

# CSC00004

## Software engineering



KHOA CÔNG NGHỆ THÔNG TIN  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

# Content

- Fundamentals
  - The Software Engineering Discipline
  - The Software Life Cycle
  - Development Phases
  - Software Engineering Methodologies
  - Documentation
  - Project Management
- Other Aspects in SE
  - Software Design: Modularity
  - Tools of the Trade
  - Software Ownership and Liability
- SE Department – FIT – HCMUS

# FUNDAMENTALS

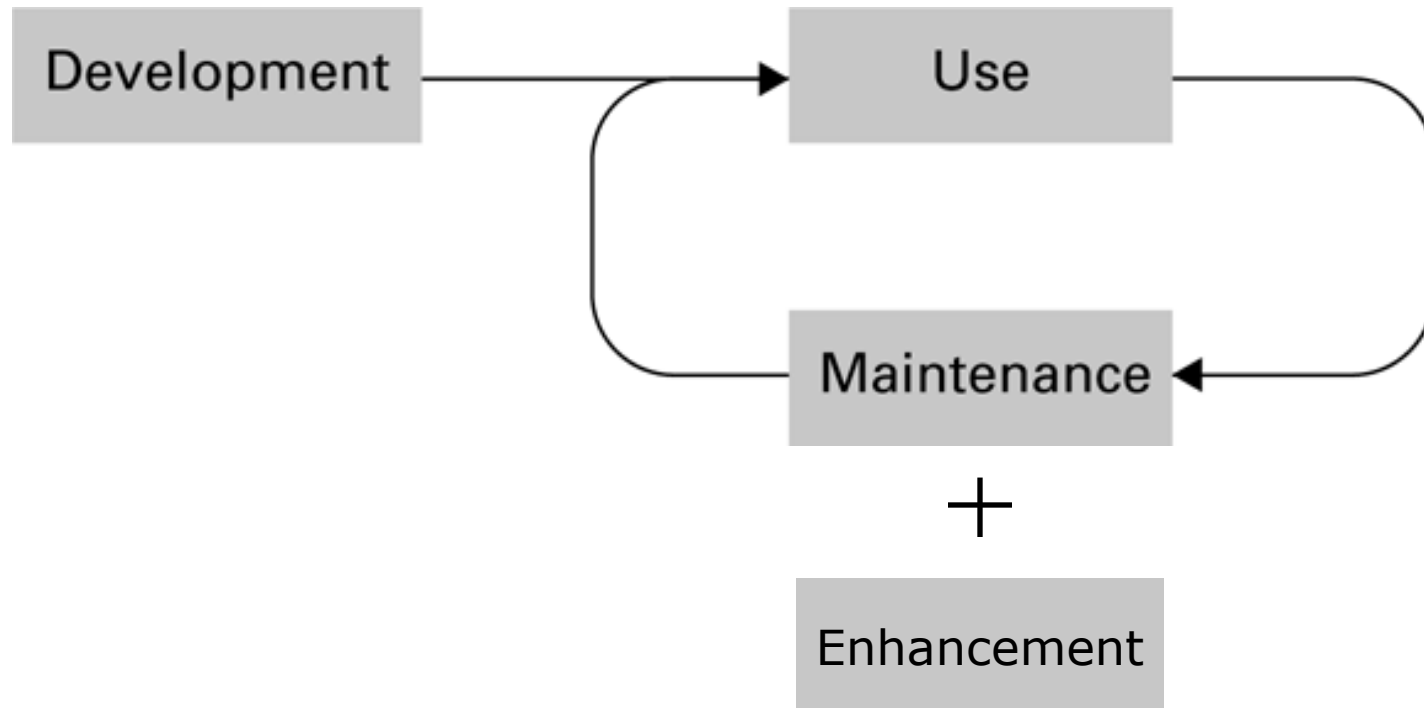
# The Software Engineering Discipline

- ☐ Distinct from other engineering fields
  - ☐ Prefabricated components
  - ☐ Metrics
  
- ☐ Practitioners versus Theoreticians
  
- ☐ Professional Organizations: ACM, IEEE...
  - ☐ Codes of professional ethics

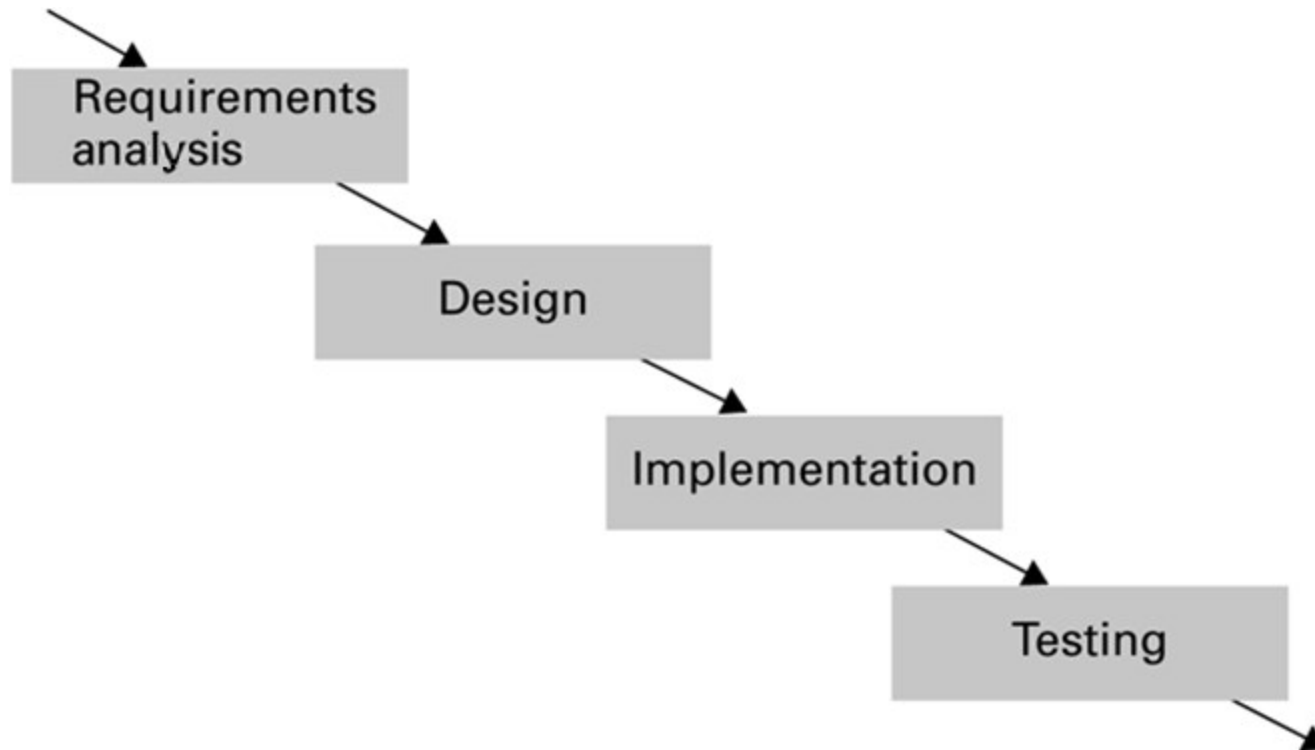
# Computer Aided Software Engineering (CASE) tools

- ☐ Project planning
- ☐ Project management
- ☐ Documentation
- ☐ Prototyping and simulation
- ☐ Interface design
- ☐ Programming
- ☐ Configuration management (git)
- ☐ Collaboration (Slack, Trello)

# The software life cycle



# The development phase of the software life cycle



# Analysis Stage

- ☐ Requirements
  - ☐ Application oriented
- ☐ Specifications
  - ☐ Technically oriented
- ☐ Software requirements document



