Introduction to Software Engineering
Part I

Jin Hyun Kim

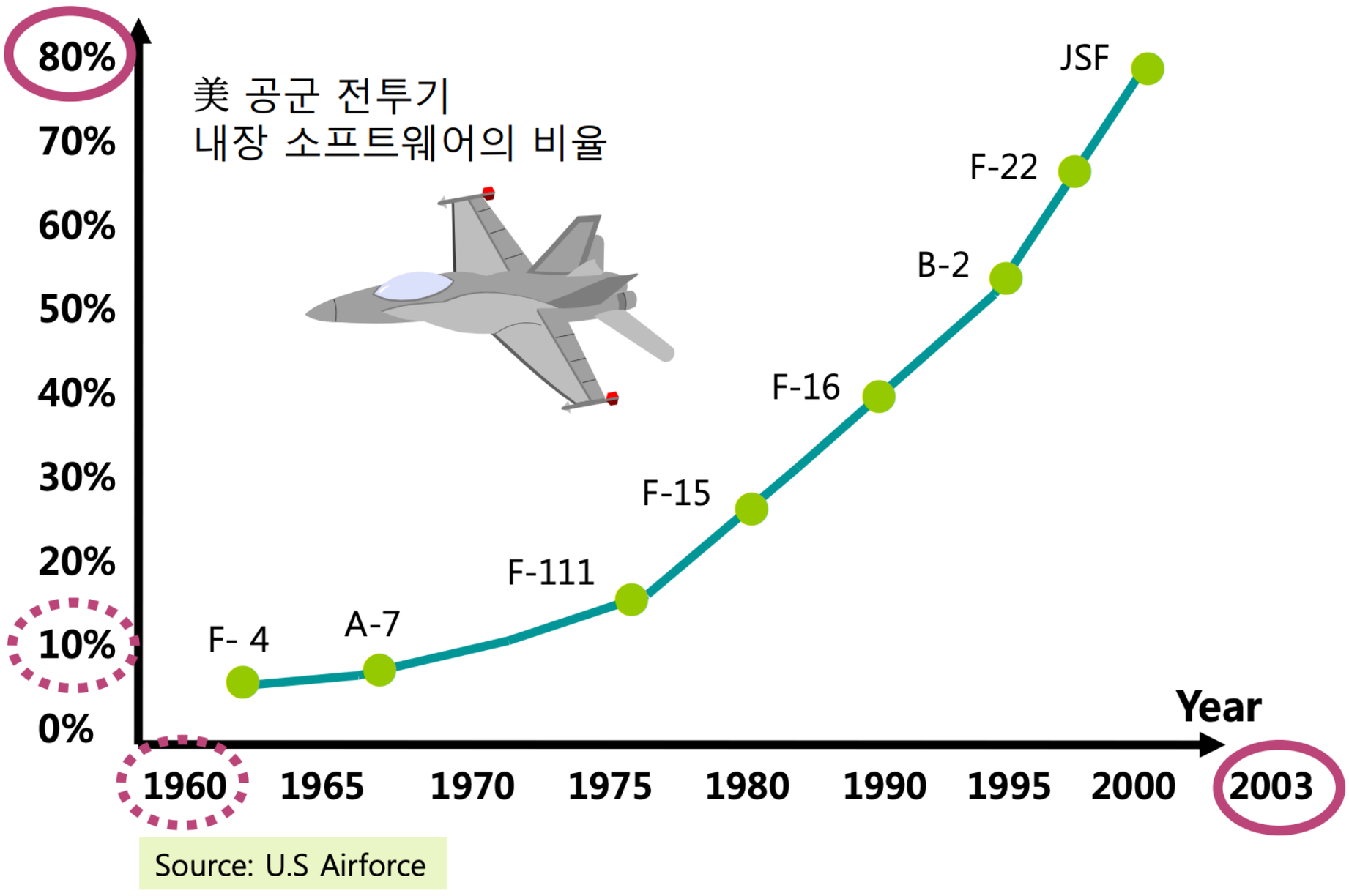
2020

References

- Software Engineering- Chap 1, 2- Ian Sommerville

Everywhere Software

Aircraft



Source: U.S Airforce

What is Software?

“Computer programs and associated documentation.”

- Sommovile

“A collection of data or computer instructions that tell the computer how to work.”

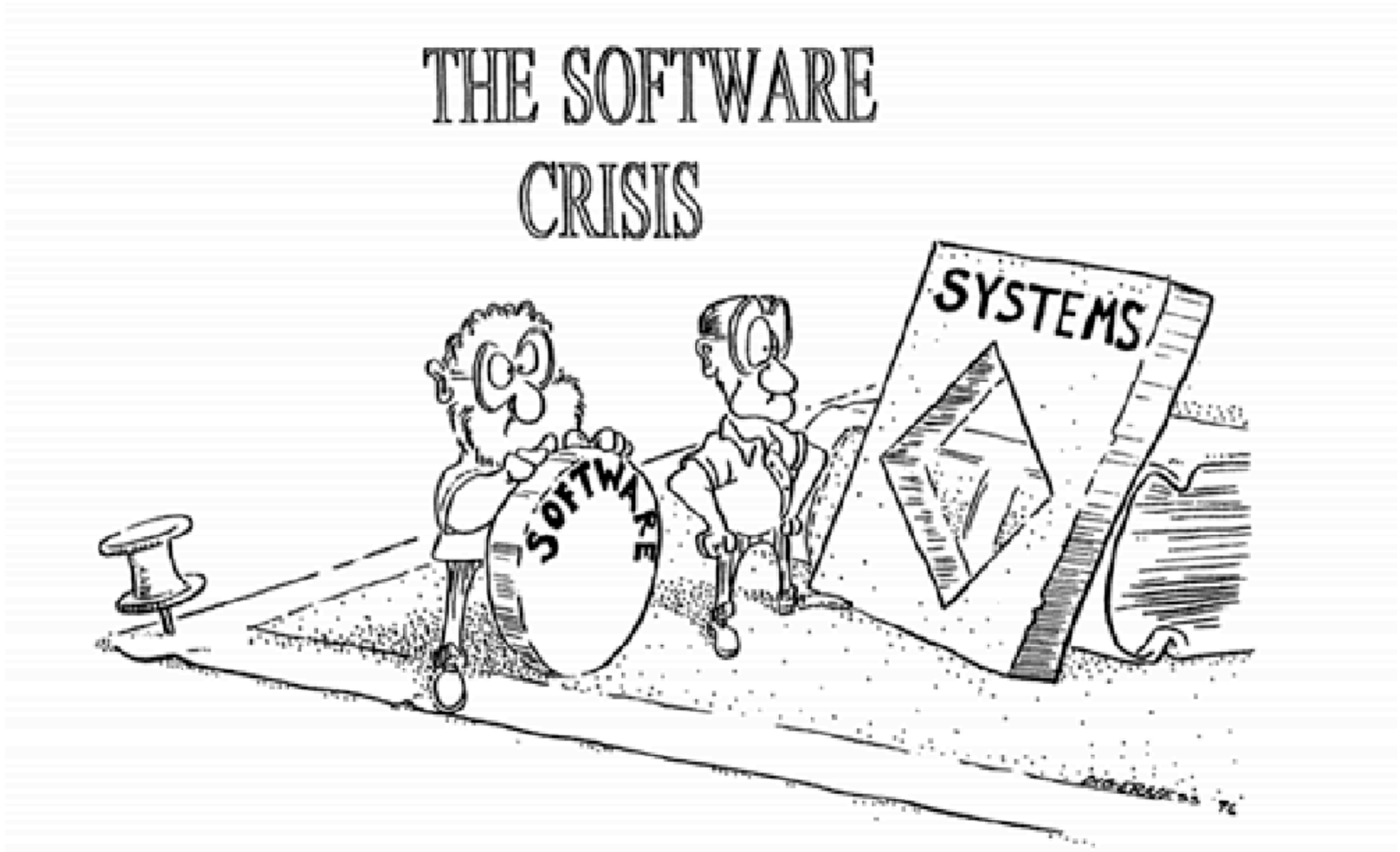
-Wiki

What is Software Engineering?

- Software engineering is concerned with **theories (principle)**, **methods** and **tools (techniques)** for professional software development.

- Software engineering is an engineering discipline 공학분야 that is concerned with all aspects of software production.

Why Software Engineering?



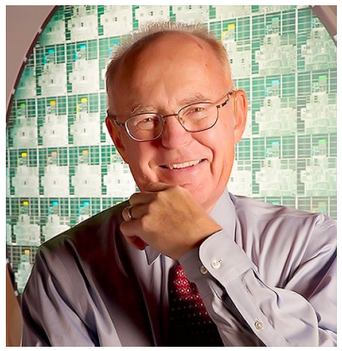
Software Crisis

- The term "software crisis" was coined by some attendees at the first NATO Software Engineering Conference in 1968 at Garmisch, Germany.

