COURSE TITLE: **SYSTEM ANALYSIS AND DESIGN**

PAPER CODE: **CSE 4205**

LECTURER: **MASESE CHUMA**

TOPIC: **STRUCTURED ANALYSIS**

CHAPTER: **FOUR**

STRUCTURED ANALYSIS

Analysts use various tools to understand and describe the information system. One of the ways is using structured analysis.

What is Structured Analysis?

Structured Analysis is a development method that allows the analyst to understand the system and its activities in a logical way.

It is a systematic approach, which uses graphical tools that analyze and refine the objectives of an existing system and develop a new system specification which can be easily understandable by user.

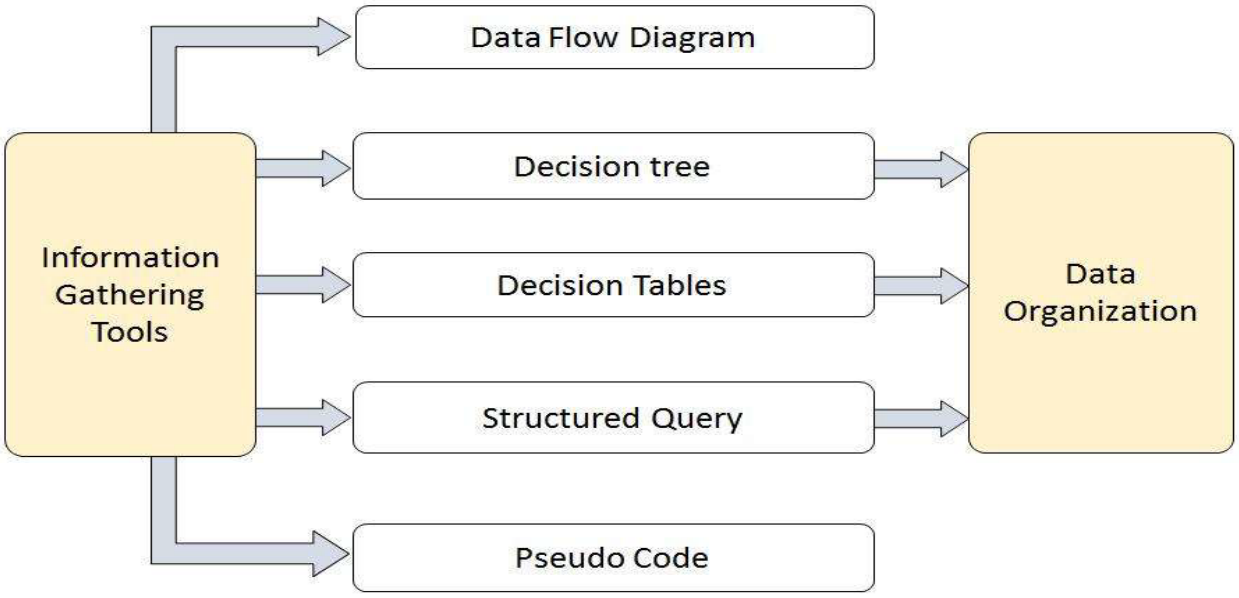
It has following attributes:

- It is graphic which specifies the presentation of application.
- It divides the processes so that it gives a clear picture of system flow.
- It is logical rather than physical i.e., the elements of system do not depend on vendor or hardware.
- It is an approach that works from high-level overviews to lower-level details.

Structured Analysis Tools

During Structured Analysis, various tools and techniques are used for system development. They are:

1. Data Flow Diagrams
2. Data Dictionary
3. Decision Trees
4. Decision Tables
5. Structured English
6. Pseudocode



Data Flow Diagrams (DFD) or Bubble Chart

It is a technique developed by Larry Constantine to express the requirements of system in a graphical form.

- It shows the flow of data between various functions of system and specifies how the current system is implemented.
- It is an initial stage of design phase that functionally divides the requirement specifications down to the lowest level of detail.
- Its graphical nature makes it a good communication tool between user and analyst or analyst and system designer.
- It gives an overview of what data a system processes, what transformations are performed, what data are stored, what results are produced and where they flow.

