COM1013 INTRODUCTION TO COMPUTER SCIENCE

Lecturer: Begüm MUTLU BİLGE, PhD

begummutlubilge+com1013@gmail.com (recommended)
bmbilge@ankara.edu.tr

Software Engineering

Software engineering is the branch of computer science that seeks principles to guide the development of large, complex software systems.

- Includes topics such as personnel and project management

Software Engineering

Lets select a large complex device (may be an automobile) and imagine being asked to **design it** and then to **supervise** its construction.

Software Engineering

Questions must be answered during the development of a large software system.

- How can you estimate the **cost** in **time**, **money**, and

other resources to complete the project? ● How can you **divide** the project into **manageable pieces**?

- How can you ensure that the **pieces** produced are **compatible**?
- How can those working on the various **pieces communicate**?
- How can you **measure progress**?
- How can you cope with the wide range of **detail**?

The Software Life Cycle

The Traditional Development Phase

Requirements Analysis

The software life cycle begins with requirements analysis

- The goal: to specify what services the proposed system will provide,
  - To identify any conditions (time constraints, security, and so on) on those services,
  - To define how the outside world will interact with the system.

Requirements Analysis

The requirements analysis process:

- Compiling and analyzing the needs of the software user;
- Negotiating with the project's stakeholders ○ over trade-offs between wants, needs, costs, and feasibility;

- Developing a set of requirements
    - that identify the features and services that the finished software system must have.

Requirements Analysis

Requirements are recorded in a document called a **Software Requirements Specification (SRS)**.

- A written agreement between all parties concerned,
    - Intends to guide the software's development and
    - provides a means of resolving disputes that may arise later in the development process.

Professional organizations such as IEEE and large software clients such as the U.S. Department of Defense have adopted standards.

Straightforward and **frequent communication** with the project’s stakeholders is mandatory.

Requirements Analysis

Design

Whereas requirements analysis provides a description of the proposed software product,

- design involves creating a plan for the construction of the proposed system.

Requirements analysis is about identifying the problem to be solved (what),

- while design is about developing a solution to the problem (how).

Diagramming and modeling

Implementation

Implementation involves the actual writing of programs, creation of data files, and development of databases.

A programmer is charged with writing programs that implement the design produced by a software analyst.

