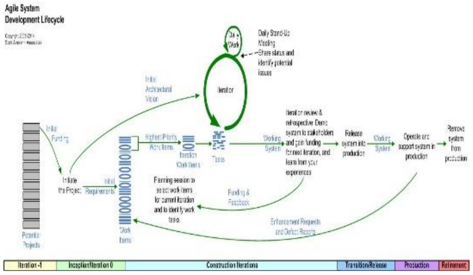
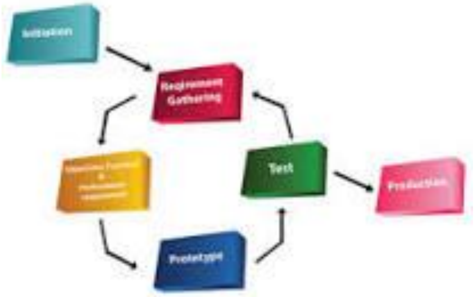
Agile method

Agile software development is an approach to **software development** under which requirements and solutions evolve through the collaborative effort of self-organizing and cross-functional teams and their customer(s)/end user(s).


Agile System
Development Lifecycle
Daily Work
Daily Stand-Up Meeting
Share status and identify potential issues
Initial Architectural Vision
Iteration
Initial Funding
Initiate the Project
Initial Requirements
Highest-Priority Work Items
Iteration Work Items
Tasks
Working System
Iteration review & retrospective: Demo system to stakeholders and gain funding for next iteration, and learn from your experiences
Release system into production
Working System
Operate and support system in production
Remove system from production
Planning session to select work items for current iteration and to identify work tasks
Funding & Feedback
Enhancement Requests and Defect Reports
Potential Projects
Work Items
Iteration -1
Inception/Iteration 0
Construction Iterations
Transition/Release
Production
Retirement

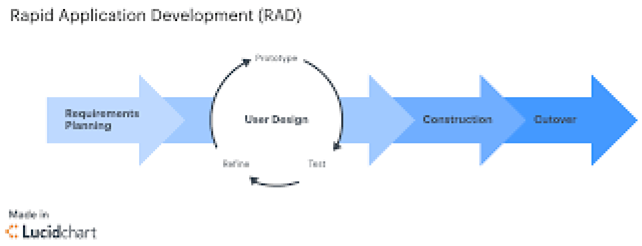
Prototyping

The **Prototyping Model** is a systems development method (SDM) in which a **prototype** (an early approximation of a final system or product) is built, tested, and then reworked as necessary until an acceptable **prototype** is finally achieved from which the complete system or product can now be developed.



Rapid application Development

Rapid application development (RAD) is an agile project management strategy popular in software development. The key benefit of a RAD approach is fast project turnaround, making it an attractive choice for developers working in a fast-paced environment like software development. This rapid pace is made possible by RAD’s focus on minimizing the planning stage and maximizing prototype development.



System Development Methodologies

1 Structured Methodology

2 Object Oriented Methodology

1 Structured Methodology

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. ... The SSADM is an open methodology based on the waterfall model.

2 Object Oriented Methodology

Object Oriented Methodology (OOM) is a system development approach encouraging and facilitating re-use of software components.

Structured System Analysis And Design Methodology (SSADM)

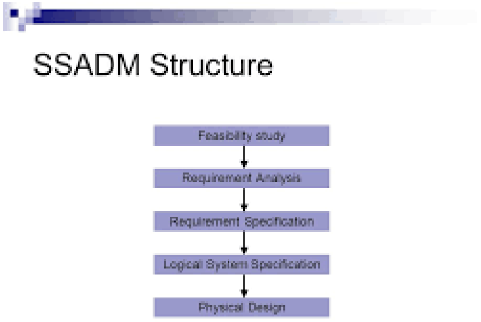
Structured Systems Analysis and Design Method, originally released as methodology, is a systems approach to the analysis and design of information systems.

□ Involves study the present system and sketches a blueprint to develop a new system or to modify the existing system

□ Introduced in early 1980s

□ One of the structured methods for System Analysis & Design

- Adopted as a standard by Central Computer and Telecommunications Agency (CCTA), UK
- Covers most of the System Development Life Cycle (SDLC) from Feasibility Study to System Design
- More focus on analysis and design Stages of the SDLC covered by SSADM



SSADM Structure

Feasibility Study

The business area is analyzed to determine whether the system development is feasible

Requirements Analysis

The requirements of the system to be developed are identified and the current business environment is modeled in terms of the processes carried out and the data stored

Requirements Specification

Detailed functional and non-functional requirements are defined and new processes are introduced to define the required processing and data storage

Logical System Specification

System to be developed as specified logically without taking technical constraints into consideration

Physical Design

Logical design is transformed into a physical design taking technical constraints into consideration

Preliminary Investigation

This is the first stage of system development life cycle. This gives a clear picture of what actually the physical system is? Preliminary investigation is done in two phases namely, problem definition and feasibility study. In the problem definition, a preliminary survey of the system is carried out to identify the scope of the system. In the feasibility study, the proposed system is evaluated for its feasibility. Feasibility of a system means whether the development of a new or improved system is practical and beneficial.

