3.0 Object-Oriented Modeling Using UML

Subject/Topic/Focus:

- Introduction to UML

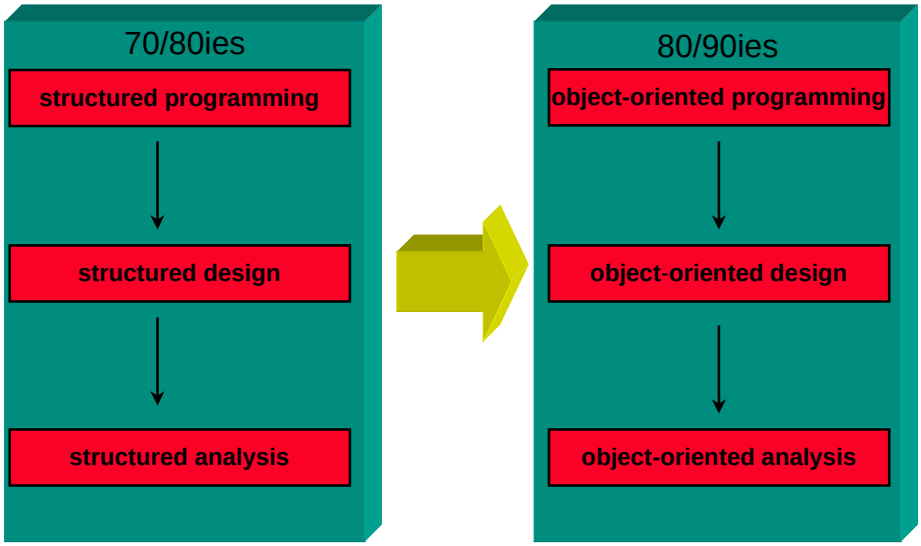
Summary:

- History of OOAD leading to UML
- UML Diagrams: Overview
- UML Models in the Objectory Software Development Process

Literature:

- Fowler

Evolution of OO Development Methods



History of OOAD leading to UML

1970 First object-oriented languages (Simula-67, Smalltalk).

1980 More than 50 different OOAD languages cause the users trouble to find complete and appropriate tools.

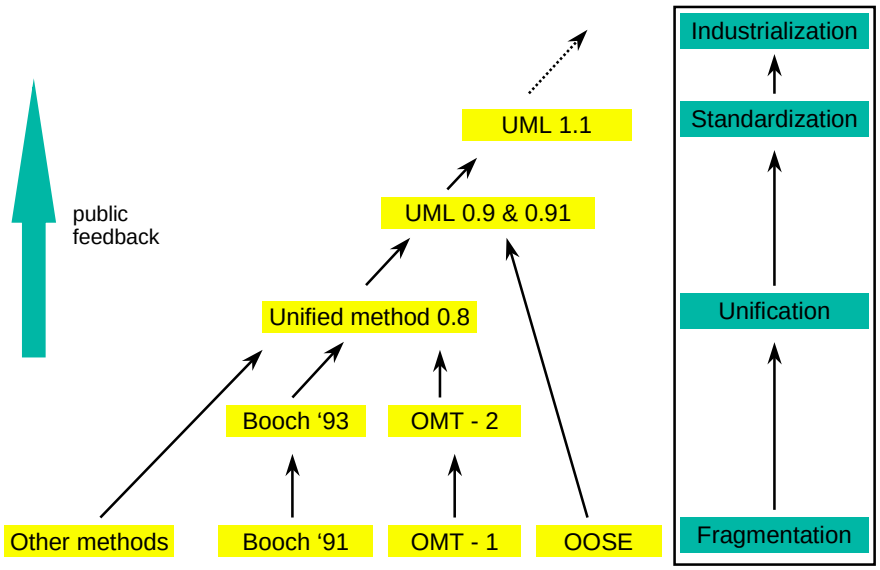
1992 New iterations of methods appear.
Booch '93, OOSE (Jacobson), OMT-2 (Rumbaugh)