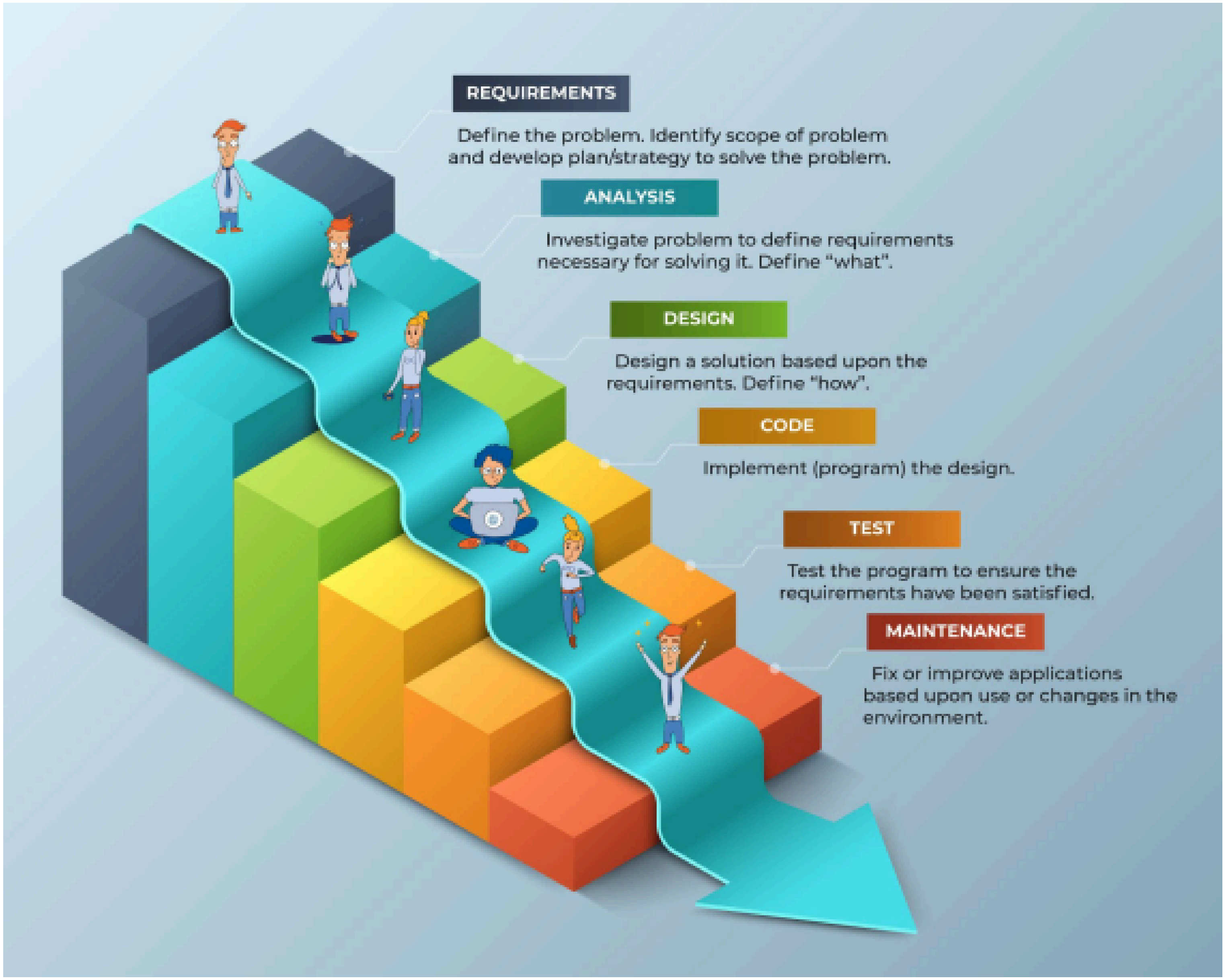
Testing

The result of each intermediate step in the entire development process should be “tested” for accuracy.

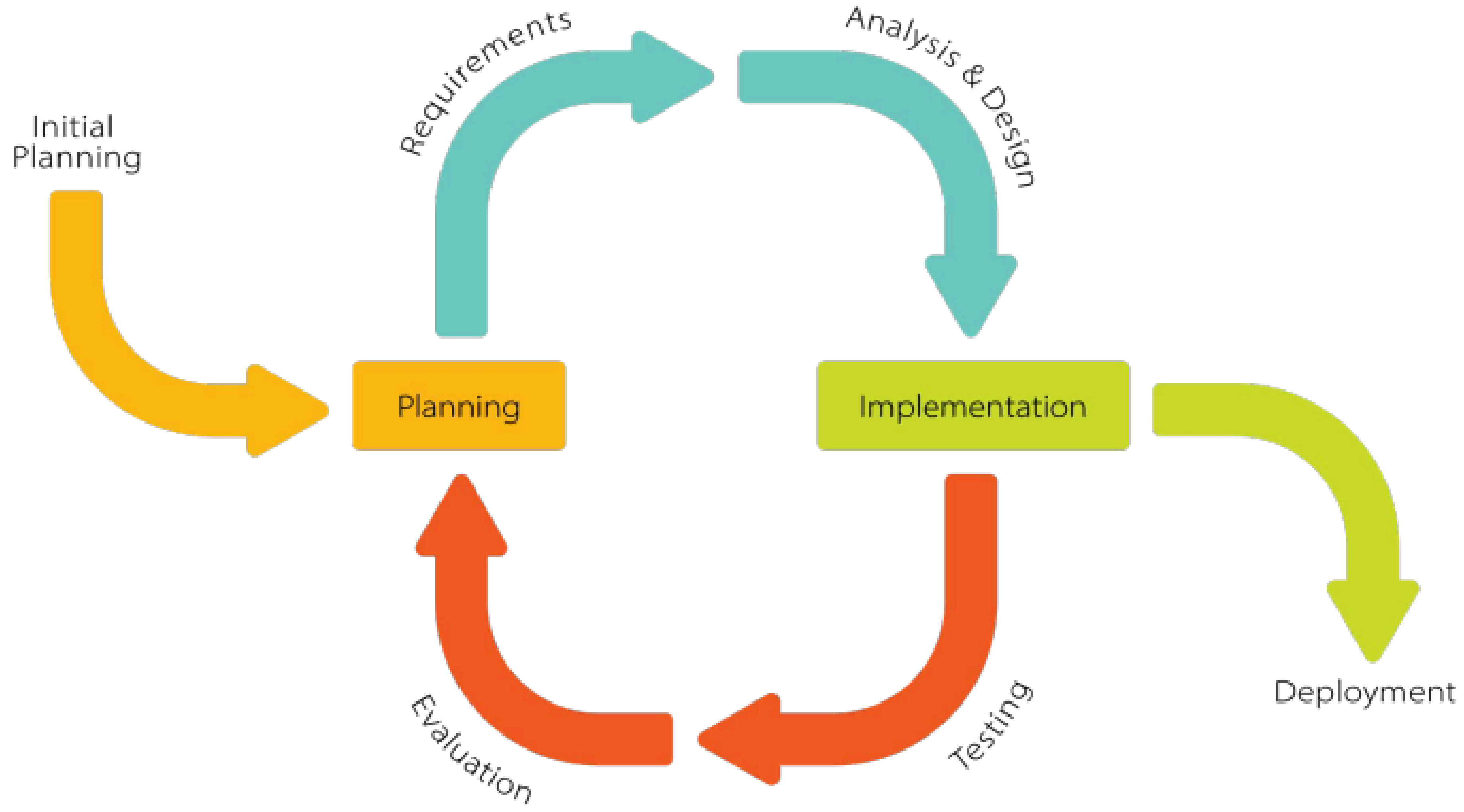
In the traditional development phase of the past, testing was essentially equated with

● the process of debugging programs and ● confirming that the final software product was compatible with the SRS.

