Laboratory Practical Report

of

SYSTEM ANALYSIS AND DESIGN

(ICT ED 447)

Submitted To

TRIBHUVAN UNIVERSITY

In Partial Fulfillment of the Requirements of the course

B.Ed. ICTE 4th Semester

Submitted By

Sanam Tamang

Symbol No.: 76214020

T.U. Regd. No.: 9-2-214-54-2019

Under the guidance of

Mr. Atul Bhattarai

Lecturer

Sukuna Multiple Campus

SUKUNA MULTIPLE CAMPUS

Sundarharaincha-12, Morang, Nepal

2079

CERTIFICATE

This is to certify that the Laboratory Practical Report

of

SYSTEM ANALYSIS AND DESIGN

(ICT ED 447)

In Partial Fulfillment of the Requirements of the course

B.Ed. ICTE 4th Semester

Submitted By

Sanam Tamang

Symbol No.: 76214020

T.U. Regd. No.: 9-2-214-54-2019

is a bonafide record of experiments carried out by him/her under by guidance.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mr. Atul Bhattarai

Lecturer

Sukuna Multiple Campus

Sundarharaincha, Morang

(Internal Examiner)

Submitted for the Final Examination on: 2079/08/09

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mr. Rajesh Kumar Chaudhary

Lecturer

Triyuga Janata Multiple Campus

Triyuga-3, Udayapur

(External Examiner)

**Contents**

[1. Discuss the use of CASE Tools. 1](#_Toc120112373)

[Components of CASE Tools 1](#_Toc120112374)

[ Central Repository 1](#_Toc120112375)

[ Upper Case Tools 1](#_Toc120112376)

[ Lower Case Tools 1](#_Toc120112377)

[ Integrated Case Tools 1](#_Toc120112378)

[2. Use Case Diagram 2](#_Toc120112379)

[a. Actors 2](#_Toc120112380)

[b. Use cases 2](#_Toc120112381)

[c. Communication links 2](#_Toc120112382)

[ Associations 2](#_Toc120112383)

[ Generalizations 2](#_Toc120112384)

[ Extensions 2](#_Toc120112385)

[ System boundary 3](#_Toc120112386)

[3. ER Diagram 4](#_Toc120112387)

[a. Entities 4](#_Toc120112388)

[b. Relationships 4](#_Toc120112389)

[c. Attributes 5](#_Toc120112390)

[4. Data Flow Diagram (DFD) 6](#_Toc120112391)

[ Process 6](#_Toc120112392)

[ Data Flow 6](#_Toc120112393)

[ Warehouse 6](#_Toc120112394)

[ Terminator 7](#_Toc120112395)

[5. UML Diagram 8](#_Toc120112396)

# 1. Discuss the use of CASE Tools.

Case stand for computer aided software engineering. It means development and maintenance of software projects with the help of various automated software tools. Case tools are set of software application programs which are used to automatic SDLC activities to develop software system. CASE stands for Computer Aided Software Engineering. It means, development and maintenance of software projects with help of various automated software tools.

CASE tools are set of software application programs, which are used to automate SDLC activities. CASE tools are used by software project managers, analysts and engineers to develop software system.

There are number of CASE tools available to simplify various stages of Software Development Life Cycle such as Analysis tools, Design tools, Project management tools, Database Management tools, Documentation tools are to name a few.

Use of CASE tools accelerates the development of project to produce desired result and helps to uncover flaws before moving ahead with next stage in software development.

Components of CASE Tools

CASE tools can be broadly divided into the following parts based on their use at a particular SDLC stage:

* Central Repository - CASE tools require a central repository, which can serve as a source of common, integrated and consistent information. Central repository is a central place of storage where product specifications, requirement documents, related reports and diagrams, other useful information regarding management is stored. Central repository also serves as data dictionary.

* Upper Case Tools - Upper CASE tools are used in planning, analysis and design stages of SDLC.
* Lower Case Tools - Lower CASE tools are used in implementation, testing and maintenance.
* Integrated Case Tools - Integrated CASE tools are helpful in all the stages of SDLC, from Requirement gathering to Testing and documentation.

# 2. Use Case Diagram

A use case diagram is a visual summarization of interactions and relationships within a system. These diagrams show a very broad view of a system. They may show systems in computer software, businesses or customer experiences. A use case diagram shows a model scenario in which individuals interact with a system using a series of specialized symbols and connectors.

A case use diagram includes a few key components:

## Actors

Actors are individuals who have a relationship within the system. They interact with each step of the process. This may include users, customers, clients or employees Actors may interact with the system internally or externally.

## Use cases

Use cases, often represented by an oval or circle encompassing text, describe a system function that is an actor or person can start or use. This function may be automatic or manual and may link to an actor or function independently.

