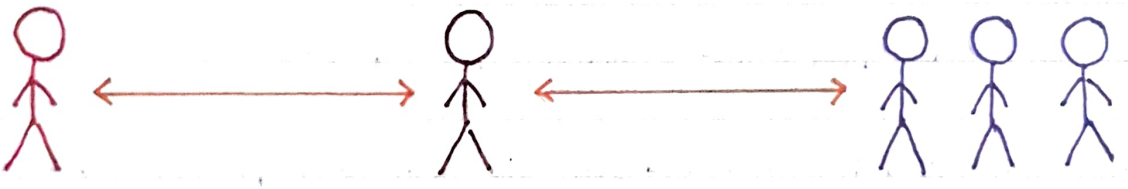
Introduction

The System Analyst

- Plays a key role in IS development projects.
- Works closely with all project team members so that the team develops the right system in an effective way.
- Understand how to apply technology in order to solve problems.
- Serve as change agents who;
  - identify organizational improvement needed
  - design systems to implement those changes
  - train and motivate others to use the systems



Client
Expert in business

System Analyst
Business Analyst (BA)

IT Team
Expert in technical

System Analyst Roles

Business Analyst

Focuses on the business issues surrounding the system

Systems Analyst

Focuses on the IS issues surrounding the system

Infrastructure Analyst

Focuses on technical issues

Change management Analyst

Focuses on the people and management issues surrounding the system installation

Project Manager

Ensures that the project is completed on time and within budget, and that the system delivers the expected vale to the organization

Systems Analyst Skills

Technical

Must understand the technical environment, technical foundation and technical solution.

Business

Must understand how IT can be applied to business situations

Analytical

Must be problem solvers

Interpersonal

Need to communicate effectively

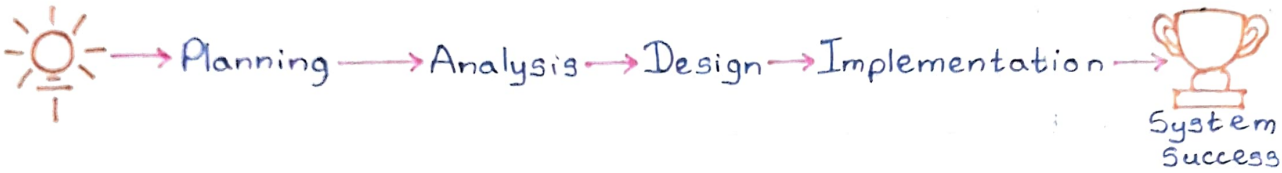
Management

Need to manage people and to manage pressure and risks

Ethical

Must deal fairly, honestly, and ethically with other project members, managers and system users.

The Systems Development Life Cycle (SDLC)



The SDLC is composed of four fundamental phases

Planning
Analysis
Design
Implementation

Each of the phases are composed of steps, which rely on techniques that produce deliverables (specific documents that explain various elements of the system).

Planning

- The fundamental process of understanding why an information system should be built, and determining how the project team will go about building it.

Steps

- Project identification and selection
- Project initiation and planning

Outcomes

- System request
- Feasibility report

Content of Feasibility Report

- Index
- Introduction
- Executive Summary

- Marketing Report
- Technical Report
- Financial Report
- Appendixes

Analysis

- The analysis phase answer the questions
  - who will use the system
  - what the system will do
  - where and when it will be used

Steps

- Study of the current system (as-is system)
- Requirements gathering

Outcome

- System proposal

Content of System Proposal

- Title page of project
- Table of contents
- Executive summary (including recommendations)
- Results of the systems study with appropriate documentation
- Systems alternatives (three or four possible solutions)
- Systems analysts' reccomendations
- Proposal summary
- Appendices (assorted documentation, summary of phases, correspondence and so on)

Design

- Decides how the system will operate, in terms of the hardware, software, and network infrastructure; the user interface, forms, and reports that will be used; and the specific programs, databases, and files that will be needed.

Steps

- Design Strategy
- Architecture Design
- Database and File Specifications
- Program Design

Outcome

- Systems specifications

Implementation

- The system is either developed or purchased (in the case of packaged software) and installed.
- Usually the longest and most expensive part of the process.

Steps

- System Construction
- Installation
- Support Plan

Planning an Information Systems Project

Feasibility Study

Practical extent to which a project can perform successfully is referred to as fisibility

Feasibility study determines whether the solution considered accomplish requirement is practical and workable in the software.

During feasibility study the following are considered.

- Resource availability
- Cost estimation for software development
- Benifits of software to the organization after it is developed
- Maintenance cost

Objectives of Feasibility Study

Establish reasons for developing software that is acceptable for users, adaptable for changes and confirmable to established standards.

Analyze whether the software will meet organization requirements.

Determine whether software can be implemented during current technology within specified budget and schedule

Determine whether the software can be

Feasibility Analysis identifies the important risks associated with the project that must be managed if the project is approved

Feasibility Analysis includes the following studies

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility
- Schedule Feasibility

Technical Feasibility

Technical Feasibility is the evaluation of whether we (IT group) are having enough technical resources to build the project.

"Can we build it?"

Considerations taken place in Technical Feasibility

- Required technology is available
- Required resources are available
- Man power (programmers, testers, debuggers, etc.) is available
- Software and Hardware required to make the project

Operational Feasibility

Operational Feasibility of the system is how well the system ultimately will be accepted by its users and incorporate into the ongoing operation of the organization

"If we build it, will they come?"

The Operational Feasibility answers the following

- Will the system be used?
- Will clients accept it?
- Any resistance from users?
  - Does the management support the project?
  - Are the users not happy with the current system? Will it reduce the time? Why will they like the new system?
  - Has the users been involved in planning and development?
- Will it benifit the organization? Does the overall response increase
- Will the accessibility of information be lost?
- Will the system affect customers in a considerable way?

Schedule Feasibility

Schedule Feasibility is the probability of a project to be completed within its scheduled time limits by a planned due date

If a project has high probability to be completed on-time, then its schedule feasibility is appraised as high.

If a project takes too time as some external environmental conditions may change hence the project can loose their benifits, profitability of the system

If the project can't be completed on time it is not schedule feasible

Schedule Feasibility can be checked by referring the following

Estimating the workload

Time available for each and every work

If it is not possible to do what we can do as a change management application

"Can we complete the project in the given time"

Economic Feasibility

Economic Feasibility analysis is also called a cost benifit analysis, that identifies the costs and benefits associated with the system

Deals with money or budget we have

Determines whether the required software is capable of generating the financial gains for the particular organization

It involves; cost incured on the software development team

estimated cost of hardware and software

cost of performing feasibility study.

It is essential considering expenses made on purchase

Software and hardware purchase

Human resource purchase

Software is set to be economically feasible if it focuses on some issues like

- Cost incured for software development for long term gains for an organization
- Cost required to conduct full software investigation
  - Requirement elicitation
  - Requirement analysis
- Cost of Hardware and Software development team training

Cost Benifit Analysis

Used for evaluating the effectiveness of the proposed system

In Economic Feasibility the most important is cost analysis

Cost Analysis

Analysis of the cost to be incured in the system and benifits deliverable out of the system

Development Costs

- Development team salaries
- Consultant fees
- Development training
- Hardware and Software
- Vendor installation
- Office space and equipment
- Data conversion costs

Operational Costs

- Software Upgrades
- Software licensing fees
- Hardware repairs
- Hardware upgrades
- Operational team salaries
- Communications charges
- User training

Tangible Benifits

- Increased sales
- Reduction in staff
- Reduction in inventory
- Reduction in IT costs
- Better supplier costs

Intangible Benifits

- Increased market share
- Increased brand recognition
- Higher quality products
- Improved customer service
- Better supplier relations

Steps of Feasibility Study

Collecting information

Information Assessment

Writing Feasibility Report

Collecting Information

Identify the information we have to search for and finding out facts related to each and every information.

These information are related to the four factors we are looking for

Information Assessment

After collecting all the information we have to analyze and come up with the result, whether the project is feasible or not

Writing Feasibility Report

Introduction about project

What it does

Details of the four feasible studies

Conclude the feasible study

Selecting Project Methodology

There is a project manager controlling and managing all set of activities happening throughout the SDLC

The manager has to follow a set of project management guidelines

These guidelines are referred to as project management life cycle (guides from inspection to completion)

Project Manager is doing

- Initiation of project
- Planning of project
- Execution of project
- Control of project
- Enclosure of project

In large organizations or large projects, the role of the project manager is fulfilled by a professional specialist in project management

In smaller organizations or small projects, the system analyst may fill this role

System Analyst may work as a Project Manager in some projects.

Project Manager has to make a set of decisions regarding the project including

- determining the best project methodology
- developing a project plan
- determining a staffing plan
- establish mechanisms to coordinate and control the project

Methodology

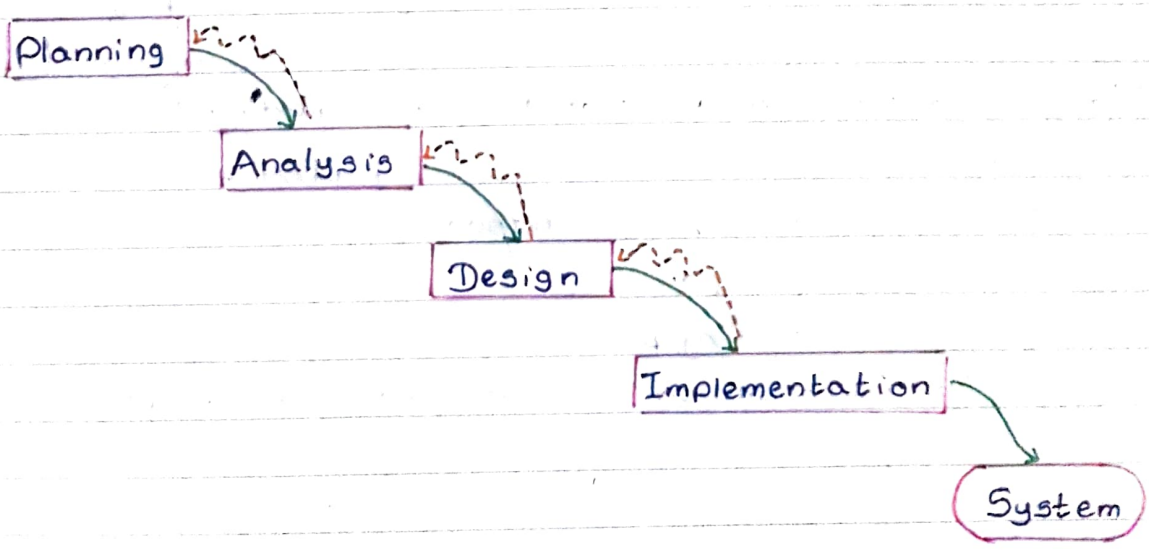
A methodology is a formalized approach to implementing the SDLC

Some methodologies have formal standards used by goverment agencies while others have been developed by consulting forums to be sold to clients. Many organizations have their internal methodologies that have been refined over the years and they explain exactly how each phase of the SDLC is to be performed in that company.

Methodologies Types

- Waterfall Development
- Parallel Development
- V model (variation of waterfall development)
- Rapid Application Development (RAD)
  - Iterative Development
  - Prototyping
- Agile Development

Waterfall Development



Analyst and users proceed sequentially from one phase to the next. Project will start from Planning then Analysis, Design and finally Implementation.

Key typicals for each phase are large documents. Sometimes may contain 1000s of pages and these documents are presented to the approval committe (client side people and IT organization) Documents should be approved before moving from one phase to the other.

Advantages

Requirements identified long before the programming begins making limiting of requirements changes when project proceeds.

Disadvantages

The design must be completely specified before programming begins.
Long time ilaccess between completion of system proposal in analysis phase and delivery of the system.
Testing is created almost as a more entuar activity that is happening in the implementation phase.
The deliverables are often a poor communication mechanism so important requirements may be overlooked in the volumes of documentation which are provided at each and every stage. So if the project team missess and important requirement expensive post implementation programming may be needed.
Users may forget the original purpouses of the system since the original idea and actual implementation.

Also in today's dynamic business environment the system that met the existing environment conditions during the analysis phase may need considerable reworks to watch the environment when it

is implemented.

Sometimes this Waterfall Development Methodology is very limited to be applied in current organization. The rework requires going backwards to initial phase and make required changes through each of the subsequent phases internally

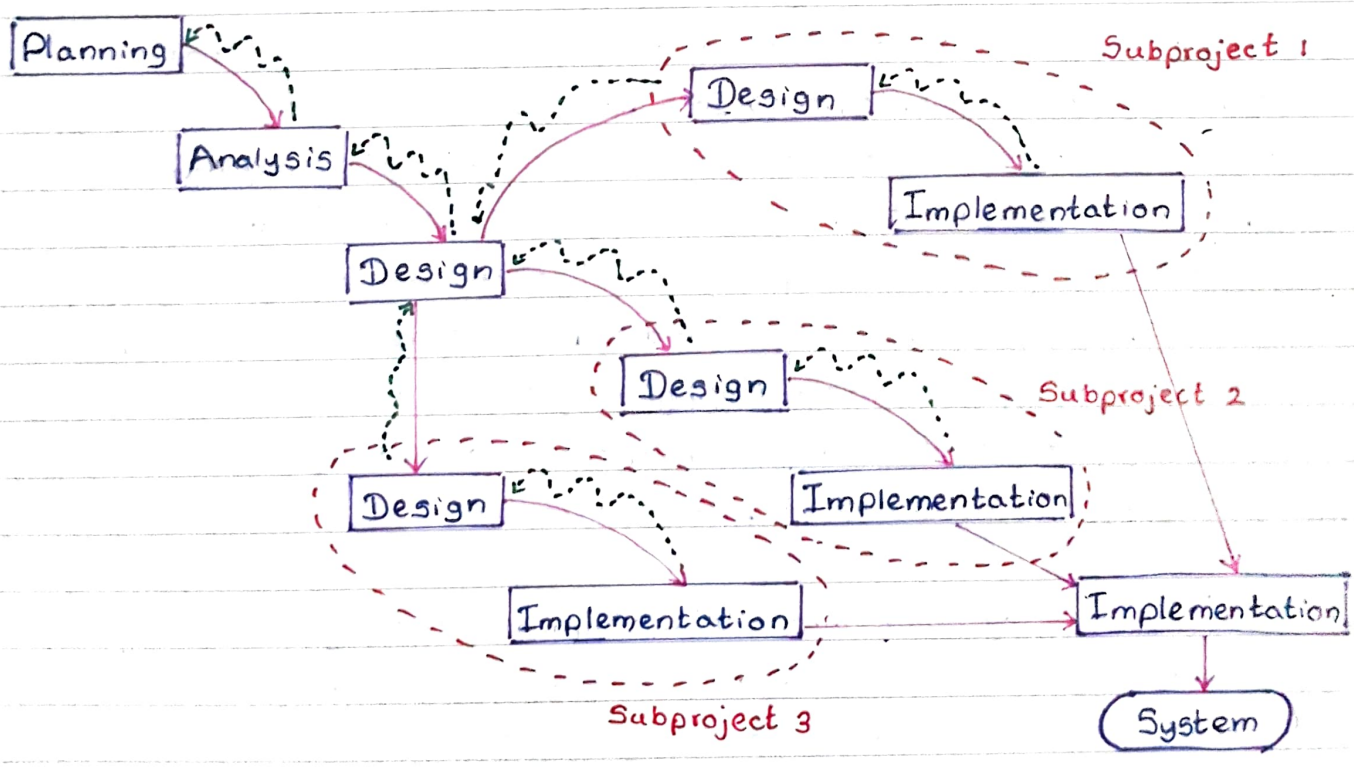
Variants of the Waterfall Development Methodology

- Parallel Development Methodology
- V-Model Methodology

Parallel Development

Evolved to address the lengthy time frames of waterfall developm-ent methodology.

