeholder has the **same expectations**
 - the customer/client is **internally** not perfectly organized
 - the client has desires and goals but no precise requirements
 - **unspoken/assumed** requirements by clients



What kind of analysis techniques do you know? And what do they target?

Requirement Analysis Techniques

Analysis Technique	Main Focus		
	“As-Is” State	“To-Be” State	Innovation Impact
Analysis of existent data and documents	<div></div>		
Observation	<div></div>		
Survey with <div><div>closed</div><div>structured</div><div>open</div></div> questions	<div></div>		
	<div></div>		
	<div></div>		
Interview	<div></div>	<div></div>	
Modelling		<div></div>	
Experiments		<div></div>	
Prototyping		<div></div>	
Participative Development (wrt analysis)		<div></div>	<div></div>

Assignment 1: Requirements Analysis & Elicitation



Assignment 1: Requirements Analysis & Elicitation

CS3213 Foundations of Software Engineering (AY21/22 Sem2)

Submission Deadline: **Tue 18/01/2022, 10 pm**

Discussion: Week 2 and 3

- You must strictly comply with the noted deadline. No late submissions!
- This is an **individual** assignment. Acts of plagiarism are subjected to disciplinary action by the university. Please refer to <https://www.nus.edu.sg/celc/programmes/plagiarism.html> for details on plagiarism and its associated penalties. *Note: all future submissions will be group assignments.*
- Please use appropriate tools to create your solutions (e.g., LibreOffice/Word/LaTeX for textual submissions, or draw.io for graphical solutions). Handwritten solutions are accepted only in exceptional cases and if they are very legible.
- Please create a **PDF document** from the solution including a **title sheet** with the exercise sheet number, your name, and matriculation number.
- Please use this scheme as the file name for the PDF document: **assignment.X.YYYYYYYYY.pdf**, where X is the exercise number and YYYYYYYY is your matriculation number.
- Please submit this PDF document via LumiNUS. In case of any discrepancies regarding the submission date, the date given in LumiNUS will count.
- There are **2 marks** to be scored for this assignment sheet. The worst score for any assignment sheet is 0 marks.

Overview

The objective of this assignment (and the few following assignments) is to provide you with a feel for what a requirements elicitation process may look when facing the development of a new application or system. This particular assignment prepares you for the requirements elicitation session with project *stakeholders*.

We will kick things off by providing you with a somewhat ambiguous description of the problem, as well as with several pointers and documents to existing systems. Your first task is then to analyze the provided description and systems, and produce questions that will help you clear out some of the misunderstandings, elicit more requirements, or even discard those that may be deemed unfeasible. The following (future) assignments will involve the design and development of several requirement specification documents.

Objectives:

- ☐ provide you with a **feel** for what a **requirements elicitation** process may look when facing the development of a new application or system
- ☐ **prepare** you for the requirements elicitation session with project stakeholders

Task 1: Analyze scenario and prepare brief report about insights

Scenario: "Bob the Tutor"

Bob is a tutor in the programming course for the first-year computer science (CS) students at his university. He had this role in earlier instances of this course, so he knows the challenges ahead of him. It can become quite stressful, also given that this year the number of incoming CS students has increased significantly. As a tutor, Bob does not only have to provide feedback and help the CS students. He also needs to test and grade their submissions. He is unsure how he will cope with so many students and their feedback requests for their programming assignments.

On top of that, Prof. Hopper will be the new lecturer this year for the programming course, and usually, a new lecturer means additions or changes, which cause extra work for Bob. She already informed Bob that this year, the CS students should be able to submit solutions not only in Python but also in the programming languages C and Java. Furthermore, the assignments should no longer be submitted in the form of e-mails to Bob, but via an online submission system. In this way, everybody in the tutor team can keep track of the submissions. Last year, Bob had some issues with his e-mails because some students tried to hijack his account, which resulted in a delicate situation, where Bob was no longer able to check his e-mails, and submissions got lost. Prof. Hopper would like to avoid that in the future. Additionally, she wants to have a weekly PDF report by Bob about the progress in submissions and his grading. Bob will probably start a spreadsheet that contains all this information.

Long story short, Bob now has to prepare additional reference solutions in C and Java, which is not easy for him because he is only an expert in Python. He needs to get familiar with these programming languages to be aware of the typical pitfalls in C and Java to be able to provide meaningful feedback for the students. Usually, Bob takes the submissions, compiles them, and manually runs some test cases. Based on the results, he graded the submissions. He always wanted to automate the build and test process but never found time to figure it out. There are certainly multiple ways to identify the root causes in the programs and also in how to repair the program. But mostly he had no time and just sent the reference solution to the CS students when they had questions for some tutorial exercises. Only on rare occasions, Bob found the time to provide feedback about the fault locations, what the error messages mean, or where to potentially fix the problem. In the past, Bob also offered interactive coding sessions, where he acted as a human debugger who directly gives hints to students as they type their programs. However, this was very time-consuming, and he probably can no longer provide this service to the students. Sometimes, when Bob enjoys his last hot Milo in the late evening, he dreams of a fully automated software system that would help him to cope with all his tutoring tasks. Bob is a senior computer science student, so maybe at some point in the future, he can implement such a system himself...

