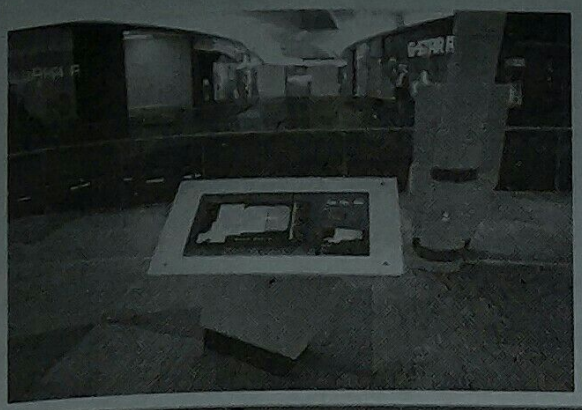

8

CHAPTER

DEVELOPMENT OF MIS

8.1 STAGES FOR DEVELOPMENT OF MIS

The System Development is the interactive process which consists of the following stages

- Preliminary Investigation
- System Analysis
- System Design
- Programming and Testing
- Implementation
- Operation and Maintenance

The following Figure shows various stages in development of MIS

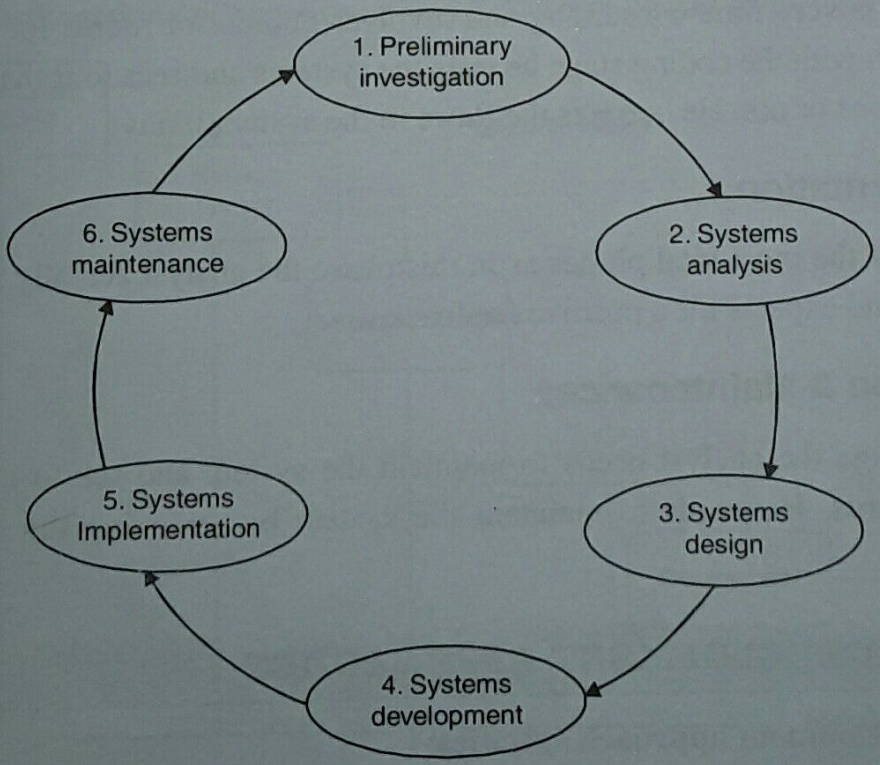

Fig 8.1 Stages in development of MIS

8.1.1. Preliminary Investigation:

One of the most tedious task is to recognize the real problem of the pre-installed system. The analyst has to spend hours and days for understanding the fault in the system. This fault could have however been overcome if the Preliminary Investigation before installing the system was properly done. This is the first stage of the development of the system. In this stage the analyst makes a survey by gathering all the available information needed for the system elements and allocation of the requirements to the software.

8.1.2 System Analysis:

The analyst understands the nature of the information and the functions of the software which are required for the system. The analyst makes a brief survey of the requirements and tries to analyze the performance of the system which is to be developed. He also makes sure that he gets enough information and resources for building the appropriate system.

8.1.3 System Design:

The analyst actually makes number of designs of the system on paper or on the computer and sees to it that the rough image made of the system comprises of all the requirements or not. Once this is done, the analyst selects and finalizes a best suited design for the development of the system.

8.1.4 Programming & Testing:

The analyst translates the design in such a way that it become source code for the system. The coding step is very time consuming and involves number of rooms for errors. Once the analyst is through with the coding stage he tests the systems and sees to it that it is working as per the expectations or not. He corrects the flaws in the system if any.

8.1.5 Implementation:

This is one of the most vital phases as in this phase the analyst actually gives the system to the customer and expects for a positive feedback.

8.1.6 Operation & Maintenance:

In the last stage the analyst needs to maintain the system and see to it that it working within the standards. He needs to maintain the system by removing the defects or flaws occurred.

8.2 SYSTEM DEVELOPMENT APPROACHES

A system development approach specifies: -

- A general process usually as a set of stages in which a project should be divided.

