INFORMATION SYSTEM & DATA PROCESSING

Sessional Assignment

ID:14646 **Name:** Abdul Musawer **Dated:** 08 May, 2020.

---

1) Define organization; also explain the structure of an organization by giving an example of a well known organization. (Note: every student should take the example of a different organization from another).

Ans: ORGANIZATION:

Organization can be defined as the collection of a group of people for the achievement of a certain purpose, task or objective. All the organizations have management structures that determine different relationships between different activities and members, subdivided tasks and assigned roles, responsibilities and authorities to carry out different tasks. Organizations are mostly open-system certain types that have an effect on the environment or are affected by the environment.

STRUCTURE OF AN ORGANIZATION:

On the major organizations have got four types of structures:

1. Functional Structure.
2. Divisional Structure.
    a. Divisional Structure: Market-Based Structure.
    b. Divisional Structure: Geographical Structure.
3. Matrix Structure.
4. Flatarchy

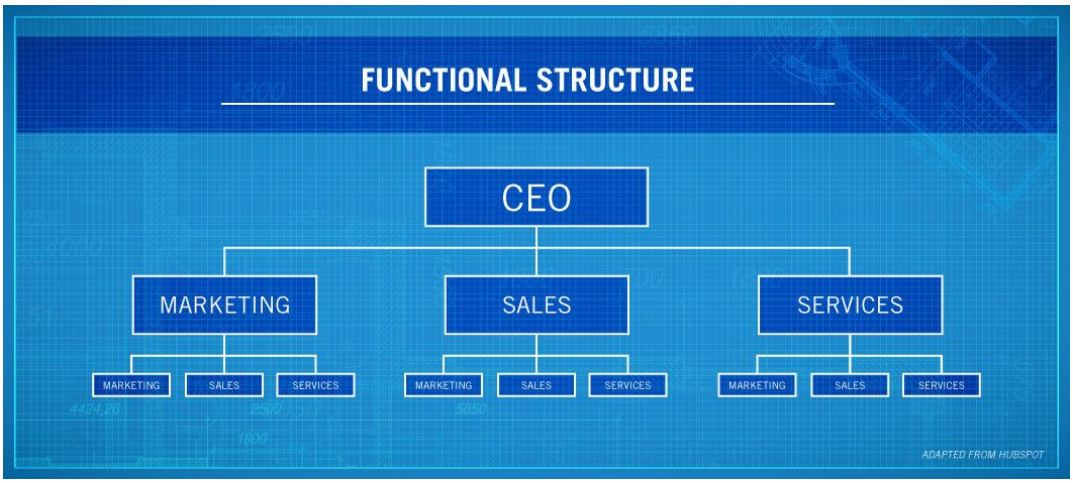
1. **FUNCTIONAL STRUCTURE:**

The functional structure is based on an organization being divided up into smaller groups with specific tasks or roles. For example, a company could have a group working in information technology, another in marketing and another in finance.

Each department has a manager or director who answers to an executive a level up in the hierarchy who may oversee multiple departments. One such example is a director of marketing who supervises the marketing department and answers to a vice president who is in charge of the marketing, finance and IT divisions.

An advantage of this structure is employees are grouped by skill set and function, allowing them to focus their collective energies on executing their roles as a department.

One of the challenges this structure presents is a lack of inter-departmental communication, with most issues and discussions taking place at the managerial level among individual departments.

