CHAPTER 5

Acquisition, Development and Implementation of Information Systems

Basic Concepts


2. System Development: It refers to the process of examining a business situation with the intent of improving it through better procedures and methods. System development can generally be thought of as having two major components:
   - System Analysis, which is the process of gathering and interpreting facts, diagnosing problems, and using the information to recommend improvements to the system.
   - System Design, which is the process of planning and structuring a new business system or one to replace or complement an existing system.

3. Achieving System Development Objectives: Achieving the objectives of the system development is essential but many times, such objectives are not achieved as desired. An analysis on 'Why organizations fail to achieve their systems development objectives' reveals bottlenecks. Some of the most notable ones are given as follows:
   (i) User Related Issues: It refers to those issues where user/customer is reckoned as the primary agent. Some of the aspects with regard to this problem are: Shifting User Needs, Resistance to Change, Lack of User Participation, Inadequate Testing and User Training.
   (ii) Developer Related Issues: It refers to the issues and challenges with regard to the developers. Some of the critical bottlenecks are: Lack of Standard Project Management and System Development Methodologies, Overworked or Under-Trained Development Staff.
   (iii) Management Related Issues: It refers to the bottlenecks with regard to organizational set up, administrative and overall management to accomplish the system development goals. Some of such bottlenecks are: Lack of Senior Management Support and Involvement, Non-development of Strategic Systems.
   (iv) New Technologies: When an organization tries to create a competitive advantage by applying advance technologies, it generally finds that attaining system development objectives is more difficult because personnel are not familiar with the technology.
4. **Accountants’ Involvement in Development Work:** An accountant can help in various related aspects during system development; some of them are as follows:

   (i) **Return on Investment (referred as RoI):** This defines the return, an entity shall earn on a particular investment i.e. capital expenditure. For this analysis, following data needs to be generated.

      (a) **Cost:** This includes estimates for typical costs involved in the development, which are *Development Costs, Operating Costs, and Intangible Costs.*

      (b) **Benefits:** The benefits, which result from developing new or improved information systems that can be subdivided into tangible and intangible benefits.

   (ii) **Computing Cost of IT Implementation and Cost Benefit Analysis:** For analysis of RoI, accountants need the costs and returns from the system development efforts.

5. **Systems Development Methodology:** A System Development methodology is a formalized, standardized, well-organized and documented set of activities used to manage a system development project. It refers to the framework that is used to structure, plan and control the process of developing an information system. Major approaches are:

   - **Waterfall Model:** This is a traditional development approach in which each phase is carried out in sequence or linear fashion. These phases include Requirements Analysis, Specifications and Design Requirements, Coding, Final Testing, and Release.

   - **The Prototyping Model:** The goal of prototyping approach is to develop a small or pilot version called a Prototype of a part or all of a system. A Prototype is a usable system or system component that is built quickly and at a lesser cost, and with the intention of modifying/replicating/expanding or even replacing it by a full-scale and fully operational system. As users work with the prototype, they learn about the system criticalities and make suggestions about ways to manage it. These suggestions are then incorporated to improve the prototype, which is also used and evaluated. Finally, when a prototype is developed that satisfies all user requirements, either it is refined and turned into the final system or if it is not suitable (economically or functionally), it is scrapped. If it is scrapped, the knowledge gained from building the prototype is used to develop the real system.

   - **The Incremental Model:** The Incremental model is a method of software development where the model is designed, implemented and tested incrementally (a little more is added each time) until the product is finished. The product is defined as finished when it satisfies all of its requirements. This model combines the elements of the waterfall model with the iterative philosophy of prototyping.

   - **Spiral Model:** The Spiral model is a software development process combining elements of both design and prototyping-in-stages. It tries to combine advantages of top-down and bottom-up concepts. It combines the features of the prototyping model and the waterfall model. The spiral model is intended for large, expensive and complicated projects. Game development is a main area where the spiral model is used and needed, that is because of the size and the constantly shifting goals of those large projects.
• **Rapid Application Development (RAD) Model:** Rapid Application Development (RAD) refers to a type of software development methodology, which uses minimal planning in favor of rapid prototyping. The planning of software developed using RAD is interleaved with writing the software itself. The lack of extensive pre-planning generally allows software to be written much faster, and makes it easier to change requirements.

• **Agile Model:** This is an organized set of software development methodologies based on the iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery; time boxed iterative approach and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen interactions throughout the development life cycle.

