**Chapter 2 : System Analysis**

System analysis is the act, process, or profession of studying an activity (such as a procedure, a business, or a physiological function) typically by mathematical means in order to define its goals or purposes and to discover operations and procedures for accomplishing them most efficiently.

**System Analysis methods :**

**-Stage 0 – Feasibility study**

In order to determine whether or not a given project is feasible, there must be some form of investigation into the goals and implications of the project. For very small scale projects this may not be necessary at all as the scope of the project is easily understood. In larger projects, the feasibility may be done but in an informal sense, either because there is no time for a formal study or because the project is a “must-have” and will have to be done one way or the other. A data flow Diagram is used to describe how the current system works and to visualize the known problems.

When a feasibility study is carried out, there are four main areas of consideration:

Technical – is the project technically possible?

Financial – can the business afford to carry out the project?

Organizational – will the new system be compatible with existing practices?

Ethical – is the impact of the new system socially acceptable?

To answer these questions, the feasibility study is effectively a condensed version of a comprehensive systems analysis and design. The requirements and usages are analyzed to some extent, some business options are drawn up and even some details of the technical implementation. The product of this stage is a formal feasibility study document. SSADM specifies the sections that the study should contain including any preliminary models that have been constructed and also details of rejected options and the reasons for their rejection.

**-Stage 1 – Investigation of the current environment**

The developers of SSADM understood that in almost all cases there is some form of current system even if it is entirely composed of people and paper. Through a combination of interviewing employees, circulating questionnaires, observations and existing documentation, the analyst comes to full understanding of the system as it is at the start of the project.

**-Stage 2 – Business system options**

Having investigated the current system, the analyst must decide on the overall design of the new system. To do this, he or she, using the outputs of the previous stage, develops a set of business system options. These are different ways in which the new system could be produced varying from doing nothing to throwing out the old system entirely and building an entirely new one. The analyst may hold a brainstorming session so that as many and various ideas as possible are generated.

The ideas are then collected to options which are presented to the user. The options consider the following:

* the degree of automation
* the boundary between the system and the users
* the distribution of the system, for example, is it centralized to one office or spread out across several?
* cost/benefit
* impact of the new system

Where necessary, the option will be documented with a logical data structure and a level 1 data-flow diagram.

The users and analyst together choose a single business option. This may be one of the ones already defined or may be a synthesis of different aspects of the existing options. The output of this stage is the single selected business option together with all the outputs of the feasibility stage.

Stage 3 – Requirements specification

This is probably the most complex stage in SSADM. Using the requirements developed in stage 1 and working within the framework of the selected business option, the analyst must develop a full logical specification of what the new system must do. The specification must be free from error, ambiguity and inconsistency. By logical, we mean that the specification does not say how the system will be implemented but rather describes what the system will do.

To produce the logical specification, the analyst builds the required logical models for both the data-flow diagrams (DFDs) and the Logical Data Model (LDM), consisting of the Logical Data Structure (referred to in other methods as entity relationship diagrams) and full descriptions of the data and its relationships. These are used to produce function definitions of every function which the users will require of the system, Entity Life-Histories (ELHs) which describe all events through the life of an entity, and Effect Correspondence Diagrams (ECDs) which describe how each event interacts with all relevant entities. These are continually matched against the requirements and where necessary, the requirements are added to and completed.

The product of this stage is a complete requirements specification document which is made up of:

* the updated data catalogue
* the updated requirements catalogue
* the processing specification which in turn is made up of
* user role/function matrix
* function definitions
* required logical data model
* entity life-histories
* effect correspondence diagrams

**Stage 4 – Technical system options**

This stage is the first towards a physical implementation of the new system. Like the Business System Options, in this stage a large number of options for the implementation of the new system are generated. This is narrowed down to two or three to present to the user from which the final option is chosen or synthesized.

However, the considerations are quite different being:

* the hardware architectures
* the software to use
* the cost of the implementation
* the staffing required
* the physical limitations such as a space occupied by the system
* the distribution including any networks which that may require
* the overall format of the human computer interface

All of these aspects must also conform to any constraints imposed by the business such as available money and standardization of hardware and software.

The output of this stage is a chosen technical system option.

**Stage 5 – Logical design**

Though the previous level specifies details of the implementation, the outputs of this stage are implementation-independent and concentrate on the requirements for the human computer interface. The logical design specifies the main methods of interaction in terms of menu structures and command structures.

One area of activity is the definition of the user dialogues. These are the main interfaces with which the users will interact with the system. Other activities are concerned with analyzing both the effects of events in updating the system and the need to make inquiries about the data on the system. Both of these use the events, function descriptions and effect correspondence diagrams produced in stage 3 to determine precisely how to update and read data in a consistent and secure way.

The product of this stage is the logical design which is made up of:

* Data catalogue
* Required logical data structure
* Logical process model – includes dialogues and model for the update and inquiry processes
* Stress & Bending moment.

**Stage 6 – Physical design**

This is the final stage where all the logical specifications of the system are converted to descriptions of the system in terms of real hardware and software. This is a very technical stage and a simple overview is presented here.

The logical data structure is converted into a physical architecture in terms of database structures. The exact structure of the functions and how they are implemented is specified. The physical data structure is optimized where necessary to meet size and performance requirements.

The product is a complete Physical Design which could tell software engineers how to build the system in specific details of hardware and software and to the appropriate standards.

**Data flow Diagram (DFD) :**

A data-flow diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

**DFD Level 0 :**

DFD Level 0 is also called a Context Diagram. It’s a basic overview of the whole system or process being analyzed or modeled. It’s designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.

