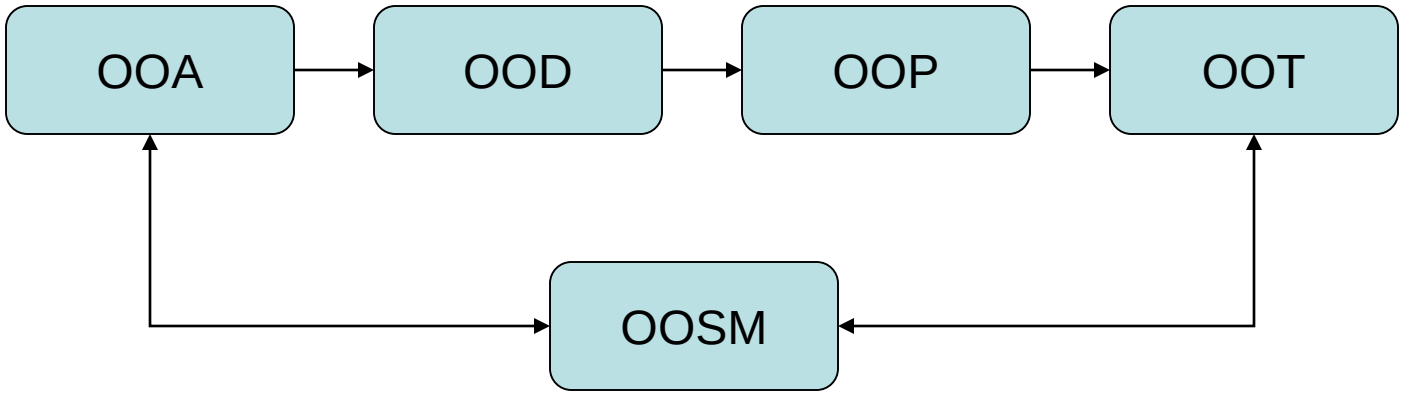
Object-Oriented Technology and UML

Qingjian Ni(倪庆剑)
Southeast University
niqingjian@gmail.com

About This Course

- The main curriculum
- Also associated with specific technology



Why This Course?

- Unified Modeling Language
- From Programmer to Systems analyst
- From Worker to Architect

Objectives

- UML notations
- Object-Oriented analysis and design using UML
- Practical projects using UML

Course References

- Grady Booch,James Rumbaugh, Ivar Jacobson. The Unified Modeling Language User Guide, 2nd Ed. 2005
- Carig Larman. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process, 3nd Ed. 2006
- Scott W. Ambler. The Elements of UML 2.0 Style 2nd Ed. 2007
- The latest UML specification
- ……

Overview

- Introduction to concepts of OO
- Introduction to RUP and UML
- UML tools
- Requirements modeling
- Static modeling
- System dynamics modeling
- System physical modeling

Grading

- 40%
  - Attendance is critical
  - Quiz
  - Projects and Reports
- 60%
  - Final Exam

Course Schedules

(There may be changes, Follow the latest notifications)

- Week 1
  - Course Overview (Tue.)
  - Introduction to RUP & UML (Thu. Room 268, Computer Building)
- Week 2
  - Usecase Diagram (Tue.)
  - UML Tools (Thu., Room 268, Computer Building)
- Week 3
  - Usecase Diagram, Class Diagram (Tue.)
  - Cases of Usecase Diagram (Thu., Room 268, Computer Building)

Course Schedules

(There may be changes, Follow the latest notifications)

- Week 4
  - Class Diagram（Thu.）
  - Cases of Class Diagram (Thu., Room 268, Computer Building)
  - Practical projects (Thu. night, Practice)
- Week 5
  - Class Diagram, Object Diagram and Package Diagram (Thu.)
  - Cases of Class Diagram (Thu., Room 268, Computer Building)
  - Practical projects (Thu. night, Practice)

Course Schedules

(There may be changes, Follow the latest notifications)

- Week 6
  - Package Diagram, Activity Diagram (Tue.)
  - Cases of Activity Diagram (Thu., Room 268, Computer Building)
  - Practical projects (Thu. night, Practice)
- Week 7
  - State Machine Diagram (Tue.)
  - Other UML Diagrams and Course Summary (Thu.)
  - Practical projects (Thu. night, Practice)
- Week 8
  - Show of Experimental works, Discussion, Q & A (Tue.)
  - Final Exam (Thu.)

