HUST

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

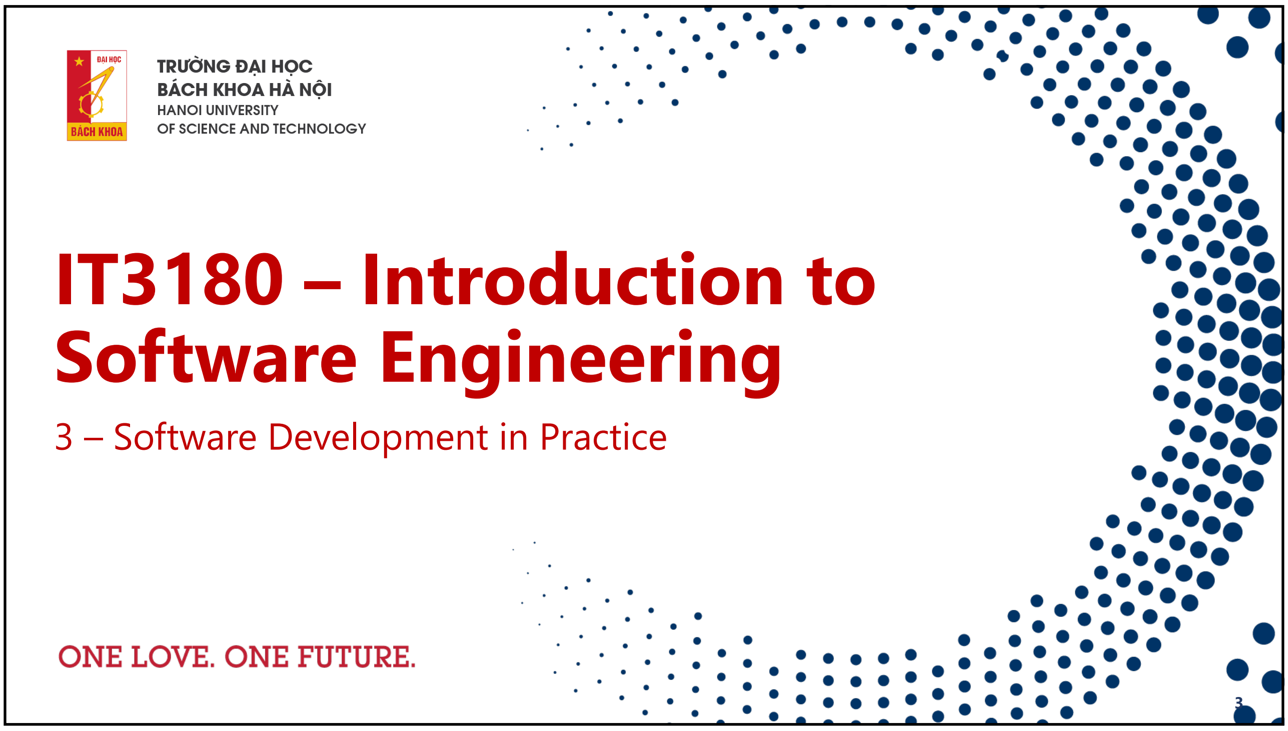
1

# SOICT

School of Information and Communication Technology

ONE LOVE. ONE FUTURE.

