

CP317 Software Engineering

week 2-2 Requirement gathering – part -2

Shaun Gao, Ph.D., P.Eng.

Agenda

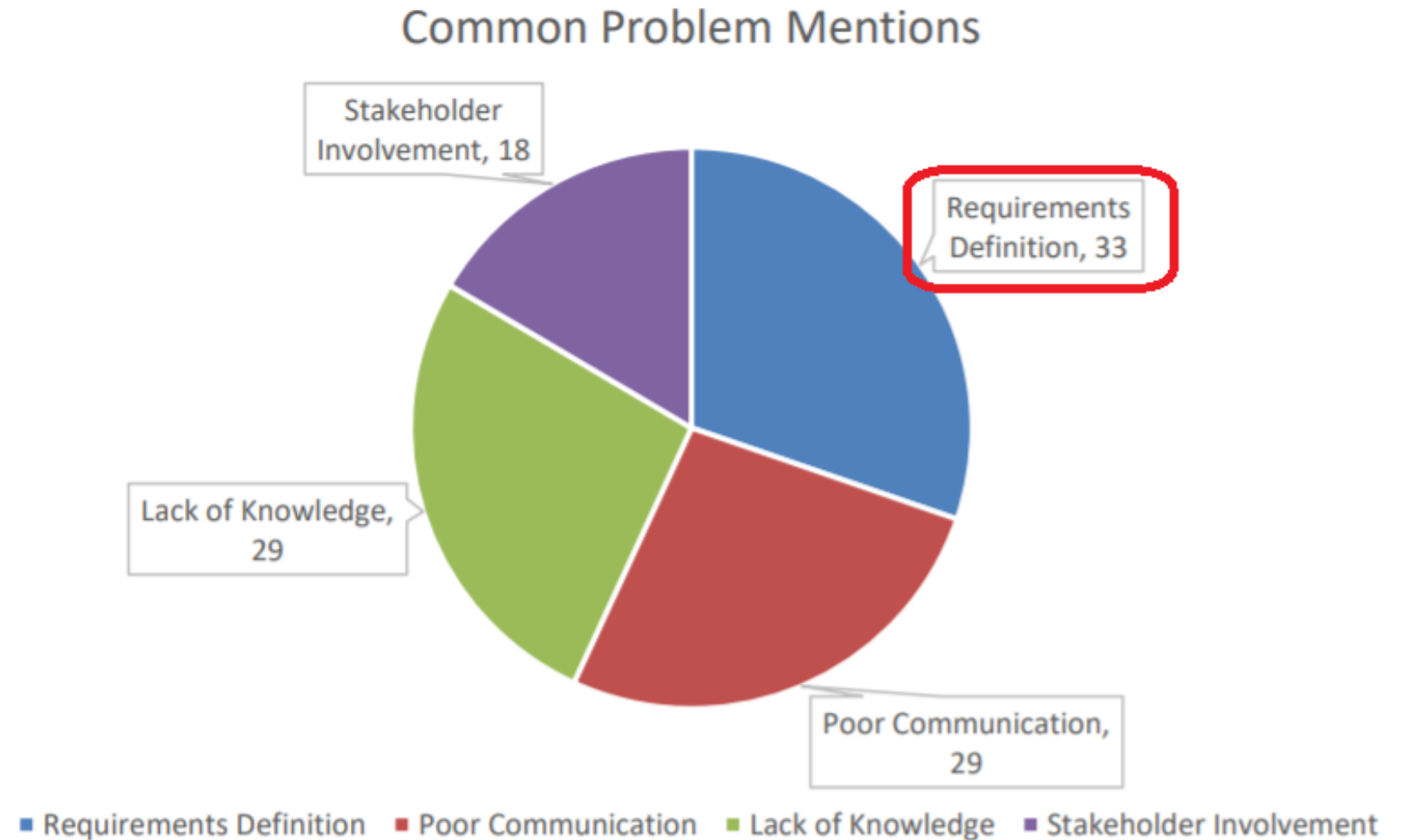
- Review week 2-1
- Gathering requirements
- Requirement elicitation
 - Techniques
 - Task analysis
 - Domain analysis
 - Brainstorm
 - Alex F. Osborn four rules about brainstorm
- Requirement specification
- Recording requirements
 - Documentation
 - Unified modeling language (UML)
- Requirements validation and verification
- Summary