Software Engineering Methodologies: Waterfall

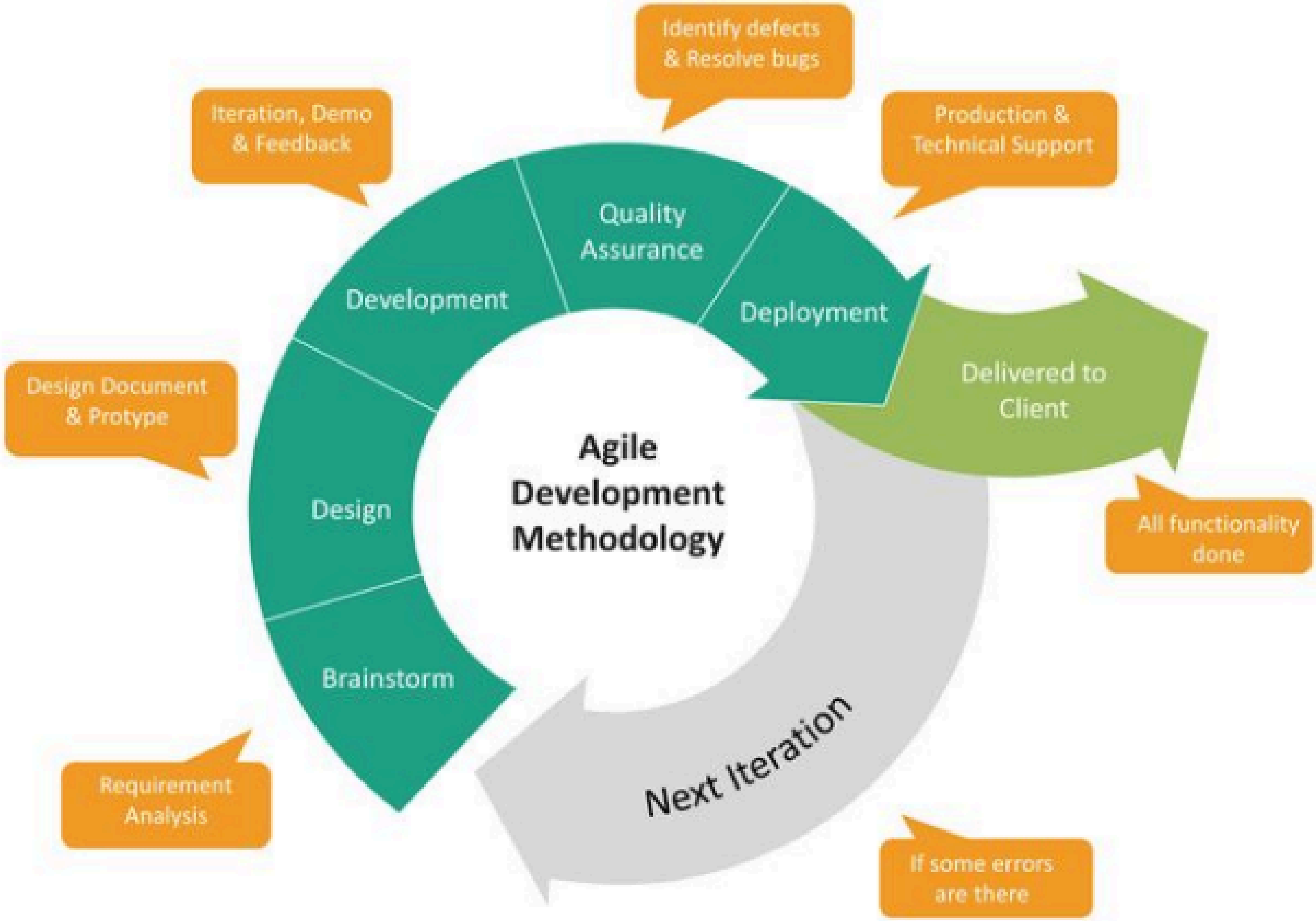

REQUIREMENTS
Define the problem. Identify scope of problem and develop plan/strategy to solve the problem.
ANALYSIS
Investigate problem to define requirements necessary for solving it. Define "what".
DESIGN
Design a solution based upon the requirements. Define "how".
CODE
Implement (program) the design.
TEST
Test the program to ensure the requirements have been satisfied.
MAINTENANCE
Fix or improve applications based upon use or changes in the environment.