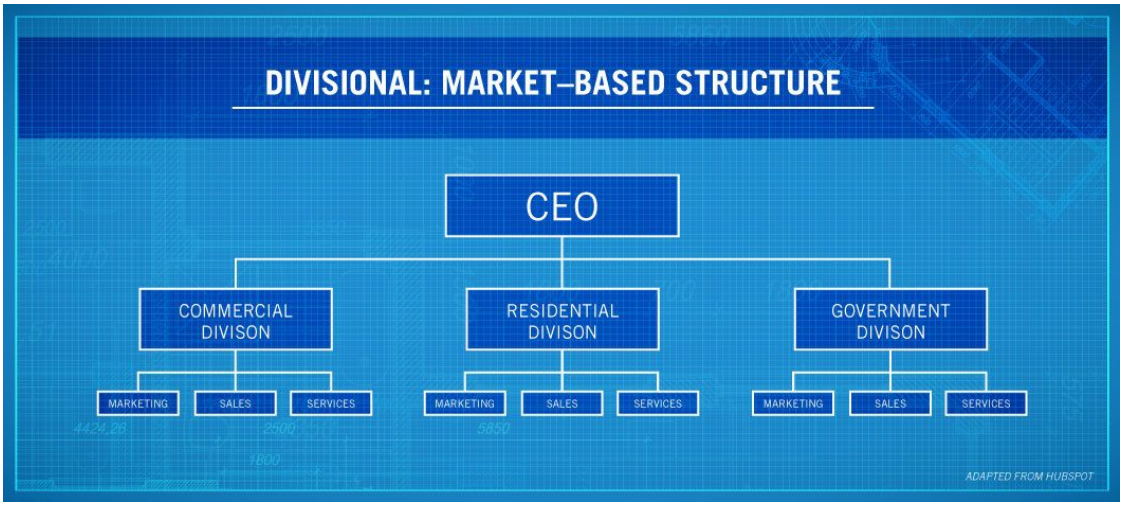


2. **DIVISIONAL STRUCTURE:**

Larger companies that operate across several horizontal objectives sometimes use a divisional organizational structure.

   a. **MARKET-BASED STRUCTURE:**

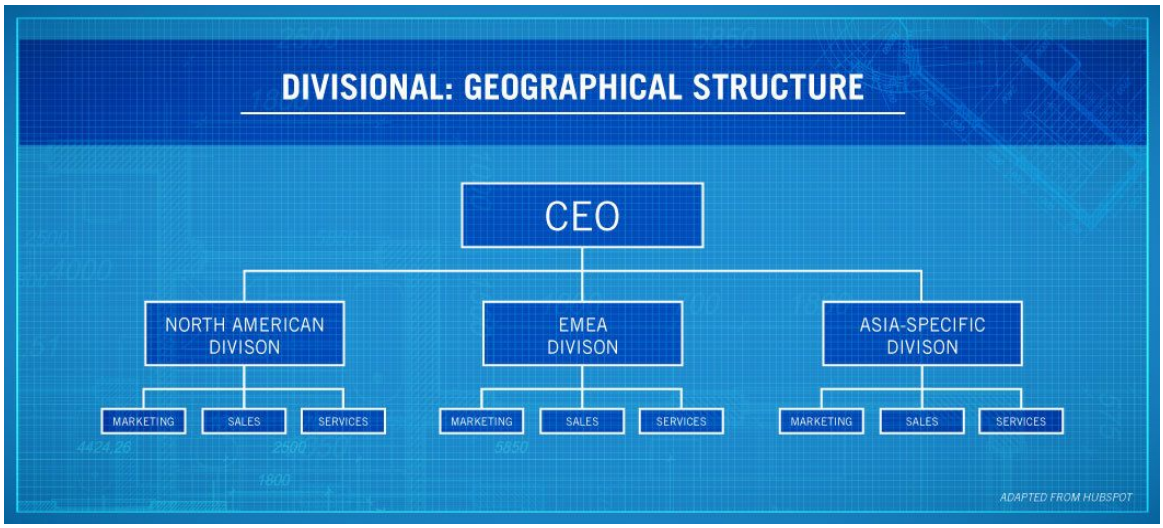
Under this structure, each division essentially operates as its own company, controlling its own resources and how much money it spends on certain projects or aspects of the division.



   b. **GEOGRAPHICAL STRUCTURE:**

This type of structure offers greater flexibility to a large company with many divisions, allowing each one to operate as its own company with one or two people reporting to the parent company's chief executive

officer or upper management staff. Instead of having all programs approved at the very top levels, those questions can be answered at the divisional level.

