Engineering in One Video (EIOV)

Watch video on YouTube

B.Tech

Software Engineering

KCS-601

With Notes

UNIT-2

Software Requirement Specifications

(SRS)

(in one video)

AKTU Exam

Engineering in One Video (EIOV)

Watch video on YouTube

Topics to be covered...

Requirement engineering process
Requirement elicitation
System documentation
Feasibility Study
Data Flow Diagram(DFD)
DFD vs Flow chart
ER diagram
Decision table vs Decision tree
SRS Document
Software Quality Assurance(SQA)
Verification and Validation
ISO 9000 models
ISO vs SEI-CMM model
Happy Ending!

Engineering in One Video (EIOV)

Watch video on YouTube

Requirement engineering process

Requirement engineering process

Requirement engineering is the process of collecting, validating and managing the requirements essential for the development of the software, specified by the clients or the end-users.
It is a process that is performed in the initial stages of any software development.

Steps in requirement engineering process:

1. Requirements Elicitation
2. Requirements Specifications
3. Requirements Verification and Validation
4. Requirements Management

Requirement engineering process

1. Requirements Elicitation:

It is related to the various ways used to gain knowledge about the project domain and requirements. The various sources of domain knowledge include customers, business manuals, the existing software of same type, standards and other stakeholders of the project.
The techniques used for requirements elicitation include interviews, brainstorming, task analysis, Delphi technique, prototyping, etc.

2. Requirements Specifications:

All the requirements including the functional as well as the non-functional requirements and the constraints are specified by these models in totality.
During specification, more knowledge about the problem may be required which can again trigger the elicitation process.
The models used at this stage include ER diagrams, data flow diagrams(DFDs) etc.

Requirement engineering process

3. Requirements Verification and Validation:

Verification: It refers to the set of tasks that ensures that the software correctly implements a specific function.

Validation: It refers to a different set of tasks that ensures that the software that has been built is traceable to customer requirements.

The main steps for this process include:

- The requirements should be complete in every sense.
- The requirements should be practically achievable.

4. Requirements Management:

Requirement management is the process of analyzing, documenting, tracking, prioritizing and agreeing on the requirement and controlling the communication to relevant stakeholders. This stage takes care of the changing nature of requirements. Being able to modify the software as per requirements in a systematic and controlled manner is an extremely important part of the requirements engineering process.

Engineering in One Video (EIOV)

Watch video on YouTube

Requirement Elicitation

Watch video on YouTube

Requirement elicitation

Requirement elicitation is the activity during which software requirements are discovered, expressed, and revealed from system requirements.

Requirements elicitation Activities:

- Knowledge of the overall area where the systems is applied.
- Interaction of system with external requirements.
- Detailed investigation of user needs.

Requirements elicitation Methods:

- Interviews
- Brainstorming Sessions
- Quality Function Deployment (QFD)
- Use Case Approach

Requirement elicitation

1. Interview:

Objective of conducting an interview is to understand the customer's expectations from the software.
Interviews maybe be open-ended or structured.

- In open-ended interviews there is no pre-set agenda. Context free questions may be asked to understand the problem.
- In structured interview, agenda of fairly open questions is prepared. Sometimes a proper questionnaire is designed for the interview.

2. Brainstorming Sessions:

- It is a group technique.
- It is intended to generate lots of new ideas hence providing a platform to share views.
- A highly trained facilitator is required to handle group bias and group conflicts.
- Every idea is documented so that everyone can see it.
- Finally, a document is prepared which consists of the list of requirements and their priority if possible.

Requirement elicitation

Problems in Requirement Elicitation:

1. Less Knowledge of technology.
2. Use of different language by developer and user.
3. Lack of skills in developer.
4. User is not providing some information due to some reason.

Engineering in One Video (EIOV)

Watch video on YouTube

Feasibility Study

EIOV

Engineering

IN ONE VIDEO

CONNECT, LEARN AND GROW

Watch video on YouTube

Feasibility Study

A feasibility study specifies whether the proposed software project is practically possible or not. Whenever there arises a need for any software you don't directly jump in developing the particular software. Instead, we must first analyze certain facts to realize whether the software is worthwhile or not and this is called the feasibility study.

Types of Feasibility Study:

1. Technical Feasibility
2. Operational Feasibility
3. Economic Feasibility
4. Legal Feasibility
5. Schedule Feasibility

Types of Feasibility Study:

1. Technical Feasibility:

Technical feasibility inspects whether software can be built at all with available tools and experts.
It helps organizations determine whether the technical team is capable of converting idea into working systems.

