Development Methodologies

Prof. Dr. Josef M. Joller
jjoller@hsr.ch

SCOPE

Outline

Historical aspects

Economic aspects

Maintenance aspects

Specification and design aspects

Team programming aspects

The object-oriented paradigm

Terminology

Scope of Software Engineering

Historical Aspects

- 1968 NATO Conference, Garmisch
- Aim: to solve the “Software Crisis”
- Software is delivered
  - Late
  - Over budget
  - With residual faults

Scope of Software Engineering (contd)

Why cannot bridge-building techniques be used to build operating systems?

- Attitude to collapse
- Imperfect engineering
- Complexity
- Maintenance

Economic Aspects

Economically viable techniques

Coding method CM_{new} is 10% faster than currently used method CM_{old}. Should it be used?

- Common sense answer
  - Of course!
- Software Engineering answer
  - Consider the effect of CM_{new} on maintenance

Maintenance Aspects

Software Life Cycle

- The way we produce software, including
  - The life-cycle model
  - The individuals
  - CASE tools

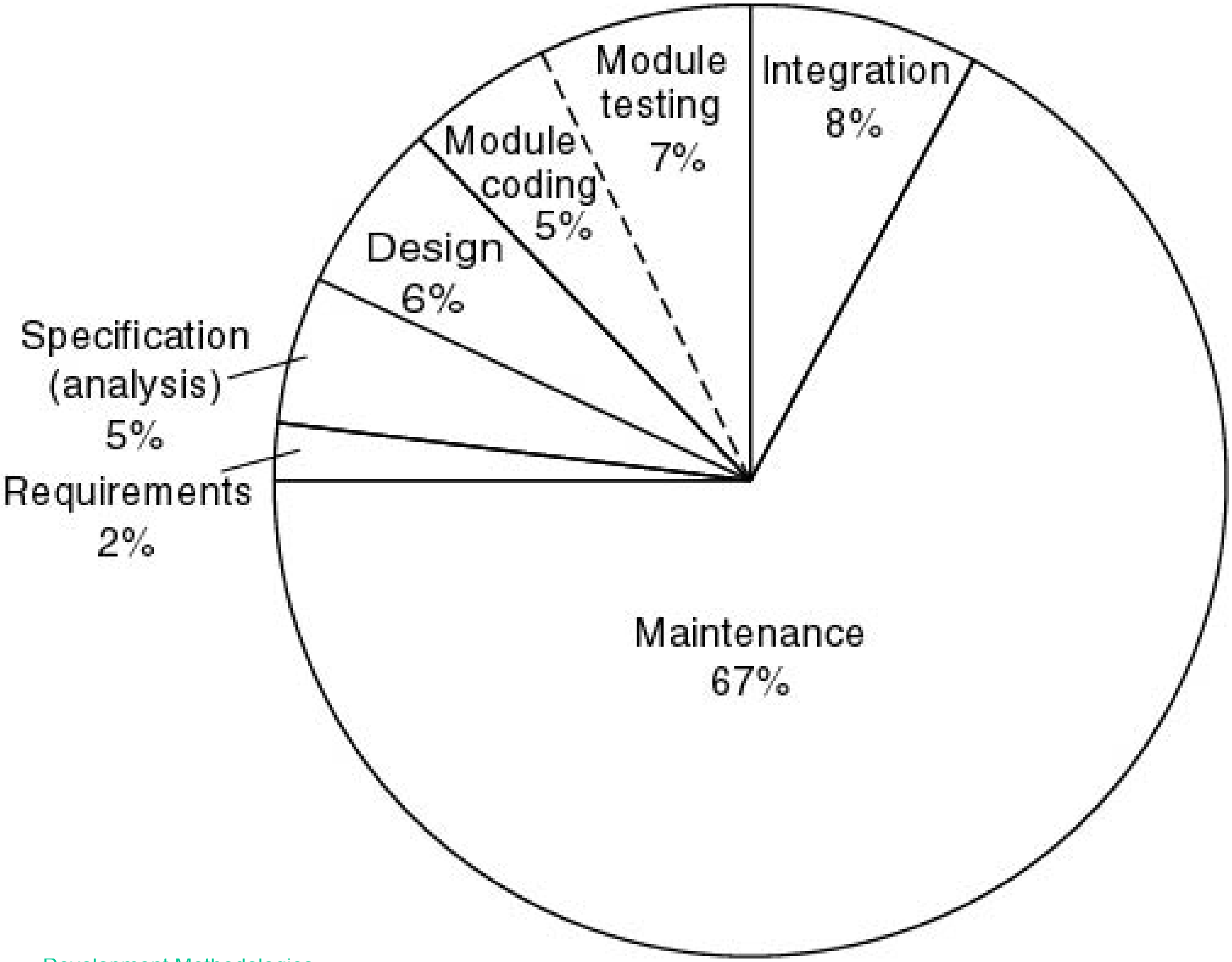
Life-cycle model

1. Requirements phase
2. Specification phase
3. Design phase
4. Implementation phase
5. Integration phase (in parallel with 4)
6. Maintenance phase
7. Retirement