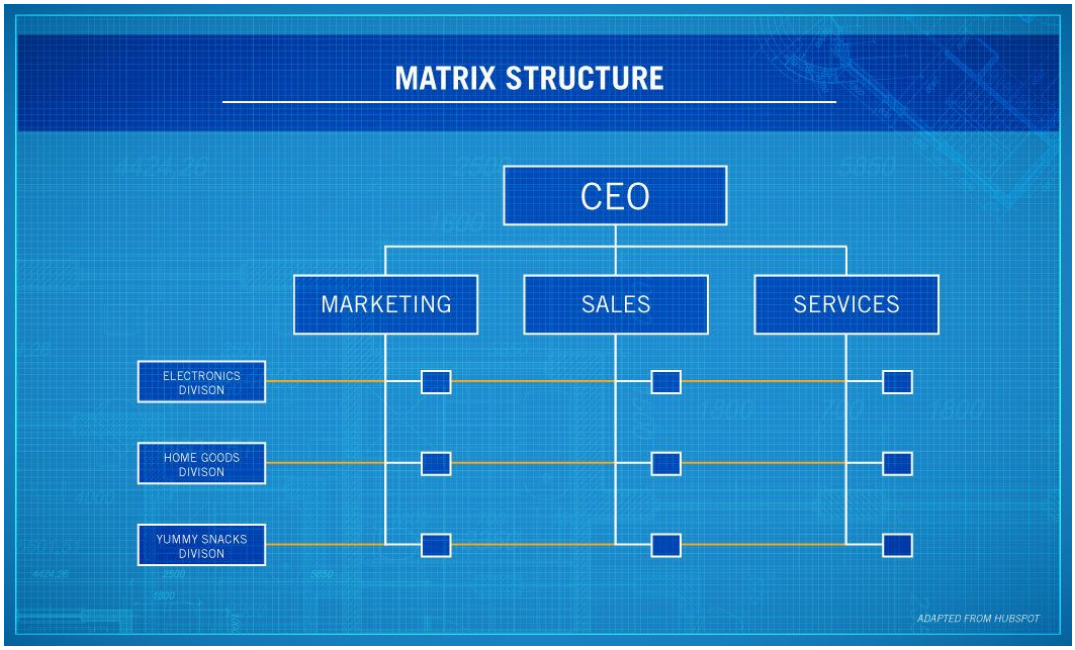


3. **MATRIX STRUCTURE:**

A hybrid organizational structure, the matrix structure is a blend of the functional organizational structure and the projectized organizational structure.

In the matrix structure, employees may report to two or more bosses depending on the situation or project. For example, under normal functional circumstances, an engineer at a large engineering firm could work for one boss, but a new project may arise where that engineer's expertise is needed. For the duration of that project, the employee would also report to that project's manager, as well as his or her boss for all other daily tasks.

Advantages of this structure is that employees can share their knowledge across the different functional divisions, allowing for better communication and understanding of each function's role. And by working across functions, employees can broaden their skills and knowledge, leading to professional growth within the company.



4. **FLATARCHY:**

While the previous three types of organizational structures may work for some organizations, another hybrid organizational structure may be better for startups or small companies.

Blending a functional structure and a flat structure results in a flatarchy organizational structure, which allows for more decision making among the levels of an organization and, overall, flattens out the vertical appearance of a hierarchy.

The best example of this structure within a company is if the organization has an internal incubator or innovation program. Within this system, the company can operate in an existing structure, but employees at any level are encouraged to suggest ideas and run with them, potentially creating new flat teams.

EXAMPLE OF GSK (GlaxoSmithKline) PHARMACEUTICALS:

_____

2) Explain System Development Life Cycle; also explain different types of system development life cycle.

Ans: SYSTEM DEVELOPMENT LIFE CYCLE:

The systems development life cycle (SDLC) is a conceptual model used in project management that describes the stages involved in an information system development project, from an initial feasibility study through maintenance of the completed application. SDLC can apply to technical and non-technical systems. In most use cases, a system is an IT technology such as hardware and software. Project and program managers typically take part in SDLC, along with system and software engineers, development teams and end-users.

Every hardware or software system will go through a development process which can be thought as an iterative process with multiple steps. SDLC is used to give a rigid structure and framework to define the phases and steps involved in the development of a system.