2. Operational Feasibility:

Operational feasibility explores how a new project will impact daily processes in your company, what procedures should be implemented, and what efforts should be taken to maintain it. Say you’re going to launch a global e-commerce platform. Then, you need to have local warehouses, local service teams, and, in some cases, even local websites in each country. It’s very difficult to realize operationally and might make no sense at all — though the initial idea looks commercially attractive and technically feasible.

Types of Feasibility Study:

3. Economic Feasibility:

Economic feasibility examines the costs and financial benefits of the project. To determine economic feasibility, a rough order of magnitude (ROM) estimate is commonly performed.
It includes all required cost for final development like Hardware, Software, Design and Development cost.

4. Legal Feasibility:

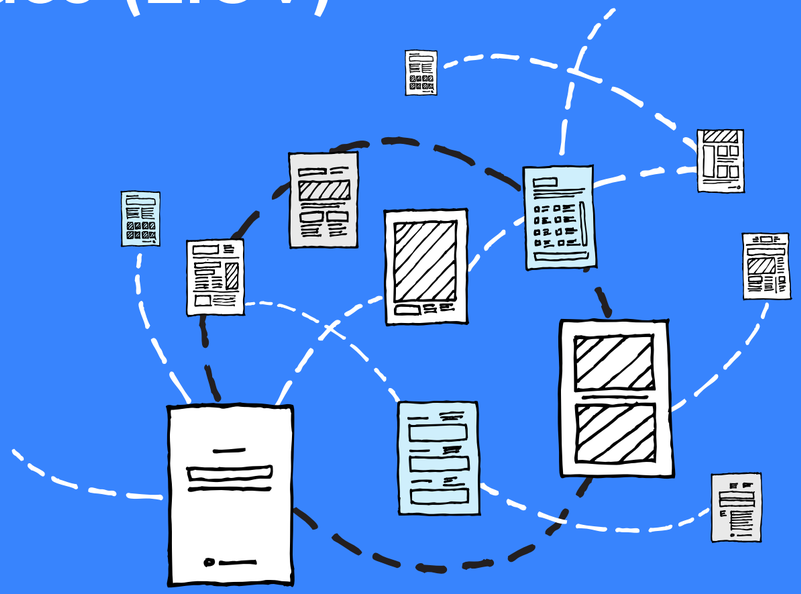
Legal feasibility makes sure that your product complies with all regulations and doesn't break any law. For example, medical software dealing with protected health information (PHI) must meet HIPAA rules. Besides that, you have to explore what legal risks there are and how they can impact your project.

5. Schedule Feasibility:

Mainly timeline/deadlines is analyzed for proposed project.
How much time the team will take to complete the final project.

Engineering in One Video (EIOV)

Watch video on YouTube



Data Flow Diagram(DFD)

Engineering in One Video (EIOV)

Watch video on YouTube

Data Flow Diagram(DFD)

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.
It shows how data enters and leaves the system, what changes the information, and where data is stored.

Data Flow Diagram(DFD)

The following observations about DFDs are essential:

1. All names should be unique. This makes it easier to refer to elements in the DFD.
2. Remember that DFD is not a flow chart. Arrows is a flow chart that represents the order of events; arrows in DFD represents flowing data. A DFD does not involve any order of events.
3. The DFD should maintain consistency across all the DFD levels.
4. A single DFD can have maximum processes upto 9 and minimum 3 processes.

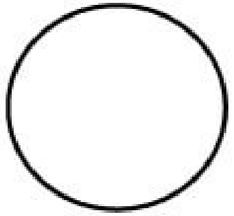
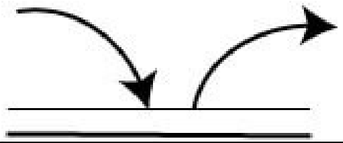
Levels of DFD

- 0-level DFD
- 1-level DFD:
- 2-level DFD:

Engineering in One Video (EIOV)