Basic Elements of DFD

DFD is easy to understand and quite effective when the required design is not clear and the user wants a notational language for communication. However, it requires a large number of iterations for obtaining the most accurate and complete solution.

The following table shows the symbols used in designing a DFD and their significance:

Symbol Name	**Meaning**
Square	Source or Destination of Data
Arrow	Data flow
Circle	Process transforming data flow
Open Rectangle	Data Store

Types of DFD

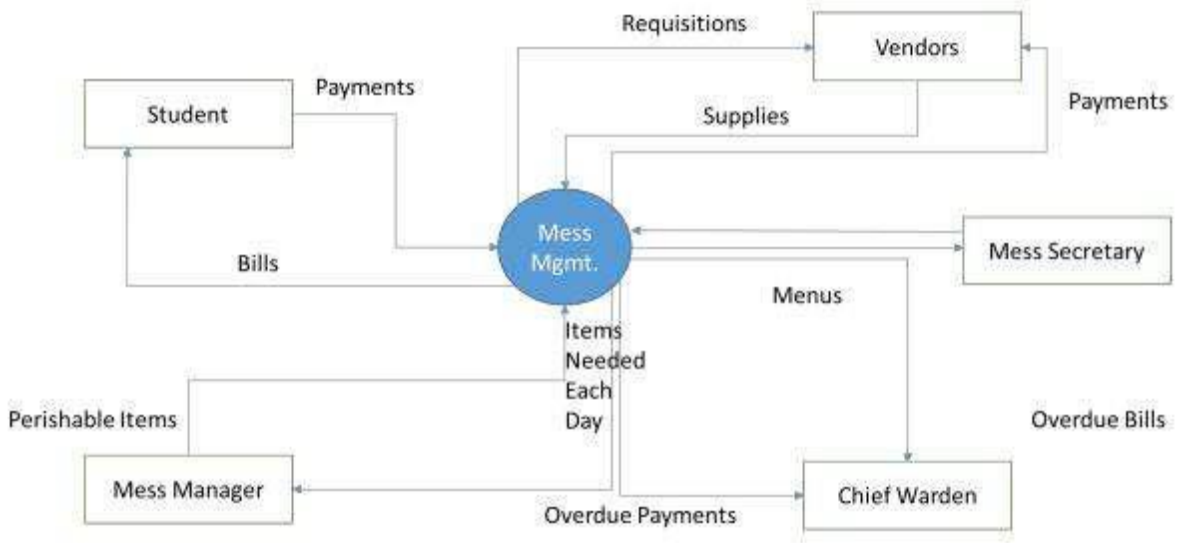
DFDs are of two types: Physical DFD and Logical DFD. The following table lists the points that differentiate a physical DFD from a logical DFD.

Physical DFD	**Logical DFD**
It is implementation dependent. It Shows which functions are performed.	It is implementation independent. It Focuses only on the flow of data between processes.
It provides low level details of hardware, software, files, and people.	It explains events of systems and Data required by each event.
It depicts how the current system operates and how a system will be implemented.	It shows how business operates; not how the system can be implemented.

Context Diagram

A context diagram helps in understanding the entire system by one DFD which gives the overview of a system. It starts with mentioning major processes with little details and then goes onto giving more details of the processes with the top-down approach.

The context diagram of mess management is shown below.



Data Dictionary

- A data dictionary is **a collection of data about data**. It maintains information about the definition, structure, and use of each data element that an organization uses.
- A data dictionary is a collection of descriptions of the data objects or items in a data model for the benefit of programmers and others who need to refer to them.
- **Data Dictionary** can be defined as collection of information of all data elements or contents of databases such as data types, text descriptions of system. It makes it easier for user and analyst to use data as well as understand and have common knowledge about inputs, outputs, components of a database, and intermediate calculations.
- A data dictionary contains metadata i.e data about the database. The data dictionary is very important as it contains information such as what is in the database, who is allowed to access it, where is the database physically stored etc.

