UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS ILMU KOMPUTER

Dasar Dasar Pemrograman 2

oleh Ade Azurat

Topik Pekan 1: Pengantar

Kuliah DDP2 - 2024/2025 Semester Genap

- Pengajar
  - Dr. Ade Azurat
  - Aprinaldi, PhD
  - I Putu Edy Suardiyana Putra, Phd
  - Evi Yulianti, PhD
  - Iis Solichah, M.Kom., M.Comp.Sc.
  - Ilma Ainur Rohma, M.Kom

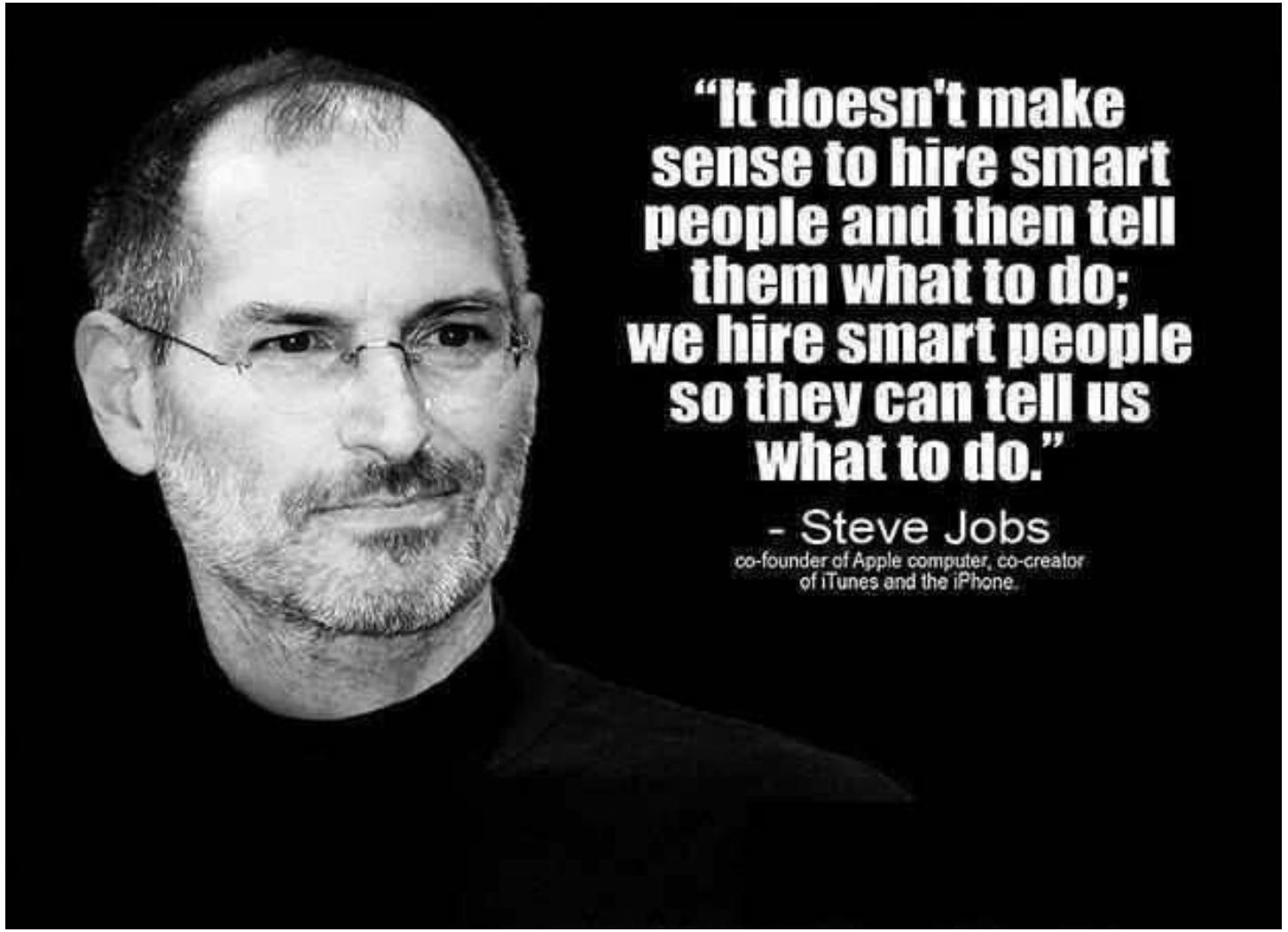
Who am I?

- Since 2004: Mostly Teaching Programming and Software Engineering related courses
- Since 2013: Teaching Scrum
- Since 2017: Applying Scrum
- March 2018: Certified Professional Scrum Master I (https://www.scrum.org/user/325127)

- A person who enjoy nature, teaching, studying and improvement.

How I grade

- D : Below Expectation
- C : Minimum Competences
- B : Expected Competences
- A : Beyond Expectation

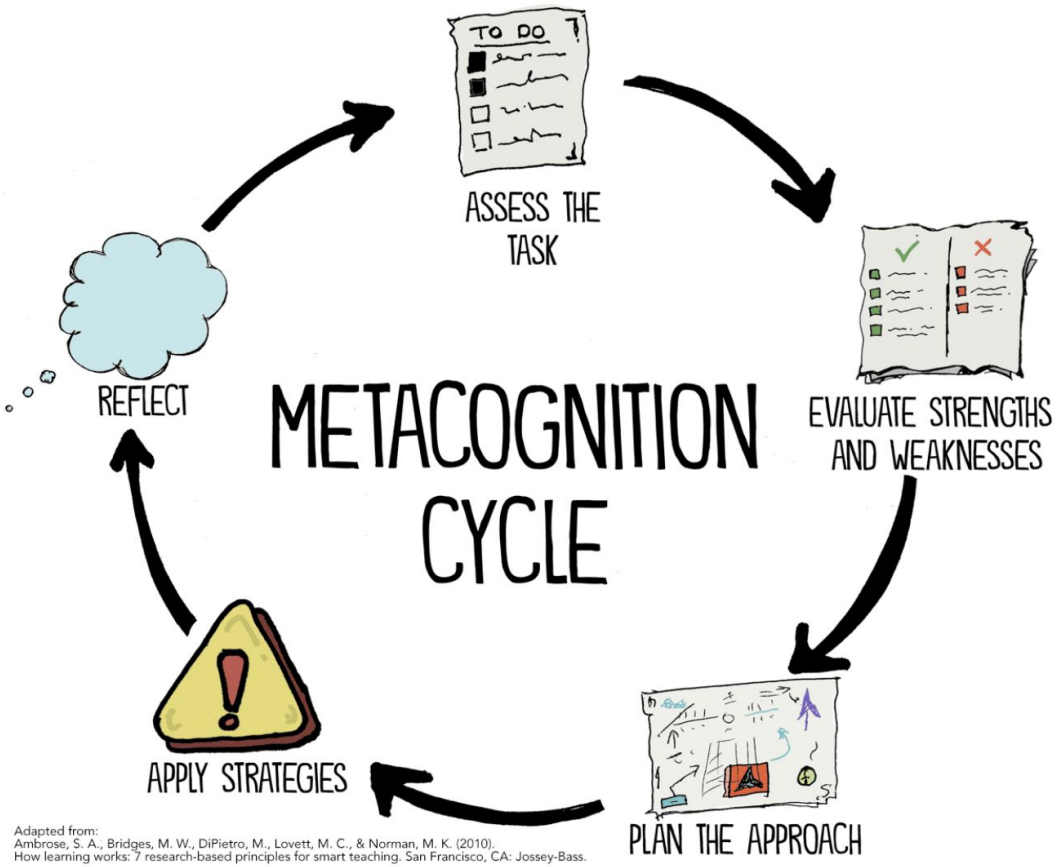


How I Teach

- traditional lecturing combined with weekly practical exercises
- flexible with evaluation and assessment
- enforce students to read and prepare themselves before class
- encourage students to be on time, disciplines, keep improving themselves and respect for people. These will help shape the professionalism for the students for their future.

Metacognition: Thinking about thinking

- If you really want to learn, and you want to learn more quickly and more deeply,
  - Pay attention to how you pay attention.
  - Think about how you think.
  - Learn how you learn.
- The trick is to get your brain to see the new material you're learning as:
  - Really Important.
  - Crucial to your well-being.
  - As important as a tiger.
- Otherwise, you're in for a constant battle, with your brain doing its best to keep the new content from sticking.



Adapted from:
Ambrose, S. A., Bridges, M. W., DiPietro, M., Lovett, M. C., & Norman, M. K. (2010). How learning works: 7 research-based principles for smart teaching. San Francisco, CA: Jossey-Bass.