Topics

- Overview of relevant background knowledge
  - Software Engineering and UML
  - Object-oriented Technology and UML

The Relationship Between Some Important Concepts

- Software Engineering
- Modeling
- Object-oriented Development Methods
- UML
- UML Tools(Such as Rational Rose)

Software Engineering and UML

Software Crisis

- Software crisis was a term used in the early days of computing science for the difficulty of writing useful and efficient computer programs in the required time.
- The software crisis was due to the rapid increases in computer power and the complexity of the problems that could be tackled.
- With the increase in the complexity of the software, many software problem arose because existing methods were neither sufficient nor up to the mark.

Software Crisis

- Projects running over-budget
- Projects running over-time
- Software was very inefficient
- Software was of low quality
- Software often did not meet requirements
- Projects were unmanageable and code is difficult to maintain

Causes of Software Crisis

- The causes of the software crisis were linked to the overall complexity of hardware and the software development process

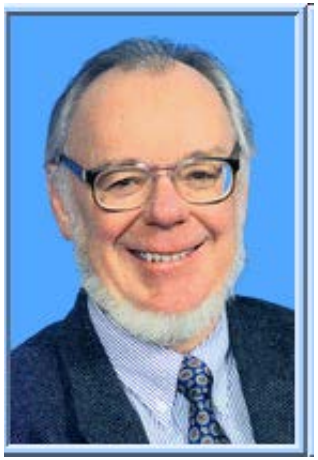
Software Engineering

- 1968, Garmisch, Germany, NATO

Software Engineering

- Peter Naur and Brian Randell



Status Of Software Development

- Software crisis persists
- Important goal of software engineering: Solving the software crisis
- UML is one of the important means

Complexity Of The Software System

- “Software entities are more complex for their size than perhaps any other human construct……”
  - By Frederick P. Brooks (1999 Turing Award Winner)
  - Originally appeared in: Brooks, Frederick P., "No Silver Bullet: Essence and Accidents of Software Engineering," IEEE Computer, Vol. 20, No. 4 (April 1987) pp. 10-19

Solution to Software Crisis?

- Software complexity is inherent
- Reduce and control the complexity of software

The basic method of controlling software complexity

- Divide
- Abstract
- Modularization
- Information hiding

Software Life Cycle

- It contains processes, activities, and tasks that are to be applied during the acquisition of a software product or service and during the supply, development, operation, maintenance and disposal of software products
- UML has applications at any stage of Software Life Cycle

The Main Phases Of The Software Life Cycle

- Requirement Analysis
- Software Design
- Coding/Programming
- Testing/Debug
- Running/Maintenance
- UML has applications at any stage of Software Life Cycle

Software Process Model

- Waterfall model
- Prototype model
- Evolutionary model
  - Incremental model
  - Spiral model

Waterfall Model

- Also known as the linear sequential model
  - Simple and easy to understand and use
  - Each phase has specific deliverables and a review process
  - Works well for smaller projects where requirements are very well understood
  - Lack of flexibility
  - Cannot accommodate changing requirements
- UML can be applied to this model

Prototype model

- The stepwise approach to design a prototype
  - Basic requirement identification
  - Developing the initial prototype
  - Review of the prototype
  - Revise and enhance the prototype
- Features
  - Circulation
  - The introduction of user evaluation
  - Stepwise refinement
- UML can also be applied to this model

Incremental Model

- The incremental model applies the waterfall model incrementally
  - The series of releases is referred to as “increments”, with each increment providing more functionality to the customers
  - After the first increment, a core product is delivered, which can already be used by the customer
  - Based on customer feedback, a plan is developed for the next increments, and modifications are made accordingly
  - This process continues, with increments being delivered until the complete product is delivered
- UML can also be applied to this model

