CASE Defined

We have mentioned CASE (Computer-Aided Software Engineering) in previous chapters. Although its meanings can vary, CASE is traditionally defined as “a comprehensive label for software designed to use computers in all phases of computer development, from planning and modeling through coding and documentation. CASE represents a working environment consisting of programs and other development tools that help managers, systems analysts, programmers, and others automate the design and implementation of programs and procedures for business, engineering, and scientific computer systems.” CASE became popular during the late 1970s as a way of automating and integrating modeling tools. It also allows for the creation and maintenance of data repositories which provide organizations with a tool to establish a central place to store all of their data elements. Figure 10.1 shows the common components of most integrated CASE products.

Data Repository Inputs

The design of CASE is based on inputs which create entries into the data repository. A data repository can be defined as a “robust data dictionary.” A data dictionary essentially provides the definition of the data element itself. A data repository, on the other hand, stores information relating to the data element’s behavior. This can include the element’s stored procedures, descriptions, and documentation about how and where it is used. Therefore, the data dictionary can actually be considered a subset of the data repository. Inputs to the data repository are typically the modeling tools discussed in Chapter 5 DFDs, ERDs, STDs, and process specifications as well as the object oriented analysis discussed in Chapter 11. These input capabilities can be summarized as follows.

---

Data Flow Diagrams (DFD)

Every named data flow will automatically require an entry into the data repository. This named flow may consist of other elements and therefore require functional decomposition down to the elementary data element level. Unique data elements that are components of data stores will also generate entries into the data repository. Figures 10.2 and 10.3 show these interfaces in the Popkin System Architect CASE product.

Entity Relational Diagrams (ERD)

The ERD interfaces with the data repository by mapping an element to particular tables (entities), that is, each unique data element within an entity points to a data repository definition (see Figure 10.4).

State Transition Diagram (STD)

The STD interfaces with the data repository via the “conditions that cause a change in state.” These condition arrows map to the data repository and allow component elements to be defined as shown in Figure 10.5.