Instead of doing Design and Implementation in sequence a general design of the whole system is performed and the project is divided into a series of subprojects that can be designed and implemented in parallel



Advantages

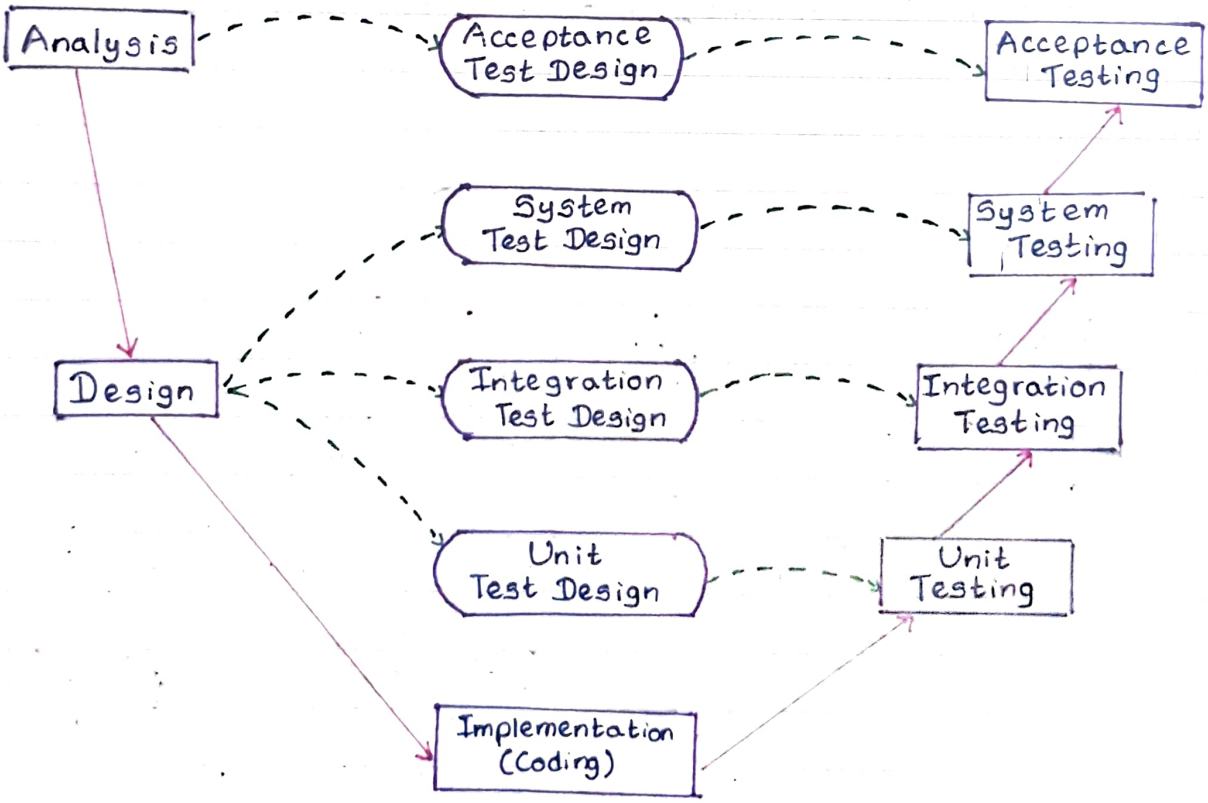
Reduces the time required to deliver a system, so it's less affacted by the changes of the business environment

Disadvantages

Suffers from problems caused by volumeless deliverables

If the sub-projects are not completely independant designed sessions in one sub project may affect another and at the end intergrating the projects may be challanging.

V Model Methodology



Each level of testing is clearly linked to a part of analysis or design phase helping to ensure high quality and relevant testing and maximize test effectiveness

Suffers from rigidity of waterfall development process

Rapid Application Development (RAD)

A combination of

- CASE (computer-aided software engineering) Tools
- JAD Sessions (joint application development)
- Fourth generation/visual programming languages (Visual Basic, .NET)
- Code generators

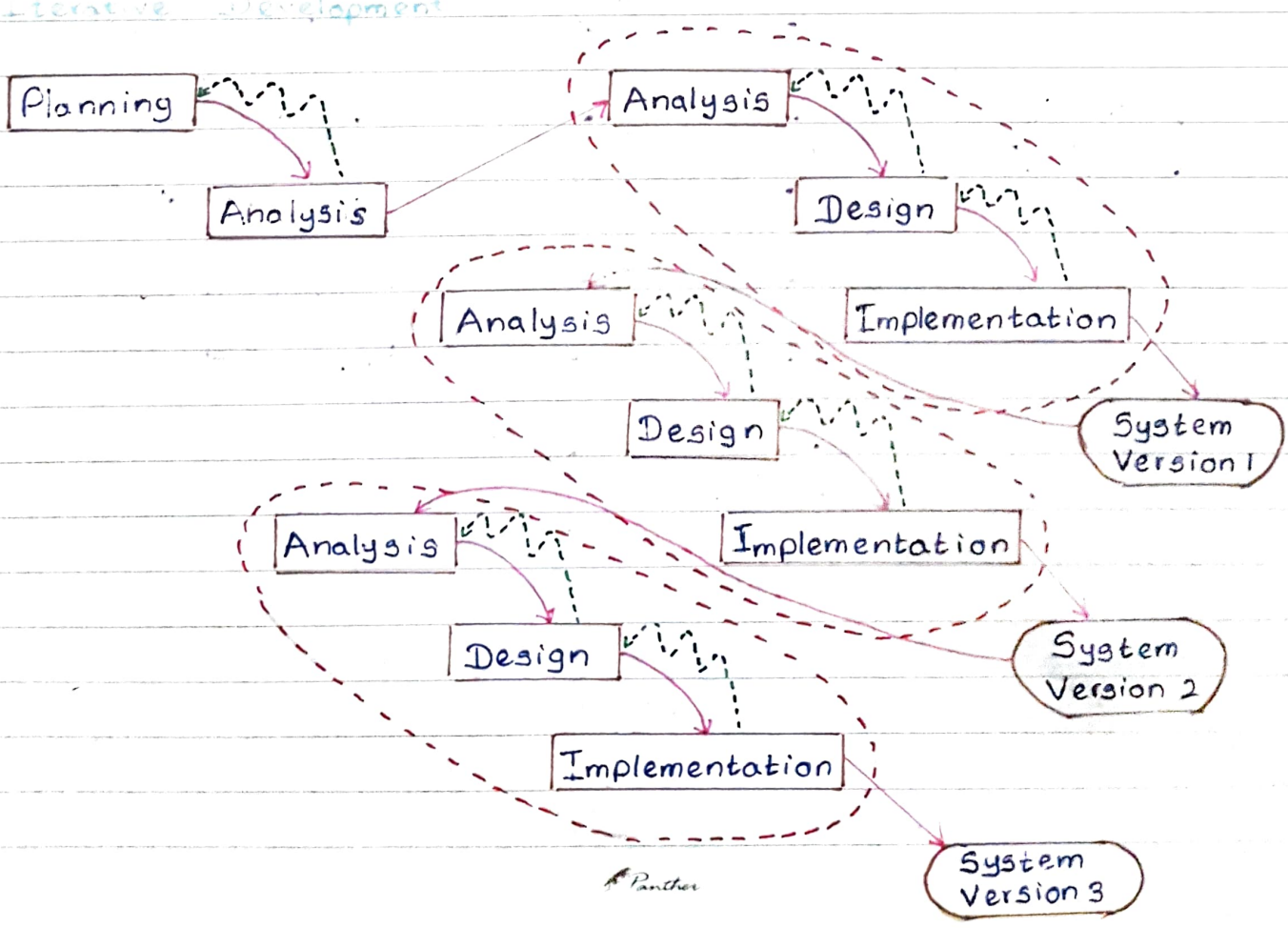
Disadvantage

User expectation may increase dramatically and system requirements may expire (scope creep/feature creep)

Rapid application Development can occur in two ways

- Iterative Development
- Prototyping

Iterative Development



Breaks the overall project into a series of versions that are developed sequentially

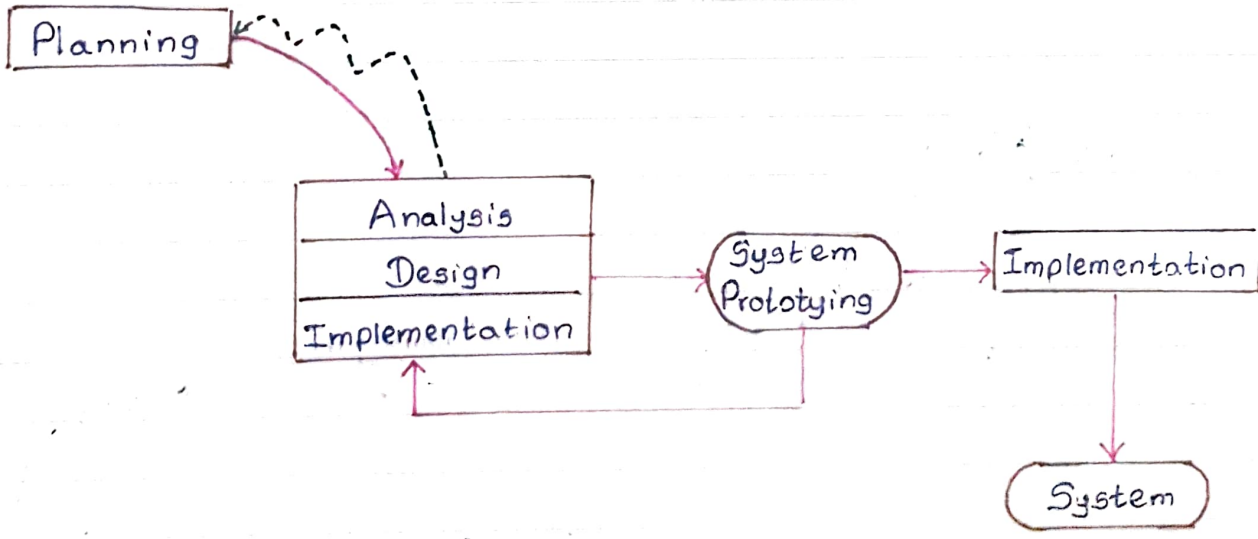
Advantages

Get preliminary version of the system for the users quickly.
As users work with the system important additional requirements may be identified and incorporated into sub sequent versions.

Disadvantages

Users begin to work with a system which is intensionally incomplete

System Prototyping



Performs the analysis, design and implementation phases concurrently in order to quickly develop a simplified version of the proposed system and give it to the users for evaluation and feedback
Called "quick and dirty" version of the system and provide minimal features

Advantages

Quickly provides the system to users to evaluate and reassure users that progress is made.

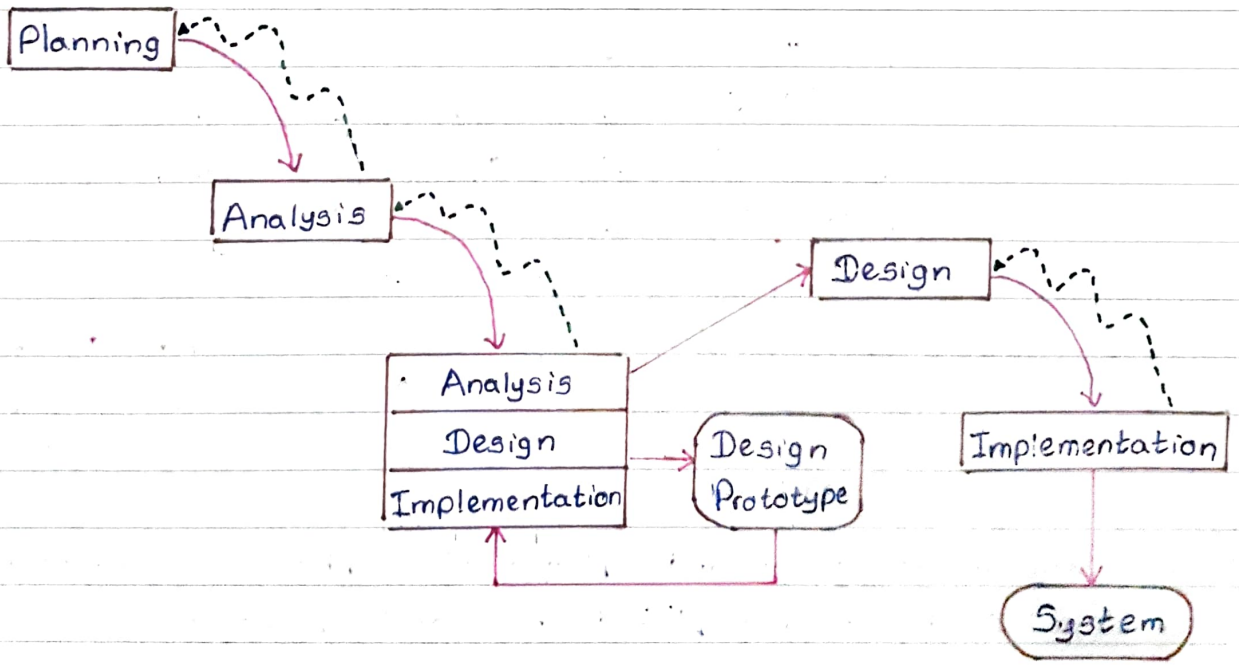
Approach is useful when users face difficulty expressing requireme-nts for the system.

Disadvantages

Lack of careful methodological analysis prior to make design and implementation decision

May have some fundamental design limitations that is a direct result of inadequet understanding of the system's true requirements early in the project.

Throwaway Prototyping



Refers to the creation of a model that will eventually be discarded rather than becoming part of the final delivered system.

Advantages

Using prototypes to refine the issue before the system is built

Produce more reliable and stable system

Disadvantages

It may take longer to deliver the system

Agile Development

A ground of programming-centric methodologies that focus on streamlining the SDLC

Includes face to face communication

Cycles are kept short (one to four weeks)

Several popular approaches to agile development

- extreme programming (XP)
- scrum
- dynamic systems development method (DSDM)

Extreme Programming

Emphasises customer satisfaction and team work.

Communication, simplicity, feedback, courage are core values

Project teams are kept small

Users are required to be available throughout the project to clear up questions and issues that may arise

The team uses common set of names and coding practices

Delivers results sooner even than RAD

Rarely get bogged down in gathering requirements for the system

Selecting the Appropriate Development Methodology

Important factors to consider in selecting the development methodology

- Clarity of user requirements
- Familiarity with Technology
- System Complexity
- System Reliability
- Short Time Schedules
- Schedule Visibility

Clarity of User Requirements

System Prototyping and Throwaway Prototyping methods are more appropriate when user requirements are unclear

Agile Development may be appropriate if on site user input is available

If user requirements are very clear waterfall development can be implemented.

Familiarity with Technology

When the system will use new technology which the analysts and the programmers are not familiar, applying the new technology early in the methodology will improve the chance of success.

Throwaway Prototyping is suitable when there is lack of familiarity with technology

Itarative Development is also good as opportunities are created to investigate the technology before Design is complete.

System Complexity

For complex systems Throwaway Prototyping is well suited but System Prototyping is not

Waterfall Development Methodology can be used but some key issues

may be overlooked
Although Iterative Development enable users to interact with the system early in the process, the project team tends to devote less attention to the problem domain.