Spiral Model

- The spiral model is similar to the incremental model, with more emphasis placed on risk analysis
- The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation
- A software project repeatedly passes through these phases in iterations (called Spirals in this model)
- The baseline spiral, starting in the planning phase, requirements are gathered and risk is assessed. Each subsequent spirals builds on the baseline spiral
- UML can also be applied to this model

UML Diagrams In The Major Phases Of The Software Life Cycle

- Requirement Analysis
  - Use case diagram…
- Software Design
  - Class diagram, interaction diagram, state machine diagram, activity diagram…
- Coding/Programming
  - Class diagram …
- Testing/Debug
  - Class diagram , component diagram, deployment diagram …
- Running/Maintenance
  - Deployment diagram …

Software Process Model and UML

- UML is a modeling language, not a method
- UML is independent of the software development process
- In a particular use, the mechanism used in different institutions are not the same, especially in
  - Which UML diagrams should be used for modeling
  - Modeling granularity
  - The way of using UML
  - ……

Software Engineering Tools And Environments

- Provides automatic and semi-automatic support for the implementation of software engineering
- Computer Aided Software Engineering (CASE) integrated environment
- Support all phases of software development
  - Analysis and design tools
  - Programming tools
  - Test tools
  - Maintenance tools
  - Project management tools
  - ……
- UML Tools
  - A wide variety of tools available

Software Engineering Methods And Techniques

- Structured method
- Object-oriented method
- Component-based method
- Agent-based method
- ……

Structured method

- Process oriented
  - Data flow is emphasized
- Functional decomposition
  - Stepwise refinement
- Advantages
  - Simple, easy to master, the method is more mature, there are more tools to support
- Disadvantages
  - Low development efficiency, reusability is poor

Object-oriented Method

- Describe the things in the real world as objects
- Objects are divided into classes, various hierarchical relationships may be constituted between various classes
- Objects interact through messages
- An object is an encapsulation of data and methods
- UML supports object-oriented technologies comprehensively

Discussion On Software Engineering And The UML

- In the software company, what is the real situation about the use of UML?

Overview of object-oriented technology and UML

What is Object-Oriented Technology

- Object-Oriented Analysis
- Object Oriented Design
- Object-Oriented Programming
- Object-Oriented Testing
- Object-Oriented Maintenance

Modeling

- Software engineering
  - Start from a series of modeling tasks
  - Analysis
    - What to do
  - Design
    - How to do
- UML is a "Unified Modeling Language", can be used for modeling software systems

Modeling Views

- Data Modeling (Object Modeling)
- Functional Modeling
- Behavioral Modeling
- The five (4+1) views of an architecture are discussed in RUP and UML

Structured Method And Object-oriented Method

- Structured analysis (SA) in Structured method
  - Focusing on functional decomposition of a system
- Object-oriented analysis (OOA)
  - Focusing on understanding the problem from the perspective of real objects
- Take a library management system as an example
  - SA: book registration, overdue fines registration …
  - OOA: library administrators, catalogs and books …

Applicable Scope Of Structured Method And Object-oriented Method

- SA
  - Requirements of customers is very clear
  - Definition of business process is complete, and will not often change
- OOA
  - The software will use object-oriented programming language
  - Customer requirements are not clear
  - Judging by past experience, customers will frequently require the addition of new features
  - The system to be developed is complex
  - ……

Features Of Structured Method And Object-oriented Method

- Structured Method
  - Focusing on data structures, algorithms and implementation steps
  - Code is difficult to reuse
  - Lack of visual modeling techniques with strong expression
- Object-oriented Method
  - System is constructed by objects
  - Sending messages between objects
  - An object combines data and behavior together
  - Visual models can be easily evolved as a model of solution

Procedural Programming and Object-Oriented Programming

- Procedural Programming
  - Programs = Algorithm + Data Structure
- Object-Oriented Programming
  - Class
  - Object
  - Inheritance
  - Messaging
- UML support for object-oriented concepts better

