Information Systems and E-Commerce

UNIT – III

Database

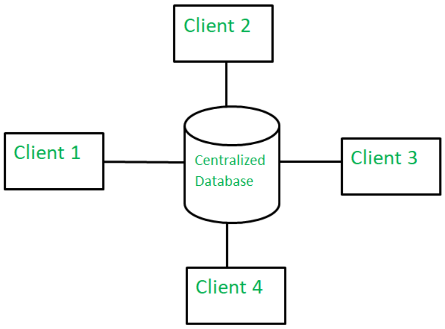
A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS).

DBMS

Database Management Systems (DBMS) are software systems used to store, retrieve, and run queries on data. A DBMS serves as an interface between an end-user and a database, allowing users to create, read, update, and delete data in the database.

Centralized and Distributed Database Systems

Centralized database management system is the system in which all the data is stored and managed in a single unit. This is also known as central computer database system. This system is mostly used in an organization, in any Business Company or in institution to centralize the tasks.



A distributed database is basically a database that is not limited to one system, it is spread over different sites, i.e., on multiple computers or over a network of computers. A distributed database system is located on various sites that don't share physical components.

Planning, and designing of Information Systems

Planning and Designing of Information Systems provides a theoretical base and a practical method of executing the planning of computerized information systems and the planning and design of individual applications.

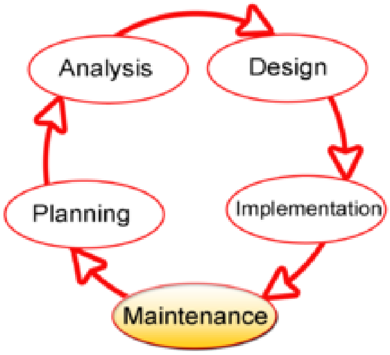
System Flow Charts

A system flowchart shows the path taken by data in a system and the decisions made during different levels. Different symbols are strung together to show data flow, including what happens to data and where it goes.

Traditional System Development Life Cycle

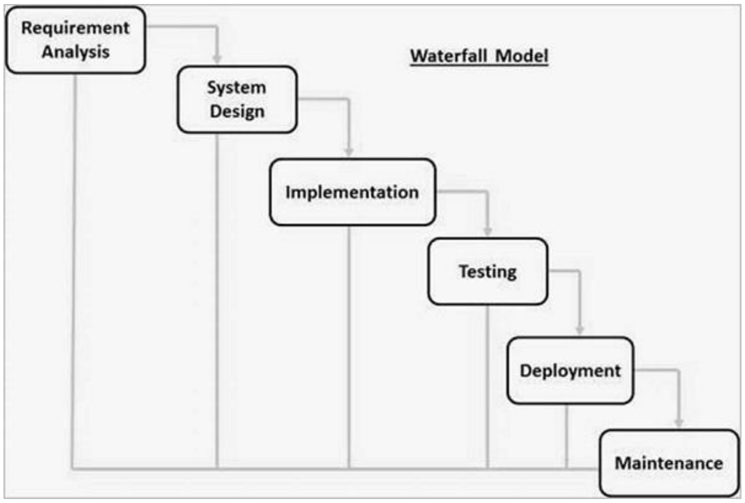
Systems development life cycle phases include planning, system analysis, system design, development, implementation, integration and testing, and operations and maintenance.

In systems engineering, information systems and software engineering, the systems development life cycle (SDLC), also referred to as the application development life-cycle, is a process for planning, creating, testing, and deploying an information system.



1. Waterfall

The waterfall approach is one of the oldest SDLC models, but it has fallen out of favor in recent years. This model involves a rigid structure that demands all system requirements be defined at the very start of a project. Only then can the design and development stages begin.
Once development is complete, the product is tested against the initial requirements and rework is assigned. Companies in the software industry typically need more flexibility that what the waterfall methodology offers, but it still remains a strong solution for certain types of projects especially government contractors.



2.Prototyping

In the prototyping methodology, the design team's focus is to produce an early model of the new system, software, or application. This prototype won't have full functionality or be thoroughly tested, but it will give external customers a sense of what's to come. Then, feedback can be gathered and implemented throughout the rest of the SDLC phases.
The prototyping approach works well for companies in emerging industries or new technologies.

Advantages of Prototyping Model

1. It provides higher customer satisfaction and get reviews from customers.
2. Requirements can change easily according to the customer. It means this model has high adaptability.
3. Errors can be removed so fast and early hence it saves a lot of cost.
4. It gives high flexibility.
5. Prototype model increases the involvement of customers.
6. Developers can identify the missing functionality very easily.
7. This model helps to reduce the risks related to the software.

