Module 4

Software Design Issues

Lesson 8

Basic Concepts in Software Design

Version 2 CSE IIT, Kharagpur

Specific Instructional Objectives

At the end of this lesson the student will be able to:

- Identify the software design activities.
- Identify the items to be designed during the preliminary and detailed design activities.
- Identify the primary differences between analysis and design activities
- Identify the important items developed during the software design phase.
- State the important desirable characteristics of a good software design.
- Identify the necessary features of a design document in order to facilitate understandability.

Software design and its activities

Software design deals with transforming the customer requirements, as described in the SRS document, into a form (a set of documents) that is suitable for implementation in a programming language. A good software design is seldom arrived by using a single step procedure but rather through several iterations through a series of steps. Design activities can be broadly classified into two important parts:

- Preliminary (or high-level) design and
- Detailed design.

Preliminary and detailed design activities

The meaning and scope of two design activities (i.e. high-level and detailed design) tend to vary considerably from one methodology to another. High-level design means identification of different modules and the control relationships among them and the definition of the interfaces among these modules. The outcome of high-level design is called the program structure or software architecture. Many different types of notations have been used to represent a high-level design. A popular way is to use a tree-like diagram called the structure chart to represent the control hierarchy in a high-level design. However, other notations such as Jackson diagram [1975] or Warnier-Orr [1977, 1981] diagram can also be used. During detailed design, the data structure and the algorithms of the different modules are designed. The outcome of the detailed design stage is usually known as the module-specification document.

Difference between analysis and design

The aim of analysis is to understand the problem with a view to eliminate any deficiencies in the requirement specification such as incompleteness,

inconsistencies, etc. The model which we are trying to build may be or may not be ready.

The aim of design is to produce a model that will provide a seamless transition to the coding phase, i.e. once the requirements are analyzed and found to be satisfactory, a design model is created which can be easily implemented.

Items developed during the software design phase

For a design to be easily implemented in a conventional programming language, the following items must be designed during the design phase.

- Different modules required to implement the design solution.
- Control relationship among the identified modules. The relationship is also known as the call relationship or invocation relationship among modules.
- Interface among different modules. The interface among different modules identifies the exact data items exchanged among the modules.
- Data structures of the individual modules.
- Algorithms required to implement each individual module.

Characteristics of a good software design

The definition of “a good software design” can vary depending on the application being designed. For example, the memory size used by a program may be an important issue to characterize a good solution for embedded software development – since embedded applications are often required to be implemented using memory of limited size due to cost, space, or power consumption considerations. For embedded applications, one may sacrifice design comprehensibility to achieve code compactness. For embedded applications, factors like design comprehensibility may take a back seat while judging the goodness of design. Therefore, the criteria used to judge how good a given design solution is can vary widely depending upon the application. Not only is the goodness of design dependent on the targeted application, but also the notion of goodness of a design itself varies widely across software engineers and academicians. However, most researchers and software engineers agree on a few desirable characteristics that every good software design for general application must possess. The characteristics are listed below:

- **Correctness:** A good design should correctly implement all the functionalities identified in the SRS document.
- **Understandability:** A good design is easily understandable.

- **Efficiency:** It should be efficient.
- **Maintainability:** It should be easily amenable to change.

Possibly the most important goodness criterion is design correctness. A design has to be correct to be acceptable. Given that a design solution is correct, understandability of a design is possibly the most important issue to be considered while judging the goodness of a design. A design that is easy to understand is also easy to develop, maintain and change. Thus, unless a design is easily understandable, it would require tremendous effort to implement and maintain it.

Features of a design document

In order to facilitate understandability, the design should have the following features:

- It should use consistent and meaningful names for various design components.
- The design should be modular. The term modularity means that it should use a cleanly decomposed set of modules.
- It should neatly arrange the modules in a hierarchy, e.g. in a tree-like diagram.