- The order in which the stages should be executed.
- Any other constraints & conditions on the execution of stages.

Basically a system development approach should answer the following questions: -

- What should we do next?
- How long shall we continue to do it?

There are several types of approaches towards developing a system: -

- Waterfall Model
- Iterative Model
- Software Prototyping Model
- Spiral Model

8.2.1 WATERFALL MODEL

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. Waterfall model is the earliest SDLC approach that was used for software development. The waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a linear-sequential life cycle model. This means that any phase in the development process begins only if the previous phase is complete. In waterfall model phases do not overlap.

8.2.2.1 Waterfall Model Design

Following is a diagrammatic representation of different phases of waterfall model.

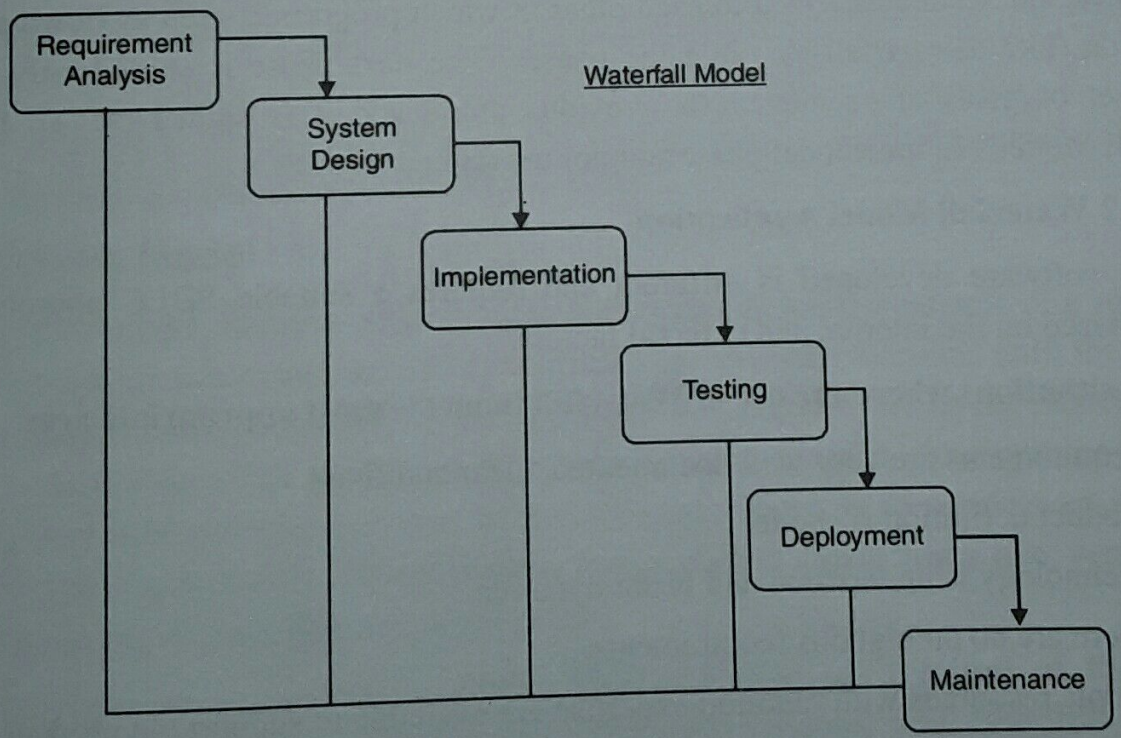

Fig 8.2 Waterfall Model

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

The sequential phases in Waterfall model are:

(a) Requirement Gathering and analysis: All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

(b) System Design: The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

(c) Implementation: With inputs from 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.

(d) 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.

(e) 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.

(f) 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.

8.2.2.2 Waterfall Model 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.

8.2.2.3 Waterfall Pros & Cons

The following table lists out the pros and cons of Waterfall model:

Tab 8.1 Pros& Cons of Waterfall Model

Pros	Cons
• 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.	• 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. • Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

8.2.2 Iterative Model

In Iterative model, iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed. An iterative life cycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which is then reviewed in order to identify further requirements. This process is then repeated, producing a new version of the software at the end of each iteration of the model.

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional

capabilities are added. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).

Following is the pictorial representation of Iterative and Incremental model:

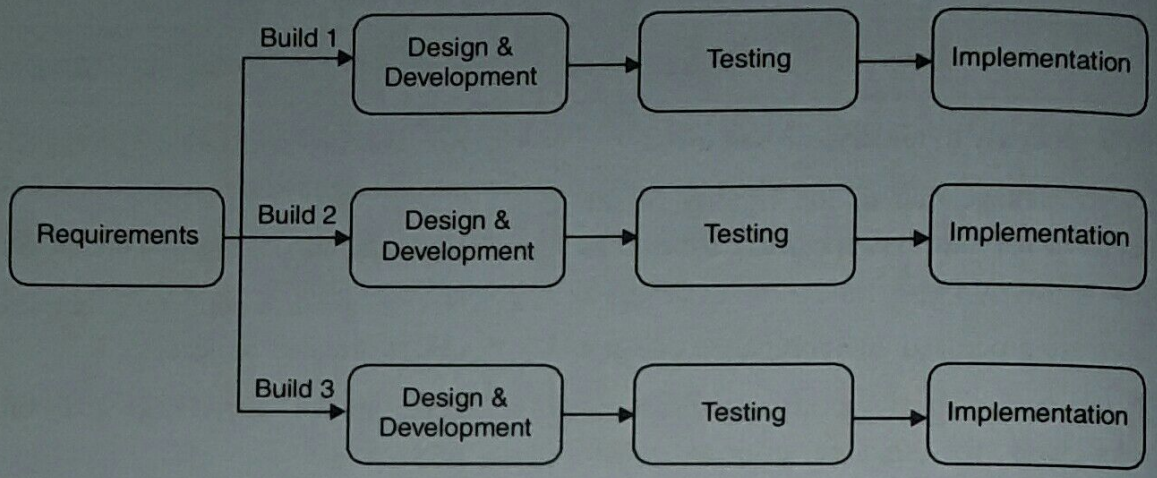

Fig 8.3 Iterative Enhancement Model

During software development, more than one iteration of the software development cycle may be in progress at the same time. So This process may be described as an incremental build approach. In incremental model the whole requirement is divided into various builds. During each iteration, the development module goes through the requirements, design, implementation and testing phases. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is ready as per the requirement.

The key to successful use of an iterative software development lifecycle is rigorous validation of requirements, and verification & testing of each version of the software against those requirements within each cycle of the model. As the software evolves through successive cycles, tests have to be repeated and extended to verify each version of the software.

8.2.2.1 Iterative Model Application

Like other SDLC models, Iterative and incremental development has some specific applications in the software industry.

This model is most often used in the following scenarios:

- Requirements of the complete system are clearly defined and understood.
- Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.
- A new technology is being used and is being learnt by the development team while working on the project.

- Resources with needed skill set are not available and are planned to be used on contract basis for specific iterations.
- There are some high risk features and goals which may change in the future.

8.2.2.2 Iterative Model Pros & Cons

The advantage of this model is that there is a working model of the system at a very early stage of development which makes it easier to find functional or design flaws. Finding issues at an early stage of development enables to take corrective measures in a limited budget.

The disadvantage with this SDLC model is that it is applicable only to large and bulky software development projects. This is because it is hard to break a small software system into further small serviceable increments/modules.

The following table lists out the pros and cons of Iterative and Incremental SDLC Model:

Tab 8.2 Pros& Cons of Iterative Model

Pros	Cons
• Some working functionality can be developed quickly and early in the life cycle. • Results are obtained early and periodically. • Parallel development can be planned. • Progress can be measured. • Less costly to change the requirements. • Testing and debugging during smaller iteration is easy. • Risks are identified and resolved during iteration; and each iteration is an easily managed milestone. • Easier to manage risk - High risk part is done first. • With every increment operational product is delivered • Issues, challenges & risks identified from each increment can be utilized/applied to the next increment. • Risk analysis is better. • It supports changing requirements.	• More resources may be required. • Although cost of change is lesser but it is not very suitable for changing requirements. • More management attention is required. • System architecture or design issues may arise because not all requirements are gathered in the beginning of the entire life cycle. • Defining increments may require definition of the complete system • Not suitable for smaller projects. • Management complexity is more. • End of project may not be known which is a risk. • Highly skilled resources are required for risk analysis. • Project progress is highly dependent upon the risk analysis phase.

• Initial Operating time is less. • Better suited for large and mission-critical projects. • During life cycle software is produced early which facilitates customer evaluation and feedback.	

8.2.3 Software Prototyping Model

The Software Prototyping refers to building software application prototypes which display the functionality of the product under development but may not actually hold the exact logic of the original software.

Software prototyping is becoming very popular as a software development model, as it enables to understand customer requirements at an early stage of development. It helps get valuable feedback from the customer and helps software designers and developers understand about what exactly is expected from the product under development.

8.2.3.1 What is Software Prototyping

- Prototype is a working model of software with some limited functionality.
- The prototype does not always hold the exact logic used in the actual software application and is an extra effort to be considered under effort estimation.
- Prototyping is used to allow the users evaluate developer proposals and try them out before implementation.
- It also helps understand the requirements which are user specific and may not have been considered by the developer during product design.

Following is the stepwise approach to design a software prototype:

- **Basic Requirement Identification:** This step involves understanding the very basics product requirements especially in terms of user interface. The more intricate details of the internal design and external aspects like performance and security can be ignored at this stage.
- **Developing the initial Prototype:** The initial Prototype is developed in this stage, where the very basic requirements are showcased and user interfaces are provided. These features may not exactly work in the same manner internally in the actual software developed and the workarounds are used to give the same look and feel to the customer in the prototype developed.
- **Review of the Prototype:** The prototype developed is then presented to the customer and the other important stakeholders in the project. The feedback is collected in an organized manner and used for further enhancements in the product under development.