Watch video on YouTube

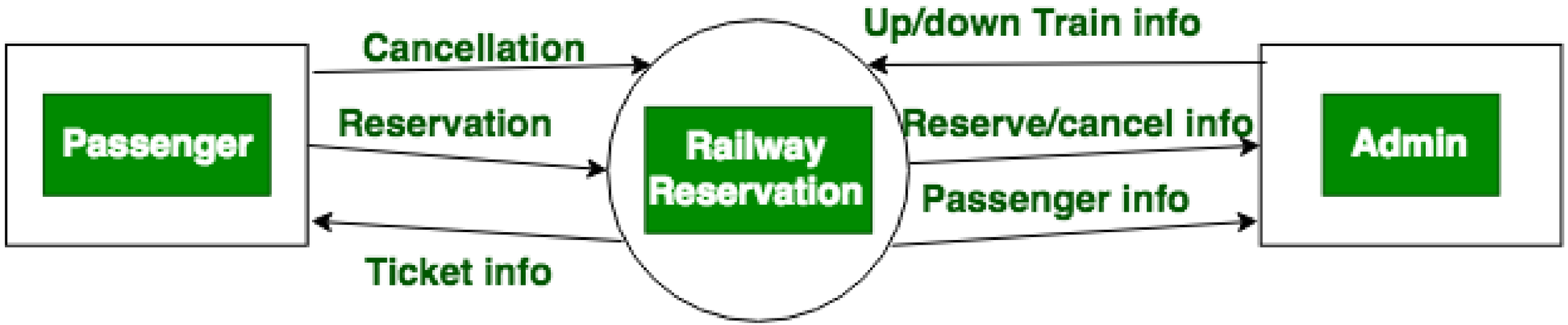
Data Flow Diagram(DFD)

Symbol	Name	Function
	Data flow	Used to Connect Processes to each , other , to sources or Sinks; te arrow head indicates direction of data flow.
	Process	Perfroms Some transformation of Input data to yield output data.
	Source of Sink (External Entity)	A Source of System inputs or Sink of System outputs.
	Data Store	A repository of data; the arrow heads indicate net inputs and net outputs to store.

Symbols for Data Flow Diagrams

Data Flow Diagram(DFD)