- ❑ **problem** description
- ❑ **"as-is"**-state
- ❑ **"to-be"**-state
- ❑ the identification **key aspects** for the stakeholders

Task 2: Question Preparation

- ❑ You need to **discover** the requirements in the **requirement elicitation** with the **stakeholders**!
- ❑ Use scenario and existing documents as baseline.
- ❑ Are aspects unclear? Does it make sense?
- ❑ Prepare **at least 10 questions**.
- ❑ **Everyone** is expected to **ask questions** and **interact** with the customer in a moderated fashion.

How to find good questions?

❑ Which topics need to be covered in the requirement specification?

- Purpose of the software
- Functional Requirements
- Requirements to External Interface
- Requirements Regarding Technical Data
- General Constraints and Requirements
- Product Quality Requirements

1. Introduction
1.1 Purpose
1.2 Scope
1.3 Product overview
1.3.1 Product perspective
1.3.2 Product functions
1.3.3 User characteristics
1.3.4 Limitations
1.4 Definitions
2. References
3. Requirements
3.1 Functions
3.2 Performance requirements
3.3 Usability requirements
3.4 Interface requirements
3.5 Logical database requirements
3.6 Design constraints
3.7 Software system attributes
3.8 Supporting information
4. Verification
(parallel to subsections in Section 3)
5. Appendices
5.1 Assumptions and dependencies
5.2 Acronyms and abbreviations

SRS outline (IEEE 29148:2018)

ISO/IEC/IEEE 29148:2018, ISO/IEC/IEEE International Standard - Systems and software engineering -- Life cycle processes -- Requirements engineering, 2018, DOI: [10.1109/IEEESTD.2018.8559686](https://doi.org/10.1109/IEEESTD.2018.8559686) → page 67 to 74

Outlook for Assignment 2

- ❑ ... after the requirement elicitation session you will need to model the requirements, e.g., with:
 - ❑ **Use Cases**, or
 - ❑ Behavioural Models like **Activity** Diagrams or **State Transition** Diagrams
- ❑ one other way of requirements modelling is called **Goal Oriented Requirement Engineering (GORE)** Modeling

Goal Oriented Requirement Engineering (GORE) Modeling

- ❑ Goal-oriented analysis focuses on the description and evaluation of alternatives and their relationship to the organizational objectives.
- ❑ Goals are introduced for pragmatic or engineering reasons – they help accomplish the objectives of several specific subtasks of RE
- ❑ Definition goal: A goal is a desirable state lying in the future, which is not reached automatically but by specific actions.
 - ❑ a Goal should be proactive.
- ❑ Goals and their dependencies are often described in conceptual models that are based on modeling languages.
- ❑ Definition goal model: “A goal model is a conceptual model. Its goals and decompositions are documented in sub-goals and as necessary further dependencies between (sub)-goals.”

Representations of Goals

- ❑ **And / OR – Trees**
- ❑ **SIG (Softgoal interdependency graphs)**
- ❑ i-star (i*)
- ❑ GRL (Goal-oriented Requirement Language)