System Reliability

V model is useful when reliability is important due to its emphasis on testing
Throwaway Prototyping is most appropriate when system reliability is a high priority.
System Prototyping is not a good choice when system reliability is critical

Short Time Schedules

Rapid Application Methodologies are well suited
Iterative Development and System Prototyping are excellent choices when time lines are short.
Waterfall based methodologies are the worst when time is short

Schedule Visibility

Waterfall Development methodologies are challenging as design and implementation occurs at end
RAD based methodologies move many critical design decisions earlier in the project.

Usefulness in Developing Systems	Structured Methodologies			RAD Methodologies			Agile Methodology
	Waterfall	Parallel	V-Model	Iterative	System Prototyping	Throwaway Prototyping	Extreme Programming
With unclear user requirements	Poor	Poor	Poor	Good	Excellent	Excellent	Excellent
With unfamiliar technology	Poor	Poor	Poor	Good	Poor	Excellent	Poor
That are complex	Good	Good	Good	Good	Poor	Excellent	Poor
That are reliable	Good	Good	Excellent	Good	Poor	Excellent	Good
With short time schedule	Poor	Good	Poor	Excellent	Excellent	Good	Excellent
With schedule Visibility	Poor	Poor	Poor	Excellent	Excellent	Good	Good

Time, Scope and Staff Management

Time Estimation

Estimation is the process of assigning projected values for time and effort

Estimation can be performed manually or with the help of an estimation software package like Costar, KnowledgePLAN

The estimates developed at the start of a project are usually based on range of possible values (e.g. the design phase will take three to four months) and gradually becomes more specific as the project moves forward (e.g. design phase will be completed on March 22)

Numbers used to calculate estimates come from several sources

- Methodology that is used
- Projects with similar tasks and technologies
- Provided by experienced developers

There are two basic ways to estimate the time required to build a system

1) Simplest Method uses the amount of time spent in the planning to predict the time required for the entire project
   Can use industry standard percentages to calculate estimates for SDLC phases
   It can be difficult to take into account the specifics of your individual project, which may be simpler or more difficult than "typical project".
2) Function Point Approach is more precise and more complex and it is hoped, more reliable way of estimating time and effort for a project.

	Planning	Analysis	Design	Implementation
Typical industry standards for business applications	15%.	20%.	35%.	30%.
Estimates based on actual figures for first stage of SDLC	Actual: 4 person-months	Estimated: 5.33 person-months	Estimated: 9.33 person-months	Estimated: 8 person-months

* Estimating Project Time Using Industry Standards

Developing the Work Plan

Work Plan is a dynamic schedule that records and keep track of all the tasks that need to be accomplished over the course of the project.

To create a work plan, the project manager identifies the tasks that need to be accomplished and determines how long each one will take. Then the tasks are organized within a work breakdown structure

Identify Tasks

A project manager can take the methodology, select the steps and deliverables that apply to the current project, and add them to work plan

If an existing methodology is not available within the organization, methodologies can be purchased from consultants or vendors, or books can serve as guidance

Using an existing methodology is the most popular way to create a work plan, because most organizations have a

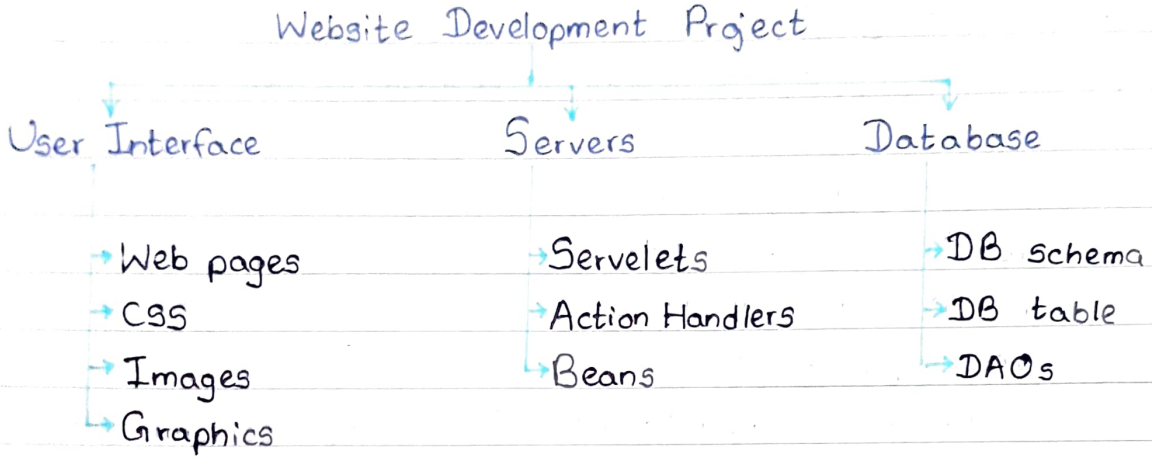
methodology that they use for projects.
The methodology that seems more appropriate for the project provides a list of steps and deliverables

Task Information	Example
Name of the task	Perform economic feasibility
Start date	Jan 05, 2020
Completion date	Jan 19, 2020
Person assigned to the task	Project sponsor Mary Smith
Deliverables	Cost-benifit analysis
Completion status	Complete
Priority	High
Resources needed	Spreadsheet software
Estimated time	16 hours
Actual time	14.5 hours

Work Breakdown

If a project manager prefers to begin from scratch, he or she can use a structured, top-down approach whereby high-level tasks are defined first and then broken down into subtasks.
Each step is then broken down in turn and numbered in a hierarchi-cal fashion.
A list of hierarchically numbered in this way is called a "work breakdown structure" and it is the backbone of the project workplan.
A "deliverable oriented hierarchical decomposition of the work to be executed by the project team.
It provides the framework for all deliverables throughout the project life cycle

Example



Task ID	Task Name	Duration	Dependency	Status
1	Design Phase	30		Open
1.1	Develop DB design doc	9		Open
1.1.1	Staging DB design	9		Open
1.1.2	Suspense DB design	9		Open
1.2	Develop rejects-handling design document	9	1.1.1, 1.1.2	Open
1.2.1	Rejects-handling engine	9		Open
1.3	Develop OLAP design doc.	8	1.1.1, 1.1.2	Open
1.3.1	Universe design	8		Open
1.4	Develop OLAP design P1	9		Open
1.4.1	High priority reports	8		Open
1.5	Develop app design doc	9		Open
1.5.1	Group consolidation, corporate reporting	9		Open
1.6	Extract, transform, load design document	2		Open

Portion of work breakdown structure

Project Work Plan

Project Work Plan is the mechanism used to manage the tasks that are listed in the work breakdown structure

Work plan is basically a table that lists all of the tasks in the work breakdown structure, along with important task information such as

- People who are assigned to perform the tasks
- The actual hours that the task took
- Variances between estimated and actual completion times

Task ID	Task Name	Assigned To	Estimated D.	Actual D.	Dependency	Status

Basic Structure of Work Plan

Gantt Chart

Used to plan projects of all sizes and useful way to show what work is scheduled to be done in a specific day

On a Gantt Chart you can see

- Start date of the project
- What the project tasks are
- Who is working on each task
- When task start and finish
- How long will each task take
- How tasks are grouped together, overlap and link with each other
- Finish date of the project

Activity

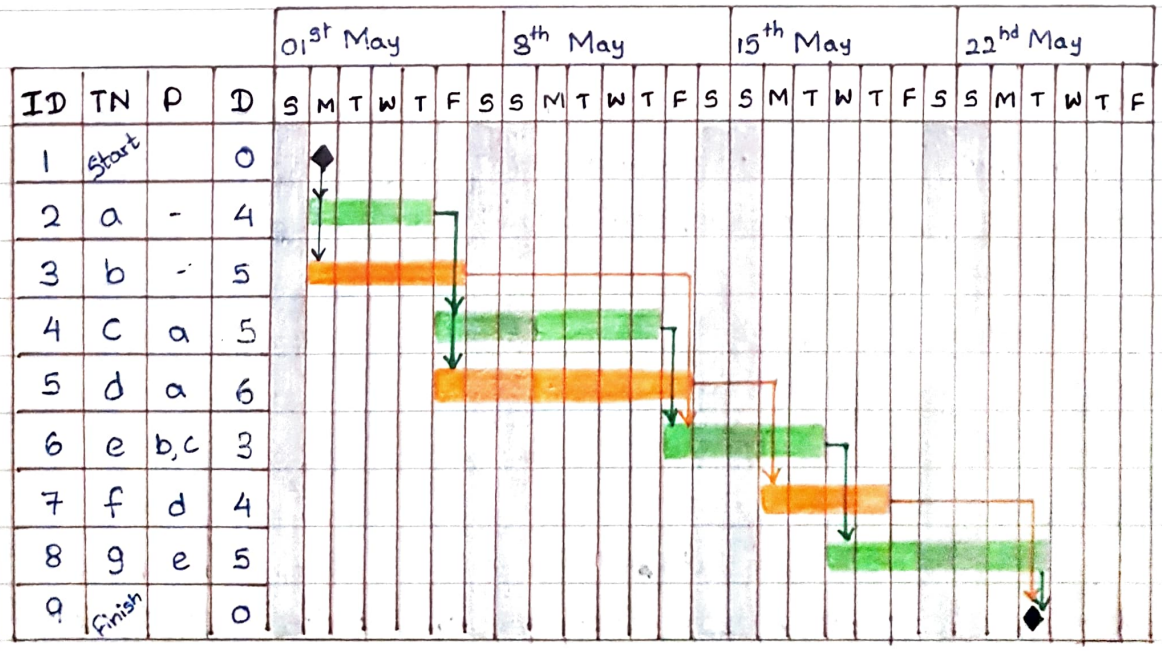
Create a Gantt chart for the project that includes the activities in the given table

Project will start on Monday (2nd May)

First day of the week in Gantt chart should be Sunday

Saturday and Sunday should be considered as holidays

Activity	Predecessor	Expected time (Days)
a	–	4
b	–	5
c	a	5
d	a	6
e	b, c	3
f	d	4
g	e	5



Staffing the Project

Staffing the project includes

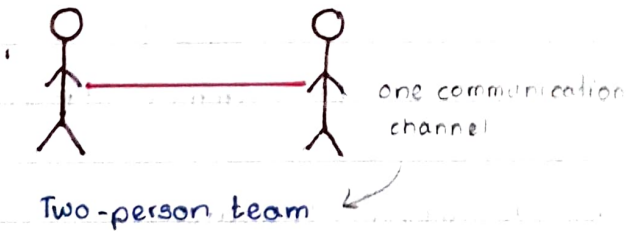
Determining how many people should be assigned to the project
Matching people's skills with the needs of the project
Motivating them to meet the project's objectives
Minimizing project team conflict that will occur over time

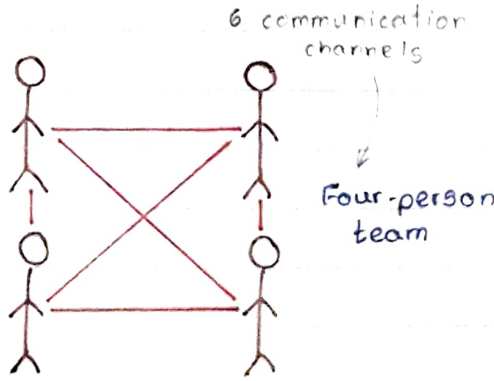
Staffing Plan

The first step to staffing is determining the avarage number of staff needed for the project

$$\text{Avg. no of staff} = \frac{\text{Total person-months of effort}}{\text{Optimal schedule}}$$

Temptation is to assign more staff to a project to shorten the project's length, but it's not wise
Staff size and productivity share a disproportionate relationship, mainly because large number of staff is hard to handle
Staff increasement lead to more dramatic gains in communication complexity.





One way to reduce efficiency losses on teams is to understand the complexity that is created in numbers and to build in a reporting structure that tempers its effects.

Staffing plan descripes the kinds of people working on project

Reporting Structure

The rule of thumb is to keep team sizes under 8 to 10 people

If more people are needed create subteams

A "functional lead" usually is assigned to manage a group of analysts

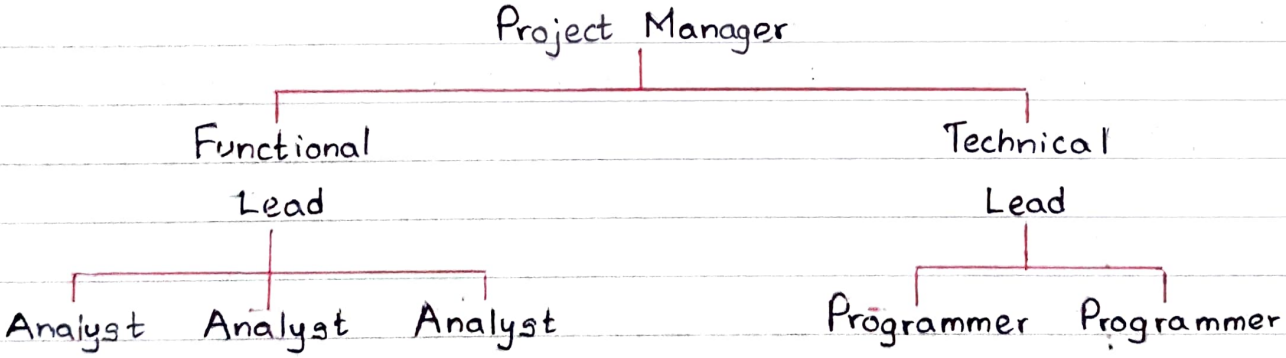
A "technical lead" oversees the progress of a group of programmers

People have both "technical skills" and "interpersonal skills", and both are important

Technical skills are useful for working with technical tasks and in trying to understand the various roles that technology plays in a particular project

Interpersonal skills are used when dealing with business users, senior management executives and other members of the team

Project charter describes project's objectives and rules



Possible Reporting Structure

Coordinating Project Activities

The act of coordinating project activities continues throughout the entire project until a system is delivered to the project sponsor and end users

CASE Tools

Computer Aided Software Engineering (CASE) is a category of softwar that automates all or part of the development process

Upper CASE used during the analysis phase to create integrated diagrams of the system and to store information regarding the system components

Lower CASE design phase tools that create the diagrams and then generate code for database tables and system functionality

Integrated CASE contains functionality found in both upper CASE and lower CASE tools in that it supports tasks that happen throughout the SDLC

Some of the good programs available in the market are

- Visible Analyst Workbench
- Oracle Designer
- Rational Rose
- Logic Works suite

CASE repository is the central component
Refered as information repository or data dictionary
Stores diagrams and other information like screen, report design
Keep track how diagrams fit together

Benifits of CASE

Tasks are faster to complete and alter
Development information is centralized
Information illustrated through diagrams
Reduce maintenance cost
Improve software quality
Enforce descipline
Assess the magnitude of changes to project

Analysis of an Information Systems Project

Analysis

Analysis refers to breaking a whole into its parts with the intent of understanding the parts' nature, function and interrelationships

Planning phase deliverables are the key inputs into the analysis phase

In Analysis phase, the system analyst works extensively with the business users of the new system to understand their needs from the new system

The basic process of analysis involves three steps

- Understand the existing situation (as-is system)
- Identify improvements
- Define requirements for the new system (to-be system)

To move the users from "as-is system" to the "to-be system" the analyst needs strong critical thinking skills.

Final deliverable of the analysis phase is the "system proposal"

System Proposal compiles the detailed requirements definition statement, use cases, process models, data model together with a revised feasibility analysis and work plan.

System proposal is presented to the approval committee in the form of a system "walk-through".

Requirements Determination

Requirements determination is performed to transform the system request's high level statement of business requirements into a more detailed, precise list of what the new system must do to provide the needed value to the business.

What is a Requirement?

A requirement is simply a statement of what the system must do or what characteristics it needs to have.