Review week 2-1

- Requirements
- The reasons why requirements are important
- The characteristics of good requirements
 - Understandable, correct, unambiguous, complete, consistent, interoperable, verifiable, traceable, prioritized, achievable
- MOSCOW method
- Requirements categories
 - Business requirements, user requirements, system requirements
- FURPS and FURPS+ methods
 - Functional requirements, Non-functional requirements
 - Differences between functional requirements and non-functional requirements

2020 Findings

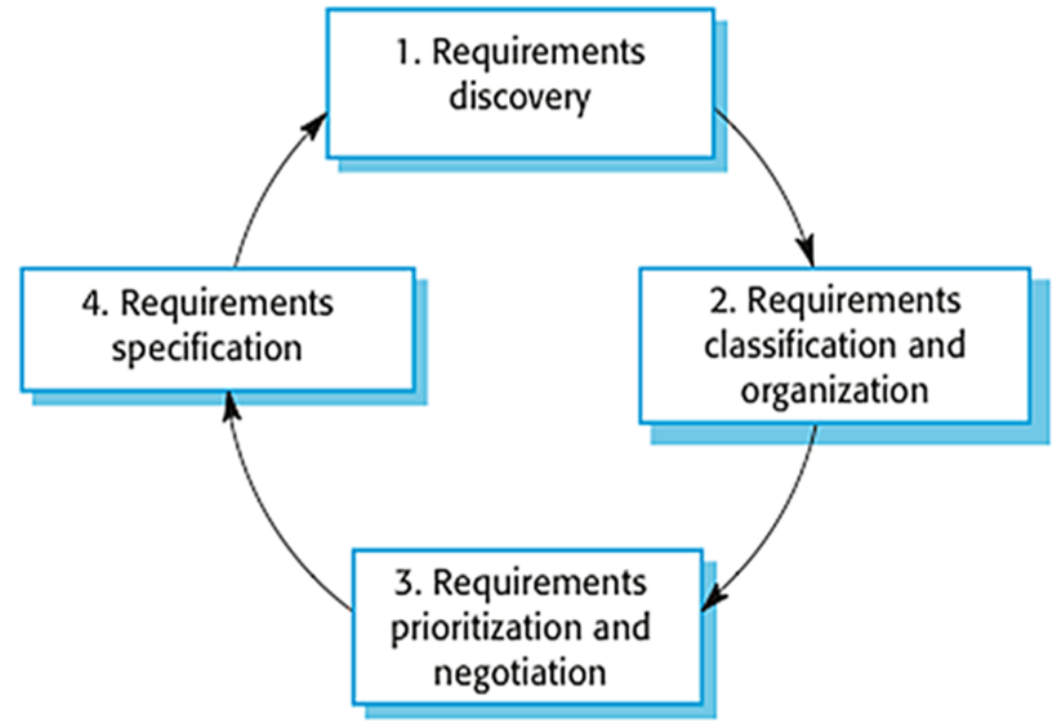
- Failure causes of IT projects



- Cite from
Capella University

Gathering requirements

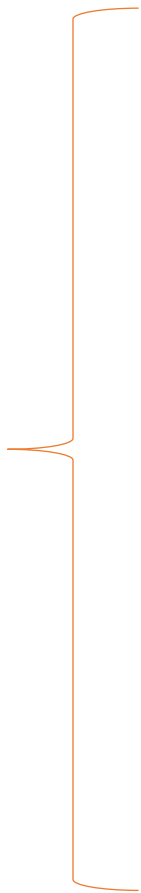
- Requirements elicitation process
 - Discovery
 - Classification and organization
 - Prioritization and negotiation
 - Specification
- Question?
 - Requirements are there, why do we need requirement discovery?
 - **Think-Pair-Share**



Requirements elicitation process

- Requirement discovery
 - Requirement discovery is the process of working with customers to understand their needs, blueprint the solution, gather detailed requirements, design technical aspects.
- Classification and organization/Categorization
 - FURPS/FURPS+
- Prioritization and negotiation
 - MoSCOW
- Specification
 - Requirements specification is a document that clearly and precisely describes each of the essential requirements (functions, interfaces, design constraint, and quality attribute) of the software and external interfaces.