SDLC MODELS:

There are many models but the most common models used are:

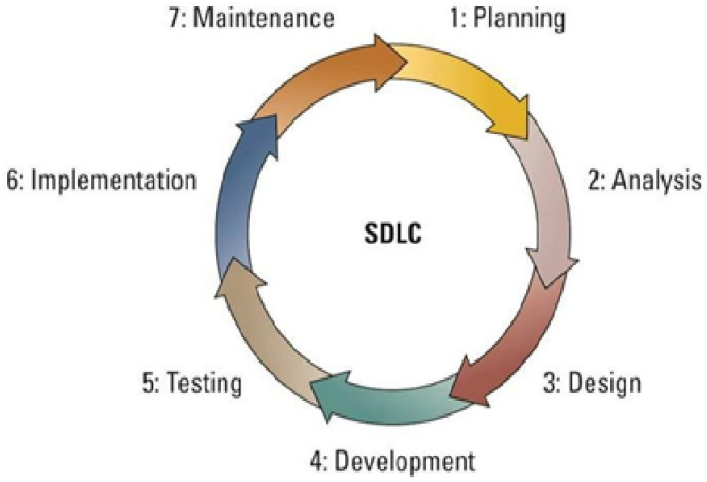
1. Waterfall Model.
2. Rapid Application Development (RAD).
3. Joint Application Development (JAD).
4. Fountain Model.
5. Spiral Model.
6. Incremental Model.
7. Agile (Mostly used in Market for software development).

STEPS IN SDLC:

Generally in the majority of models the following steps are followed:

1. **Analysis:** The existing system is evaluated. Deficiencies are identified. This can be done by interviewing users of the system and consulting with support personnel.
2. **Plan and requirements:** The new system requirements are defined. In particular, the deficiencies in the existing system must be addressed with specific proposals for improvement. Other factors defined include needed features, functions and capabilities.
3. **Design:** The proposed system is designed. Plans are laid out concerning the physical construction, hardware, operating systems, programming, communications and security issues.
4. **Development:** The new system is developed. The new components and programs must be obtained and installed. Users of the system must be trained in its use.
5. **Testing:** All aspects of performance must be tested. If necessary, adjustments must be made at this stage. Tests performed by quality assurance (QA) teams may include systems integration and system testing.

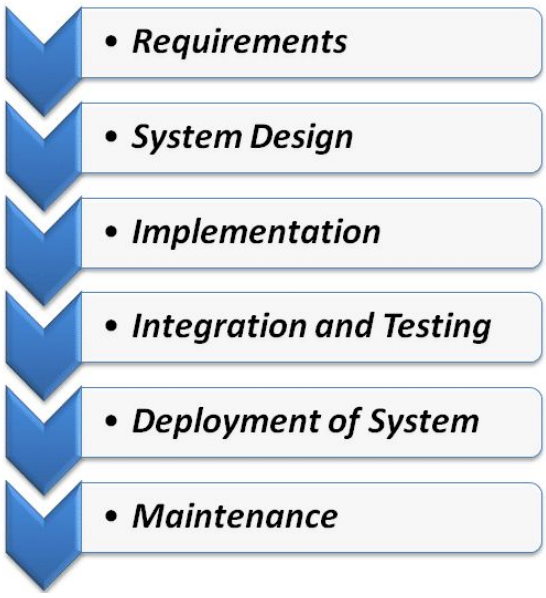
6. **Deployment:** The system is incorporated in a production environment. This can be done in various ways. The new system can be phased in, according to application or location, and the old system gradually replaced. In some cases, it may be more cost-effective to shut down the old system and implement the new system all at once.
7. **Upkeep and maintenance:** This step involves changing and updating the system once it is in place. Hardware or software may need to be upgraded, replaced or changed in some way to better fit the needs of the end-users continuously. Users of the system should be kept up-to-date concerning the latest modifications and procedures.



1. WATERFALL MODEL:

The 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. This means that any phase in the development process begins only if the previous phase is complete.

There is no going back to the earlier phase as you have completed. This model was the base of SDLC and is not probably used today.



2. AGILE MODEL:

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software products. Agile Methods break the product into small incremental builds. These builds are provided in iterations.

