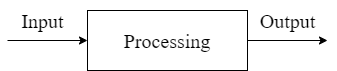
**System Analysis and Design**

# Overview of System Analysis and Design

The word System is derived from Greek word Systema, which means an organized relationship between any set of components to achieve some common cause or objectives. A system is “an orderly grouping of interdependent components linked together according to a plan to achieve a specific goal.” System is created to solve problems. One can think of the system approach as an organized way of dealing with a problem. In this dynamic world, the subject System Analysis and Design (SAD), mainly deals with software development activities.

In simple words we can define a system as a collection of components that work together to realize some objectives forms a system. Basically, there are three major components in every system, namely input, processing and output.



* 1. **Introduction to system analysis and Design:**

System development can generally be thought of as having two major components: System analysis and System design. System design is the process of planning a new business system or one to replace or complement an existing system. But before this planning can be done, we must thoroughly understand the old system and determine how computers can best be used to make its operation more effective. System analysis, then, is the process of gathering and interpreting facts, diagnosing problems, and using the information to recommend improvements to the system. This is the job of the systems analyst.

**System Analysis:**

Systems analysis is the process by which an individual studies a system such that an information system can be analyzed, modeled, and a logical alternative can be chosen. Systems analysis projects are initiated for three reasons: problems, opportunities, and directives. The people involved include systems analysts, sponsors, and users. The process by which systems are developed can be described by the [systems development life cycle](https://www.sciencedirect.com/topics/computer-science/system-development-life-cycle). The tasks, techniques, and tools used by the systems development life cycle can be referred as a methodology.

**System Design:**

System design is the process of defining the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system. It is meant to satisfy specific needs and requirements of a business or organization through the engineering of a coherent and well-running system.

* 1. **Information system and its type:**

An information system is a set of interrelated components that works together to collect, process, store and breakdown the information to support decision making. Any specific information system aims to support operations, management and decision making. In other words, an information system means a collection of interrelated components which work together to **gather, process, store,**and **break down** the information to help decision making.

**Types/Dimensions:**

1. **Management Information System:**

The management information system provides aid to managers by automating different processes that were initially done manually. Business activities like business performance tracking and analysis, making business decisions, making a business plan, and defining workflow. It also provides feedback to the managers by analyzing the roles and responsibilities.

* It enhances the efficiency and productivity of the company
* It provides a clear picture of the organization’s performance
* It adds value to the existing products, introduces innovation and improves product development
* It assists in communication and planning for business processes
* It helps the organization provide a competitive advantage

1. **Decision Support System:**

A decision support system is an information system that analyses business data and other information related to the enterprise to offer automation in decision making or problem-solving. A manager uses it in times of adversities arising during the operation of the business. Generally, the decision support system is used to collect information regarding revenue, sales figures or inventory. It is used across different industries, and the decision support system is a popular information system.

1. **Transaction Processing System:**

The transaction processing system automates the transaction collection, modification, and retrieval process. The peculiar characteristic of this type of information system is that it increases the performance, reliability and consistency of business transactions. It helps businesses perform daily operations smoothly without hassle.

1. **Experts System:**

Experts systems include expertise in order to add managers in diagnosing problem or in problem-solving. These systems are based on the principle of artificial intelligence research. Experts System is a knowledge-based information system. It uses its knowledge about a specify are to act as an expert consultant to users. Knowledge-base and software modules are the consultant of an expert system. These modules perform inference on the knowledge and offer answers to a user’s question.