The users of the database normally don't interact with the data dictionary, it is only handled by the database administrators

- A data dictionary is a structured repository of data elements in the system. It stores the descriptions of all DFD data elements that is, details and definitions of data flows, data stores, data stored in data stores, and the processes.
- A data dictionary improves the communication between the analyst and the user. It plays an important role in building a database. Most DBMSs have a data dictionary as a standard feature. For example, refer the following table:

Sr. No.	Data Name	Description	No. of Characters
1	ISBN	ISBN Number	10
2	TITLE	Title	60
3	SUB	Book Subjects	80
4	ANAME	Author Name	15

What Data Dictionary consists of

Data Dictionary consists of the following information −

- Name of the tables in the database
- Constraints of a table i.e. keys, relationships, etc.
- Columns of the tables that related to each other
- Owner of the table
- Last accessed information of the object
- Last updated information of the object

An example of Data Dictionary can be personal details of a student −

Example

<StudentPersonalDetails>

An example of Data Dictionary can be personal details of a student −

Example

<StudentPersonalDetails>

Student_ID	Student_Name	Student_Address	Student_City

-

Field Name	Datatype	Field Length	Constraint	Description
Student_ID	Number	5	Primary Key	Student id
Student_Name	Varchar	20	Not Null	Name of the student
Student_Address	Varchar	30	Not Null	Address of the student
Student_City	Varchar	20	Not Null	City of the student

Types of Data Dictionary

Here are the two types of data dictionary −

Active Data Dictionary

The DBMS software manages the active data dictionary automatically. The modification is an automatic task and most RDBMS has active data dictionary. It is also known as integrated data dictionary.

Passive Data Dictionary

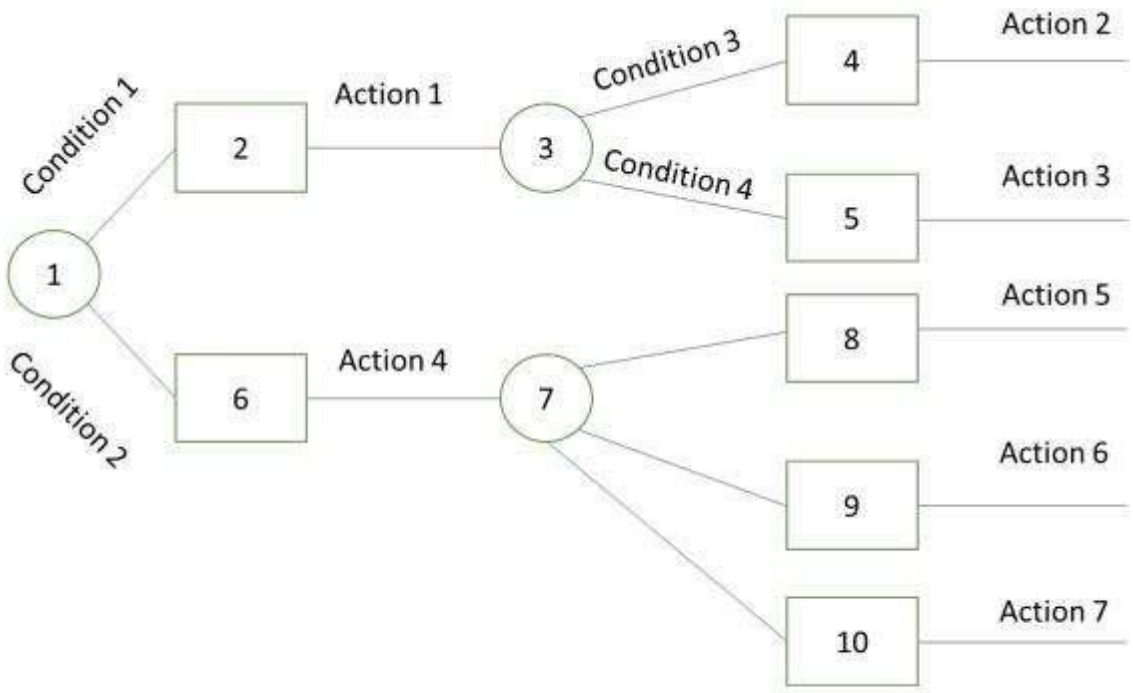
Managed by the users and is modified manually when the database structure change. Also known as non-integrated data dictionary

Decision Trees

Decision trees are a method for defining complex relationships by describing decisions and avoiding the problems in communication. A decision tree is a diagram that shows alternative actions and conditions within horizontal tree framework. Thus, it depicts which conditions to consider first, second, and so on.