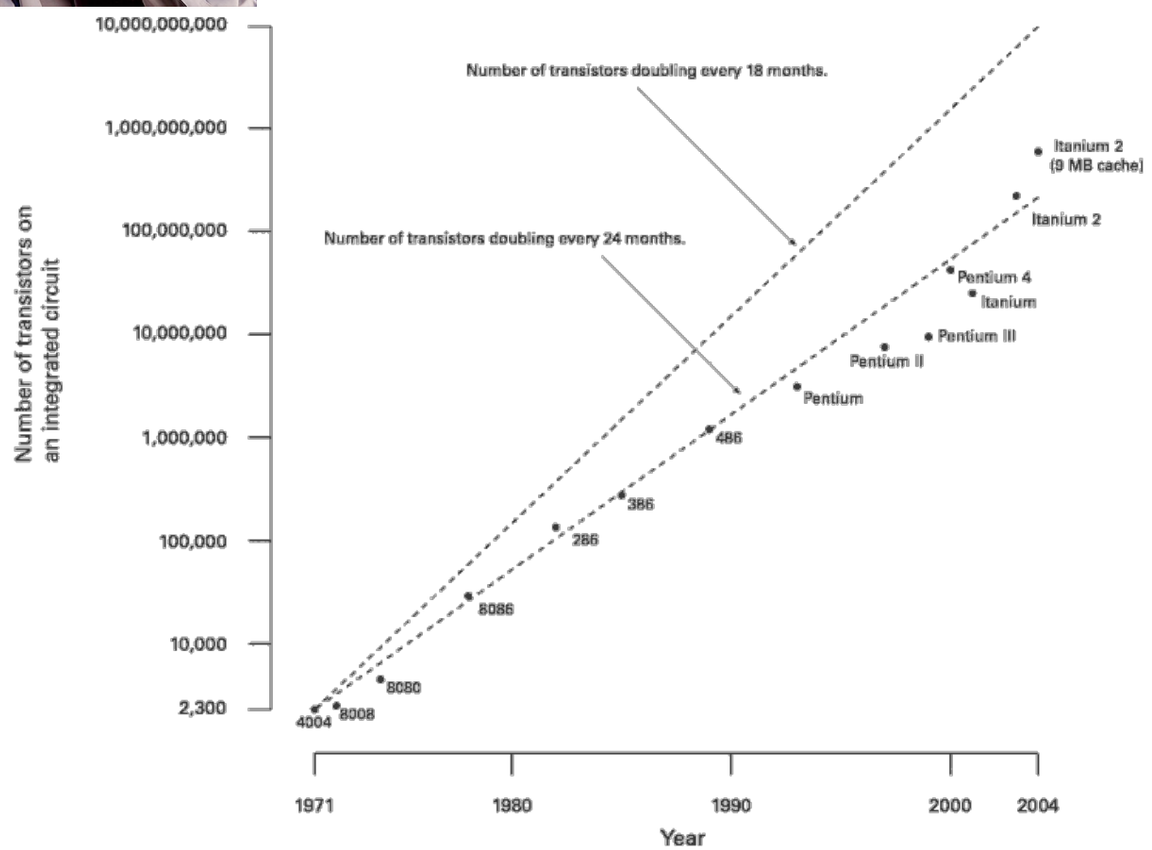
- Edsger Dijkstra's 1972 ACM Turing Award Lecture makes reference to this same problem:

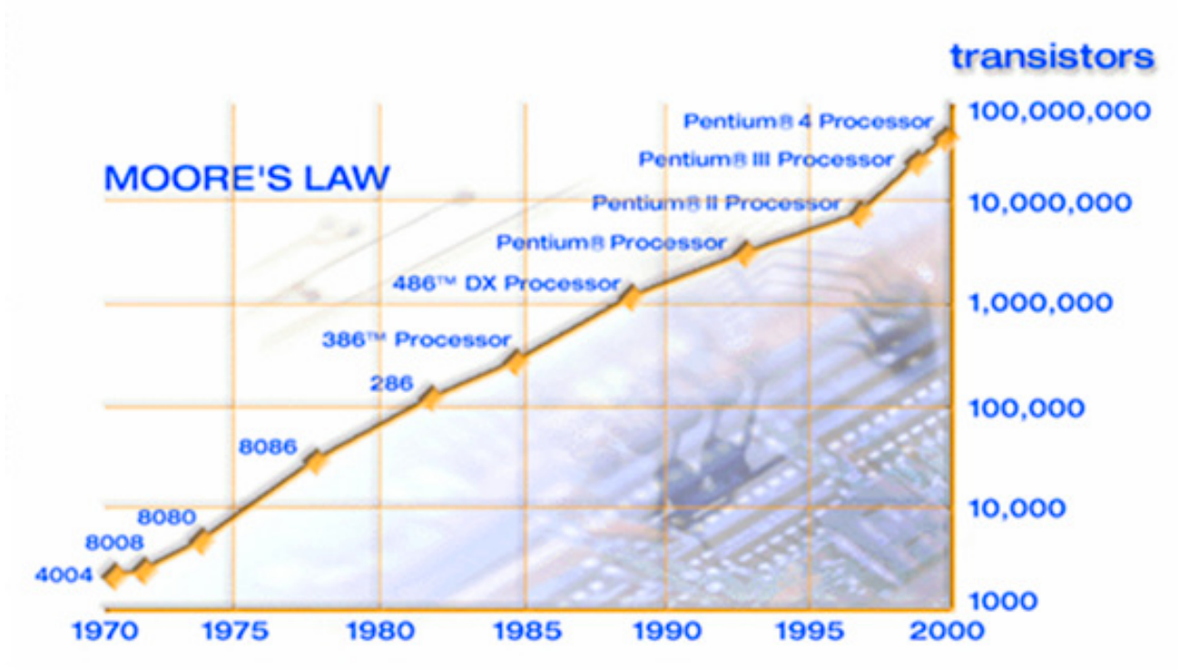
- The major cause of the software crisis is that the machines have become several orders of magnitude more powerful! To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now **we have gigantic computers, programming has become an equally gigantic problem**. —Edsger Dijkstra, The Humble Programmer (EWD340), Communications of the ACM

Moore’s Law



The Co-Founder of Intel,
Gordon Moore 고든 무어



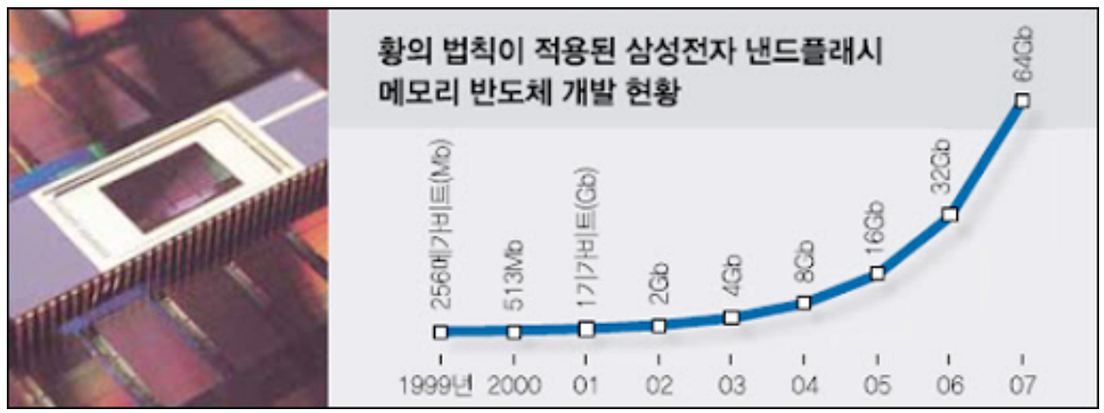


▶ 1989년 인텔 486 : 120만 개
▶ 1993년 인텔 펜티엄 : 310만 개
▶ 1995년 인텔 펜티엄 프로 : 550만 개
▶ 2001년 인텔 펜티엄4 프로세서 : 4천200만 개
▶ 2004년 인텔 펜티엄4 HT 프로세서 : 1억 2천500만 개
▶ 2012년 인텔 3세대 코어 i5 프로세서(아이비브리지) : 10억 개
▶ 2015년 인텔 5세대 코어M 프로세서(브로드웰) : 13억

https://m.blog.naver.com/PostView.nhn?blogId=heungmusoft&logNo=220706021071&proxyReferer=https%3A%2F%2Fwww.google.com%2F