Business Requirements
What the business needs
User Requirements
What the user needs to do
Functional Requirements
What software should do
Non Functional Requirements
Characteristics the system should have

Activity

Categorize these requirements into business (B), user (U), functional (F) and non functional (N) requirements.

be accessible to web users (N)
include the company logo and color scheme (N)
restrict access to profitability information (N)
include actual and budgeted cost information (F)
provide management reports (F)
have 2 second maximum response time for predefined queries and 10 minute maximum response time for ad hoc queries (N)
include information from all company subsidiaries (F)
print subsidiary reports in the primary language of the subsidiary (N)
provide monthly rankings of salesperson performance (F)
include sales information that is updated daily
increase market share (B)
shorten order processing time (F)
reduce customer service costs (F)
lower inventory spoilage (F)
improve responsiveness to customer service requests
schedule a client appointment (U)
place a new customer order (U)
re-order the inventory (U)
determine available credit of clients
look up account balances (U)

Requirements Elicitation Techniques

Interviews

In general, interviews are conducted one on one (one interviewer and one interviewee) but sometimes due to time constraints several people are interviewed at the same time

There're five basic steps to the interview process

- Selecting interviewees
- Designing interview questions
- Preparing for the interview
- Conducting the interview
- Post-interview follow up

Selecting interviewees

Who has relevant information

Who is accessible

Who is willing to give relevant information

Who is most able to give the information.

Designing interview questions

Closed-ended questions

- Require specific answers
- Used when the analyst is looking for specific, precise information
- Don't uncover why the answer is the way it is, nor do they uncover information that the interviewer does not think to ask ahead of time.

How many telephone orders are received per day?

How do customers place orders?

What information is missing from the monthly sales report?

Do you handle a lot of request?

Open ended questions

Leave room for elaboration on the part of interviewee

Designed to gather rich information and give the interviewee more control over the information that is revealed during the interview.

What do you think about the way invoices are currently handled?

What are some of the problems you face on a daily basis?

What are some of the improvements you would like to see in the way invoices are processed?

Probing questions

Follow up on what has just been discussed in order for the interviewer to learn more, and they often are used when the interviewer is unclear of the interviewee's answer.

Why?

Can you give me an example?

Can you explain that in a bit more detail

Face to Face or Remote Interviewing

	Face to Face	Remote
+	Non verbal information Contexual information Control over interview situation Trust binding	Time saving Money saving Maybe better acceptance rate because interviews can be sliced
—	Expensive Time consuming	Only acoustic information Probably shorter answer as they must be written down Reduced control over interview

Advantages

- Simple, direct technique
- Enable observation of non-verbal behaviour
- Useful before developing a questionnaire
- Flexible and adaptable to individual situation
- Allow the participants and interviewer to have full discussion

Disadvantages

- Large amount of qualitative data hard to analyse
- Training is required to conduct effective interview
- Interview is costly in terms of money and time
- The interview results are often affected by interviewer's mode of asking question
- Documentation depend on interviewer's interpretation

Joint Application Development (JAD)

- Allows the project team, users and management to work together to identify requirements for the system
- Can reduce scope creep by 50%.
- Structure process in which 10 to 20 users meet under the direction of a "facilitator" skilled in JAD technique
- Facilitator is a person who sets the meeting agenda and guides the discussion, but does not join in the discussion as a participant

Advantages

- Allow key users to participate effectively
- Users are more likely to feel a sense of ownership
- Produces a more accurate statement of system requirements, a better understanding of common goals, and stronger commitment to the success of the new system

Disadvantages

More expensive than traditional methods
Can be cumbersome if the group is too large
Requires significant planning and scheduling effort
It opens up a lot of scope for interpersonal conflict

Document Analysis

Used to understand the as-is system
Forms, reports, policy manuals, organization charts represent describe the formal system that the organization uses

Advantages

Being in the language and words of the participants
Ready to analysis without the necessary transcription that is required observational on interview data

Disadvantages

Documents are sometimes difficult to locate and obtain
Information may not be available to the public
Documents may be incomplete, inauthentic or inaccurate
In the personal documents such as diaries or letters, the handwrit-ing may be hard to read

Observation

Observation is the act of watching process being performed
Powerful tool to gain insight into the as-is system
Enables the analyst to see the reality of a situation, rather than listening to others describe it in interviews or JAD sessions

Advantages

Get real time specific insights from user behaviour

Avoid the bias of people telling you what you want to hear, rather than what they need.
It is relatively inexpensive
Can do work measurements

Disadvantages

Exceptions are difficult to capture in one session
Prone to bias reflected in the form of seeing what they expect to see and what they want to see
People may perform differently when being observed

Questionnaires

Set of written questions for obtaining information from individua-ls.
Often used when there is a large number of people from whom information and opinions are needed

Good Questionnaire Design

Begin with nonthreatening and interesting questions
Group items into logically coherent sections
Do not put important items at the very end of the questionnaire
Do not crowd a page with too many items
Avoid abbreviations
Avoid biased or suggestive items or terms
Number questions to avoid confusion
Pretest the questionnaire to identify confusing questions
Provide anonymity to respondents.

Advantages

Often can be answered quickly
People can complete at their convenience

Relatively inexpensive way to gather data from a large number
Allow for anonymity
Response can be quickly tabulated.

Disadvantages

Return rate is often low
No guarantee than an individual will answer all questions
No opportunity to reword or explain misunderstood questions
Can not observe body language
Difficult to prepare

	Interviews	JAD	Questionnaries	Document Analysis	Observation
Type of information	As-is improvements to-be	As-is improvements to-be	As-is improvements	As-is	As-is
Depth of information	High	High	Medium	Low	Low
Breadth of information	Low	Medium	High	High	Low
Integration of information	Low	High	Low	Low	Low
User involvement	Medium	High	Low	Low	Low
Cost	Medium	Low Medium	Low	Low	Low Medium

Process Modeling using DFD

Introduction

Process model is a graphical way of representing how a business system should operate

Further clarifies the requirements definition and use cases

Can be used to document the as-is system or the to-be system, whether computerized

Different techniques of process model

- Business process modeling notation (BPMN)
- Flowchart
- Data Flow Diagram
- Integration Definition for Function Modeling (IDEF)

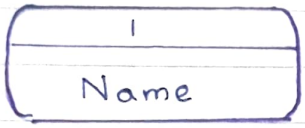
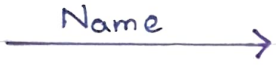
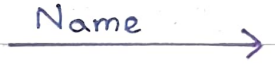
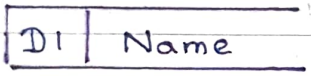
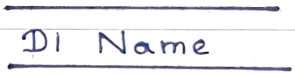
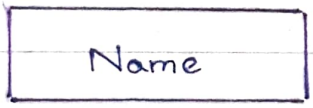
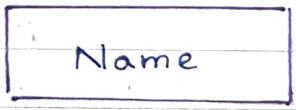
Data Flow Diagrams (DFD)

Technique that diagrams the business process and the data that passes among them

Two types of models

- Logical Process Model
  - Describe processes, without suggesting how they are conducted.
- Physical Process Model
  - Provide information that is needed to ultimately build the system

Elements of Data Flow Diagrams

	Gane and Sarson Symbol	DeMarco and Yourdan Symbol
Process		
Data Flow		
Data Store		
External Entity		

Process

Activity or a function that is performed for some

Data Flow

Single piece of data, or a logical collection of several pieces of information.

Data Store

Collection of data that is stored in some way

External Entity

Person, organization, organization unit or system that is external to the system but interacts with it.

Activity

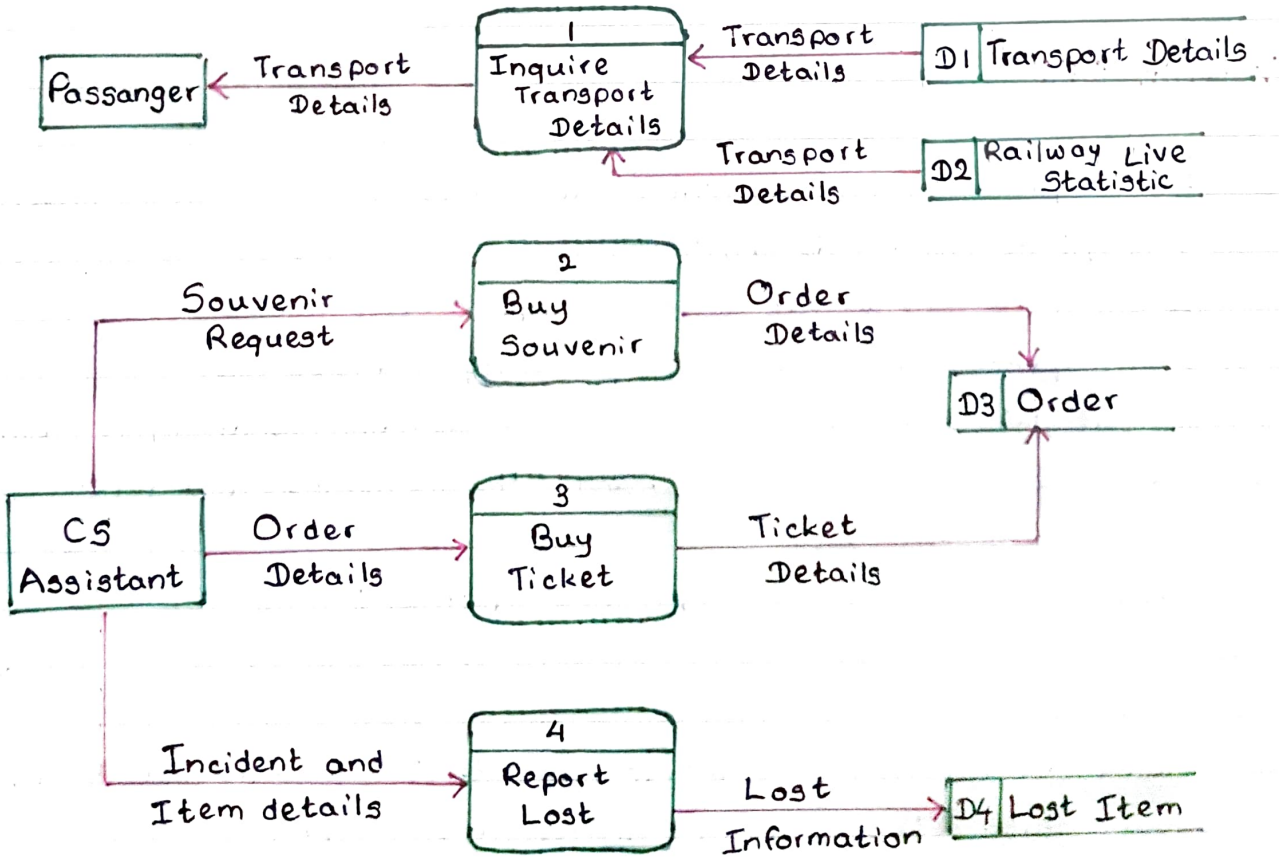
Customer Service System

A passenger can receive Railway Transport details from the Inquiry Transport Details process and these details are provided by the data stores Transport Details and Live Statistic

CS Assistant can initiate the Buy Souvenir process providing a Souvenir request, which will result in having the Order details stored in the Order data store.

CS Assistant can also initiate the Buy Ticket process by providing Order details and the Ticket details will be stored again in the Order data store

Finally CS Assistant can initiate the Report Lost process by providing the Incident and Item details and Lost information will be stored in the Lost Item database.



Validating the DFD

There are two fundamentally different types of problems that can occur in DFDs

Syntax Error

Refers to the structure of the DFDs and whether the DFDs follow the rules of DFD language

Thought as grammatical errors made by the analyst

Semantics Error

Refers to the meaning of the DFDs and whether they accurately describe the business process being modeled.

Semantics errors can be thought of as misunderstandings by the analyst in collecting, analyzing, and reporting information about the system.

Syntax Errors are easity identify than semantics errors

Within DFD

Process

Every process has a unique name that is an action-oriented verb phrase and a description

Every process have at least one input data flow (Miracle error)

Every process have at least one output data flow (Black hole)

Output data flows usually have different names than input data flows because the process changes the input into a different output in some way

There are between 3 to 7 processes per DFD

Data Flow

Every data flow has a unique name that is a noun and description

Every data flow connects to at least one process

Data flows only in one direction

A minimum number of data flow lines cross

Data Store

Every data store has a unique name that is a noun and descript
-ion.

Every data store has at least one input data flow

Every data store has at least one output data flow

External Entity

Every external entity has a unique name that is a noun and description

Every external entity has at least one input or output data flow

Across DFDs

Context diagram

Every set of DFDs must have one context diagram

Viewpoint

There is a cosistent viewpoint for the entire set of DFDs

Decomposition

Every process is wholly and completely described by the processes on its children DFDs

Balance

Every data flow, data store and external entity on a higher level DFD is shown on the lower-level DFD that decomposes it