1995 Unification, UML 0.9 by Booch, Rumbaugh

1997 Standardization, UML 1.1 by Booch, Rumbaugh, Jacobson

Object Management Group (OMG) adapts UML as OOAD standard

History of UML



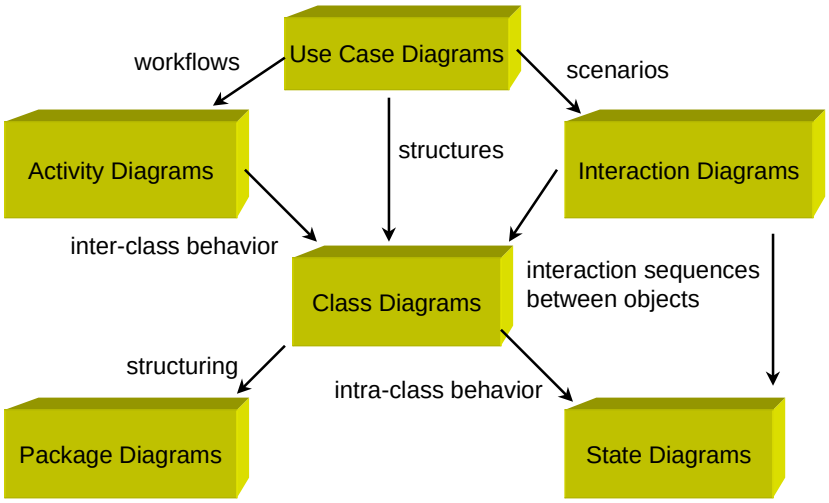
The Unified Modeling Language

- Booch and Rumbaugh started working towards a Unified Modeling Language (UML) in 1994 under the auspices of Rational Inc.
- UML only offers a model notation, not a methodology for how to do modeling.
- UML is used by the development method Objectory (Jacobson at Rational).
- UML was proposed by Rational Inc. and by Hewlett-Packard as a standard for object-oriented analysis and design and was adopted by the OMG.
- Vendors modify their CASE tools to make them consistent with UML.

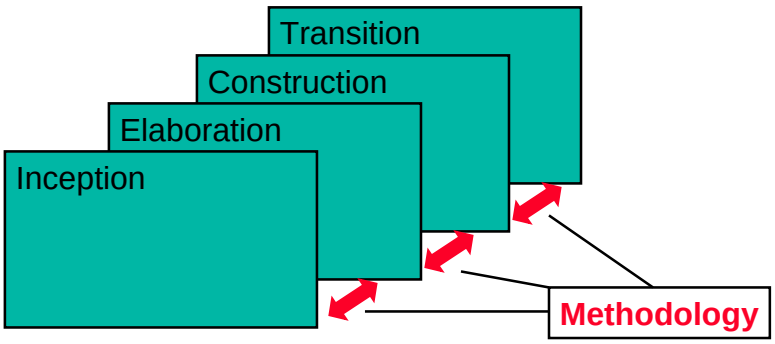
UML Diagrams (1)

❍ Use Case Diagrams	Nodes: Actor, Use (case) Links: Involvement, Extension, Usage
❍ Class Diagrams	Nodes: Class Links: Association, Generalization
❍ Interaction Diagrams	Nodes: Object Links: Message, Lifeline
❍ State Diagrams	Nodes: State, Sub-State Links: Transition
❍ Activity Diagrams	Nodes: Activity Links: Guard, Synchronization
❍ Package Diagrams	Nodes: Package Links: Dependency
❍ Deployment Diagrams	Nodes: Processor, Node Links: Dependency

UML Diagrams (2)