0 Level DFD:



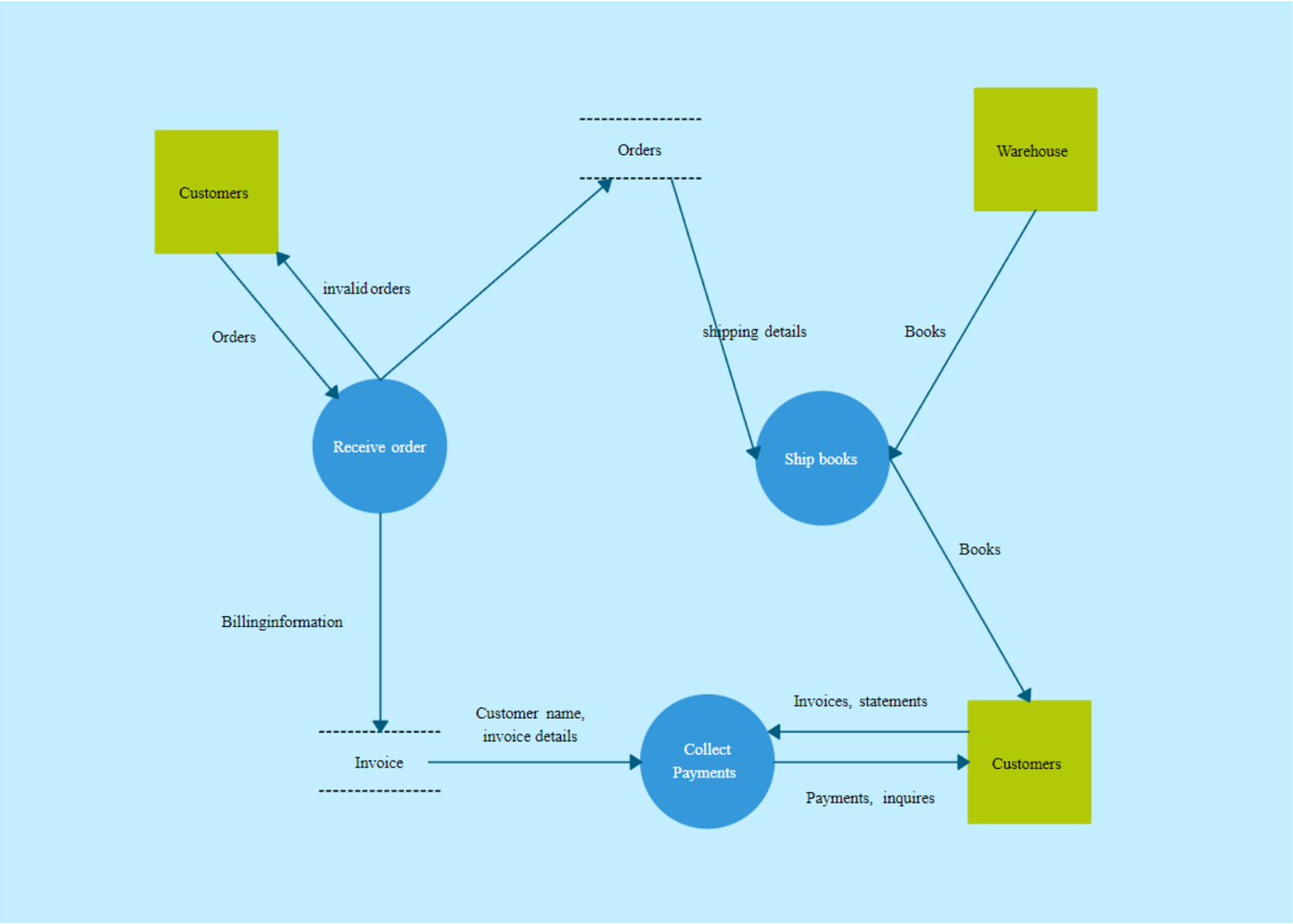
0-LEVEL DFD

Engineering in One Video (EIOV)

Watch video on YouTube

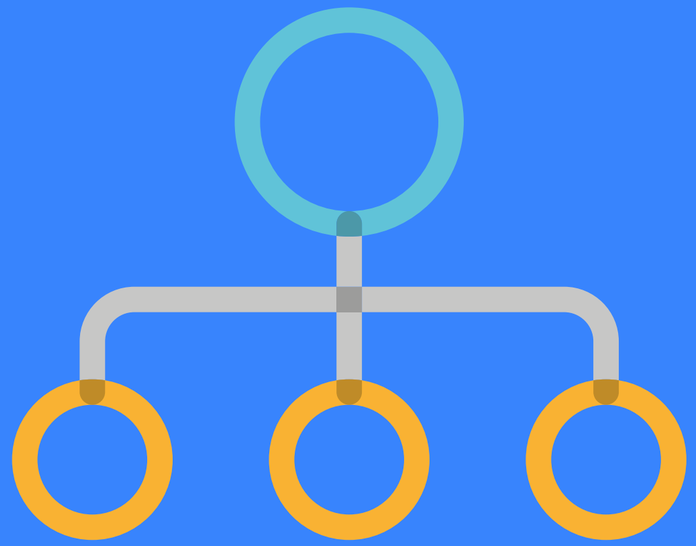
Data Flow Diagram(DFD)

1 Level DFD:



Engineering in One Video (EIOV)

Watch video on YouTube



DFD

VS

Flow chart

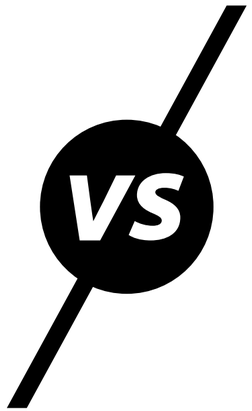
Engineering in One Video (EIOV)

Watch video on YouTube

DFD vs Flow chart

DFD

1. DFD represents the flow of control in program.
2. It is the view of the system at a lower level.
3. It shows the how to make system function.
4. It is not very suitable for a complex system.
5. Types of DFD:
    a. Physical DFD
    b. Logical DFD