Semantics

Appropriate Representation

User validation

Role-play process

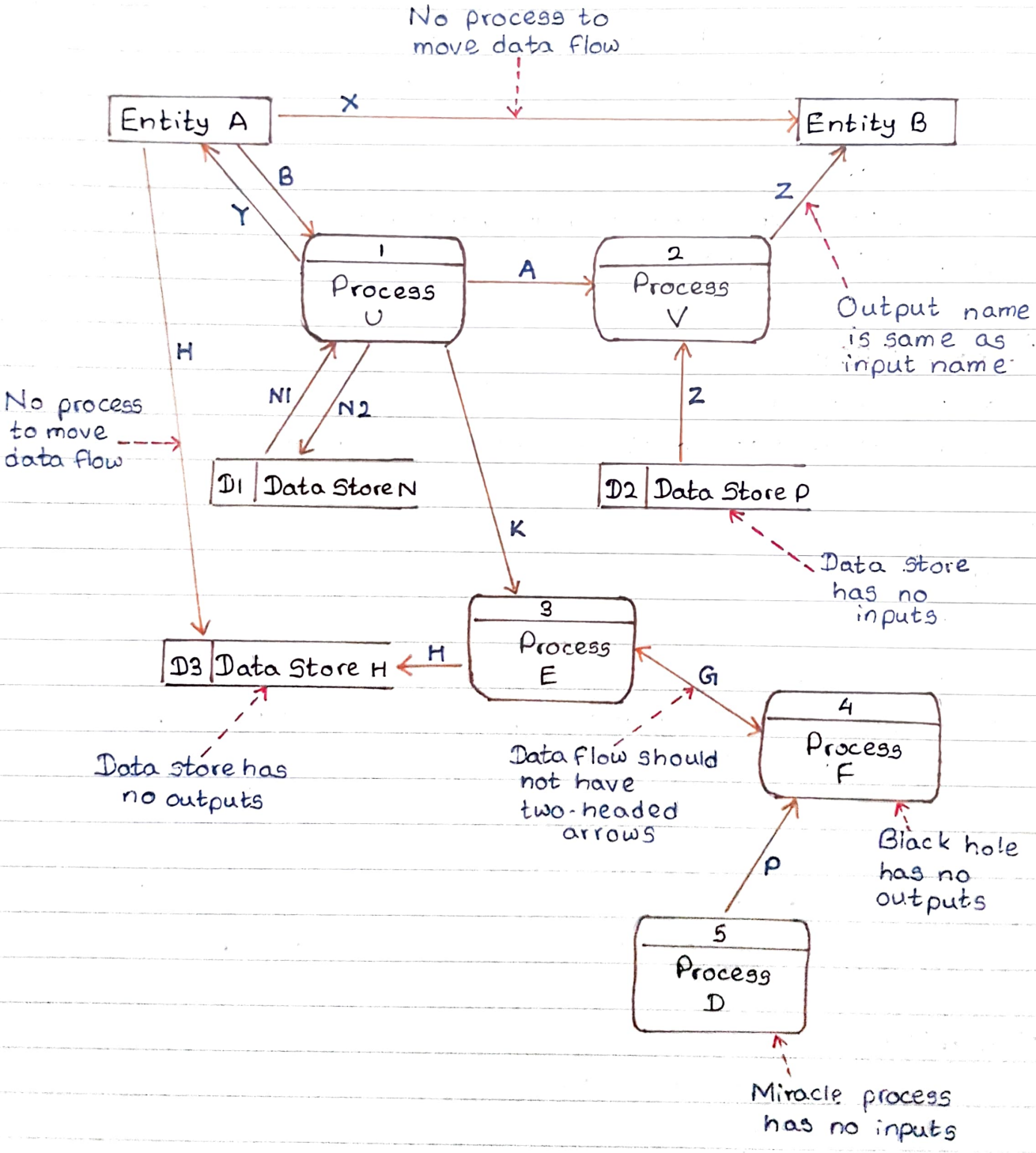
Consistent Decomposition

Examine lowest level DFDs

Cosistent Terminology

Examine names carefully

Some common errors

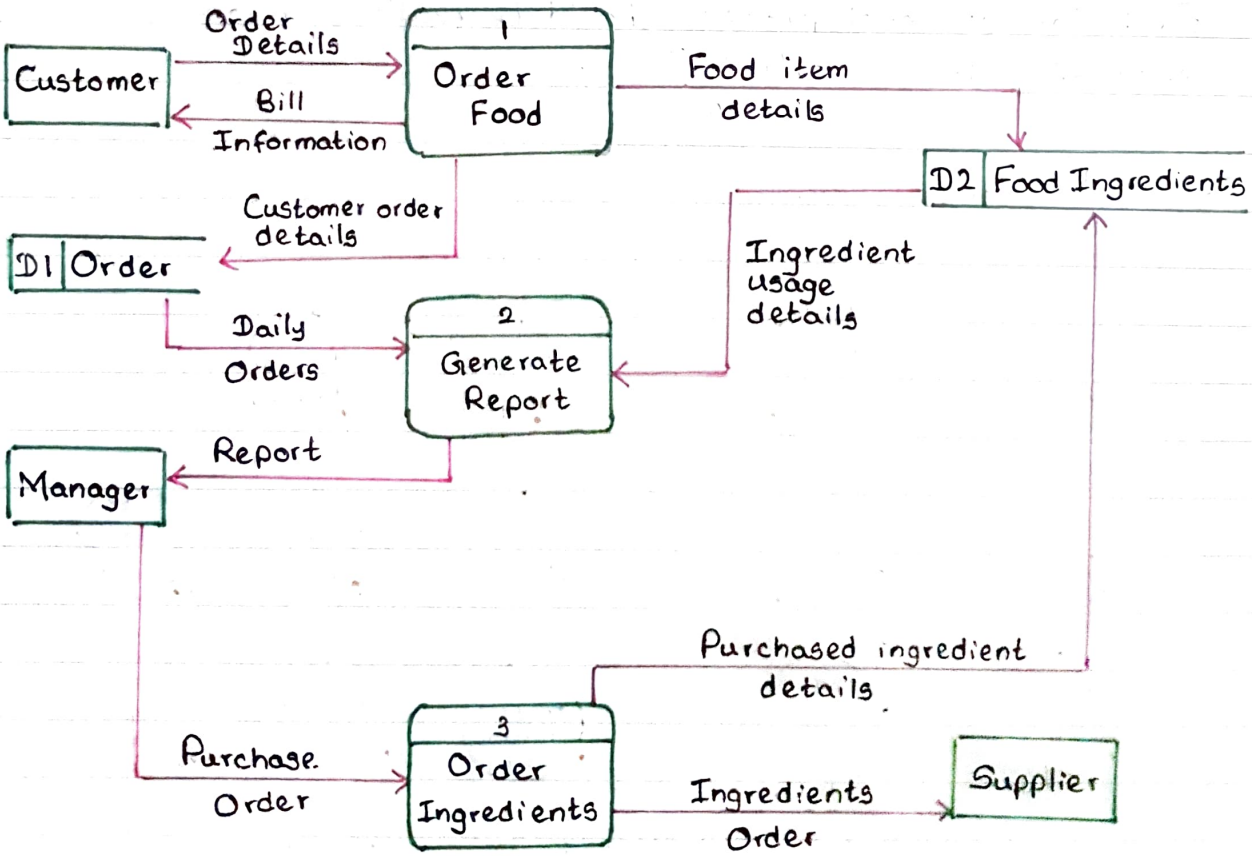


Activity

A customer can place an order to the system. The Order Food process receives the order, sends the food request to the kitchen, store customer order details in the Order data store, and store the food item details in the Food ingredients data store. The process also delivers a bill to the customer.

Manager can receive reports through the generate reports process which takes Ingredients usage details and daily orders as input from the Food Ingredients and Order data store respectively.

Manager can also initiate the Order Ingredients process by sending a purchase order. The process sends the ingredients order to the supplier and stores the purchased ingredient details in the Food ingredients data store.



Decomposition of Business Processes

Business processes are too complex to be explained in one DFD

Decomposition of business process into a hierarchy of DFDs, each representing a lower level of detail

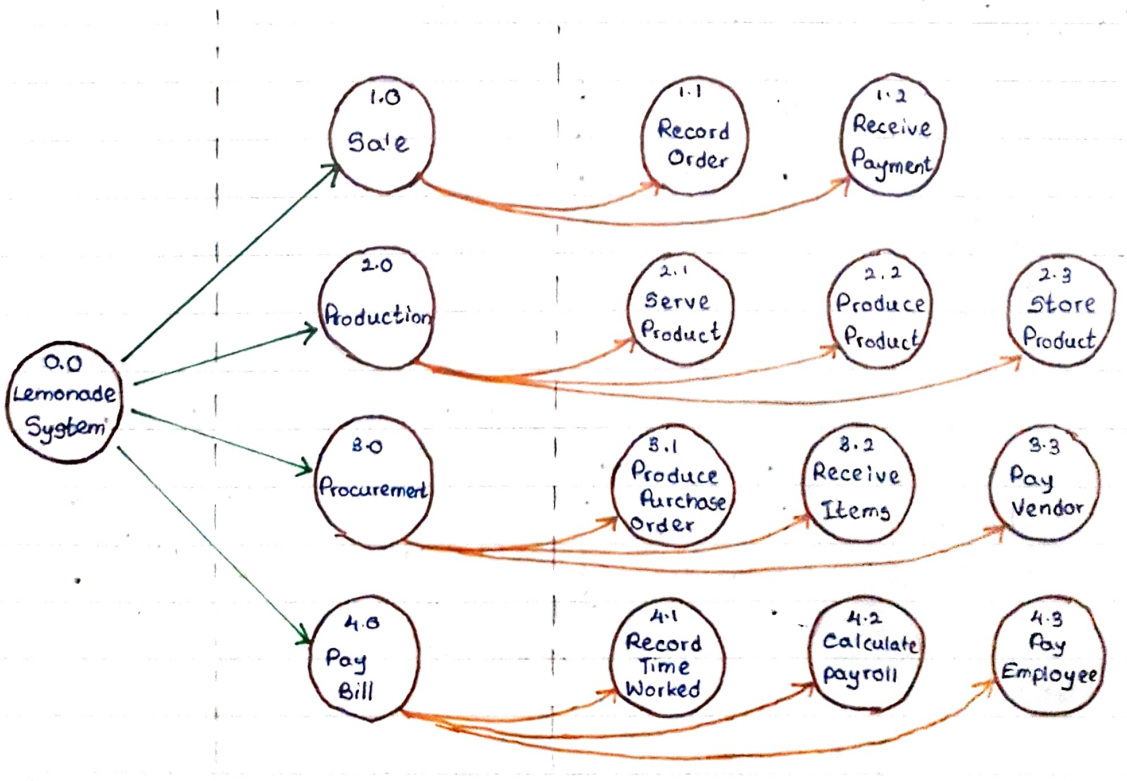
- Context Diagram
- Level 0 DFD
- Level 1 DFD
- Level 2 DFD
- .....

Rules of thumb

There should be at least 3 and no more than 7-9, processes on every DFD

Each lowest level process should be realized in about 25-50 lines of code.

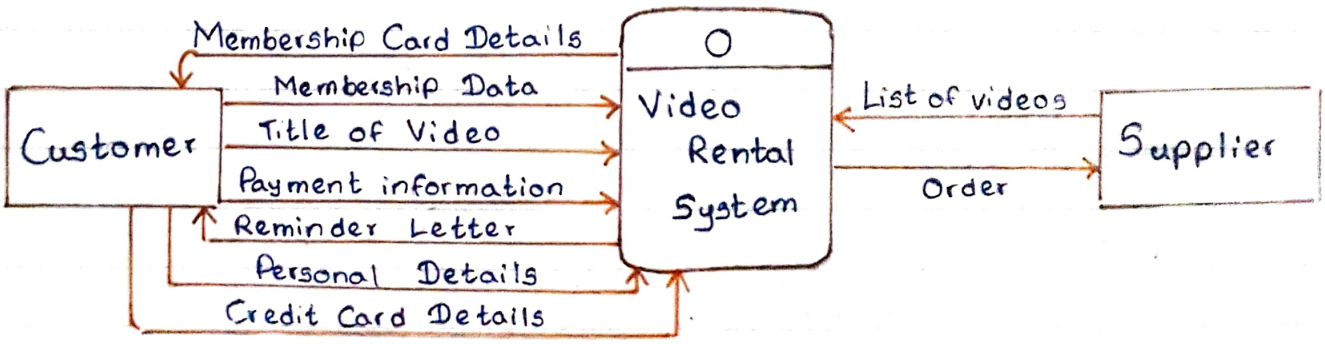
Process Decomposition



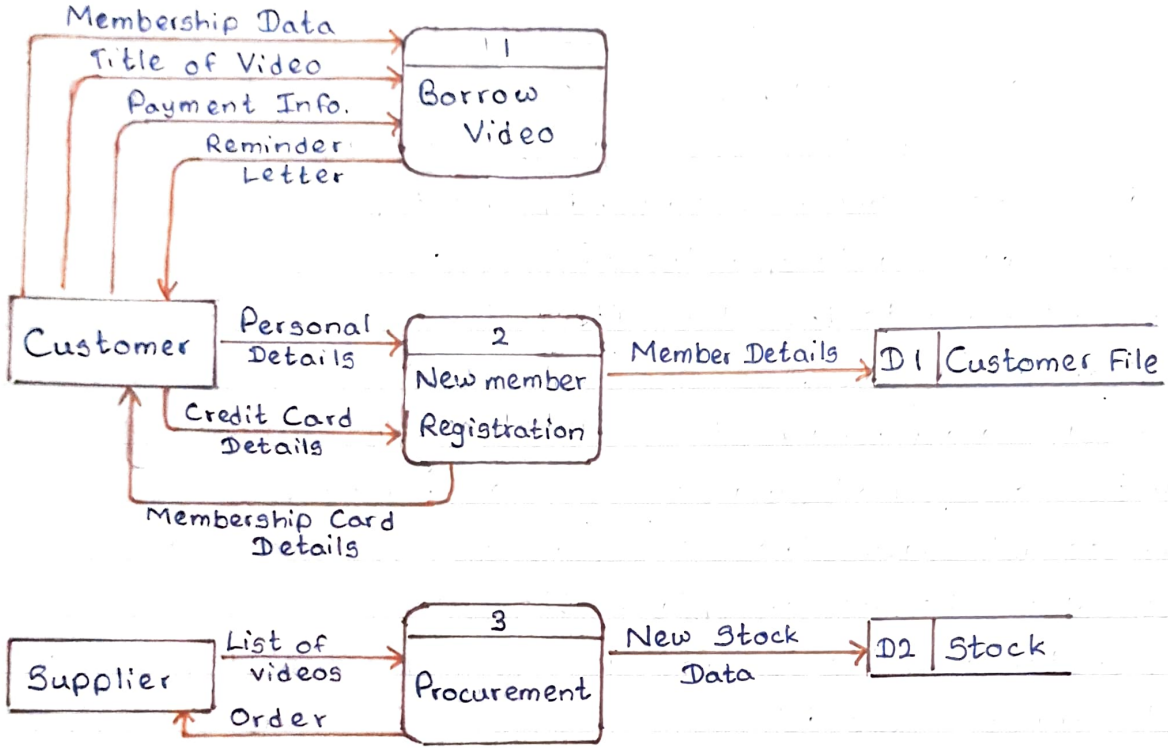
Activity

Video-Rental is a small video rental store. The store lends videos to customers for a fee, and purchases its videos from a local supplier
A customer wishing to borrow a video provides the title of the video they desire, their membership card, and payment - payment is always with the credit card used to open the customer account. The customer then returns the video to the store after watching it.
If a loaned video is overdue by a day the customer's credit card is charged, and a reminder letter is sent to them. Each day after that a further card is made, and each week a reminder letter is sent. This continues until either the customer returns the video, or the charges are equal to the cost of replacing the video
New customers fill out a form with their personal details and credit card details, and the counter staff gives the new customer a membership card. Each new customer's form is added to the customer file.
The local video supplier sends a list of available titles to Video Rental LTD, who decide whether to send them an order and payment. If an order is sent then the supplier sends the requested videos to the store. For each new video a new stock form is completed and placed in the stock file

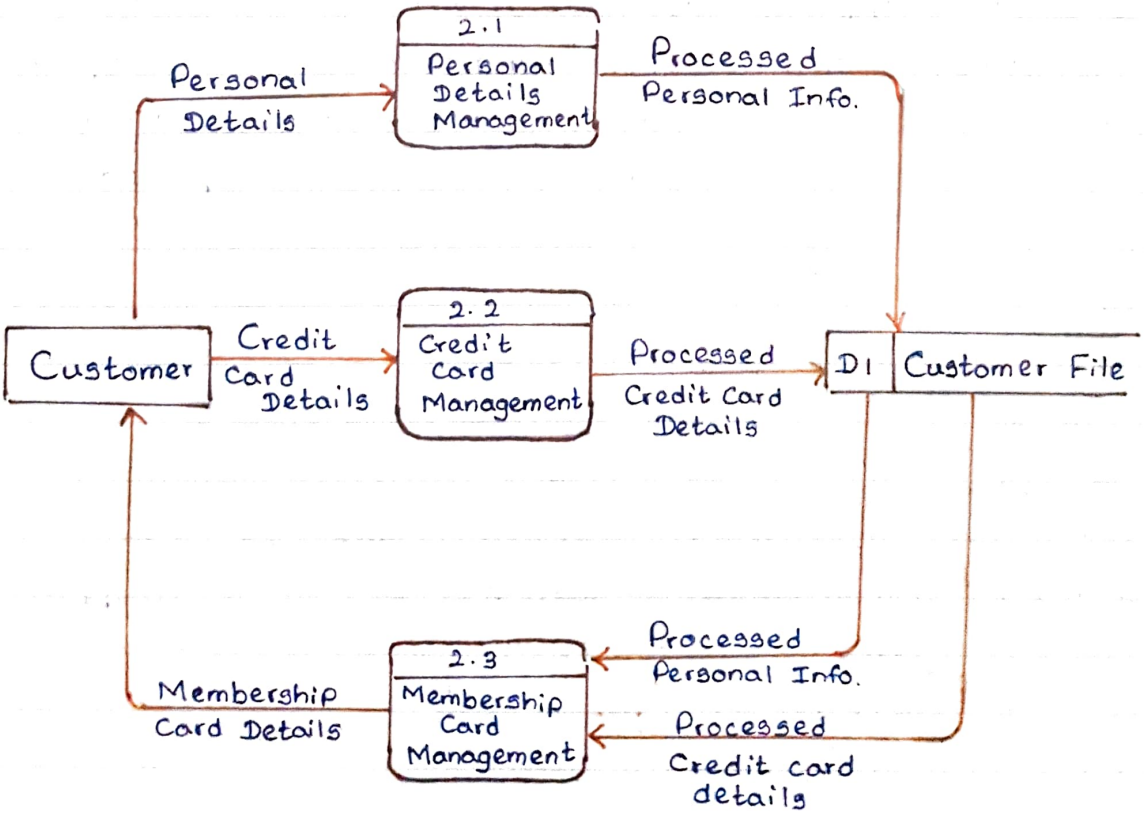
Context Diagram



Level 0 DFD



Level 1 DFD



Moving Into Design

Transition from Requirement to Design

The purpose of the analysis phase is to figure out what the business needs.

