Al-Farabi KAZAKH NATIONAL UNIVERSITY

# Software Design

**Lecture 1**

# Basic concepts and definitions

A.R. Turganbayeva

Almaty
2016

- Review of modern technologies of software design.
- Organization of the software development process

# Vocabulary

- Waterfall – сарқырама - водопад
- gathering – жинау - сбор
- Conversion – түрлендіру - преобразование
- Deployment – орналастыру - развертывание, размещение
- Domain model - домен моделі - модель предметной области
- Compliance – сәйкестігі - согласие
- Flaws – кемшіліктер - дефекты, недостатки
- commercial operation – коммерциялық операциялар - коммерческая эксплуатация
- Maintain – қолдау - поддержка

# The Software Challenge

- *People may come and go, but software may remain*
  - A software product is often expected to be used for an extended period of time by someone who did not write the program and who is not intimately familiar with its internal design
- *Software may evolve*
  - New features may be added, environments may change, so initial specification may be incomplete

# The Software Specification Challenge

- *Software specification is not easy*
  - It should be generated at the beginning of project and maintained up-to-date while the software goes through changes
  - It should be clarified through extensive interaction between the users and the system analyst, and then approved by the users
  - It should be clear and understandable to any programmer

- For the successful implementation of an IT project rather choose effective technology and development tools to provide the necessary budget, and to find skilled developers.
- In any organization, there are rules and techniques of project participants (customers, analysts, developers, testers, technical writers) distribute among themselves tasks interact with each other, create project artifacts (specifications, source code, documentation).
- These rules may be well organized or chaotic, or be formally documented to exist in the minds of the project team, but in any case it is their combination is called a **process of development**.

# A Process of Software Development

- Process - a special case of the more general concept of software development methodologies.
- Examples of methodologies are structured programming or object-oriented analysis and design.