Kegiatan Belajar

- Beban Belajar (dalam semester genap regular – 15 pekan)

  (Menurut SK, sks pemelajaran 1 sks setara 45 jam pemelajaran)

  - **45 jam / 15 pekan = 3 jam per pekan untuk 1 sks**
  - **4 sks = 12 Jam per pekan**

- Alokasi **12 jam per pekan**
  - **4** jam → Perkuliahan, Senin, Rabu
  - **2** jam → Praktikum bersama asdos setiap Jum'at.
  - **6** jam → Tugas, belajar dan latihan secara mandiri

Alokasikan waktu minimal 12 jam per pekan untuk DDP2 Semester Genap
Bila masih kesulitan mengikuti materi,
jangan sungkan bertanya ke tutor atau dosen.

Skema Penilaian (dapat diubah oleh dosen di akhir nanti)

- 10% Praktikum (Setiap Jum'at)
- 10% Kuis
- 10% Tugas pemrograman (ada 4, dievaluasi tiap pekan)
- 30% UTS (programming)
- 40% UAS (programming)
- 5% Bonus: Kewenangan dosen dalam memberikan kepada yang dianggap layak. Salah satu acuan misalnya adalah prestasi, latihan pekanan atau achievement di codewars setelah UTS dan UAS nya cukup baik, atau disiplin kehadiran peserta atau hal lain yang positif dari peserta kuliah.

Agenda dan Teks book

- ❑ Bahasa Pemrograman Object Oriented Java
- ❑ Unit Test
- ❑ Java Bytecode, Virtual Machine, JRE, JDK
- ❑ Compiling and running Java Source Code
- ❑ Brief Coding Guide
- ❑ Debugger, Programming Error

- ❑ Baca Buku Acuan: Introduction to Java: Programming and Data Structures
- ❑ atau https://greenteapress.com/thinkjava7/html/

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS ILMU KOMPUTER

What is Programming?

What is

Dasar-Dasar Pemrograman

DDP1: Belajar naik sepeda

- Komunikasi dengan komputer
- Penyelesaian masalah dengan komputer
- Ketrampilan dasar pemrograman

DDP2: Rutin bersepeda

- Menggunakan cara dan konsep konsep tambahan.
- Bisa untuk skala yang lebih besar
- Memodelkan permasalahan nyata
- Menerapkan bestpractice

SDA: Motor kecil

- Mempelajari teknik yang dapat membuat programming lebih efisien
- Kualitas program yang lebih baik/cepat

PBP: Mountain Bike, Cross Country

- Complete web/mobile programming
- Menghubungkan dengan teknologi web/mobile
- Menggunakan framework/library, bisa dinikmati umum
- Code lebih banyak, team work

Competitive Programming

- Acrobatic one wheel bike
- Butuh latihan lama
- Ada minat dan bakat serta ketekunan ekstra
- lebih kepada untuk kepentingan lomba
- Sangat trampil bersepeda ☺️

How to learn programming

- Seperti belajar ber-sepeda…. Mencoba dan berlatih!
- Programming: Kombinasi antara pengetahuan dan ketrampilan
  - Belajar: konsep, terminologi
  - Berlatih: ketrampilan, kecepatan, kecekatan, *best practice*
- Tugas tidak dirancang untuk membebani peserta, melainkan untuk memberikan pengalaman ketrampilan kepada peserta.
- Semakin banyak berlatih, semakin trampil dan siap untuk konsep selanjutnya yang kemudian akan menyiapkan keterampilan untuk bekerja!

Learning Programming in AI Era

- AI tools sudah sampai pada level yang bisa sangat membantu profesional dalam programming
- Skill AI tools sudah melebihi basic programming skill.
- Kelak, sebagai profesional kalian akan menggunakan AI Tools ini untuk men-speed-up pekerjaan.
- Pada saat itu, skill kalian sudah relatif lebih tinggi dari AI Tools.

Pada saat ini, saat anda sedang membangun basic skill set anda, menggunakan AI tools dapat merusak upaya tersebut sehingga skill set programming anda akan sulit terbentuk!

Kesungguhan belajar!

Facebook
Full-time · 8 yrs

- **Operations Program Analytics**
  Dec 2018 - Sep 2021 · 2 yrs 10 mos
  Singapore

  Provide insights to improve advertiser experience, i
  opportunities for current ecosystem of Facebook ac

- **Systems Engineer**
  Jun 2016 - Nov 2018 · 2 yrs 6 mos
  Menlo Park

  Developed automations, machine learning models,
  efforts in Facebook content review ecosystem.

- **Systems Project Manager**
  Oct 2014 - Jun 2016 · 1 yr 9 mos
  County Dublin, Ireland

  Project managed tool and workflow improvements i
  trust and safety globally in Facebook.

- **Market Specialist, Indonesian**
  Oct 2013 - Jun 2015 · 1 yr 9 mos
  Dublin

  Use market specific knowledge, signals and insight
  its users in the Indonesian market

Term 2005/2006 - 1 | Indeks Prestasi Term : 2.60
Status Akademis *(Academic Status)*: Aktif *(Active)*

IKI10201	Pengantar Sistem Digital & Praktikum *Introduction to Digital Systems & Lab.*	4	N/A	B+	13.2	2005/2006 - 1
IKI10600	Matematika Diskret I *Discrete Mathematics I*	3	N/A	D	3	2005/2006 - 1
IKI10820	Dasar-Dasar Pemrograman *Foundations of Programming*	4	N/A	D	4	2005/2006 - 1
UUI11001	MPK Terintegrasi *Integrated Personality Development Skills*	6	N/A	B+	19.8	2005/2006 - 1
UUI11010	MPK Bahasa Inggris *English*	3	N/A	A	12	2005/2006 - 1

Term 2005/2006 - 2 | Indeks Prestasi Term : 2.29
Status Akademis *(Academic Status)*: Aktif *(Active)*

IKI10030	Fisika I *Physics I*	3	N/A	B-	8.1	2005/2006 - 2
IKI10041	Kalkulus I *Calculus I*	3	N/A	C+	6.9	2005/2006 - 2
IKI10230	Pengantar Organisasi Komputer *Introduction to Computer Organization*	3	N/A	C	6	2005/2006 - 2
IKI10610	Matematika Diskret II *Discrete Mathematics II*	3	N/A	C	6	2005/2006 - 2
IKI10830	Desain & Pemrogaman Berorientasi Objek *Object Oriented Design & Programming*	4	N/A	C	8	2005/2006 - 2
IKI20081	Kalkulus II *Calculus II*	3	N/A	C-	5.1	2005/2006 - 2
UUI12020	MPK Agama Islam *Islamic Religious Instruction*	2	N/A	B+	6.6	2005/2006 - 2
UUI12030	MPK Seni/Olahraga *Art/Sport*	1	N/A	A-	3.7	2005/2006 - 2