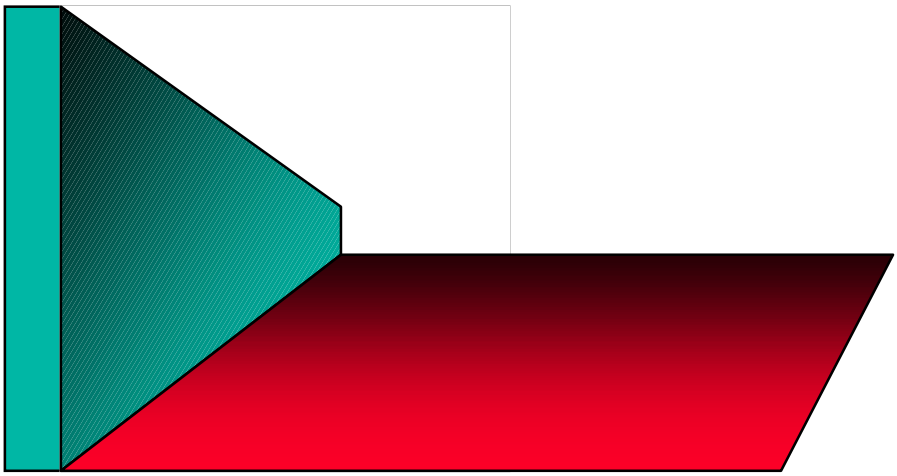


Objectory and UML



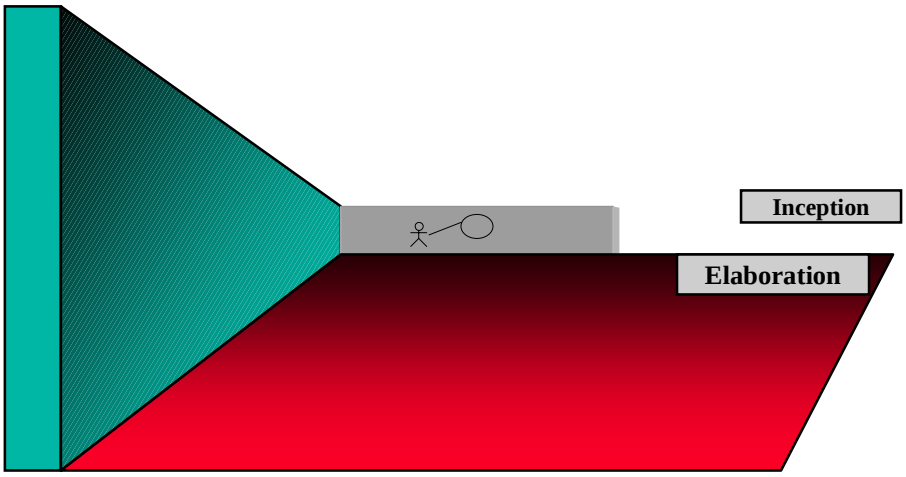
Software development is a process in **phases**.
This process has to follow a **methodology**.
Each phase is supported by graphs & **diagrams**.
There are different kinds of documents and various usage of them.
UML is an essential **language** for diagrams, offering computer support as well as the right patterns for the various stages of refinement and viewpoints.

