SOFTWARE ENGINEERING

Chapter 1

Overview of software engineering

Content

I. Basic concepts

II. Software development processes

III. Career positions in software development

IV. Ethics in software engineering

Basic concepts about Software Engineering

Software

- **Software** = Program + Document
- **Software product** = Program + Document + Support
- Type of software products
  - **Generic Product**: Stand-alone systems that are marketed and sold to any customer who wishes to buy them.
  - **Bespoke Product**: Software that is commissioned by a specific customer to meet their own needs (Customized Product).

Essential attributes of good software

- **Maintainability** : Software should be written in such a way that it can evolve to meet changing customer needs.
- **Dependability** : Includes the reliability, security and safety of the software.
- **Efficiency** : Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilization, 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.

The role of software

- The economies of All developed countries are strongly dependent on software.
- More and more systems are controlled by software.
- Software spending take a significant portion of GNP (Gross National Product) in all developed countries.

=> How about in Vietnam?

We have enough software?

- Software has been written since the first programmable computers.
- Softwares are interested and developed very early
- A lot of softwares have been written

→ **There are so many softwares**

- In fact, the software product does not meet the requirements of users:
    - Not enough in quantity
    - Lack of quality
    - Not on time

→ **The software is not enough for users**

Objective reasons

- Unlimited user needs
  - All fields, industries, organizations and individuals have the need to use software.
  - User requirements are getting higher and more complex. Systems are required to have new features that were previously thought to be impossible.
  - The fast changing of hardwares.

Subjective reasons

- Software development is not highly professional. It is like an art (impromptu), not yet seen as a science.
- The software development process has not been unified
- No unified standard for software measurement.

Solution

→ It is necessary to have one/more professional process in software development to enhance the effectiveness of this special industry

Software engineering

- Software engineering is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use.
- Engineering discipline
  - Using appropriate theories and methods to solve problems bearing in mind organizational and financial constraints.
- All aspects of software production
  - Not just technical process of development. Also, project management and the development of tools, methods etc. to support software production.

Importance of SE

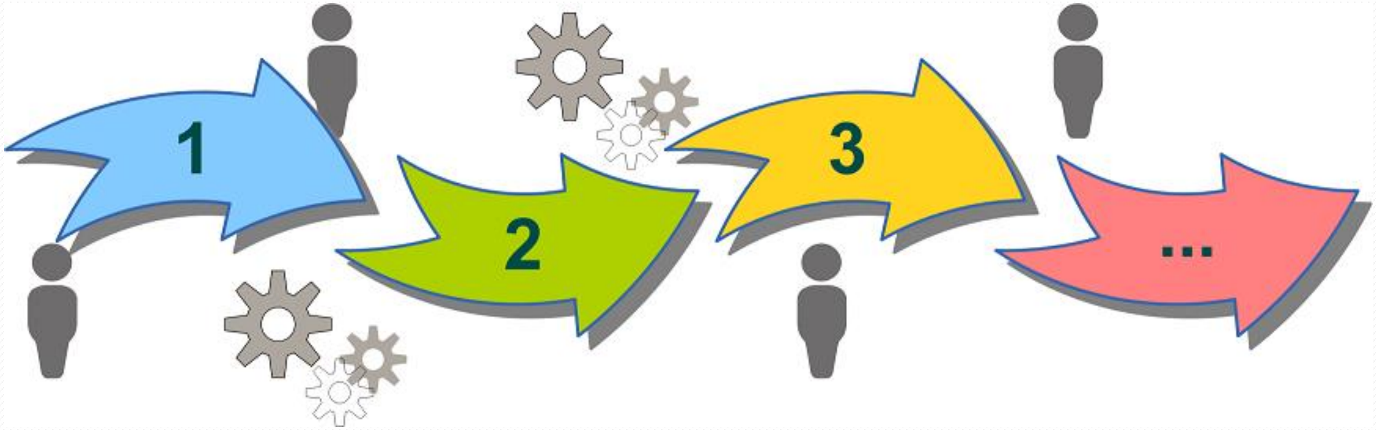
- More and more, individuals and society rely on advanced software systems. We need to be able to produce reliable and trustworthy systems economically and quickly.
- It is usually cheaper, in the long run, to use software engineering methods and techniques for software systems rather than just write the programs as if it was a personal programming project. For most types of system, the majority of costs are the costs of changing the software after it has gone into use.

Software engineering

- Being an engineering major involves three elements:
  - Procedure/process
  - Method
  - Tool

→ To develop efficient software with given requirements

SE – Process/Procedure



- Identify the stages and the order of execution of the stages
- Determine the documents, products to be delivered and how to do it
- Set milestones (milestones) and deliverables

Software engineering – Method

- Specific ways to complete the stages
- Each stage of software making has its own methods: Method specification, design, programming, testing, operation, maintenance, etc.

Software engineering – Tool

- Computer Aided Software Engineering (CASE) tools provide automatic/semi-automatic help for each method in each stages of the software development process
- Eg. IDE (editor, compiler, debugger, testing, version/source control,..,)

Software engineering – principles

Some basic principles apply to all types of software systems, regardless of the development techniques used:

- Must understand and manage the process when building software. Different processes are used for different types of software.
- Reliability and performance are important for all types of systems.
- Understanding and managing software specifications and requirements (what the software should do) is important.
- Should reuse already developed software rather than write new software.

Web-based software engineering

- The Web is a platform for running applications, and organizations are increasingly developing web-based systems rather than local systems.
- Web services allow application functionality to be accessed over the web.
- Cloud computing is an approach to providing computing services where applications run remotely in the 'cloud'.
    - Users do not buy software but buy a service and pay for what had been used (SaaS)

Web-based software engineering

SaaS
SOFTWARE AS A SERVICE

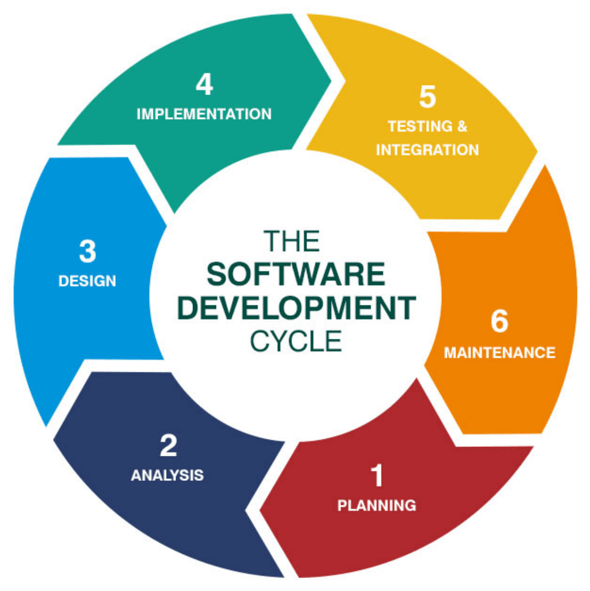
Web-based software engineering

- **Software reuse**
  - Software reuse is the predominant approach to building web-based systems. When building these systems, it is possible to assemble them from existing software systems and components.
- **Incremental and agile development**
  - Web-based systems should be developed and distributed incrementally. People now generally recognized that it is impractical to define all the requirements for such systems in advance.