Software Engineering Methodologies: Iterative & Incremental



Whereas the incremental model carries the notion of extending each preliminary version of a product into a larger version, the iterative model encompasses the concept of refining each version. In reality, the incremental model involves an underlying iterative process, and the iterative model may incrementally add features.

Software Engineering Methodologies: Agile



Software Engineering Methodologies

Tools of the Trade

In this section we investigate some of the **modeling techniques** and **notational systems** used during the analysis and design stages of software development.

Dataflow

Although the imperative paradigm seeks to build software in terms of procedures or functions, a way of identifying those functions is to consider the data to be manipulated rather than the functions themselves.

Dataflow analysis leads to the identification of functions.

A dataflow diagram is a means of representing the information gained from such dataflow studies.

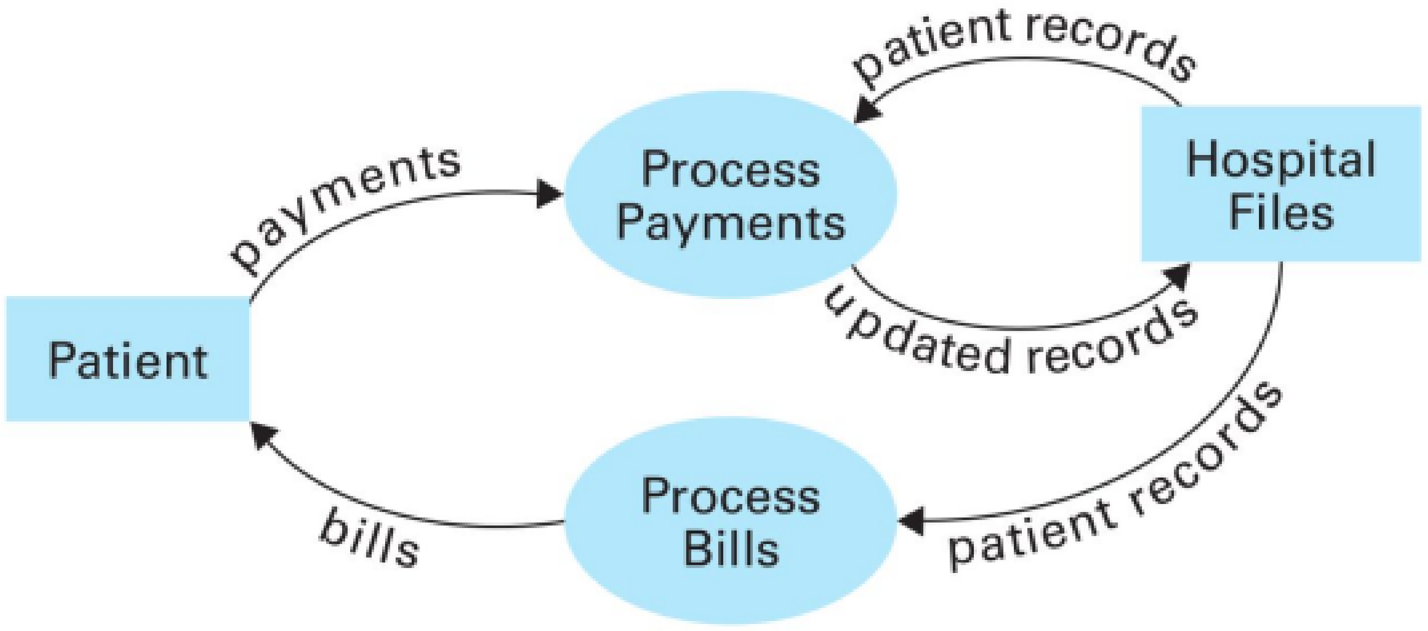
- Assist in identifying procedures during the design stage of software development,

- useful when trying to **gain an understanding** of the proposed system during the analysis stage.
- improves **communication** between clients and software engineers

Dataflow

In a dataflow diagram,

- **arrows** represent **data paths**,
- **ovals** represent **points at which data manipulation occurs**, and
- **rectangles** represent **data sources** and **stores**.