Decision trees depict the relationship of each condition and their permissible actions.

A square node indicates an action and a circle indicates a condition. It forces analysts to consider the sequence of decisions and identifies the actual decision that must be made.



Decision Tree – Generalized Structure

The major limitation of a decision tree is that it lacks information in its format to describe what other combinations of conditions you can take for testing. It is a single representation of the relationships between conditions and actions.

For example, refer the following decision tree:

Decision Tables

- Decision tables are a method of describing the complex logical relationship in a precise manner which is easily understandable.

- ❖ It is useful in situations where the resulting actions depend on the occurrence of one or several combinations of independent conditions.
- ❖ It is a matrix containing row or columns for defining a problem and the actions.

Components of a Decision Table

Condition Stub: It is in the upper left quadrant which lists all the condition to be checked.

Action stub: It is in the lower left quadrant which outlines all the action to be carried out to meet such condition.

Condition Entry: It is in upper right quadrant which provides answers to questions asked in condition stub quadrant.

Action Entry: It is in lower right quadrant which indicates the appropriate action resulting from the answers to the conditions in the condition entry quadrant.

The entries in decision table are given by Decision Rules which define the relationships between combinations of conditions and courses of action. In rules section,

Y shows the existence of a condition.

N represents the condition, which is not satisfied.

A blank - against action states it is to be ignored.

X (or a check mark will do) against action states it is to be carried out.

For example, refer the following table:

CONDITIONS	**Rule 1**	**Rule 2**	**Rule 3**	**Rule 4**
Advance payment	Y	N	N	N
Purchase amount = RS 10,000	-	Y	Y	N
Regular Customer	-	Y	N	-
ACTIONS				
Give 5% discount	X	X	--	-
Give no discount	-	-	X	X

Structured English

Structure English is derived from structured programming language which gives more understandable and precise description of process. It is based on procedural logic that uses construction and imperative sentences designed to perform operation for action.

- It is best used when sequences and loops in a program must be considered and the problem needs sequences of actions with decisions.
- It does not have strict syntax rule. It expresses all logic in terms of sequential decision structures and iterations.

For example, see the following sequence of actions:

if customer pays advance
then
Give 5% Discount
Else
if purchase amount >=10,000
then
if the customer is a regular customer
then Give 5% Discount
else No Discount
end if else
No Discount
end if
end if

Pseudocode

A pseudocode does not conform to any programming language and expresses logic in plain English.

It may specify the physical programming logic without actual coding during and after the physical design.

It is used in conjunction with structured programming.

It replaces the flowcharts of a program.

Advantages and disadvantages of Pseudocode

Advantages:

1. It can be easily in any word processor.
2. It can be easily modified as compared to flowchart.
3. Its implementation is very useful in structured design elements.
4. It can be written easily.
5. It can be read and understood easily.
6. Converting a pseudocode to programming language is very easy as compared with converting a flowchart to programming language

Disadvantages:

1. It is not visual.
2. We do not get a picture of the design.
3. There is no standardized style or format, so one pseudocode may be different from another.
4. For a beginner, It is more difficult to follow the logic or write pseudocode as compared to flowchart.
5. Create an additional level of documentation to maintain.
6. Introduce error possibilities in translating to code.
7. May require tool to extract pseudocode and facilitate drawing flowcharts.

Guidelines for Selecting Appropriate Tools

Use the following guidelines for selecting the most appropriate tool that would suit your requirements:

- Use DFD at high- or low-level analysis for providing good system documentations.
- Use data dictionary to simplify the structure for meeting the data requirement of the system.
- Use structured English if there are many loops and actions are complex.
- Use decision tables when there are a large number of conditions to check and logic is complex.
- Use decision trees when sequencing of conditions is important and if there are few conditions to be tested.