Software development process

Software development process

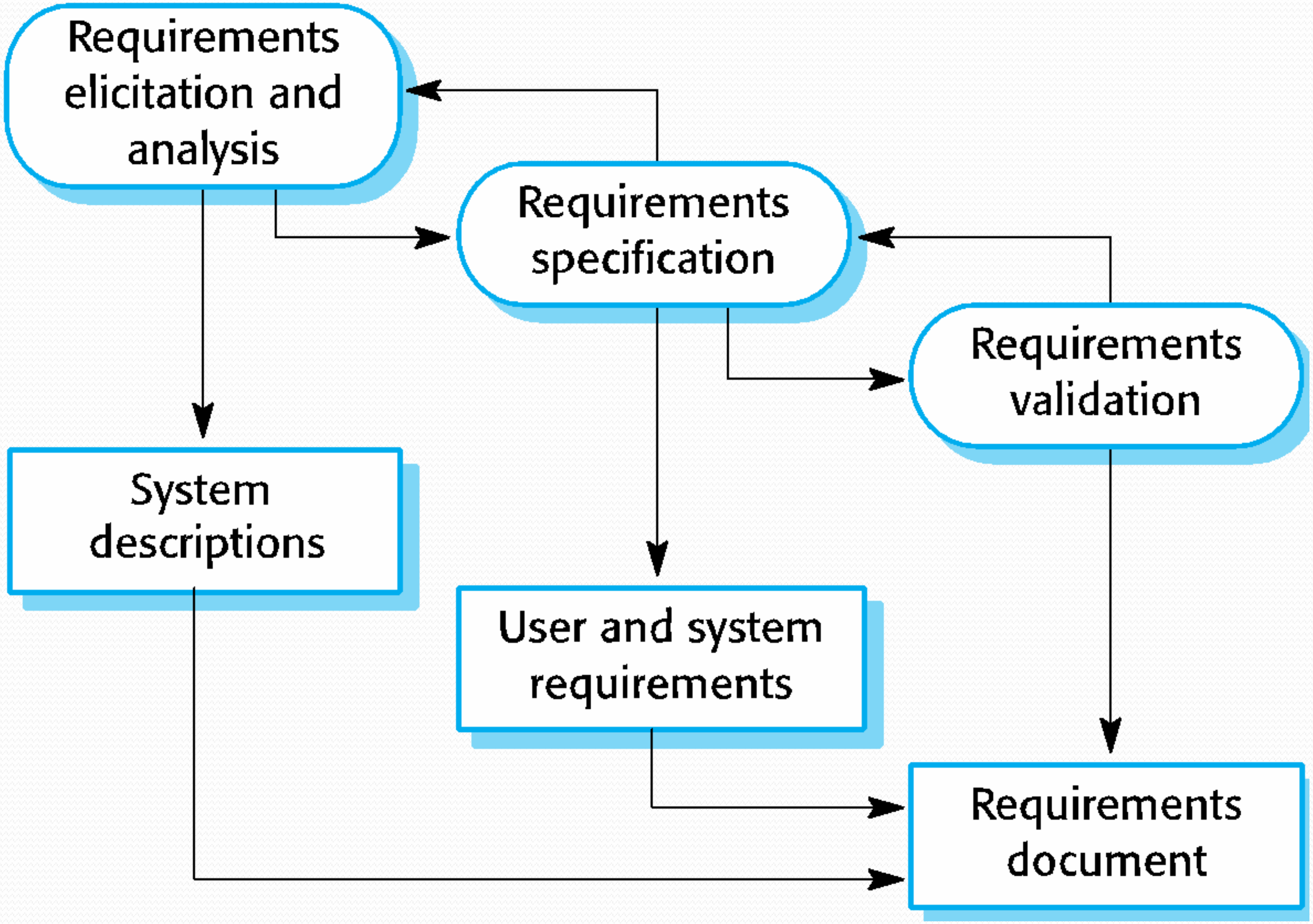
Software development process (SDP) is a set of stages that must be performed in the process of building a software system.