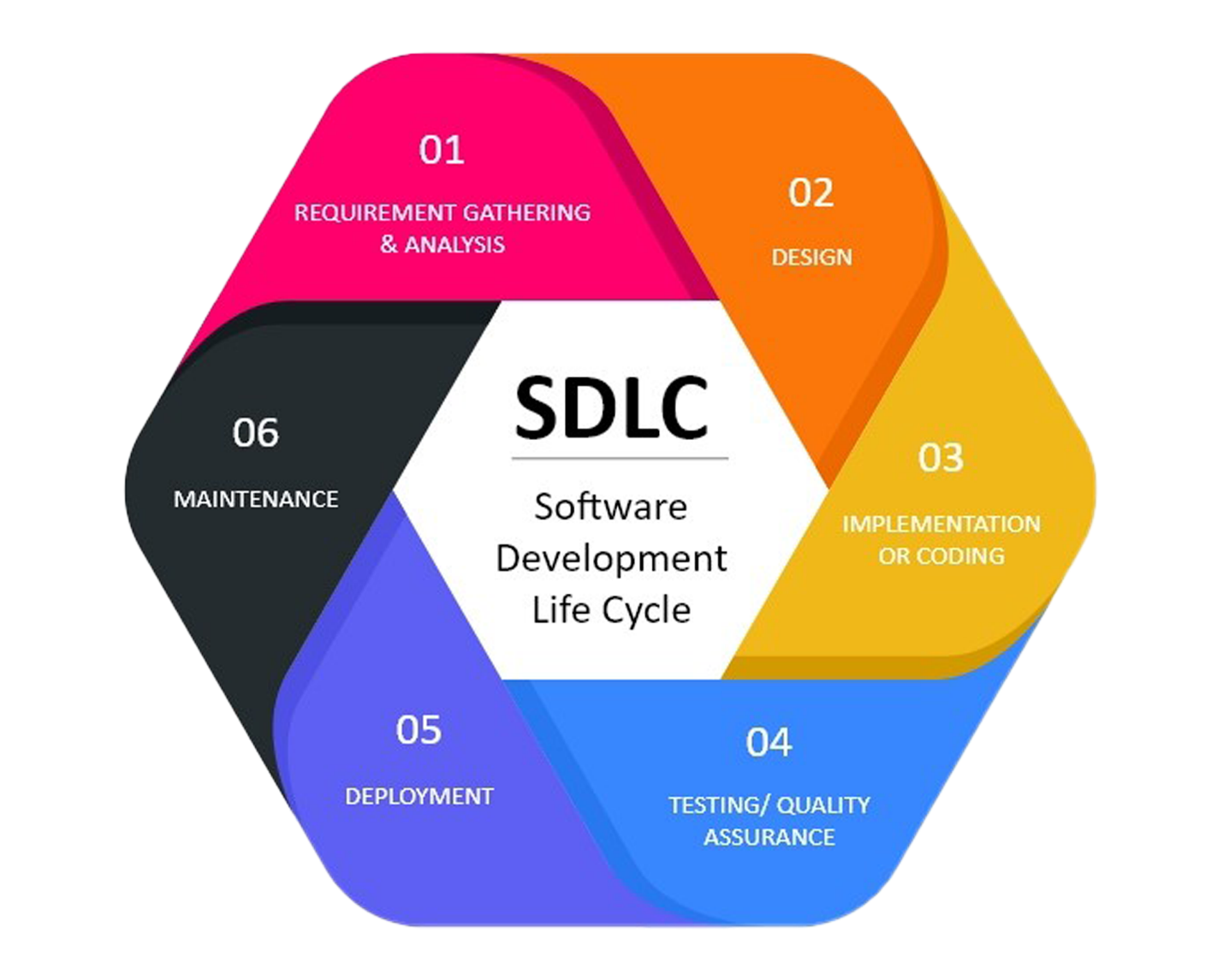
* 1. **Stakeholders of Information System:**

In the information system the system owners, system users, system designers, system builders and all of the above system analysts and the project managers all these are commonly known as the stakeholders. Whatever their roles in an organization are it doesn’t matter, all the thing common in them are that, they are information workers. These are some of the persons who are known as Information System stakeholders:

* System Owners
* System Users
  + Internal System Users
    - Clerical and Service workers
    - Technical and professional staffs
    - Supervisors, Middle managers and executive managers
  + External System User
    - Customers
    - Suppliers
    - Partners
    - Employees
* System Designers
  + Database Administrators
  + Network Architects
  + Web Architects
  + Graphics Artists
  + Security Experts
  + Technology specialists
* System Builders
  + Application programmers
  + System programmers
  + Databases Programmers
  + Network Administrators
  + Security Administrators
  + Webmasters
  + Software Integrators
* System Analysts
  1. **System Development Life cycle (SDLC):**

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates. SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.



1. **SDLC Model:**

There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models". Each process model follows a Series of steps unique to its type to ensure success in the process of software development.

Some of the SDLC models are:

* Waterfall Model
* Spiral Model
* Prototype Model
* **Waterfall Model:**

The waterfall is a cascade SDLC model that presents the development process like the flow, moving step by step through the phases of analysis, projecting, realization, testing, implementation, and support. This SDLC model includes gradual execution of every stage. Waterfall implies strict documentation. The features expected of each phase of this SDLC model are predefined in advance.