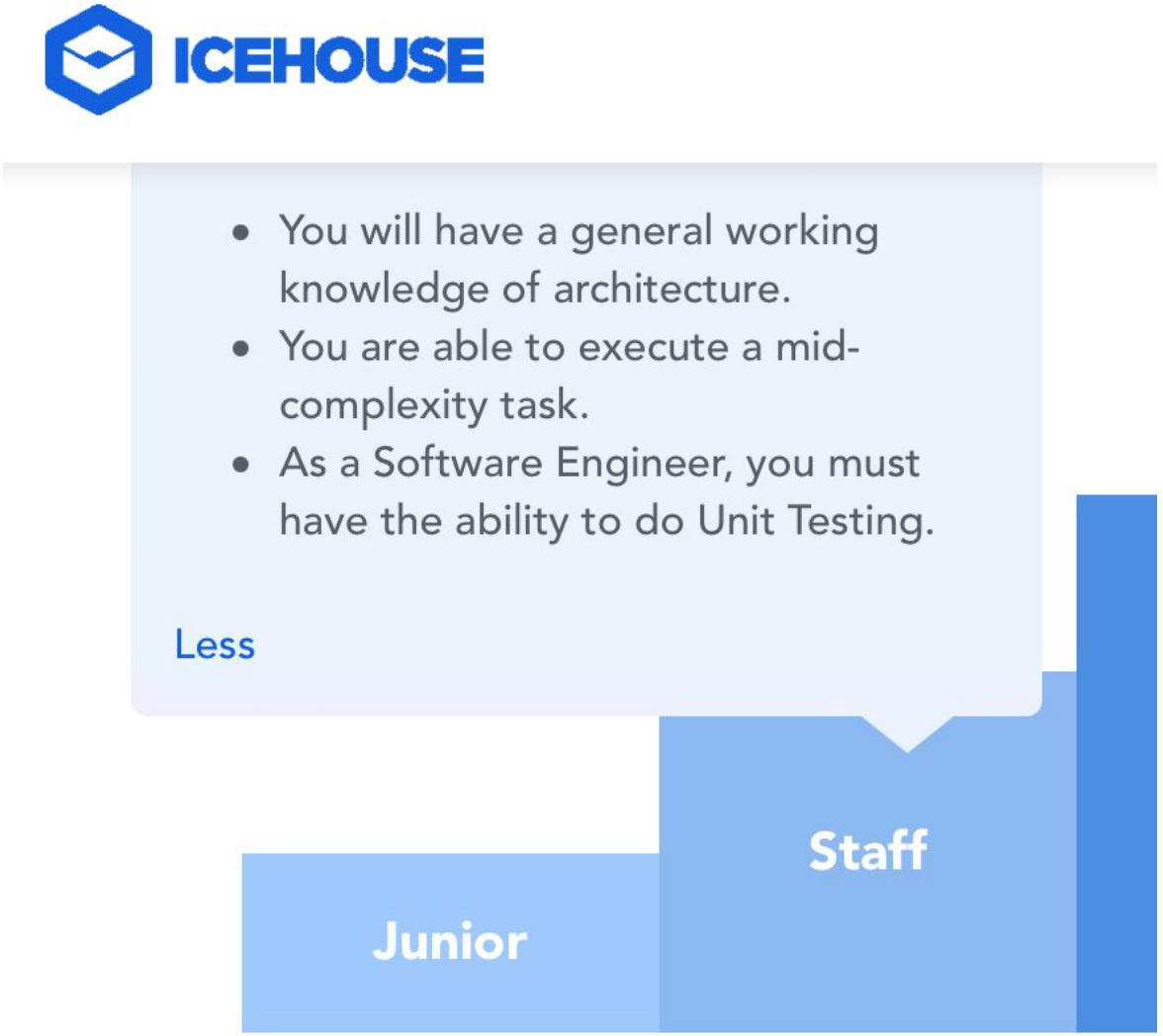
Poin-poin penting pada DDP2

- Unit Test (untuk tugas dan tutorial wajib)
- Debugger
- Git (individual, belum perlu team) - wajib untuk tugas
- Praktikum (dikerjakan dengan computer lab Fasilkom secara terisolasi)
- Kuis
- Tugas (di-push ke git repository, didemokan dan dimonitor tiap pekan)

- Banyak Berlatih, ulangi kembali bila perlu!

Unit test is a must for your future career

- Unit Test wajib digunakan.
- Unit Test relative lebih mudah dari framework apapun yang akan kalian gunakan nanti
- Isu nya hanya lebih kepada kebiasaan saja, sebagaimana membersihkan kamar setiap pagi.
- Ini perlu kita biasakan semenjak awal



Instalasi

- JDK (Java Development Kit) 21 (Bukan hanya JRE)
- Visual Studio Code
- Junit 4
- Git

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS ILMU KOMPUTER

Mari Belajar Java

Mari belajar Coding di repl

Bisa fork beberapa contoh dari: https://repl.it/@AdeAzurat/DDP2-Pekan01

dan coba coding bersama di: https://repl.it/join/uhplqbvl-adeazurat

Jangan lupa buat akun dan berlatih lah di https://www.codewars.com

dan juga simak slot kuliah di Scele!

Objectives

- To understand computer basics, programs, and operating systems (§§1.2–1.4).
- To understand the meaning of Java language specification, API, JDK, and IDE (§1.6).
- To write a simple Java program (§1.7).
- To display output on the console (§1.7).
- To explain the basic syntax of a Java program (§1.7).
- To create, compile, and run Java programs (§1.8).
- To use sound Java programming style and document programs properly (§1.9).
- To explain the differences between syntax errors, runtime errors, and logic errors (§1.10).

Objectives

- To write Java programs to perform simple computations (§2.2).
- To obtain input from the console using the **Scanner** class (§2.3).
- To use identifiers to name variables, constants, methods, and classes (§2.4).
- To use variables to store data (§§2.5–2.6).
- To program with assignment statements and assignment expressions (§2.6).
- To use constants to store permanent data (§2.7).
- To name classes, methods, variables, and constants by following their naming conventions (§2.8).
- To explore Java numeric primitive data types: **byte**, **short**, **int**, **long**, **float**, and **double** (§2.9.1).
- To read a **byte**, **short**, **int**, **long**, **float**, or **double** value from the keyboard (§2.9.2).
- To perform operations using operators **+**, **-**, *****, **/**, and **%** (§2.9.3).
- To perform exponent operations using **Math.pow(a, b)** (§2.9.4).
- To write integer literals, floating-point literals, and literals in scientific notation (§2.10).
- To write and evaluate numeric expressions (§2.11).
- To obtain the current system time using **System.currentTimeMillis()** (§2.12).
- To use augmented assignment operators (§2.13).
- To distinguish between postincrement and preincrement and between postdecrement and predecrement (§2.14).
- To cast the value of one type to another type (§2.15).
- To describe the software development process and apply it to develop the loan payment program (§2.16).
- To write a program that converts a large amount of money into smaller units (§2.17).
- To avoid common errors and pitfalls in elementary programming (§2.18).

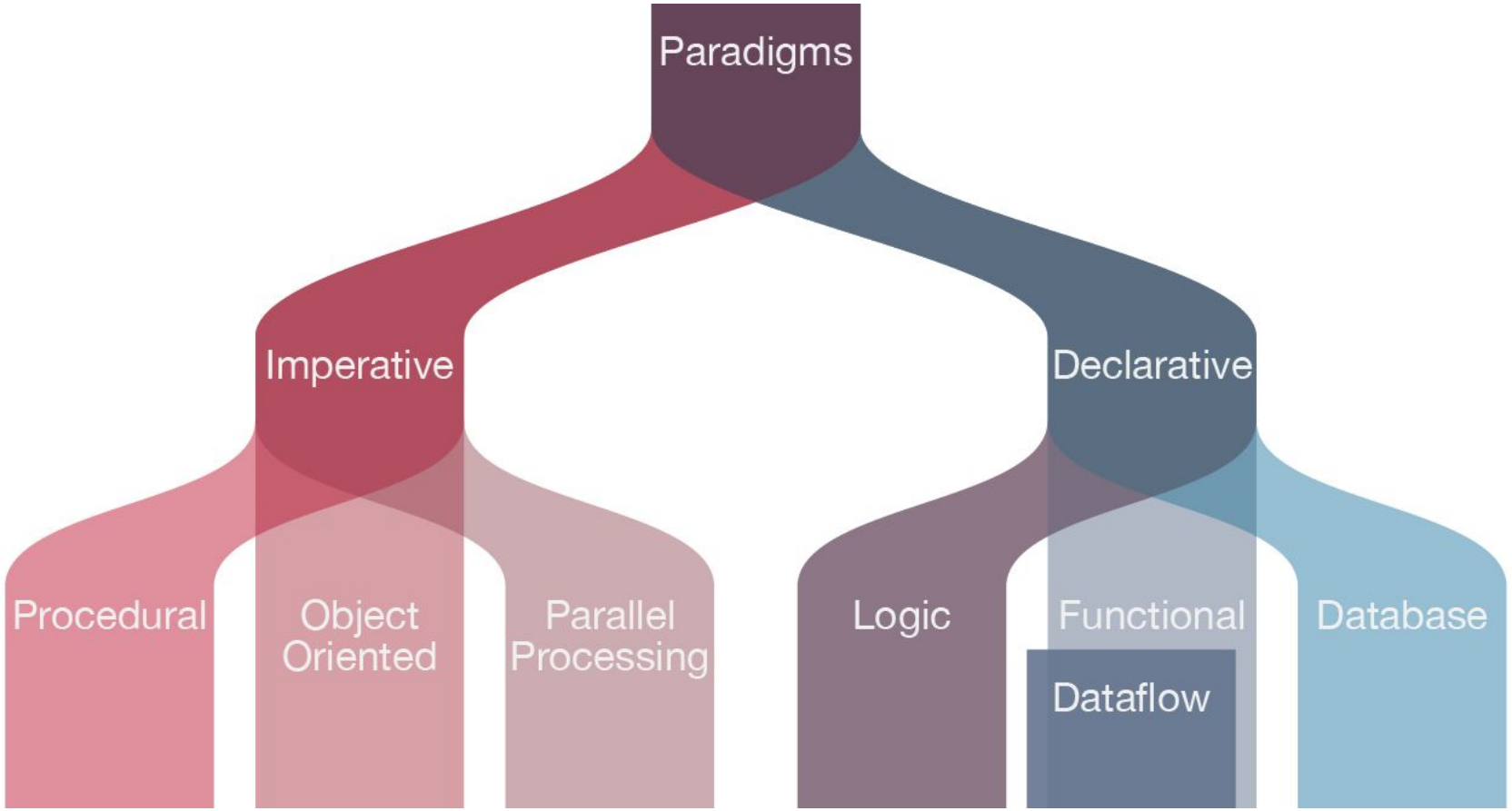
History of Java

- The first object oriented programming language was Simula-67
  - designed to allow programmers to write simulation programs
- In the early 1980's, Smalltalk was developed at Xerox PARC
  - New syntax, large open-source library of reusable code, bytecode, platform independence, garbage collection.
- late 1980's, C++ was developed by B. Stroustrup,
  - Recognized the advantages of OO but also recognized that there were tremendous numbers of C programmers
- In 1991, engineers at Sun Microsystems started a project to design a language that could be used in consumer 'smart devices': Oak
  - When the Internet gained popularity, Sun saw an opportunity to exploit the technology.
  - The new language, renamed Java, was formally presented in 1995 at the SunWorld '95 conference.