Unified Modeling Language -UML

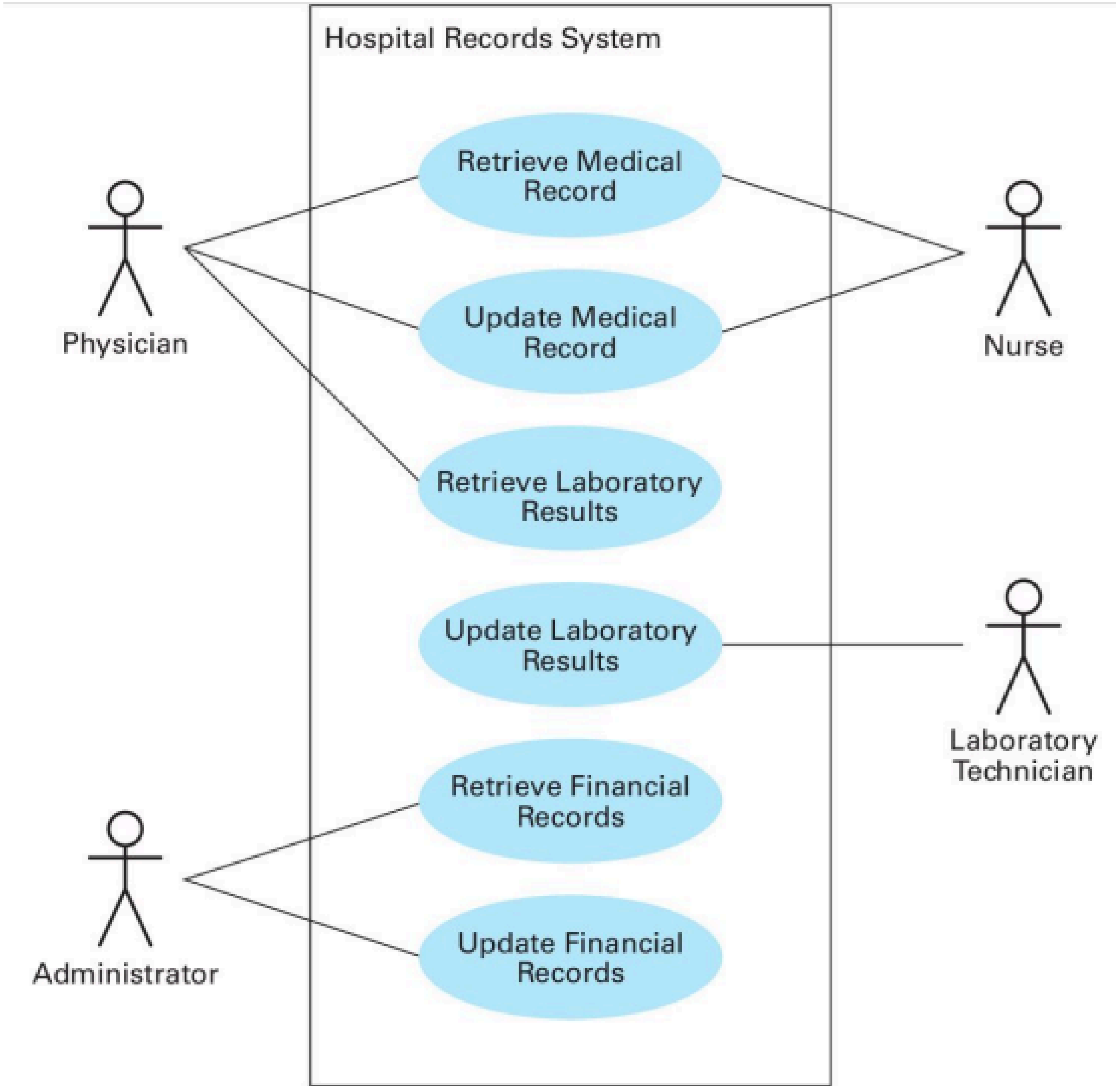
Modern **collection** of tools

- Has been developed with the object-oriented paradigm in mind.

[redacted]

- Captures the image of the proposed system from the user’s point of view

- Depicts the proposed system as a large rectangle in which
    - **interactions** (called use cases) **between the system and its users** are represented as **ovals** and
    - **users** of the system (called actors) are represented as **stick figures** (even though an actor may not be a person).


Hospital Records System
Retrieve Medical Record
Update Medical Record
Retrieve Laboratory Results
Update Laboratory Results
Retrieve Financial Records
Update Financial Records
Physician
Nurse
Laboratory Technician
Administrator

A

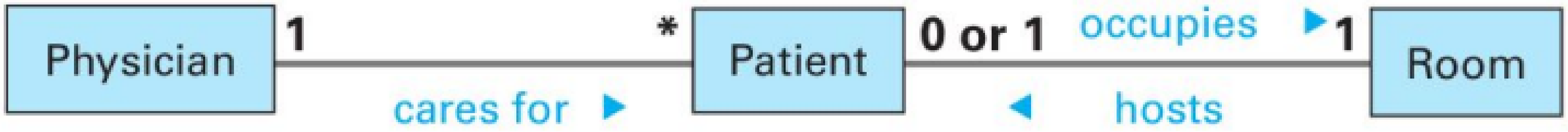
simple use case diagram

Unified Modeling Language

[redacted]

A notational system for representing the structure of classes and relationships between classes.