- **Revise and enhance the Prototype:** The feedback and the review comments are discussed during this stage and some negotiations happen with the customer based on factors like, time and budget constraints and technical feasibility of actual implementation. The changes accepted are again incorporated in the new Prototype developed and the cycle repeats until customer expectations are met.

Prototypes can have horizontal or vertical dimensions. Horizontal prototype displays the user interface for the product and gives a broader view of the entire system, without concentrating on internal functions. A vertical prototype on the other side is a detailed elaboration of a specific function or a sub system in the product.

The purpose of both horizontal and vertical prototype is different. Horizontal prototypes are used to get more information on the user interface level and the business requirements. It can even be presented in the sales demos to get business in the market. Vertical prototypes are technical in nature and are used to get details of the exact functioning of the sub systems. For example, database requirements, interaction and data processing loads in a given sub system.

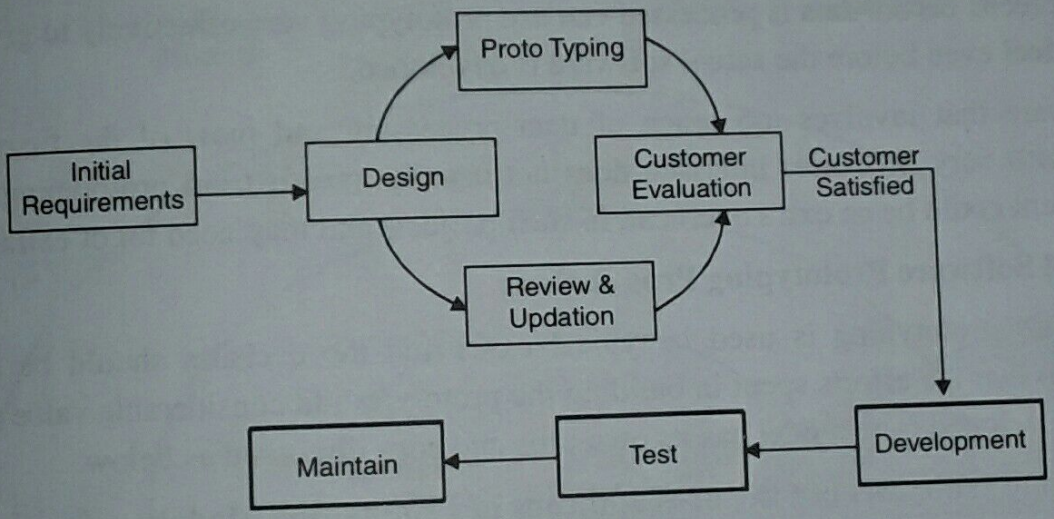

Fig 8.4 Prototyping Model

8.2.3.2 Types of Software Prototyping

There are different types of software prototypes used in the industry. Following are the major software prototyping types used widely:

(a) Throwaway/Rapid Prototyping: Throwaway prototyping is also called as rapid or close ended prototyping. This type of prototyping uses very little efforts with minimum requirement analysis to build a prototype. Once the actual requirements are understood, the prototype is discarded and the actual system is developed with a much clear understanding of user requirements.

(b) Evolutionary Prototyping: Evolutionary prototyping also called as breadboard prototyping is based on building actual functional prototypes with minimal functionality in the beginning. The prototype developed forms the heart of the future prototypes on top of

which the entire system is built. Using evolutionary prototyping only well understood requirements are included in the prototype and the requirements are added as and when they are understood.

(c) Incremental Prototyping: Incremental prototyping refers to building multiple functional prototypes of the various sub systems and then integrating all the available prototypes to form a complete system.

(d) Extreme Prototyping: Extreme prototyping is used in the web development domain. It consists of three sequential phases. First, a basic prototype with all the existing pages is presented in the html format. Then the data processing is simulated using a prototype services layer. Finally, the services are implemented and integrated to the final prototype. This process is called Extreme Prototyping used to draw attention to the second phase of the process, where a fully functional UI is developed with very little regard to the actual services.

8.3.2.3 Software Prototyping 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.

8.3.2.4 Software Prototyping Pros & Cons

Software prototyping is used in typical cases and the decision should be taken very carefully so that the efforts spent in building the prototype add considerable value to the final software developed. The model has its own pros and cons discussed as below.

Following table lists out the pros and cons of Prototyping Model:

Tab 8.3 Pros& Cons of Spiral Model

Pros	Cons
• Increased user involvement in the product even before implementation • Since a working model of the system is displayed, the users get a better understanding of the system being developed. • Reduces time and cost as the defects can be detected much earlier. • Quicker user feedback is available leading to better solutions.	• Risk of insufficient requirement analysis owing to too much dependency on prototype • Users may get confused in the prototypes and actual systems. • Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans. • Developers may try to reuse the existing

• Missing functionality can be identified easily • Confusing or difficult functions can be identified	prototypes to build the actual system, even when it's not technically feasible • The effort invested in building prototypes may be too much if not monitored properly.