Feasibility is evaluated from developers' and users' point of view. Developers see whether they have the required technology and manpower to develop the new system. Is the new system really going to be beneficial to the users? Does the user have the financial ability to get developed that type of a system? The feasibility of the system is evaluated on the three main aspects: technical, economical, operational and organizational.

Technical feasibility:

this evaluates whether the developers have ability to construct the proposed system. The technical assessment helps answer the question such as whether the technology needed for the system exists, how difficult system will be to develop, and whether the developers have enough experience using that technology

Economic feasibility:

this studies cost and benefits to evaluate whether the benefits justify the investments in the system development. Can the development cost be justified? An important outcome of the economic feasibility study is the cost benefit analysis.

Operational feasibility:

this assesses the willingness and ability of the users to support and use the proposed system. Will the system be used when it is developed and installed? Will there be resistance from users to the system development?

Organizational Feasibility

It is to define the legal and corporate structure of a business. An organizational feasibility study may also include professional background information about the founders and principals of the business and what skills they can contribute to the business.

Requirement Analysis

requirement is a functional or non-functional need to be implemented in the system. Functional means providing particular service to the user.

For example, in context to banking application the functional requirement will be when customer selects "View Balance" they must be able to look at their latest account balance.

requirement can also be a non-functional, it can be a performance requirement. For example, a non-functional requirement is where every page of the system should be visible to the users within 5 seconds.

Requirement analysis is the process of studying and analyzing the user needs to arrive at a definition of the problem domain and system requirements. The main objective of requirement analysis is to discover the boundaries of the new system and how system must interact within the new problem domain. Requirement analysis helps to detect and resolve conflicts between (user) requirements.

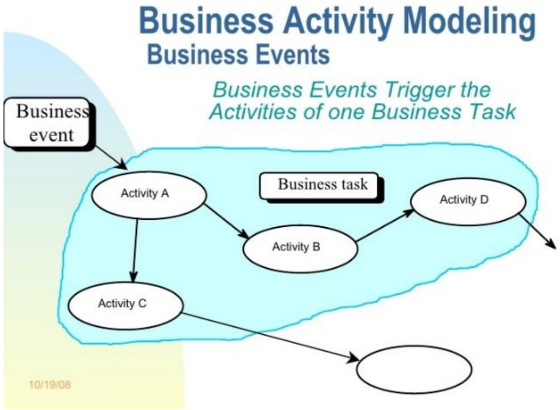
- Functional requirements: What activities that the system should carry out.
- Describe the requested functionality/behavior of the system
- Non Functional requirements: How well or with in what limits requirement should be satisfied
- IEEE standard for requirement definition
- Essential requirements are defined with “Shall” whereas nice to have requirements are defined with “Should”

Analytical Tools Business Activity Modeling

A start-off technique for understanding what’s going on in the system under investigation. It is used to show the business activities that the actors in the system’s environment do and their associations.

Business Activity Model

Business Activity Modelling is a technique to show high-level **activities** and logical dependencies as a conceptual **model**, based on a stakeholders perspective.



Business Activity Modeling

Business Events

Business Events Trigger the Activities of one Business Task

Data Flow Modeling (DFM)

Data Flow Modeling (DFM) DFM is used to model data processing in the system. It is used to define partitions into sub systems. DFM consists of a set of Data Flow Diagrams (DFD) and associated textual descriptions. DFD illustrates the way in which the data is passed around the system, how data is processed within the system, where data is stored in the system.

<u>Components of a DFD</u>

□ External Entities

□ represents people, organizations or other systems external to the system under investigation

□ acts as a source or a recipient of data

□ name should refer to a generic type, not to an instance of that type

□ Data Flows

□ show flows of data to, from and within the system

□ link other components in a DFD

□ could be one-way or two-way

□ represented with solid arrows, however between two external components are shown by dashed arrows and intersections should be avoided

□ Processes

□ represents business activities carried out in the system

- each process has three properties: Id, Name and Location
- processes that don’t need any further decomposition is called elementary processes
- Data Stores
- used to hold data within the system
- four types: Manual (M), Computerized (D), Temporary (T) and Manual Temporary T(M)
- each data store has three properties: Id, Type and Name Context Diagram
- a DFD with the highest level of abstraction
- represents the entire system as a single process
- shows how system interacts with its external entities

Document Flow Diagram

- acts as a bridge between the Context Diagram and Level 1 DFD
- illustrates how documents (papers, conversations, data passed among computers) passed in the system

Level 1 DFD

- provides a higher level overview of the system’s data processing
- shows data movements among the major components of the system
- must be consistent with the Context Diagram

Lower level DFDs

- provides a way to go into more details successively
- enables top down approach
- must be consistent with the higher level DFD Elementary