Requirements elicitation techniques



NO.	Techniques	Details
1	Communications	It includes interview, surveys and questionnaires
2	Task analysis	Team of engineers and developers may analyze the operation for which the new system is required.
3	Domain analysis	It is the process by which a software engineer learns background information. Embedded, middleware, etc.
4	Brainstorming	A discussion meeting is held among various stakeholders.

Requirements elicitation techniques – cont.

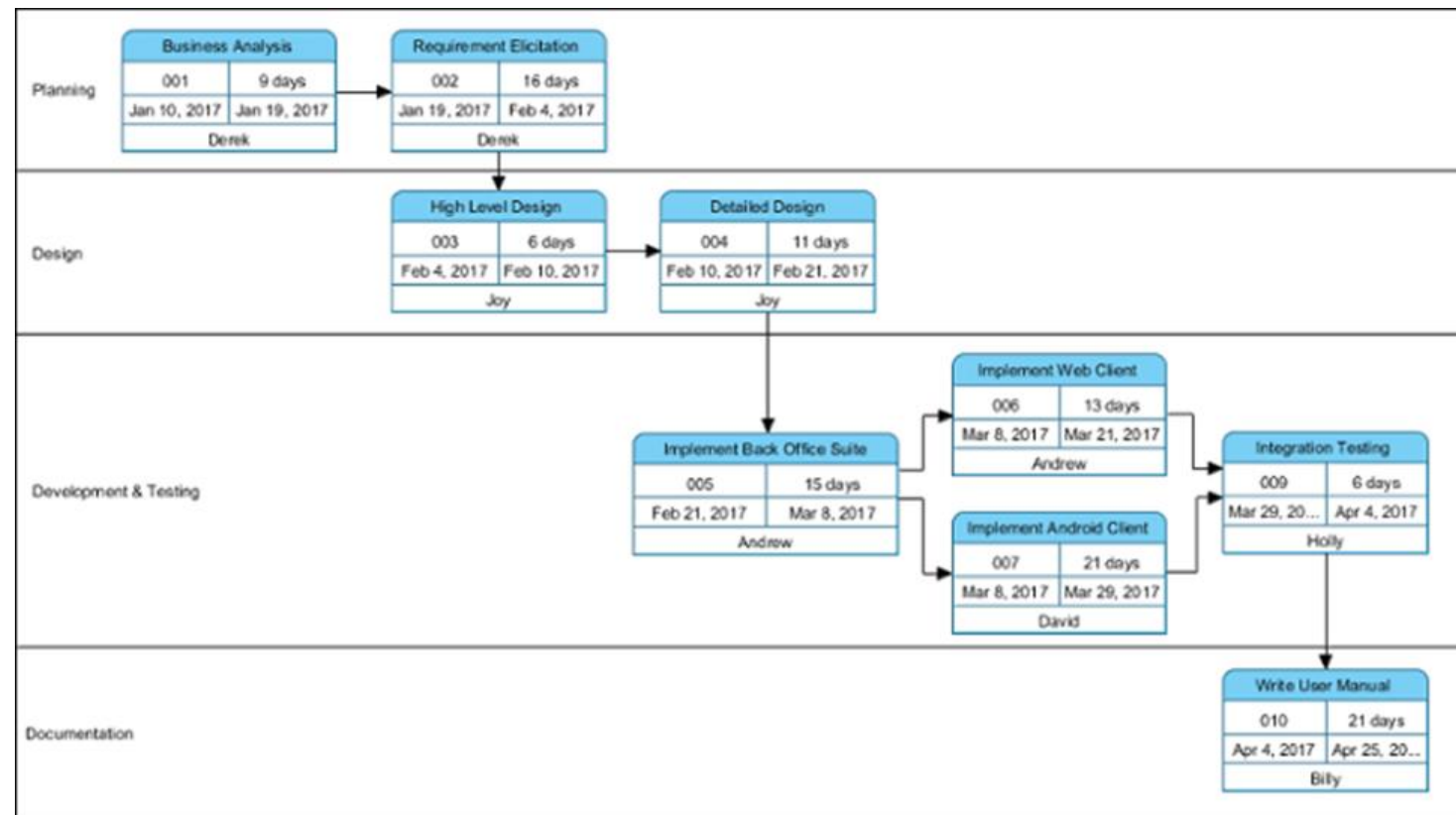
NO.	Techniques	Details
5	Prototyping	It is building a prototype product (for example, user interface) without adding detail functionality to interpret the features of intended software product.
6	Observation	Team of experts visit the client's organizations or workplace. They observe the actual working of the existing systems.