# Design Stage

- ☐ Methodologies and tools
- ☐ Human interface (psychology and ergonomics)

# Implementation Stage

- ☐ Create system from design
  - ☐ Write programs
  - ☐ Create data files
  - ☐ Develop databases
- ☐ Role of “software analyst” versus “programmer”

# Testing Stage

- ☐ Validation testing
  - ☐ Confirm that system meets specifications
- ☐ Defect testing
  - ☐ Find bugs
- ☐ Level of test
  - ☐ Unit testing
  - ☐ Integration testing
  - ☐ System testing

# Software Testing Strategies

- ☐ Glass-box testing (white-box testing)
  - ☐ Pareto principle
  - ☐ Basis path testing
- ☐ Black-box testing
  - ☐ Boundary value analysis
  - ☐ Redundancy testing
  - ☐ Beta testing

# Software Engineering Methodologies

- ☐ Waterfall Model
- ☐ Incremental Model
  - ☐ Prototyping (Evolutionary vs. Throwaway)
- ☐ Open-source Development
- ☐ New Methodologies
  - ☐ Agile
  - ☐ Extreme Programming

# Scrum

## The Agile: Scrum Framework at a glance

Inputs from Executives,  
Team, Stakeholders,  
Customers, Users



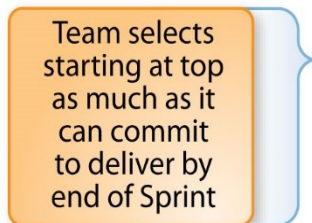
**Product Owner**



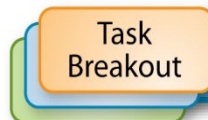
**The Team**



**Product Backlog**



**Sprint Planning Meeting**



**Sprint Backlog**



# Documentation

- ☐ User Documentation
  - ☐ Printed book for all customers
  - ☐ On-line help modules
- ☐ System Documentation
  - ☐ Source code
  - ☐ Requirement, analysis, design, test, maintenance documents
- ☐ Technical Documentation
  - ☐ For installing, customizing, updating, etc.
- ☐ Support documentation
  - ☐ For management, team working, collaboration

# Project Management

- A dynamic process that utilizes the appropriate resources of the organization in a controlled and structured manner, to achieve some clearly defined objectives identified as needs
  
- Consist of
  - ▣ Project Planning
  - ▣ Project Monitoring and Control
  - ▣ Management of Scope, Time, Cost, Risk, Quality, Human Resource, Communication, Procurement

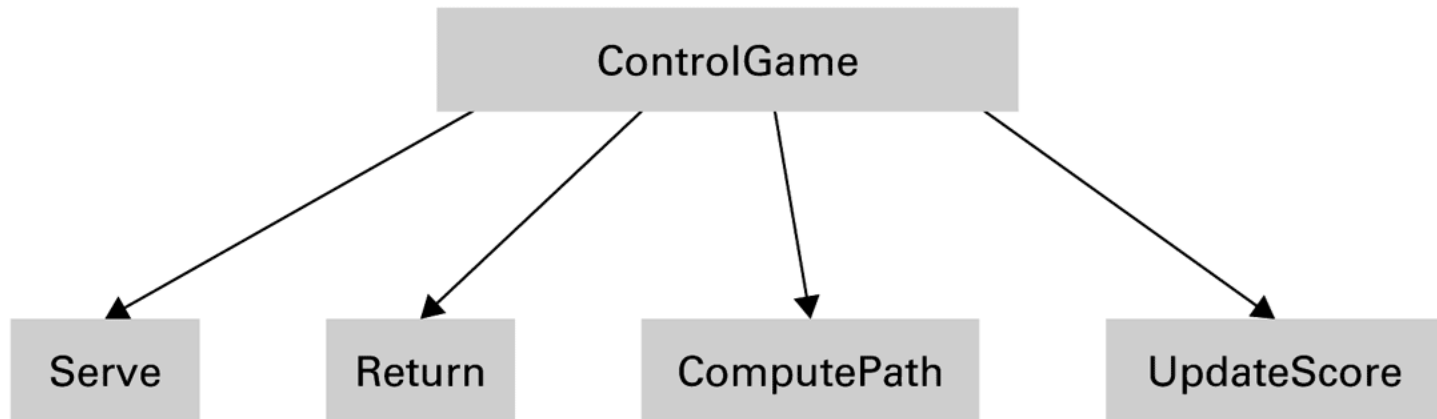


# OTHER ASPECTS OF SE

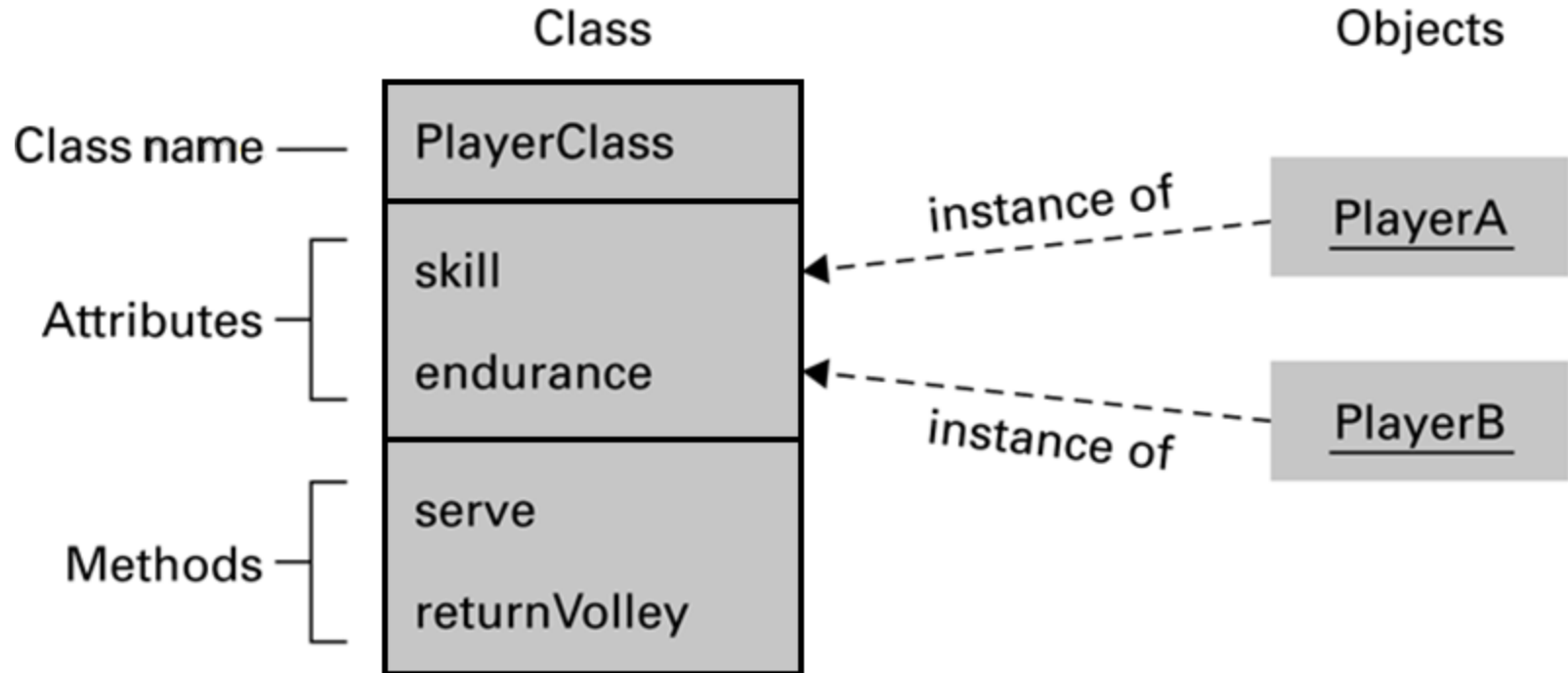
# Software Design – Modularity

- Functions – Imperative paradigm
  - ▣ Structure charts
- Class – Object-oriented paradigm
  - ▣ Collaboration diagrams
- Components – Component architecture
- Services – Service-oriented architecture (SOA)