Example

❑ Meeting Scheduler

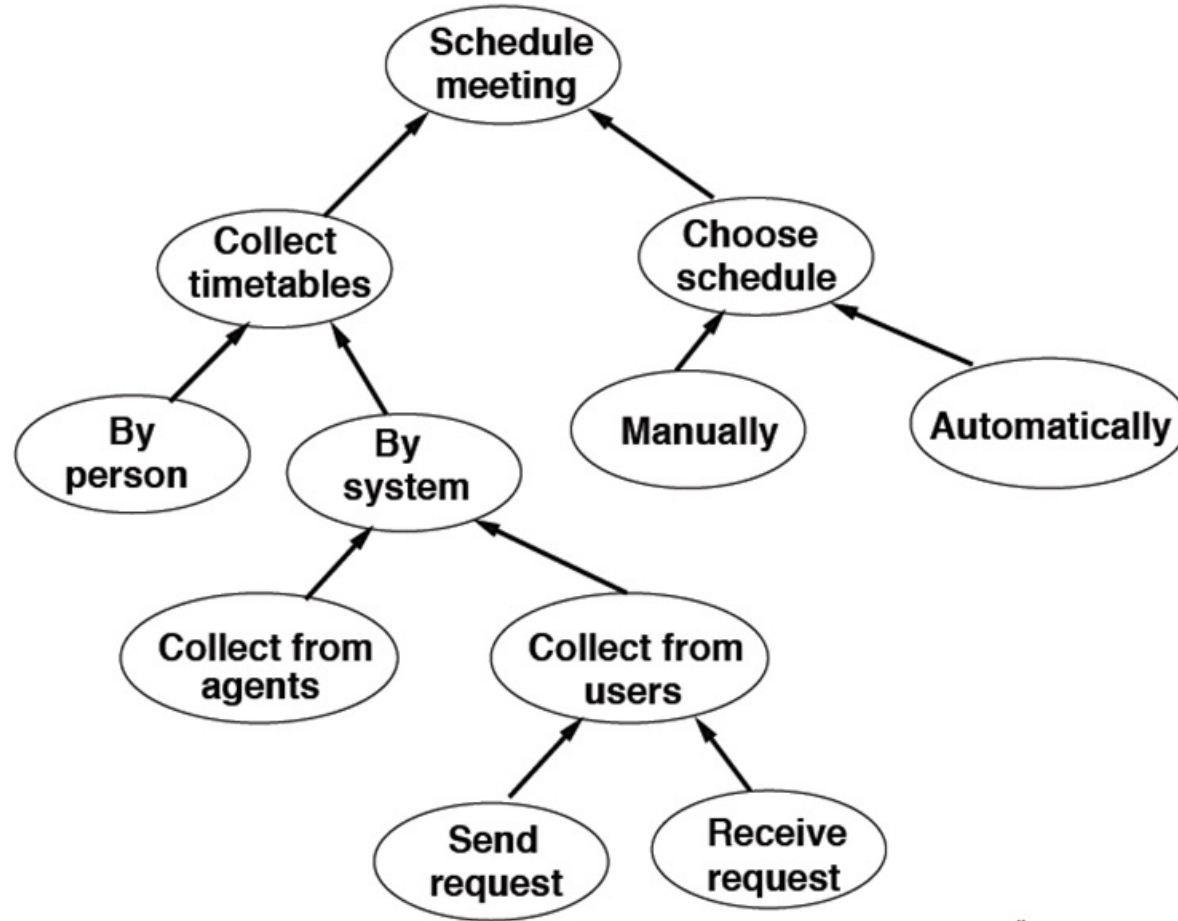
- ❑ Assists the initiator in scheduling a meeting
- ❑ Meeting should be convenient for participants
- ❑ Participants should be available