# Software Design

- Deriving a solution which satisfies software requirements

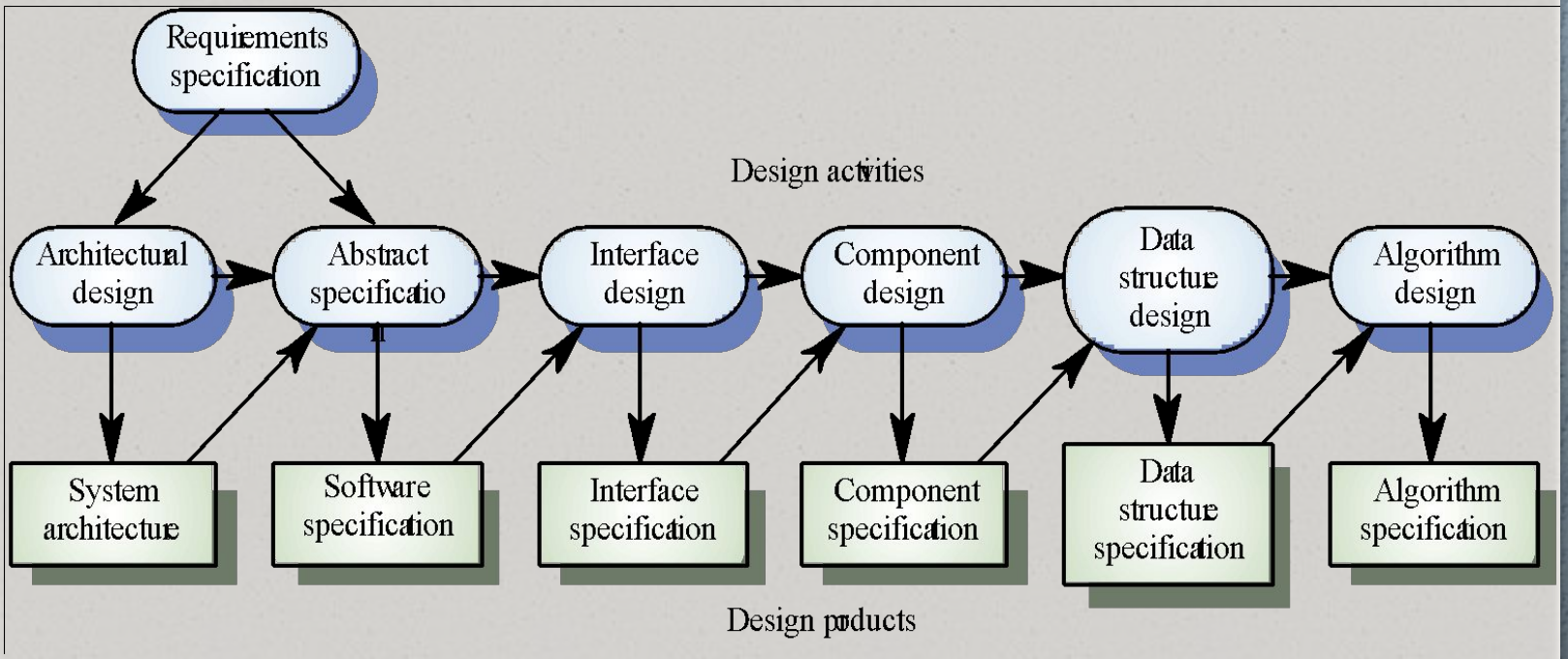
# Stages of Design

- Problem understanding
  - Look at the problem from different angles to discover the design requirements.
- Identify one or more solutions
  - Evaluate possible solutions and choose the most appropriate depending on the designer's experience and available resources.
- Describe solution abstractions
  - Use graphical, formal or other descriptive notations to
    describe the components of the design.
- Repeat process for each identified abstraction until the design is expressed in primitive terms.

# The Design Process

- Any design may be modelled as a directed graph made up of entities with attributes which participate in relationships.
- The system should be described at several different levels of abstraction.
- Design takes place in overlapping stages. It is artificial to separate it into distinct phases but some separation is usually necessary.

# Phases in the Design Process



# Design of small and large systems

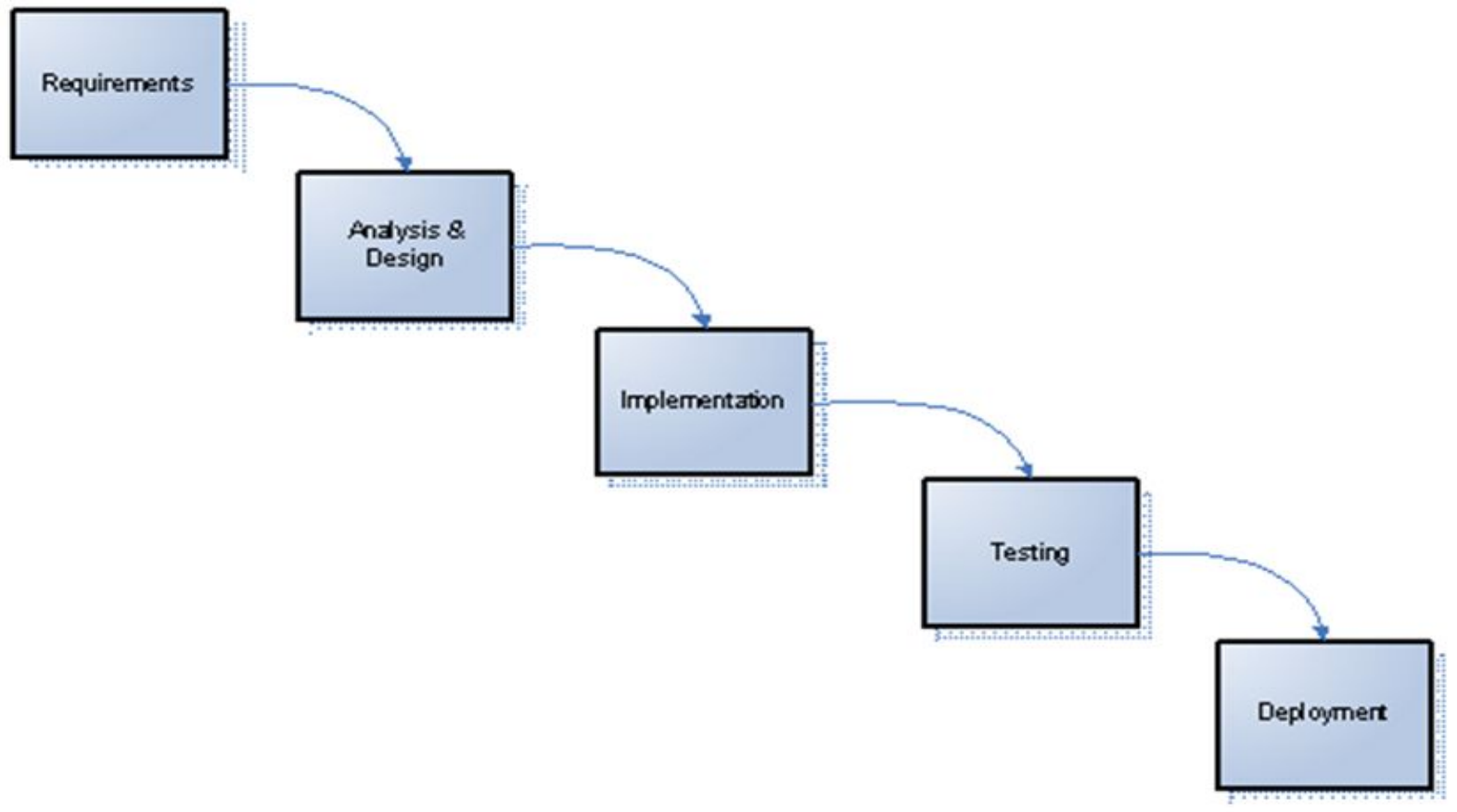
# The **Software Life Cycle** – The Life and Death of Software