Object Oriented Language

- Bahasa Pemrograman Java adalah salah satu dari bahasa pemrogramanan berorientasi obyek (OO)
- Paradigma OO ini sudah muncul sejak 1967 dengan bahasa bernama Simula.
- Namun baru tahun 90an, OO mulai populer bersamaan dengan mulai muncul nya bahasa pemrograman Java, yang hingga kini masih menjadi salah satu bahasa populer didunia.

Taxonomy Paradigma Pemrograman

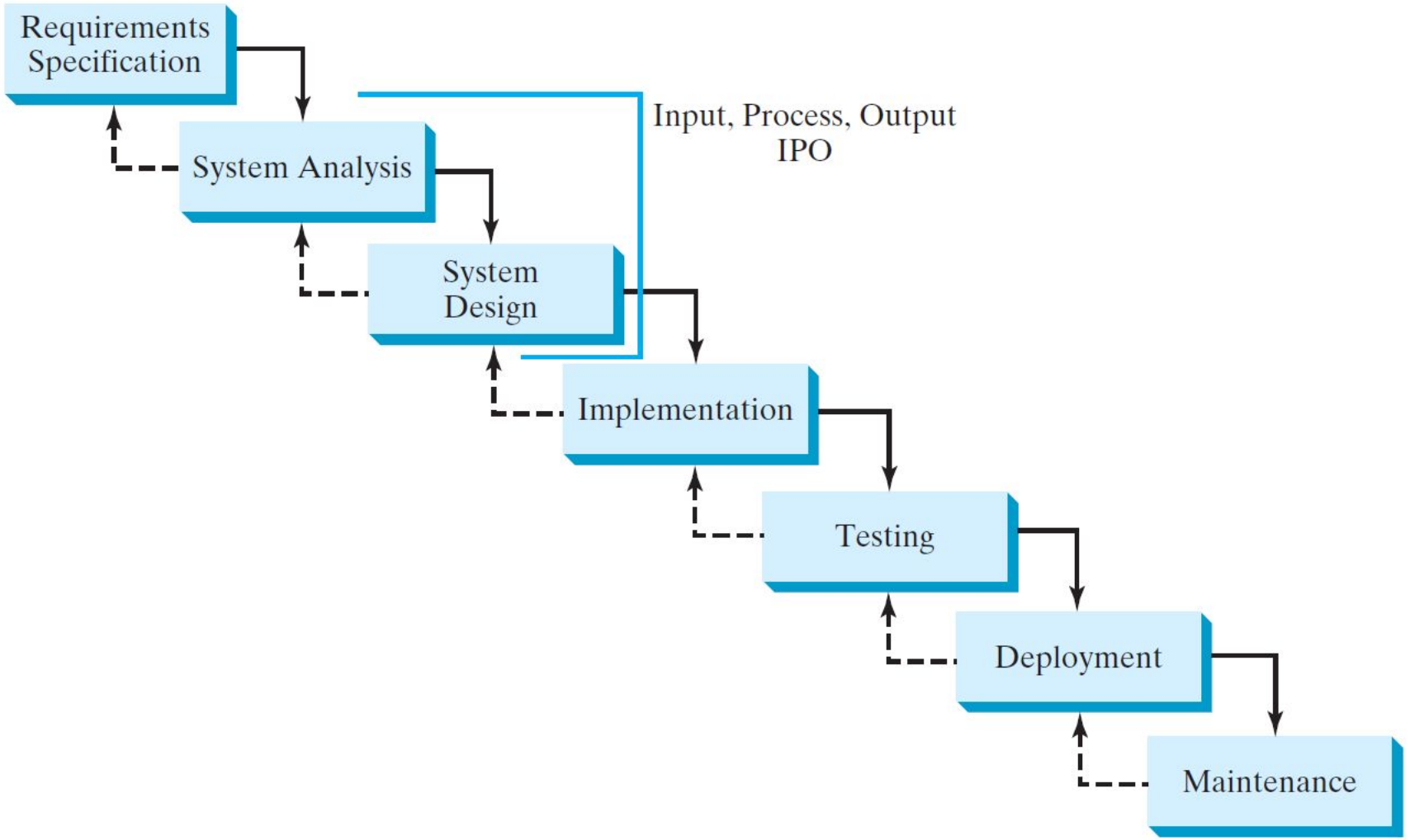


Sumber: https://www.typesnuses.com/types-of-programming-languages-with-differences/

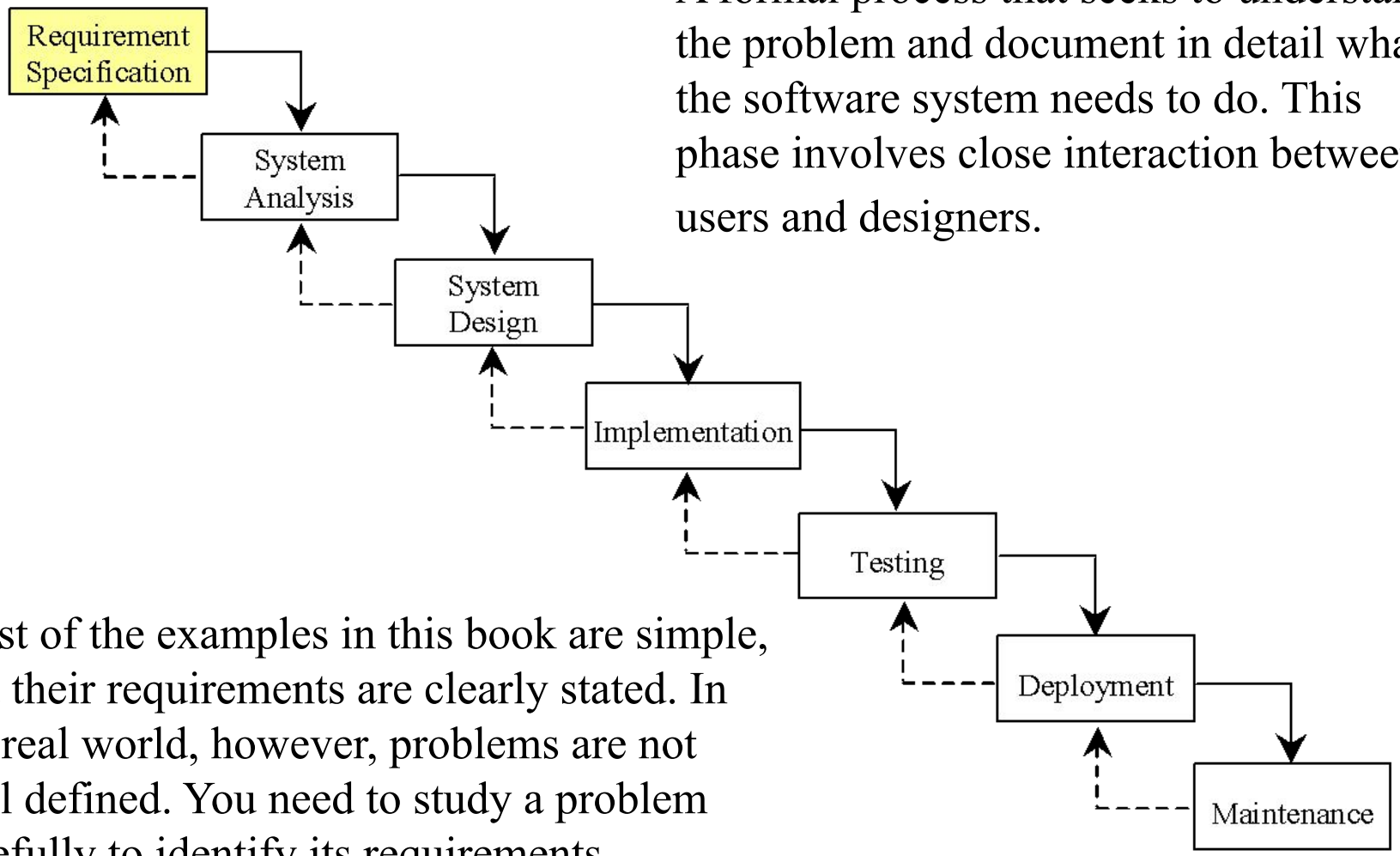
Pemrograman Berorientasi Obyek

- Tahun 1980an, terjadi yang disebut software crisis
- Hal ini terjadi karena tingkat kebutuhan software yang tinggi dan kompleksitas dan baris code yang semakin sulit untuk dipahami oleh programmer
- Salah satu pendekatan yang membantu memudahkan programmer dalam merancang dan memahami aplikasi skala besar dengan meng-abstraksikan kompleksitas dan keterhubungan code dalam program adalah dengan menggunakan paradigma ber-***orientasi obyek***.