The waterfall life cycle model is considered one of the best-established ways to handle complex projects. This approach allows avoiding many mistakes that may appear because of insufficient control over the project. However, it results in pervasive documentation development. It is beneficial to the developers who may be working with the product in the future, but it takes a long time to write everything down.

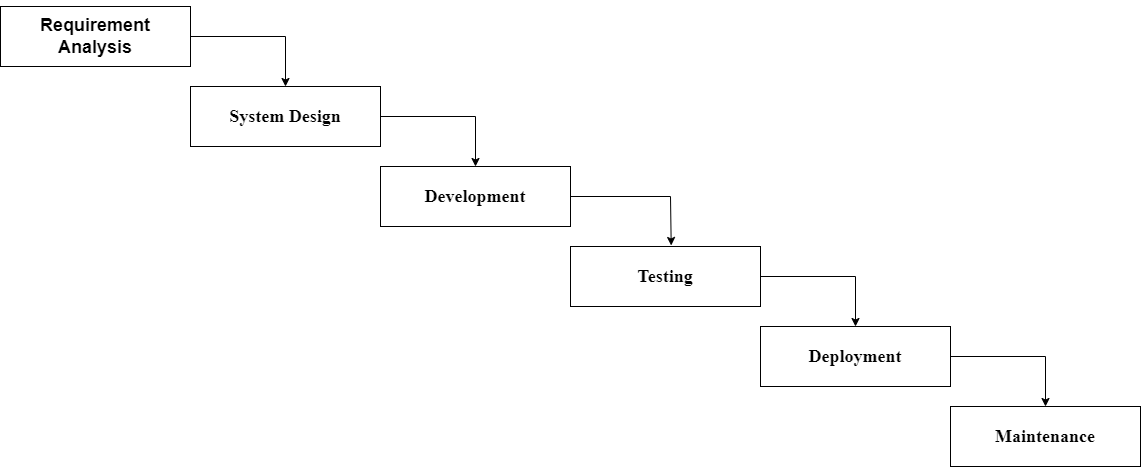


Fig: Waterfall Model

* **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
* **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
* **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
* **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
* **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
* **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

**Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are :

* Requirements are very well documented, clear and fixed.
* Product definition is stable.
* Technology is understood and is not dynamic.
* There are no ambiguous requirements.
* Ample resources with required expertise are available to support the product.
* The project is short.

**Advantages:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Some of the major advantages of the Waterfall Model are as follows −

* Simple and easy to understand and use
* Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
* Phases are processed and completed one at a time.
* Works well for smaller projects where requirements are very well understood.
* Clearly defined stages.
* Well understood milestones.
* Easy to arrange tasks.
* Process and results are well documented.

**Disadvantage:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows −

* No working software is produced until late during the life cycle.
* High amounts of risk and uncertainty.
* Not a good model for complex and object-oriented projects.
* Poor model for long and ongoing projects.
* Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
* It is difficult to measure progress within stages.
* Cannot accommodate changing requirements.
* Adjusting scope during the life cycle can end a project.
* **Spiral Model:**

The spiral model is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model. The spiral model is used by software engineers and is favored for large, expensive and complicated projects.

The spiral model enables gradual releases and refinement of a product through each phase of the spiral as well as the ability to build prototypes at each phase. The most important feature of the model is its ability to manage unknown risks after the project has commenced; creating a prototype makes this feasible



Fig: Spiral Model

The spiral model has four phases. A software project repeatedly passes through these phases in iterations called Spirals.

**Identification:**

This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.

**Design:**

The Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and the final design in the subsequent spirals.

**Construct and Build:**

The Construct phase refers to production of the actual software product at every spiral. In the baseline spiral, when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.

**Evaluation and Risk Analysis:**

Risk Analysis includes identifying, estimating and monitoring the technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

**Application:**

The Spiral Model is widely used in the software industry as it is in sync with the natural development process of any product, i.e., learning with maturity which involves minimum risk for the customer as well as the development firms.

The following pointers explain the typical uses of a Spiral Model −

* When there is a budget constraint and risk evaluation is important.
* For medium to high-risk projects.
* Long-term project commitment because of potential changes to economic priorities as the requirements change with time.
* Customer is not sure of their requirements which is usually the case.
* Requirements are complex and need evaluation to get clarity.
* New product line which should be released in phases to get enough customer feedback.
* Significant changes are expected in the product during the development cycle.

