Structured Systems Analysis and Design Method (SSADM)

Structured systems analysis and design methodology (SSADM) is a set of standards for systems analysis and application design.

It uses a formal methodical approach to the analysis and design of information systems.

It was developed by Learmonth Burchett Management Systems (LBMS) and the Central Computer Telecommunications Agency (CCTA) in 1980-1981 as a standard for developing British database projects.  
  
The SSADM is an open methodology based on the waterfall model. It has been used by many commercial businesses, consultants, educational establishments and CASE tool developers.

SSADM follows the waterfall life cycle model starting from the feasibility study to the physical design stage of development.

One of the main features of SSADM is the intensive user involvement in the requirements analysis stage.

The users are made to sign off each stage as they are completed assuring that requirements are met.

The users are provided with clear, easily understandable documentation consisting of various diagrammatic representations of the system.

SSADM breaks up a development project into stages, modules, steps and tasks.

The first and foremost model developed in SSADM is the data model.

It is a part of requirements gathering and consists of well-defined stages, steps and products.

The techniques used in SSADM are logical data modeling, data flow modeling and entity behavior modeling.

* Logical Data Modeling: This involves the process of identifying, modeling and documenting data as a part of system requirements gathering. The data are classified further into entities and relationships.
* Data Flow Modeling: This involves tracking the data flow in an information system. It clearly analyzes the processes, data stores, external entities and data movement/flow.
* Entity Behavior Modeling: This involves identifying and documenting the events influencing each entity and the sequence in which these events happen.

Some of the important characteristics of SSADM are:

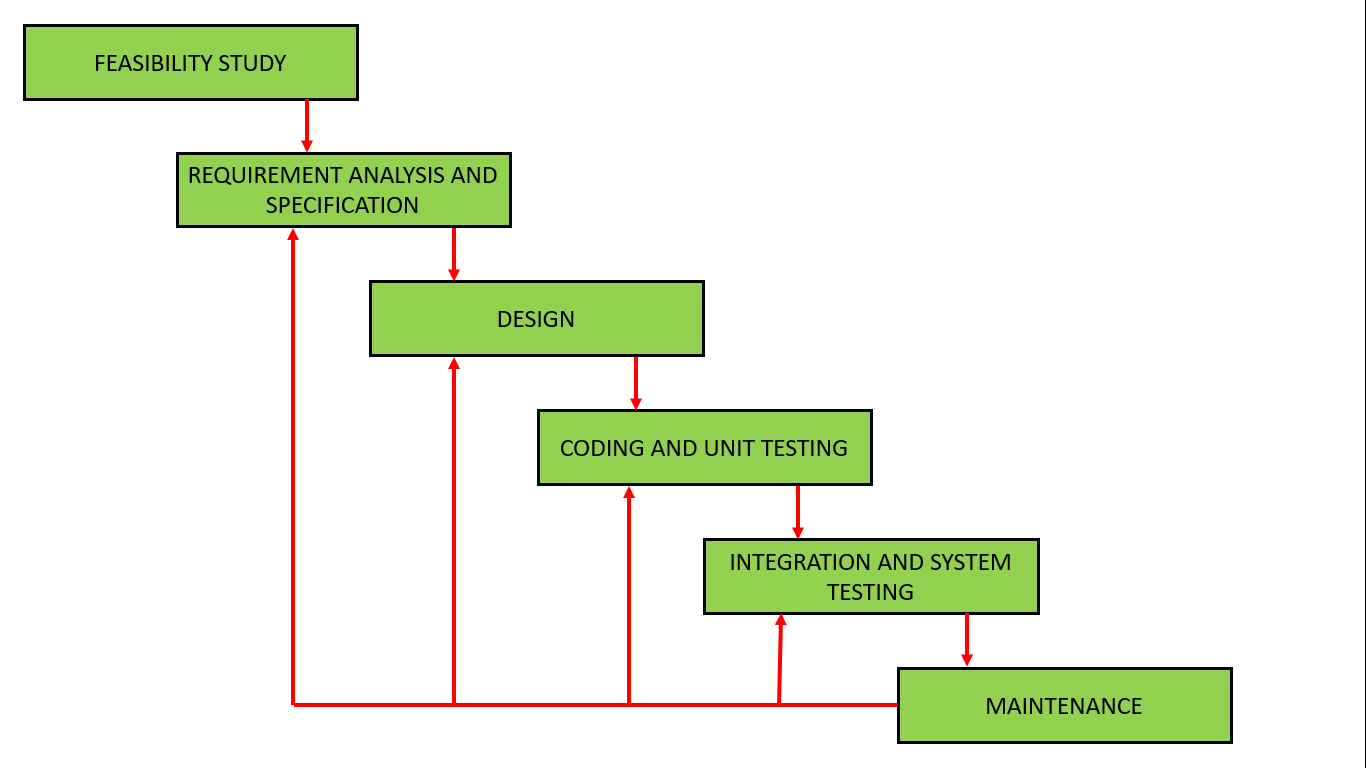
* Dividing a project into small modules with well-defined objectives
* Useful during requirements specification and system design stage
* Diagrammatic representation and other useful modeling techniques
* Simple and easily understood by clients and developers
* Performing activities in a sequence