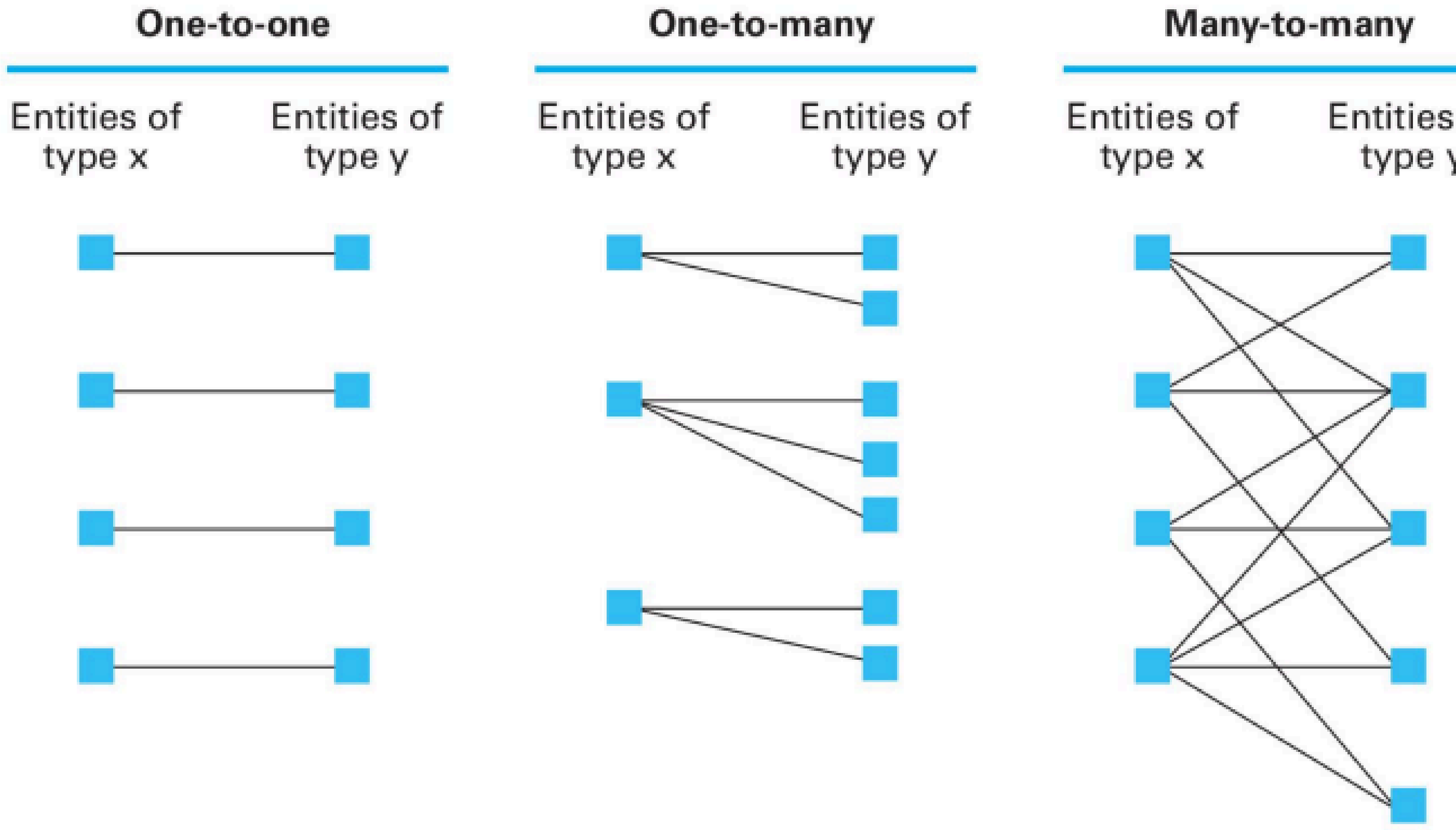
- **Classes** are represented by **rectangles** and
- **Associations** are represented by **lines**.
    - Association lines may or may not be **labeled**.
    - If they are labeled, a bold **arrowhead** can be used to indicate the **direction**.
    - A class diagram can also convey the multiplicities of those associations.