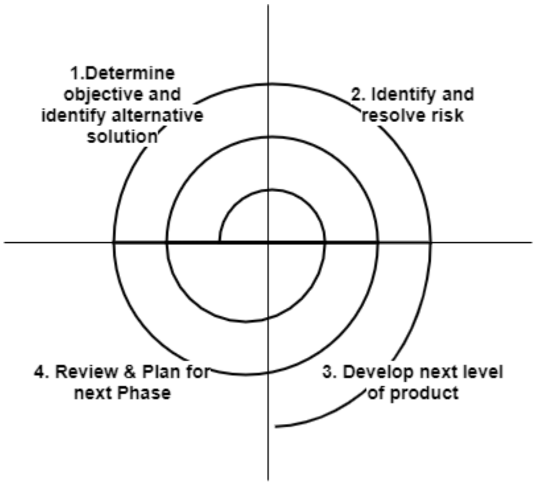
Disadvantages of Prototyping Model

1. This model can be more costly.
2. Prototype model can be more complex if a prototype is reviewed by the customers again and again.
3. It does not provide clear documents.
4. This model can take more time to develop a software product.
5. If a customer refuses to accept the prototype, then there may be wastage of a lot of cost and developer's efforts.

3. Spiral

The spiral methodology allows teams to adopt multiple SDLC models based on the risk patterns of the given project. A blend of the iterative and waterfall approaches, the challenge with the spiral model knows when the right moment to move onto the next phase is.

Business that aren't sure about their requirements or expect major edits during their mid to high-risk project can benefit from the scalability of this methodology.


1.Determine objective and identify alternative solution
2. Identify and resolve risk
3. Develop next level of product
4. Review & Plan for next Phase

Advantages of Model:

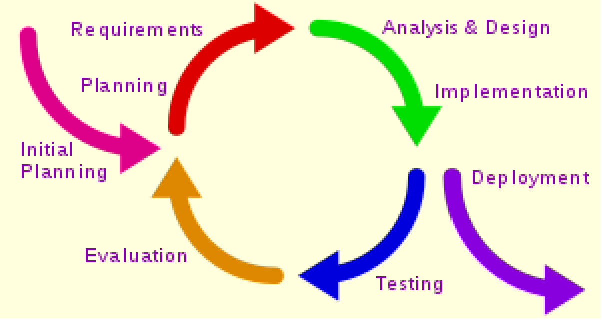
- For Variety of Situations model is very flexible such as reuse, component-based development, and prototyping.
- For every spiral risk analysis is performed as it is an integral part of the life cycle which leads to an increase in confidence in the project.

Disadvantages of Spiral Model:

- Knowledgeable and experienced staff is required.
- For developers, it is a more complicated and risk-driven model.

4. Iterative

The iterative methodology takes the waterfall model and cycles through it several times in small increments. Rather than stretching the entire project across the phases of the SDLC, each step is turned into several mini-projects that can add value as the product evolves.



The iterative approach shares many of the same goals as the agile model, except external customers are less involved and the scope of each increment is normally fixed.

Modern System Development Methods

The system development life cycle, known as the SDLC, is the industry-standard approach to managing phases of an engineering project. Think of it as the equivalent to the scientific method for software development and other IT initiatives. The common breakdown of the SDLC includes seven phases that trace a product or project from a planned idea to its final release into operation and maintenance.

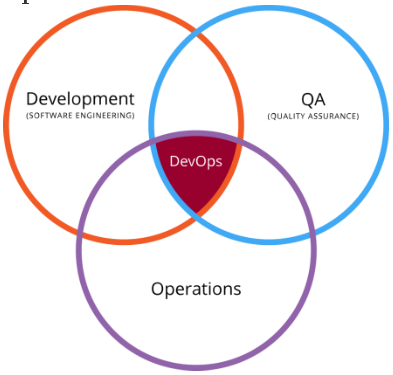
There is flexibility within the SDLC. In recent decades a number of different models and methods have gained popularity. Consider one of the following six approaches when establishing an SDLC in your organization.

1. Agile

The agile methodology is the opposite of the waterfall approach. Rather than treating requirements, design, and testing as large sequential steps, an agile model makes them all ongoing processes that require involvement from developers, management, and customers. Work is typically broken into 2-4 week segments known as “sprints,” in which the responsible teams tackle the major needs of their customers and perform testing as they go. Agile tends to work well in small organizations, especially startups, where speed and flexibility is essential.

2.DevOps

DevOps is one of the newest SDLC methodologies and is being adopted by many software companies and IT organizations. As its name suggests, the premise of DevOps is to bring development teams together with operational teams in order to streamline delivery and support.



The advantages of such an approach are that changes become more fluid, while organizational risk is reduced. Teams must have flexible resources in order for a DevOps arrangement to succeed.

System Development Life Cycle

Following are the seven phases of the SDLC

1. Planning

This is the first phase in the systems development process. It identifies whether or not there is the need for a new system to achieve a business's strategic objectives. This is a preliminary plan (or a feasibility study) for a company's business initiative to acquire the resources to build on an infrastructure to modify or improve a service. The company might be trying to meet or exceed expectations for their employees, customers and stakeholders too. The purpose of this step is to find out the scope of the problem and determine solutions. Resources, costs, time, benefits and other items should be considered at this stage.