8.2.4 SPIRAL MODEL

In 1986, the spiral model was first mentioned by *Barry Boehm*. It is a combination of a waterfall model and iterative model. Each phase in spiral model begins with a design goal and ends with the client reviewing the progress. The development team in Spiral-SDLC model starts with a small set of requirement and goes through each development phase for those set of requirements. The development team adds functionality for the additional requirement in every-increasing spirals until the application is ready for the production phase.

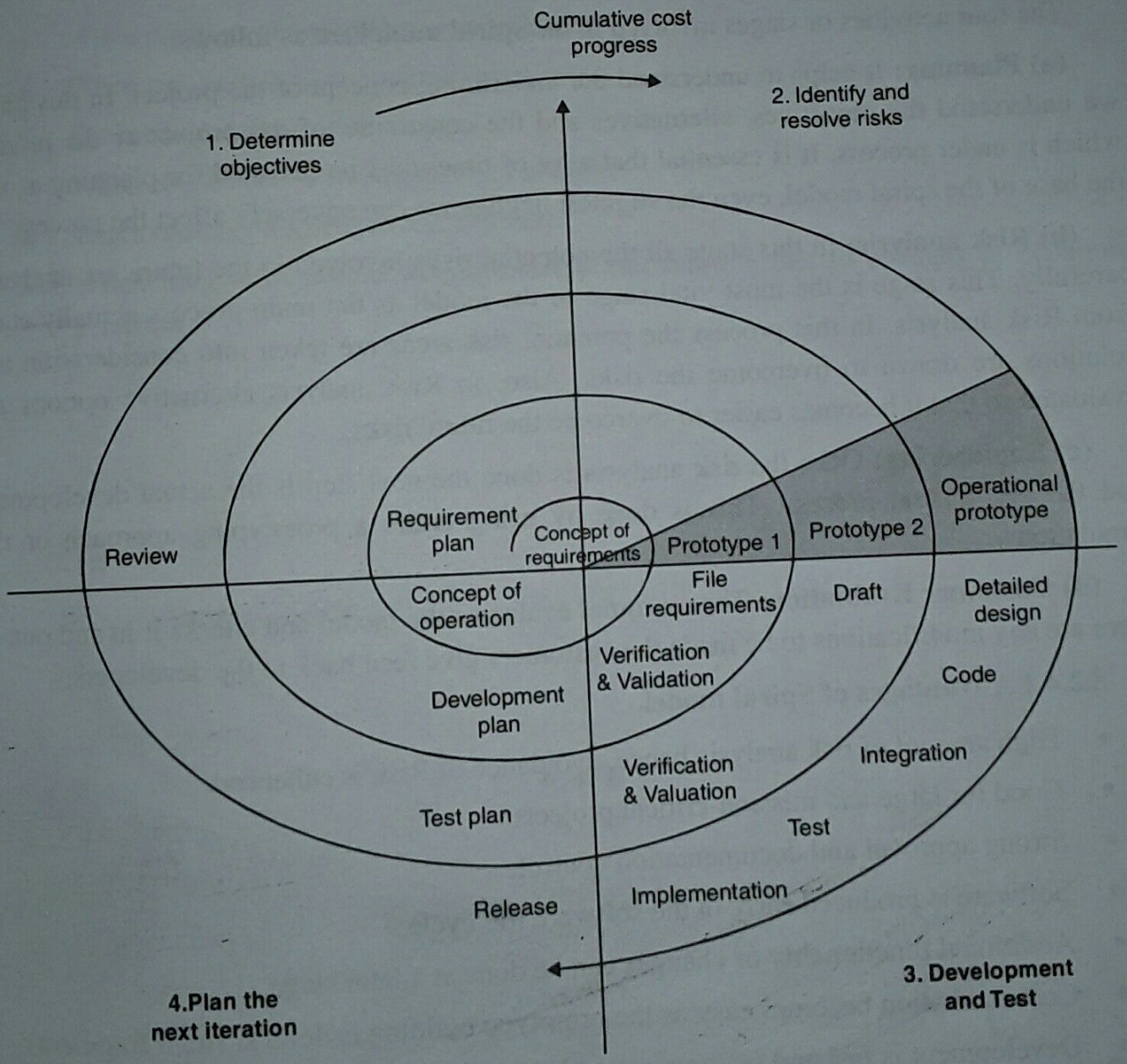

Fig 8.5 Spiral Model

There are number of reasons because of which ongoing project may stop. In order to overcome the hurdles in the future it is essential to make a brief study of the possibilities of the future problems and likewise steps must be taken for avoiding such problems. This will help the project to be carried on nonstop without any future hurdles and it also help to save time and money. Spiral model, is one such model which is useful in such situations where the future planning is essentials to eliminate the possible risks in the future. It helps to guide the risks that the project may face in the near future and what steps should be taken t overcome such risks. The *Spiral model* has a unique process design which helps to eliminate the risks in the future.

The above figure shows four stages of spiral model. The four stages are nothing but a part of a circular path which is critical in providing the guidelines for the future. Each stage of the *spiral model* is represented by one quadrant of the Cartesian design. The cost incurred is represented by the radius of the model.

