UNIT III

1. The requirements model, manifested by *scenario-based, class-based, flow-oriented, and behavioral elements*,
2. The *architectural design* defines the relationship between major structural elements of the software,
3. The *interface design* describes how the software communicates with systems that interoperate with it, and with humans who use it.
4. The *component-level* design transforms structural elements of the software architecture into a procedural description of software components.
5. The design should provide a complete picture of the software, addressing the data, functional, and behavioral domains from an implementation perspective.
6. FURPS—functionality, usability, reliability, performance, and supportability
7. *A procedural abstraction* refers to a sequence of instructions that have a specific and limited function.
8. *A data abstraction* is a named collection of data that describes a data object
9. Cohesion is an indication of the relative functional strength of a module.
10. Coupling is an indication of the relative interdependence among modules.
11. Refactoring is the process of changing a software system in such a way that it does not alter the external behavior of the code [design] yet improves its internal structure.
12. Software architecture must model the structure of a system
13. and the manner in which data and procedural components collaborate with one another.
14. An architectural style is a transformation that is imposed on the design of an entire system.
15. An archetype is an abstraction (similar to a class) that represents one element of system behavior.
16. ACD stands for architectural context diagram
17. *Sharing dependencies* represent dependence relationships among consumers who use the same resource
18. *Flow dependencies* represent dependence relationships between producers and consumers of resources
19. *Constrained dependencies* represent constraints on the relative flow of control among a set of activities.
20. Component is “a modular, deployable, and replaceable part of a system that encapsulates implementation and exposes a set of interfaces.”
21. ISP stands for Interface Segregation Principle.
22. Interface is the equivalent of an abstract class that provides a controlled connection between design classes.

23. Example for Tabular Design Notation is Decision tables