Requirements elicitation techniques – cont.

- Task analysis

- Task analysis is the process of learning how a task is accomplished, including a detailed description of activities of the task, conditions, necessary equipment, and any other factors.
- "task" is often used interchangeably with activity or process.

PERT chart



Requirements elicitation techniques – cont.

- Task analysis questions

Who is going to use the system?

What tasks do they now perform?

What tasks are desired?

How are the tasks learned?

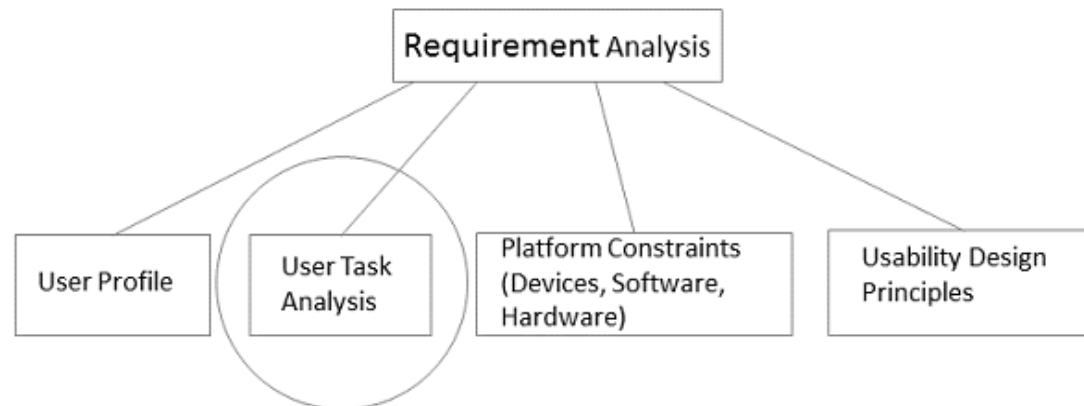
Where are the tasks performed?

What's the relationship between user & data?

How often are the tasks performed?

What are the time constraints on the tasks?

What happens when things go wrong?



Requirements elicitation techniques – cont.

- **Domain analysis**

- It is a process by which a software engineer learns back ground information.
- “Domain” means the general field of business or technology in which the customers expect to use the software product.
 - For example, airline reservation systems, medical diagnosis systems, etc.

- **Purpose:**

- Gain sufficient information, understand the system/problems, and make good decisions

- **Benefits of domain analysis:**

- Shorten the duration of the project
- Increase the quality of software products
- Anticipation of extensions

