SYSTEM DESIGN

Design is the highly creative phase in the software development where the designer plans "how" a software system should be produced in order to make it functional, reliable, and reasonably easy to understand, modify and maintain. Requirements analysis tells "what" of the system while designing phase tells "how" of the system. The goal of design phase is to take the SRS document as input and to produce the output as a final product. Judging the goodness of a design involves many subjective factors which depend on the particular application. In other words, a design solution which can be judged good for one application may not be considered good for another application. Understandability of a design is a major factor which is used to evaluate the goodness of a design since a design that is easily understandable is also easy to maintain and change. Modular design is one of the fundamental principles of a good design. Decomposition of a problem into modules facilitates taking advantage of the divide and conquers principle. If different modules are almost independent of each other, then each module can be understood separately, eventually reducing the complexity greatly. There are two main approaches to design a system:

(1) Data Centered Approach

(2) Process Centered Approach

In both the approaches the other factor cannot be ignored, *i.e.*, process cannot be ignored in data centered approach and vice-versa. At the end of requirement analysis, we have list of DFD, data dictionary, ERD and process specification. These items will form the basic inputs to the design phase.

In data centered approach, we define data structure first and then the process while in process centered approach, we defined processes first and data structure at the end.

A system can be derived in many ways. Let us have a look at airline reservations. It is possible to have all data related to reservations in a big computer at a central location and all canters can use this repository of data. Alternatively, the data can be stored at multiple locations and computers can be networked to pass data to each other. Thus, we have no alternatives to system architecture-which is the better alternative. Each alternative has its own advantages and disadvantages.

Scanned with CamSca

The general tasks involved in the design process are:

- Design the overall system processes
- Segmenting the system into smaller, compact workable modules
- Designing the database structure
- Specifying the details of the programme to be created to achieve the desired functionality
- Designing the input and output documents
- Designing controls for the system
- Documenting the system design
- System reviews

5.1 THE DESIGN PROCESS

The software design is an activity which is after the requirements analysis activity. This phase beginswhen the requirements document for the system to developed is available. Design is an important phasein the software development life cycle, it bridges the requirements specification and the final solution forsatisfying the requirements. The goal of the design process is to produce a model of the system, whichcan be used later to build that system. The model thus produced is called the design of the system.

The design process for the software has two levels:

(i) **System Design or Top-Level Design :** Using this, the modules that are needed for the system are decided; the specifications of these modulesand how these modules need to be connected are also decided.

(ii) **Detailed Design or Logic Design :** Using this, the internal design of the modules are decided or how the specifications of the modules canbe satisfied are decided. This type of design essentially expands the system design to contain moredetailed description of the processing logic and data structures so that the design is sufficiently complete for coding. A design can be object-oriented or function-oriented. In function-oriented design, the design consists of module definitions, with each module supporting a functional abstraction. In object-oriented design, themodules in the design represent data abstraction.

5.2 DESIGN CONCEPTS

The design concepts provide basic criteria for design quality. There are a number of fundamental design concepts that has been of interest: Abstraction, refinement, modularity, software architecture, control hierarchy, structured partitioning, data structure, software procedure, information hiding.

Scanned with CamSca

5.2.1 Abstraction

Abstraction is a means of describing a program function, at an appropriate level of detail. It deals with problems at some level of generalization without regard to irrelevant low level details. At the highest level of abstraction, a solution is stated in broad terms using the language of problem environment. At the lower levels of abstraction, more procedural details are given.

There are three important levels of abstraction.

(i) Procedural abstraction : A procedural or functional abstraction is a named sequence of instructions that has a specific and limited function.

(ii) Data abstraction : A data abstraction is a named collection of data that describes abstract data types, objects, operations on objects by suppressing the representation and manipulation details.

Consider an example sentence, "open the door". Here the word "open" is an of example of procedural abstraction, which implies a long sequence of procedural steps like: walk to the door, hold the door-knob, turn the knob and pull the door, move away from the door. The word "door" is an example of data abstraction, which has certain attributes like dimensions, weight, door type etc, which describe the door.

(iii) Control abstraction : It implies a program control mechanism without specifying internal details. That is, stating the desired effect without stating exact mechanism of control, co-routines, exception handling.

Example:

ON interrupt DO

save STACK_A and call Exception_handler_a;

5.2.3 Stepwise Refinement

Stepwise refinement is a top-down approach where a program is refined as a hierarchy of increasing levels of detail. It causes the designer to elaborate on the original statement, providing more and more details at end of each refinement step. The process of refinement may start during the requirements analysis and conclude when the detail of the design is sufficient for conversion into code. As tasks are refined, the data associated with the tasks may have to be refined, decomposed, or structured. And processing procedures and data structures are likely to be refined in parallel.

5.2.3 Software Architecture

While refinement is about the level of detail, architecture is about structure of software. It refers to the overall structure of the software and the ways in which that structure provides the conceptual integrity for a system. The architecture of the procedural and data elements of a design represents a software solution for the real-world problem defined by the requirements analysis. A set of architectural patterns enable a software engineer to reuse the design level concepts.

Scanned with CamSca