


Chapter

1

*Software
Engineering
Principles*



Third Edition

C⁺⁺ *Plus* Data Structures

Nell Dale



The Software Life Cycle

- Problem analysis
- Requirements elicitation
- Software specification
- High- and low-level design
- Implementation
- Testing and Verification
- Delivery
- Operation
- Maintenance



Software Engineering

- **A disciplined approach to the design, production, and maintenance of computer programs**
- **that are developed on time and within cost estimates,**
- **using tools that help to manage the size and complexity of the resulting software products.**



Programmer ToolBoxes

- Hardware—the computers and their peripheral devices
- Software—operating systems, editors, compilers, interpreters, debugging systems, test-data generators, and so on
- Ideaware – shared body of knowledge



An Algorithm Is . . .

- A logical sequence of discrete steps that describes a complete solution to a given problem computable in a finite amount of time.



Goals of Quality Software

- It works.
- It can be modified without excessive time and effort.
- It is reusable.
- It is completed on time and within budget.



Detailed Program Specification

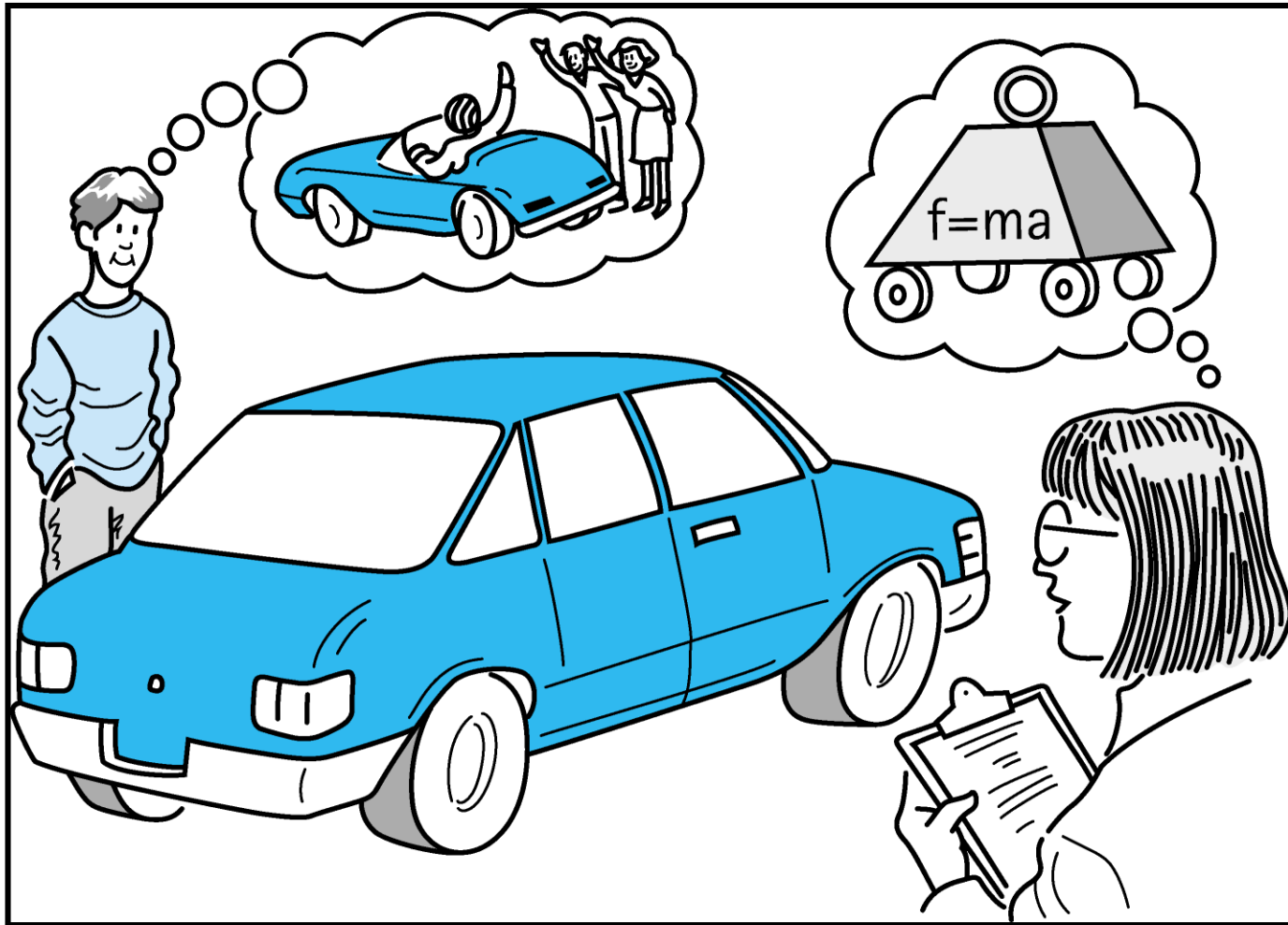
- Tells *what* the program must do, but *not how* it does it.
- Is written documentation about the program.