6. **System Development Life Cycle (SDLC):** SDLC provides system designers and developers to follow a sequence of activities. It consists of a generic sequence of steps or phases in which each phase of the SDLC uses the results of the previous one. These phases are given as under:

<table>
<thead>
<tr>
<th>S. No.</th>
<th>PHASE NAME</th>
<th>NATURE OF ACTIVITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Preliminary Investigation</td>
<td>Determining and evaluating the strategic feasibility of the system and ensure that the solution fits the business strategy.</td>
</tr>
<tr>
<td>2.</td>
<td>Systems Requirements Analysis</td>
<td>Analyzing the typical system requirements, in view of its functionalities, deliverables etc.</td>
</tr>
<tr>
<td>3.</td>
<td>Systems Design</td>
<td>Designing the system in terms of user interface, data storage and data processing functions on the basis of the requirement phase by developing the system flowcharts, system and data flow diagrams, screens and reports.</td>
</tr>
<tr>
<td>4.</td>
<td>Systems Development</td>
<td>Developing the system as per the system designed to fulfill of requirements to the satisfaction of all stakeholders.</td>
</tr>
<tr>
<td>5.</td>
<td>Systems Testing</td>
<td>Requisite testing to ensure valid and reliable implementation.</td>
</tr>
<tr>
<td>6.</td>
<td>Systems Implementation</td>
<td>Operationalization of the developed system for acceptance by management and user before migration of the system to live environment and data conversion from legacy system to the new system.</td>
</tr>
</tbody>
</table>
5.4 Information Systems Control and Audit

7. Preliminary Investigation: It is predominantly aimed to determine and analyze the strategic benefits in implementing the system through evaluation and quantification of - productivity gains; future cost avoidance; cost savings, and Intangible benefits like improvement in morale of employees. The deliverable of the preliminary investigation includes a report including feasibility study observations. Major activities are:

(i) Identification of Problem: The first step in an application development is to define the problem clearly and precisely, which is done only after the critical study of the existing system and several rounds of discussions with the user group.

(ii) Identification of Objectives: After the identification of the problem, it is easy to work out and precisely specify the objectives of the proposed solution.

(iii) Delineation of Scope: The scope of a solution defines its typical boundaries. It should be clear and comprehensible to the user management stating the extent and what will be addressed by the solution and what will not. The typical scope determination may be performed on the following dimensions: Functionality Requirements, Data to be Processed, Control Requirements, Performance Requirements, Constraints, Interfaces, and Reliability requirements.

Two primary methods with the help of which scope of the project can be analyzed are: Reviewing Internal Documents and Conducting Interviews.

(iv) Feasibility Study: After possible solution options are identified, project feasibility i.e. the likelihood that these systems can be implemented in the organization is determined. The Feasibility Study of a system is evaluated under following dimensions: Technical, Financial, Economic, Schedule/Time, Resources, Operational, Behavioral and Legal

(v) Reporting Results to Management: After the analyst articulates the problem, defines it along with its scope, s/he provides one or more solution alternatives, estimates the cost and benefits of each alternative and reports these results to management.

8. System Requirements Analysis: This phase includes a thorough and detailed understanding of the current system, identification of the areas that need modification to solve the problem, determination of user/managerial requirements and to have fair idea about various systems development tools. Major deliverable is Systems Requirements Specification (SRS).

A generic set of process is described as follows:

(i) Fact Finding: Every system is built to meet some set of needs; for example, the need of the organization for lower operational costs, better information for managers, smooth operations for users or better levels of services to customers. Various fact-finding techniques/tools are: Documents, Questionnaires, Interviews and Observation.
(ii) **Analysis of the Present System**: Detailed investigation of the present system involves collecting, organizing and evaluating facts about the system and the environment in which it operates. The following areas should be studied in depth: **Reviewing Historical Aspects, Analyzing Inputs, Reviewing Data Files, Reviewing Methods, Procedures and Data Communications, Analyzing Outputs, Reviewing Internal Controls, Modeling the Existing System, Undertaking Overall Analysis of the Existing system**.

(iii) **System Analysis of Proposed Systems**: After a thorough analysis of each functional area of the present information system, the proposed system specifications must be clearly defined, which are determined from the desired objectives set forth at the first stage of the study.

(iv) **System Development Tools**: Many tools and techniques have been developed to improve current information systems and to develop new ones. Major tools used for system development specification or representations can be classified into four categories based on the systems features. These are: **System Components and Flows, User Interface, Data Attributes and Relationships, Detailed System Processes**.

Some popular tools are: **Structured English, Flowcharts, Data Flow Diagrams, Decision Tree, Decision Table, CASE Tools, System Components Matrix, Data Dictionary, User Interface Layout and Forms**.

(v) **Systems Specification**: At the end of the analysis phase, the systems analyst prepares a document called Systems Requirement Specifications (SRS).

(vi) **Roles Involved in SDLC**: A variety of tasks during the SDLC are performed by special teams/committees/individuals based on requisite expertise as well as skills. Some of the generic roles are: **Steering Committee, Project Manager, Project Leader, Systems Analyst / Business Analyst, Module Leader/Team Leader, Programmer/Developers, Database Administrator, Quality Assurance, Testers, Domain Specialist, IS Auditor**.

9. **Systems Design**: The key objective of this phase to design an Information System that best satisfies users/managerial requirements. Design phase documents/deliverables include a ‘blueprint’ for the design with the necessary specifications for hardware, software, people and data resources. Major activities are as follows:

(a) **Architectural Design**: Architectural design deals with the organization of applications in terms of hierarchy of modules and sub-modules. At this stage, we identify major modules; functions and scope of each module; interface features of each module; modules that each module can call directly or indirectly and Data received from / sent to / modified in other modules.

