**4. Analysis Modeling**

At a technical level, software engineering begins with a series of modeling tasks that lead to a complete specification of requirements and a comprehensive design representation for the software to be built. The analysis model, actually a set of models, is the first technical representation of a system. Over the years many methods have been proposed for analysis modeling. However, two now dominate. The first; structured analysis is a classical modeling method and object oriented analysis. Analysis is a model building activity. Applying the operational analysis principles (The information domain of a problem must be represented and understood, the functions that the software is to perform must be defined, the behavior of the software (as a consequence of external events) must be represented, the models that depict information, function, and behavior must be partitioned in a manner that uncovers detail in a layered (or hierarchical) fashion, the analysis process should move from essential information toward implementation detail.) We create and partition data, functional, and behavioral models that depict the essence of what must built.

**4.1 The Elements of the Analysis Model**

The analysis model must achieve three primary objectives: (1) to describe what the customer requires, (2) to establish a basis for the creation of a software design, and (3) to define a set of requirements that can be validated once the software is built. To accomplish these objectives, the analysis model derived during structured analysis takes the form illustrated in Figure 4.1.

At the core of the model lies the data dictionary—a repository that contains descriptions of all data objects consumed or produced by the software. Three different diagrams surround the core. The entity relation diagram (ERD) depicts relationships between data objects. The ERD is the notation that is used to conduct the data modeling activity. The attributes of each data object noted in the ERD can be described using a data object description. The data flow diagram (DFD) serves two purposes: (1) to provide an indication of how data are transformed as they move through the system and (2) to depict the functions (and sub functions) that transform the data flow. The DFD provides additional information that is used during the analysis of the information domain and serves as a basis for the modeling of function. A description of each function presented in the DFD is contained in a process specification (PSPEC). The state transition diagram (STD) indicates how the system behaves as a consequence of external events. To accomplish this, the STD represents the various modes of behavior (called states) of the system and the manner in which transitions are made from state lo state. The STD serves as the basis for behavioural modeling. Additional information about the control aspects of the software is contained in the control specification (CSPEC). The analysis model encompasses each of the diagrams, specifications, descriptions, and the dictionary noted in Figure 4.1.



Figure 4.1 the structure of the analysis model

The ERD enables software engineers to identify data objects and their relationships using a graphical notation.

**4.2 Data modeling**

Data modeling answers a set of specific questions that are relevant to any data processing application. What are the primary data objects to be processed by the system? What is the composition of each data object and what attributes describe the object? Where do the objects currently reside? What are the relationships between each object and other objects? What are the relationships between the objects and the processes that transform them? To answer these questions, data modeling methods make use of the entity relationship diagram. The ERD described in detail later in this section, enables software engineers to identify data objects and their relationships using a graphical notation. In the context of structured analysis, the ERD defines all data that are entered, stored, transformed, and produced within an application. The entity relationship diagram focuses only on data (and therefore satisfies the first operational analysis principles), representing a "data network" that exists for a given system. The ERD is especially useful for applications in which data and the relationships that govern data are complex. Unlike the data flow diagram (used to represent how data are transformed), data modeling considers data independent of the processing that transforms the data.

**4.2.1 Data Objects, Attributes, and Relationships**

The data model consists of three interrelated pieces of information: the data object, the attributes that describe the data object, and the relationships that connect data objects to one another.

**Data objects**: A data object is a representation of almost any composite information that must be understood by software. By composite information, we mean something that has a number of different properties or attributes. Therefore, width (a single value) would not be a valid data object, but dimensions (incorporating height, width, and depth) could be defined

as an object.

For example, a person or a car (Figure 4.2) can be viewed as a data object in the sense that either can be defined in terms of a set of attributes. The data object description incorporates the data object and all of its attributes. Data objects (represented in bold) are related to one another. For example, person can own car, where the relationship own connotes a specific "connection" between person and car.

**Attributes**: Attributes define the properties of a data object and take on one of three different characteristics. They can be used to (1) name an instance of the data object, (2) describe the instance, or (3) make reference to another instance in another table. In addition, one or more of the attributes must be defined as an identifier—that is, the identifier attribute becomes a "key" when we want to find an instance of the data object. In some cases, values for the identifier(s) are unique, although this is not a requirement. Referring to the data object car, a reasonable identifier might be the ID number.

**Relationships**: Data objects are connected to one another in different ways. Consider two data objects, book and bookstore. These objects can be represented using the simple notation illustrated in Figure 4.3. A connection is established between book and bookstore because the two objects are related.



Figure 4.2 Data objects, attributes and relationships



Figure 4.3 Relationships