Unified Modeling Language

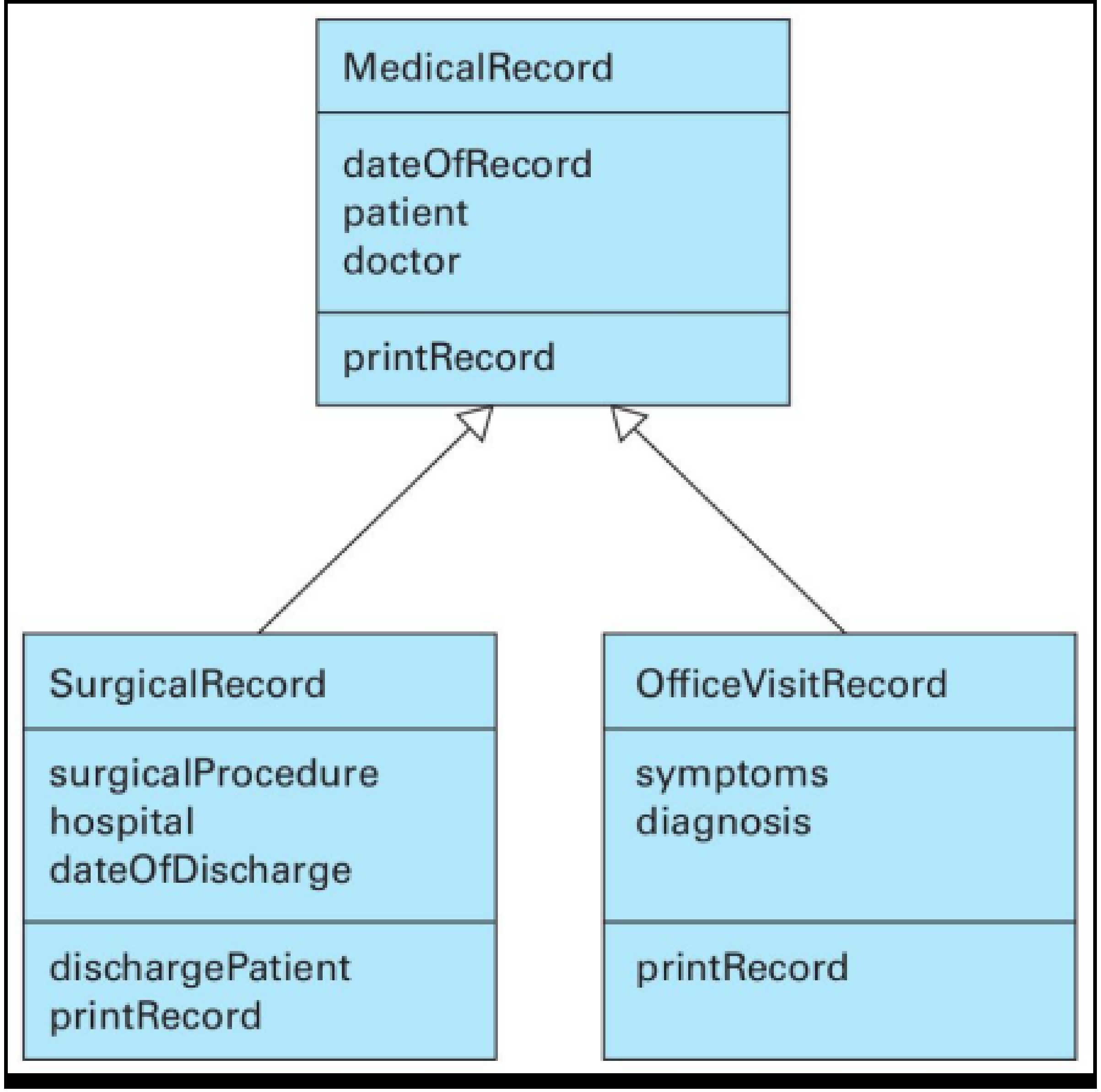
[redacted]

Association multiplicities occur in three basic forms



Unified Modeling Language

[redacted]