❑ Goal Modeling with different notations

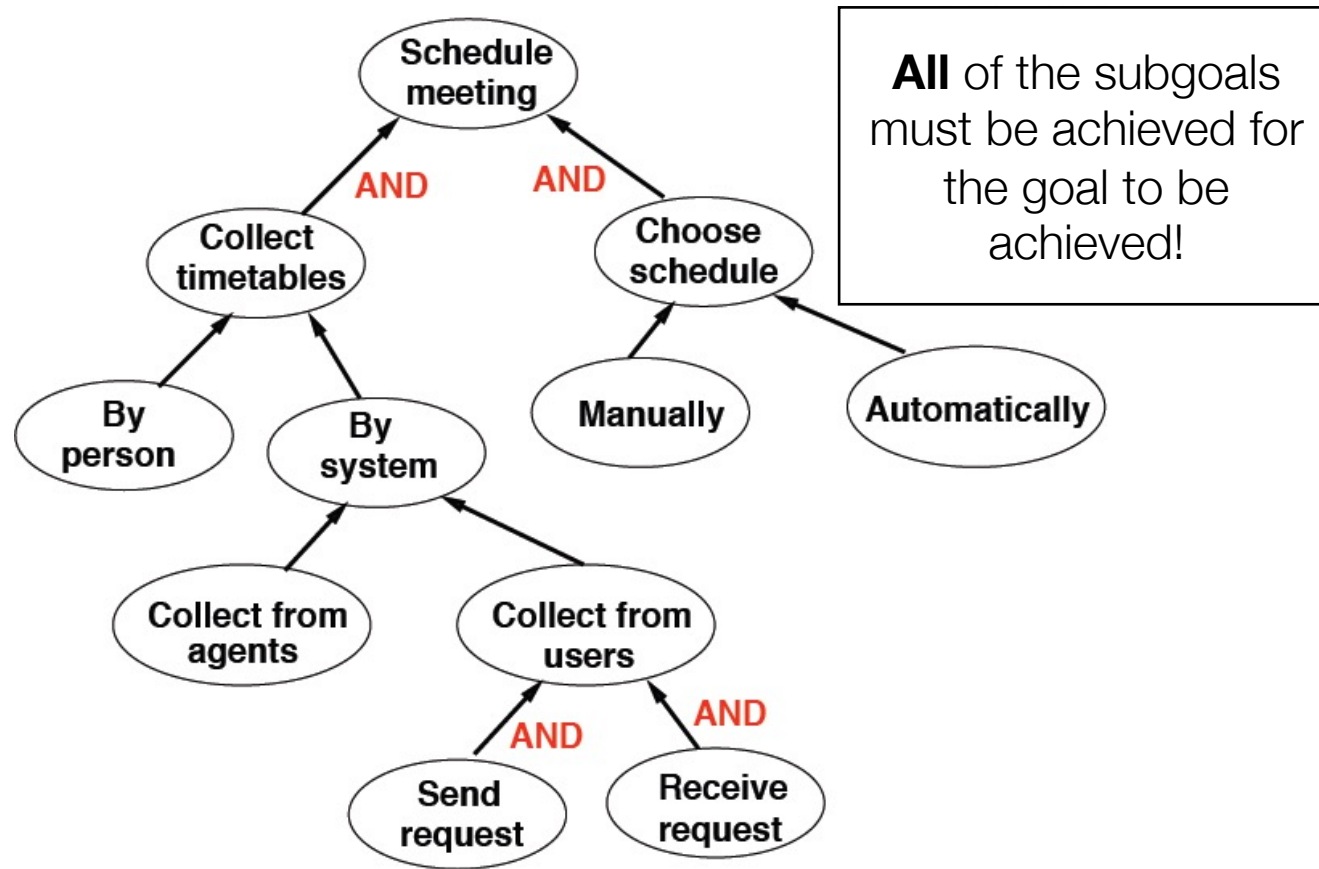


Based on slides by Betty H.C. Cheng, Professor, Computer Science and Engineering, Michigan State University