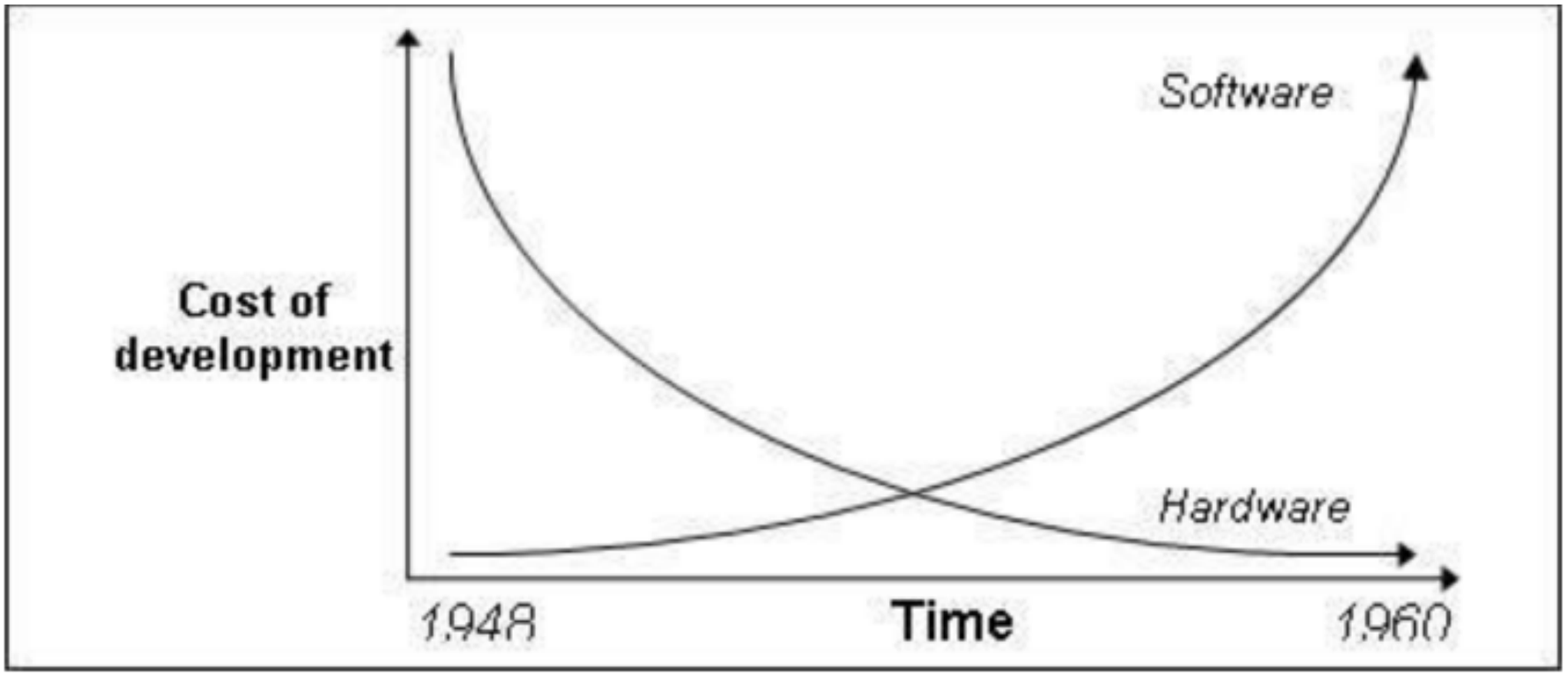
Hwang’s Law

전 삼성전자 반도체부문 CEO
황창규,



- “메모리 반도체의 집적도는 매년 2배로 증가한다’는 반도체 신성장 이론”
- 황창규 삼성전자 반도체총괄 사장이 2002년 2월 국제반도체회로학술회의(ISSCC)에서 주창
- 미국 인텔사의 고든 무어 전 회장의 ‘1년 반마다 2배로 증가한다’는 ‘무어의 법칙’을 깬 것으로 유명

SW Development Costs



SW Development Cost

- Software **costs** often dominate computer system costs.
    - The costs of software on a PC are often greater than the hardware cost.
- Software **costs** more to **maintain** than it does to develop.
    - For systems with a long life, maintenance costs may be several times than development costs.
- Software engineering is concerned with cost-effective software development.

Software Failure

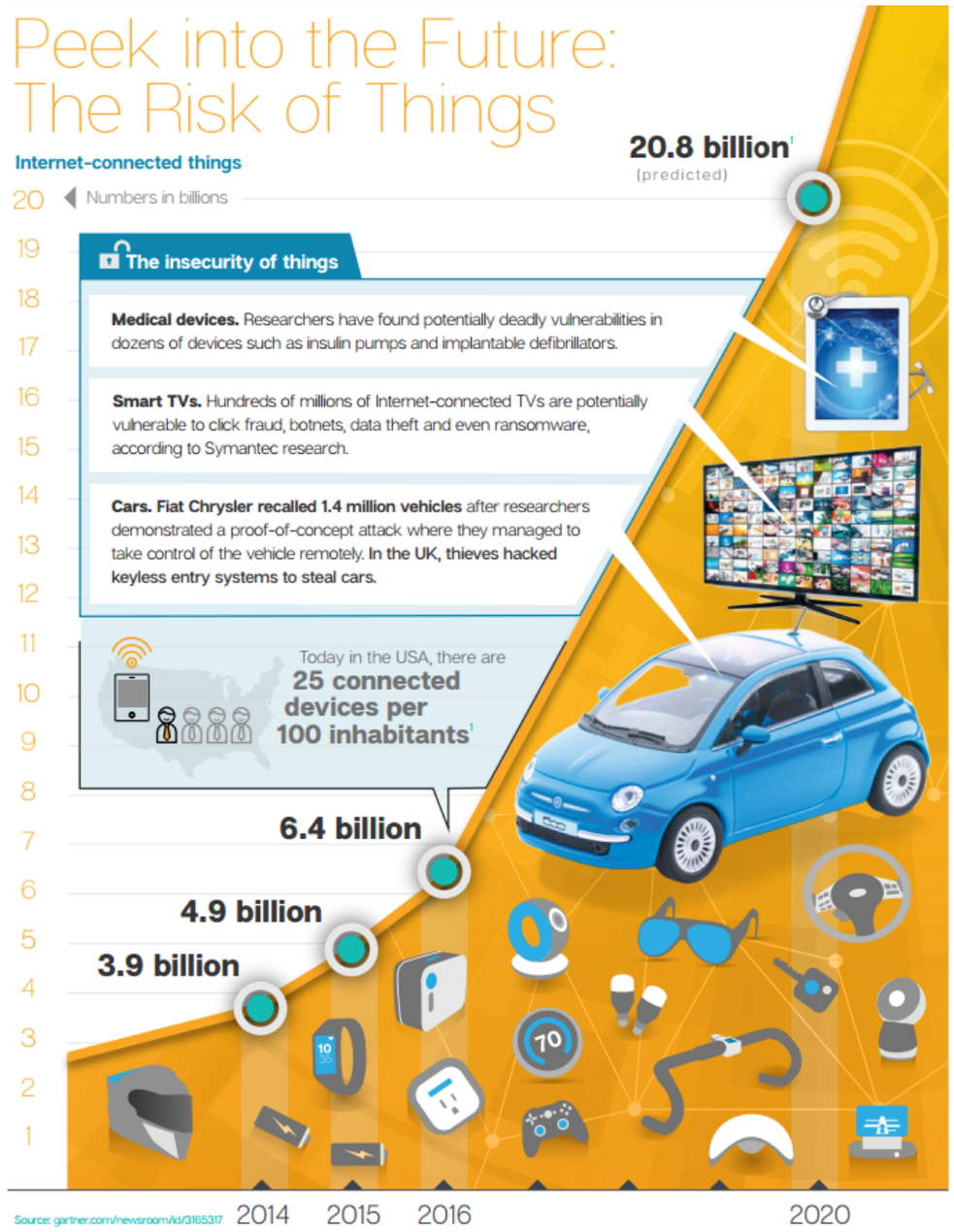
- Denver baggage handling system ($300M)
- Power blackout in NY (2003)
- Ariane 5 (1996)
- Mars Pathfinder (1997)
- Mars Climate Orbiter ($125M,1999)
- The Patriot Missile (1991)
- USS Yorktown (1998)
- Therac-25 (1985-1988)
- London Ambulance System (£9M, 1992)
- Pacemakers (500K recalls during 1990-2000)
- Numerous computer-related incidents with commercial ai (https://www.fss.aero/accident-reports/browse_type.php? type=operator)

Security

46% *The world population is connected to the internet*

500K *Attacks every minutes*

20.8B *Devices that will be under cyber attack*



Key Concepts

- Good Software
- What are Key Challenges SE faces?
- Fundamental SW Engineering
- What are the best software engineering techniques and methods?

Good Software

- Good software should deliver the **required functionality** and **performance** to the user and should be **maintainable, dependable** and **usable.**

What are Key Challenges?

- Coping with increasing **diversity**, demands for **reduced delivery times** and developing **trustworthy** software.

Fundamental SW Engieering

- Software **specification**, software **development**, software **validation (verification)** and software **evolution**.

What are the best software engineering techniques and methods?

- While all software projects have to be professionally managed and developed, **different techniques** are **appropriate** for **different types of system**.

- For example, games should always be developed using a series of prototypes whereas safety critical control systems require a complete and analyzable specification to be developed. You can’t, therefore, say that one method is better than another.

Essential Software Properties

Product characteristic	Description
Maintainability	Software should be written in such a way so that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment.
Dependability and security	Software dependability includes a range of characteristics including reliability, security and safety. **Dependable software should not cause physical or economic damage** in the event of system failure. Malicious users should not be able to access or damage the system.
Efficiency	Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilisation, etc.
Acceptability	Software must be acceptable to the type of users for which it is designed. This means that it must be understandable, usable and compatible with other systems that they use.

Software Engineering

- **Theory (Principle)** + **Methods** + **Tools (Techniques)**
  - **E.g. Object-oriented development, Model-based development**

Why Software Engineering?