(b) **Design of Data/Information flow**: The design of the data and information flow is a major step in the conceptual design of the new system. In designing the data / information flow for the proposed system, inputs required are - existing data / information flows, problems with the present system, and objective of the new system.
5.6 Information Systems Control and Audit

(c) Design of Database: Design of the database involves determining its scope ranging from local to global structure. The scope is decided on the basis of interdependence among organizational units. The design of the database involves four major activities: Conceptual Modeling, Data Modeling, Storage Structure Design and Physical Layout Design.

(d) User Interface Design: It involves determining the ways in which users will interact with a system. The points that need to be considered while designing the user interface are: Source documents to capture raw data, hard-copy output reports, Screen layouts for dedicated source-document input, Inquiry screens for database interrogation, Graphic and color displays, and Requirements for special Input/Output device.

(e) Physical Design: For the physical design, the logical design is transformed into units, which in turn can be decomposed further into implementation units such as programs and modules. During physical design, the primary concern of the auditor is effectiveness and efficiency issues.

(f) System's Operating Platform: In some cases, the new system requires an operating platform including hardware, network and system software not currently available in an organization.

10. System Acquisition: After a system is designed either partially or fully, the next phase of the systems development starts, which relates to the acquisition of operating infrastructure including hardware, software and services.

(a) Acquisition Standards: Management should establish acquisition standards that address the security and reliability issues as per current state-of-the art development standards.

(b) Acquiring Systems Components from Vendors: At the end of the design phase, the organization gets a reasonable idea of the types of hardware, software and services; it needs for the system being developed. The following considerations are valid for both acquisition of hardware and software: Vendor Selection, Geographical Location of Vendor, Presentation by Selected Vendors, Evaluation of Users Feedback.

(c) Other Acquisition Aspects and Practices: In addition to the above, there are several other acquisition aspects and practices also, which are: Hardware Acquisition, Software Acquisition, Contracts, Software Licenses and Copyright Violations, Validation of Vendors’ proposals, Methods of Validating the proposal like Checklists, Point-Scoring Analysis, Public Evaluation Reports, Benchmarking Problems related to Vendor’s Solutions, Testing Problems.

11. System Development: This phase is supposed to convert the design specifications into a functional system under the planned operating system environment. Application programs are written, tested and documented, and system testing conducting. Finally, it results into a fully functional and documented system. A good coded application and programs should have the following characteristics: Reliability, Robustness, Accuracy, Efficiency, Usability and...
Other related aspects of this phase are as follows:

(a) **Program Coding Standards**: The logic of the program outlined in the flowcharts is converted into program statements or instructions at this stage. For each language, there are specific rules concerning format and syntax.

(b) **Programming Language**: Application programs are coded in the form of statements or instructions and the same is converted by the compiler to object code for the computer to understand and execute.

(c) **Program Debugging**: Debugging refers to correcting programming language syntax and diagnostic errors so that the program compiles cleanly. A clean compilation means that the program can be successfully converted from the source code written by the programmer into machine language instructions. For example, bugs may also arise when the program processes data (e.g., invalid input) resulting in abrupt terminations or errors. These will not be identified when computing.

(d) **Testing the Programs**: A careful and thorough testing of each program is imperative to successful installation of any system. The programmer should plan the testing to be performed, including testing of all possible exceptions.

(e) **Program Documentation**: Writing of narrative procedures and instructions for people, who will use software is done throughout the program life cycle. Managers and users should carefully review both internal and external documentation in order to ensure that the software and system behave as the documentation indicates.

(f) **Program Maintenance**: The requirements of business data processing applications are subject to periodic change. This calls for modification of various programs. There are usually separate categories of programmers called maintenance programmers, who are entrusted with this task.

12. **System Testing**: Testing is a process used to identify the correctness, completeness and quality of developed computer software. Different levels/facets of Testing are given as under:

(i) **Unit Testing**: Unit testing is a software verification and validation method in which a programmer tests if individual units of source code are fit for use. A unit is the smallest testable part of an application, which may be an individual program, function, procedure, etc. or may belong to a base/super class, abstract class or derived/child class. There are five categories of tests that a programmer typically performs on a program unit, which are:

   - **Functional Tests**: Functional Tests check ‘Whether programs do, what they are supposed to do or not’.

   - **Performance Tests**: Performance Tests should be designed to verify the response time, the execution time, throughput, primary and secondary memory utilization traffic
5.8 Information Systems Control and Audit

rates on data channels and communication links.

- **Stress Tests**: Stress testing is a form of testing that is used to determine the stability of a given system or entity. It involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results.

- **Structural Tests**: Structural Tests are concerned with examining the internal processing logic of a software system. For example, if a function is responsible for tax calculation, verification of the logic is a structural test.

- **Parallel Tests**: In Parallel Tests, the same test data is used in the new and old system and the output results are then compared.