2



TRƯỜNG ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

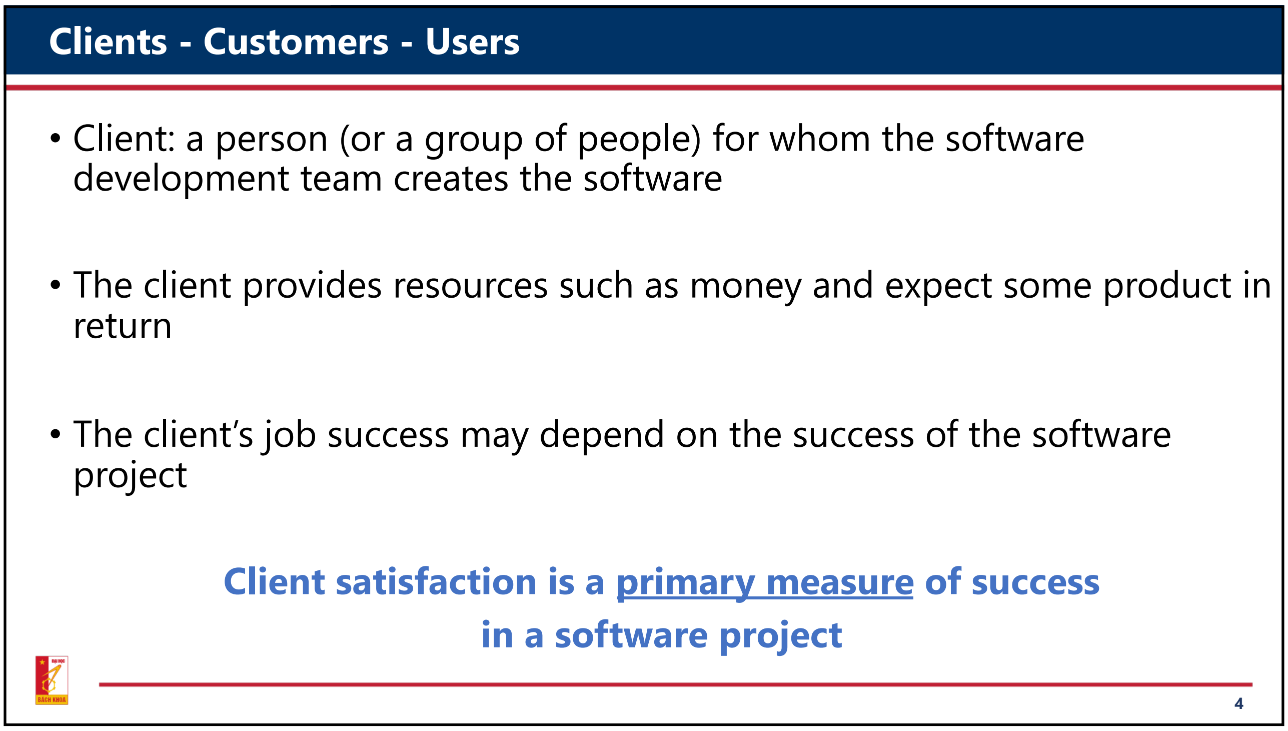
# IT3180 – Introduction to Software Engineering

3 – Software Development in Practice

ONE LOVE. ONE FUTURE.

3

3



## Clients - Customers - Users

- Client: a person (or a group of people) for whom the software development team creates the software
- The client provides resources such as money and expect some product in return
- The client's job success may depend on the success of the software project

**Client satisfaction is a <u>primary measure</u> of success in a software project**

4

4

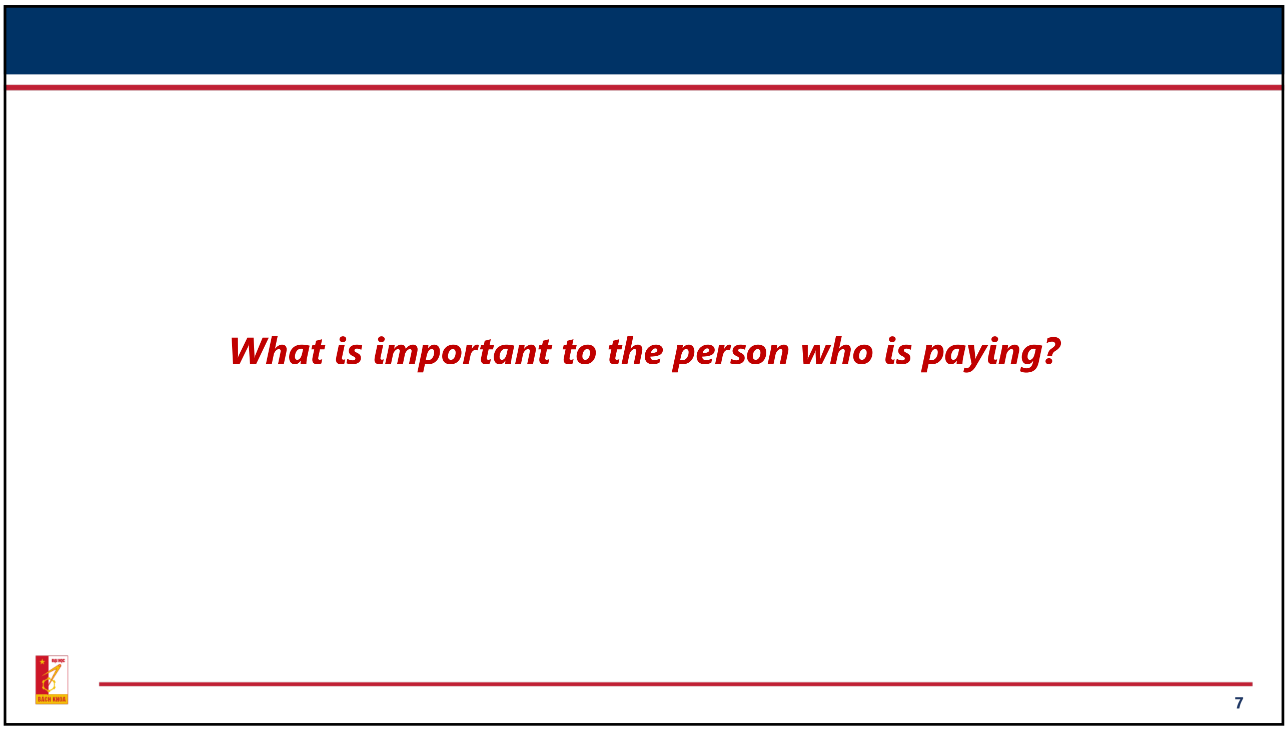
## Clients - Customers - Users

- Customer is the person who buys the software or selects it for use by an organization
- User is a person who actually uses the software
  - With personal software, the user and the customer may be the same
  - In organizations, the customers and the users are usually different
- Example: Many people in our university use Microsoft Team