Flow chart

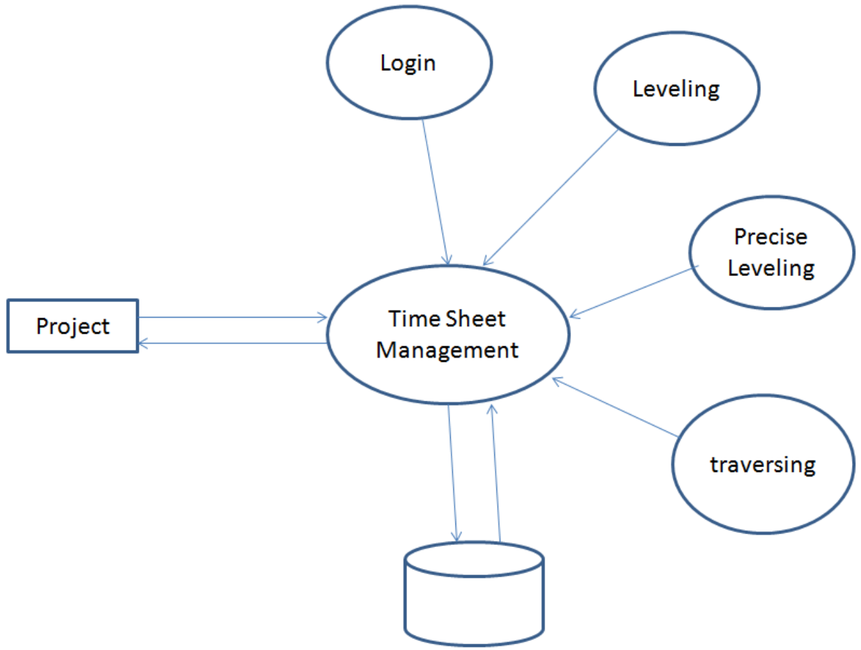
1. Flow chart represent the processes and data flow between them.
2. It is the view of the system at a high level.
3. It defines the functionality of the system.
4. It is used for complex systems.
5. Types of Flow chart:
    a. System flow chart
    b. Data flow chart
    c. Document flow chart
    d. Program flow chart

Engineering in One Video (EIOV)

Watch video on YouTube

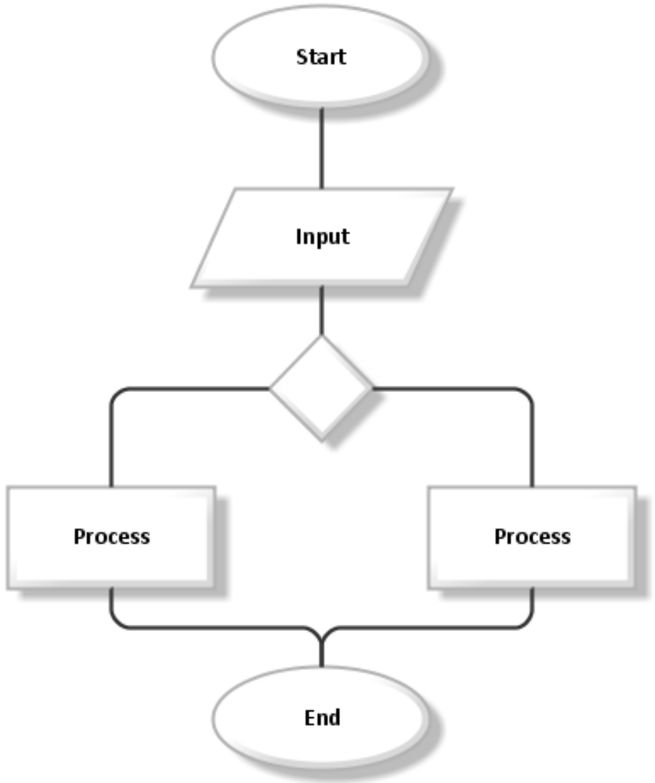
DFD vs Flow chart

DFD



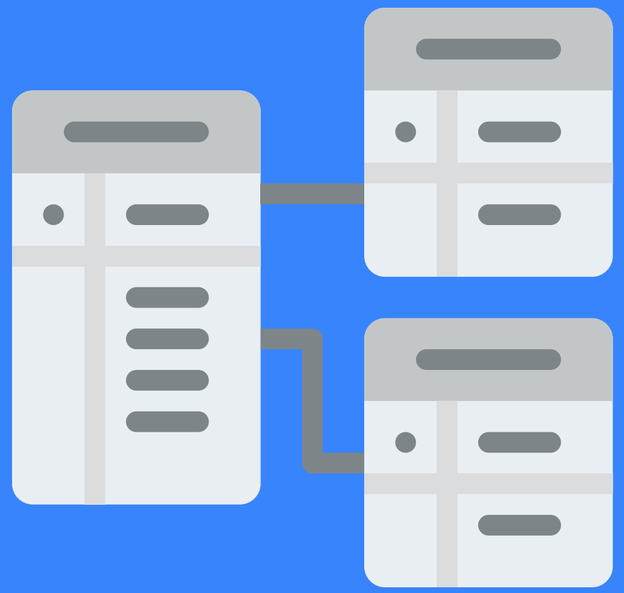
VS

Flow chart



Engineering in One Video (EIOV)