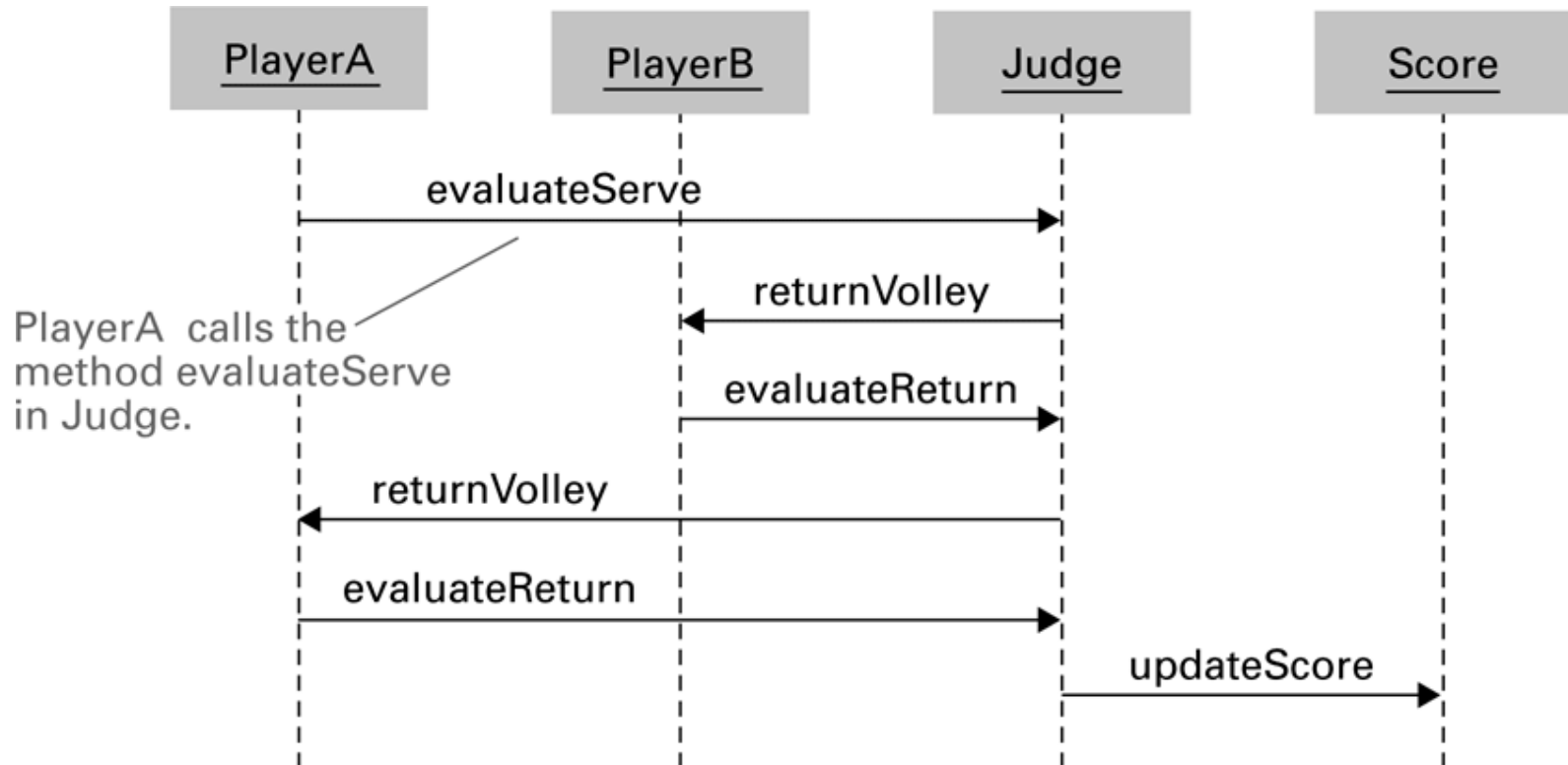
# A simple structure chart



# The structure of PlayerClass and its instances



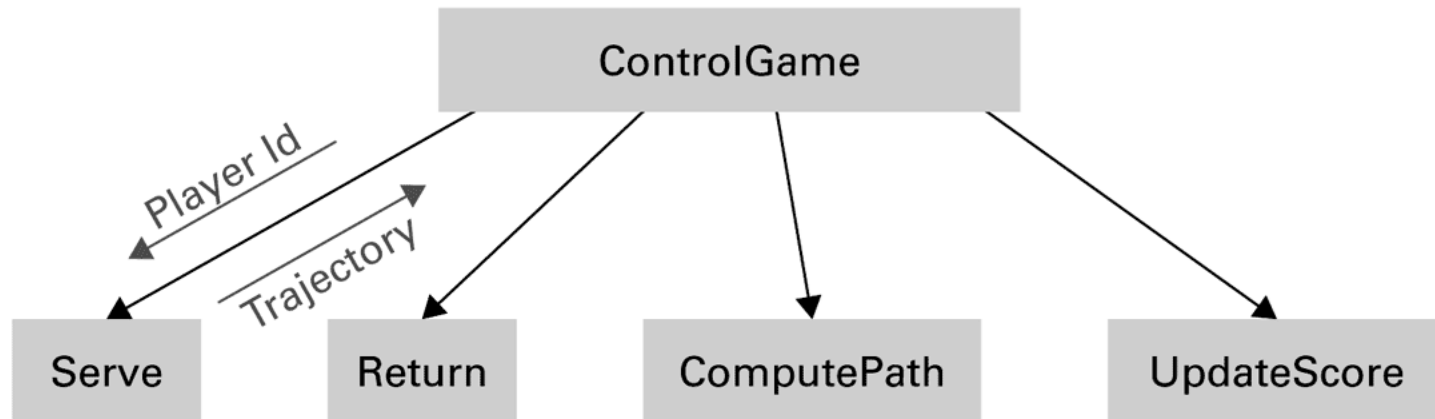
# The interaction between objects resulting from PlayerA's serve