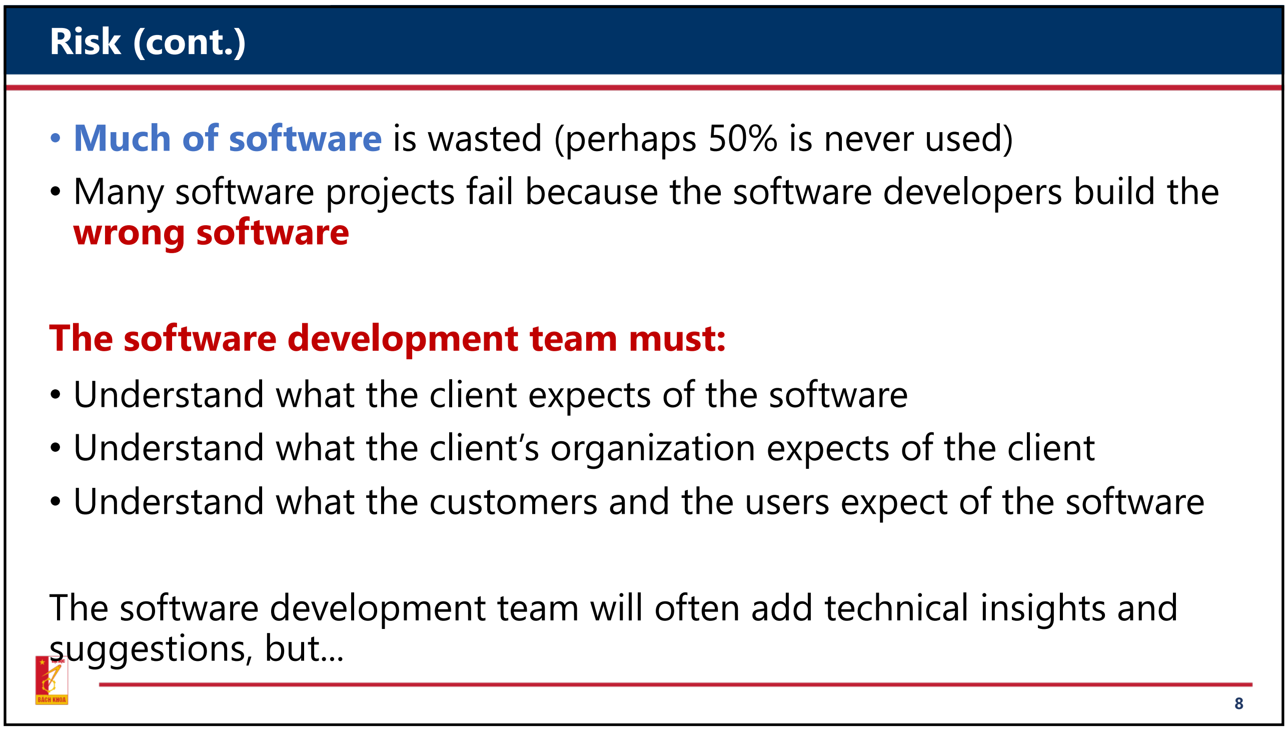
Who is the customer? Who are the users?

## Risk

- **Many (probably most) software development projects run into difficulties**
- Problems
  - Does not work as expected (**FUNCTION**)
  - Over budget (**COST**)
  - Late delivery (**TIME**)
- Competing goals
  - Every software project has a trade-off between **function**/**cost**/**time**
  - Extra functions add extra costs for development, testing, maintainance etc.



*What is important to the person who is paying?*



## Risk (cont.)

- **Much of software** is wasted (perhaps 50% is never used)
- Many software projects fail because the software developers build the **wrong software**

**The software development team must:**

- Understand what the client expects of the software
- Understand what the client's organization expects of the client
- Understand what the customers and the users expect of the software

The software development team will often add technical insights and suggestions, but…

***Client satisaction is a primary measurement of <u>success</u> in a <u>software project</u>***

## Minimizing Risk: Communication with Client

- **Feasibility studies** (should we begin the project?)
- **Separation** of **requirements** (what the client wants) from **design** (how the developers meet the requirements)
- **Milestones** (how the developers report or demonstrate progress to the clients) and **releases**
- **Acceptance** (how the client tests that the software meets the requirements) and **user testing**
- **Handover** (ensuring that the client receives a package that can be operated and supported over a long time period)

## Minimizing Risk: Visibility

- **Visibility** : The people who take the responsibility must know what is happening
- **The problem** (as seen by a manager) must rely on others for reports of progress or difficulties

*However, software developers:*

- Have difficulty evaluating progress
- Are usually optimistic about progress
- Consider reporting a waste time
- etc.

