Introduction

The classic distinction between analysis and design is that analysis describes *what* a system must do and design describes *how* to do it; roughly speaking, analysis decisions are implementation-independent, design decisions tend to be implementation-dependent. During analysis we are concerned with understanding and modelling the users' requirements; analysis produces a specification of what the new system will do. During design we decide how to construct the system that will deliver the users' requirements; actually constructing the system is an implementation activity. The design is effectively an abstraction of the final code; it will omit much of the detail in the code but, nevertheless, will provide the essence of the structure and interactions of the programs. If we use a suitable CASE tool we can generate much of the code from the design models.

Design activities include those concerned with overall system design, and those concerned with detailed design. Overall system design activities include designing the overall architecture of the system, selecting a strategy for coping with persistent data and designing a user interface. Software and hardware platforms must be chosen and a test plan produced. Designing a test plan and choosing the software and hardware are beyond the scope of this book.

Detailed design activities include the detailed specifications of classes, their relationships and interactions. The models should contain sufficient information to allow them to be used as program specifications. Detailed design activities are discussed in Chapter 10.

The product of design activities will be a layered diagram of the system architecture, a set of component and deployment diagrams, a database definition, a test plan and screen and report layouts, a set of detailed class diagrams and supporting documentation such as data dictionary and operation specifications and a set of detailed interaction diagrams.

In this chapter we discuss the overall architecture in terms of a layered view of the system. We discuss packages and dependencies: how to use packages to manage large class diagrams and to arrange the system in layers. We introduce two new types of diagram, component diagrams and deployment diagrams, which can be used to specify the arrangement of the hardware components and the assignment of software components to hardware. The component diagram describes the physical software components of the system and their dependencies: such things as executable files, databases and libraries of classes. The deployment diagram shows the hardware components of the system, i.e.