Agile model is the combination of iterative and incremental process models. Steps involved in agile SDLC models are:

- Requirement gathering
- Requirement Analysis
- Design
- Coding
- Unit testing
- Acceptance testing

Principles of Agile model:

- To establish close contact with the customer during development and to gain a clear understanding of various requirements, each Agile project usually includes a customer representative on the team. At the end of each iteration stakeholders and the customer representative review the progress made and re-evaluate the requirements.
- The Agile model relies on working software deployment rather than comprehensive documentation.
- Frequent delivery of incremental versions of the software to the customer representative in intervals of a few weeks.
- Requirement change requests from the customer are encouraged and efficiently incorporated.
- It emphasizes on having efficient team members and enhancing communications among them is given more importance. It is realized that enhanced communication among the development team members can be achieved through face-to-face communication rather than through the exchange of formal documents.
- It is recommended that the development team size should be kept small (5 to 9 peoples) to help the team members meaningfully engage in face-to-face communication and have a collaborative work environment.
- Agile development processes usually deploy Pair Programming. In Pair programming, two programmers work together at one work-station. One does coding while the other reviews the code as it is typed in. The two programmers switch their roles every hour or so.

3. RAPID APPLICATION DEVELOPMENT (RAD):

RAD model is Rapid Application Development model. It is a type of incremental model. In the RAD model the components or functions are developed in parallel as if they were mini projects. The developments are time boxed, delivered and then assembled into a working prototype.

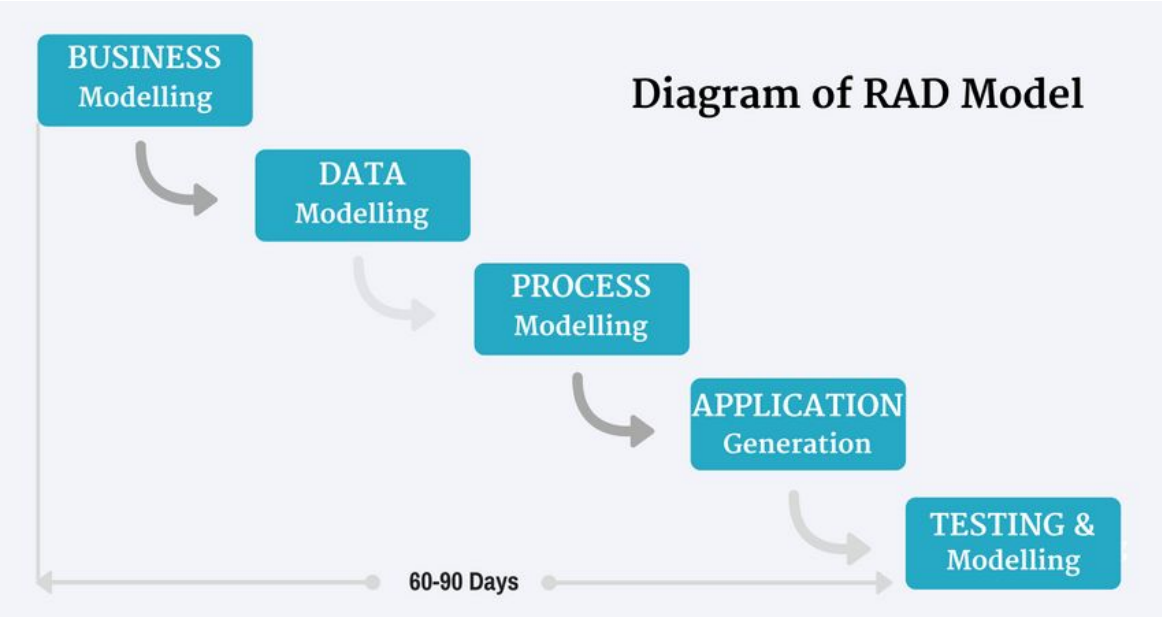
Rapid Application Development focuses on gathering customer requirements through workshops or focus groups, early testing of the prototypes by the customer using iterative concepts, reuse of the existing prototypes (components), continuous integration and rapid delivery.

RAD Model - Application

RAD models can be applied successfully to the projects in which clear modularization is possible. If the project cannot be broken into modules, RAD may fail.

The following pointers describe the typical scenarios where RAD can be used −

- RAD should be used only when a system can be modularized to be delivered in an incremental manner.
- It should be used if there is a high availability of designers for modeling.
- It should be used only if the budget permits use of automated code generating tools.
- RAD SDLC model should be chosen only if domain experts are available with relevant business knowledge.
- Should be used where the requirements change during the project and working prototypes are to be presented to customers in small iterations of 2-3 months.