***You will make regular progress reports on your projects***

## Minimizing Risk: Short Development Cycle

Risk is minimized by frequent delivery of working software (weeks rather than months)

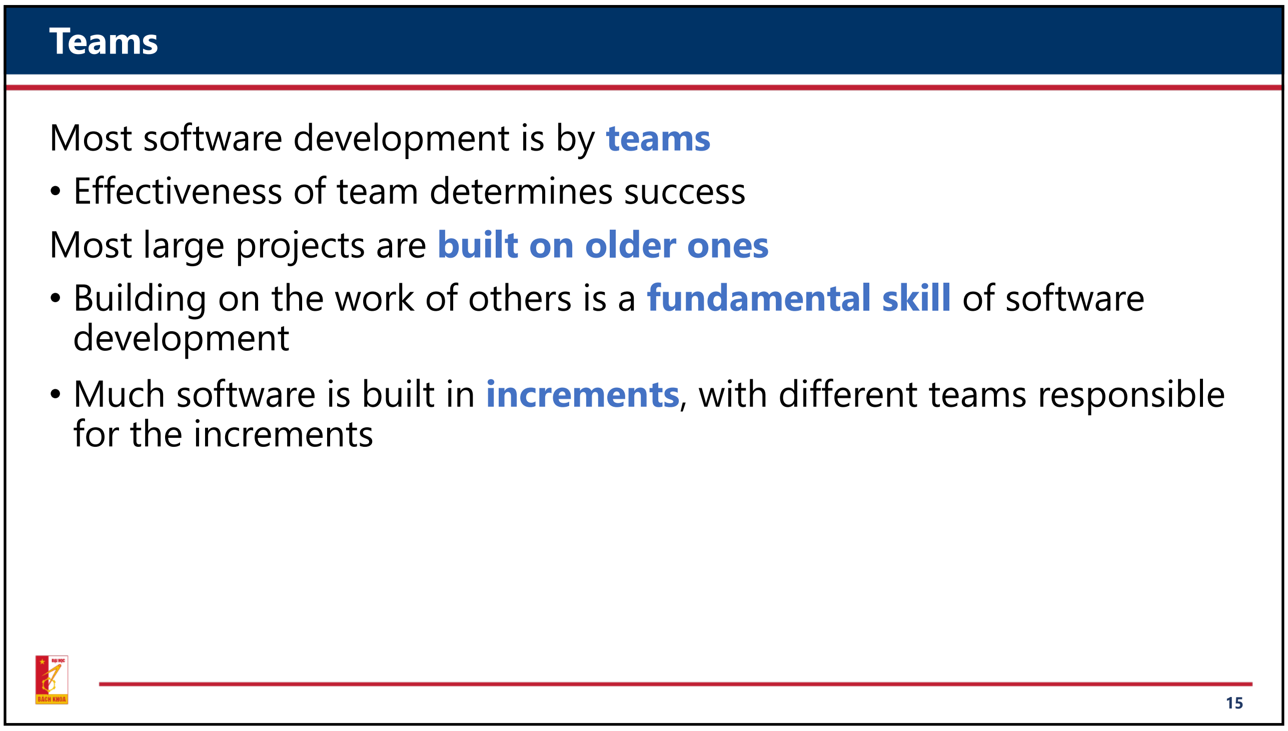
- Client, customers, and users can evaluate the developers' work
- Opportunities to adapt to changing circumstances

***This is one of the basic principles of agile software development***

## Software Project Scale

- Large and very large systems have different needs
- A big project may be of 100 to 10000+ person per years

- Every large system is developed by many people, who are constantly changing
- Before a big project is completed the requirements have changed many times
- No large system is ever complete
- Nobody comprehends more than a fraction of the project



