***8.1 Introduction***

No program or system design is perfect. Communication between the user and the designer is not always complete or clear and time is usually short. The result is errors.

The number and nature of errors in a new design depend on several factors.

1. Communication between the user and the designer.
2. The programmer’s ability to generate a code that reflects exactly the system specifications.

**Design Objectives**

The two operational design objectives continually sought by developers are systems reliability and maintainability.

**Reliable Systems**

A system is said to have *reliability* if it does not produce dangerous or costly failures when it is used in a reasonable manner, that is, in a manner that a typical user expects is normal.

There are three approaches to reliability namely, error avoidance, error detection

and error tolerance.

**Error Avoidance**

**Error Detection and Correction**

**Error Tolerance**

**Causes of Errors**

**Maintenance of Systems**

From 60 to 90 percent of the overall cost of software during the life of a

system is spent on maintenance

**Maintainable Designs**

More accurately defining the user’s requirements during systems

development.

2. Assembling better system documentation.

3. Using more effective methods for designing processing logic and

communicating it to project team members.

4. Making better use of existing tools and techniques.

5. Managing the systems engineering process effectively.

**Software Design**

These principles should guide software design:

􀂃 *Modularity and Partitioning.*

Each system should consist of a hierarchy of modules. Lower level modules are generally smaller in scope and size compared to higher – level modules and serve to partition processes into separate functions.

􀂃 *Coupling*

Modules should have little dependence on other modules in a system.

􀂃 *Cohesion*

Modules should carry out a single processing function.

􀂃 *Span of Control*

Modules should interact with and manage the functions of a limited number of

lower-level modules.

*Size*

The number of instructions contained in a module should be limited to that module size is generally small.

􀂃 *Shared Use*

Functions should not be duplicated in separate modules, but established in a single module that can be invoked by any other module when needed.

**Coupling**

Coupling refers to the strength of the relationship between modules in a system. In general, good designers seek to develop the structure of a system so that one module has little dependence on any other module.

Loose coupling minimizes the interdependence between modules. We can achieve

this in the following ways.

􀂙 Control the number of parameters passed between modules

􀂙 Avoid passing unnecessary data to called modules

􀂙 Pass data (whether upward or downward) only when needed

􀂙 Maintain superior/subordinate relationship between calling and called

modules.

􀂙 Pass data, not control information

**Cohesion**

In properly modularized, cohesive systems, the contents of the modules are so designed that they perform a specific function and are more easily understood by people than systems designed by other methods.

There are four general types of modules contents:

1. Module contents determined by function performed,

2. Module contents determined by data used,

3. Module contents determined by logic of processing and

4. Module contents not closely related.

**Span of Control**

Span of control refers to the number of subordinate modules controlled by a calling module. In general, we should seek to have no more than five to seven subordinate modules.

**Module Size**

How large should a program module be? While it is impossible to fix a specific number of instructions, there are useful guidelines to manage module size.

**Cost and Benefit Categories**

Hardware

Personnel

Facility costs

Operating costs

Supply costs

**Procedure for Cost/ Benefit Determination**

Cost/ benefit analysis is a procedure that gives a picture of the various costs, benefits and rules associated with a system. The determination of costs and benefits entails the following steps:

1. Identify the costs and benefits pertaining to given project.

2. Categorize the various costs and benefits for analysis.

3. Select a method of evaluation.

4. Interpret the results of the analysis.

5. Take action.

This project **cost management** process will help you to:

* Identify each of the costs within your project
* Ensure that expenses are approved before purchasing
* Keep a central record of all costs incurred
* Control the overall cost of your project

**Classifications of Costs and Benefits**

**Tangible or Intangible Costs and Benefits**

**Intangible:** For example, the cost of the breakdown of an online system during banking hours will cause the bank to lose deposits and waste human resources.

**Fixed or Variable- Costs and Benefits**

**Fixed:** Examples insurance. In contrast, variable costs are incurred on a regular (weekly, monthly) basis. They are usually proportional to work volume and continue as long as the system is in operation. An example is a decrease in the number of personnel by 20 percent resulting from the use of a new computer.

**Variable Benefits**: Variable benefits, on the other hand, are realized on a regular basis. For example, consider a safe deposit tracking system that saves 20 minutes preparing customer notices compared with the manual system. The amount of time saved varies with the number of notices produced.

In summary, cost/ benefit analysis is a tool for evaluating projects rather than a

replacement of the decision-maker. In real-life business situations, whenever a choice

among alternatives is considered, cost / benefit analysis is an important tool. Like any

tool, however, it has problems:

**Valuation problems:-**

**Distortion problems:-**

**Completeness problems:-**