In terms of techniques, Unit Testing is classified as Static Analysis Testing and Dynamic Testing. Such typical testing techniques are:

(a) **Static Testing**: Static Analysis Tests are conducted on source programs and do not normally require executions in operating conditions. Typical static analysis techniques include: Desk Check, Structured Walk Through and Code Inspection.

(b) **Dynamic Analysis Testing**: Such testing is normally conducted through execution of programs in operating conditions. Typical techniques for dynamic testing and analysis include:

- **Black Box Testing**: Black Box Testing takes an external perspective of the test object, to derive test cases. These tests can be functional or non-functional, though they are usually functional.

- **White Box Testing**: It uses an internal perspective of the system to design test cases based on internal structure. It requires programming skills to identify all paths through the software.

- **Gray Box Testing**: It is a software testing technique that uses a combination of black box testing and white box testing.

(ii) **Integration Testing**: Integration testing is an activity of software testing in which individual software modules are combined and tested as a group. This is carried out in the following two manners:

- **Bottom-up Integration**: It is the traditional strategy used to integrate the components of a software system into a functioning whole. It consists of unit testing, followed by sub-system testing, and then testing of the entire system.

- **Top-down Integration**: It starts with the main routine, and stubs are substituted, for the modules directly subordinate to the main module.

**Regression Testing**: In the context of the integration testing, the regression tests ensure that changes or corrections have not introduced new faults. The data used for the regression tests should be the same as the data used in the original test.
(iii) **System Testing:** It is a process in which software and other system elements are tested as a whole. System testing begins either when the software as a whole is operational or when the well-defined subsets of the software's functionality have been implemented. The types of testing that might be carried out are:

- **Recovery Testing:** This is the activity of testing ‘how well the application is able to recover from crashes, hardware failures and other similar problems’.
- **Security Testing:** This is the process to determine whether an Information System protects data and maintains functionality as intended or not.
- **Stress or Volume Testing:** Stress testing is a form of testing that is used to determine the stability of a given system or entity. It involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results.
- **Performance Testing:** Performance testing is used to determine the speed or effectiveness of a computer, network, software program or device. This testing technique compares the new system's performance with that of similar systems using well defined benchmarks.

(iv) **Final Acceptance Testing:** It is conducted when the system is just ready for implementation. During this testing, it is ensured that the new system satisfies the quality standards adopted by the business and satisfies users. Thus, final acceptance testing has two major parts:

- **Quality Assurance Testing:** It ensures that the new system satisfies the prescribed quality standards and the development process is as per the organization’s quality assurance policy, methodology and prescriptions.
- **User Acceptance Testing:** It ensures that functional aspects expected by users have been well addressed in the new system. There are two types of user acceptance testing: Alpha Testing and Beta Testing.

13. **System Implementation:** The process of ensuring that the information system is operational and then allowing users to take over its operation for use and evaluation is called Systems Implementation. Some of the generic activities involved in system implementation stage are:

(i) **Equipment Installation:** The hardware required to support the new system is selected prior to the implementation phase. Major tasks are: Site Preparation, Installation of New Hardware / Software and Equipment Checkout.

(ii) **Training Personnel:** Training is a major component of systems implementation. When a new system is acquired, which often involves new hardware and software, both users and computer professionals generally need some type of training.

(iii) **System Change-Over Strategies:** Conversion or changeover is the process of changing over or shifting from the old system (may be a manual system) to the new system. Four types of popular implementation strategies are:
• **Direct Implementation / Abrupt Change-Over:** This is achieved through an abrupt takeover – an all or no approach. With this strategy, the changeover is done in one operation, completely replacing the old system in one go.

• **Phased Changeover:** With this strategy, implementation can be staged with conversion to the new system taking place gradually.

• **Pilot Changeover:** With this strategy, the new system replaces the old one in a single operation but only on a small scale e.g. in one division. Any errors can be rectified or further beneficial changes can be introduced and replicated throughout the whole system in good time with least disruption.

• **Parallel Changeover:** This is considered the most secure method with both systems running in parallel over an introductory period. The old system remains fully operational while the new systems come online. With this strategy, the old and the new system are both used alongside each other, both being able to operate independently. If all goes well, the old system is stopped and new system carries on as the only system.

**(iv) System technical changeover or Conversion** activities include Procedure Conversion, File Conversion, System conversion and Scheduling Personnel and Equipment.

14. **Post Implementation Review:** A Post Implementation Review answers the question “Did we achieve what we set out to do in business terms?” Typical evaluations include the following: Development Evaluation, Operational Evaluation and Information Evaluation.

15. **System Maintenance:** Maintaining the system is an important aspect of SDLC. Maintenance can be categorized in the following ways:

• **Scheduled Maintenance:** Scheduled maintenance is anticipated and can be planned for insuring operational continuity and avoidance of anticipated risks.

• **Rescue Maintenance:** Rescue maintenance refers to previously undetected malfunctions that were not anticipated but require immediate troubleshooting solution.

• **Corrective Maintenance:** Corrective maintenance deals with fixing bugs in the code or defects found during execution.