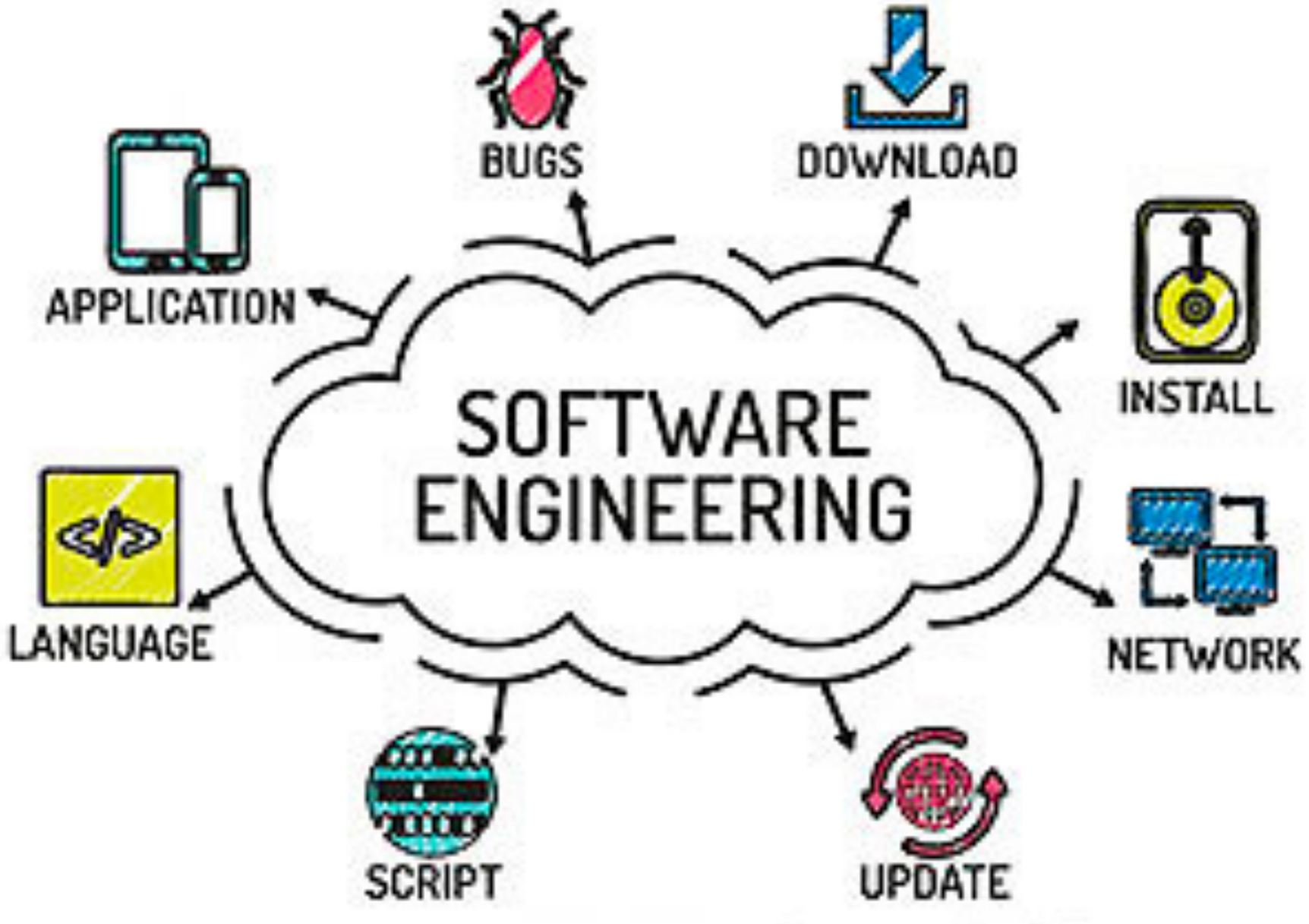
- **Software Crisis (’60)**

- Complexity, Dependability, Costs UP, UP, UP!!!

- Safety, Security, Reliability, Maintainability, Efficiency DOWN, DOWN, DOWN!!!

In next

- Software Process and Activities



Introduction to Software Engineering
Part II

Jin Hyun Kim

2020

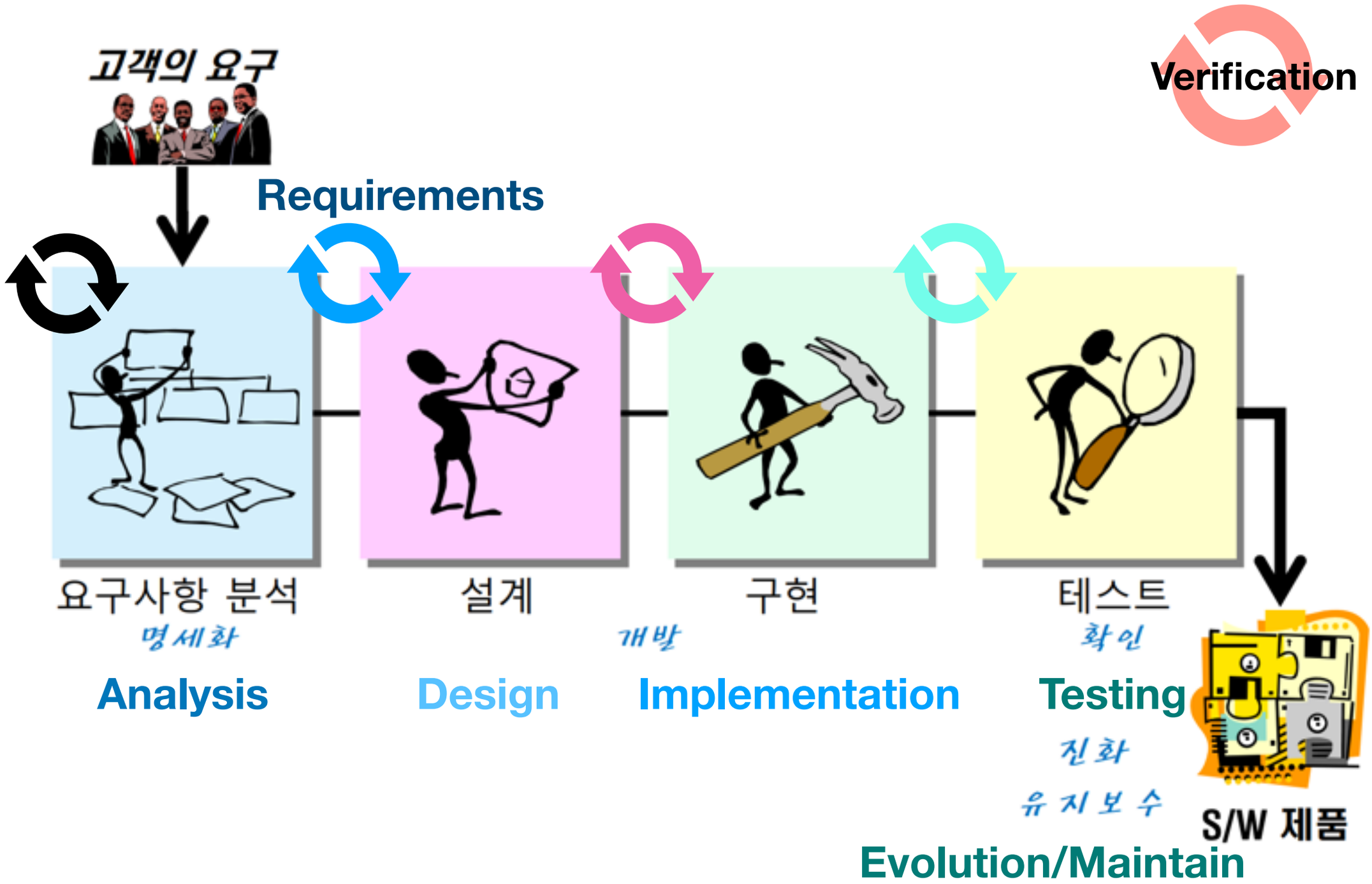
In this Class

- Software Life Cycle
- Software Development Process Models
    - Waterfall, Incremental, Reuse-Oriented, and V-Model
- Requirements, Design, Implementation, Testing, …
- S