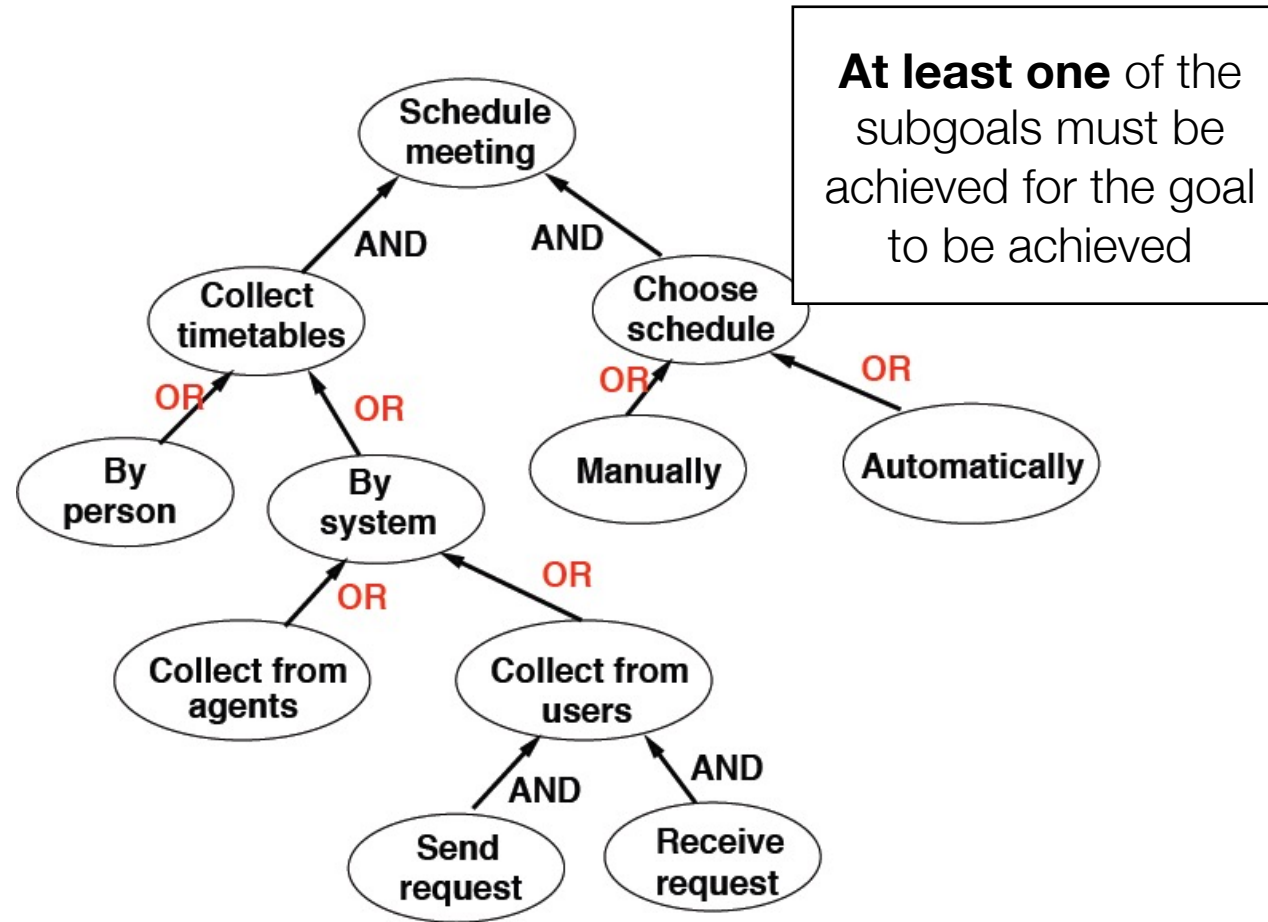
Goal Model



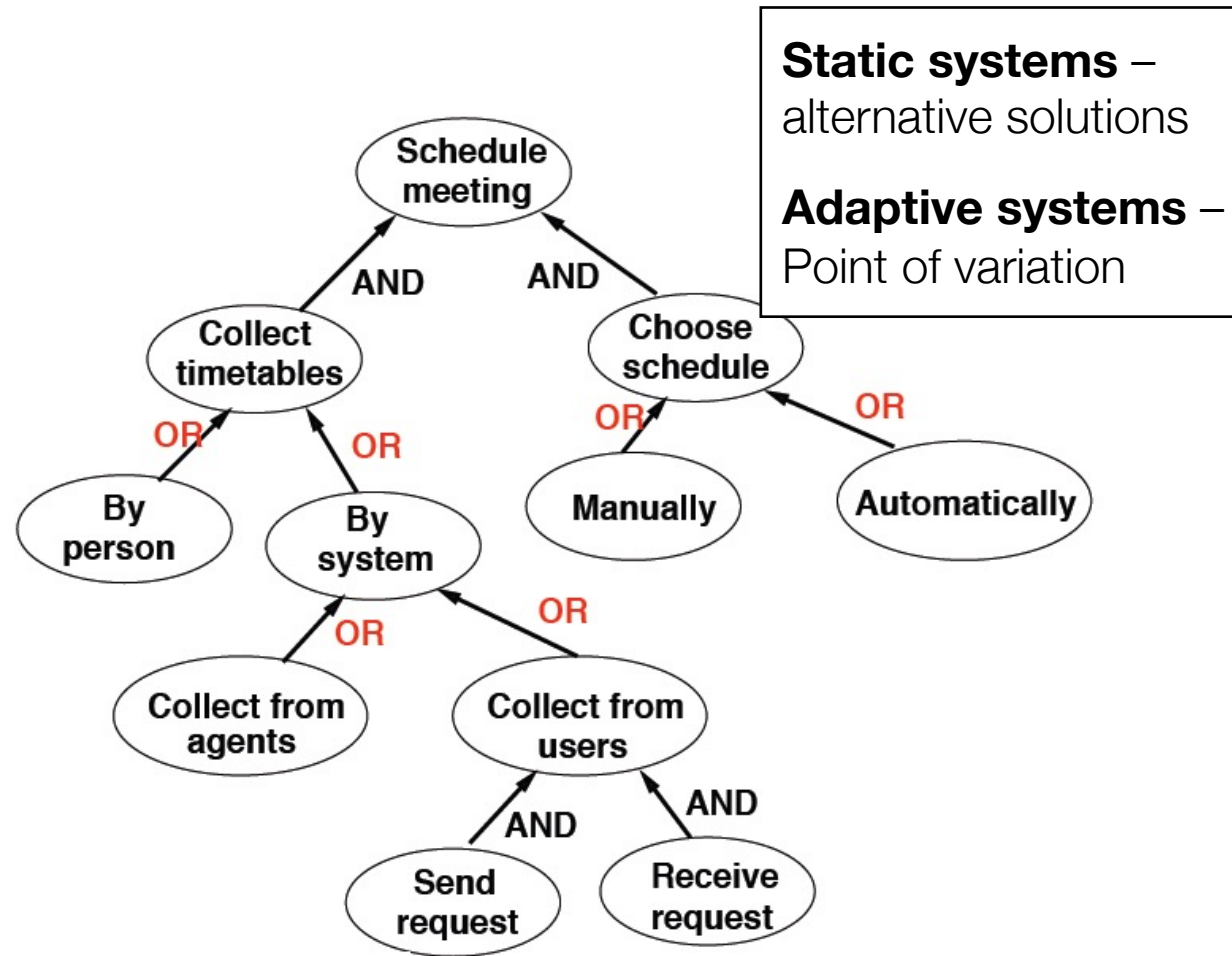
AND-Refinement Tree



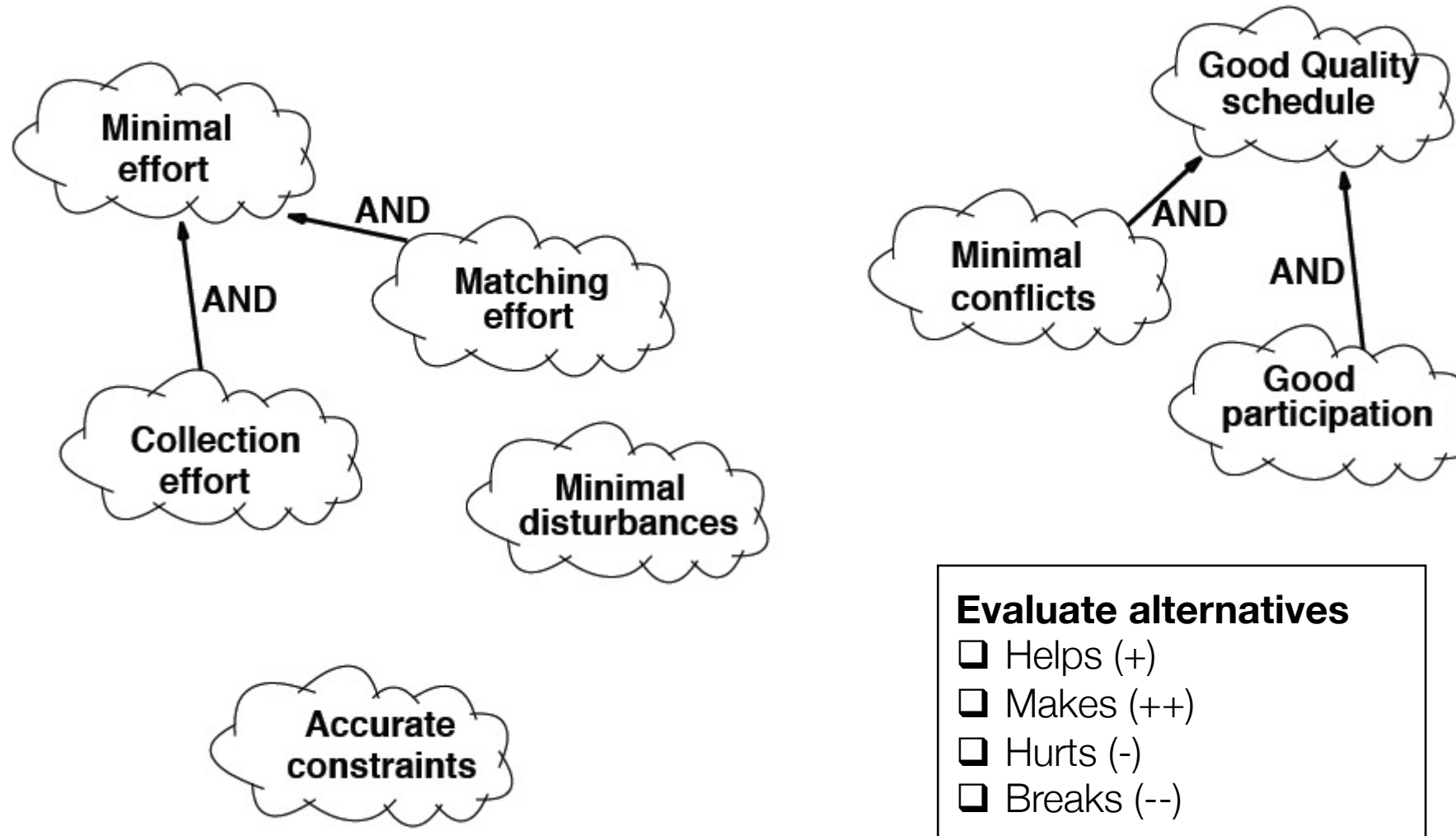
OR-Refinement Tree