• **Adaptive Maintenance:** Adaptive maintenance consists of adapting software to changes in the environment, such as the hardware or the operating system.

• **Perfective Maintenance:** Perfective maintenance mainly deals with accommodating to the new or changed user requirements and concerns functional enhancements to the system and activities to increase the system’s performance or to enhance its user interface.

• **Preventive Maintenance:** Preventive maintenance concerns with the activities aimed at increasing the system’s maintainability, such as updating documentation, adding comments, and improving the modular structure of the system.
16. Operation Manuals: It is typical user’s guide, also commonly known as Operations Manual. Moreover, it may be a technical communication document intended to give assistance to people using a particular system.

Question 1

Discuss the key characteristics of Waterfall Model in brief. Also explain its major weaknesses.

Answer

Key characteristics of Waterfall Model are given as follows:

- Project is divided into sequential phases, with some overlap and splash back acceptable between phases.
- Emphasis is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.
- Tight control is maintained over the life of the project through use of extensive written documentation, as well as through formal reviews and approval/signoff by the user and information technology management occurring at the end of most phases before beginning the next phase.

Though it is a highly useful model but it suffers from various weaknesses too. Experts and practitioners identify a number of weaknesses including the following:

- It is criticized to be inflexible, slow, costly, and cumbersome due to significant structure and tight controls.
- Project progresses forward, with only slight movement backward.
- There is a little to iterate, which may be essential in some situations.
- It depends upon early identification and specification of requirements, even if the users may not be able to clearly define ‘what they need early in the project’.
- Requirement inconsistencies, missing system components and unexpected development needs are often discovered during design and coding.
- Problems are often not discovered until system testing.
- System performance cannot be tested until the system is almost fully coded, and under capacity may be difficult to correct.
- It is difficult to respond to changes, which may occur later in the life cycle, and if undertaken it proves costly and are thus discouraged.
- It leads to excessive documentation, whose updation to assure integrity is an uphill task and often time-consuming.
- Written specifications are often difficult for users to read and thoroughly appreciate.
- It promotes gap between users and developers with clear division of responsibility.
5.12 Information Systems Control and Audit

Question 2

Briefly explain Prototyping approach.

Answer

Prototyping approach: This approach is basically used for development of such systems as decision support systems, management information systems and expert systems. The goal of prototyping approach is to develop a small or pilot version called a prototype of part or all of a system. A prototype is a usable system or component that is built quickly and at a lesser cost, and with the intention of being modified or replacing it by a full-scale and fully operational system. As users work with the prototype, they make suggestions about the ways to improve it. These suggestions are then incorporated into another prototype, which is also used and evaluated and so on. Finally, when a prototype is developed that satisfies all user requirements, either it is refined and turned into the final system or if it is not suitable. If it is scrapped, the knowledge gained from building the prototype is used to develop the real system.

Experimenting with the prototype helps users to identify additional requirements and needs that they might have overlooked or forgotten to mention. In addition, with prototyping, users have a clearer visual picture of what the final version will look like and they do not have to sign off on a system, which is presented to them in the form of diagrams and specifications lists.

Question 3

Describe major strengths of Prototyping model.

Answer

Major strengths of prototyping model are given as follows:

- It improves both user participation in system development and communication among project stakeholders.
- It is especially useful for resolving unclear objectives; developing and validating user requirements; experimenting with or comparing various design solutions, or investigating both performance and the human computer interface.
- Potential exists for exploiting knowledge gained in an early iteration as later iterations are developed.
- It helps to easily identify, confusing or difficult functions and missing functionality.
- It enables to generate specifications for a production application.
- It encourages innovation and flexible designs.
- It provides for quick implementation of an incomplete, but functional, application.
- It typically results in a better definition of users’ needs and requirements than traditional systems development approach.
A very short time period is normally required to develop and start experimenting with a prototype. This short time period allows system users to immediately evaluate proposed system changes.

Since system users experiment with each version of the prototype through an interactive process, errors are hopefully detected and eliminated early in the developmental process. As a result, the information system ultimately implemented should be more reliable and less costly to develop than when traditional systems development approach is employed.

Question 4

*Explain major strengths and weaknesses of Spiral model.*

**Answer**

Major strengths of Spiral model are given as follows:

- It enhances risk avoidance.
- It is useful in helping for optimal development of a given software with iterations based on project risk.
- It can incorporate Waterfall, Prototyping, and Incremental methodologies as special cases in the framework, and provide guidance as to which combination of these models best fits a given software iteration, based upon the type of project risk. For example, a project with low risk of not meeting user requirements but high risk of missing budget or schedule targets would essentially follow a linear Waterfall approach for a given software iteration. Conversely, if the risk factors were reversed, the Spiral methodology could yield an iterative prototyping approach.

