Software Design Models and Patterns

Introduction
A software solution starts to form during the design phase by means of design models, with the aid of design patterns. Software design models have been evolving over some 50 years, and according to Roger S. Pressman (2005), they are largely categorized as either structured or object-oriented. Erich Gamma introduced software design patterns in 1995 (Gamma, Helm, Johnson, & Vlissides, 1995). These are generally applied to object-oriented solutions.

Design Models
Design models enable developers to model, from a process perspective, how to design application solutions. Pressman (2005) states that models generally define the process of design as a series of steps of refinement or elaboration. Within these models, one elaborates the data elements, architecture (various structures), and procedures required in the solution. From a waterfall perspective, the process is generally categorized into two phases: a high-level design and a detailed design. A spiral method takes this same process through a number of elaborations that finally arrive at the solution. In addition to these models, one can use prototyping or a 4GL method or a combination of any of the above (Pressman, 1993).

Within these models, one can employ two generally accepted approaches: structured and object-oriented. In a structured approach, the data elements, architecture (structure), and procedures of the application are developed as separate and distinct entities. In some models, relationships can be drawn between the three in the documentation, but the entities themselves are separate and distinct implementations.

In an object-oriented (OO) design approach, a model of the program is created by defining the system component, also known as classes in most OO paradigms, and the relationships between them. Each class is defined in terms of states (data) and behaviors (methods). The process of analysis and design, in most methodologies, follows an iterative approach where the design is successively refined until an acceptable design for the system is reached. One might also refer to this design model as the architecture.

Design Patterns
Design patterns enable developers to describe a possible solution using an existing framework as well as to describe why a particular design approach is chosen. Speaking about building construction, Christopher Alexander (1977)
Software Design Models and Patterns

said:

Each design pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.

Gamma et al. (1995) asserts that the same holds true for object-oriented design patterns. Regardless of the environment in which the pattern occurs, Gamma et al. (1995) states there are four essential elements for a pattern: the name, the problem, the solution, and the consequences. The pattern name is used to describe a design problem, its solutions, and its consequences. The problem describes when to apply the pattern. The solution describes the elements that make up the design, their relationships, responsibilities, and collaborations. Finally, the consequences are the results and trade-offs of applying the pattern.

Patterns are used widely in object-oriented design efforts and drive many of the solutions for today’s business problems. They provide the developer with the ability to fashion appropriate and highly successful object-oriented design models to solve virtually any business problem.

References