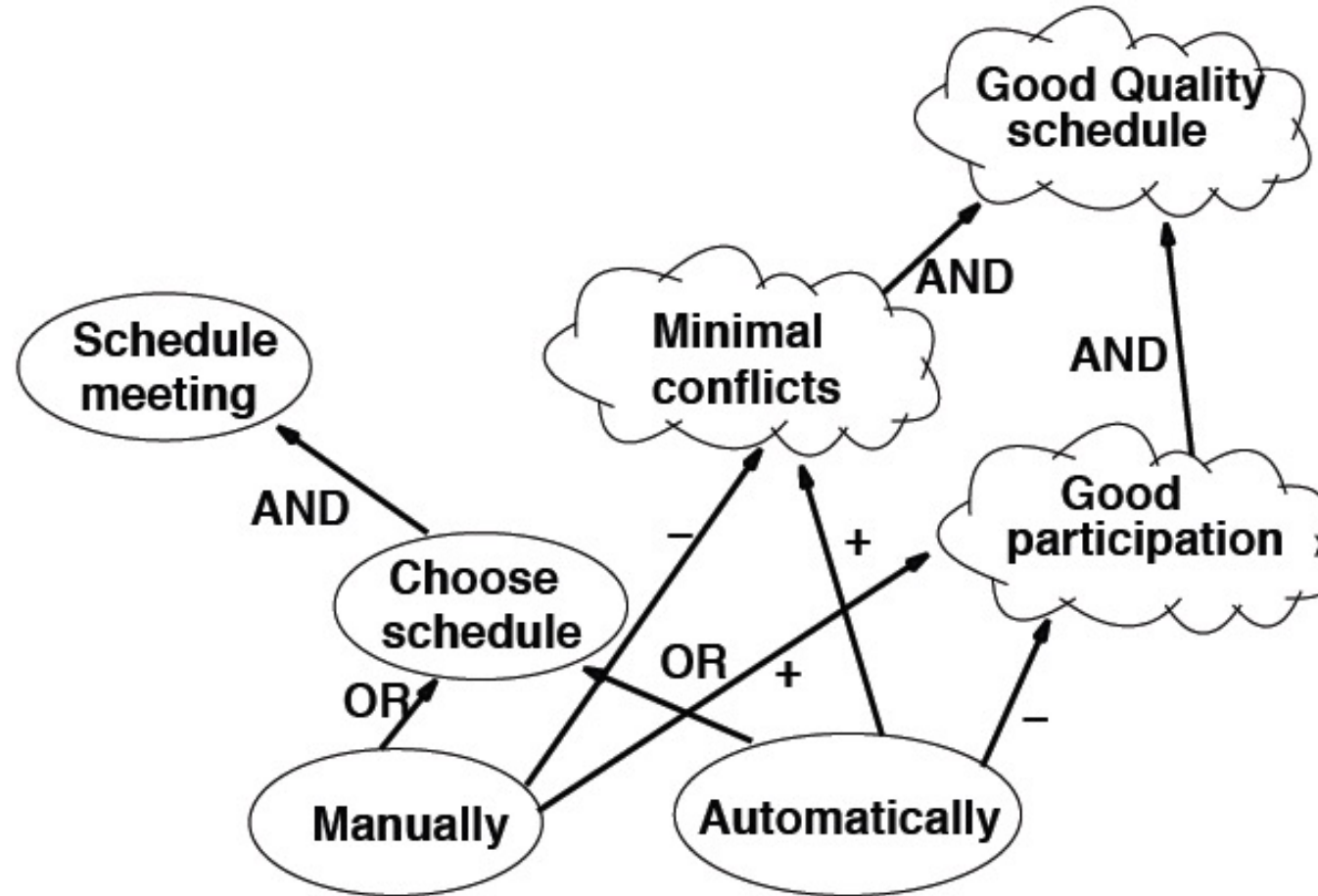
Interpretations of OR Refinements



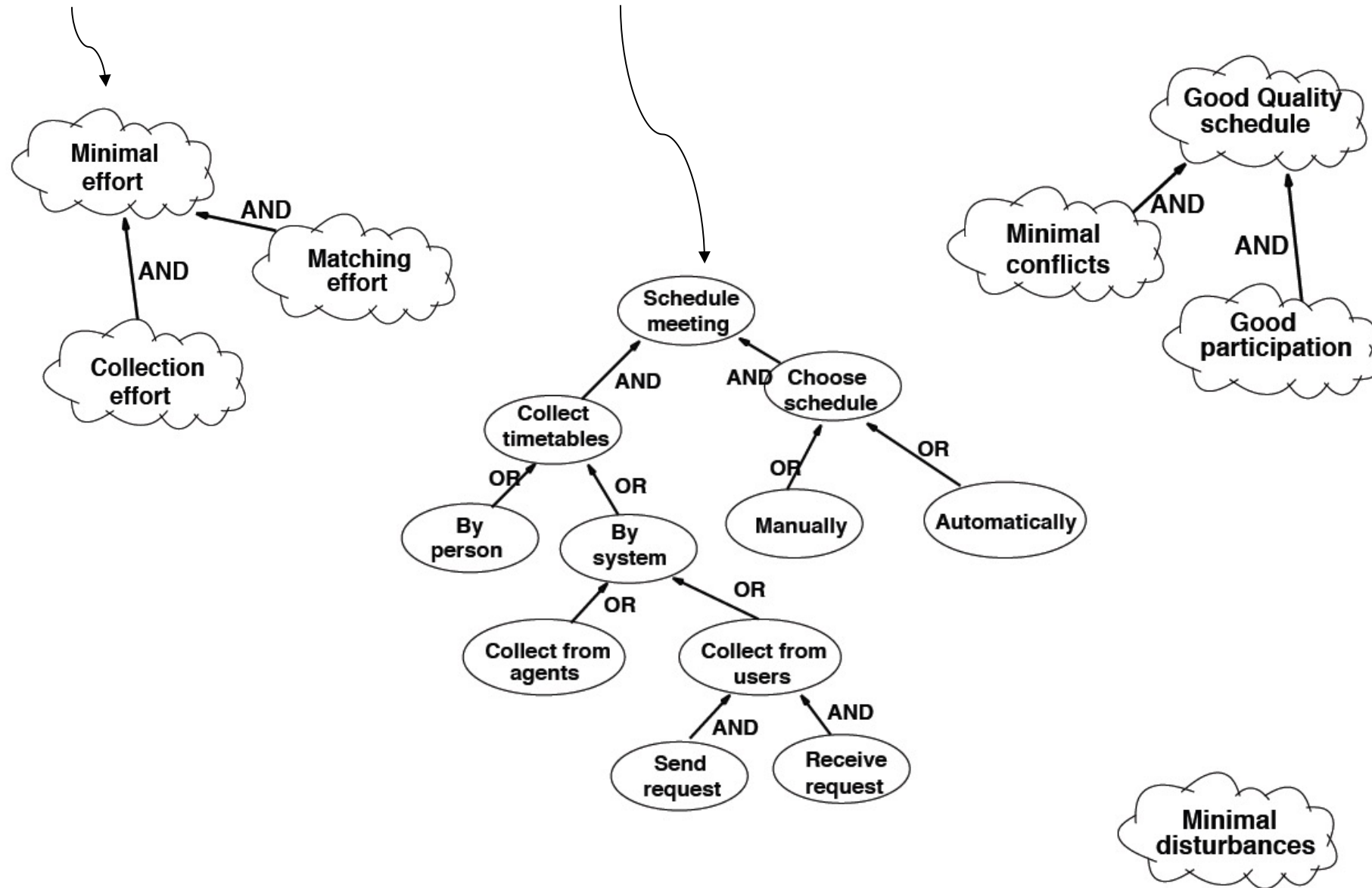
Soft Goal Graphs



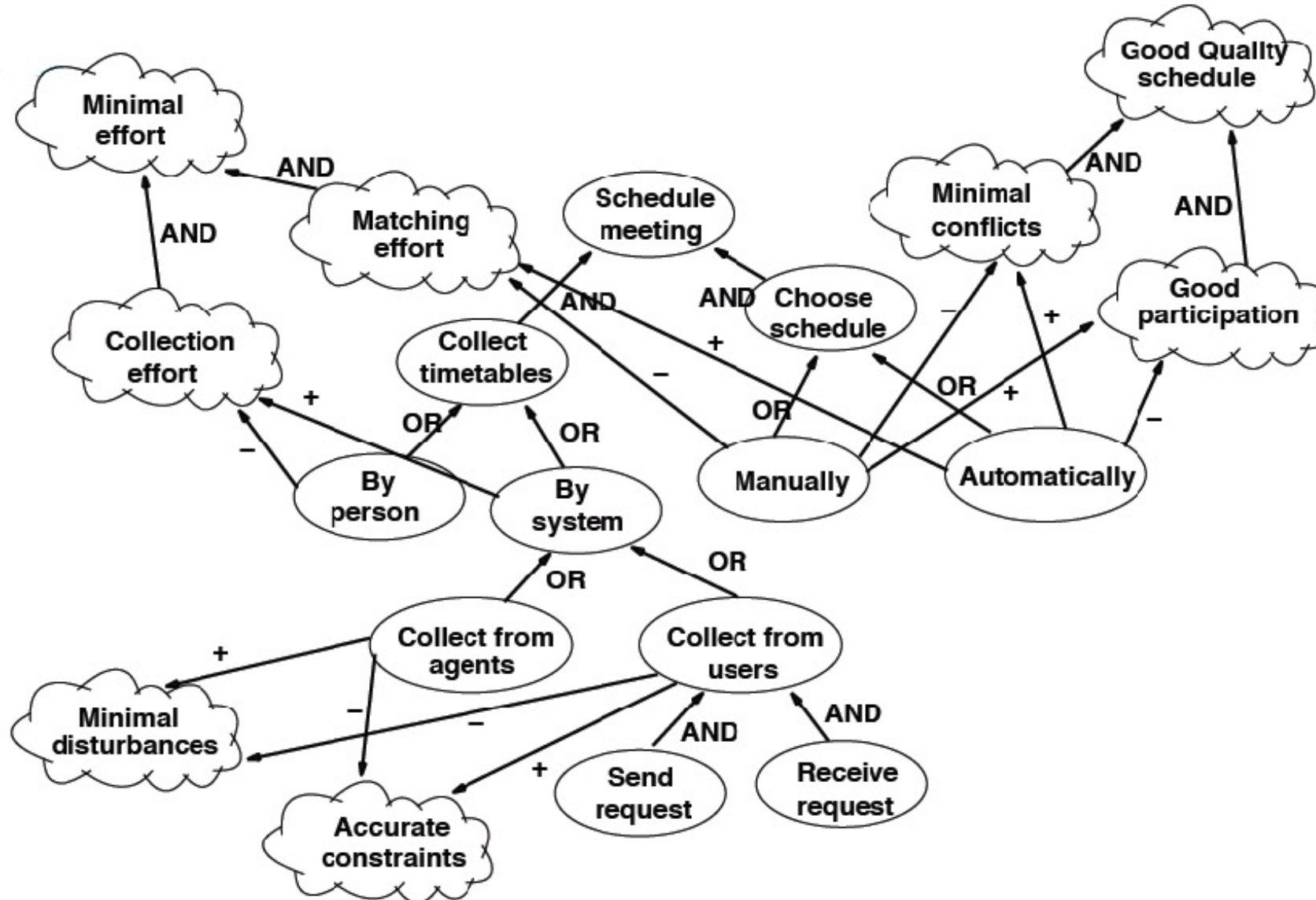
Contributions to Softgoals



Soft Goals & Hard Goals



How to combine



Conclusion

- ❑ **Lab** is the practical extension of the lecture. Key aspect will be the **course project**.

Next Week: Requirement Elicitation Session

- Focus on Assignment 1 and **prepare** for the session!
- **Everyone** is expected to **ask questions** and **interact** with the customer.
- Keep in mind that you will have to model the requirements.