Diagrams and Process



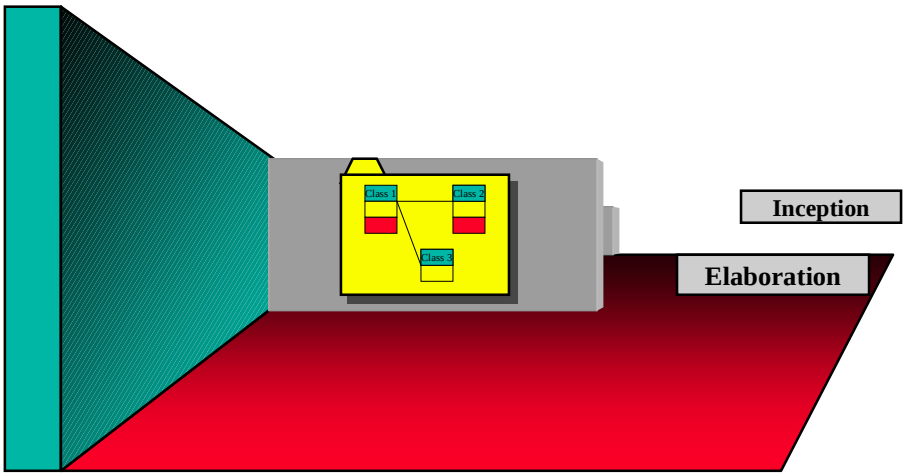
Diagrams and Process

Use Case Diagrams



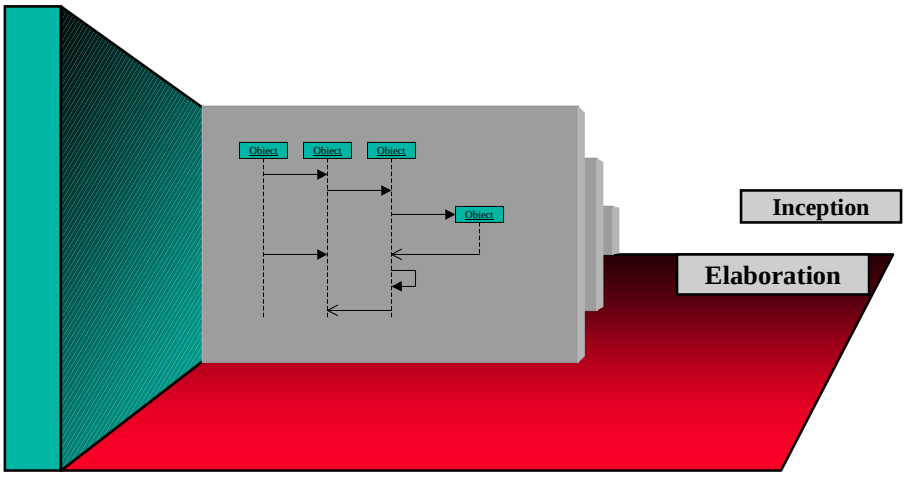
Diagrams and Process

Class & Package Diagrams



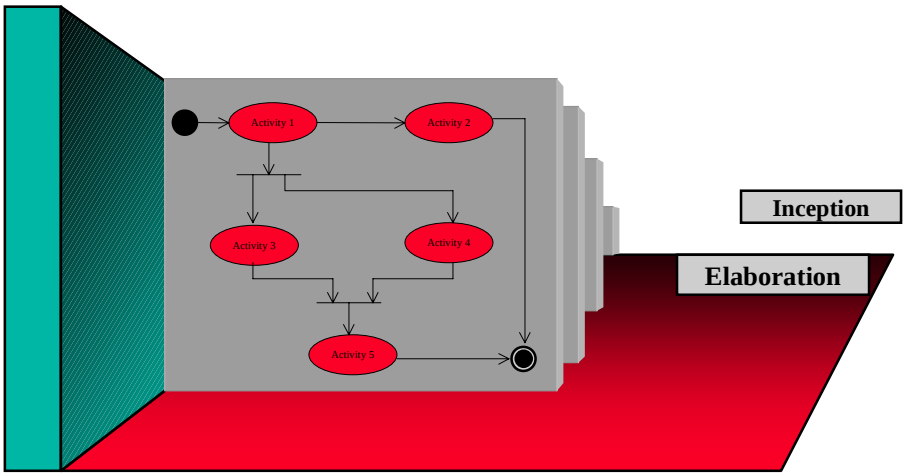
Diagrams and Process

Interaction Diagrams (Scenarios)



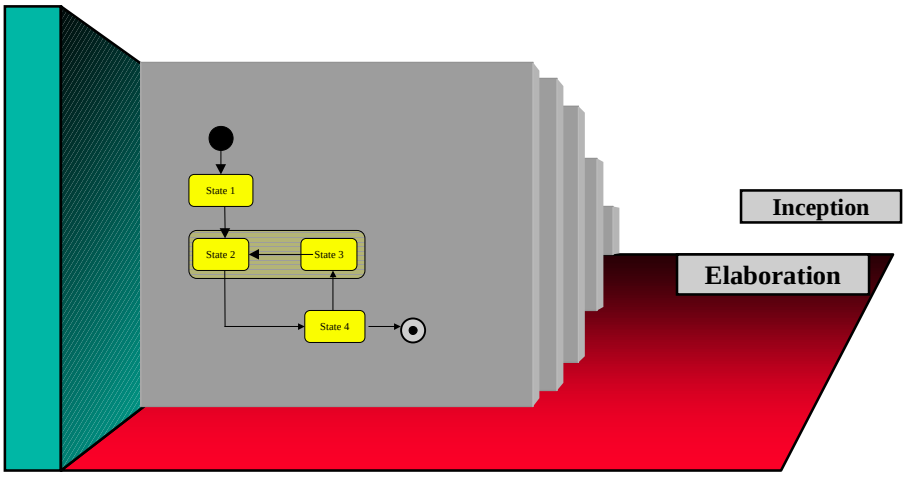
Diagrams and Process

Activity Diagrams (Workflow, Interclass Behavior)



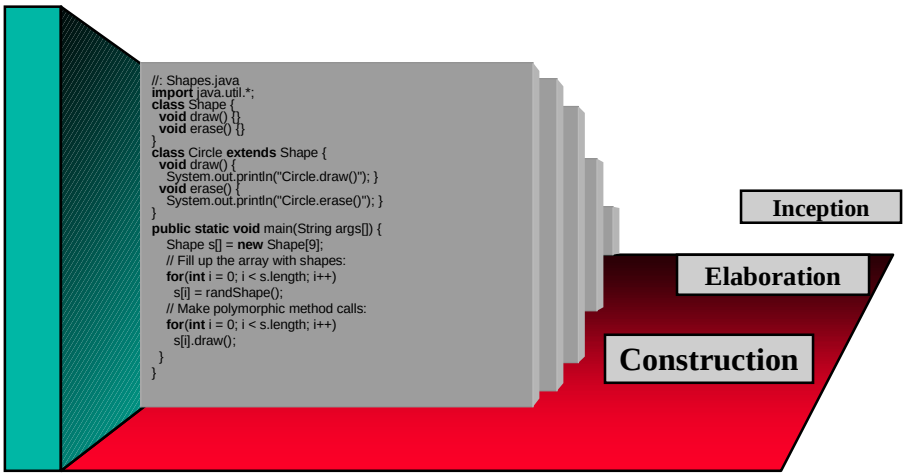
Diagrams and Process

State Transition Diagrams (Intraclass Behavior)



Texts and Process

Source Code



Diagrams and Process

Deployment Diagrams