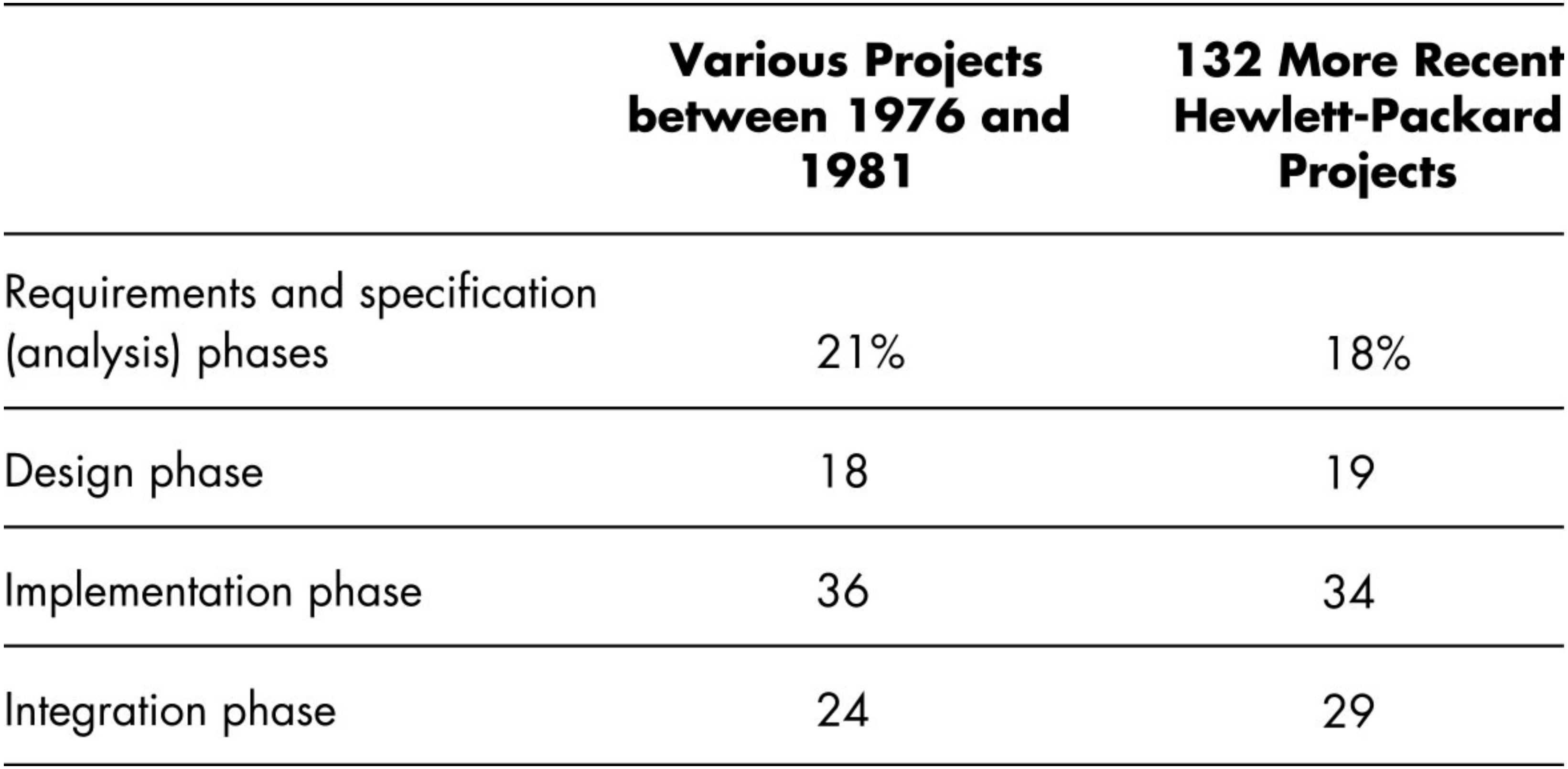
Approximate Relative Cost of Each Phase

1976–1981 data

Maintenance constitutes 67% of total cost



Comparative Relative Cost of Each Phase



	Various Projects between 1976 and 1981	132 More Recent Hewlett-Packard Projects
Requirements and specification (analysis) phases	21%	18%
Design phase	18	19
Implementation phase	36	34
Integration phase	24	29

Good and Bad Software

Good software is maintained—bad software is discarded

Different types of maintenance

- Corrective maintenance [about 20%]
- Enhancement
    - Perfective maintenance [about 60%]
    - Adaptive maintenance [about 20%]

Effect of CMnew on maintenance

- How much can we improve the maintenance phase?