Major weaknesses of Spiral model are given as follows:

- It is challenging to determine the exact composition of development methodologies to use for each iteration around the Spiral.
- It may prove highly customized to each project, and thus is quite complex and limits reusability.
- A skilled and experienced project manager is required to determine how to apply it to any given project.
- No established controls exist for moving from one cycle to another cycle. Without controls, each cycle may generate more work for the next cycle.
- There are no firm deadlines, cycles continue with no clear termination condition leading to inherent risk of not meeting budget or schedule.
Question 5
What do you understand by agile model of software development? Also explain its major strengths and weaknesses in brief.

Answer
Agile Model: This is an organized set of software development methodologies based on the iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery; time boxed iterative approach and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen interactions throughout the development life cycle.

Strengths: Major strengths of agile model identified by the experts and practitioners include the following:

- Agile methodology has the concept of an adaptive team, which enables to respond to changing requirements.
- The team does not have to invest time and efforts and finally find that by the time they delivered the product, the requirement of the customer has changed.
- Face to face communication and continuous inputs from customer representative leaves little space for guesswork.
- The documentation is crisp and to the point to save time.
- The end result is generally the high quality software in least possible time duration and satisfied customer.

Weaknesses: Major weaknesses identified by the experts and practitioners include the following:

- In case of some software deliverables, especially large ones, it is difficult to assess the efforts required at the beginning of the software development life cycle.
- There is lack of emphasis on necessary designing and documentation.
- Agile increases potential threats to business continuity and knowledge transfer. By nature, Agile projects are extremely light on documentation because the team focuses on verbal communication with the customer rather than on documents or manuals.
- Agile requires more re-work and due to lack of long-term planning and the lightweight approach to architecture, re-work is often required on Agile projects when the various components of the software are combined and forced to interact.
- The project can easily get taken off track if the customer representative is not clear about the final outcome.
- Agile lacks attention to outside integration.
Question 6

State and briefly explain the stages of System Development Life Cycle (SDLC).

Answer

System Development Life Cycle: The system development process is initiated when it is realized that a particular business process of the organization needs computerization or improvement. The system development life cycle is a set of six activities which are closely related. These activities after a certain stage can be done parallel to each other. For example the development can be started for some components (sub-systems) which are at the advanced stage of designing. The systems development life cycle method consists of the following activities:

- Preliminary investigation,
- Requirements analysis or systems analysis,
- Design of system,
- Development of software,
- Systems testing, and
- Implementation and maintenance.

The activities are briefly explained below:

- **Preliminary investigation:** When the user comes across a problem in the existing system or a totally new requirement for computerization, a formal request has to be submitted for system development. It consists of three parts; Request Classification Feasibility Study and Request Approval. Generally the request submitted is not stated clearly; hence requires detailed study. On receipt of request and identification of needs, the feasibility study is conducted which includes the aspects related to technical, economic and operational feasibility and is normally conducted by a third party depending upon the quantum and size of the requirements. Approval is sought from top management to initiate the system development.

- **Requirements analysis or systems analysis:** Once the request of the system development is approved, the detailed requirement study is conducted in close interaction with the concerned employees and managers to understand the detailed functioning, short-comings, bottlenecks and to determine the features to be included in the system catering to the needs and requirements of users. This process is termed as “System Requirement Study (SRS)” or System analysis.

- **Design of the system:** This activity evolves the methodology and steps to be included in the system to meet identified needs and requirements of the system. The analyst designs the various procedures, report, inputs, files and database structures and prepares the comprehensive system design. These specifications are then passed on to the Development Team for program coding and testing.
• **Acquisition and development of software:** Once the system design details are resolved and SRS is accepted by the user, the hardware and software details along with services requirements are determined and procured choosing the best-fit options. Subsequently, choices are made regarding which products to buy or lease from which vendors. The choice depends on many factors such as time, cost and availability of programmers. In case of in-house development, the analyst works closely with the programmers. The analyst also works with users to develop documentation for software and various procedure manuals.

• **Systems testing:** Once all the programs comprising the system have been developed and tested, the system needs to be tested as a whole. System testing is conducted with various probable options and conditions to ensure that it does not fail in any condition. The system is expected to run as per the specifications made in the SRS and users’ expectations. Live test data are input for processing, and results are examined. If it is found satisfactory, it is eventually tested with actual data from the current system.

• **Implementation and maintenance:** By the time of accomplishment of the above activities, it is ensured that the requisite hardware and software are installed and the users are trained on the new system to carry out operations independently. For sometimes, hand-holding may be done by the system development team. The operations are monitored closely to ensure users’ satisfaction. The system is maintained and modified to adapt to changing needs of users and business to ensure long-term acceptance of the system.

**Question 7**

The top management of company has decided to develop a computer information system for its operations. Is it essential to conduct the feasibility study of system before implementing it? If answer is yes, state the reasons. Also discuss three different angles through which feasibility study of the system is to be conducted.

**Answer**

Yes, it is essential to carry out the feasibility study of the project before its implementation. After possible solution options are identified, project feasibility-the likelihood that these systems will be useful for the organization—is determined. Feasibility study refers to a process of evaluating alternative systems through various angles so that the most feasible and desirable system can be selected for development. It is carried out by system analysts.