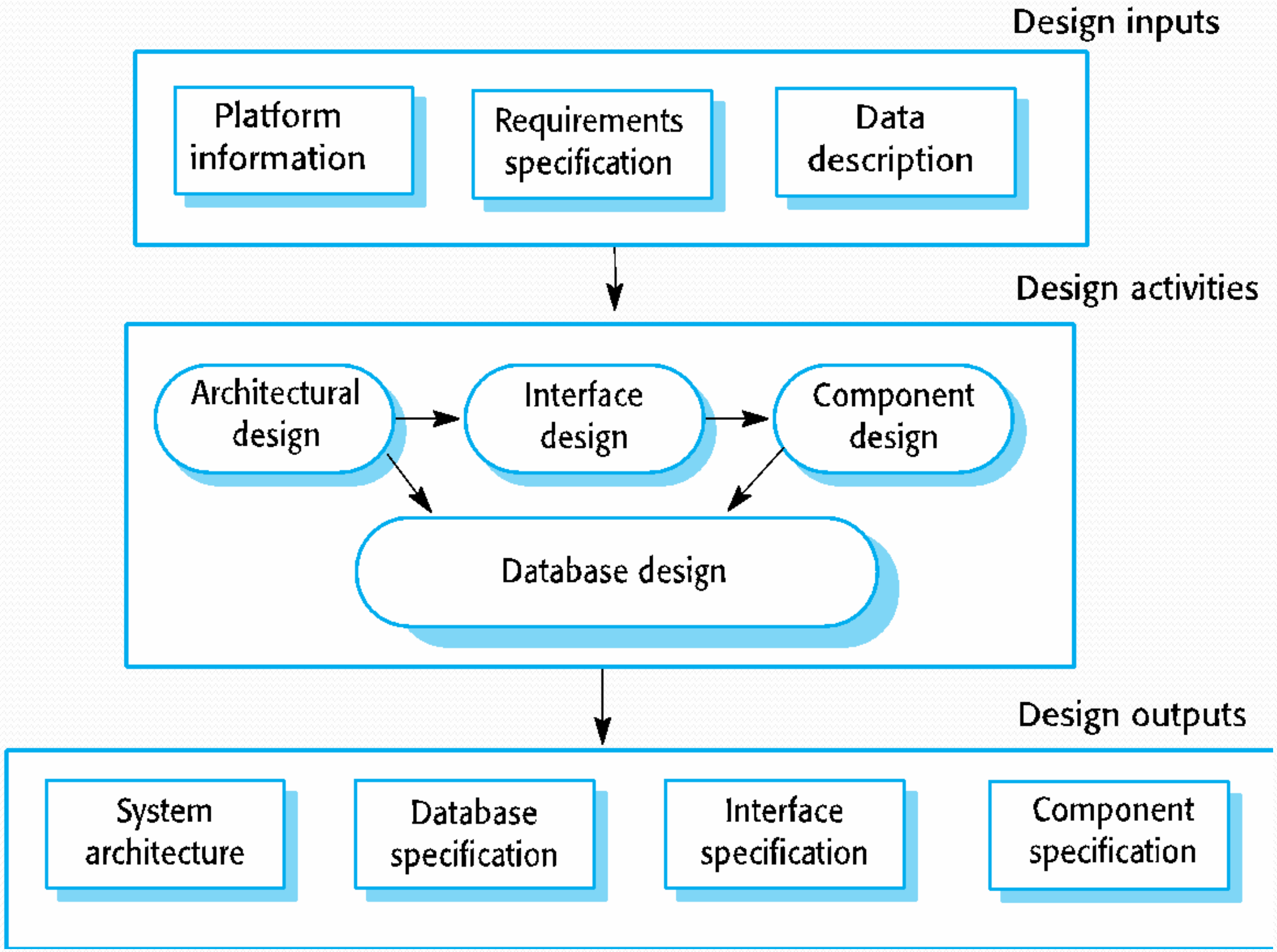
Main stages/phases of SDP

- **Specification/Analysis** : Determine the requirements and constraints of the system (Know the problem fully and exactly)
- **Design** : Define solutions to solve the problem
- **Implementation/Programming** : Using programming languages to turn designs into programs
- **Testing** : Evaluating checking that it does what the customer wants.
- **Maintenance, improvement** : Bug fixes, changing the system in response to changing customer needs, improvement,..

Specification/Analysis


Requirements elicitation and analysis
Requirements specification
Requirements validation
System descriptions
User and system requirements
Requirements document

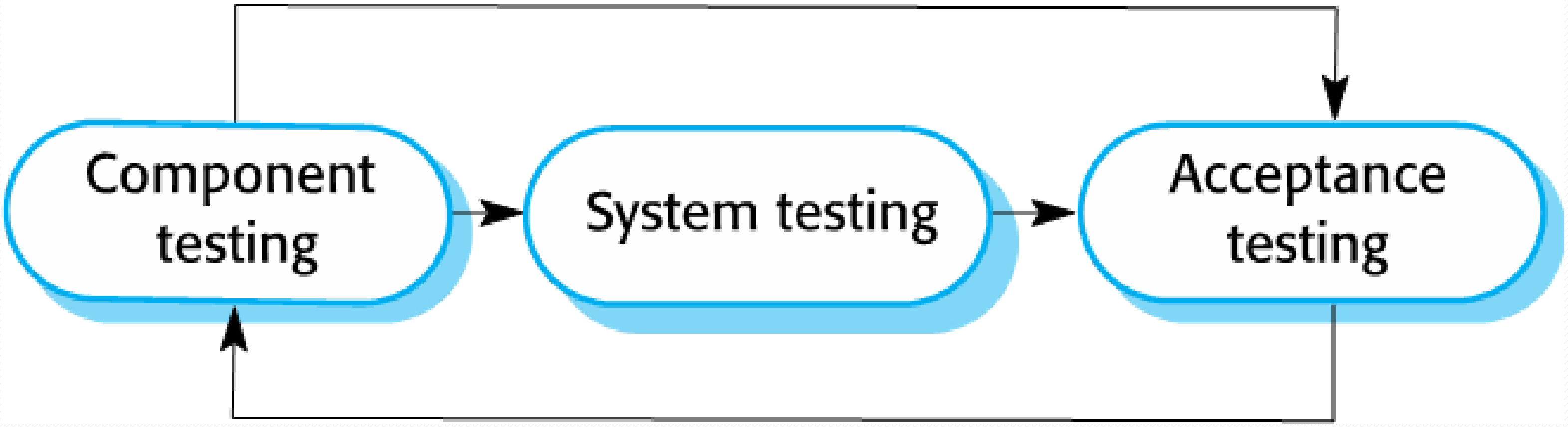
Design


Design inputs
Platform information
Requirements specification
Data description
Design activities
Architectural design
Interface design
Component design
Database design
Design outputs
System architecture
Database specification
Interface specification
Component specification