The purpose of the design phase is to decide how to build it

System Design is determination of the overall system architecture consisting of a set of physical processing components, hardware, software, people and the communication among them that will satisfy system's essential requirements.

In the initial part of design, the business requirements are converted into system requirements that describe the technical details for building the system.

Activities in the Design Phase

Determine preferred system acquisition strategy
Design the architecture for the system
Make hardware and software selections
Design system navigation, inputs and outputs
Convert logical process model to physical process model
Update CASE repository with additional system details
Design the programs that will perform the system processes
Convert logical data model to physical data model
Revise CRUD matrix
Design the way in which data will be stored
Compile final system specification

System Specification

At the end of the design phase, the final deliverable called system specification is created.

Reccomended System Aquisition Strategy
System Aquisition Weighted Alternative Matrix
Architecture Design

Hardware and Software Specification
Interface Design
Physical Process Model
Program Design Specification
Physical Data Model
Data Storage Model
Updated CRUD Matrix
Updated CASE Repository Entries

System Acquisition Strategies

There're three primary ways to approach the creation of a new system

- Develop a custom application in-house
- Buy a packaged system and (possibly) customize it
- Outsourcing

Custom Development

Building a new system from scratch

Pros

Allows developers to be flexible and creative in the way they solve business problems
Allows to take advantage of current technologies that can support strategic efforts
Builds technical skills and functional knowledge within the organization

Cons

Requires a dedicated effort that include long hours and hard work
Requires a variety of skills, but high skilled IS professionals are difficult to hire and retain
Risks associated with building a system from ground up can be quite high

Packaged Software

Software written for common business needs

Efficient to buy programs that have already been created and tested, and it can be bought and installed quickly compared with a custom system

Can range from small single function tools to huge all encompassing system such as ERP (enterprise resource planning) applications.

System Integration

Process of building new systems by combining packaged software, existing legacy systems, and new software written to integrate them.

Key challange is finding ways to integrate the data produced by the different packages and legacy systems

Outsourcing

Hiring an external vendor, developer, or service provider to create or supply the system

Application Service Providers (ASPs) supply software application or services over the internet

Low cost of entry and short setup time

Risk of

- Compromising confidential information
- Losing control over future development
- Losing important skills of in-house professionals

	When to Use Custom Development	When to Use a Packaged System	When to Use Outsourcing
Business Need	The business need is unique	The business need is common	The business need is not core to the business
In-house Experience	In house functional and technical experience exists	In house functional experience exists	In house functional or technical experience do not exist
Project Management	The project has a highly skilled project manager and a proven methodology	The project has a manager who can coordinate vendor's efforts	The project has a highly skilled project manager at the level of the organization that matches the scope of outsourcing deal
Project Skills	There is a desire to build in house skills	The skills are not strategic	The decision to outsource is a strategic decision
Time Frame	The time frame is flexible	The time frame is short	The time frame is short or flexible

Object Oriented Design

The object oriented approach views a system as a collection of self contained objects, including both data and processes

Unified Modeling Language (UML) is a standard set of diagramming techniques for object oriented systems analysis and design

Any object oriented approach must be

- Use case driven - use cases are the primary modeling tool
- Architecture centric - underlying architecture of the evolving system drives the specification, construction and documentation.
- Iterative and incremental - development undergoes continous testing throughout the life of the project.

Concepts like polymorphism, encapsulation and inheritance taken together allow analysts to break complex system into smaller, more managable components, to work on the components individually, and to more easily piece the components back together to form a system.

Unified Modeling Language Version 2.0

Objective of Unified Modeling Language (UML) is to provide common vocabulary of object oriented terms and diagramming techniques

Diagrams are broken into two major groupings

- Structure Diagrams represent data and static relationship
  - Class, Object, Package, Deployment, Component, Composite Structure
- Behavioural Diagrams represent dynamic relationships among the objects
  - Activity, Sequence, Communication, Interaction Overwiew Timing Behavioral State Machine, Protocol State Machine, Use Case

Four UML diagramming techniques dominate the object oriented projects

- Use Case diagrams
- Class diagrams
- Sequence diagrams
- Behaviour state machine diagrams

Use Case Diagrams

Describes how an external user triggers an event to which the system must respond

Summarizes all the use cases together in one diagram

Actor

Person or system that derives benifit from, and is external to the system

Labeled with its role

Can be associated with other actors by a specialization / superclass association denoted by an arrow with a hollow arrowhead

Placed outside system boundary.

Actor role name

Use Case

Represents a major piece of system functionality

Can extend another use case

Can use another use case

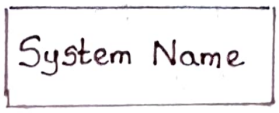
Placed inside system boundry

Labeled with descriptive verb-noun phrase

System Boundary

Includes the name of the system inside or on top

Represent scope of system

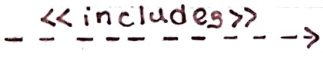


Association Relationship

Links an actor with the use case(s) with which it interacts

<<includes>> Relationship

Relationship between use cases in which one use case is stereotypica
-lly included within the other use case



<<extend>> Relationship

Specifies how and when the behaviour defined in usually supplimentary

Generalization Relationship

Abstract or concrete actors/use cases and specialized them

Brief Use Case Description

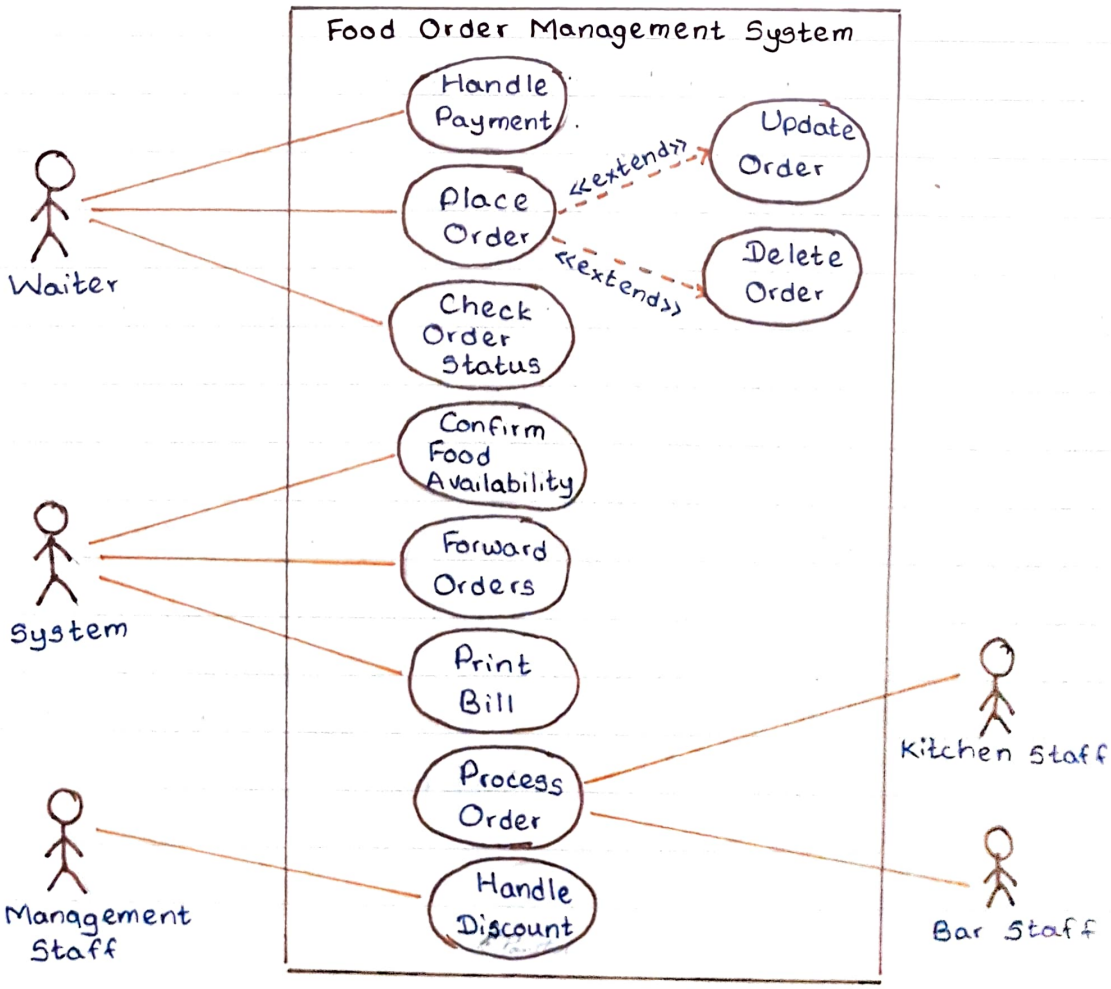
Often one sentence description that provides a quick overview of a use case

Eg Look up customer

User/actor enters customer account number, and the system retrieves and displays customer and account data.

Activity

After taking orders for a table the waiters place the orders into the system at the PC. The system must confirm availability of the dishes. Should an item not be available, waiter may change or delete a customer order. Dishes to be prepared are sent to the kitchen, drinks orders to the bar by the system. Starters and main course orders are usually taken together. Drink and dessert orders may be taken separately. Kitchen staff sees the dish orders on their screen, prepare them in an appropriate sequence and confirm preparation to the system when complete, similarly with the bar. When a waiter sees the completion indications on his terminal he collects the items and take them to the table. Waiter can also check the status of order. At the end of the meal the waiter will have the system print a bill and enter the details of payment for it. The management can give discounts.



DFD Fragments

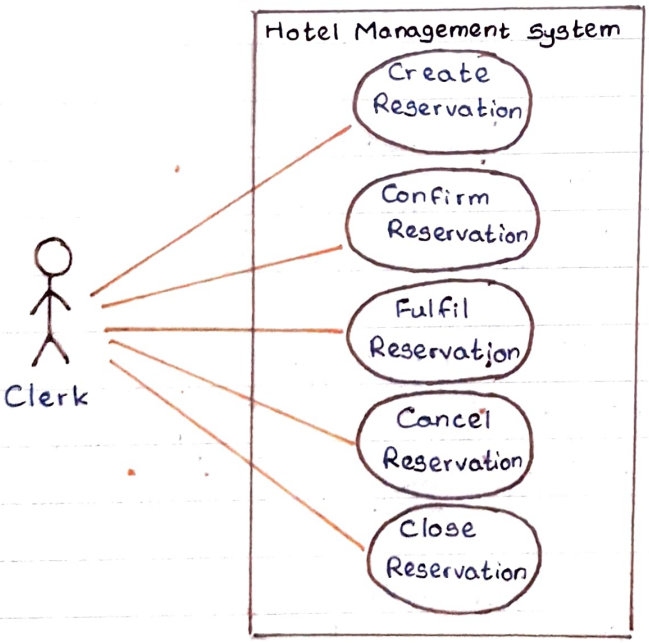
DFD fragment is one part of a DFD that eventually will be combined with other DFD fragments to form a DFD diagram

Each use case is converted into one DFD fragment

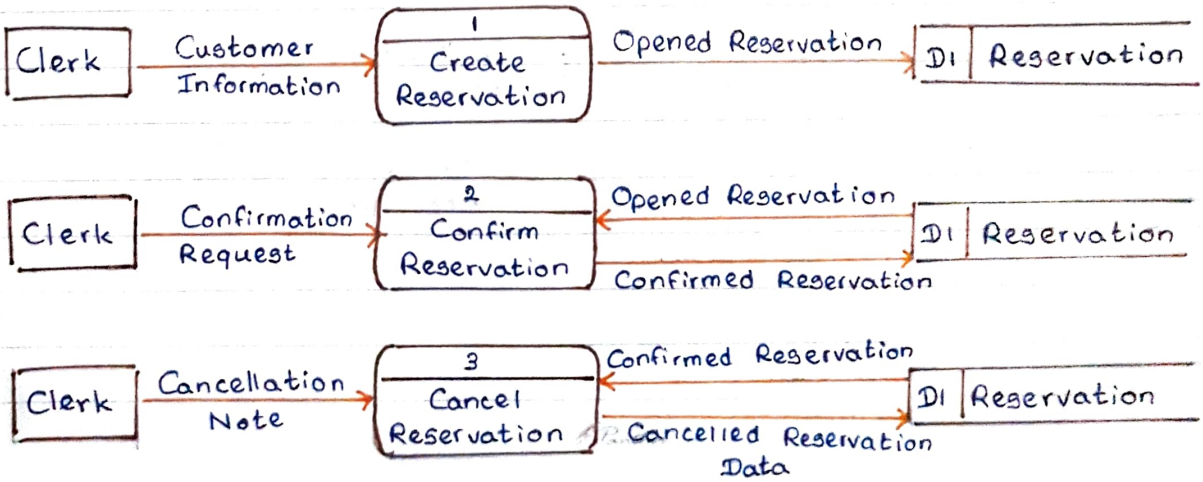
When a set of DFD fragments are connected together a 0 Level DFD is formed

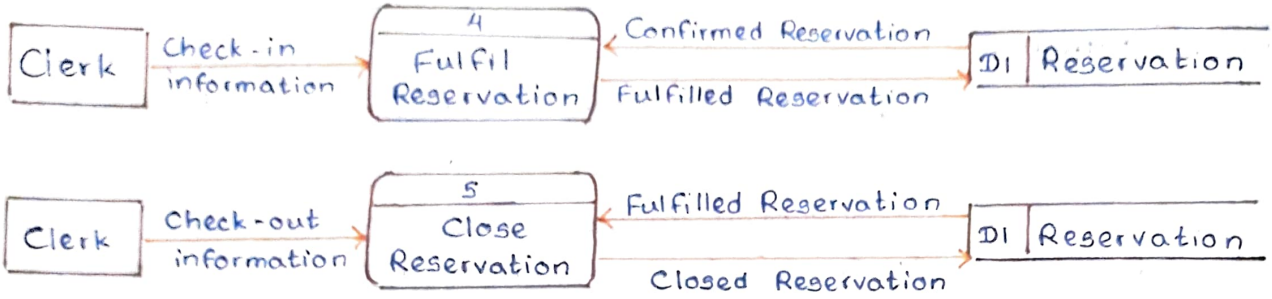
When creating 0 Level DFD from DFD fragments repeated external entities, data stores are discarded.