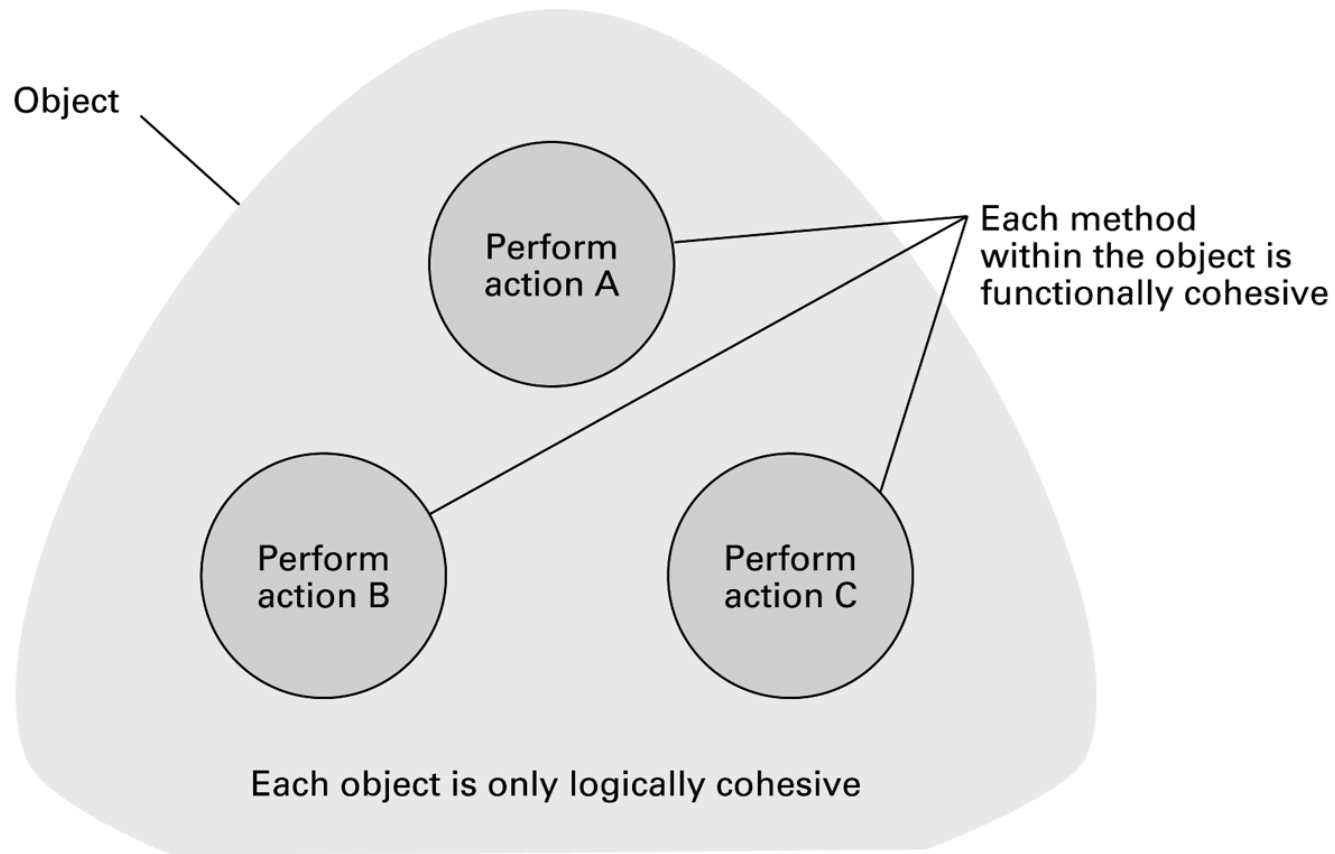
# Coupling vs. Cohesion

- ☐ Coupling
  - ☐ Control coupling
  - ☐ Data coupling
  
- ☐ Cohesion
  - ☐ Logical cohesion
  - ☐ Functional cohesion

# A structure chart including data coupling



# Logical and functional cohesion within an object





# Unified Modeling Language

## ☐ Use Case Diagram

- ☐ Use cases

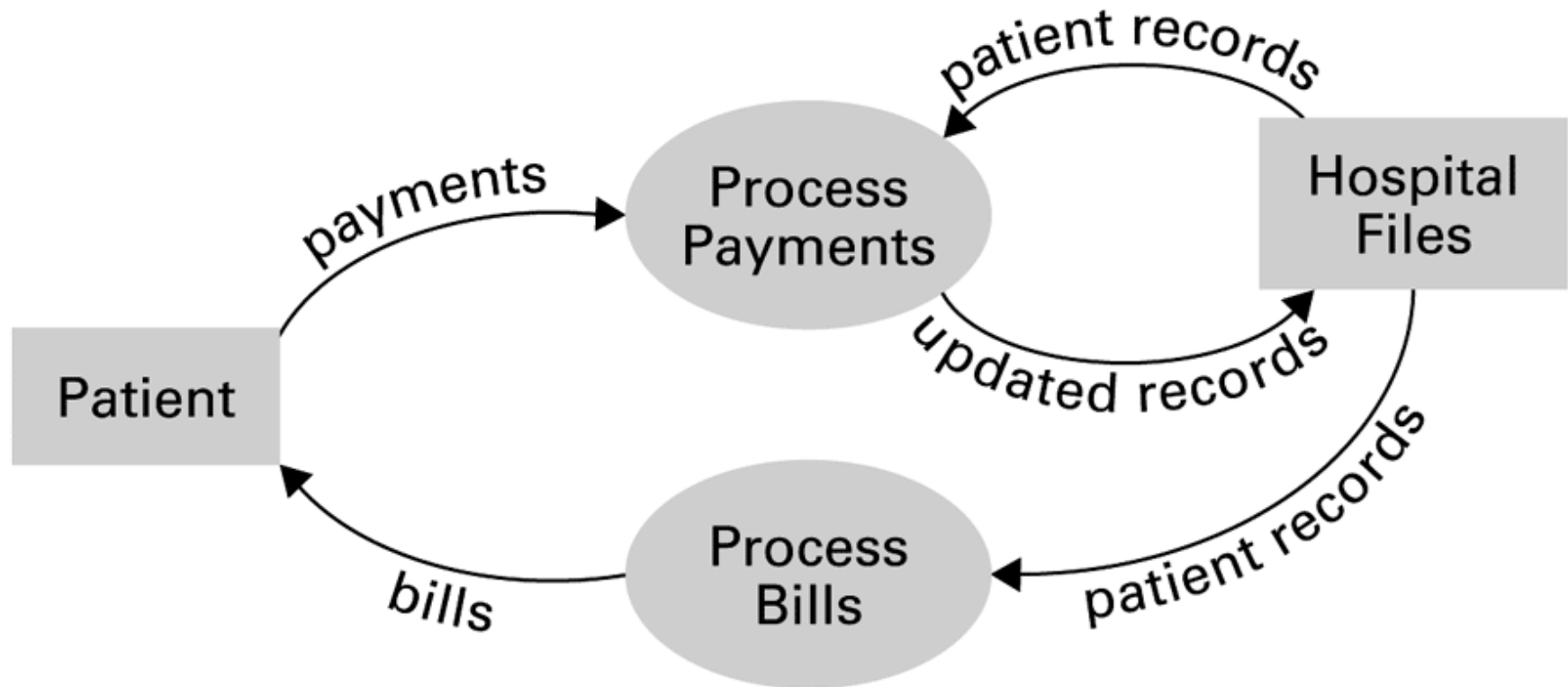
- ☐ Actors

## ☐ Class Diagram

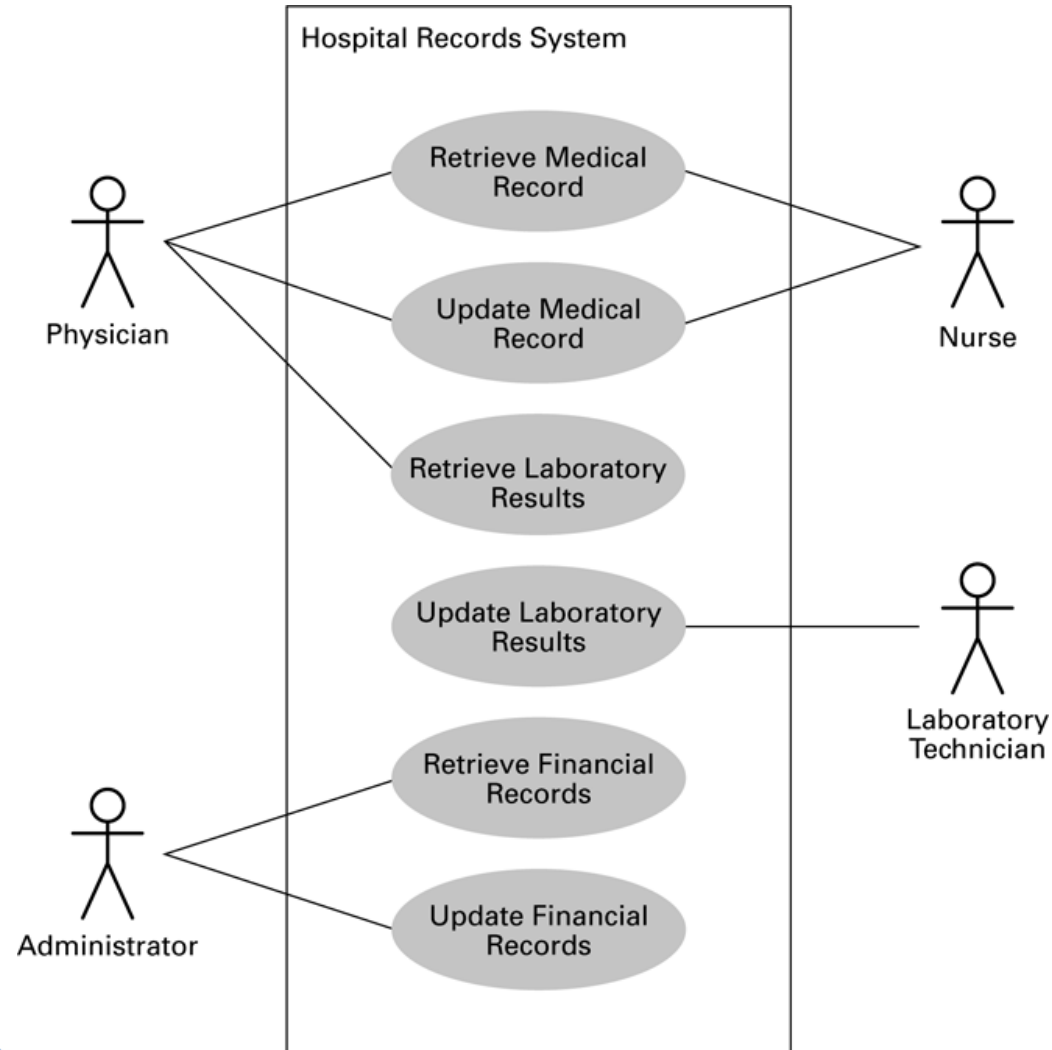
- ☐ Class

- ☐ Relationship: generalization, association, dependency

# A simple dataflow diagram



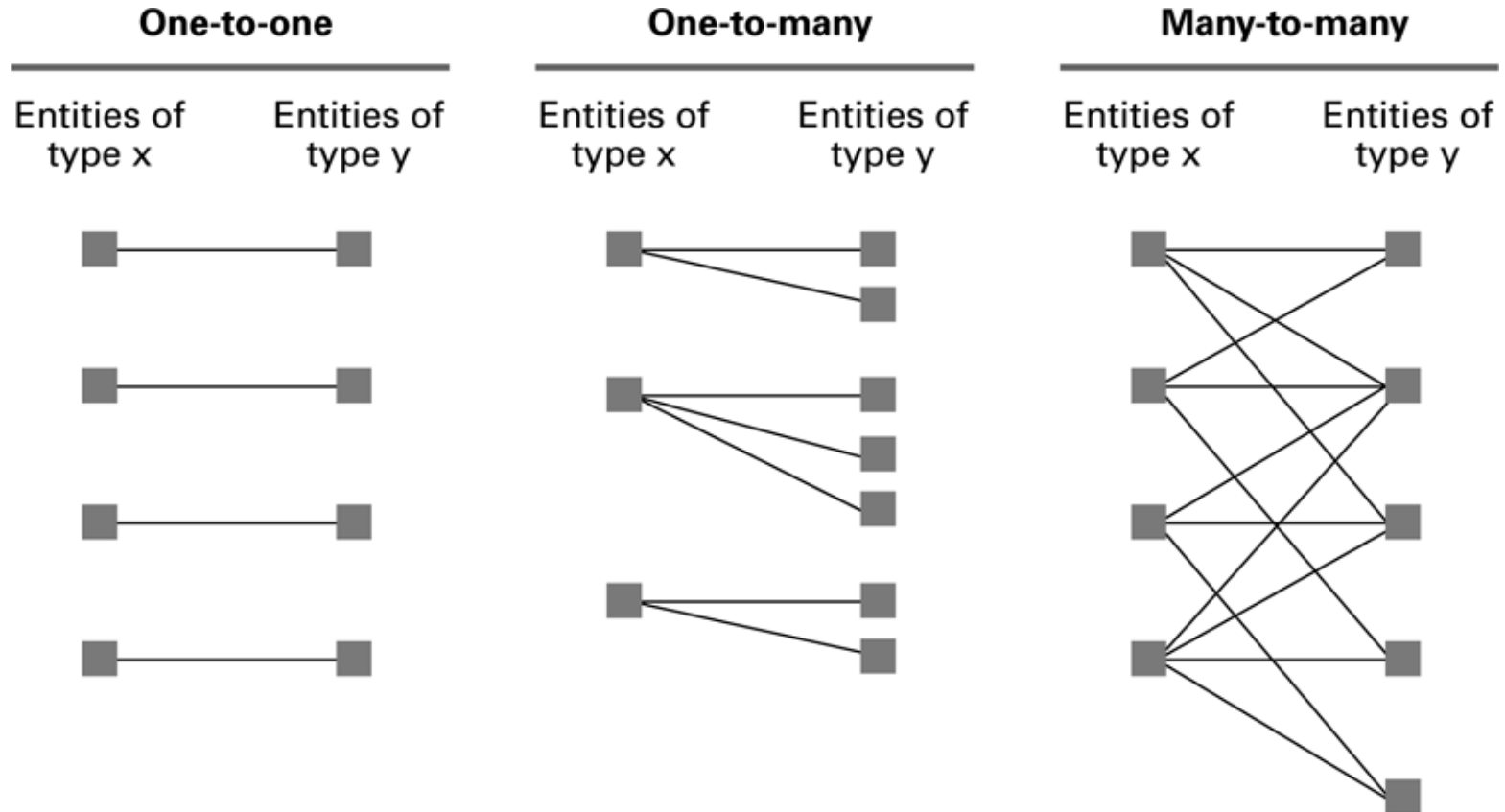
# A simple use case diagram



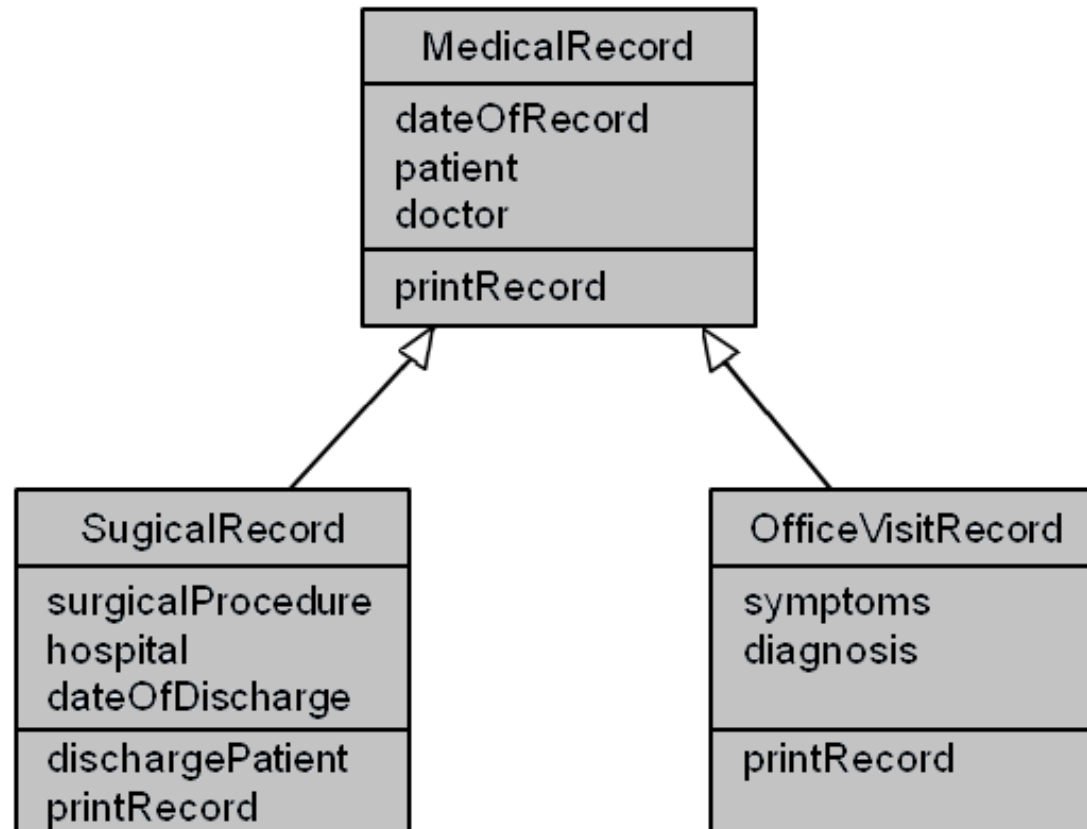
# A simple class diagram



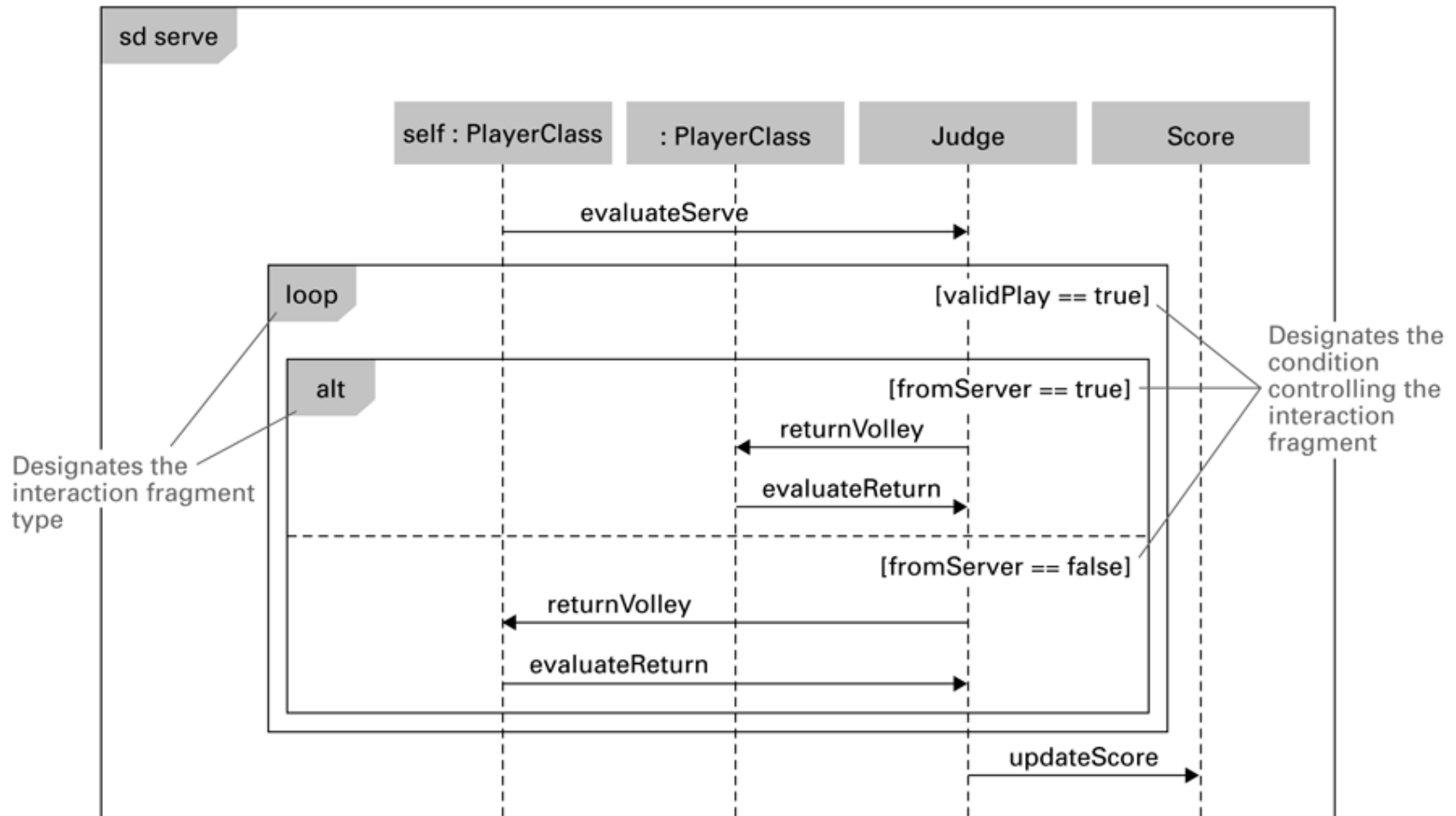
# Relationships between entities of types X and Y



# A class diagram depicting generalizations



# A sequence diagram depicting a generic volley



# Structured Walkthroughs

- ☐ “Theatrical” experiment