**Advantages:**

The advantages of the Spiral SDLC Model are as follows −

* Changing requirements can be accommodated.
* Allows extensive use of prototypes.
* Requirements can be captured more accurately.
* Users see the system early.
* Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.

**Disadvantages:**

The disadvantages of the Spiral SDLC Model are as follows −

* Management is more complex.
* End of the project may not be known early.
* Not suitable for small or low risk projects and could be expensive for small projects.
* Process is complex
* Spiral may go on indefinitely.
* Large number of intermediate stages requires excessive documentation.
* **Prototype model:**

**Prototyping Model** is a software development model in which prototype is built, tested, and reworked until an acceptable prototype is achieved. It also creates base to produce the final system or software. It works best in scenarios where the project’s requirements are not known in detail. It is an iterative, trial and error method which takes place between developer and client.

A prototype usually turns out to be a very crude version of the actual system, possible exhibiting limited functional capabilities, low reliability, and inefficient performance as compared to actual software.



Fig: Prototype Model

**Requirement gathering and analysis:**

A prototyping model starts with requirement analysis. In this phase, the requirements of the system are defined in detail. During the process, the users of the system are interviewed to know what is their expectation from the system.

**Quick Design:**

The second phase is a preliminary design or a quick design. In this stage, a simple design of the system is created. However, it is not a complete design. It gives a brief idea of the system to the user. The quick design helps in developing the prototype.

**Build a prototype:**

In this phase, an actual prototype is designed based on the information gathered from quick design. It is a small working model of the required system.

**Initial user evaluation:**

In this stage, the proposed system is presented to the client for an initial evaluation. It helps to find out the strength and weakness of the working model. Comment and suggestion are collected from the customer and provided to the developer.

**Refining Prototype:**

If the user is not happy with the current prototype, you need to refine the prototype according to the user’s feedback and suggestions. This phase will not over until all the requirements specified by the user are met. Once the user is satisfied with the developed prototype, a final system is developed based on the approved final prototype.

**Implementation Product and Maintain:**

Once the final system is developed based on the final prototype, it is thoroughly tested and deployed to production. The system undergoes routine maintenance for minimizing downtime and prevent large-scale failures.

**Application:**

* Software Prototyping is most useful in development of systems having high level of user interactions such as online systems.
* Systems which need users to fill out forms or go through various screens before data is processed can use prototyping very effectively to give the exact look and feel even before the actual software is developed.
* Software that involves too much of data processing and most of the functionality is internal with very little user interface does not usually benefit from prototyping.
* Prototype development could be an extra overhead in such projects and may need lot of extra efforts.

**Advantages:**

* Helps team member to communicate effectively
* Customer satisfaction exists because the customer can feel the product at a very early stage.
* There will be hardly any chance of software rejection.
* Quicker user feedback helps you to achieve better software development solutions.
* Allows the client to compare if the software code matches the software specification.
* It helps you to find out the missing functionality in the system.
* It also identifies the complex or difficult functions.
* Encourages innovation and flexible designing.

**Disadvantages:**

* Prototyping is a slow and time taking process.
* The cost of developing a prototype is a total waste as the prototype is ultimately thrown away.
* Prototyping may encourage excessive change requests.
* Sometimes customers may not be willing to participate in the iteration cycle for the longer time duration.
* There may be far too many variations in software requirements when each time the prototype is evaluated by the customer.
* Poor documentation because the requirements of the customers are changing.
* It is very difficult for software developers to accommodate all the changes demanded by the clients.
  + 1. **Introduction to analysis and design tools:**

Software analysis and design includes all activities, which help the transformation of requirement specification into implementation. Requirement specifications specify all functional and non-functional expectations from the software.

Some of the few analysis and design tools used by software designers are:

1. Data Flow Diagram
2. Structured English
3. Pseudo-Code
4. Decision Tables
5. Entity Relationship Model
6. Data Dictionary

**Data Flow Diagram:**

Data flow diagram is graphical representation of flow of data in an information system. It is capable of depicting incoming data flow, outgoing data flow and stored data. The DFD does not mention anything about how data flows through the system.