The stages of SSADM include:

* Determining feasibility
* Investigating the current environment
* Determining business systems options
* Defining requirements
* Determining technical system options
* Creating the logical design
* Creating the physical design

Each of these stages applies certain techniques and a sequence of analysis. They include conventions and procedures for recording and interpreting the information with the help of diagrams and text.

Software Development Life Cycle (SDLC): Iterative Waterfall Model



**Advantages of Iterative Waterfall Model:**

* **Feedback Path –**  
  In the classical waterfall model, there are no feedback paths, so there is no mechanism for error correction. But in the iterative waterfall model feedback path from one phase to its preceding phase allows correcting the errors that are committed and these changes are reflected in the later phases.
* **Simple –**  
  Iterative waterfall model is very simple to understand and use. That’s why it is one of the most widely used software development models.
* **Cost-Effective –**  
  It is highly cost-effective to change the plan or requirements in the model. Moreover, it is best suited for agile organizations.
* **Well-organized –**  
  In this model, less time is consumed on documenting and the team can spend more time on development and designing.

**Drawbacks of Iterative Waterfall Model:**

* **Difficult to incorporate change requests –**  
  The major drawback of the iterative waterfall model is that all the requirements must be clearly stated before starting the development phase. Customers may change requirements after some time but the iterative waterfall model does not leave any scope to incorporate change requests that are made after the development phase starts.
* **Incremental delivery not supported –**  
  In the iterative waterfall model, the full software is completely developed and tested before delivery to the customer. There is no scope for any intermediate delivery. So, customers have to wait a long for getting the software.
* **Overlapping of phases not supported –**  
  Iterative waterfall model assumes that one phase can start after completion of the previous phase, but in real projects, phases may overlap to reduce the effort and time needed to complete the project.
* **Risk handling not supported –**  
  Projects may suffer from various types of risks. But, the Iterative waterfall model has no mechanism for risk handling.
* **Limited customer interactions –**  
  Customer interaction occurs at the start of the project at the time of requirement gathering and at project completion at the time of software delivery. These fewer interactions with the customers may lead to many problems as the finally developed software may differ from the customers’ actual requirements.

**Software Design Approaches:**

**Two radically different approaches to software design: function-oriented design and object oriented design. [Rajib Mall]**

**Function oriented design:**

**Design approaches based on function oriented approaches:**

**→Structured design approach by Constantine and Yourdon [1979]**

**→Structured design approach by Jackson [1975]**

**→The Warnier – Orr methodology [1977, 1981]**

**→The step-wise refinement approach by Wirth [1971]**

**→The Hatley-Pirbhai methodology [1987]**

**→ Learmonth Burchett Management Systems (LBMS) and the Central Computer Telecommunications Agency (CCTA) in 1980-1981**

**Object oriented design:**

**Object will have state information; attribute & behaviour;**

**Attribute is nothing but data members; behaviour is represented through member functions/ methods.**

**[**

**Software Requirement Specifications (SRS)**

**[Functional Requirement; Non-Functional Requirement]**

**Requirement Engineering**

**Requirement Management / Project Management**

**User Requirements / System Requirements**

**Software Design Approaches: Function Oriented Design / Object Oriented Design**

**Function Oriented Design → Structured System Analysis & Design (SSAD)**

**Data Flow Diagram & Entity Relationship (E-R) Diagram**

**Data Dictionary**

**Logical Design / Physical Design**

**Software Development Life Cycle [SDLC]**

**]**