References

- Software Engineering- Chap 1, 2- Ian Sommerville

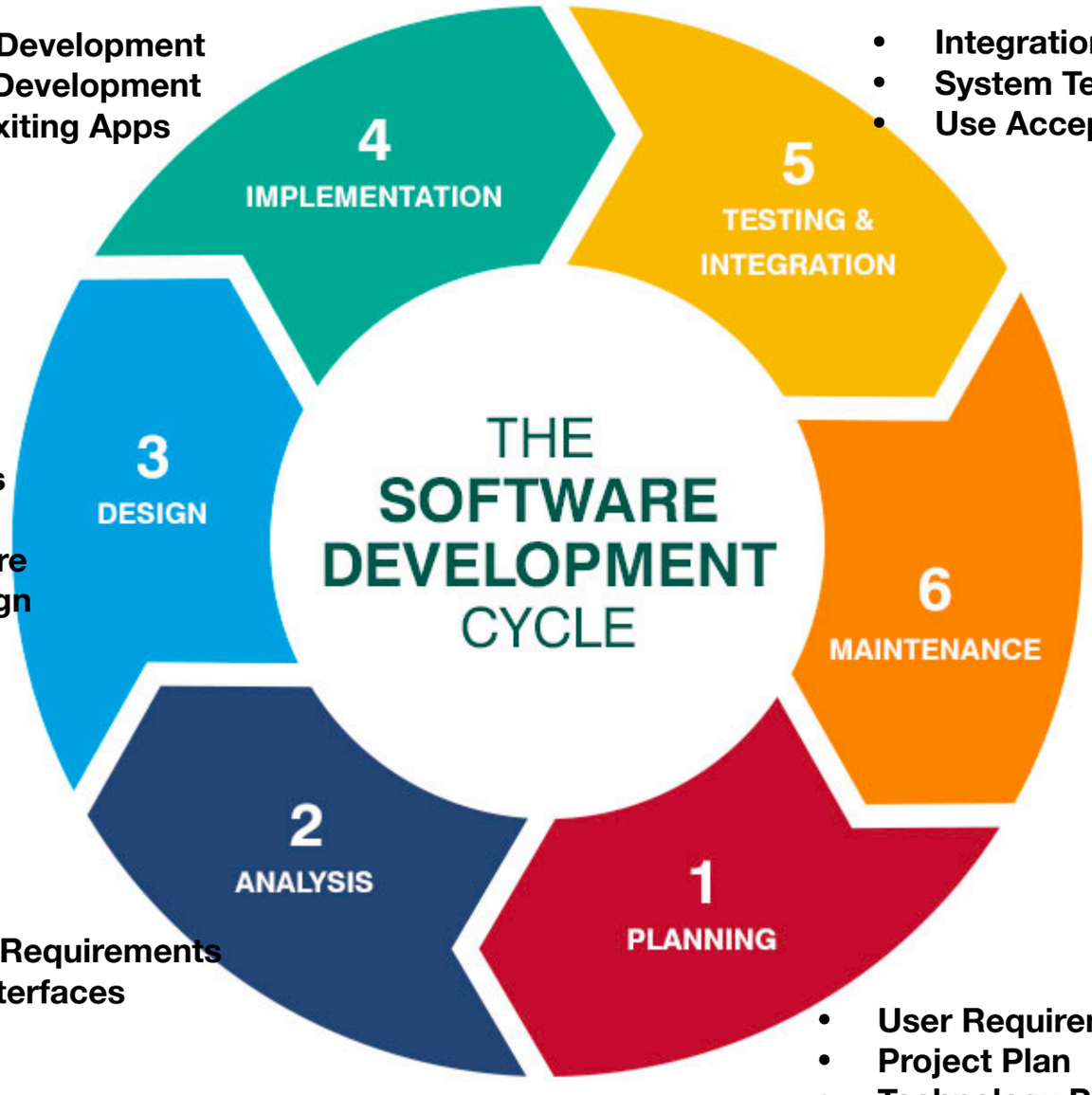
SW Life Cycle



SW Development Lifecycle

- Application Code Development
- System Interface Development
- Integration with Exiting Apps
- Unit Testing

- Integration Testing
- System Testing
- Use Acceptance Testing



- Detailed specifications
- Finalize User Interface
- Application Architecture
- System Interface Design
- Test Plans

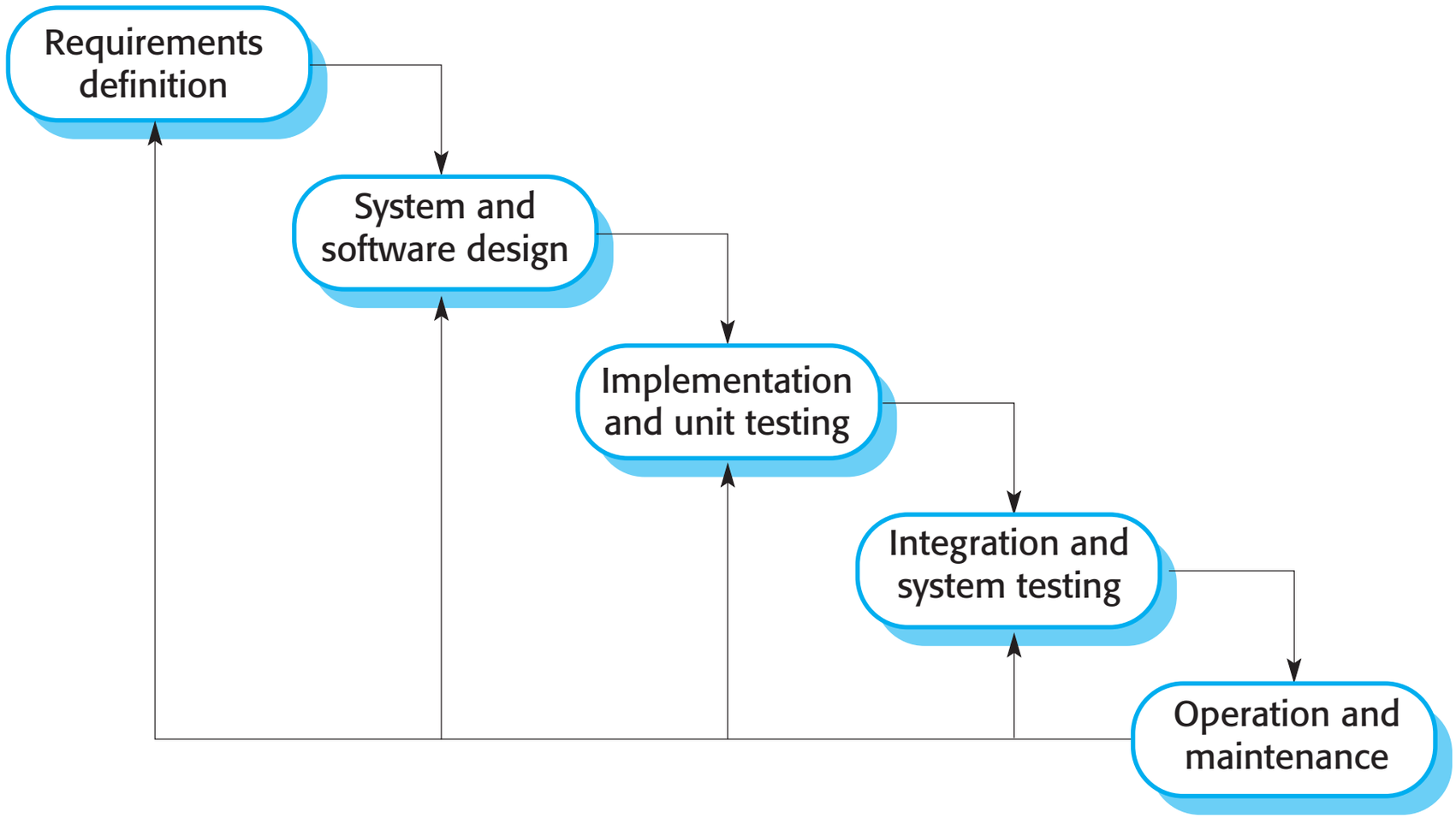
- Functionality Requirements
- Initial User Interfaces