Software Development Process



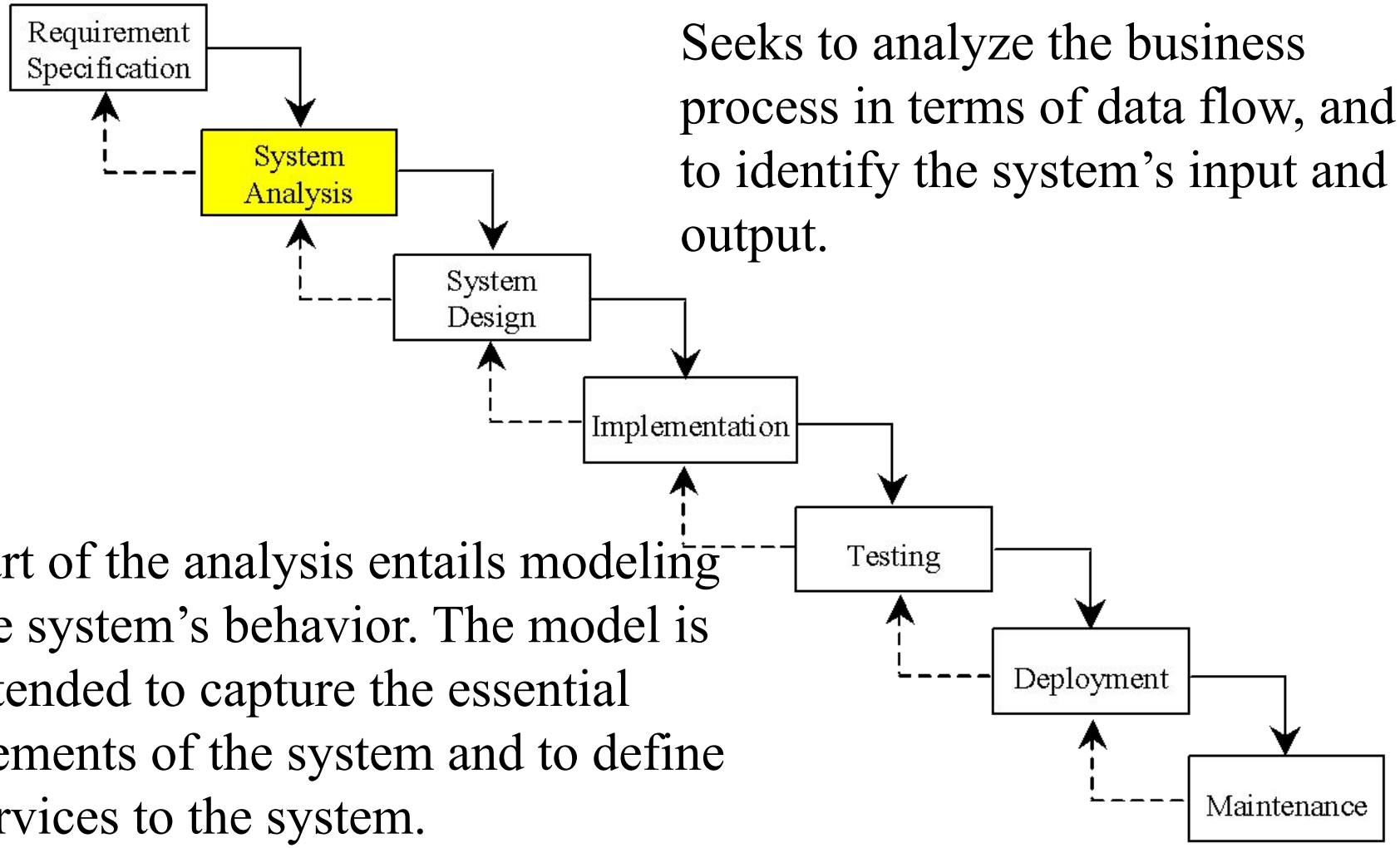
Requirement Specification



A formal process that seeks to understand the problem and document in detail what the software system needs to do. This phase involves close interaction between users and designers.

Most of the examples in this book are simple, and their requirements are clearly stated. In the real world, however, problems are not well defined. You need to study a problem carefully to identify its requirements.

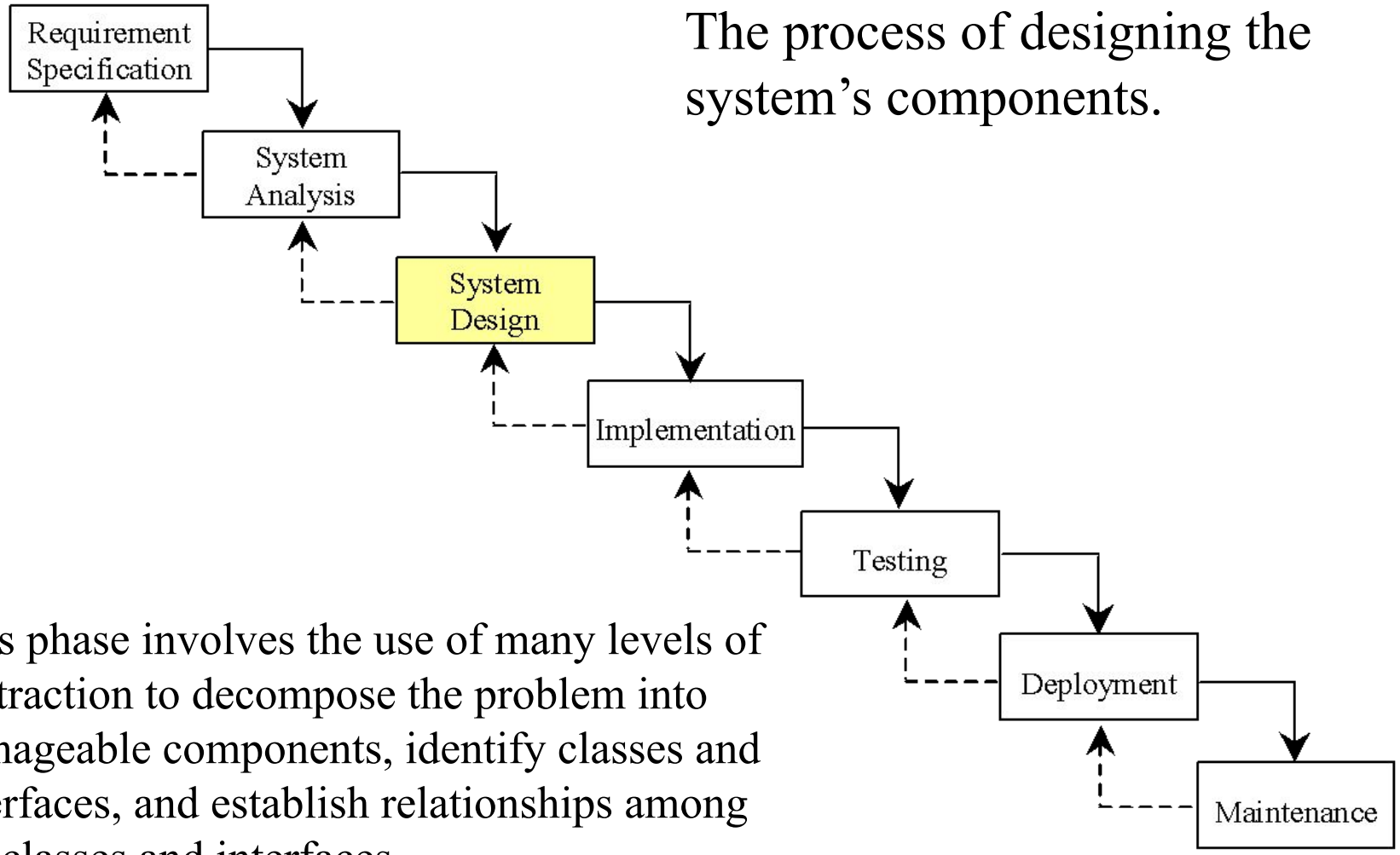
System Analysis



Seeks to analyze the business process in terms of data flow, and to identify the system's input and output.

Part of the analysis entails modeling the system's behavior. The model is intended to capture the essential elements of the system and to define services to the system.