## Teams

Most software development is by **teams**

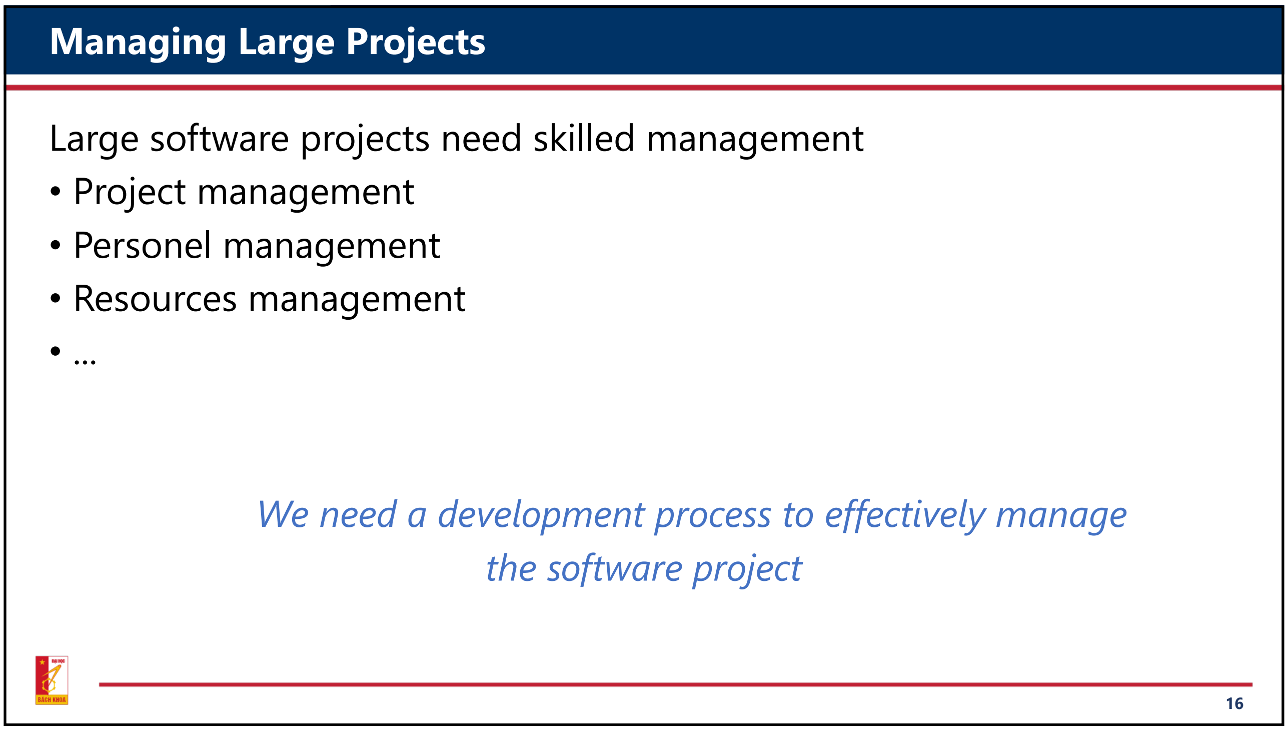
- Effectiveness of team determines success

Most large projects are **built on older ones**

- Building on the work of others is a **fundamental skill** of software development
- Much software is built in **increments**, with different teams responsible for the increments

15

15



## Managing Large Projects

Large software projects need skilled management

- Project management
- Personel management
- Resources management
- …

*We need a development process to effectively manage the software project*

16

16

## Software Development Processes

**Fundamental assumptions:**

- Good processes lead to **good software**
- Good processes **reduce risk**
- Good processes **enhance visibility**
- Good processes **enable teamwork**

## Variety of Software Processes

Software products are very varied…

… Therefore, there is no standard process for all software development projects

BUT successful software development projects all need to address similar issues

- This creates a number of **process steps** that should be part of all software projects

## Process Steps in all Software Development

Feasibility Study
Requirements
User Interface Design
System Design
Program Development (detail design and coding)
Acceptance and release
Operation and Maintenance

These steps may be repeated many times during the development cycle

*It is essential to distinguish among these steps and to be clear which you are doing at any given moment*