- User Requirements
- Project Plan
- Technology Platform Selection

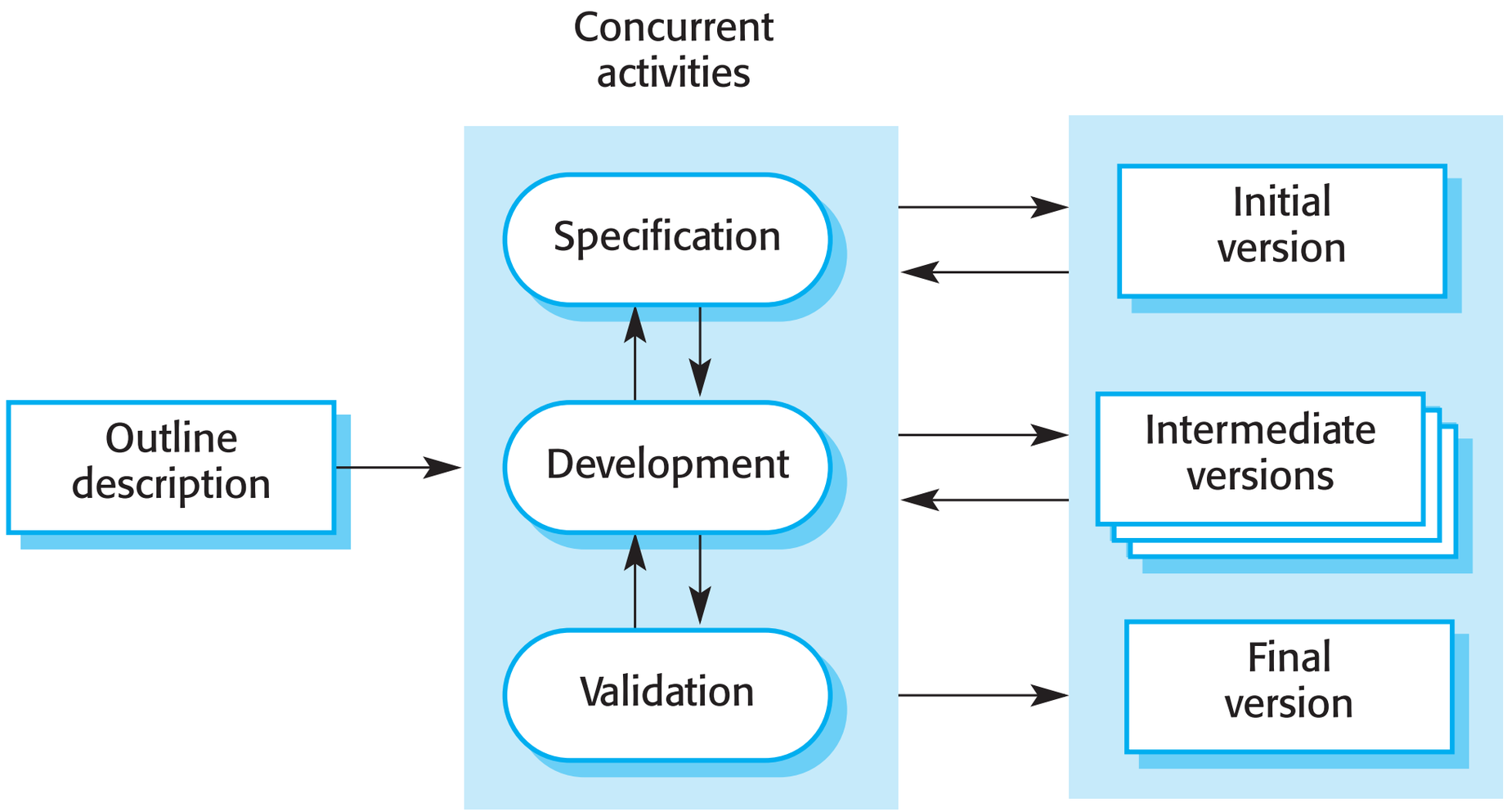
Process Models

- Waterfall 폭포수 모델
  - V-Model
- Incremental 점진적 모델
- Reuse-Oriented 재활용 중심 모델

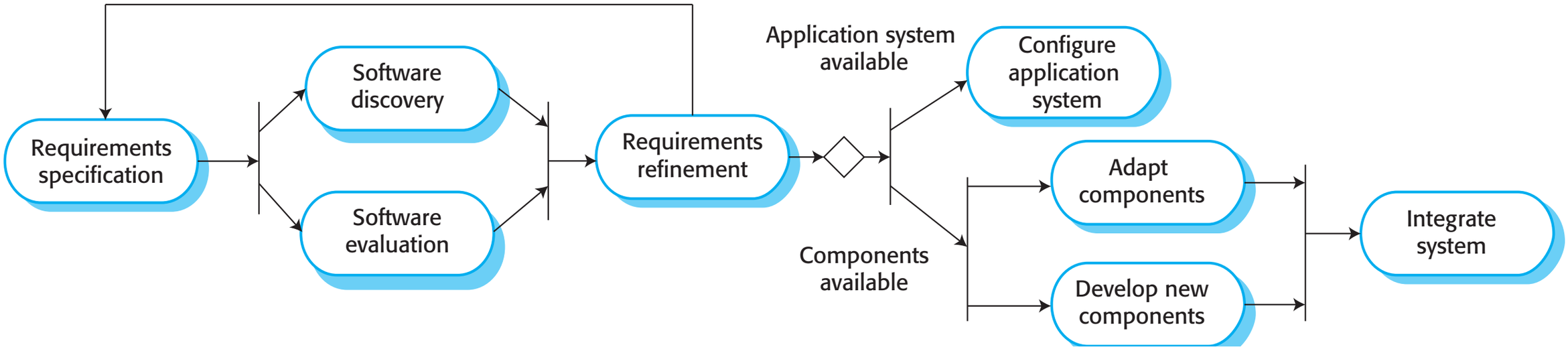
Waterfall


Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

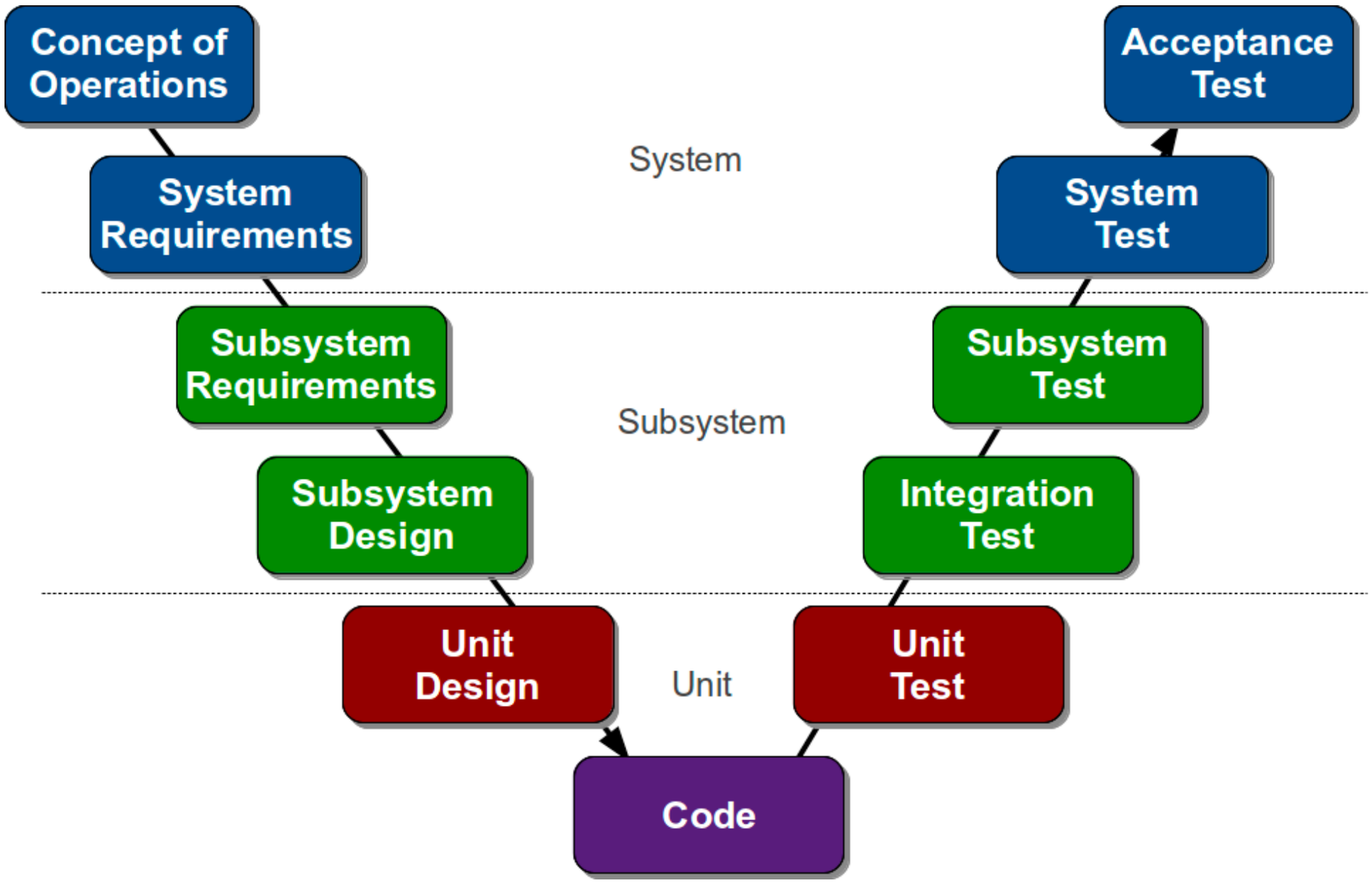
Incremental



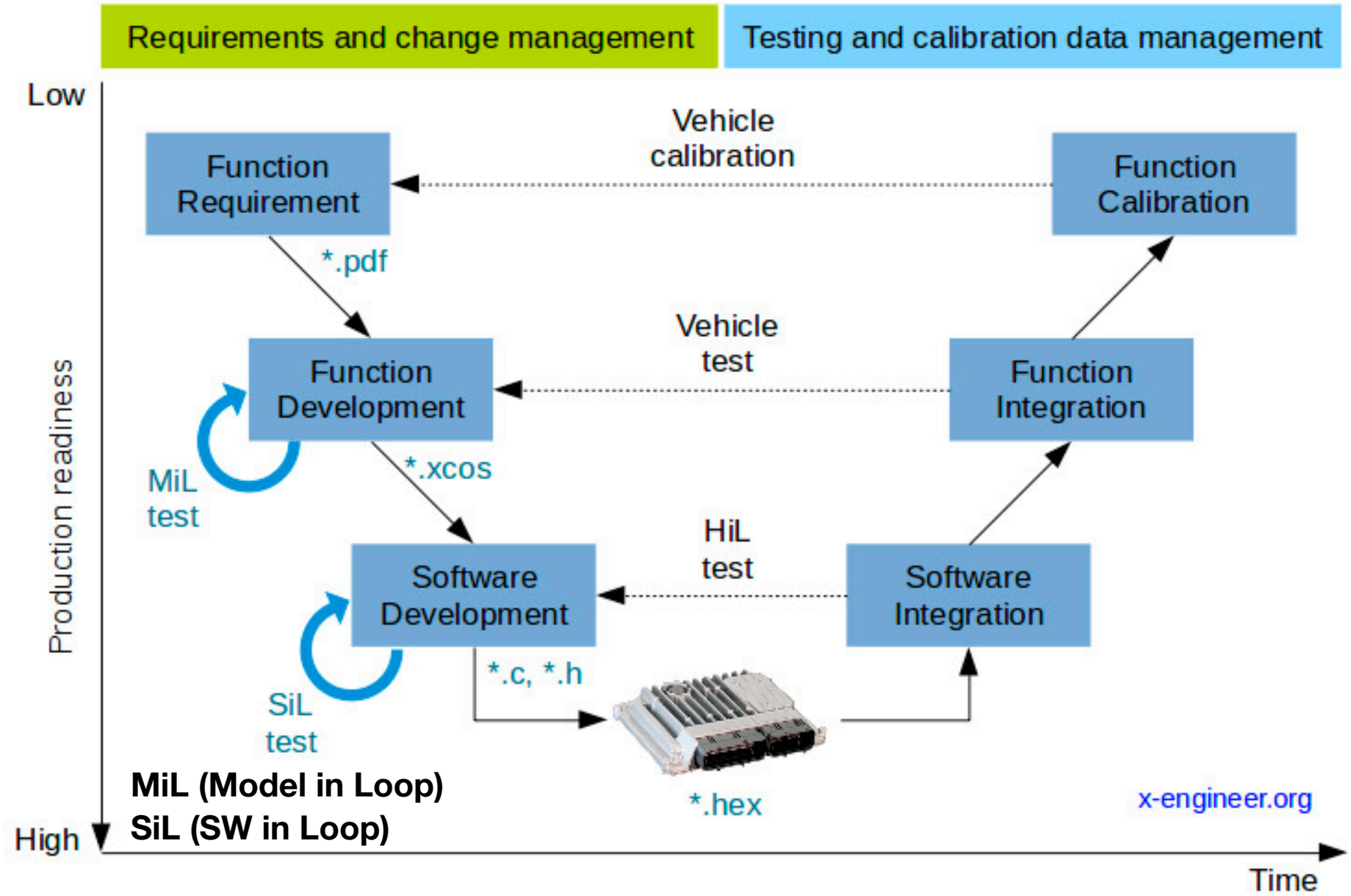
Reuse-Oriented



V-Model


Concept of Operations
System Requirements
Subsystem Requirements
Subsystem Design
Unit Design
Code
Unit Test
Integration Test
Subsystem Test
System Test
Acceptance Test