## Communication links

A solid, straight line can represent an actor's interaction with a use case or use cases' interactions. Actors may connect to use cases and other people through a variety of communication links representing different relationships such as:

* Associations

Association links represent interactions between actors and use cases

### Generalizations

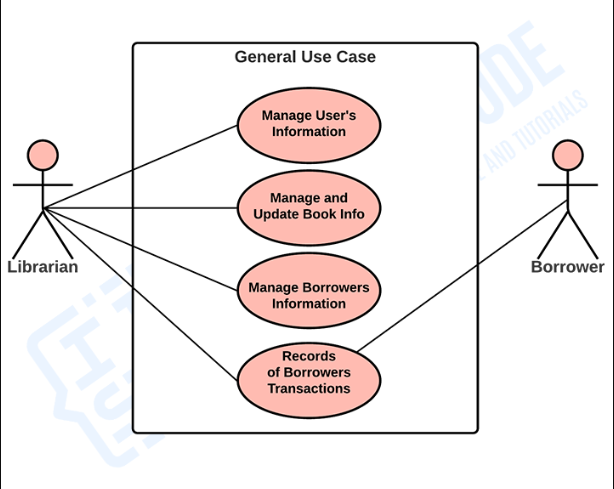
Generalization links represent relationships between actors that depend on each other's functions to complete their role within a system.

### Extensions

Extension links represent optional functions within a system. These links may connect actors to use cases or multiple related use cases without depending on one another for functionality.

### System boundary

The system boundary is a visual limitation of the entire system module. For complex systems, there may be many system boundaries that represent smaller, more specific modules. For example, a business use case diagram may have system boundaries for smaller components, such as marketing, sales and accounting.



# 3. ER Diagram

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs. By defining the entities, their attributes, and showing the relationships between them, an ER diagram can illustrate the logical structure of databases. This is useful for engineers hoping to either document a database as it exists or sketch out a design of a new database. An ER diagram has three main components: entities, relationships, and attributes connected by lines.

1. Entities, which are represented by rectangles. An entity is an object or concept about which you want to store information.



* A weak entity is an entity that must defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone

.

1. Relationships, which are represented by diamond shapes, show how two entities share information in the database. Relationship can be described as the connection between two or more entities.



* In some cases, entities can be self-linked.



1. Attributes, which are represented by ovals. A key attribute is the unique, distinguishing characteristic of the entity. For example, an employee's social security number might be the employee's key attribute.  
   

* A multivalued attribute can have more than one value. For



* example, an employee entity can have multiple skill values. A derived attribute is based on another attribute. For example, an employee's monthly salary is based on the employee's annual salary.



* **Connecting lines**, solid lines that connect attributes and show the relationships of entities in the diagram.
* **Cardinality** specifies the numerical attribute of the relationship between entities. It can be one-to-one, many-to-one, or many-to-many.

# 4. Data Flow Diagram (DFD)

Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.

Data flow diagrams can be divided into logical and physical. The logical

data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow. **DFD** is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and

outputs of each entity and the process itself.

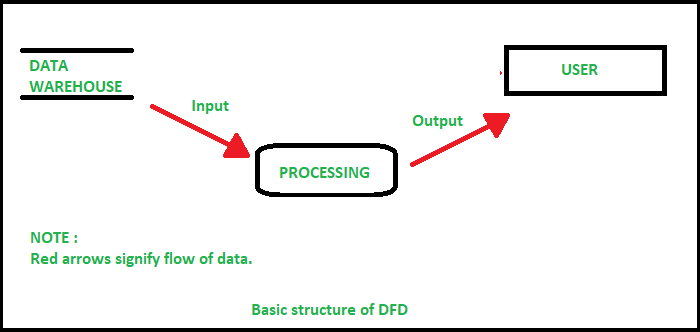
DFD does not have control flow and no loops or decision rules are present. Specific operations

depending on the type of data can be explained by a flowchart. Data Flow Diagram can be represented in several ways. The DFD belongs to structured-analysis modeling tools. Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes. The Data Flow Diagram has 4 components:

* Process  
  Input to output transformation in a system takes place because of process function. The symbols of a process are rectangular with rounded corners, oval, rectangle or a circle. The process is named a short sentence, in one word or a phrase to express its essence
* Data Flow  
  Data flow describes the information transferring between different parts of the systems. The arrow symbol is the symbol of data flow. A relatable name should be given to the flow to determine the information which is being moved. Data flow also represents material along with information that is being moved. Material shifts are modeled in systems that are not merely informative. A given flow should only transfer a single type of information. The direction of flow is represented by the arrow which can also be bi-directional.
* Warehouse  
  The data is stored in the warehouse for later use. Two horizontal lines represent the symbol of the store. The warehouse is simply not restricted to being a data file rather it can be anything like a folder with documents, an optical disc, a filing cabinet. The data warehouse can be viewed independent of its implementation. When the data flow from the warehouse it is considered as data reading and when data flows to the warehouse it is called data entry or data updating.
* Terminator  
  The Terminator is an external entity that stands outside of the system and communicates with the system. It can be, for example, organizations like banks, groups of people like customers or different departments of the same organization, which is not a part of the model system and is an external entity. Modeled systems also communicate

with terminator. Levels of DFD It uses hierarchy to maintain transparency thus multilevel DFD’s can be created. Levels of DFD are as follows:

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



# 5. UML Diagram

The **Unified Modeling Language (UML)**is a language used in the field of software engineering that represent the components of the Object-Oriented Programming concepts. It is the general way to define the whole software architecture or structure.

In Object-Oriented Programming, we solve and interact with complex algorithms by considering themselves as objects or entities. These objects can be anything. It can be the bank or a bank manager too. The object can be a vehicle, animal, machine, etc. The thing is how we interact and manipulate them that they can perform tasks and they should.

The tasks can be interacting with other objects, transferring data from one object to another, manipulating other objects, etc. The single software could have hundreds or even thousands of objects. So, **UML** provides us a way to represent and track those objects in a diagram to become a blueprint of our software architecture.