**DFD Level 1 :**

Level 1 - The Level 0 DFD is broken down into more specific, Level 1 DFD. Level 1 DFD depicts basic modules in the system and flow of data among various modules. Level 1 DFD also mentions basic processes and sources of information.

* It provides a more detailed view of the Context Level Diagram.
* Here, the main functions carried out by the system are highlighted as we break into its sub-processes.

**Data Dictionary :**

A data dictionary, or metadata repository, as defined in the IBM Dictionary of Computing, is a "centralized repository of information about data such as meaning, relationships to other data, origin, usage, and format". Oracle defines it as a collection of tables with metadata. The term can have one of several closely related meanings pertaining to databases and database management systems (DBMS):

A document describing a database or collection of databases

An integral component of a DBMS that is required to determine its structure

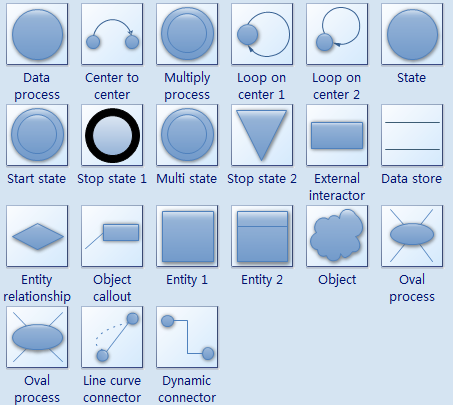
A piece of middleware that extends or supplants the native data dictionary of a DBMS

**Data Flow Diagrams Symbols**

There are some symbols that are used in the drawing of business process diagrams (data flow diagrams). These are now explained, together with the rules that apply to them.

Flow diagrams in general are usually designed using simple symbols such as a rectangle, an oval or a circle depicting a processes, data stored or an external entity, and arrows are generally used to depict the data flow from one step to another.

A DFD usually comprises of four components. These four components can be represented by four simple symbols. These symbols can be explained in detail as follows: External entities (source/destination of data) are represented by squares; Processes (input-processing-output) are represented by rectangles with rounded corners; Data Flows (physical or electronic data) are referred to by arrows; and finally, Data Stores (physical or electronic like XML files) are presented by open-ended rectangles



**Data flow diagrams** present the logical flow of information through a system in graphical or pictorial form. Data flow diagrams have only four symbols, which makes it useful for communication between analysts and users. Data flow diagrams (DFDs) show the data used and provided by processes within a system. DFDs make use of four basic symbols.

Create structured analysis, information flow, process-oriented, data-oriented, and data process diagrams as well as data flowcharts.

**External Entity**

An external entity is a source or destination of a data flow which is outside the area of study. Only those entities which originate or receive data are represented on a business process diagram. The symbol used is an oval containing a meaningful and unique identifier.

**Process**

A process shows a transformation or manipulation of data flows within the system. The symbol used is a rectangular box which contains 3 descriptive elements:

Firstly an identification number appears in the upper left hand corner. This is allocated arbitrarily at the top level and serves as a unique reference.

Secondly, a location appears to the right of the identifier and describes where in the system the process takes place. This may, for example, be a department or a piece of hardware. Finally, a descriptive title is placed in the centre of the box. This should be a simple imperative sentence with a specific verb, for example 'maintain customer records' or 'find driver'.

**Data Flow**

A data flow shows the flow of information from its source to its destination. A data flow is represented by a line, with arrowheads showing the direction of flow. Information always flows to or from a process and may be written, verbal or electronic. Each data flow may be referenced by the processes or data stores at its head and tail, or by a description of its contents.

**Data Store**

A data store is a holding place for information within the system:

It is represented by an open ended narrow rectangle. Data stores may be long-term files such as sales ledgers, or may be short-term accumulations: for example batches of documents that are waiting to be processed. Each data store should be given a reference followed by an arbitrary number.

**Resource Flow**

A resource flow shows the flow of any physical material from its source to its destination. For this reason they are sometimes referred to as physical flows.

The physical material in question should be given a meaningful name. Resource flows are usually restricted to early, high-level diagrams and are used when a description of the physical flow of materials is considered to be important to help the analysis.

**External Entities**

It is normal for all the information represented within a system to have been obtained from, and/or to be passed onto, an external source or recipient. These external entities may be duplicated on a diagram, to avoid crossing data flow lines. Where they are duplicated a stripe is drawn across the left hand corner, like this.

The addition of a lowercase letter to each entity on the diagram is a good way to uniquely identify them.

**Processes**

When naming processes, avoid glossing over them, without really understanding their role. Indications that this has been done are the use of vague terms in the descriptive title area - like 'process' or 'update'.

The most important thing to remember is that the description must be meaningful to whoever will be using the diagram.

**Data Flows**

Double headed arrows can be used (to show two-way flows) on all but bottom level diagrams. Furthermore, in common with most of the other symbols used, a data flow at a particular level of a diagram may be decomposed to multiple data flows at lower levels.

**Data Stores**

Each store should be given a reference letter, followed by an arbitrary number. These reference letters are allocated as follows:

'D' - indicates a permanent computer file  
'M' - indicates a manual file  
'T' - indicates a transient store, one that is deleted after processing.

In order to avoid complex flows, the same data store may be drawn several times on a diagram. Multiple instances of the same data store are indicated by a double vertical bar on their left hand edge.

**Project’s Level 0 and Level 1:**