- ☐ Class-responsibility-collaboration cards



# Design Patterns

- Well designed “templates” for solving recurring problems
- Examples:
  - ▣ Adapter pattern: Used to adapter a module’s interface to current needs
  - ▣ Decorator pattern: Used to control the complexity involved when many different combinations of the same activities are required
- Inspired by the work of Christopher Alexander in architecture

# Software Ownership

## ☐ Copyright

- ☐ Allow a product to be released while retaining ownership of intellectual property
- ☐ Asserted in all works:
  - Specifications
  - Source code
  - Final product

# Software Ownership (continued)

## ☐ Software License

- ☐ A legal agreement that grants the user certain permissions without transferring ownership
- ☐ Open source license

## ☐ Patents

- ☐ Must demonstrate that it is new, usable, and not obvious to others with similar backgrounds
- ☐ Process is expensive and time-consuming

# DEPARTMENT OF SOFTWARE ENGINEERING

# Introduction

- ☐ Room: I82
- ☐ Phone: (028) 38 324 467 (ext: 802)
- ☐ Head of department
  - ☒ Dr. Nguyen Van Vu
- ☐ Vice head of department
  - ☒ Dr. Nguyen Thi Minh Tuyen

# Human resources

- 31 staffs are working in VN
  - Associate Professor – 1
  - Doctor – 5
  - M.Sc. – 24
  - B.Sc. – 1 (graduate student)



# Software engineering

- A discipline related to all aspects of professional software development
  
- Goal : develop a software system with
  - ▣ High quality
  - ▣ Low cost
  - ▣ On time
  - ▣ Customer satisfaction



# Importance of SE

- ☐ Software is an indispensable part of modern life
- ☐ The industries depend on software
- ☐ SE helps to develop software
  - ☐ Lower costs
  - ☐ Shorter time
  - ☐ Better response