Object and Instance

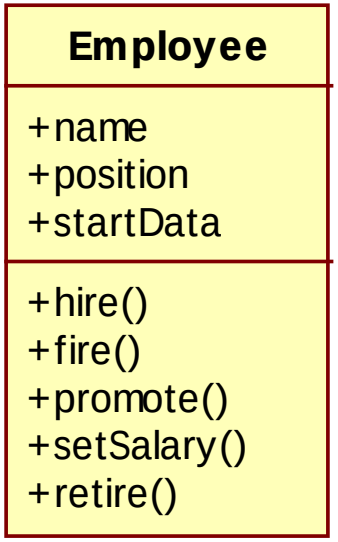
- Object
  - An entity described in a system
  - A basic unit constituting a system
  - Object includes properties and methods
  - Objects communicate via messages
- Instance
  - The concept is similar to the object, but there is a broader meaning than the object
  - In UML, object is the class instance, collaboration is the instance of use case, link is the instance of association…

Class

- Classes are the most important building block of any object-oriented system
- A class is a description of a set of objects that share the same attributes, operations, relationships, and semantics
- A class implements one or more interfaces

Class

- Name
- Attributes
- Operations



Encapsulation

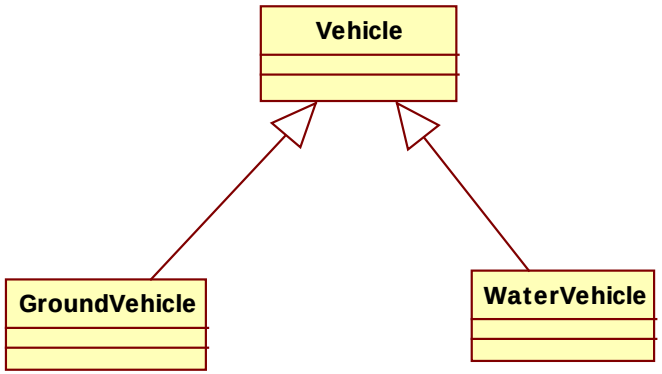
- Encapsulation
  - Encapsulation is the packing of data and functions into a single component
  - Encapsulation means that the internal representation of an object is generally hidden from view outside of the object's definition. Typically, only the object's own methods can directly inspect or manipulate its fields