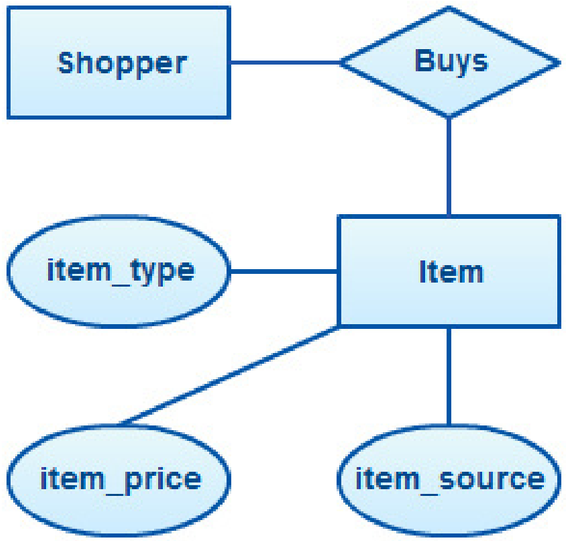
Watch video on YouTube



ER diagram

ER diagram

Entity relationship diagrams are used in software engineering during the planning stages of the software project. They help to identify different system elements and their relationships with each other. It is often used as the basis for data flow diagrams or DFD's as they are commonly known.
For example, an inventory software used in a retail shop will have a database that monitors elements such as purchases, item, item type, item source and item price. Rendering this information through an ER diagram would be something like this:

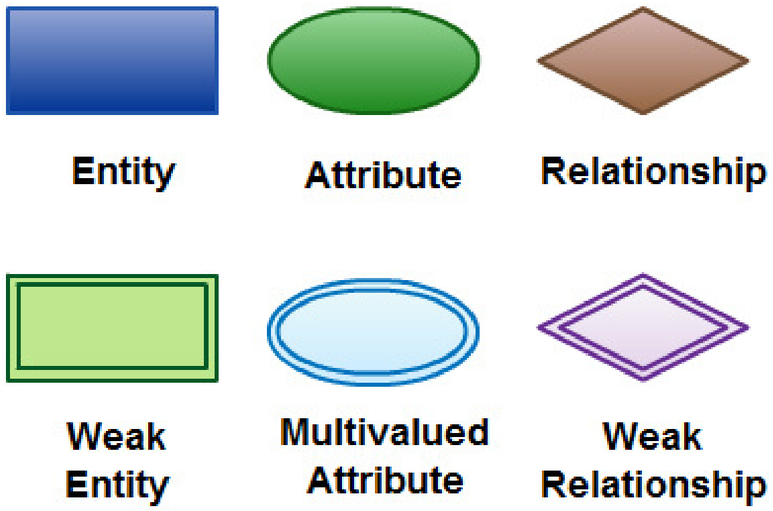


ER diagram

How to Draw ER Diagrams

Below points show how to go about creating an ER diagram.

1. **Identify all the entities** in the system. An entity should appear only once in a particular diagram. Create rectangles for all entities and name them properly.
2. **Identify relationships** between entities. Connect them using a line and add a diamond in the middle describing the relationship.
3. **Add attributes** for entities. Give meaningful attribute names so they can be understood easily.



Engineering in One Video (EIOV)

Watch video on YouTube

Decision table

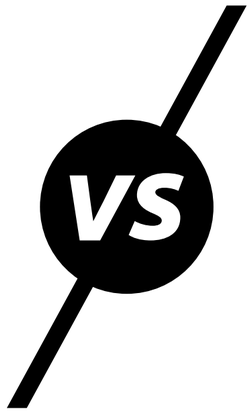
vs

Decision tree

Decision table vs Decision tree

Decision Table

1. Decision Tables are tabular representation of conditions and actions.
2. We can derive decision table from decision tree.
3. In Decision Tables, we can include more than one ‘or’ condition.
4. It is used when there are small number of properties.
5. It is used for simple logic only.



Decision Tree

1. Decision Trees are graphical representation of every possible outcome of a decision.
2. We can not derive decision tree from decision table.
3. In Decision Trees, we can not include more than one ‘or’ condition.
4. It is used when there are more number of properties.
5. It can be used for complex logic as well.