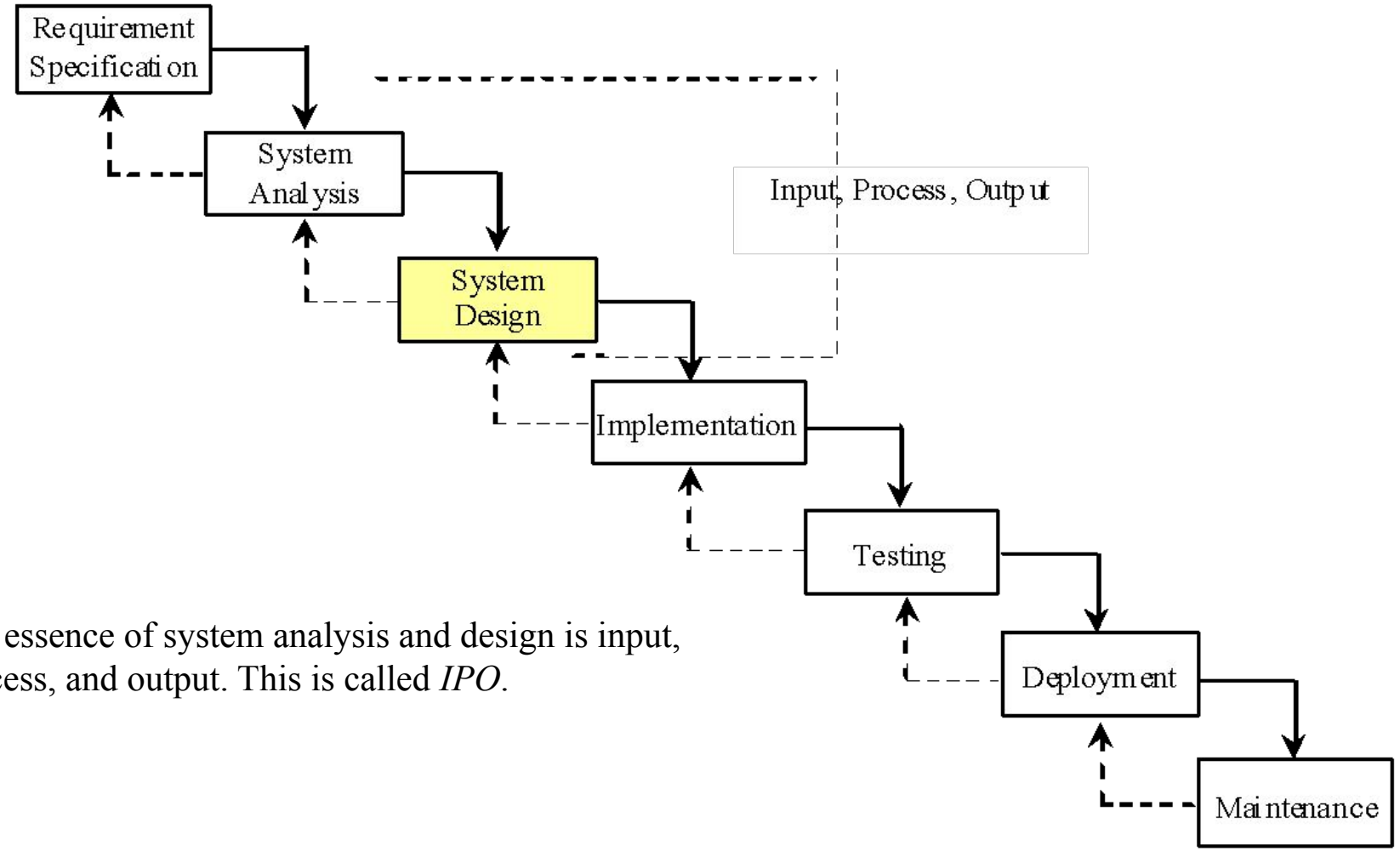
System Design



The process of designing the system's components.

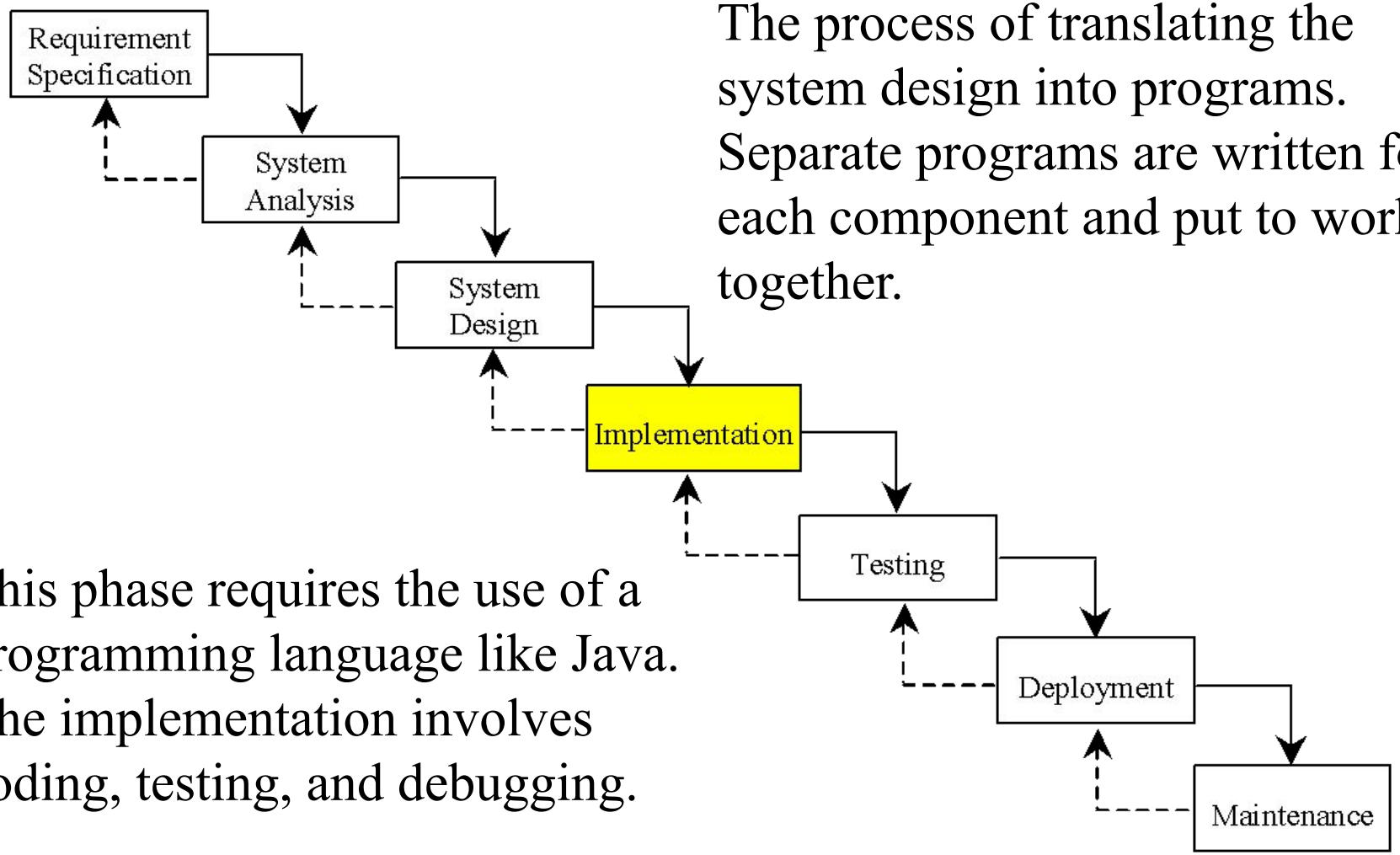
This phase involves the use of many levels of abstraction to decompose the problem into manageable components, identify classes and interfaces, and establish relationships among the classes and interfaces.

IPO



The essence of system analysis and design is input, process, and output. This is called *IPO*.