Specification and Maintenance Faults

60 to 70 percent of faults are specification and design faults

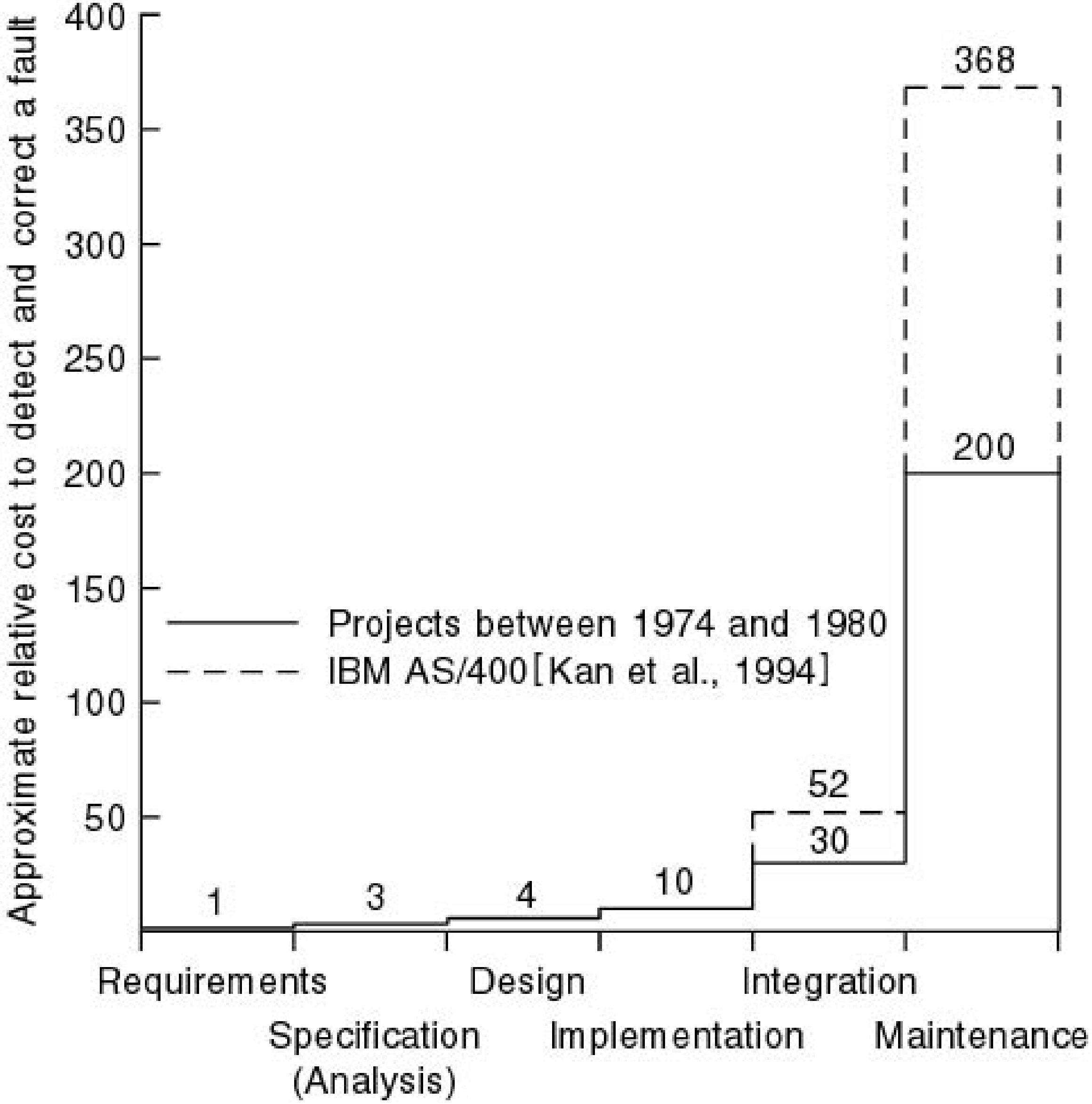
Data of Kelly, Sherif, and Hops [1992]

- 1.9 faults per page of specification
- 0.9 faults per page of design
- 0.3 faults per page of code

Faults at end of the design phase of the new version of the product

- 13% of faults from previous version of product
- 16% of faults in new specifications
- 71% of faults in new design

Cost to Detect and Correct a Fault



Team Programming Aspects

Hardware is cheap

- We can build products that are too large to be written by one person in the available time

Teams

- Interface problems
- Meetings
- Qualification
- New technologies
- Fast changing business environment

The Object-Oriented Paradigm

The structured paradigm had great successes initially

- It started to fail with larger products (> 50,000 LOC)

Maintenance problems (today, up to 80% of effort)

Reason: structured methods are

- Action oriented (finite state machines, data flow diagrams); or
- Data oriented (entity-relationship diagrams, Jackson’s method);
- But not both (Objects are!)

The Object-Oriented Paradigm (contd)

Both data and actions are of equal importance

Object:

- Software component that incorporates both data and the actions that are performed on that data

Example:

- Bank account
  - Data: account balance
  - Actions: deposit, withdraw, determine balance

Key Aspects of Object-Oriented Solution

Conceptual independence

- Encapsulation

Physical independence

- Information hiding

Impact on development

- Physical counterpart

Impact on maintenance

- Independence effects

Responsibility-Driven Design

Also called "Design by Contract"