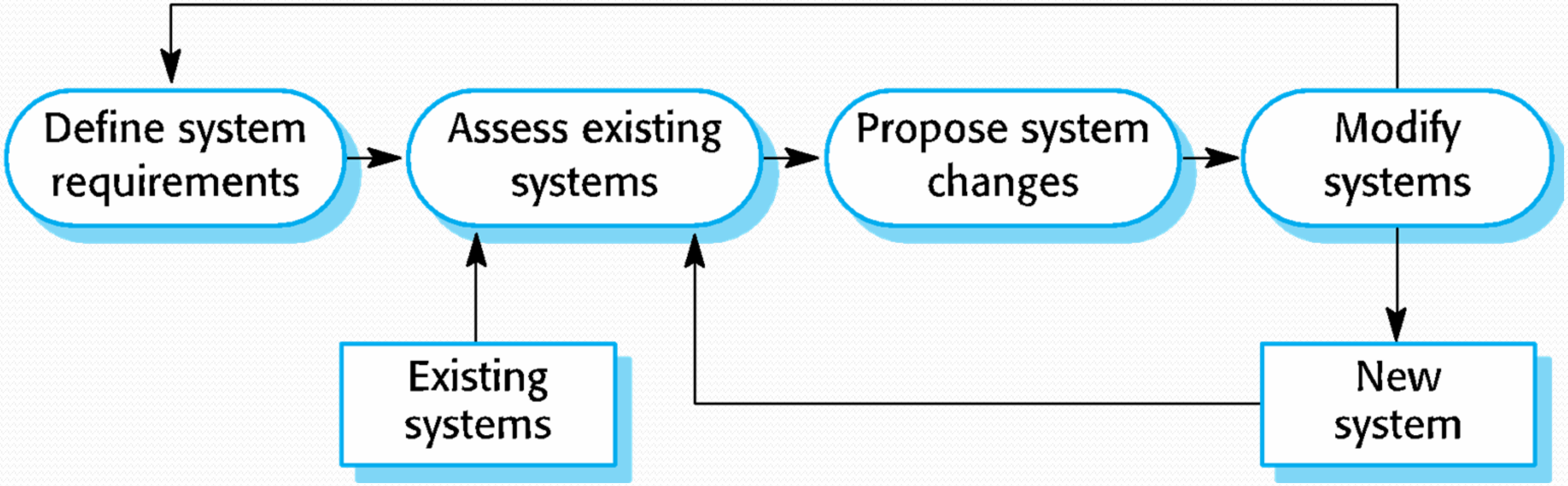
Implementation/Programming

- Transform designs into executable programs.
- Programming is a personal activity with no standard procedure.
- Debugging is the activity of finding program errors and fixing these errors.

Testing



Maintenance/Evolution



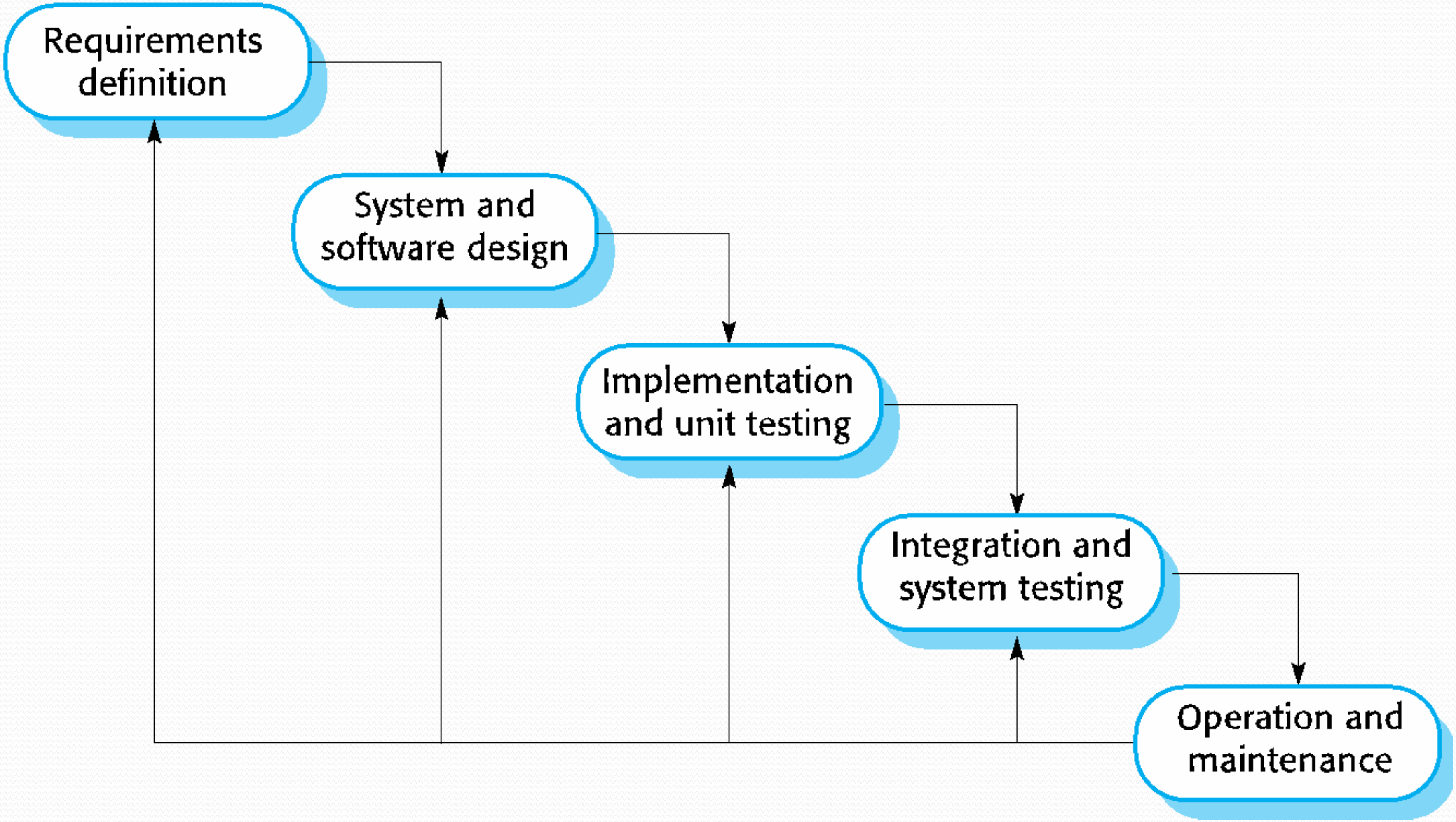
Some software process models

- Waterfall model
- Incremental/Iterative development model
- Intergration and reuse model

Software process models

- **Waterfall model**
  - Plan driven model. Separate and distinct phases of specification and development.
- **Incremental model**
  - Specification, design, programming, testing is done round by round. Can be scheduled or flexible.
- **Integration and reuse model**
  - The system is assembled from existing components. Can be scheduled or flexible.
- In practice, most large systems are developed using a process that combines all these models.