_____

3) Explain Incremental model and Spiral; also explain the main difference between spiral and incremental model.

Ans: 1. <u>INCREMENTAL MODEL:</u>

Incremental Model is a process of software development where requirements are divided into multiple standalone modules of the software development cycle. In this model, each module goes through the requirements, design, implementation and testing phases. Every subsequent release of the module adds function to the previous release. The process continues until the complete system is achieved.

<u>The various phases of incremental model are as follows:</u>

1. Requirement analysis: In the first phase of the incremental model, the product analysis expertise identifies the requirements. And the system functional requirements are understood by the requirement analysis team. To develop the software under the incremental model, this phase performs a crucial role.

2. Design & Development: In this phase of the Incremental model of SDLC, the design of the system functionality and the development method are finished with success. When software develops new practicality, the incremental model uses style and development phase.

3. Testing: In the incremental model, the testing phase checks the performance of each existing function as well as additional functionality. In the testing phase, the various methods are used to test the behavior of each task.

4. Implementation: Implementation phase enables the coding phase of the development system. It involves the final coding that is in the designing and development phase and tests the functionality in the testing phase. After completion of this phase, the number of the product working is enhanced and upgraded up to the final system product

Advantages of Incremental model:

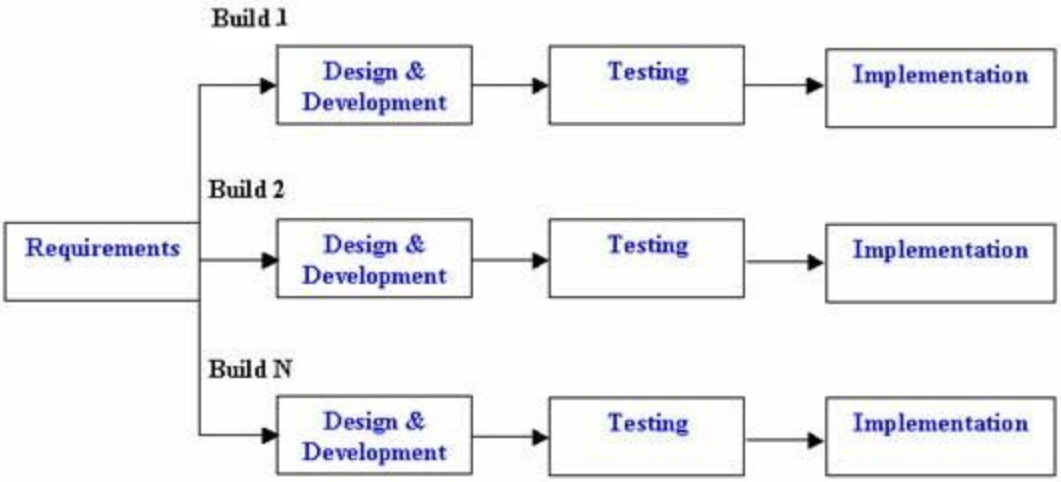
- Generates working software quickly and early during the software life cycle.
- This model is more flexible – less costly to change scope and requirements.
- It is easier to test and debug during a smaller iteration.
- In this model customers can respond to each build.
- Lowers initial delivery cost.
- Easier to manage risk because risky pieces are identified and handled during it'd iteration.

Disadvantages of Incremental model:

- Needs good planning and design.
- Needs a clear and complete definition of the whole system before it can be broken down and built incrementally.
- Total cost is higher than the waterfall.