The four activities or stages involved in the **spiral model** are as follows:

(a) Planning: It helps to understand the underlining concept of the project. In this phase we understand the objectives, alternatives and the constraints of the project or the product which is under process. It is essential that a lot of time must be invested for planning as it is the base of the spiral model, even the slightest negligence can adversely affect the process.

(b) Risk analysis: In this stage all the potential risks involved in the future are analyzed carefully. This stage is the most vital stage of the model as the main process actually starts from Risk analysis. In this process the potential risk areas are taken into consideration and solutions are drawn to overcome the risks. Also, in Risk analysis alternative options are evaluated so that it becomes easier to overcome the future risks.

(c) Engineering: Once the risk analysis is done the next step is the actual development and the verification process. This is done by using either a prototyping approach or the simulation.

(d) Customer Evaluation: The customer evaluates the model and checks it in and out. If there are any modifications to be made the customers give feed back to the developers.

8.2.4.1 Advantages of Spiral model

- High amount of risk analysis hence, avoidance of Risk is enhanced.
- Good for large and mission-critical projects.
- Strong approval and documentation control.
- Software is produced early in the software life cycle.
- Additional functionality or changes can be done at a later stage
- Cost estimation becomes easy as the prototype building is done in small fragments
- Development is fast and features are added in a systematic way
- There is always a space for customer feedback

8.4.2.2 Disadvantages of Spiral model

- Can be a costly model to use.
- Risk of not meeting the schedule or budget
- Risk analysis requires highly specific expertise.
- Project's success is highly dependent on the risk analysis phase.
- It works best for large projects only also demands risk assessment expertise. Doesn't work well for smaller projects.
- Documentation is more as it has intermediate phases

8.4.2.3 When to use Spiral-SDLC Model?

- When project is large
- When releases are required to be frequent
- When creation of a prototype is applicable
- When risk and costs evaluation is important
- For medium to high-risk projects
- When requirements are unclear and complex
- When changes may require at any time
- When long term project commitment is not feasible due to changes in economic priorities

✦ ✦ ✦

9

CHAPTER

DECISION

9.1 INTRODUCTION

We all know what a decision is; we make them all the time. An examination of the literature reveals the somewhat perplexing fact that most books on management and decision theory do not contain a specific definition of what is meant by decision. One can find detailed descriptions of decision trees, discussions of game theory and analyses of various statistical treatments of payoffs matrices under conditions of uncertainty, but the definition of the decision activity itself is often taken for granted and is associated with a choice made between alternative courses of action. In addition, the term decision can be defined as a "report or statement after the fact." In the case of human decisions, we can extend the definition to include references to the cognitive processes that result in the selection of some course of action. However, studies have shown that most people are much poorer at decision making than they think.

According to Ofstad, a concise description of alternative definitions has given:

- To say that a person has made a decision may mean that he has started a series of behavioral reactions in favor of something, or
- He has made up his mind to do a certain action, which has no doubts that he ought to do.
- But the most common is to make a judgement regarding what one ought to do in a certain situation after having deliberated on some alternative courses of action

The decision event is interesting in that it provides a threshold and incorporates the temporal dimension into the decision. Once a decision is "made", the conclusion upon which alternative course of action will be taken is reached, and the action initiated, that threshold has been crossed. The amount of time between the recognition that a decision is required and the decision event represents the total amount of time for the decision process. While some decisions are made with ample time, others are made without the benefit of time. Regardless

of the amount of time taken to reach the conclusion on an alternative, once the decision event has occurred, the decision is "made."

Perhaps the most important component of the decision definition is placing a decision into action. Until we act on a conclusion of a decision process, and cross the threshold of the decision event, we are still able to consider alternatives and "change our minds." Decision event and decision action are tightly coupled in that the action defines the event.

Once an action has been taken, we enter the consequence stage of a decision. Consequences are the results of actions. They may be insignificant, even undetectable, or they may be catastrophic. One of the key considerations in the magnitude and importance of a decision is the potential consequences.

9.2 TYPES OF DECISIONS

Decisions can be classified in various ways based on:

- Purpose of Decision-making
- Levels of Programmability
- Complexity
- Knowledge of Outcomes

9.2.1 Classification based on Purpose of Decision-making

On the basis of the purpose of decision-making activities, the organizational decisions are divided into three categories:

Strategic Planning Decisions: Strategic planning decisions are those decisions in which the decision-maker develops objectives and allocates resources to achieve these objectives. Such decisions are taken by strategic planning level (top level) managers.

Management Control Decisions: Management control decisions are taken by management control level (middle level) managers and deal with the use of resources in the organization.

Operational Control Decisions: Operational control decisions deal with the day-to-day problems that affect the operation of the organization. These decisions are taken by the managers at operational level (bottom level) of the organization.

9.2.2 Classification based on Levels of Programmability

Simon on the basis of level of the programmability of a decision, proposed two types of decisions:

(a) Programmed, also known as structured decisions

(b) Non-programmed, also known as unstructured decisions.

Programmed/Structured Decisions: Programmed or structured are those decisions, which

are well defined and some specified procedure or some decision rule might be applied to reach a decision. Such decisions are routine and repetitive and require little time for developing alternatives in the design phase. Programmed or structured decisions have traditionally been made through habit, by operating procedures or with other accepted tools.