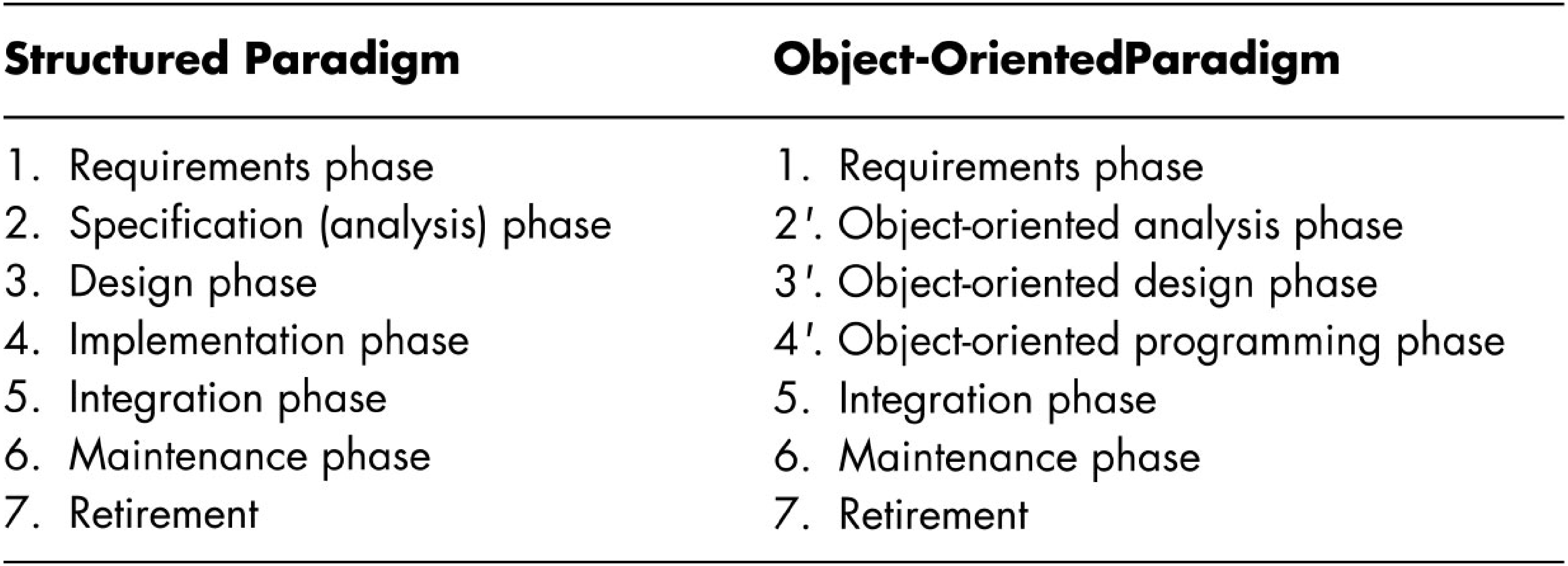
Send flowers to your aunt in Iowa City

- Call 1-800-FLOWERS
- Where is 1-800-FLOWERS?
- Which Iowa City florist does the delivery?
- Information hiding

Object-oriented paradigm

- "Send a message to a method [action] of an object"

Transition From Analysis to Design



Structured Paradigm	Object-OrientedParadigm
1. Requirements phase	1. Requirements phase
2. Specification (analysis) phase	2′. Object-oriented analysis phase
3. Design phase	3′. Object-oriented design phase
4. Implementation phase	4′. Object-oriented programming phase
5. Integration phase	5. Integration phase
6. Maintenance phase	6. Maintenance phase
7. Retirement	7. Retirement

Structured paradigm:

- Jolt between analysis (what) and design (how)

Object-oriented paradigm:

- Objects enter from very beginning

Analysis/Design "Hump"

Systems analysis

- Determine **what** has to be done

Design

- Determine **how** to do it
- Architectural design—determine modules
- Detailed design—design each module

Removing the “Hump”