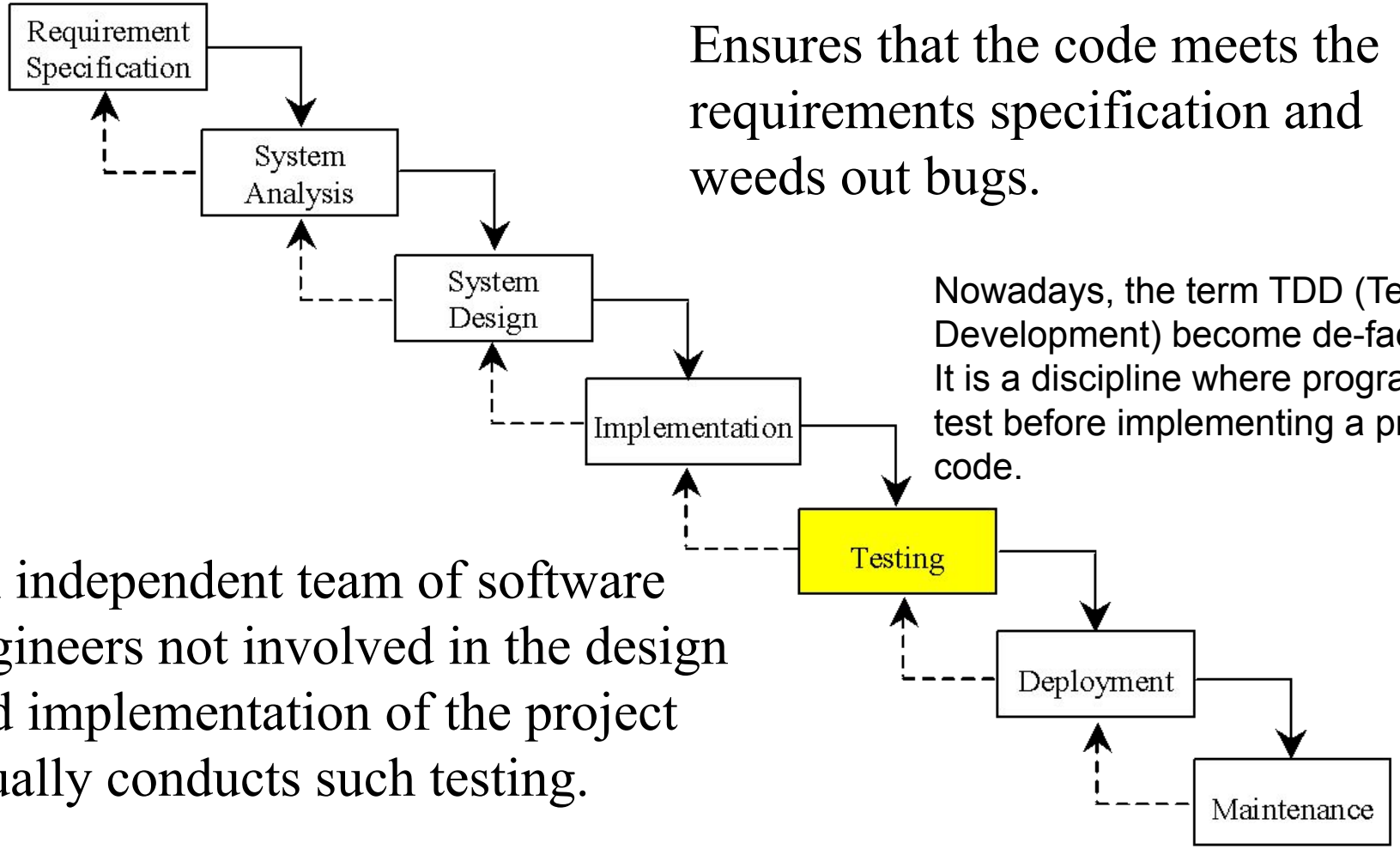
Implementation



The process of translating the system design into programs. Separate programs are written for each component and put to work together.

This phase requires the use of a programming language like Java. The implementation involves coding, testing, and debugging.

Testing

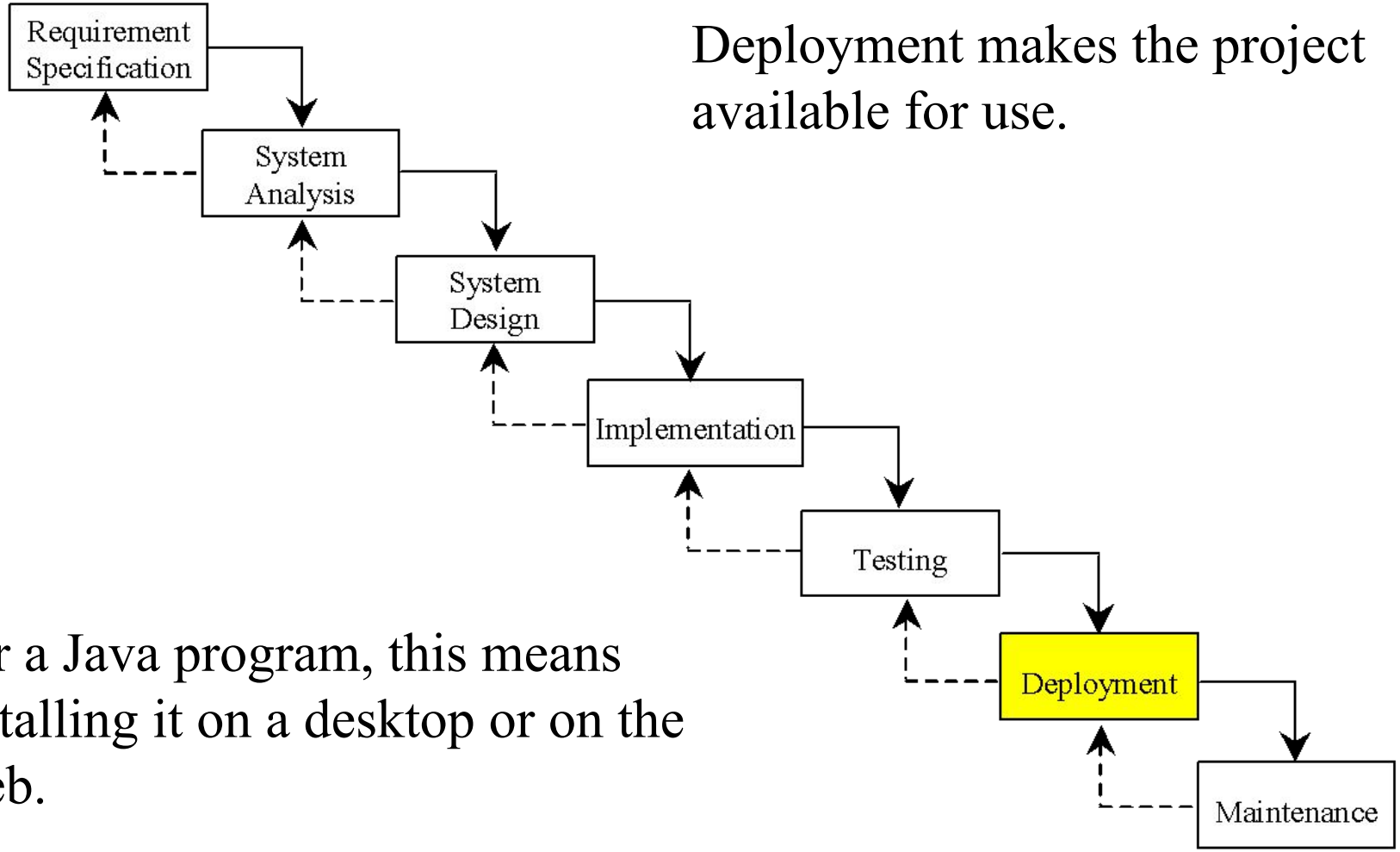


Ensures that the code meets the requirements specification and weeds out bugs.

Nowadays, the term TDD (Test Driven Development) become de-facto best-practice. It is a discipline where programmer provide unit test before implementing a production source code.

An independent team of software engineers not involved in the design and implementation of the project usually conducts such testing.

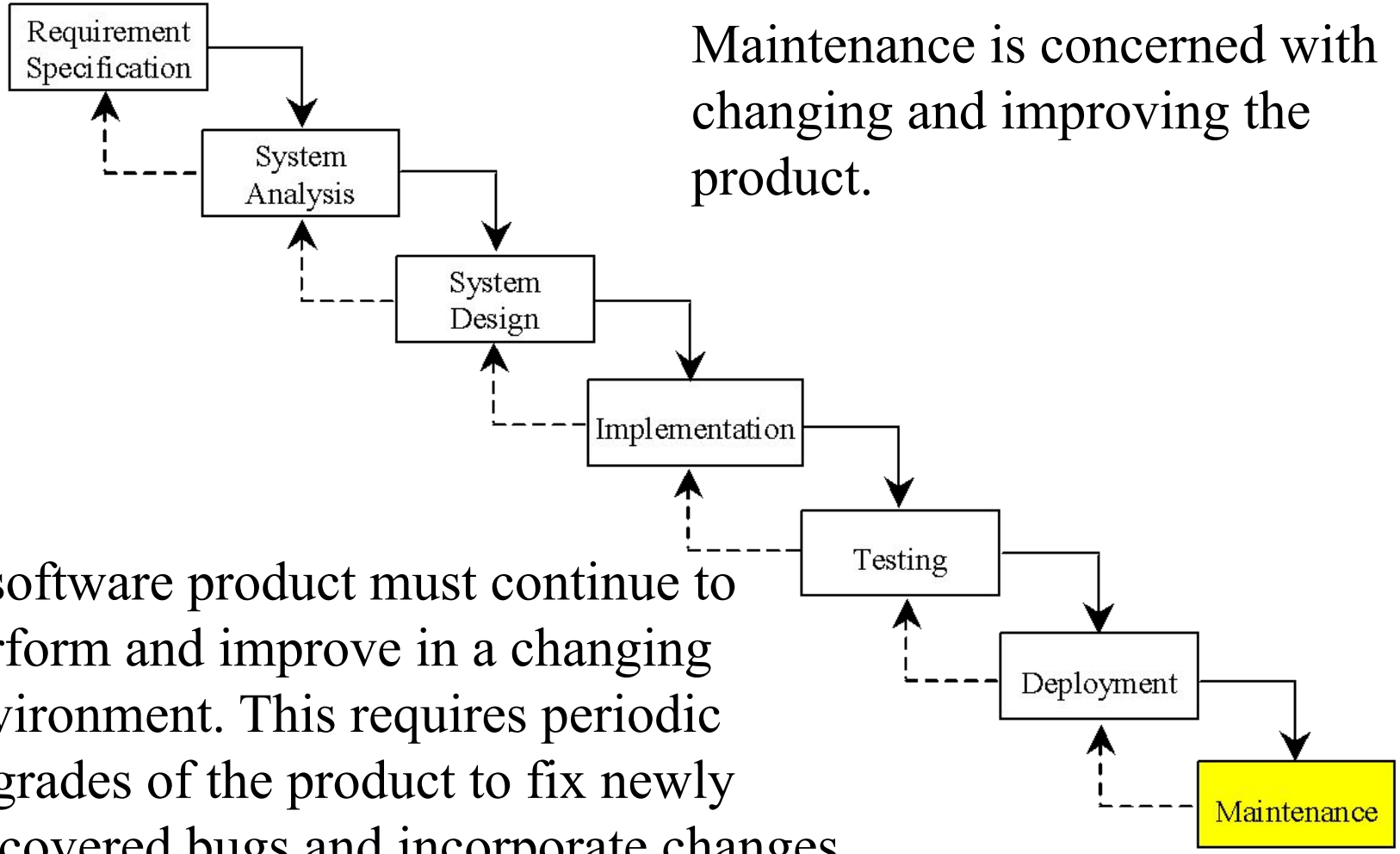
Deployment



Deployment makes the project available for use.