19

19

## Feasibility

A **feasibility study** precedes the decision to begin a project

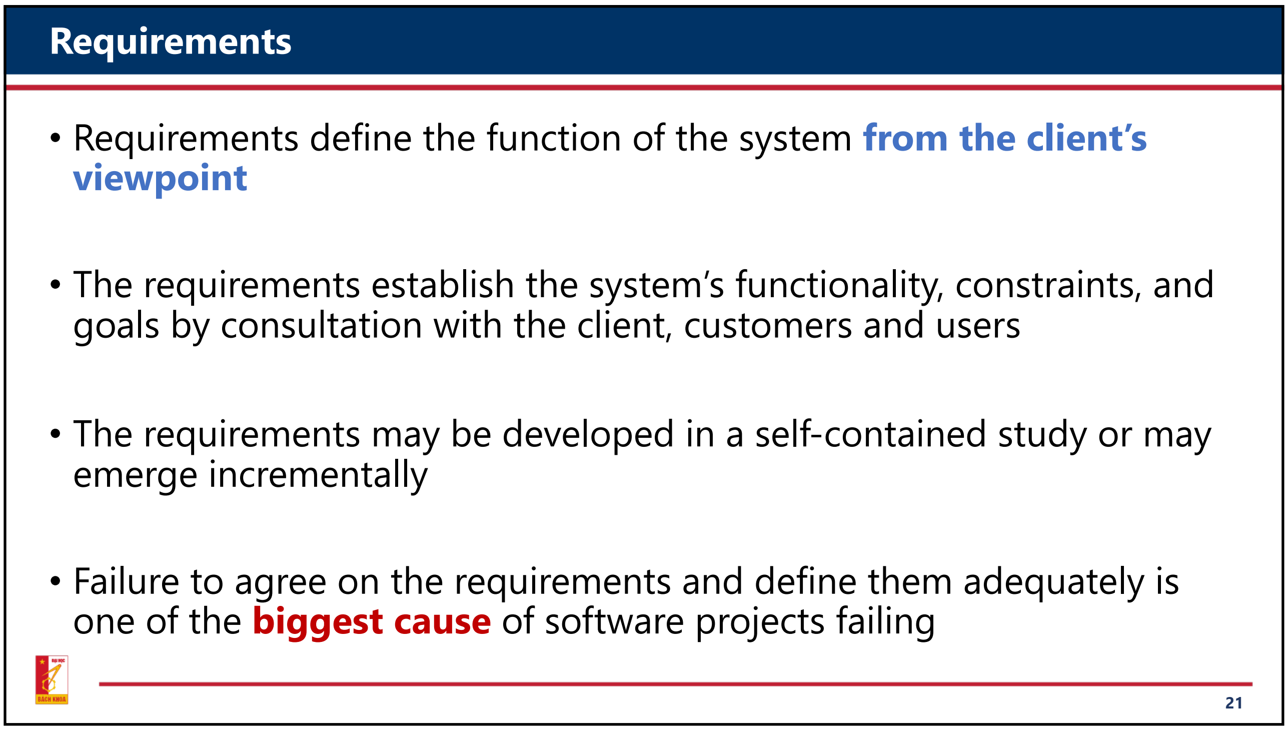
- What is the scope of the proposed project?
- Is the project technically feasible?
- What are the project benefits?
- What are the costs, timetable?
- Are the resources available?
- What are the risks and how can they be managed?