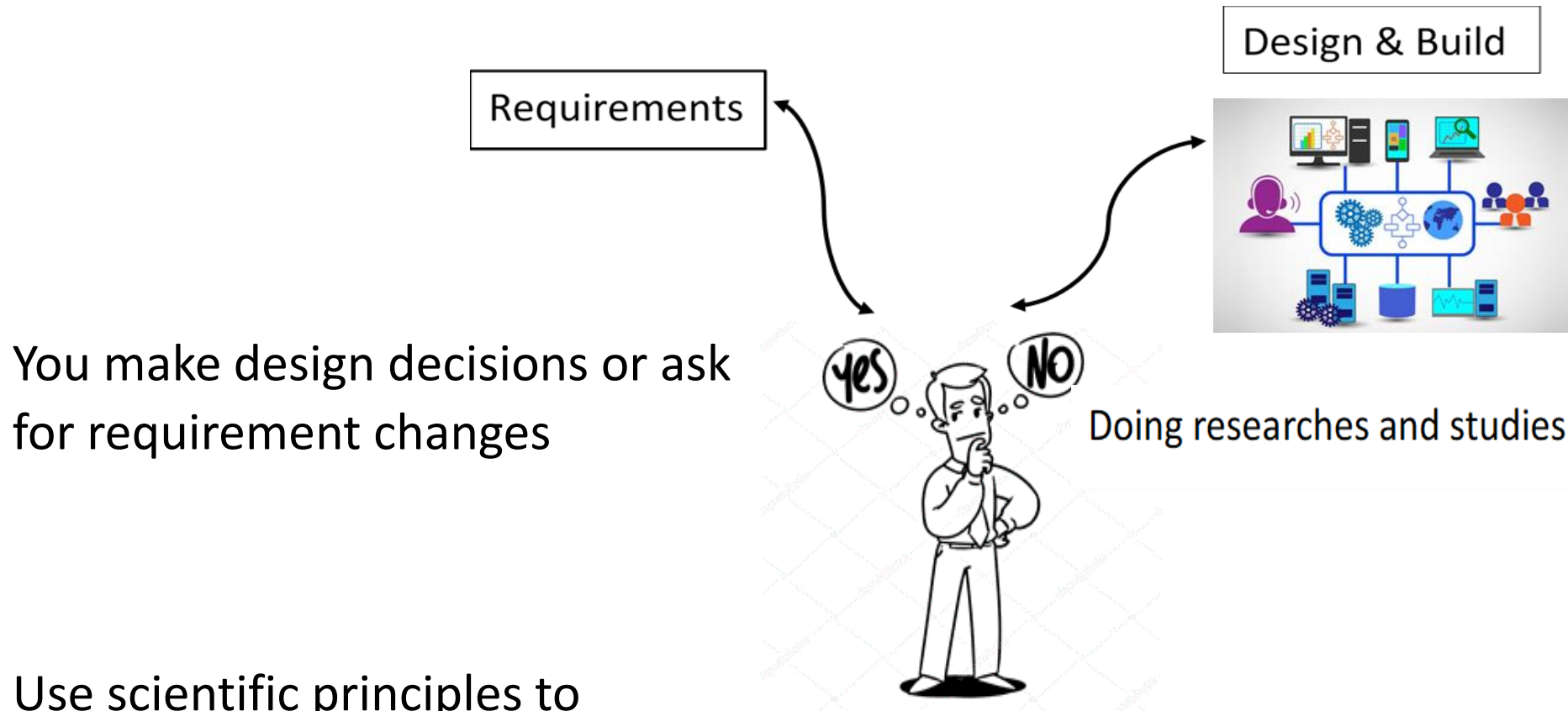
Requirements elicitation techniques – cont.

- **Brainstorm:**

- Brainstorm is a group creativity technique by which efforts are made to find a conclusion for a specific problem by gathering a list of ideas spontaneously contributed by its members.
- **Four rules of Osborn's method in Brainstorm**
- Alex F. Osborn developed methods for creative problem solving in 1939.

NO.	Four rules of Osborn's method	Meanings
1	Focus on quantity	Do everything you can to keep the idea flowing
2	Withhold criticism	Criticism can make people stop contributing
3	Encourage unusual ideas	You can always “turn down a wild idea” but you may need to think way outside of the box to find creative solutions.
4	Combine and improve ideas	Form new ideas by combining other ideas.