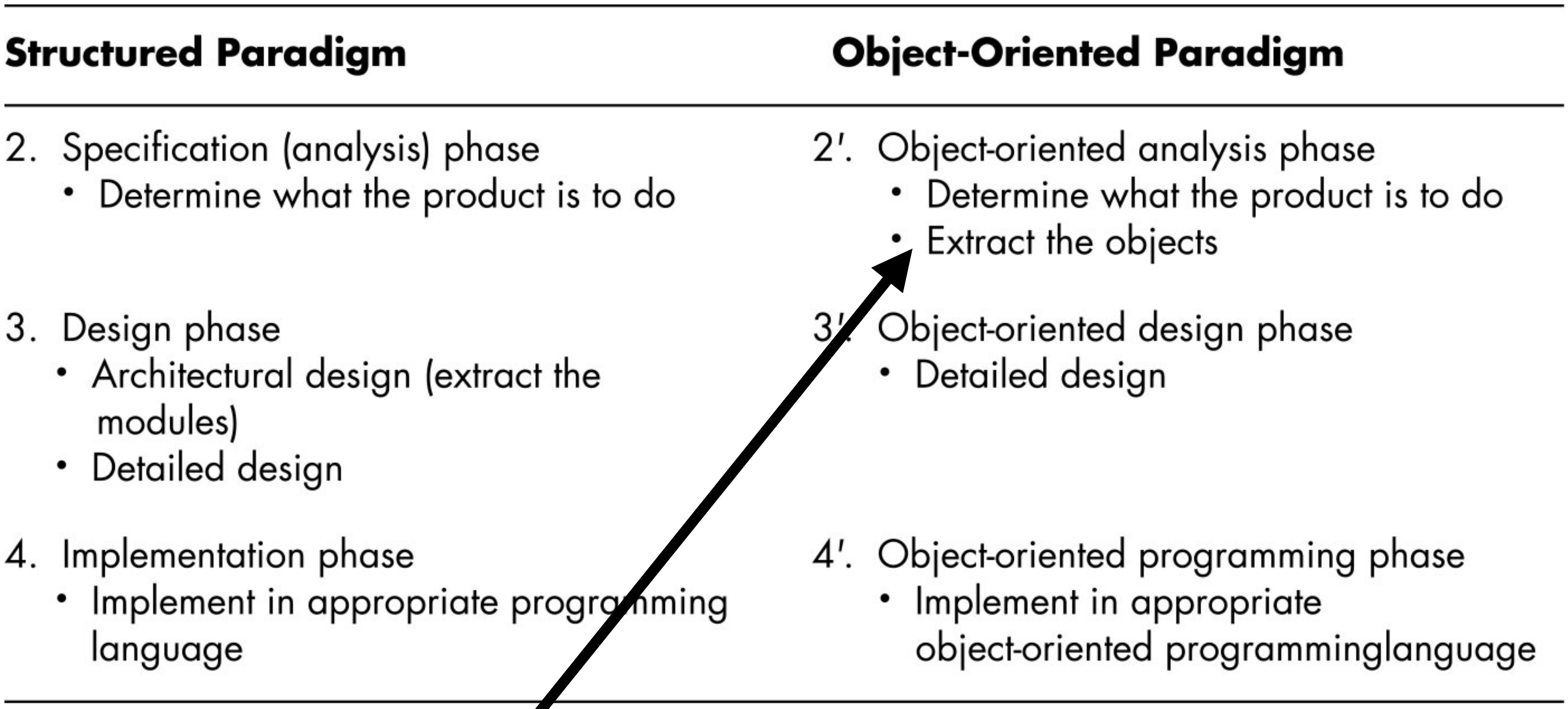
Object-oriented analysis

- Determine **what** has to be done
- *Determine the objects*

Object-oriented design

- Determine **how** to do it
- *Design the objects*

In More Detail



Structured Paradigm	Object-Oriented Paradigm
2. Specification (analysis) phase • Determine what the product is to do	2′. Object-oriented analysis phase • Determine what the product is to do • Extract the objects
3. Design phase • Architectural design (extract the modules) • Detailed design	3′. Object-oriented design phase • Detailed design
4. Implementation phase • Implement in appropriate programming language	4′. Object-oriented programming phase • Implement in appropriate object-oriented programminglanguage

Objects enter here