A feasibility study leads to a **decision**: **go** or **no-go**

**Lecture 6 is about Feasibility Study**

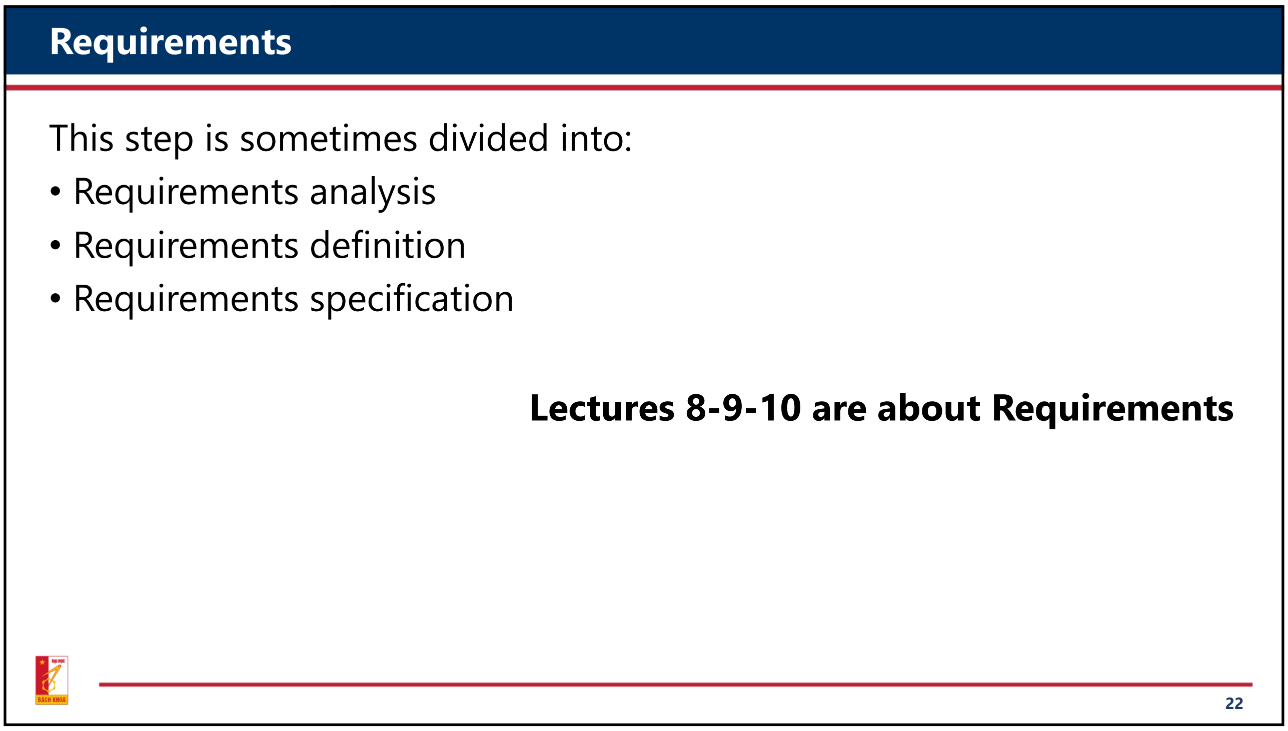
20

20



## Requirements

- Requirements define the function of the system **from the client's viewpoint**
- The requirements establish the system's functionality, constraints, and goals by consultation with the client, customers and users
- The requirements may be developed in a self-contained study or may emerge incrementally
- Failure to agree on the requirements and define them adequately is one of the **biggest cause** of software projects failing



## Requirements

This step is sometimes divided into:

- Requirements analysis
- Requirements definition
- Requirements specification

**Lectures 8-9-10 are about Requirements**

## User Interface Design

- Usability is of great importance in many modern applications and software systems. That requires good user interface design
- User interfaces need to be evaluated with users
- This requires iterative development:
  - Design the user interface
  - Test with users
  - Revise the user interface
  - *Repeat*

**Lectures 11-12-13 are about User Experience**

## System Design

Design describes the system from the **software developers' viewpoint**

**System Design**:

- Establish a system architecture, both hardware and software that matches the requirements
- Security and performance are parts of the system design

**Models**:

- Models are often used to represent the requirements, system design and program design
- This course teaches the basic concepts of the Unified Modeling Language (UML)

**Lecture 14-15-16-17 focuses on System Design**

## Program Development

Program development:

- Takes the system design and user interface design and builds programs that meet the requirements

Program development is sometimes divided into two parts:

- Program design (often refered as Detail/Module design)
- Implementation (coding)

Often these two parts overlap or are combined

**Lectures 18-19-20 are about Program Development**

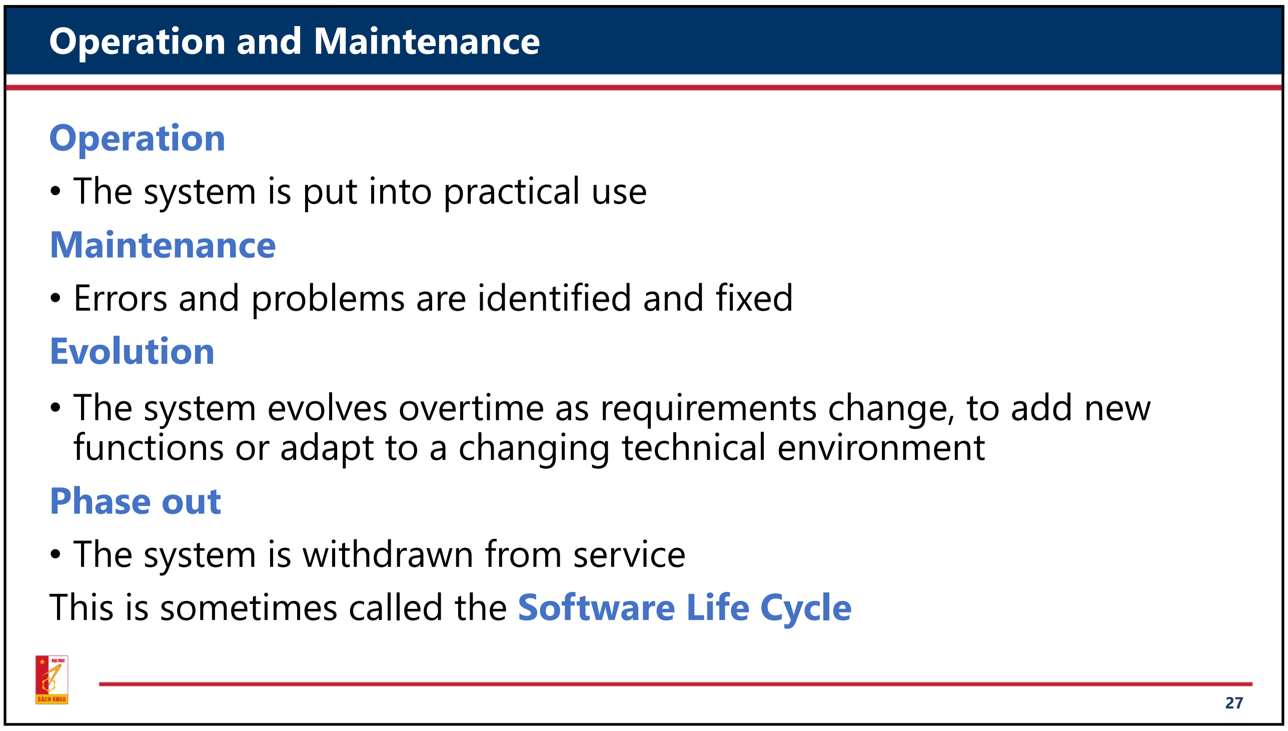
## Acceptance and Release

### Acceptance testing

- The system is tested against the requirements by the client, often with selected customers and users