Abstraction

- A model of a complex system that includes only the details essential to the perspective of the viewer of the system.
- Programs are abstractions.

Abstraction (cont.)





Information Hiding

- The practice of hiding the details of a module with the goal of controlling access to the details from the rest of the system.
- ~~A programmer can concentrate on one module at a time.~~
- ~~Each module should have a single purpose or identity.~~



Stepwise Refinement

- A problem is approached in stages. Similar steps are followed during each stage, with the only difference being the level of detail involved. Some variations:
 - Top-down
 - Bottom-up
 - Functional decomposition
 - Round-trip gestalt design



Visual Aids – CRC Cards

Class Name:	Superclass:	Subclasses:
Primary Responsibility		
Responsibilities	Collaborations	



Procedural vs. Object-Oriented Code

“Read the specification of the software you want to build. Underline the verbs if you are after procedural code, the nouns if you aim for an object-oriented program.”

Grady Booch, “What is and Isn’t Object Oriented Design,” 1989.



Two Approaches to Building Manageable Modules

FUNCTIONAL DECOMPOSITION

Divides the problem into **more easily handled subtasks**, until the functional modules (subproblems) can be coded.

FOCUS ON: processes

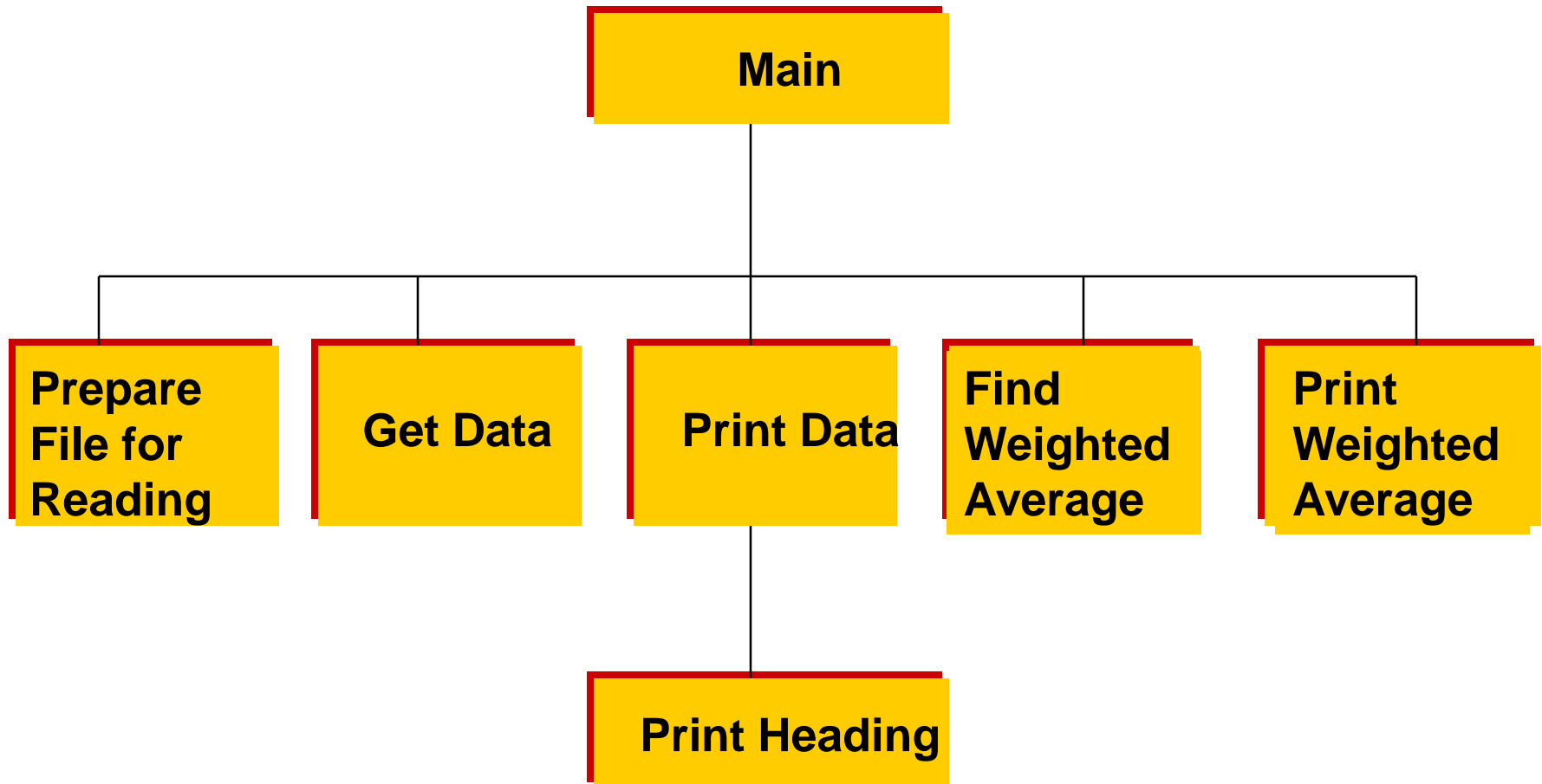
OBJECT-ORIENTED DESIGN

Identifies various **objects composed of data and operations**, that can be used together to solve the problem.