The Feasibility Study of the system is undertaken from three angles i.e. Technical, Economic and Operational. The proposed system is evaluated from a technical viewpoint first and if technically feasible, its impact on the organization and staff is assessed. If a compatible technical and social system can be devised, it is then tested for economic feasibility.

**Technical Feasibility:** It is concerned with hardware and software. Essentially, the analyst ascertains whether the proposed system is feasible with existing or expected computer
Acquisition, Development and Implementation of Information Systems

hardware and software technology. The technical issues usually raised during the feasibility stage of investigation include the following:

- Does the necessary technology exist to do what is suggested (and can it be acquired)?
- Does the proposed equipment have the technical capacity to hold the data required to run the new system?
- Will the proposed system provide an adequate response to inquiries, regardless of the number or location of users?
- Can the system be expanded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access, and data security?

Some of the technical issues to be considered are given in the following table:

<table>
<thead>
<tr>
<th>Table: Technical Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Design Considerations</strong></td>
</tr>
<tr>
<td>Communications Channel configuration</td>
</tr>
<tr>
<td>Communications Channel</td>
</tr>
<tr>
<td>Communications network</td>
</tr>
<tr>
<td>Computer programs</td>
</tr>
<tr>
<td>Data storage medium</td>
</tr>
<tr>
<td>Data storage structure</td>
</tr>
<tr>
<td>File organization and access</td>
</tr>
<tr>
<td>Input medium</td>
</tr>
<tr>
<td>Operations</td>
</tr>
<tr>
<td>Output frequency</td>
</tr>
<tr>
<td>Output medium</td>
</tr>
<tr>
<td>Output scheduling</td>
</tr>
<tr>
<td>Printed output</td>
</tr>
<tr>
<td>Processor</td>
</tr>
<tr>
<td>Transaction processing</td>
</tr>
<tr>
<td>Update frequency</td>
</tr>
</tbody>
</table>

Due to tremendous advancements in computer field, the technology is available for most business data processing systems but sometimes not within the constraints of the firm's
resources or its implementation schedule. Therefore, tradeoffs are often necessary. A technically feasible system may not be economically feasible or may be so sophisticated that the firm's personnel cannot effectively operate it.

**Economic Feasibility:** It includes an evaluation of all the incremental costs and benefits expected if the proposed system is implemented. This is the most difficult aspect of the study. The financial and economic questions raised by analysts during the preliminary investigation are for the purpose of estimating the following:

- The cost of conducting a full system investigation.
- The cost of hardware and software for the class of applications being considered.
- The benefits in the form of reduced costs or fewer costly errors.
- The cost if nothing changes (i.e. the proposed system is not developed).

The procedure employed is the traditional cost-benefit study.

**Operational Feasibility:** It is concerned with ascertaining the views of workers, employees, customers and suppliers about the use of computer facility. The support or lack of support that the firm's employees are likely to give to the system is a critical aspect of feasibility. A system can be highly feasible in all respects except the operational and fails miserably because of human problems. Some of the questions, which may help in conducting the operational feasibility of a project, are stated below:

- Is there sufficient support for the system from management and from users? If the current system is well liked and used to the extent that persons will not be able to see reasons for a change, there may be resistance.
- Are current business methods acceptable to user? If they are not, users may welcome a change that will bring about a more operational and useful system.
- Have the users been involved in planning and development of the project? Early involvement reduces chances of resistance to the system and changes in general and increases the likelihood of successful projects.
- Will the proposed system cause harm? Will it produce poorer results in any respect or area? Will loss of control result in any area? Will accessibility of information be lost? Will individual performance be poorer after implementation than before? Will performance be affected in an undesirable way? Will the system slow performance in any area?

**Question 8**

What are the possible advantages of SDLC from the perspective of IS Audit?

**Answer**

From the perspective of the IS Audit, following are the possible advantages of SDLC:

- The IS auditor can have clear understanding of various phases of the SDLC on the basis of the detailed documentation created during each phase of the SDLC.
The IS Auditor on the basis of his/her examination, can state in his/her report about the compliance by the IS management with the procedures, if any, set by management.

If the IS Auditor has technical knowledge and ability to handle different areas of SDLC, s/he can be a guide during the various phases of SDLC.

The IS auditor can provide an evaluation of the methods and techniques used through the various development phases of the SDLC.

Question 9

What are the major aspects that need to be kept in mind while eliciting information to delineate scope?

Answer

Major aspects that need to be kept in mind while eliciting information to delineate scope are given as follows:

- Different users may represent the problem and required solution in different ways. The system developer should elicit the need from the initiator of the project (alternately called champion or executive sponsor of the project). Addressing his concerns should be the basis of the scope.

- While the initiator of the project may be a member of the senior management, the actual users may be from the operating levels in an organization. An understanding of their profile helps in designing appropriate user interface features.

- While presenting the proposed solution for a problem, the development organization has to clearly quantify the economic benefits to the user organization. The information required has to be gathered at this stage. For example, when a system is proposed for road tax collection, data on the extent of collection and defaults is required to quantify benefits that will result to the Transport Department.