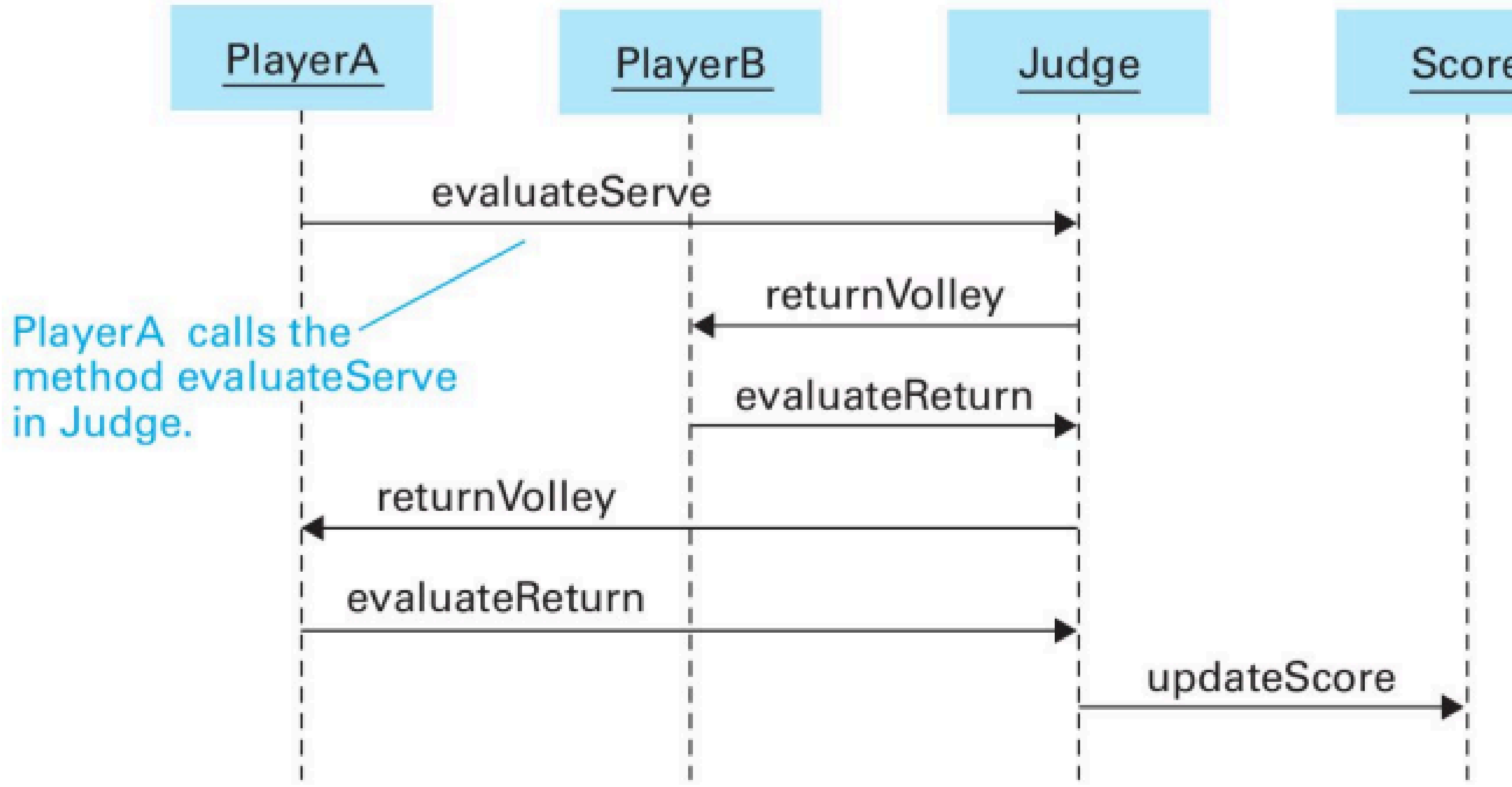


A class diagram depicting generalizations

Unified Modeling Language

Interaction diagrams: sequence diagram

● that depicts the communication between the individuals



Unified Modeling Language

A sequence diagram depicting a generic volley

Software Testing Documentation

Documentation is an important part of a final software package, and its development is, therefore, an important topic in software engineering.

Software documentation serves three purposes, leading to three categories of documentation:

- user documentation,
- system documentation, and
- technical documentation.

User documentation

The purpose of user documentation is to explain the features of the software and describe how to use them.

- intended to be read by the user of the software and is therefore expressed in the terminology of the application.

- an important marketing tool
- help packages

System documentation

The purpose of system documentation is **to describe the software's internal composition** so that the software can be **maintained later** in its life cycle.

2 major components of system documentation:

- The source version of all the programs in the system
- The design documents including
  - the software requirements specification and ○ records showing how these specifications were

obtained during design

Technical documentation

The purpose of technical documentation is to describe **how a software system should be installed and serviced**

- (such as adjusting operating parameters, installing updates, and reporting problems back to the software’s developer).

The distinction between technical documentation and user documentation is blurred in the PC arena

● Especially for the cases where the user is the person who also installs and services the software ● However, in multiuser environments, the distinction is sharper.

Extra Reading

Brookshear JG, Smith D, Brylow D.

“Computer science: an overview”.

Read the following section in the textbook.

Chapter: 7.9 Software Ownership and Liability

Summarize the section in 1 page (no figure should be included.)

Template for summarization assignments:

https://docs.google.com/document/d/1i1tU3AdIvPgRxqzGmXrsnVVCRXws5V3caDTCPMbPJ_0/edit?usp=sharing

Research Themes

1. Artificial Intelligence in Software Engineering Domain

2. Software Project Management
3. Software Quality
4. Brain Computer Interaction
5. Secure Coding

COM1013 INTRODUCTION TO COMPUTER SCIENCE

Lecturer: Begüm MUTLU BİLGE, PhD

begummutlubilge+com1013@gmail.com (recommended) bmbilge@ankara.edu.tr