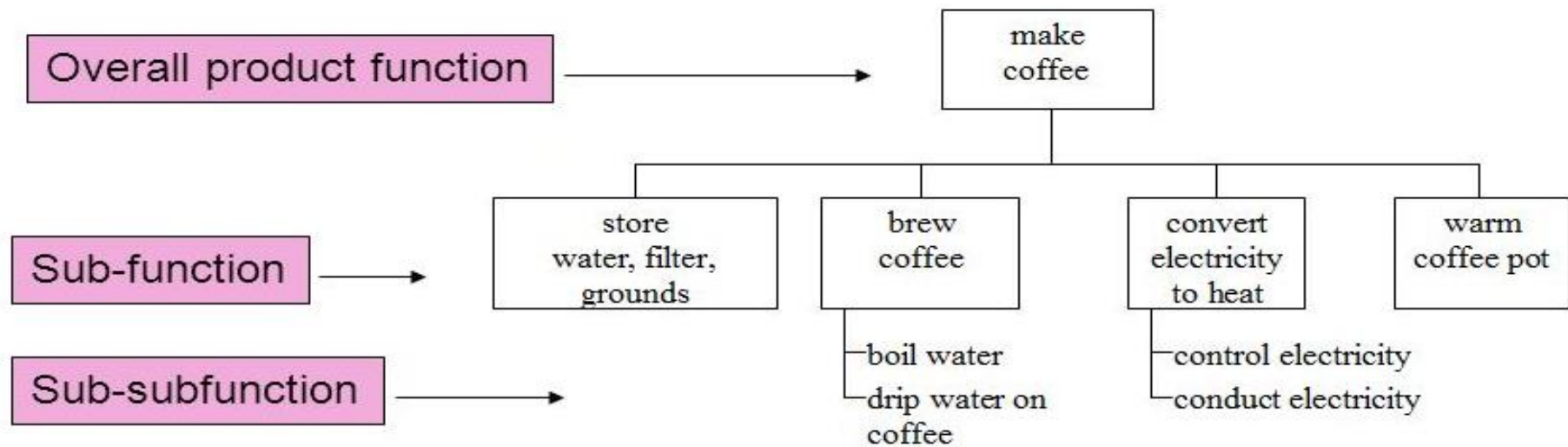
Requirement specification



Use scientific principles to
design and build high quality
software products

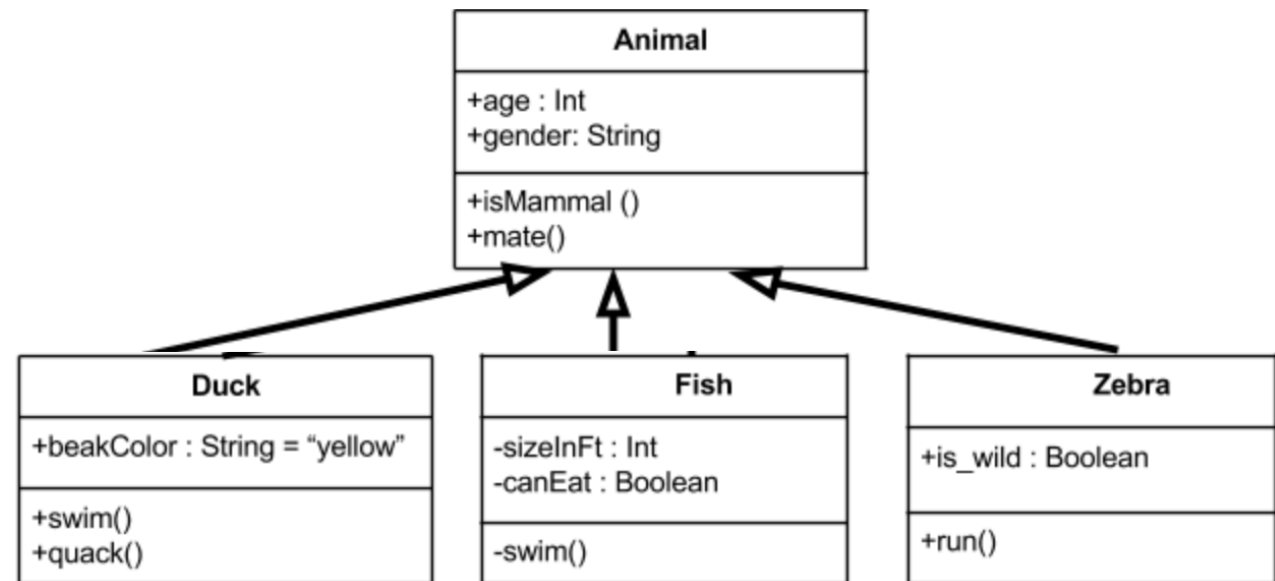
Requirement specification – cont.

- Software requirement decomposition
 - Definition: software requirement decomposition is to break down a software system from the system context level to system functions and data entities levels.
 - An example:



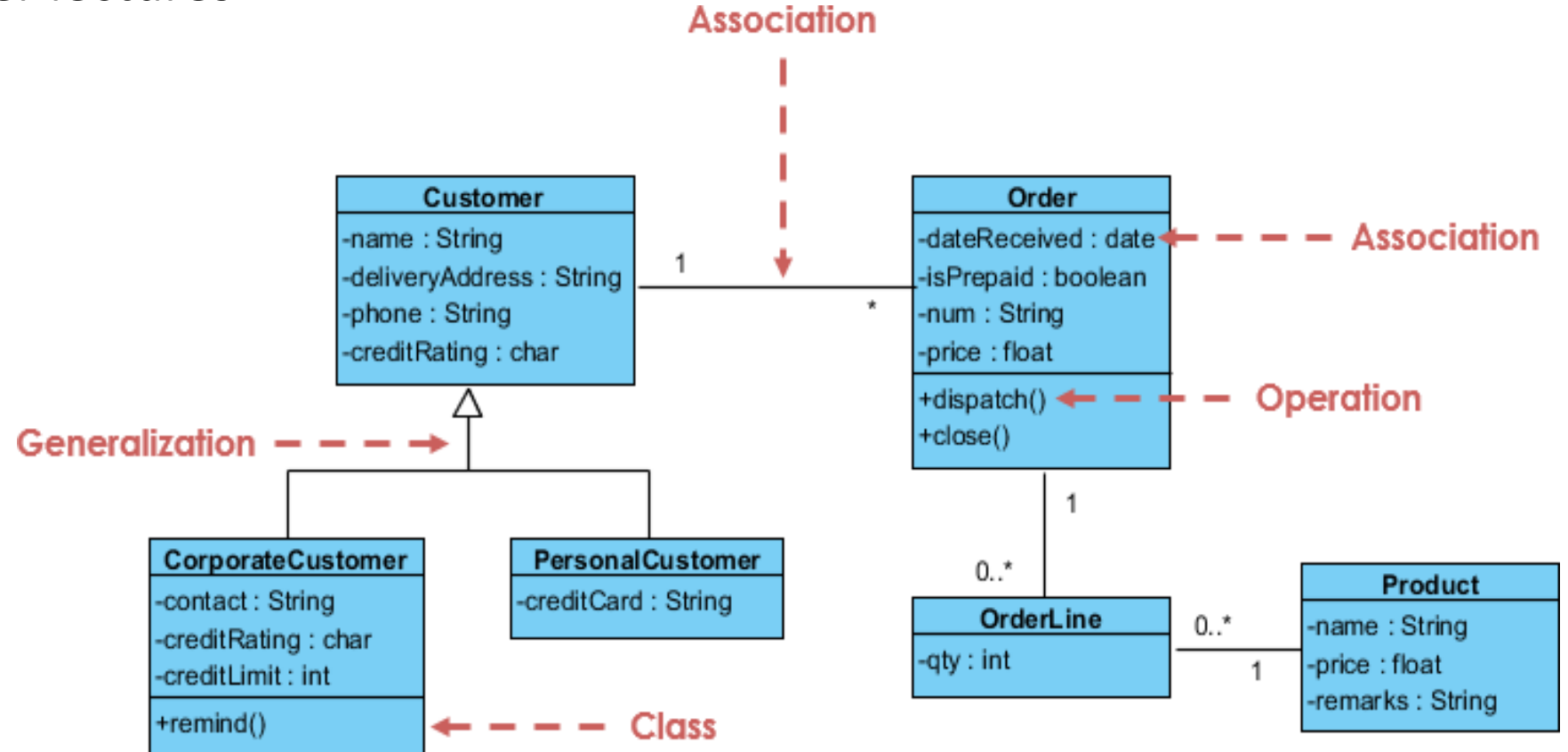
Recording requirements

- Unified modeling language (UML): <https://www.uml.org/>
- UML is a general-purpose notational language for specifying and visualizing complex software, especially large, object-oriented project.
- **UML is not a programming language;** it is rather a visual language. Engineers use UML diagrams to portray the behavior and structure of a system.
- UML is applicable for any types of software design
 - Database design



Recording requirements – cont.

- Unified modeling language (UML)
 - more in later lectures



Validation and verification

- Requirement validation
 - Requirement validation is the process of making sure that **the requirements say the right things.**
- Requirement verification
 - Requirement verification is the process of checking that **the software product actually satisfies the requirements.**

Validation and verification

- Verification vs. validation in software engineering

Verification	Validation
Are you building it right?	Are you building the right thing?
Ensure that the software system meets all the functionality.	Ensure that functionalities meet the intended behavior.
Have static activities as it includes the reviews, walkthroughs, and inspections to verify that software is correct or not.	Have dynamic activities as it includes executing the software against the requirements.

Summary

- Gathering requirements
- Requirement elicitation
 - Six techniques: communication, task analysis, domain analysis, brainstorm, prototype, observation
 - Alex F. Osborn four rules about brainstorm
- Requirement specification
- Recording requirements
 - Documentation
 - Unified modeling language (UML)
- Requirements validation and verification
 - Verification vs. validation
- Please apply what you learned to the group project.

Announcement

- It is your responsibility to find a group by the end of September 2024.
- If you cannot find one, I am willing to help.