FOCUS ON: data objects



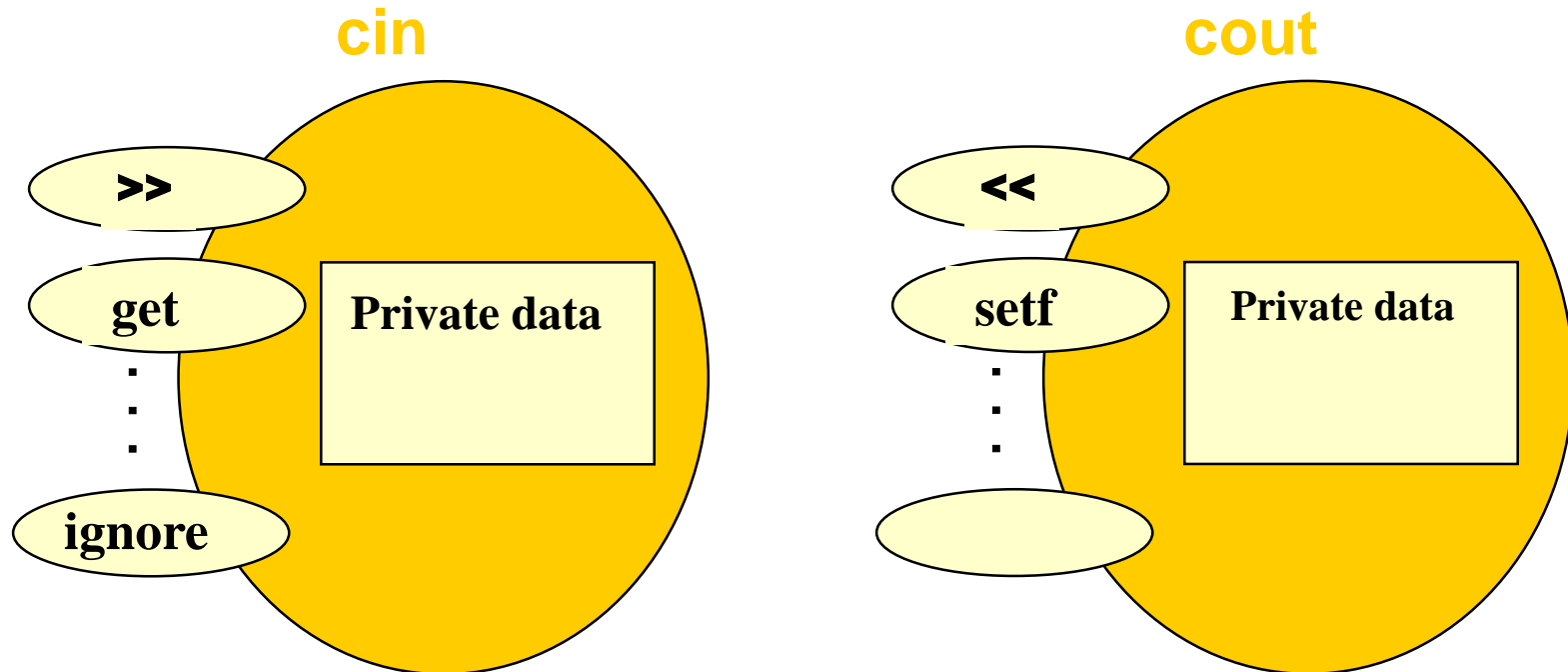
Functional Design Modules





Object-Oriented Design

A technique for developing a program in which the solution is expressed in terms of objects -- self-contained entities composed of data and operations on that data.





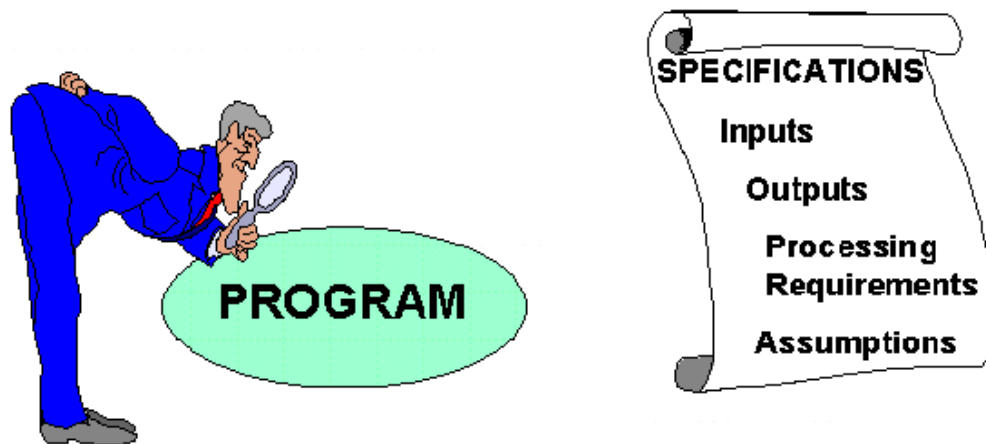
For Each Test Case:

- Determine inputs.
- Determine the expected behavior of the program.
- Run the program and observe the resulting behavior.
- Compare the expected behavior and the actual behavior.



