Chapter II

The Objects Model and the Class Diagram

This chapter describes in detail the components of the objects model (including objects, classes, attributes, relationships, and functions), and the class diagram which represent them.

Similarities and Differences Between the Objects Model and the Entity Relationship Model

The objects model (or object oriented [OO] model) is a conceptual-application model that is used to define a database schema representing a certain reality. The model views the world as consisting of objects belonging to classes. The objects of these classes have attributes, behavior (i.e., functions), and various relationships with other objects.

The objects model can be presented as a class diagram (also termed OO diagram or objects diagram). Like an entity relationship diagram (ERD), the class diagram has two main goals:

1. To serve as a communication medium between the developers (analysts/designers) and the users or their representatives. The diagram is created as a result of the interactions between the two parties, during which they
discover and define the users’ information needs; the diagram serves like a contract between these two sides which summarizes the users’ needs.

2. To be the basis for further development of the information system (IS). Based on the diagram, it should be possible to design the database schema of the application, and (partially) the functions that it will have to perform. For that, it is necessary to transform the class diagram into an equivalent verbal description—an objects schema. This is done using an object definition language (ODL), similar to data definition language (DDL) in the relational model. In principle, all components of the class diagram are mapped to the objects schema. However, the objects schema includes more details which are not included in the diagram. For example, in the diagram each attribute has a name, and some attributes may have specific constraint definitions (e.g., key, unique); in the objects schema there are more detailed definitions, including the attributes’ domains or data types (e.g., numeric, char., real, etc.) and lengths. Another example, in the class diagram, we only write the names of the classes’ functions, while in the objects schema we specify the parameters of the functions.

As aforementioned, there is a great deal of similarity between the OO and ER models and diagrams, since the ER model is one of the sources from which the objects model originated. But there are differences between the two models, which we will review later on. One of these differences is that the ER model is “static,” that is, it only deals with the data structure, while the objects model also includes “behavior,” that is, the functions that operate on the data.

The rest of this chapter is dedicated to describing the components of the objects model and the class diagram. The description is organized in four main categories: objects and classes, attributes, relationships, and functions.

### Objects and Classes

#### Object

An object represents a thing that exists in the real world, tangible or intangible. In the real world there are many different things, but we are only interested in those which are relevant to the business or the organization, and for which data needs to be stored in the IS, as based on the users’ needs.

An object can be tangible, for example, a student, an employee, an item or a book. But it can also be an intangible thing, for example, a purchase order, a delivery