### Delivery and release

- After successful acceptance testing, the system is delivered to the client and released into production or marketed to customers



## Operation and Maintenance

**Operation**

- The system is put into practical use

**Maintenance**

- Errors and problems are identified and fixed

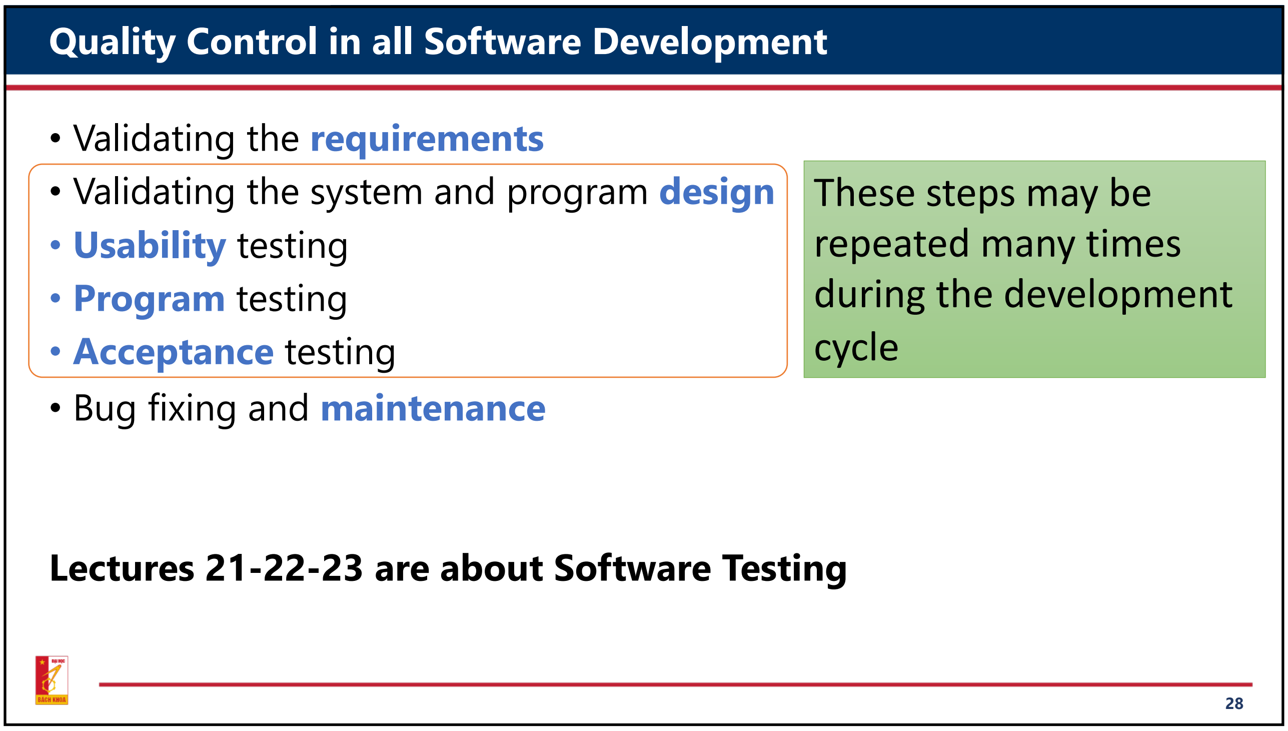
**Evolution**

- The system evolves overtime as requirements change, to add new functions or adapt to a changing technical environment

**Phase out**

- The system is withdrawn from service

This is sometimes called the **Software Life Cycle**



## Quality Control in all Software Development

- Validating the **requirements**
- Validating the system and program **design**
- **Usability** testing
- **Program** testing
- **Acceptance** testing
- Bug fixing and **maintenance**

These steps may be repeated many times during the development cycle

**Lectures 21-22-23 are about Software Testing**

## Categories of Testing

The term testing is used in several different contexts which are easily confused:

### User Testing

- Versions of the user interface are tested by users
- Their experience may lead to changes in the requirements or the design

### Program Testing

- The development team tests components individually (unit testing) or in combination (system testing) against the design to find bugs

### Acceptance Testing

- The client tests the final version of the system or parts of the system against the requirements

## 3. Software development in Practice

(end of lecture)

ONE LOVE. ONE FUTURE.