Waterfall model



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

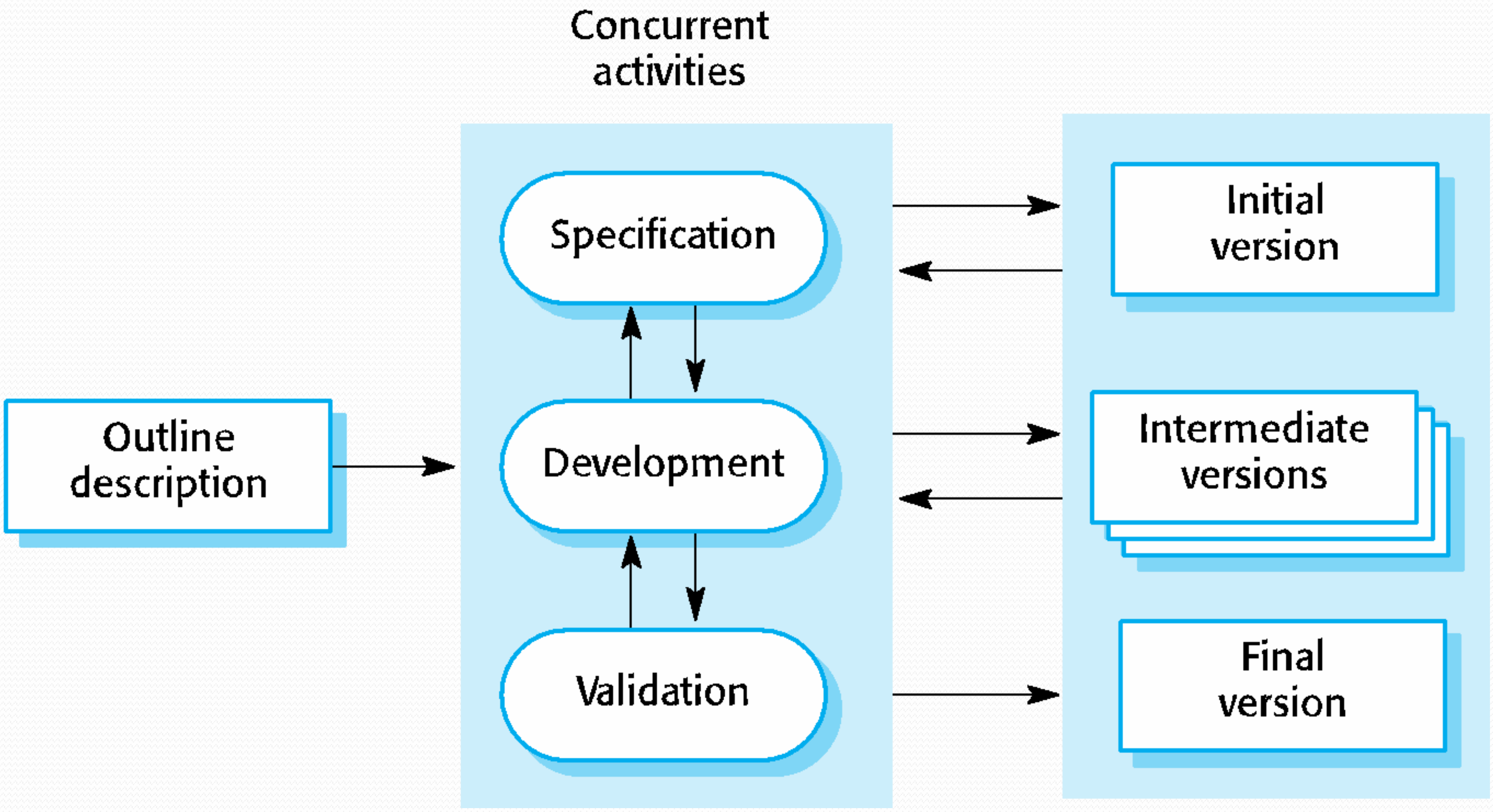
Waterfall model

- There are distinctly defined stages in the waterfall model:
  - Requirements definition (analysis and specification)
  - Design
  - Implementation/Program
  - Testing
  - Operation and maintenance
- The main limitation of the waterfall model is the difficulty of making changes after the process is in progress. In principle, a phase (stage) must be completed before moving on to the next phase.