Use Case Diagram

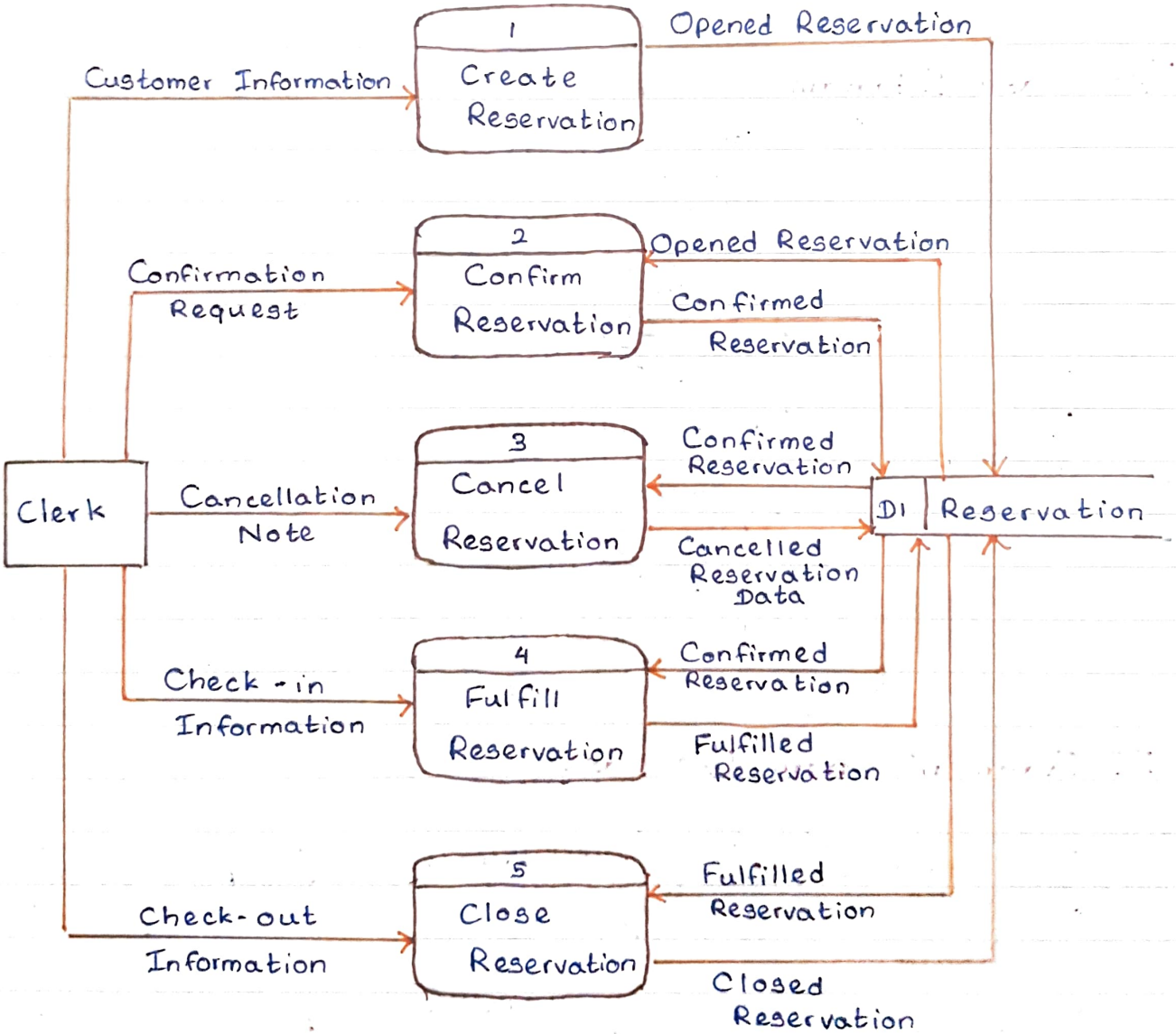


DFD Fragments from Use Case





Level 0 DFD from Fragments



Classes and Class Diagrams

Class

Category or classification of a set of objects or things

Domain Class

Classes that describe objects from the problem domain

Class Diagram

Diagram consisting of classes and association among the classes

Domain Model Class Diagram

Class diagram that only includes classes from the problem domain

Purpose of the Class Diagram

- Analysis and design of the static view of an application
- Describe responsibilities of a system
- Base for component and deployment diagrams
- Forward and reverse engineering

A Class

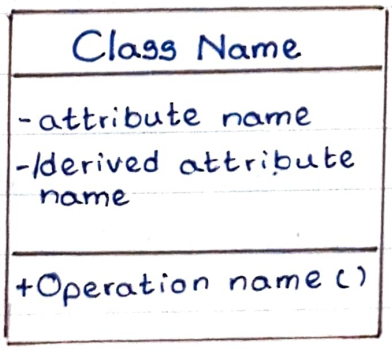
Represents a kind of person, place, or thing about which the system must capture and store information.

Has a name typed in bold and centered in its top compartment

Has a list of attributes in the middle compartment

Has a list of operations in its bottom compartment

Does not show expliticity show operations that are available to all classes



An Attribute

Represents properties that describe the state of an object
Can be derived from other attributes shown by placing a slash before the attribute's name.
Visibility can be public (+), protected (#), or private (-)
Attributes can be Simple Attributes, Identifier/Key Attribute, Composite Attribute, Derived Attribute.

- number
- artist
- price

A Method

Represents the actions or functions that a class can perform
Can be classified as a constructor, query or update operation

Constructor creates a new instance of a class

insert employee()
insert spouse()

Query makes information about the state of an object available to other objects

calculate employee salary()
place request for vacation()
calculate sick days()
increment number of employee vacation days()
find employee address()

Update change the value of some or all of the object's attributes

change employee name()
change employee address()

Includes paranthesis that may contain special parameters or information needed to perform the operation

An Association

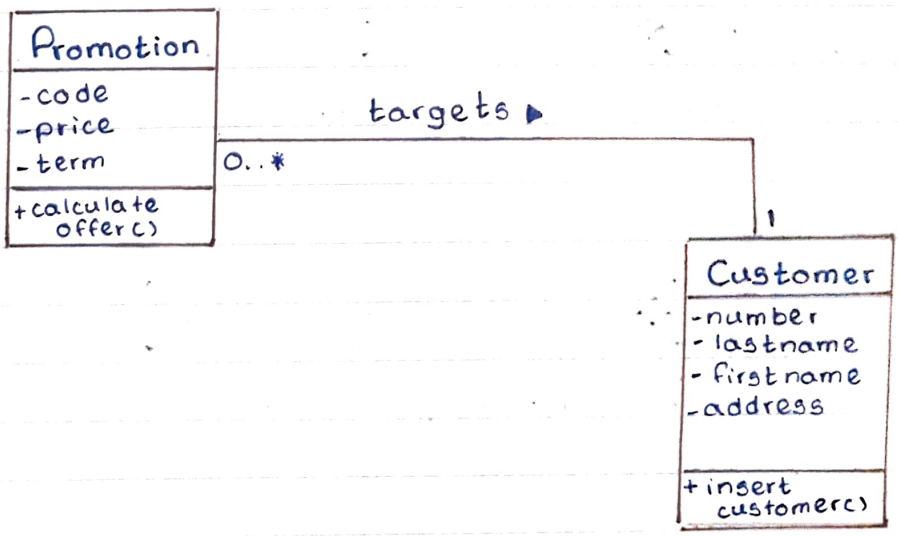
Represents a relationship between multiple classes or a class itself
Labled by a verb phrase or role name, whichever better represents the relationship.
Can exist between one or more classes
Contains multiplicity symbols which represent the minimum and maximum times a class instance can be associated with the related class instance

verb phrase

Multiplicity

Exactly one 1
Zero or more 0..*
One or more 1..*
Zero or one 0..1
Specified range 2..4 Two to four
Multiple, disjoint ranges 1..3,5 One to three & five

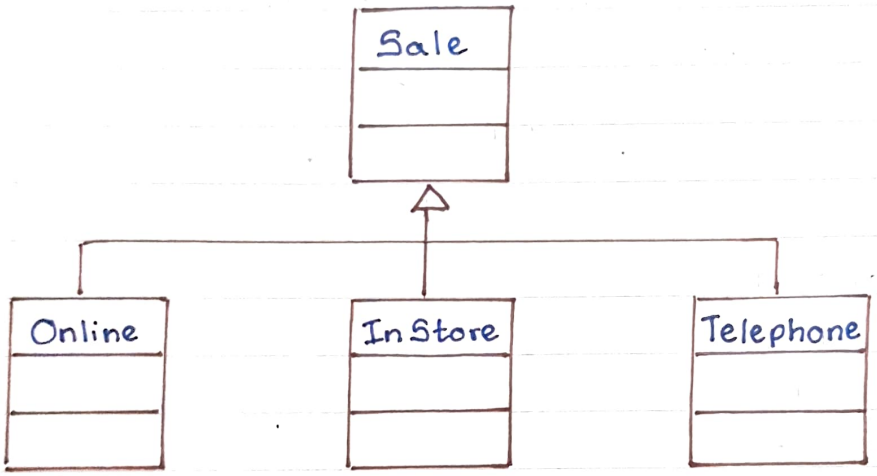
▶ shows the direction of association



Generalization / Specialization

Shows that one class (sub class) inherits from another class (super class), meaning properties and operations of superclass are also valid for objects of the sub class.

Generalization path is shown by a solid line from the sub class to superclass and hollow arrow pointing super class



Whole Part Relationships

Whole part relationships are when one class represents the whole object and other classes represent parts

Composition

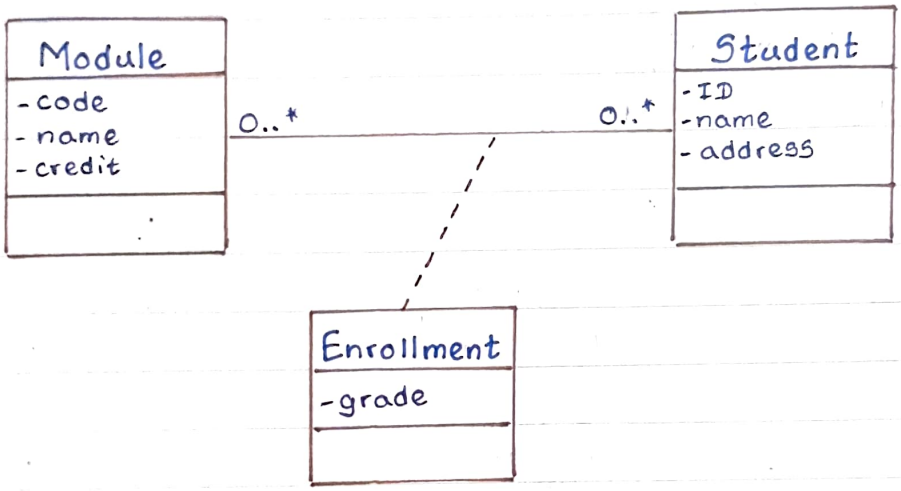
Every car has an engine

Aggregation

Cars may have passangers

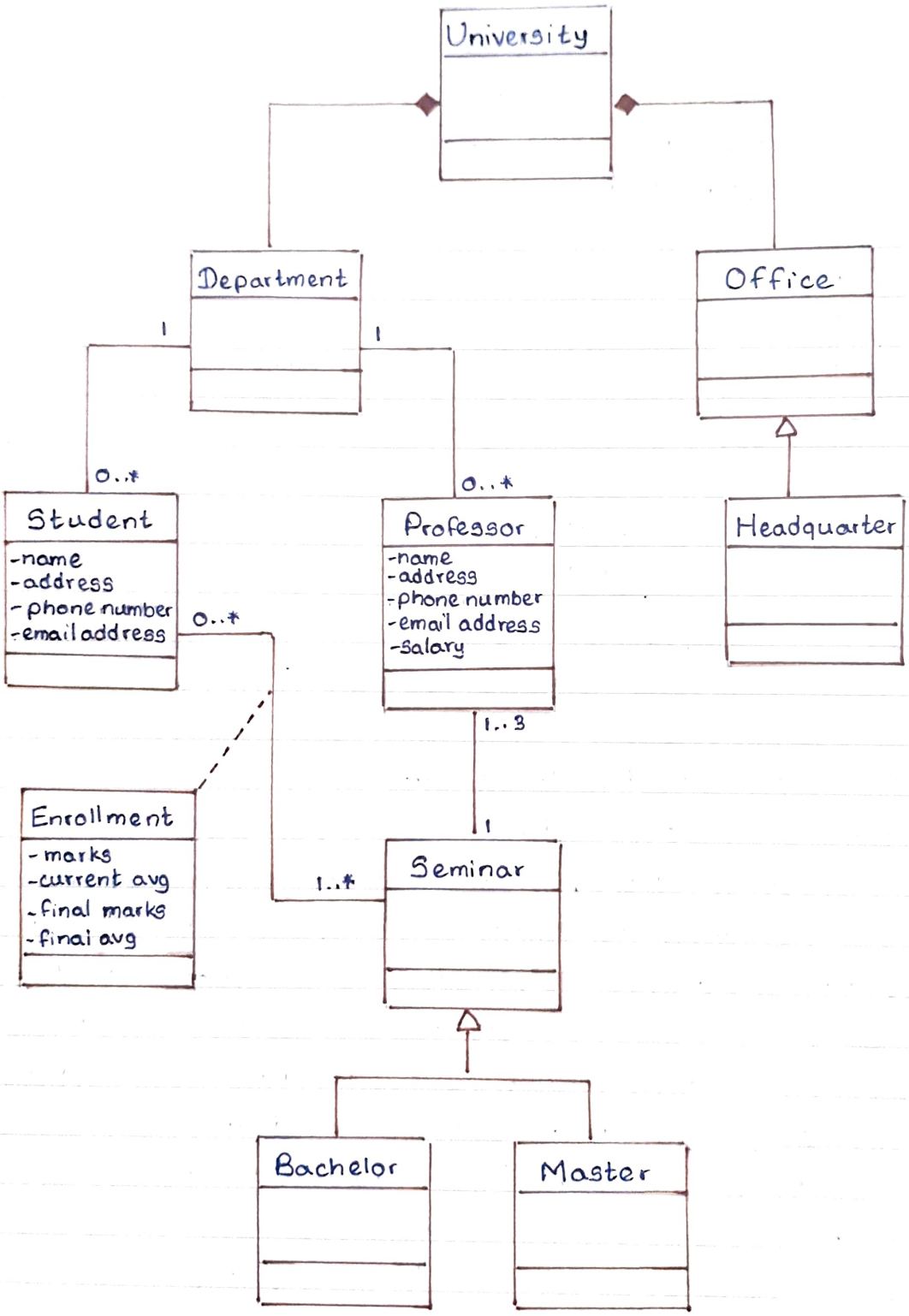
Association Class

An association that is treated as a class in a many to many association because it has attributes that need to be remembered The attributes depend on both the classes which the association is connected to.



Activity

A University consists of departments and offices. One office acts as a headquarter of all offices. Professors and students are allocated to departments. One student/professor can have only one department A professor has a name, address, phone number, email address, and salary. A student has also a name, etc., but no salary. A student however has an avarage mark (of final marks of his/her seminars) A seminar has a name and a number. When a student is enrolled in a seminar, the marks of this enrollment are recorded and current avarage as well as the final mark (if there is one) can be obtained from the enrollment. One professor teaches one seminar. Each seminar has atleast one at most three teachers. There are two types of seminars; bachelor and master.


University
Department
Office
1
1
0..*
0..*
Student
-name
-address
-phone number
-email address
Professor
-name
-address
-phone number
-email address
-salary
Headquarter
0..*
1..3
1
Enrollment
- marks
-current avg
-final marks
-final avg
1..*
Seminar
Bachelor
Master

Sequence Diagram

Illustrates the objects that participate in a use case and the messages that pass between them over time for one use case

It's a dynamic diagram that supports a dynamic view of the evolving systems.

There are two types of sequence diagrams.

- Generic Sequence Diagram
  - Show all possible scenarios for a use case
- Instance Sequence Diagram
  - Decipts a single scenario within the use case

Purposes

To capture the dynamic behaviour of a system

To describe the message flow in the system

To describe structural organization of the objects

To describe the interaction among objects.