Non-programmed /Unstructured Decision: Decisions, which are not well defined and have not pre-specified procedures decision rule are known as unstructured or non-programmed decisions.

The following table shows comparison between structured and non-structured decisions:

Tab 9.1 Structured vs Non-Structured Decisions

Structured Decisions	Un-structured Decisions
• Well defined	• Not well defined
• Repetitive & routine.	• Occasional & unique.
• It involves less cost.	• It involves in high cost.
• Algorithms can be designed for it.	• Algorithms cannot be designed for it.
• Such decision can be made with the help of computer.	• Such decision cannot be made with the help of computer.

9.2.3 Classification based on complexity

Some decisions are routine, simple and easy to take. Knowledge about such decisions is very well known. The outcomes of each alternative of such decisions are well known and certain. No surprises, for example, let us assume that an operator at the quality checking department checks a product based on fixed specifications. If the product fails to meet any of the specifications, it is rejected, else accepted. In such a situation, the role of the operator is clearly laid out, the decision rules are well laid out and the best choices to take are known in advance. The information on which the decision-making is based comes from within the organization. This is an example of simple operational decision-making. Not all decisions however, are so simple and structured. In complex decisions, the process outcomes of each stage of the decision-making process are not well known. Information about the decision-making process is not known. These decisions are complex. Normally, strategic decisions are of this type.

9.2.4 Classification based on Knowledge of Outcomes

Another approach of classifying decisions is the level of knowledge of outcomes. An outcome defines what will happen, if a decision is made or course of action taken. When there is more than one alternative, the knowledge of outcome becomes important. On the

basis of the level of knowledge of outcomes, decision-making can be classified into three categories.

- **Decision under certainty:** Decision-making under certainty takes place when the outcome of each alternative is fully known. There is only one outcome for each alternative.
- **Decision under risk:** Decision-making under risk occurs when there is a possibility of multiple outcomes of each alternative and a probability of occurrence can be attached to each outcome.
- **Decision under uncertainty:** Decision-making under uncertainty takes place when there are a number of outcomes for each alternative & the probabilities of their occurrences are not known.

9.3 ROLE OF INFORMATION IN DECISION MAKING

Information plays a vital role in decision-making. Even to take very simple decisions, we need information. To understand the role played by information in decision-making, we have to understand how decisions are taken.

Decision-making is basically a process that includes the following stages:

(a) Intelligence which deals with the problem identification and the data collection on the problem.

(b) Design which deals with the generation of alternative solutions to the problem at hand.

(c) Choice which is selecting the 'best' solution from amongst the alternative solutions using some criterion.

Imagine a simple decision like the one a driver makes when he puts on the brakes to stop a speeding vehicle when he sees a child crossing the road. The driver decides on braking based on a lot of information processing that happens in his brain. At every stage of the decision-making he uses information that he captures visually. All decisions are like this.

First we get information about a problem, format it into a structure and then factor in the information about the context in which the problem has occurred. Like in the above case instead of the child being at the middle of the road and crossing it, the driver would have seen the child about to cross over with a few steps only he would probably not have braked to stop but would have slowed down, as he would have calculated that by the time the vehicle reaches the crossing stage, the child would already have passed. So if the problem was structured as 'how to not hit the child crossing the road?', and if the child was at the middle of the road, the driver would have braked but had the child been at at ninety per cent completion level of crossing the road, the driver would have only slowed down and not braked to stop. Therefore, we see that the context has a major role in the decision-making and information is

required both about the problem and about the context in which the problem occurred. The next stage for the decision maker would be to generate alternatives. In the driver's case such possible alternatives would be

- to stop by braking
- to slow down
- to take a sharp turn towards left or right to avoid the child
- press the horn so that the child crosses the road fast
- To drive the vehicle on to the footpath and out of the road to avoid collision, etc.

So the decision-maker generates these possible solutions to the problem at hand based on information about such possible solutions. Each of the alternatives represents a possible solution, which one can generate if one has information about them. In the case of the driver, obviously, he needs knowledge and information to generate these alternatives, i.e., to stop by breaking the driver would need to know that *braking stops the vehicle.* If he is unaware of this crucial information he would not have been able to generate this alternative. So information is vital for generation of alternatives. Now for the choice part also, the decision maker needs to have information about the suitability of each alternative to decide, which the 'best' is. In our example, the driver calculates the 'payoff' for each alternative based on his calculation of the outcome that again is based on information. He selects the 'best' option that solves the problem. Thus, we can see that information is the key to the decision making process, without information and the right kind of information decision-making is not possible. Information plays a crucial role in every stage of the decision-making process.

Decision-making is the most important task of managers in an organization. Therefore, to enable managers to take good quality decisions, it is very important to provide them with the right kind of information. Information management in organizations therefore assumes a special significance. In most organizations, business or otherwise, a systematic systems based method is used for information management. Systems based information management works best under a computerized environment and such computer based information management system is normally called 'Management Information Systems (MIS)', which provides the service of information supply to the managers enabling them to take informed decisions.