2. Systems Analysis and Requirements

The second phase is where businesses will work on the source of their problem or the need for a change. In the event of a problem, possible solutions are submitted and analyzed to identify the best fit for the ultimate goal(s) of the project. This is where teams consider the functional requirements of the project or solution. It is also where system analysis takes place—or analyzing the needs of the end users to ensure the new system can meet their expectations. Systems analysis is vital in determining what a business's needs are, as well as how they can be met, who will be responsible for individual pieces of the project, and what sort of timeline should be expected.

There are several tools businesses can use that are specific to the second phase. They include:

- CASE (Computer Aided Systems/Software Engineering)
- Requirements gathering
- Structured analysis

3. Systems Design

The third phase describes, in detail, the necessary specifications, features and operations that will satisfy the functional requirements of the proposed system which will be in place. This is the step for end users to discuss and determine their specific business information needs for the proposed system. It's during this phase that they will consider the essential components (hardware and/or software) structure (networking capabilities), processing and procedures for the system to accomplish its objectives.

4. Development

The fourth phase is when the real work begins—in particular, when a programmer, network engineer and/or database developer are brought on to do the major work on the project. This work includes using a flow chart to ensure that the process of the system is properly organized. The development phase marks the end of the initial section of the process. Additionally, this phase signifies the start of production. The development stage is also characterized by instillation and change. Focusing on training can be a huge benefit during this phase.

5. Integration and Testing

The fifth phase involves systems integration and system testing (of programs and procedures)—normally carried out by a Quality Assurance (QA) professional—to determine if the proposed design meets the initial set of business goals. Testing may be repeated, specifically to check for errors, bugs and interoperability. This testing will be performed until the end user finds it acceptable. Another part of this phase is verification and validation, both of which will help ensure the program's successful completion.

6. Implementation

The sixth phase is when the majority of the code for the program is written. Additionally, this phase involves the actual installation of the newly-developed system. This step puts the project into production by moving the data and components from the old system and placing them in the new system via a direct cutover. While this can be a risky (and complicated) move, the cutover typically happens during off-peak hours, thus minimizing the risk. Both system analysts and end-users should now see the realization of the project that has implemented changes.

7. Operations and Maintenance

The seventh and final phase involves maintenance and regular required updates. This step is when end users can fine-tune the system, if they wish, to boost performance, add new capabilities or meet additional user requirements.

Importance of the SDLC

If a business determines a change is needed during any phase of the SDLC, the company might have to proceed through all the above life cycle phases again. The life cycle approach of any project is a time-consuming process. Even though some steps are more difficult than others, none are to be overlooked. An oversight could prevent the entire system from functioning as planned.
Systems development specialists at Innovative Architects possess extensive experience in managing these types of projects. If you have a situation at your organization and you think a customized software solution may be what you need, contact us today. Consultants at Innovative Architects will be able to quickly guide you through each of these steps, ensuring you can have your new system online as soon as possible.

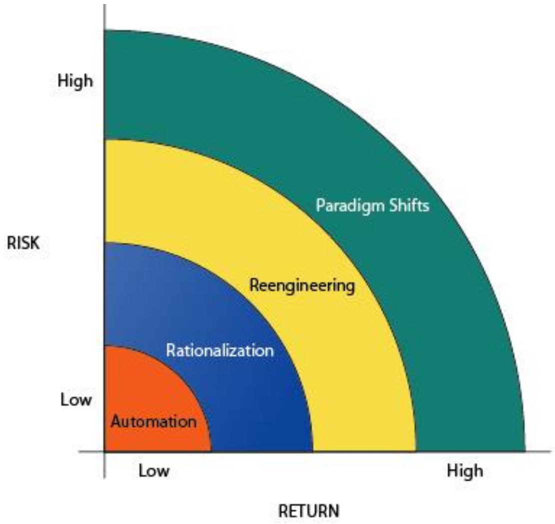
SYSTEM AS PLANNED ORGANISATIONAL CHANGE

The process of building a new information system is one kind of planned organizational change. System builders must understand how a system will affect the organization as a whole. They must consider how the nature of work groups will change and how much change is needed.

Four kinds of structural organizational change which are enabled by information technology are:

- Automation: Enables employees to perform their tasks more efficiently and effectively

- Rationalization of procedures: The streamlining of standard operation procedures, eliminating obvious bottlenecks, so that automation can make operating procedures more efficient
- Business process reengineering: Analyzes, simplifieds, and redesigns business processes with a mind to radically reduce business costs
- Paradigm shift: A radical reconceptualization of the nature of the business and the nature of the organization



Organization Change Carries Risks And Rewards

The most common forms of organizational change are automation and rationalization. These relatively slow-moving and slow-changing strategies present modest returns but little risk. Faster and more comprehensive change—such as reengineering and paradigm shifts—carries high rewards but offers substantial chances of failure.