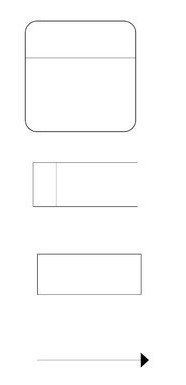
There is a prominent difference between DFD and Flowchart. The flowchart depicts flow of control in program modules. DFDs depict flow of data in the system at various levels. DFD does not contain any control or branch elements.

**Types of DFD:**

Data Flow Diagrams are either Logical or Physical.

* **Logical DFD** - This type of DFD concentrates on the system process, and flow of data in the system. For example in a Banking software system, how data is moved between different entities.
* **Physical DFD** - This type of DFD shows how the data flow is actually implemented in the system. It is more specific and closer to the implementation.

Here are some notations used in DFD:

****

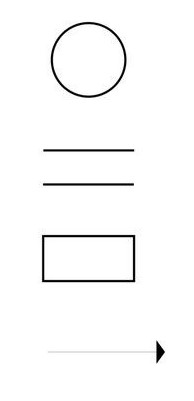
Data Flow

Source/sink

Data Store

Process

Fig: Gane and Sarson Symbols

****

Source/sink

Data Store

Process

Data Flow

Fig: DeMarco and Yourdon Symbols.

**Structured English:**

Structured English is the It uses plain English words in structured programming paradigm. It is not the ultimate code but a kind of description what is required to code and how to code it. The following are some tokens of structured programming.

Example

Enter Customer\_Name

SEEK Customer\_Name in Customer\_Name\_DB file

IF Customer\_Name found THEN

Call procedure USER\_PASSWORD\_AUTHENTICATE ()

ELSE

PRINT error message

Call procedure NEW\_CUSTOMER\_REQUEST ()

ENDIF

**Pseudo Code:**

Pseudo code is written more close to programming language. It may be considered as augmented programming language, full of comments and descriptions. Pseudo code avoids variable declaration but they are written using some actual programming language’s constructs, like C, Fortran, Pascal etc. Pseudo code contains more programming details than Structured English. It provides a method to perform the task, as if a computer is executing the code.\

Example:

BEGIN

NUMBER s1, s2, sum

OUTPUT ("Input number1:")

INPUT s1

OUTPUT ("Input number2:")

INPUT s2

sum=s1+s2

OUTPUT sum

END

**Decision Table:**

A **Decision Table** is a tabular representation of inputs versus rules/cases/test conditions. It is a very effective tool used for both complex software testing and requirements management. Decision table helps to check all possible combinations of conditions for testing and testers can also identify missed conditions easily.

Example:

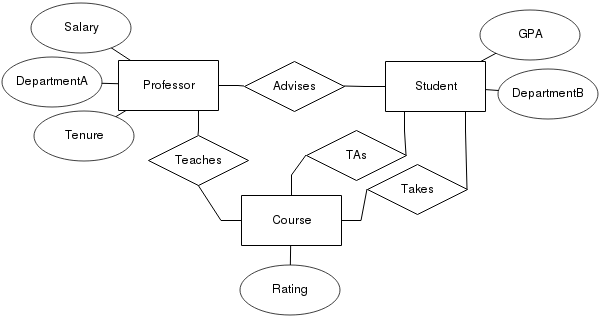
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Conditions** | **Rule 1** | **Rule 2** | **Rule 3** | **Rule 4** |
| **Username (T/F)** | F | T | F | T |
| **Password (T/F)** | F | F | T | T |
| **Output (E/H)** | E | E | E | H |

**Entity Relation Diagram (ER-Diagram):**

**ER Diagram** stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.

Example:

****

* Entity - An entity in ER Model is a real world being, which has some properties called attributes. Every attribute is defined by its corresponding set of values, called domain.
* Relationship - The logical association among entities is called relationship. Relationships are mapped with entities in various ways. Mapping cardinalities define the number of associations between two entities.

Mapping cardinalities:

* + one to one
  + one to many
  + many to one
  + many to many

**Data Dictionary:**

Data Dictionary is a collection of names, definitions, and attributes about data elements that are being used or captured in a database, information system, or part of a research project. It describes the meanings and purposes of data elements within the context of a project, and provides guidance on interpretation, accepted meanings and representation. A Data Dictionary also provides metadata about data elements. The metadata included in a Data Dictionary can assist in defining the scope and characteristics of data elements, as well the rules for their usage and application.