Engineering in One Video (EIOV)

Watch video on YouTube

SRS Document

EIOV

Engineering

IN ONE VIDEO

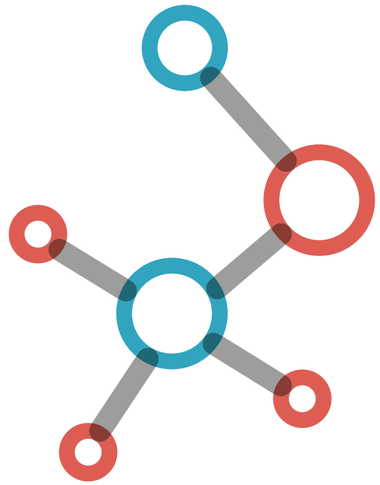
CONNECT, LEARN AND GROW

SRS Document

A Software Requirements Specification (SRS) is a document that describes the nature of a project, software or application. In short, the purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements.

SRS structure and its parts:

1. Introduction
2. General description
3. Functional Requirements
4. Interface Requirements
5. Performance Requirements
6. Design Constraints
7. Non-Functional Attributes
8. Preliminary Schedule and Budget
9. Appendices



SRS Document

1. Introduction:

(i) Purpose of this Document –
At first, main aim of why this document is necessary and what's purpose of document is explained and described.

(ii) Scope of this document –
In this, overall working and main objective of document and what value it will provide to customer is described and explained. It also includes a description of development cost and time required.

(iii) Overview –
In this, description of product is explained. It's simply summary or overall review of product.

SRS Document

2. General description:

It includes objective of user, a user characteristic, features, benefits, about why its importance is mentioned. It also describes features of user community and application domain.

3. Functional Requirements:

It Describes the possible effects of a software system, in other words, what the system must accomplish.
Each functional requirements should be specified in following manner:

- Description
- Criticality
- Technical issues
- Cost and schedule
- Risks

SRS Document

4. Interface Requirements:

In this, software interfaces which mean how software program communicates with each other or users either in form of any language, code, or message are fully described and explained.
Each functional requirements should be specified in following manner:
User interfaces
GUI
API
CLI

5. Performance Requirements:

In this, how a software system performs desired functions under specific condition is explained. Specifies speed and memory requirements.

SRS Document

6. Design Constraints:

In this, constraints which simply means limitation or restriction are specified and explained for design team. Examples may include use of a particular algorithm, hardware and software limitations, etc.

7. Non-Functional Attributes:

In this, non-functional attributes are explained that are required by software system for better performance. An example may include Security, Portability, Reliability, Reusability, Application compatibility, Data integrity, Scalability capacity, etc.

8. Preliminary Schedule and Budget:

In this, initial version and budget of project plan are explained which include overall time duration required and overall cost required for development of project.

9. Appendices :

Definitions, Acronyms, Abbreviations, References.

Characteristics of good SRS

1. **Correctness:** functionality, performance, design, constraints, attributes.
2. **Completeness**
3. **Consistency**
4. **Unambiguousness**
5. **Modifiability**
6. **Verifiability:** SRS is correct when the specified requirements can be verified with a cost-effective system to check whether the final software meets those requirements. The requirements are verified with the help of reviews.
7. **Testability**

Engineering in One Video (EIOV)

Watch video on YouTube

Software Quality Assurance(SQA)

Software Quality Assurance(SQA)

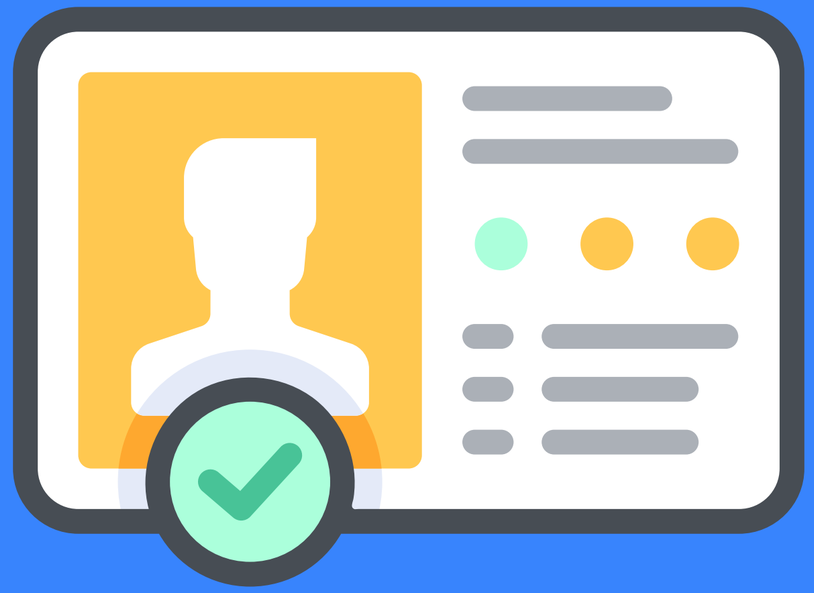
Software Quality Assurance (SQA) is simply a way to assure quality in the software. It is the set of activities which ensure processes, procedures as well as standards are suitable for the project and implemented correctly.
Software Quality Assurance is a process which works parallel to development of software. It focuses on improving the process of development of software so that problems can be prevented before they become a major issue.

