Section 4
Detailed Object Design

1. Overview
2. Reusing pattern solutions
3. Specifying interfaces

Section 4.1
Detailed Object Design Overview

1. Purpose
2. Work products
3. Breakdown

4.1.1 Purpose of Detailed Object Design

- To specify the solution domain
- To close the gap between:
  - application domain objects
  - commercial off-the-shelf (COTS) components
- How?
  - identify solution domain objects
- This is still not an algorithmic activity
  - creativity is important...

Purpose (cont.)

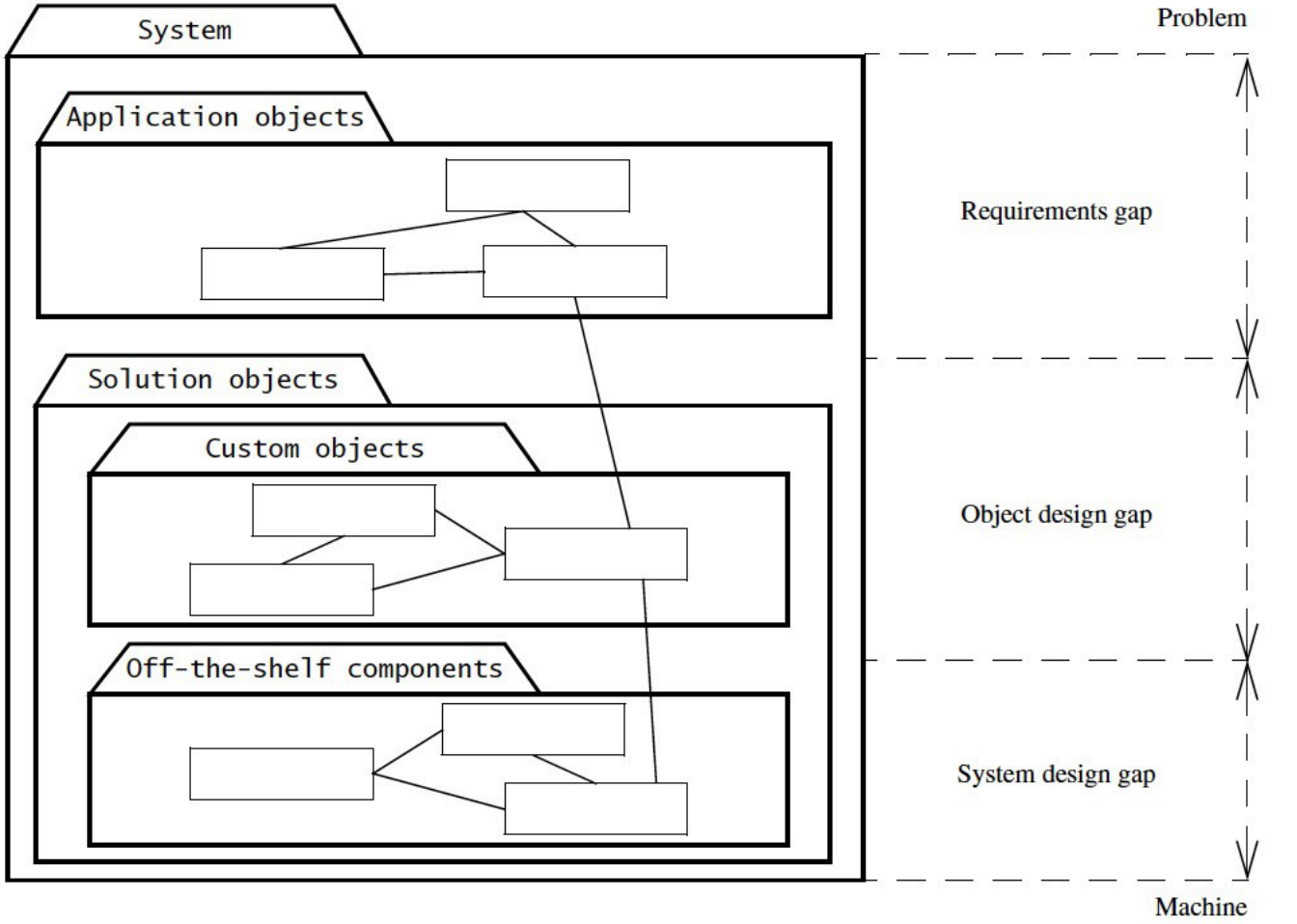
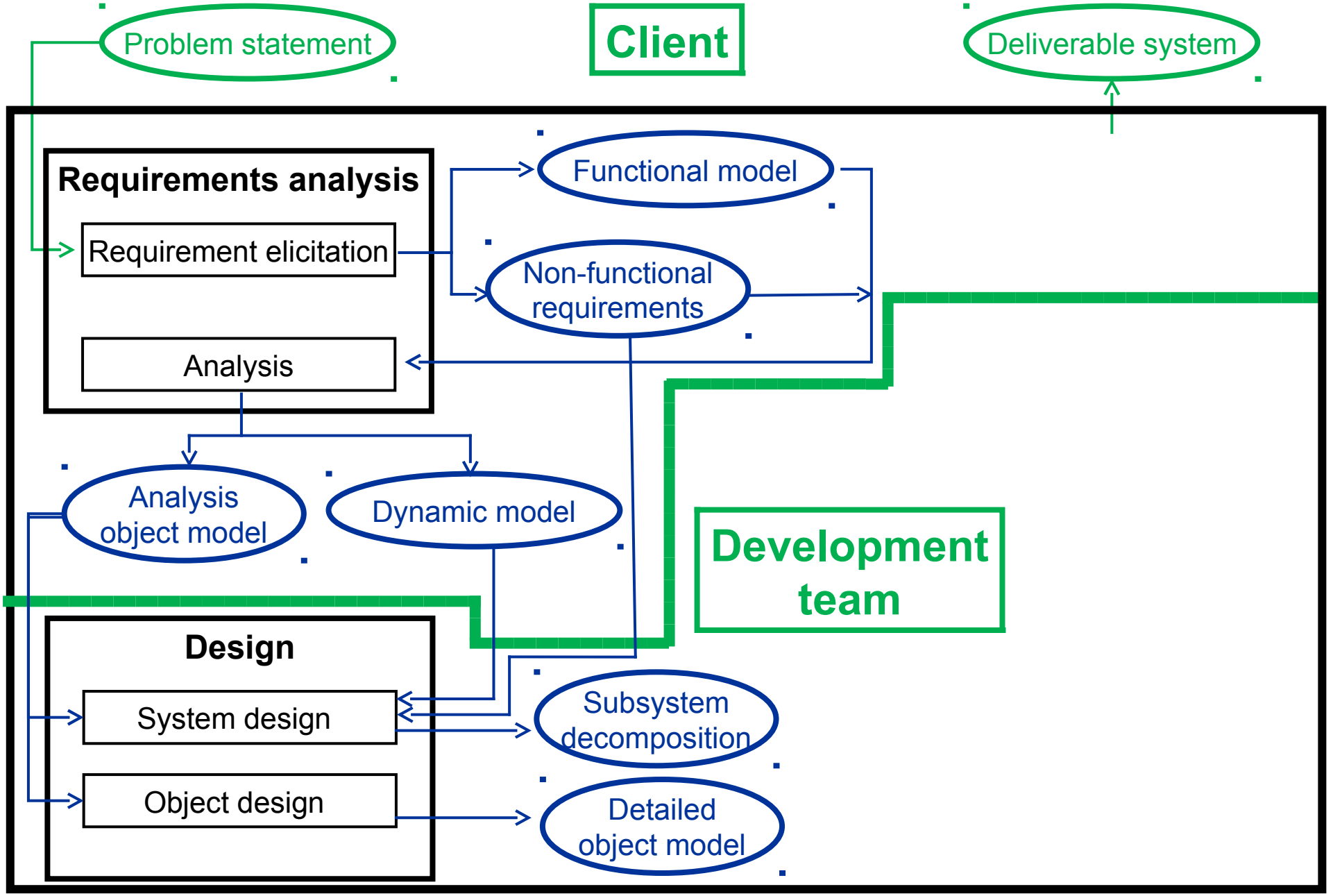


Figure 8-1 Object design closes the gap between application objects identified during requirements and off-the-shelf components selected during system design (stylized UML class diagram).

Purpose (cont.)

- Input to detailed object design
  - analysis object model
  - subsystem design model
    - subsystem decomposition
    - system architecture strategies
- Output of detailed object design
  - detailed object model

4.1.2 Work Products



Detailed Object Design Tasks

- Main tasks
  - identify opportunities for software reuse
    - additional COTS components
    - design patterns
  - specify services
    - interface specifications
  - restructure object model
    - improve object model for understandability and maintainability
  - optimize object model
    - improve object model to meet performance requirements

- Object design tasks occur:
  - concurrently
  - iteratively

4.1.3 Breakdown

- Detailed object design consists of two parts:
  - reuse
    - reusing existing components and patterns
  - specifying class interfaces
    - operations, type signatures, contracts

Breakdown (cont.)

- Reuse
  - COTS
    - class libraries
    - additional components for data structures and services
  - existing design patterns
    - may need to be adapted

Breakdown (cont.)

- Specifying class interfaces
  - subsystems are specified in terms of class interface
    - public attributes and operations
    - operations, arguments, type signatures
    - exceptions
  - output
    - complete interface specification for each subsystem
    - also called Application Programming Interface (API)