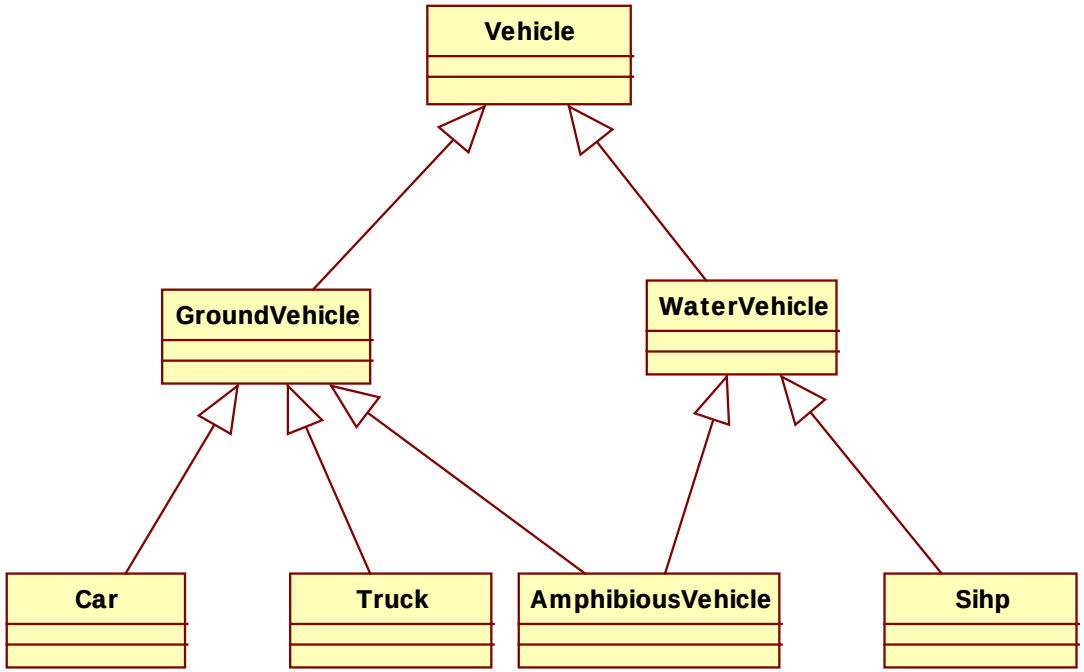
Inheritance

- Inheritance
  - In object-oriented programming (OOP), inheritance is when an object or class is based on another object or class, using the same implementation (inheriting from a class) or specifying implementation to maintain the same behavior
  - It is a mechanism for code reuse and to allow independent extensions of the original software via public classes and interfaces
- Override and overload
- Single inheritance and multiple inheritance

UML Diagrams Of Single Inheritance And Multiple Inheritance

- Single Inheritance Multiple Inheritance





Polymorphism

- Polymorphism
  - In programming languages, polymorphism is the provision of a single interface to entities of different types
- Concepts associated with polymorphism
  - Inheritance
  - Override
  - ……

Message

- Message passing sends a message to a process and relies on the process and the supporting infrastructure to select and invoke the actual code to run
- Message passing is a key feature to object-oriented programming
- Message passing
  - Synchronous
  - Asynchronous
- In UML, message passing between objects can be expressed