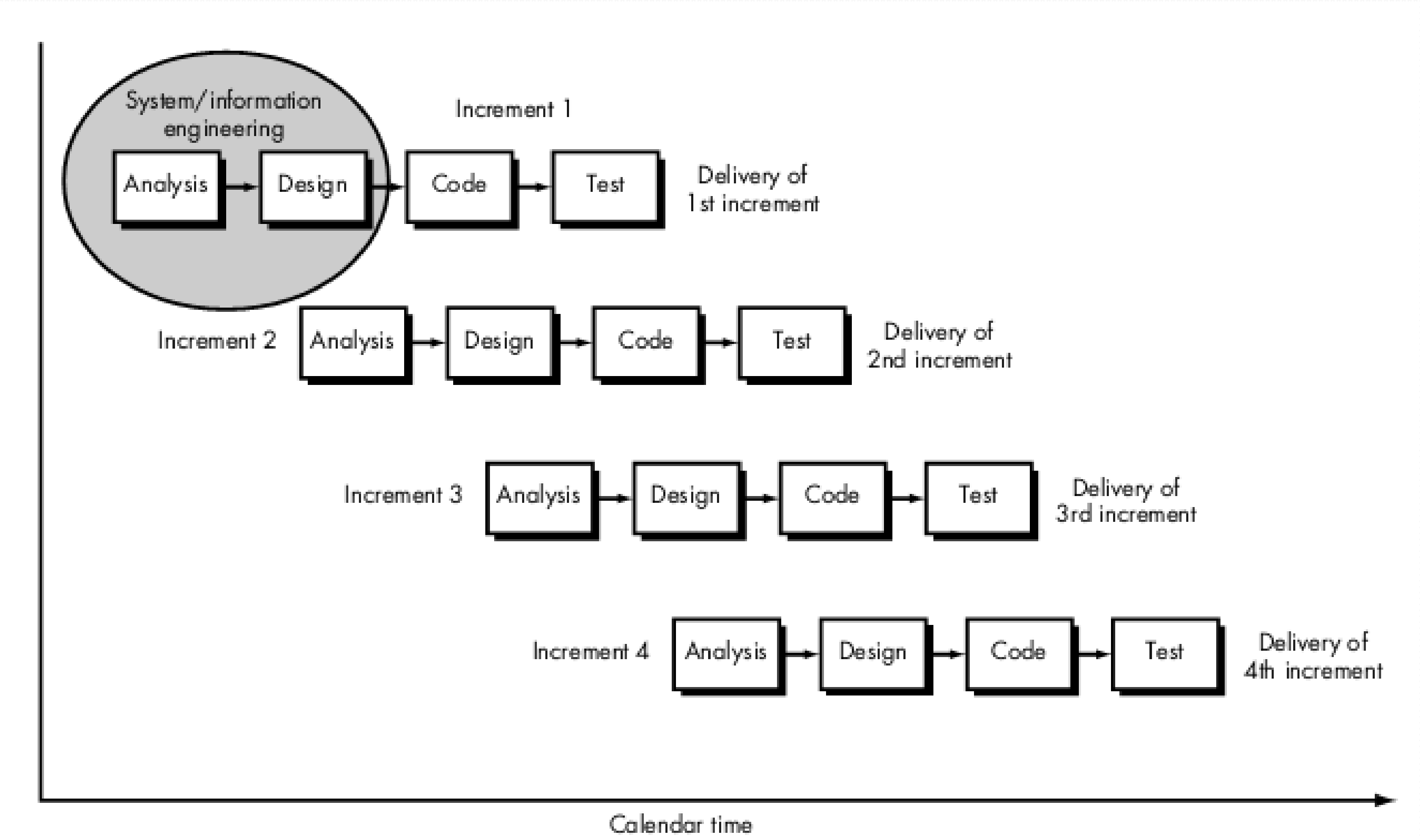
Waterfall model

- **Pros:**
  - Simple, natural and easy to manage.
- **Cons:**
  - Customers have to provide requirements clearly and completely from the outset. It is difficult to meet the changes later.
  - In fact, very few systems have stable business and requirements.
  - Customers often must wait a long time to see the first version of the product