Benefits of Software Quality Assurance (SQA):

1. SQA produces high quality software.
2. High quality application saves time and cost.
3. SQA is beneficial for better reliability.
4. SQA is beneficial in the condition of no maintenance for a long time.
5. High quality commercial software increase market share of company.
6. Improving the process of creating software.
7. Improves the quality of the software.

Engineering in One Video (EIOV)

Watch video on YouTube



Verification vs Validation

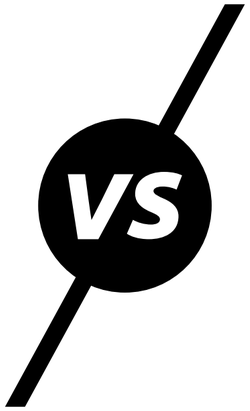
Engineering in One Video (EIOV)

Watch video on YouTube

Verification and Validation

Verification

1. We check whether we are developing the right product or not.
2. Verification is also known as static testing.
3. The execution of code does not happen in the verification testing.
4. In verification testing, we can find the bugs early in the development phase of the product.
5. Verification is done before the validation testing.



Validation

1. We check whether the developed product is right.
2. Validation is also known as dynamic testing.
3. In validation testing, the execution of code happens.
4. In the validation testing, we can find those bugs, which are not caught in the verification process.
5. After verification testing, validation testing takes place.

Objectives of verification and validation

- Determine that it performs its intended functions correctly.
- Ensure that it performs no unplanned functions.
- Measure and assess the quality and reliability of software.

Engineering in One Video (EIOV)

Watch video on YouTube

ISO 9000 models

ISO 9000 models

ISO (International Standards Organization) is a group or consortium of 63 countries established to plan and fosters standardization. ISO declared its 9000 series of standards in 1987. It serves as a reference for the contract between independent parties. The ISO 9000 standard determines the guidelines for maintaining a quality system. The ISO standard mainly addresses operational methods and organizational methods such as responsibilities, reporting, etc. ISO 9000 defines a set of guidelines for the production process and is not directly concerned about the product itself.

1. **ISO 9001:** This standard applies to the organizations engaged in design, development, production, and servicing of goods. This is the standard that applies to most software development organizations.
2. **ISO 9002:** This standard applies to those organizations which do not design products but are only involved in the production. Examples of these category industries contain steel and car manufacturing industries that buy the product and plants designs from external sources and are engaged in only manufacturing those products. Therefore, ISO 9002 does not apply to software development organizations.
3. ISO 9003: This standard applies to organizations that are involved only in the installation and testing of the products. For example, Gas companies.

ISO 9000 models

Why ISO Certification required by Software Industry?

There are several reasons why software industry must get an ISO certification. Some of reasons are as follows :

- This certification has become a standards for international bidding.
- It helps in designing high-quality repeatable software products.
- It emphasis need for proper documentation.
- It facilitates development of optimal processes and totally quality measurements.

Features of ISO 9001 Requirements :

- Document control
- Planning
- Review
- Testing
- Handling

Engineering in One Video (EIOV)

Watch video on YouTube

ISO
vs
SEI-CMM model

Engineering in One Video (EIOV)

Watch video on YouTube

ISO vs SEI-CMM model

ISO

1. ISO stands for International Organization for Standardization
2. It is created for hard goods manufacturing industries.
3. ISO9000 is recognized and accepted in most of the countries.
4. This establishes one acceptance level.
5. ts certification is valid for three years.
6. It focuses on inwardly processes.
7. It has no level.

SEI-CMM

1. SEI-CMM stands for Software Engineering Institute - Capability Maturity Model
2. It is created for software industry.
3. SEI-CMM is used in USA, less widely elsewhere.
4. It assesses on 5 levels.
5. It has It focus outwardly.no limit on certification.
6. It focus outwardly.
7. It has 5 levels:
   (a). Initial
   (b). Repeatable
   (c). Defined
   (d). Managed
   (e). Optimized

Happy Ending!

Congratulations!