An Object

Participates in a sequence by sending and/or receiving messages

Placed across the top of the diagram

A Lifeline

Denotes the life of an object during a sequence,

Contains an x at the point at which the class no longer interacts

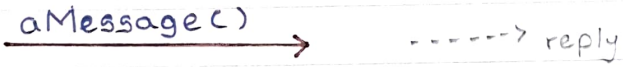
A focus of Control

Long narrow rectangle placed atop lifeline

Denotes when an object is sending or receiving messages

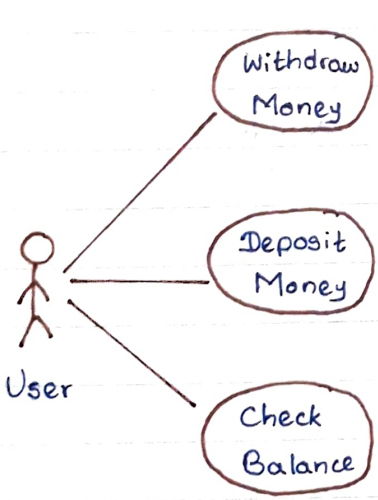
A Message

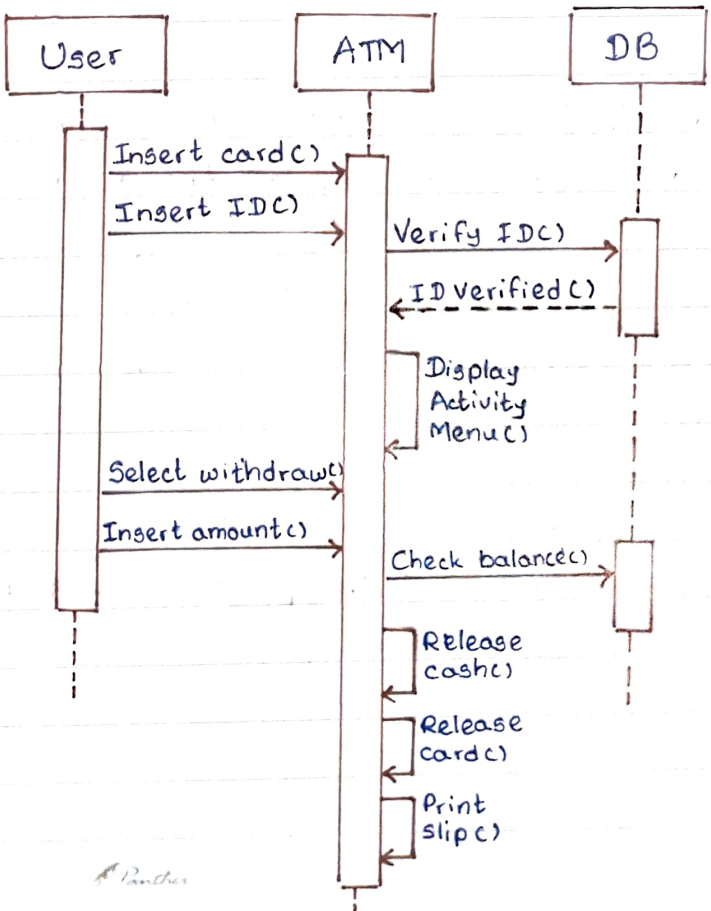
Conveys information from an object to another one



Object Destruction

An X placed at the end of an object's lifeline to show that is going out of existance





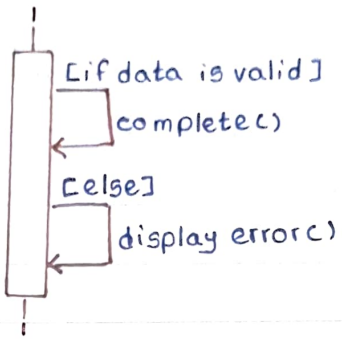
Alternative Combination

If data is valid

complete

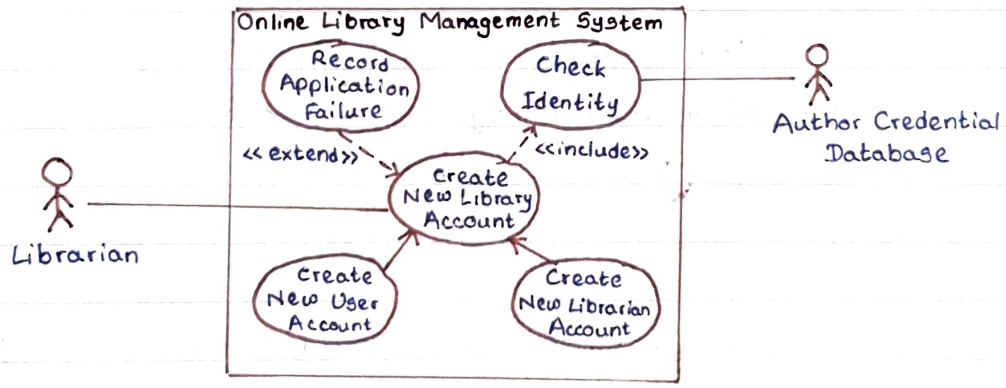
Else

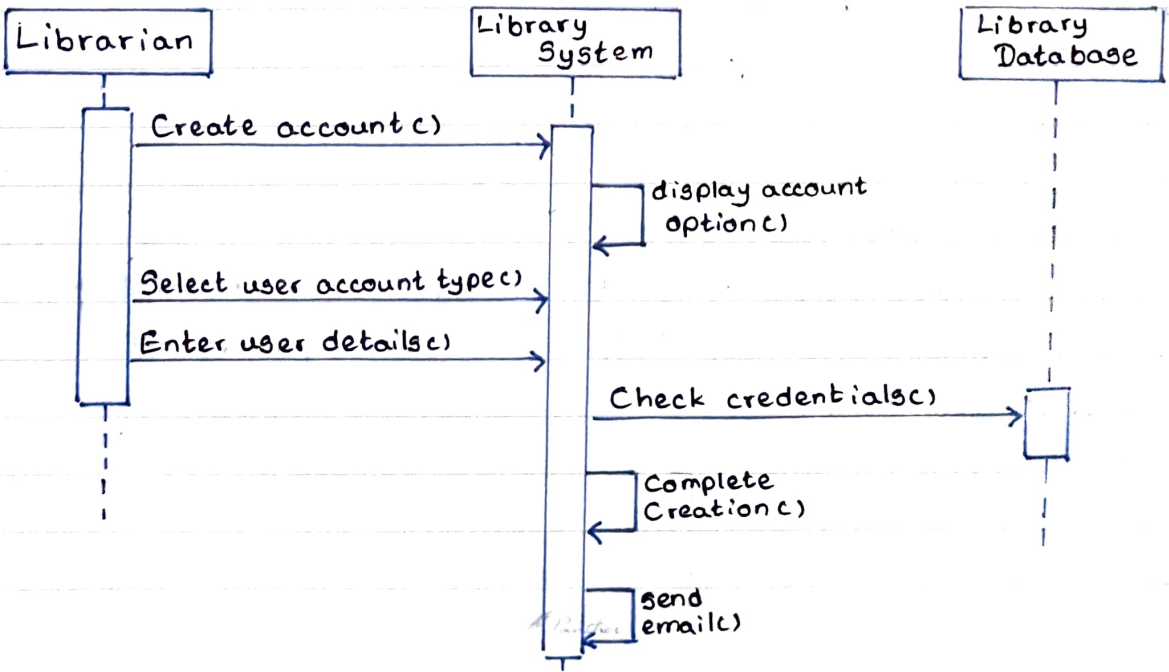
display error



Activity

Librarian request the system to create a new online library account. The librarian then selects the library user account type. Librarian enter user's details. The user's credentials are checked by the system, then the new user account creation is completed and a summary of the new user account is emailed to the user





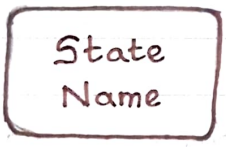
State Transition Diagram

Behavioral State Transition Diagram is a dynamic model that shows the different states that a single class passes through during its life in response to events, along with its responses and actions

A State

Shown as a rectangle with rounded corners

Has a name that represents the state of an object



An Initial State

Shown as a small filled-in circle

Represents the point at which an object begins to exist

A Final State

Shown as a circle surrounding a small solid filled-in circle (bull's eye)

Represents the completion of activity

An Event

Noteworthy occurance that triggers a change in state

Can be a designated condition becoming true, the receipt of an explicit signal from one object to another, or the passage of a designated period

Used to label a transition

Event Name

A Transition

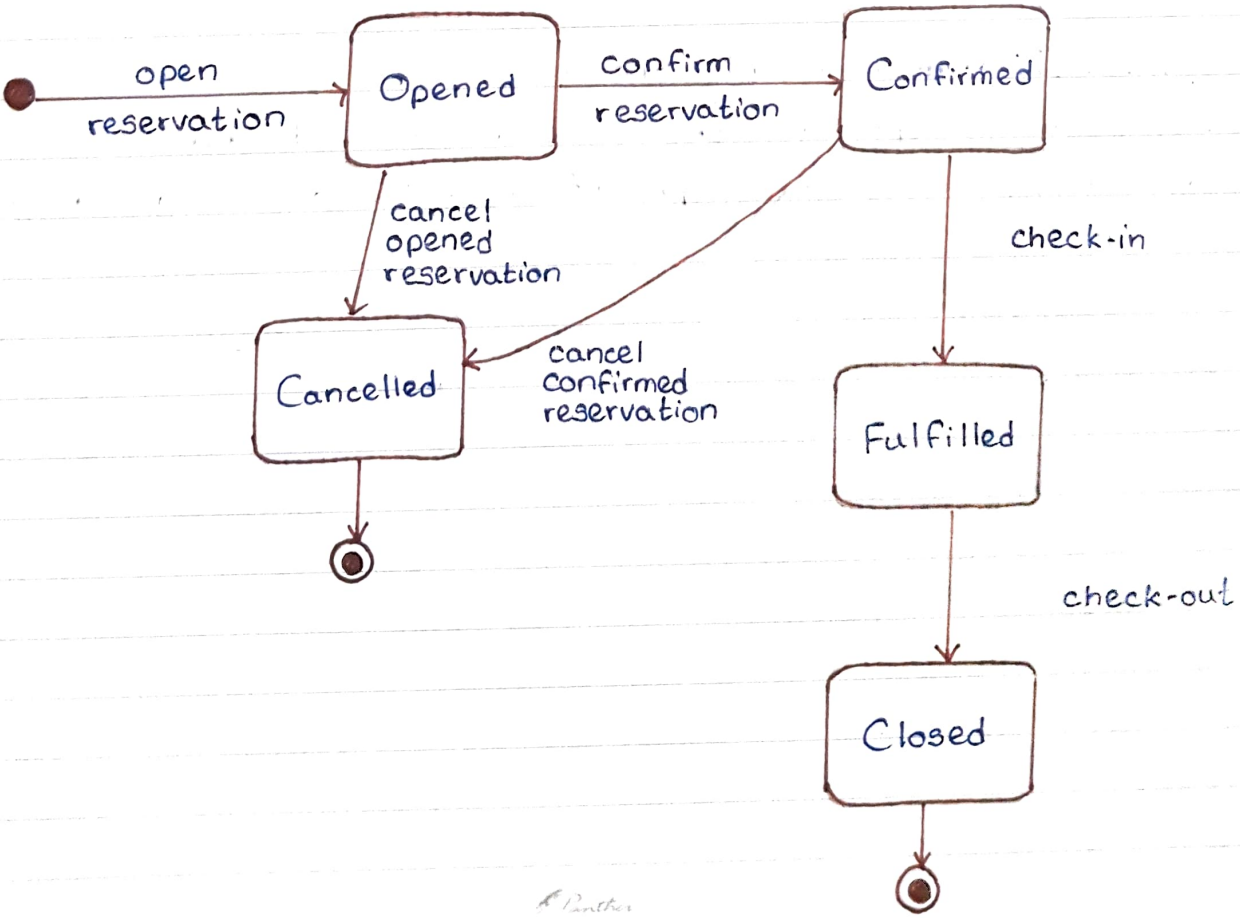
Indicates that an object in the first state will enter the second state

Triggered by the occurence of the event labeling the transition

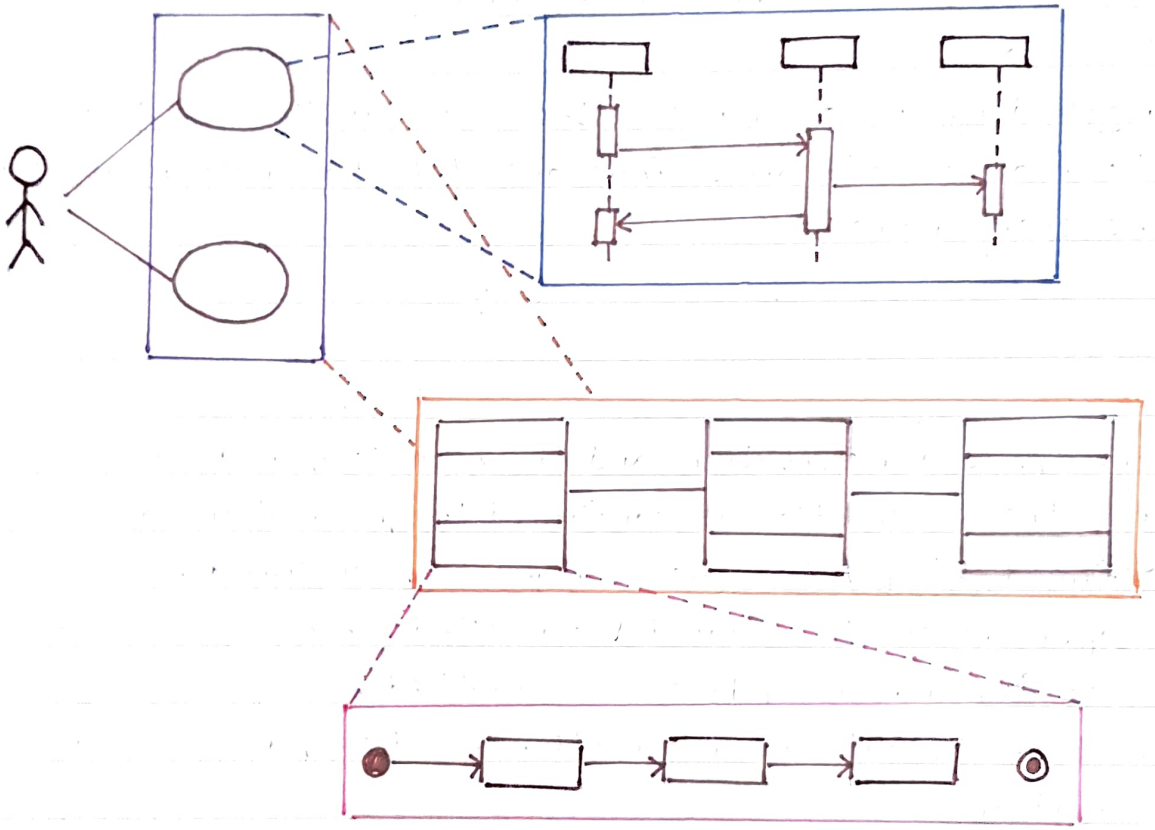
Shown as a solid arrow from one state to another, labled by the event name

Activity

Clerk opens a reservation and the room is opened for reservation. When the reservation is confirmed the reservation converts to confirmed. As per the guests' request a confirmed reservation or an open reservation can be cancelled. When the guest checks-in the confirmed room updates as a fulfilled reservation. When the guest checks-out, the reservation will be updated as a closed reservation



Mapping All Diagrams



Sequence Diagram represents one use case

Class Diagram represents the whole use case diagram

State Transition Diagram represents a single class in class diagram