Incremental development model



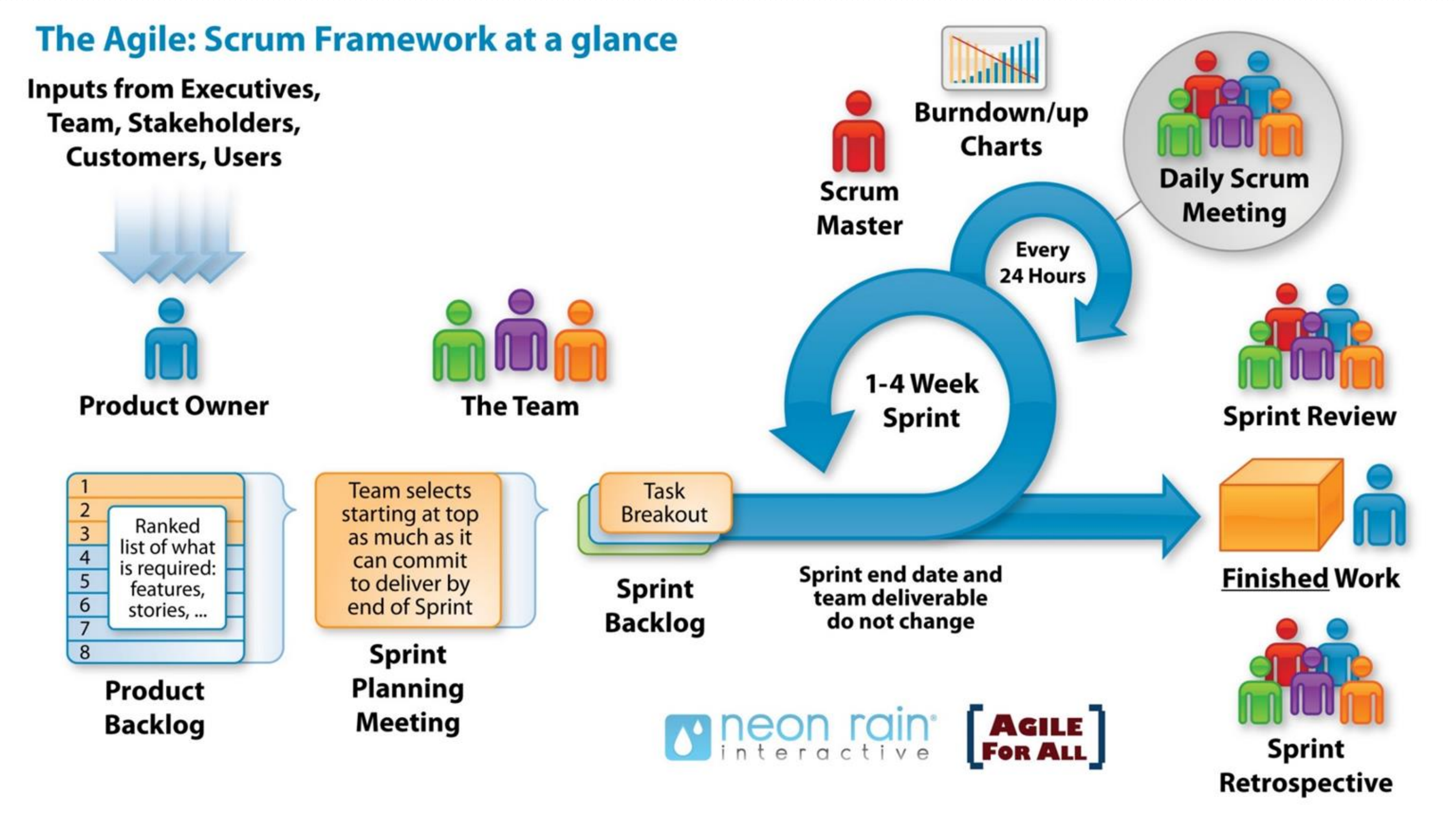
Incremental development model



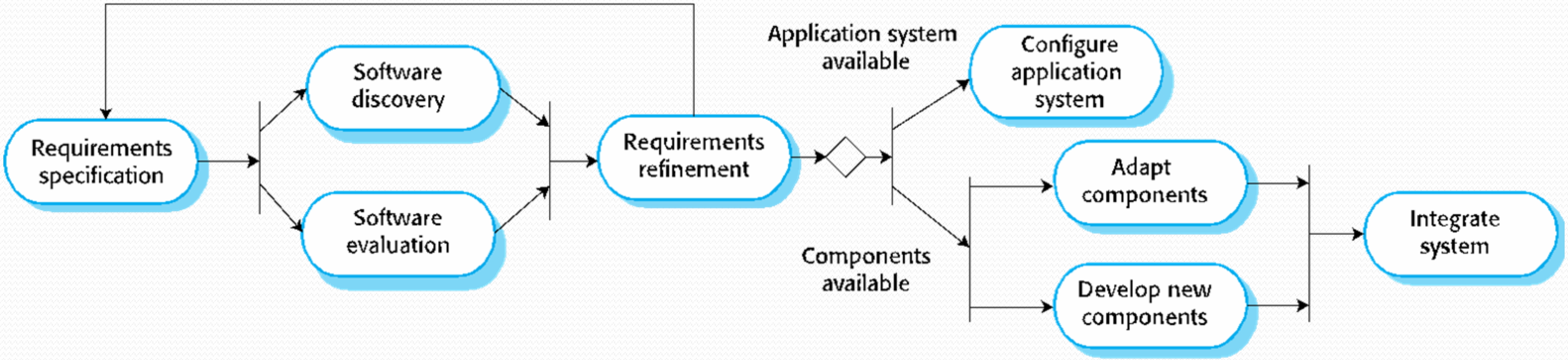
Incremental development model

- This model is proposed based on the idea that instead of having to build and transfer the system once, it will be divided into many rounds, increasing. Each round is a partial result of a set of requirements.
- User requirements are prioritized. The higher the priority requirement, the earlier it is in the development rounds.

Agile/Scrum


The Agile: Scrum Framework at a glance
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint Review
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective
neon rain interactive
AGILE FOR ALL

Intergration and reuse model



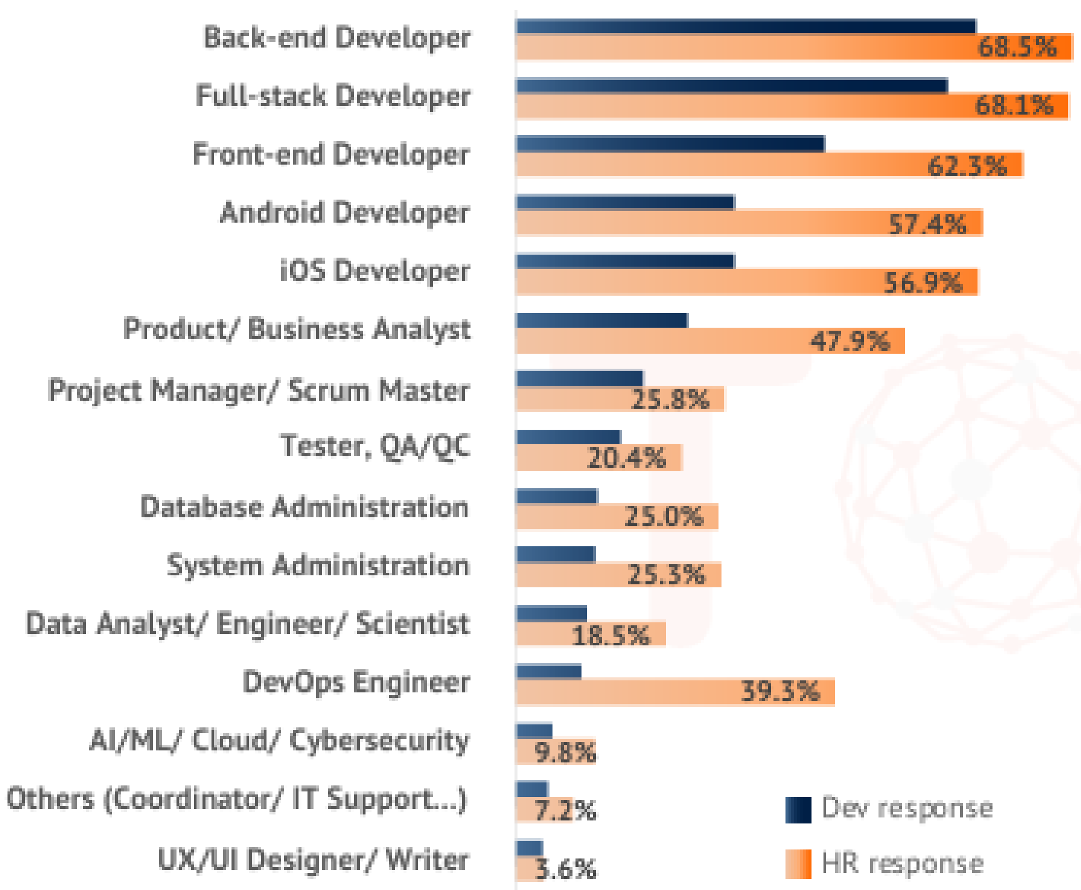
Intergration and reuse model

- This model is based on systematic reuse techniques; where the system is integrated from multiple existing or commercial COTS (Commercial-off-the-shelf) components.
- Components can be customized according to user requirements
- Reuse is now the standard method for building many types of business systems

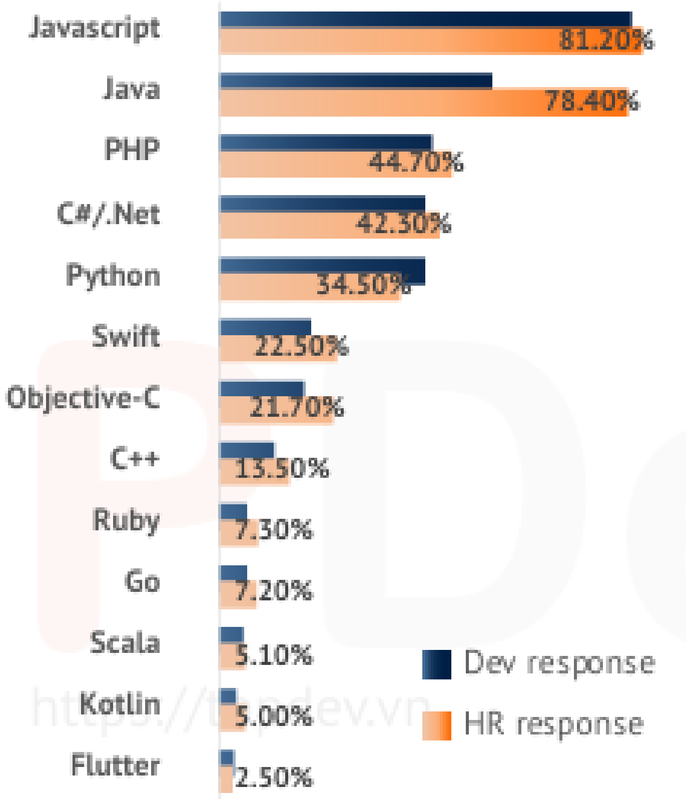
Career positions in software development