# Importance of SE (cont.)

## □ Examples

### □ Programing languages are easier to use

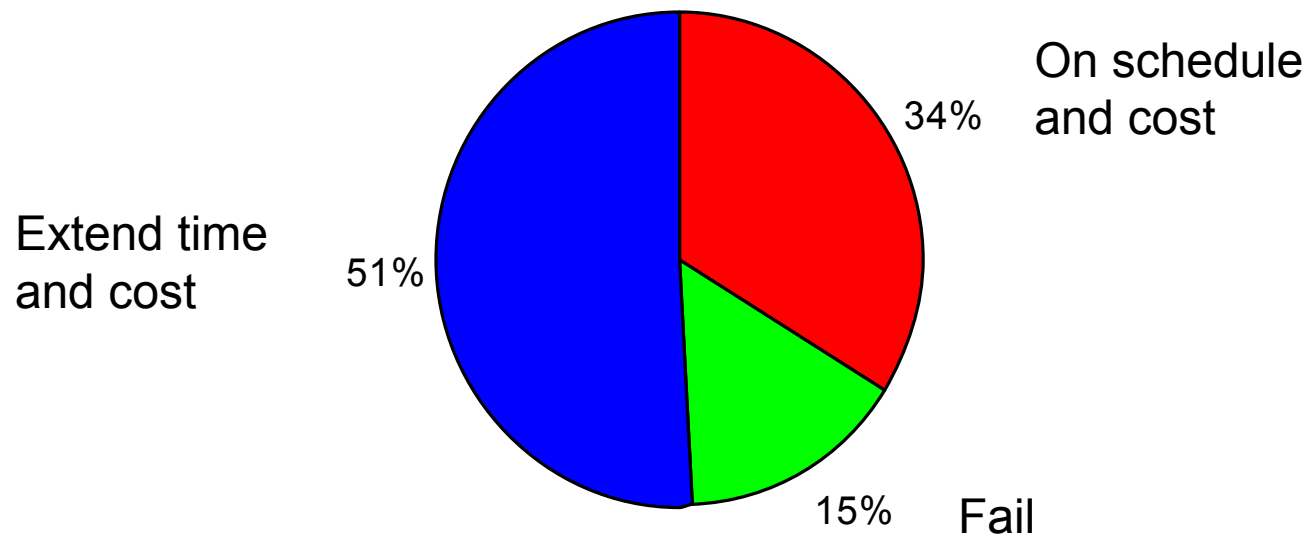
- From Assembly to Fortran, C, Cobol, C++, Java

### □ From waterfall process to agile process

### □ Programming tools are more diverse and modern

# Challenges in SE

- Report of Standish Group Chaos 2003
  - ▣ The rate of successful software projects is low



# Challenges in SE (cont.)

- Software is different from hardware
  - ▣ Cannot touch
  - ▣ Flexible, easy to edit
  - ▣ Depend on human
  - ▣ Diverse
    - Environment
    - OS, devices
    - Functions, development approaches

# Educational objectives

- ☐ Ability to analyze requirements, design, test and deploy software systems
- ☐ Ability to learn to use and self-study software development tools
- ☐ Ability to self-study and research new technologies, methods and processes in software industry

# Required courses of SE major

□ Students accumulate at least 5 courses

## Nhập môn công nghệ phần mềm

Quản lý dự án phần mềm

Lập trình Windows

Phát triển ứng dụng Web

Phát triển game

Phân tích và quản lý yêu cầu phần mềm

Phân tích và thiết kế phần mềm

Kiểm chứng phần mềm

Phát triển phần mềm cho thiết bị di động

# Optional courses

- Students accumulate at least 5 courses, which contain at least 2 courses (8 credits) of SE department

Các chủ đề nâng cao trong CNPM

Thuật toán tổ hợp và ứng dụng

Nhập môn tư duy thuật toán

Quy hoạch tuyến tính

Khởi nghiệp (2 tín chỉ)

Thiết kế **giao diện**

**Mô hình hóa** phần mềm

**Kiến trúc** phần mềm

**Thanh tra** mã nguồn

Kiến tập nghề nghiệp (3 tín chỉ)