System
Subsystem
Unit

Example of V-Model



https://x-engineer.org/graduate-engineering/modeling-simulation/model-based-design/essential-aspects-of-the-v-cycle-software-development-process/

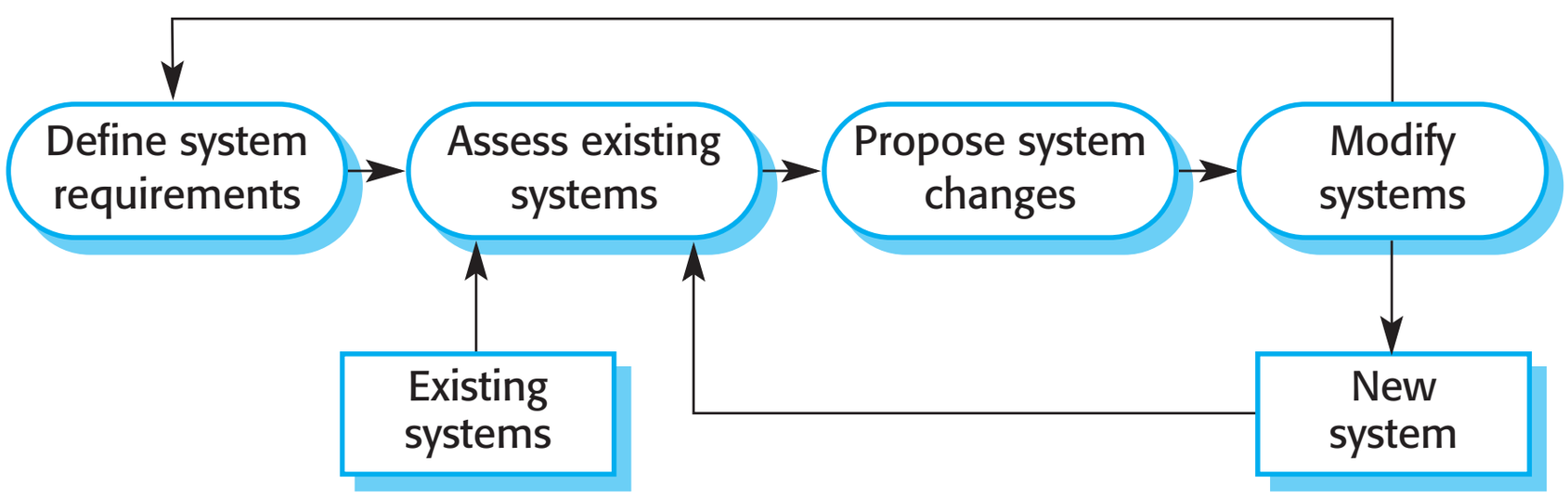
Think

- What is Pros. And Cons for each sw development process?

- Survey about the above questions and summarize them in 1 page.

SW Evolution

- SW is flexible and can change according to business needs

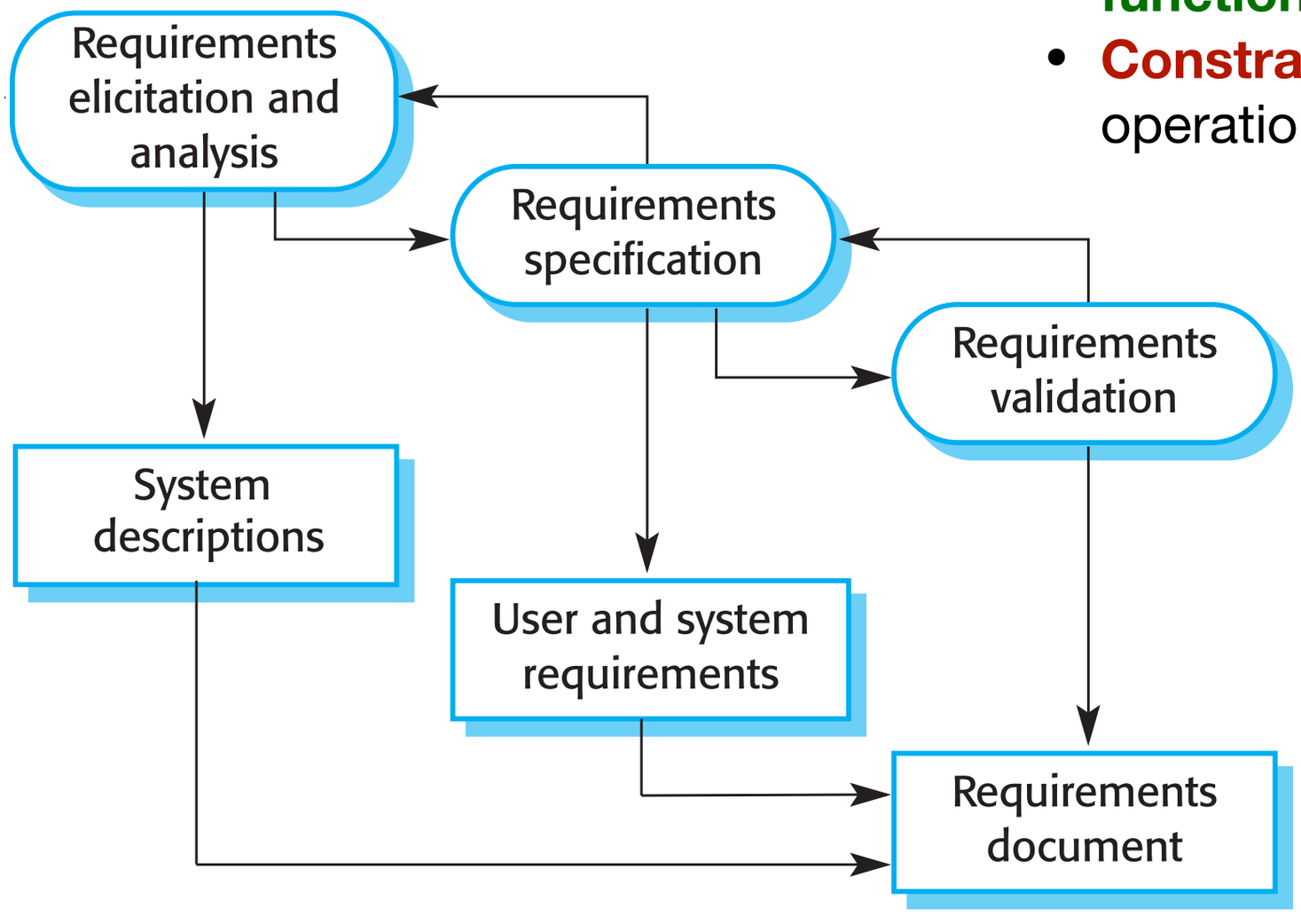


Key Activities in SW Process

- Requirements
- Specification
- Design
- Implementation
- Validation (vs Verification vs Testing)