For a Java program, this means installing it on a desktop or on the Web.

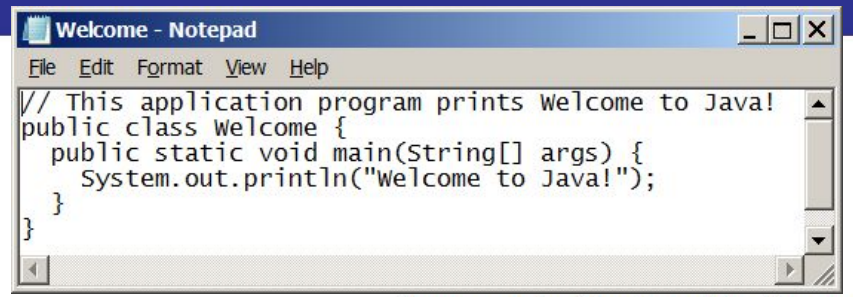
Maintenance

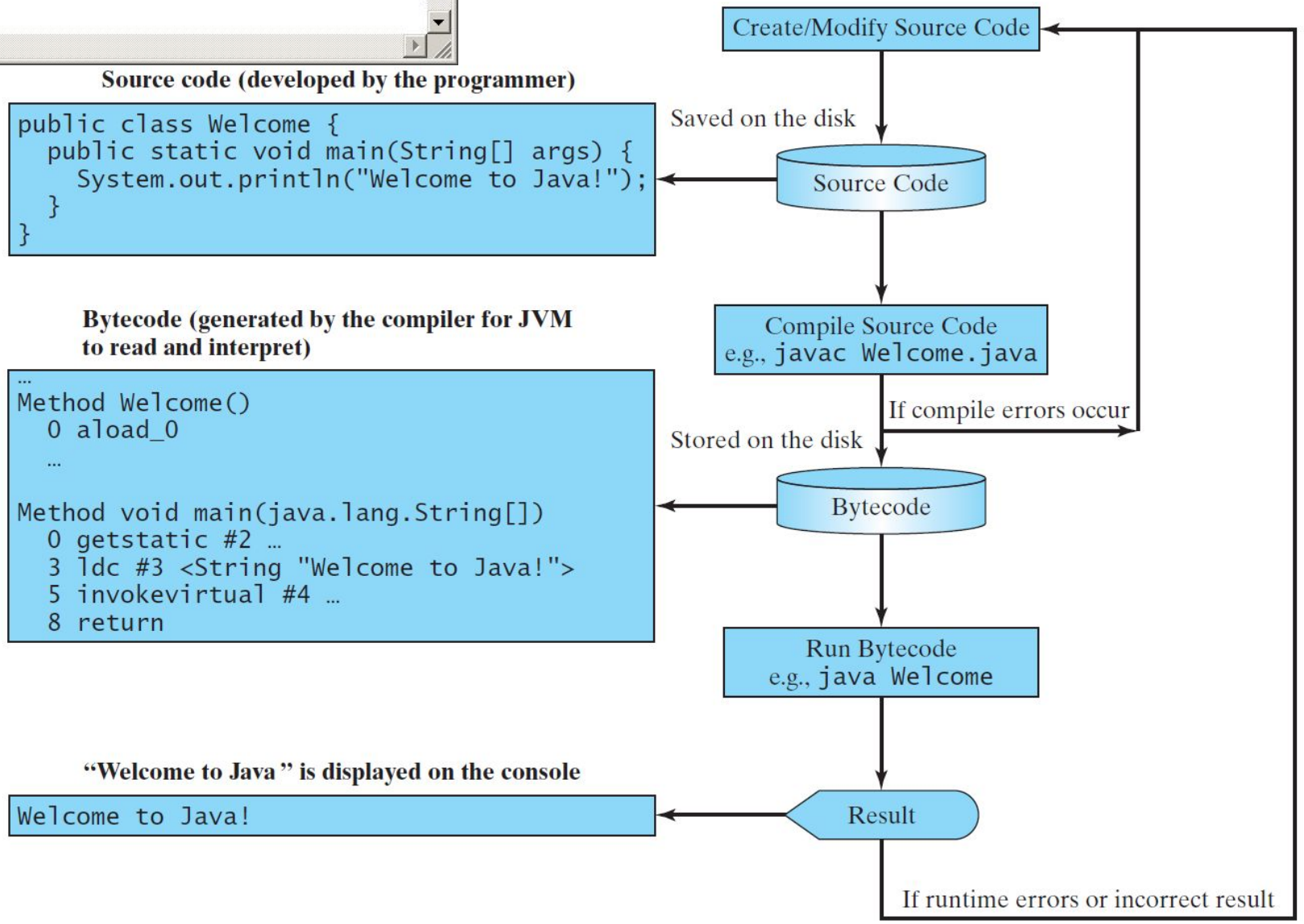


Maintenance is concerned with changing and improving the product.

A software product must continue to perform and improve in a changing environment. This requires periodic upgrades of the product to fix newly discovered bugs and incorporate changes.

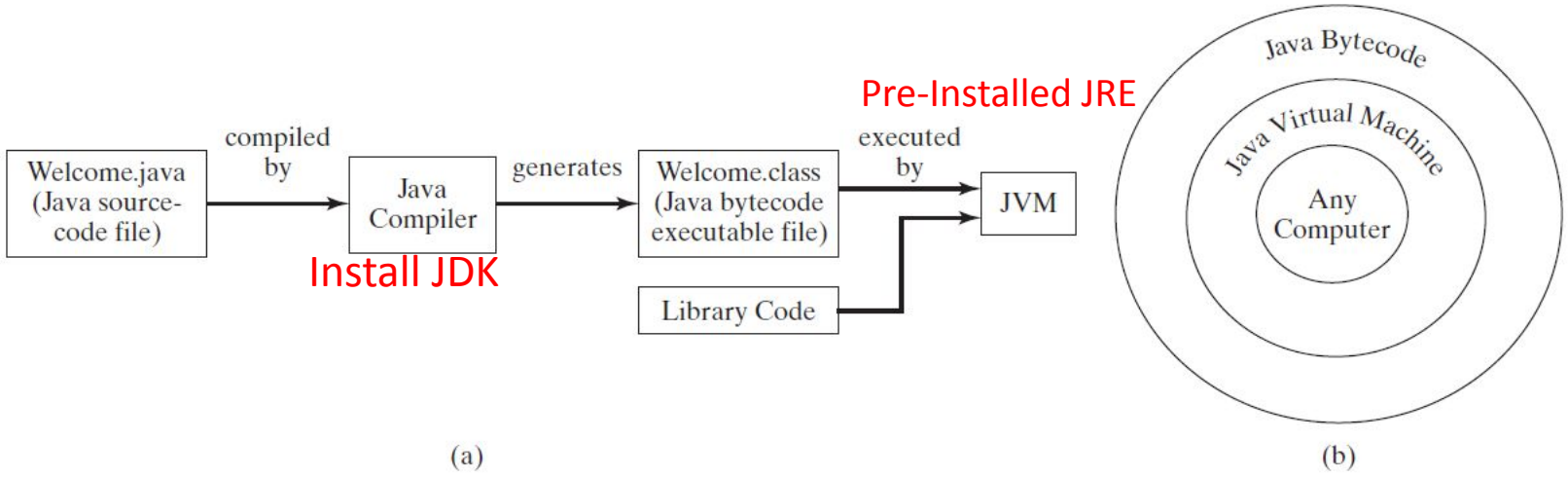
Creating, Compiling, and Running Programs





Compiling Java Source Code

You can port a source program to any machine with appropriate compilers. The source program must be recompiled, however, because the object program can only run on a specific machine. Nowadays computers are networked to work together. Java was designed to run object programs on any platform. With Java, you write the program once, and compile the source program into a special type of object code, known as *bytecode*. The bytecode can then run on any computer with a Java Virtual Machine, as shown below. Java Virtual Machine is a software that interprets Java bytecode.



UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

Selamat Berlatih!

Perhatikan lagi List Objective yang perlu dikuasai pekan ini.
Baca buku acuan dan berlatih!
Bila masih belum yakin tanyakan ke dosen atau tutor.

Semangat !