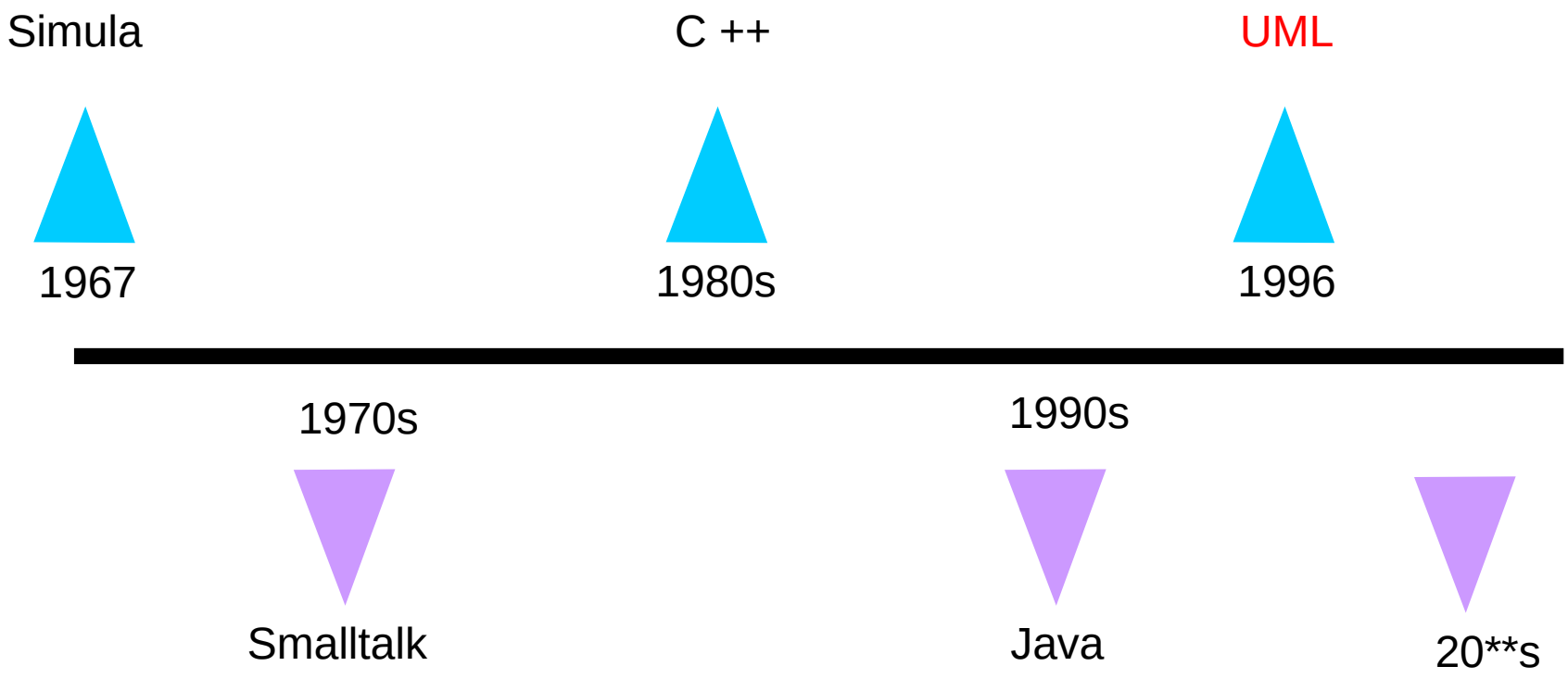
UML & Object-oriented Method

- UML has a very good support for key object-oriented concepts
  - Object
  - Class
  - Inheritance
  - Message passing