When to use the Incremental model:

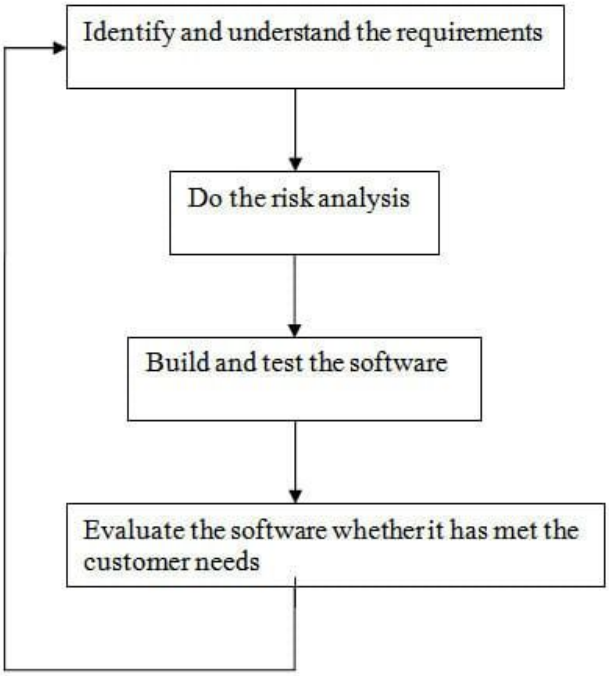
- This model can be used when the requirements of the complete system are clearly defined and understood.
- Major requirements must be defined; however, some details can evolve with time.
- There is a need to get a product to the market early.
- A new technology is being used
- Resources with needed skill set are not available
- There are some high risk features and goals.



Incremental Life Cycle Model

2. SPIRAL MODEL:

The spiral model is a combination of sequential and prototype models. This model is best used for large projects which involve continuous enhancements. There are specific activities that are done in one iteration (spiral) where the output is a small prototype of the large software. The same activities are then repeated for all the spirals until the entire software is built.



A spiral model has 4 phases:

1. Planning phase
2. Risk analysis phase
3. Engineering phase
4. Evaluation phase.

1. PLANNING PHASE:

ACTIVITIES PERFORMED:

- Requirements are studied and gathered.
- Feasibility study.
- Reviews and walkthroughs to streamline the requirements.
-

DELIVERABLES/ OUTPUT:

- Requirements understanding document
- Finalized list of requirements.

2. **RISK ANALYSIS PHASE:**

ACTIVITIES PERFORMED:

- Requirements are studied and brainstorming sessions are done to identify the potential risks.
- Once the risks are identified , risk mitigation strategy is planned and finalized.

DELIVERABLES/ OUTPUT:

- Document which highlights all the risks and its mitigation plans.

3. **ENGINEERING PHASE:**

ACTIVITIES PERFORMED:

- Actual development and testing if the software takes place in this phase

DELIVERABLES/ OUTPUT:

- Code
- Test cases and test results
- Test summary report and defect report.

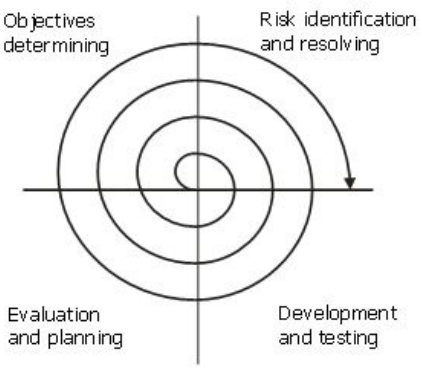
4. **EVALUATION PHASE:**

ACTIVITIES PERFORMED:

- Customers evaluate the software and provide their feedback and approval

DELIVERABLES/ OUTPUT:

- Features implemented document



DIFFERENCE BETWEEN INCREMENTAL MODEL AND SPIRAL MODEL:

Spiral and incremental model can be easily understand through the following table:

PROPERTIES OF MODEL	INCREMENTAL MODEL	SPIRAL MODEL
Planning in early stage	Yes	Yes
Returning to an earlier phase	Yes	Yes
Handle Large-Project	Not Appropriate	Appropriate
Detailed Documentation	Yes but not much	Yes
Cost	Low	Expensive
Requirement Specifications	Beginning	Beginning
Flexibility to change	Easy	Easy
User Involvement	Intermediate	High
Maintenance	Promotes Maintainability	Typical
Duration	Very Long	Long
Risk Involvement	Low	Medium to high risk
Framework Type	Linear + Iterative	Linear + Iterative
Testing	After every iteration	At the end of the engineering phase
Overlapping Phases	Yes (As parallel development is there)	No
Maintenance	Maintainable	Yes
Re-usability	To some extent	To some extent
Time-Frame	Long	Long
Team size	Not Large Team	Large Team

_____