Program Verification

- Program Verification is the process of determining the degree to which a software product fulfills its specifications.





Verification vs. Validation

Program verification asks,

“Are we doing the job right?”

Program validation asks,

“Are we doing the right job?”

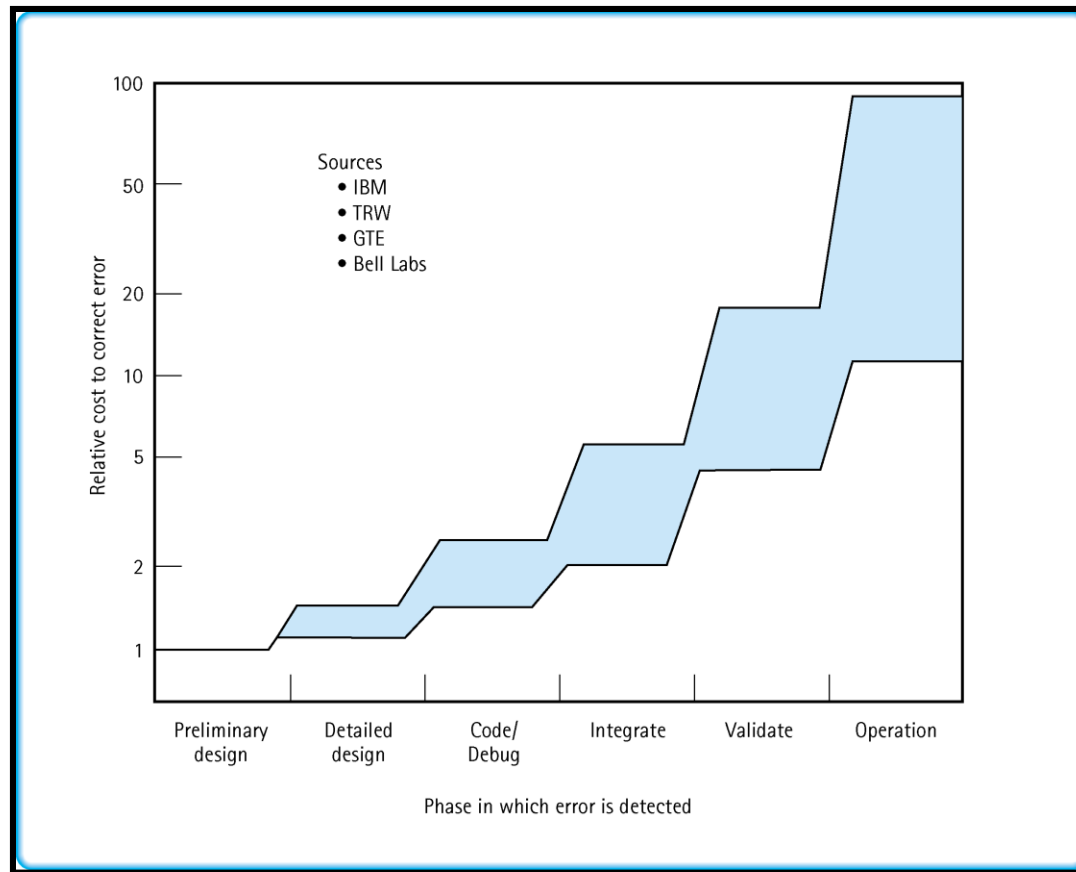
*B.W. Boehm, Software Engineering Economics,
1981.*



Types of Errors

- Specification
- Design
- Coding
- Input

Cost of a Specification Error Based on When It Is Discovered





Controlling Errors

Robustness The ability of a program to recover following an error; the ability of a program to continue to operate within its environment

Preconditions Assumptions that must be true on entry into an operation or function for the postconditions to be guaranteed.

Postconditions Statements that describe what results are to be expected at the exit of an operation or function, assuming that the preconditions are true.