Type of work

TOP HIRING TECH POSITIONS VS MOST POPULAR POSITIONS AMONG IT INDUSTRY

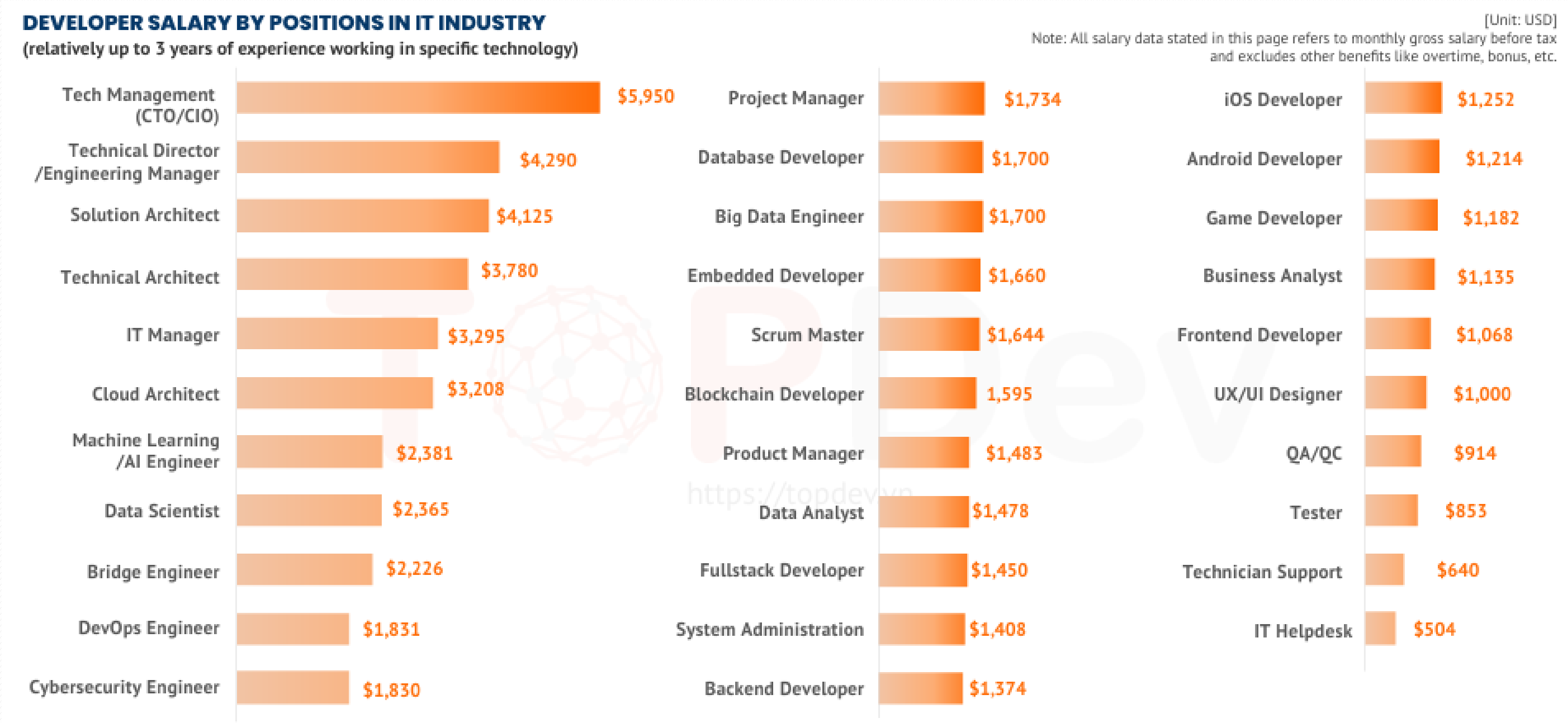


TOP HIRING TECH STACKS VS MOST POPULAR TECH STACKS



Nguồn: topdev

Income



DEVELOPER SALARY BY POSITIONS IN IT INDUSTRY

(relatively up to 3 years of experience working in specific technology)

[Unit: USD]

Note: All salary data stated in this page refers to monthly gross salary before tax and excludes other benefits like overtime, bonus, etc.

Source: topdev

Ethics in software engineering

Software engineering ethics

- Software engineering involves wider responsibilities than simply the application of technical skills.
- Software engineers must behave in an honest and ethically responsible way if they are to be respected as professionals.
- Ethical behaviour is more than simply upholding the law but involves following a set of principles that are morally correct.

Issues of professional responsibility

- Intellectual property rights
  - Engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc. They should be careful to ensure that the intellectual property of employers and clients is protected.
- Computer misuse
  - Software engineers should not use their technical skills to misuse other people’s computers. Computer misuse ranges from relatively trivial (game playing on an employer’s machine, say) to extremely serious (dissemination of viruses).

ACM/IEEE Code of Ethics

- The professional societies in the US have cooperated to produce a code of ethical practice.
- Members of these organisations sign up to the code of practice when they join.
- The Code contains eight Principles related to the behaviour of and decisions made by professional software engineers, including practitioners, educators, managers, supervisors and policy makers, as well as trainees and students of the profession.

The ACM/IEEE Code of Ethics

Software Engineering Code of Ethics and Professional Practice

ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices

PREAMBLE

The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that are included in the full version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations, the details can become legalistic and tedious; without the details, the aspirations can become high sounding but empty; together, the aspirations and the details form a cohesive code.

Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles:

Ethical principles

1. PUBLIC - Software engineers shall act consistently with the public interest.

2. CLIENT AND EMPLOYER - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.

3. PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

4. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.

5. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

6. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

7. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.

8. SELF - Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

Review

- Definition of software and software products
- Types, features and roles of software
- Software engineering definition, three elements of software engineering
- Features and Pros/Cons of some Software Process Model: waterfall model, Incremental development model, Intergration and reuse model.
- Some career positions in software developments and demands of current market
- Some principles of ethics in software engineering