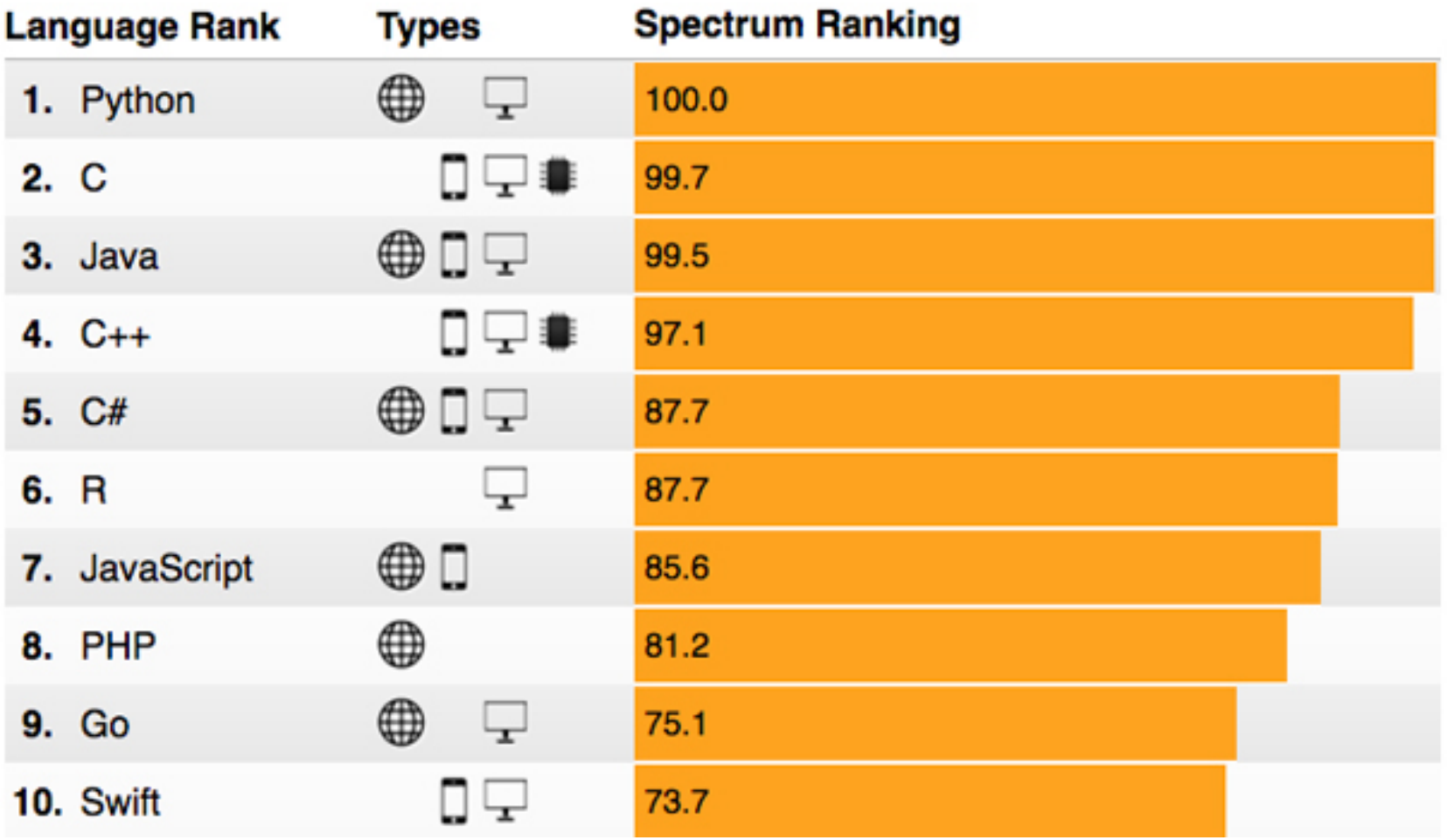
Object-oriented Programming Language

- Simula
- Smalltalk
- Ada
- C++
- Java
- C#

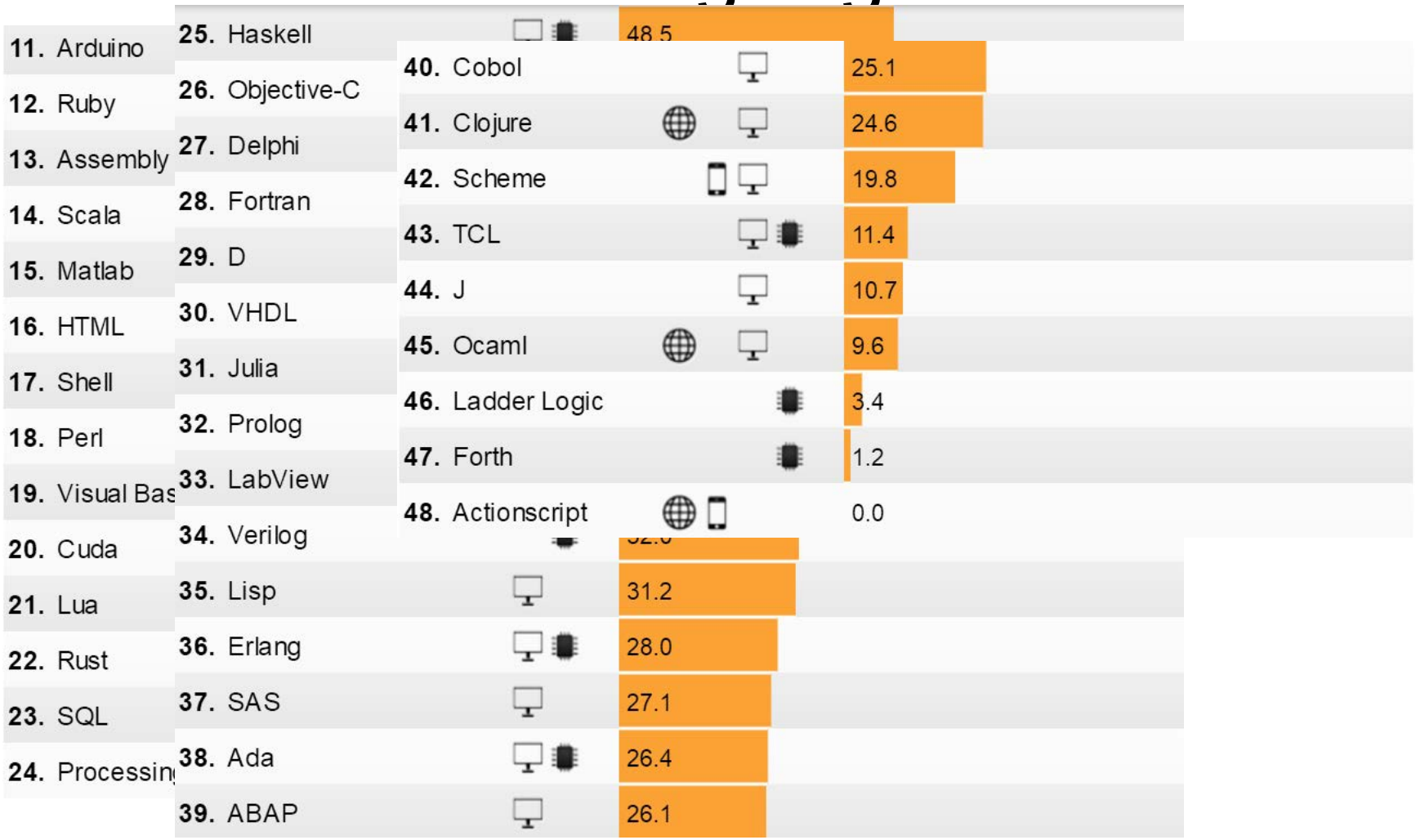
Major object technology milestones



The 2017 Top Programming Languages



The 2017 Top Programming Languages



C++

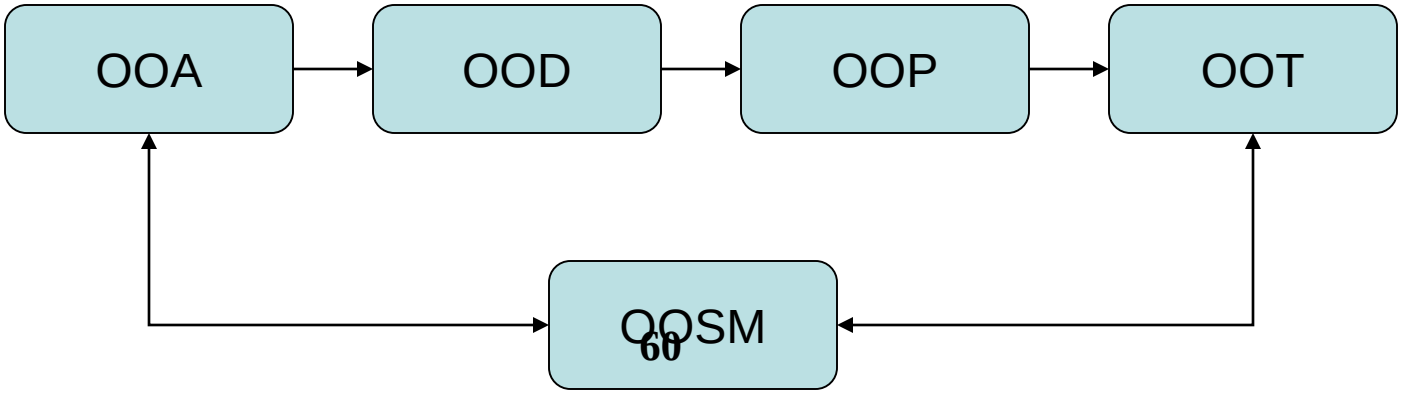
- Advantages
  - Anything can be used at any level of the operating system
  - Probably the fastest existing object-oriented language
  - So many popular C++ compilers, and so many C++ programmers
- Disadvantages
  - It is difficult to ensure software security at runtime due to Extensive use of pointers
- Currently, there are many mainstream UML tools support C++

JAVA

- Features
  - Platform-independent, high portability
  - Garbage collection mechanism
  - No pointer
- Currently, there are also many mainstream UML tools support Java

Object-oriented Programming Language

- These views have been generally accepted
  - Programming is not the hardest part of the software development
  - Requirement analysis and design is essential
  - The focus of software development should not only be aligned with the programming phase



UML & Object-oriented Programming Language

- UML is not a programming language
- Some UML tools support automatic code generation for certain object-oriented programming languages

The role of UML in System Modeling

- Throughout the entire software life cycle
- Can be used for each stage
- The key features of object-oriented can be expressed in UML

Summary

- Software crisis
- Software Life Cycle
- Object-oriented technology