# Optional courses (cont.)

Mẫu thiết kế HĐT và ứng dụng

Lập trình hướng đối tượng nâng cao

Lập trình ứng dụng **Java**

Đặc tả hình thức

Công nghệ **XML** và ứng dụng

Công nghệ **Java** cho hệ thống **phân tán**

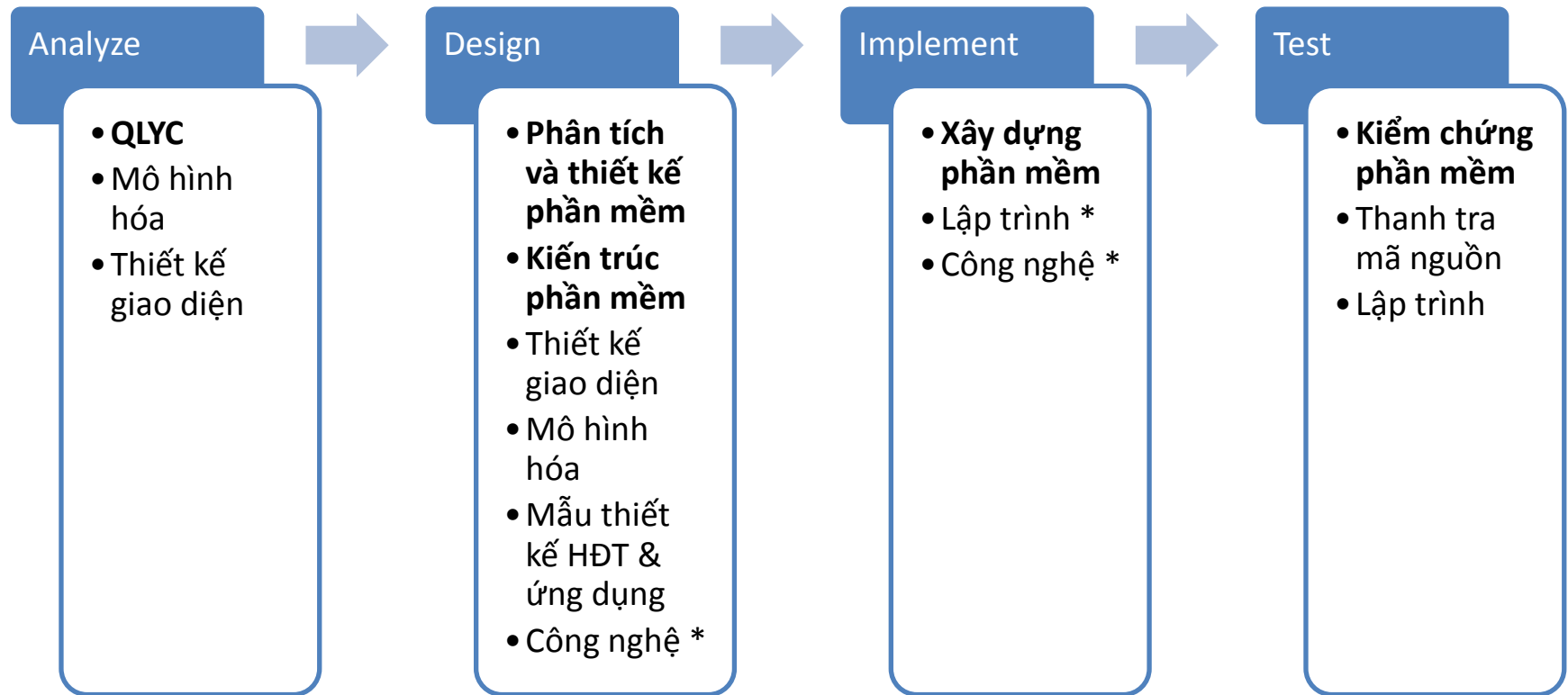
Phát triển phần mềm **nguồn mở**

Phát triển phần mềm cho **hệ thống nhúng**

Thực tập thực tế

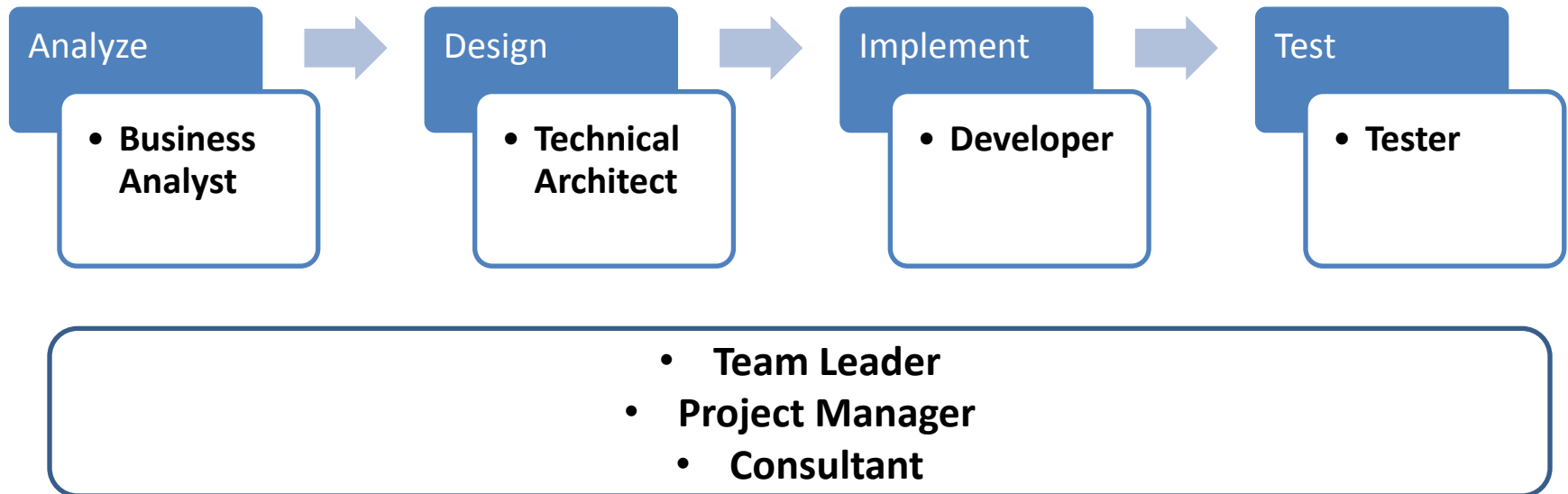


# Correlation of courses and SE

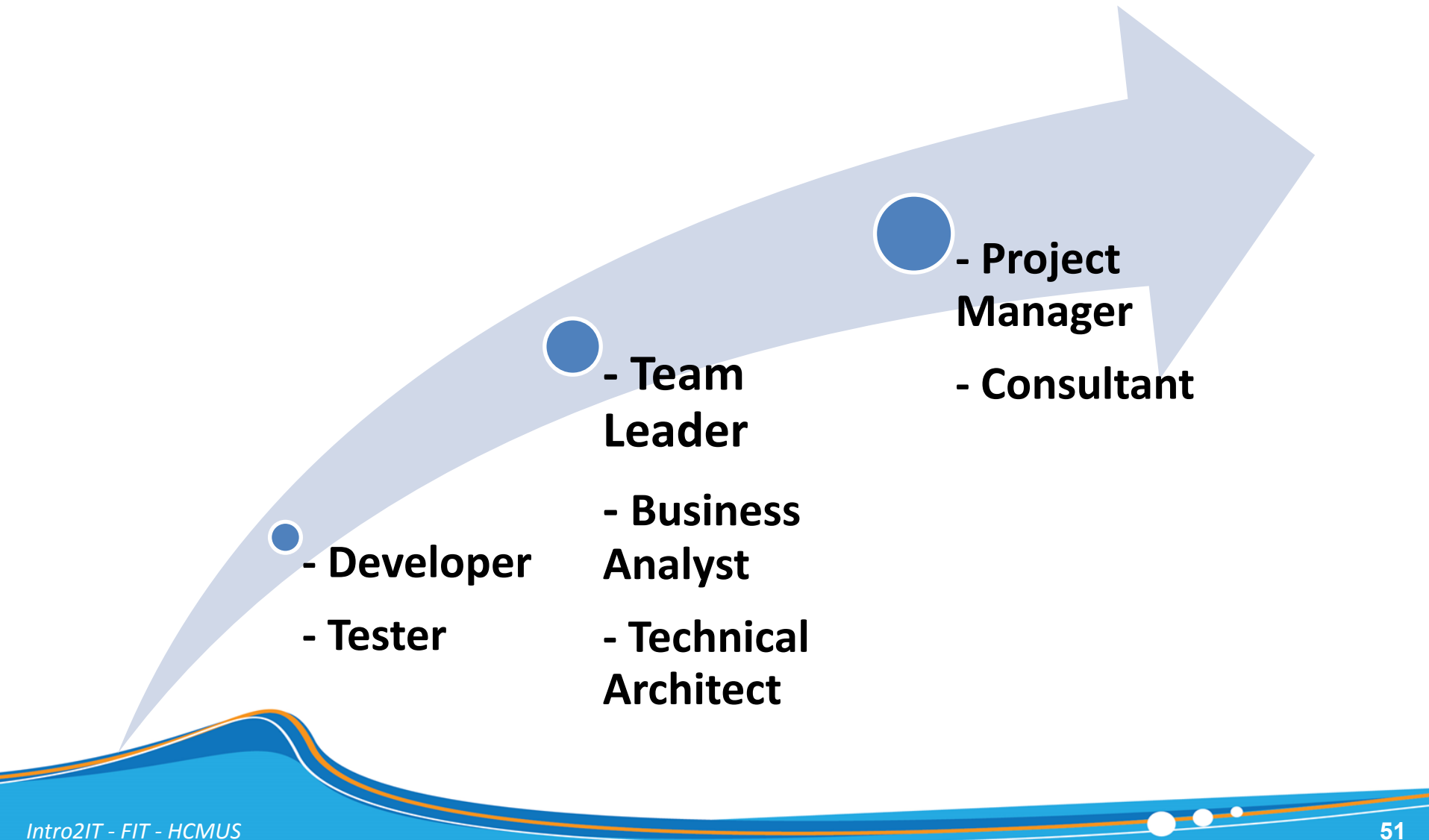


- Nhập môn CNPM
- Quản lý dự án phần mềm
- Các chủ đề nâng cao trong CNPM

# Career orientation



# Career orientation (cont.)



# Research directions

- ☐ Software estimation
- ☐ Test automation
- ☐ Mining software repositories
- ☐ Software cloud-based services
- ☐ Software engineering education
- ☐ Software verification and validation
- ☐ Human computer interaction

# Research directions (cont.)

- ☐ Software design, software architecture and software design patterns
- ☐ Engineering of desktop, web and mobile enterprise software
- ☐ Cloud computing application development
- ☐ Applying AI/ML/NLP to address software engineering problems

# Research groups

- ☐ Associate Prof. Trần Minh Triết
- ☐ Dr. Đinh Bá Tiến
- ☐ Dr. Nguyễn Văn Vũ
- ☐ Dr. Nguyễn Thị Minh Tuyên
- ☐ Dr. Ngô Huy Biên