Requirements

- **Demanded services** and **functionalities**
- **Constraints** on system operations and developments



Designs

- Build a design realizing requirements specification
  - Structure, architecture, DB, interfaces, and reusable components,
  - Some requirements can be removed due to design/ implementation constraints

Implementation

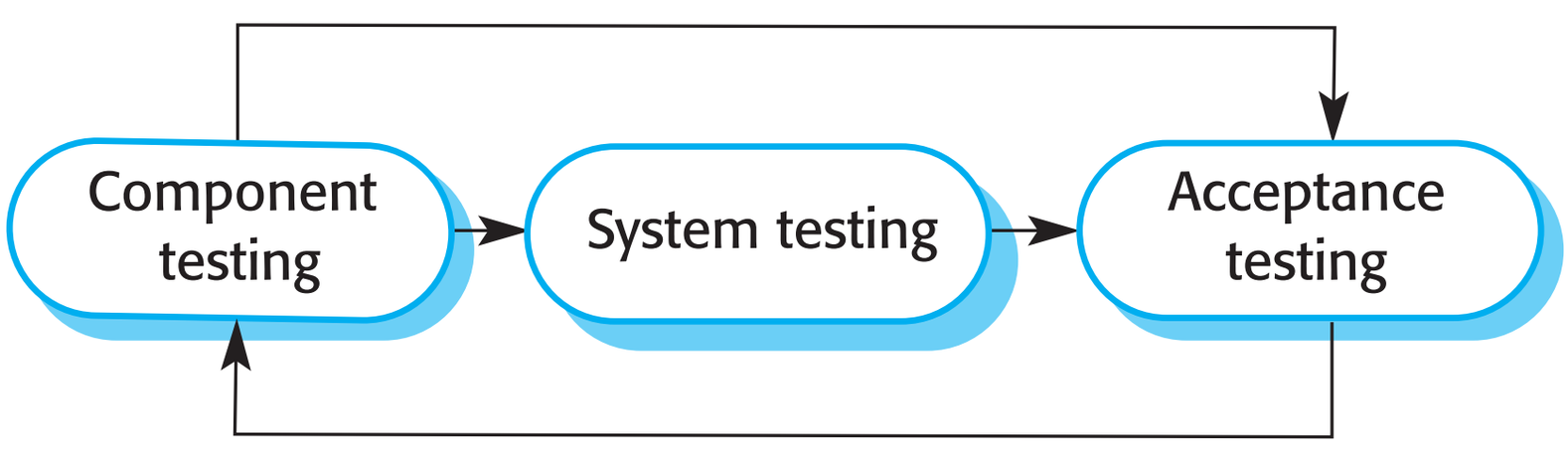
- Translate a system design into an executable program;
  - C, JAVA, Python, …
- Configure application systems such that the system including them can satisfy the requirements.
- Design and implementation can be interleaving
- Debugging involved

SW Validation

- Validation & Verification
  - Show that a system both **conforms** to its specification and **meets** the requirements of the system customer
  - Verification is to check a system is checked w.r.t. the previous stage requirements, e.g. requirements <-> design, design <-> implementation
  - Validation is to check a system is checked w.r.t. customer’s requirements, e.g. testing and simulation.
    - Usually, it needs scenarios for testing and simulation

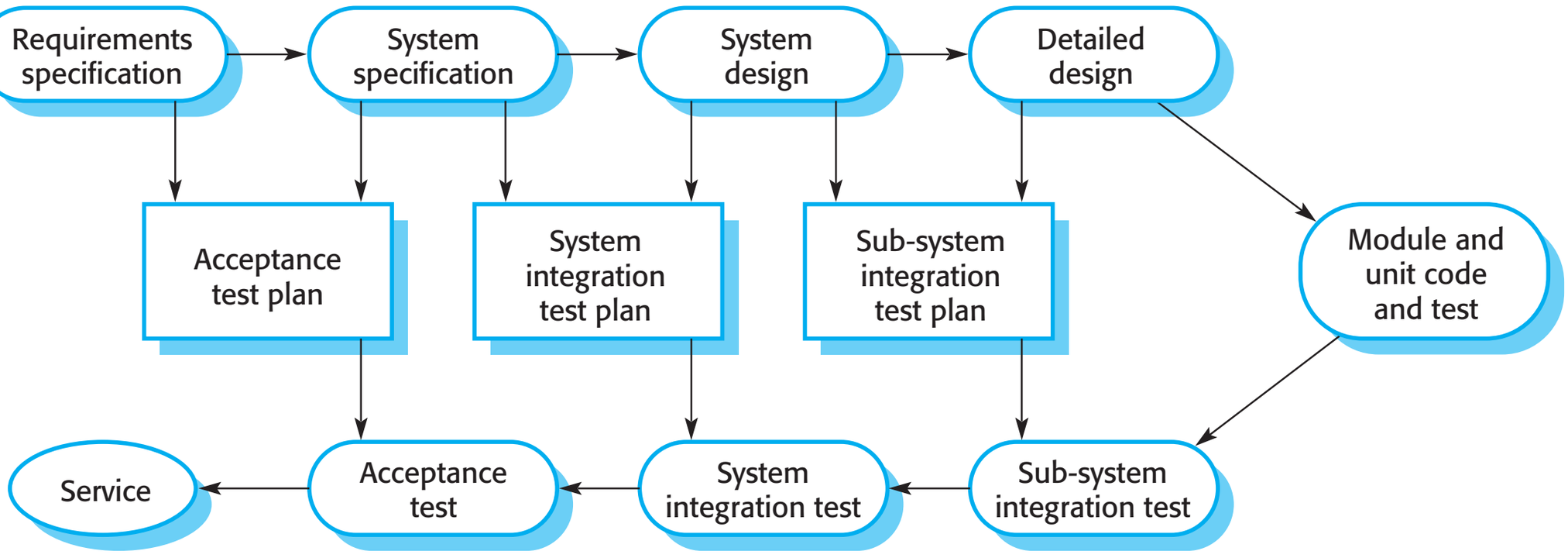
Testing

Components may be functions or objects or coherent groupings of these entities.



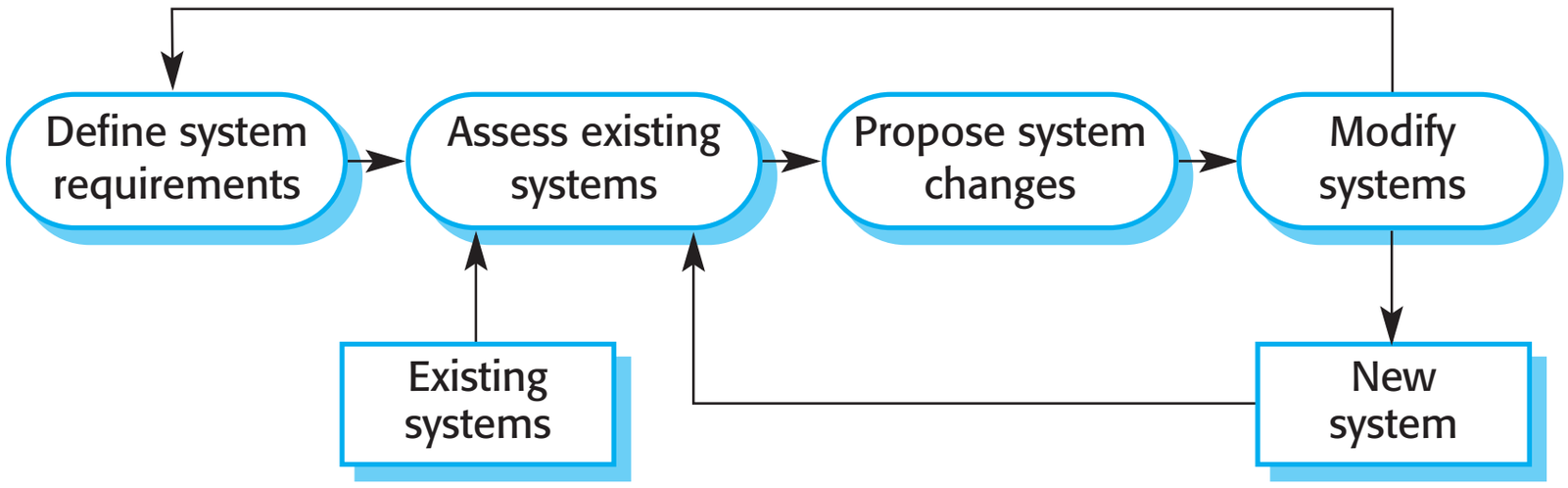
Customer testing
Testing with customer data to check that the system meets the customer’s needs.

Testing in V Model



Software Evolution

- Software changes to support the business, i.e. as requirements change through changing business circumstances



Summary

- SW Development Life Cycle
- SW Development Process Models and Activities
- In next class, we will take a look at
  - Cyber Physical Systems