- Software products go through several stages as they mature from initial concept to finished product
- The sequence of stages is called a **life cycle**

# Waterfall Model

- Model Falls (waterfall model or serial development) - probably the most famous and historically appeared one of the first development process.
- He was described in the article Royce (W.W. Royce) in 1970.
- The basic idea is that the development process is divided into well-defined phases, performed strictly sequentially.
- The name "waterfall" appeared due to the appearance of the diagram depicting the process:

# The Diagram of Waterfall Model



# The classic waterfall model includes the following areas

- **Develop requirements**: gathering business requirements of the customer and their conversion to the functional requirements of a software product.
- **Analysis and Design**: the development of a domain model, the design of the database schema, object model, user interface, etc.
- **Realization** (implementation): creation of a product according to the specifications developed in the previous step.
- **Testing**: includes the verification of compliance with the functionality of the software needs of users (validation), as well as finding flaws in implementation.
- **Deployment**: user training, system installation, transfer into commercial operation.

# Key considerations for the use of such a model development

- As you know, the cost of correcting mistakes made in the implementation of the project depends on how quickly these errors are detected and corrected.
- The error in the requirements simply correct requirements at the design stage, but if it becomes aware of after the completion of the deployment, the consequences can be catastrophic.
- Waterfall model tends to reduce as far as possible, the number of long-lived errors.
- For this design development should not start until the requirements are not identified with sufficient quality, the coding is not started until the complete system design, etc.

# Design Principles in Software Life Cycle Activities

- **Top-down approach**: breaking a system into a set of smaller subsystems
- **Object-oriented approach**: identification of a set of objects and specification of their interactions
- **UML diagrams** are a design tool to illustrate the interactions between
  - Classes
  - Classes and external entities

# Requirements Analysis, Use Cases, and Sequence Diagrams

- First step in analysis is to study the problem of input and output requirements carefully to make sure they are understood and make sense
- ***Use case***: list of the user actions and system responses for a particular sub-problem in the order that they are likely to occur
- ***Sequence diagram***: shows all the objects involved in this use case across the horizontal axis, time is shown along the vertical axis