The following table elaborates the role of information at every stage of decision making:

Tab 9.2 Role of Information in stages of decision making

Stages of Decision-making	Role of Information
Identification and structuring of problem/opportunity	One needs information to identify a problem and put it in a structured manner. Without information about a problem or opportunity, the decision-making process does not even start.

Generation of alternatives	Information is a key ingredient in the generation of alternatives for decision-making. One has to have information about possible solutions to generate alternatives.
Choice of best alternative	Based on the information about the suitability of the alternatives, a choice is made to select the best alternative.

It may be worthwhile to mention here that MIS does not necessitates the use of computer based technology, but the use of computers and information technology makes MIS suitable for business organizations in a competitive environment as it helps to provide timely and accurate information. MIS done manually, without the help of computers is neither timely nor accurate.

9.4 ROLE OF MIS IN DECISION MAKING

Management Information System (MIS) is basically concerned with the process of collecting, processing, storing and transmitting relevant information to support the management operations in an organization. Thus, the success of decision-making, which is the heart of administrative process, is highly dependent on available information. MIS is an organized, automated, and diverse information system that gathers, stores, processes, and distributes data associated with different departments of the organization. This data is processed in various forms, such as graphs, diagrams, charts, and reports to generate accurate, relevant and valuable information for the management. This information is further communicated to the various departments to be used for decision-making and business management. MIS system provides central storage of all the business information. There are various types of MIS systems which are used to gain better understanding of the market and enterprise. MIS is used across all levels in an organization. For example, MIS provides vital information at senior levels to help make strategic decisions. At other levels, MIS observes an organization's activities and distributes information to everyone in the organization and customers. MIS is very important for every organization because it not only collects and manages information, but also represents it in various formats useful for the management to make important organizational decisions.

MIS is a system providing management with accurate and timely information. Such information is necessary to facilitate the decision-making process and enable the organizations planning, control, and operational functions to be carried out effectively. MISs increase competitiveness of the firm by reducing cost and improving processing speed. Making decisions is an important part of working in the business environment. Companies often make decisions regarding operational improvements or selecting new business opportunities for maximizing the company's profit. Companies develop a decision-making process based on individuals responsible for making decisions and the scope of the company's business operations. A useful tool for making business decisions is a management information system.

Historically, MIS was a manual process used to gather information and funnel it to individuals responsible for making decisions. MIS is useful in the area of decision making as it can monitor by itself disturbances in a system, determine a course of action and take action to get the system in control. It is also relevant in nonprogrammer decisions as it provides support by supplying information for the search, the analysis, the evaluation and the choice and implementation process of decision making. MIS in decision making as it provides information that is needed for better decision making on the issues affecting the organization regarding human and material resources.

In nutshell, Effective decision making demands accurate, timely and relevant information. MIS provides accurate and timely information necessary to facilitate the decision-making process and enable the organizations planning, control, and operational functions to be carried out effectively. MIS also plays the crucial role of providing a wide range of streamlined options from which decision-makers are able to make their preferred choices and this ensures that whatever choices are made by decision makers, the outcome, more often than not, becomes positive. This, as a matter of fact, is the reason why many decision makers tend to prefer using MIS tools when making tough business choices. MIS as renowned concept, having good decision choices guarantees viable decisions in our businesses.

Given below is a brief description of these three stages of the decision-making process and the role of MIS.

Intelligence Stage: In this stage, an information system may provide information about internal as well as external environments. Internal information is generated from the functional areas, whereas external information is collected from various sources, such as databases, newspapers, government reports, personal contacts, etc. Availability of a large amount of information makes it necessary to scan the environment and data sources to get the relevant information. Thus, information systems can be used to scan the business environment of an organization. In order to get the required information in the intelligence phase of decision-making, MIS must be designed so as to answer pre-specified as well as Ad hoc queries (unique, unscheduled, situation-specific) made by the decision-maker. In other words, information system design may have various models and a query language capability (decision support system capability).

Design Stage: At this stage, various alternatives are developed and evaluated. In the case of structured decisions, information systems can support by quantifying and automating a decision-making process.

On the other hand, for semi-structured to unstructured decisions, information systems can support such decision-making by providing:

- The ability to make Ad hoc queries for information in the organizational databases
- The ability to reach a decision in an interactive process.

Thus, information systems should be designed to incorporate various models of business operations and advanced statistical, optimization techniques, etc., so that these could be used to manipulate information already collected in the intelligence stage to develop and evaluate various alternatives.

Choice Stage: It is the choice stage in which a course of action is selected and feedback is collected on the implemented decision. Information systems can provide summarized and organized information to the decision-makers at this stage. Several models may be used to select the most appropriate alternative and thus help decision-makers select the best course of action. Information systems can also help the decision-maker monitor the successful implementation of a decision by providing feedback. During the process of decision-making, if the decision-maker chooses to return to any of the preceding stage for more information, such information support is again provided by the information system. An information system, to support the choice stage of the decision-maker, should have optimization models and suggestion models.

✦ ✦ ✦