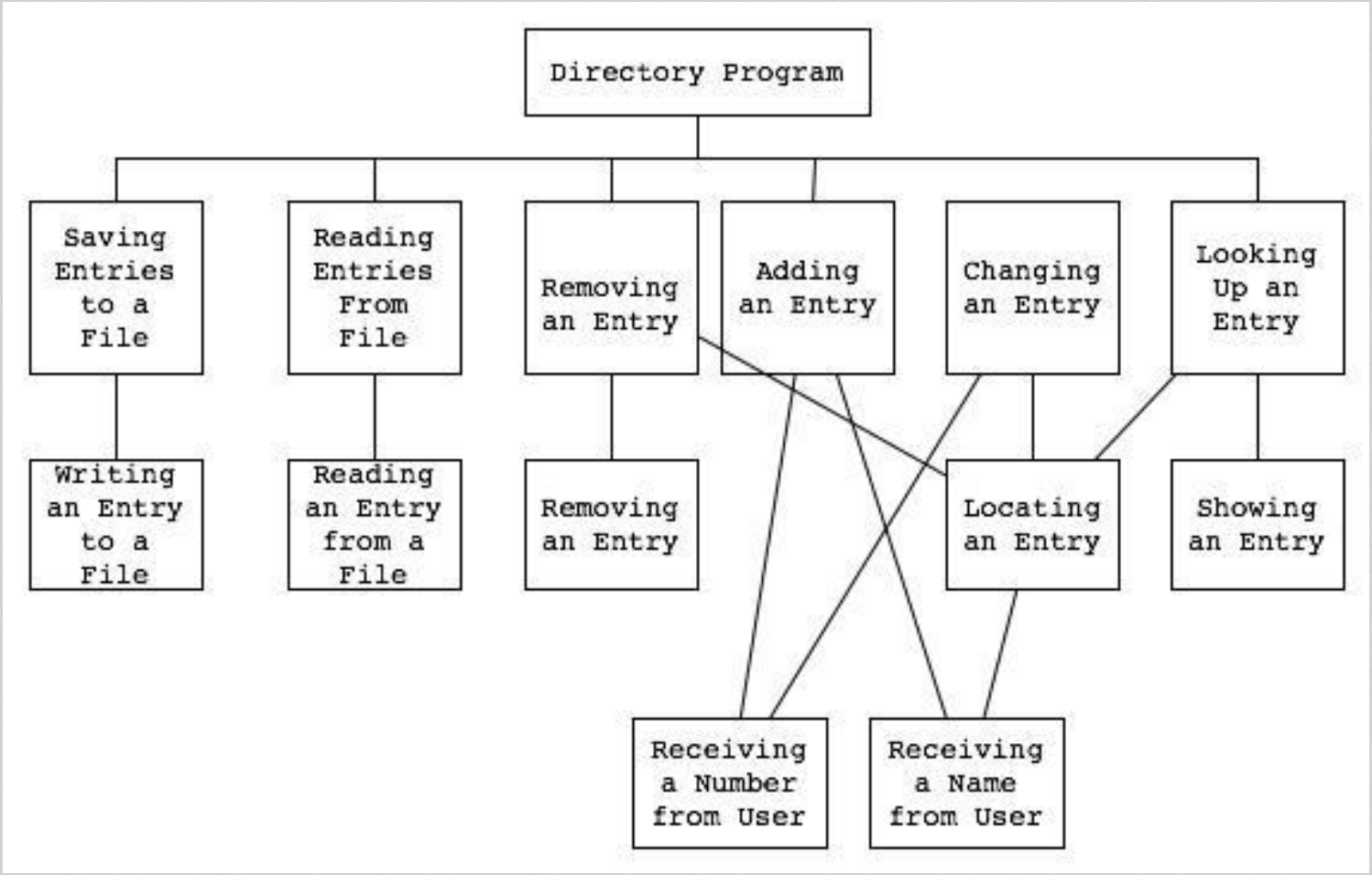
# Pre- and Postconditions

- **Precondition**: a statement of any assumptions or constraints on the method data before the method begins execution
- **Postcondition**: a statement that describes the result of executing a method

# An Example: Telephone Directory

- Maintain a collection of telephone directory entries, where each entry is referred to by a unique name.
- Can read from a file, save to a file, lookup, add, remove, and change phone number

# Dependencies Among Possible Actions



# Things you already know (about) …

- Java programs (you know and love)
- Classes and objects (you can create and use)
- Inheritance (you understand and can extend)
- Abstract classes (you remember what they are)
- Interfaces (your contractual obligations)

# The modern technologies of software design

- Rapid application development (RAD)
- Spiral model
- Component-Based Model
- Heavy and lightweight processes
- XP-processe
- CMMI
- Agile Manifesto
- Microsoft Solutions Framework (MSF)
- OpenUP &OpenUP/MDD
- Model Driven Development

**Thank you for the attention!**