- It is also necessary to understand the impact of the solution on the organization- its structure, roles and responsibilities. Solutions, which have a wide impact, are likely to be met with greater resistance. ERP implementation in organizations is a classic example of change management requirement. Organizations that have not been able to handle it may have a very poor ERP implementation record with disastrous consequences.

- While economic benefit is a critical consideration when deciding on a solution, there are several other factors that have to be given weightage too. These factors are to be considered from the perspective of user management and resolved. For example, in a security system, how foolproof it is, may be a critical factor.

Question 10

Discuss in detail, how the analysis of present system is made by the system analyst?
A Company is offering a wide range of products and services to its customers. It relies heavily on its existing information system to provide up to date information. The company wishes to enhance its existing system. You being an information system auditor, suggest how the investigation of the present information system should be conducted so that it can be further improved upon.

Answer

Detailed investigation of the present system involves collecting, organizing and evaluating facts about the system and the environment in which it operates. Enough information should be assembled so that a qualified person can understand the present system without visiting any of the operating departments. Review of existing methods, procedures, data flow, outputs, files, inputs and internal controls should be intensive in order to fully understand the present system and its related problems.

The following areas may be studied in depth:

- **Review historical aspects**: A brief history of the organization is a logical starting point for the analysis of the present system. The historical facts should identify the major turning points and milestones that have influenced its growth. A review of annual reports can provide an excellent historical perspective. A historical review of the organization chart can identify the growth of management levels as well as the development of various functional areas and departments. The system analyst should identify what system changes have occurred in the past. These should include operations that have been successful or unsuccessful with computer equipment and techniques.

- **Analyze inputs**: A detailed analysis of present inputs is important since they are basic to the manipulation of data. Source documents are used to capture the originating data for any type of system. The system analyst should be aware of the various sources from where data can be initially captured, keeping in view the fact that outputs for one area may serve as an input for another area. The system analyst must understand the nature of each form, what is contained in it, who prepared it, from where the form is initiated, where it is completed, the distribution of the form and other similar considerations. If the analyst investigates these questions thoroughly, he will be able to determine how these inputs fit into the framework of the present system.

- **Review data files maintained**: The analysts should investigate the data files maintained by each department, noting their number and size, where they are located, who uses them and the number of times per given time interval these are used. Information on common data files and their size will be an important factor, which will influence the new information system. This information may be contained in the systems and procedures manuals. The system analyst should also review all online and off line files which are maintained in the organization as these will reveal information about data that are not contained in any output. The related cost of retrieving and processing data is another
important factor that should be considered by the systems analyst.

- **Review methods, procedures and data communications**: Methods and procedures transform input data into useful output. A method is defined as a way of doing something; a procedure is a series of logical steps by which a job is accomplished. A procedure's review is an intensive survey of the methods by which each job is accomplished, the equipment utilized and the actual location of the operations. Its basic objective is to eliminate unnecessary tasks or to perceive improvement opportunities in the present information system. A system analyst also needs to review and understand the present data communications used by the organization. He must review the types of data communication equipment including data interface, data links, modems, dial-up and leased lines and multiplexers. The system analyst must understand how the data communications network is used in the present system so as to identify the need to revamp the network when the new system is installed.

- **Analyze outputs**: The outputs or reports should be scrutinized carefully by the system analysts in order to determine how well they will meet the organization’s needs. The analysts must understand what information is needed and why, who needs it and when and where it is needed. Additional questions concerning the sequence of the data, how often the form reporting is used, how long it is kept on file, etc. must be investigated. Often many reports are a carryover from earlier days and have little relevance to current operations. Attempt should be made to eliminate all such reports in the new system.

- **Review internal controls**: A detailed investigation of the present information system is not complete until internal controls are reviewed. Locating the control points helps the analyst to visualize the essential parts and framework of a system. An examination of the present system of internal control may indicate weaknesses that should be removed in the new system. The adoption of advanced methods, procedures and equipment might allow much greater control over the data.

- **Model the existing physical system and logical system**: As the logic of inputs, methods, procedures, data files, data communications, reports, internal control and other important items are reviewed and analyzed in a top down manner, the process must be properly documented. The logical flow of the present information system may be depicted with the help of system flow charts. The physical flow of the existing system may be shown by employing data flow diagrams. During the process of developing the data flow diagram, work on data dictionary for the new information system should be begun. The data elements needed in the new system will often be found in the present system. Hence, it is wise to start the development of the data dictionary as early as possible.

The flow charting and diagramming of present information not only organizes the facts, but also helps disclose gaps and duplication in the data gathered. It allows a thorough comprehension of the numerous details and related problems in the present operation.

- **Undertake overall analysis of present system**: Based upon the aforesaid investigation of the present information system, the final phase of the detailed investigation includes
5.22 Information Systems Control and Audit

the analysis of:
- the present work volume
- the current personnel requirements
- the present benefits and costs

Each of these must be investigated thoroughly.

Question 11

*Explain two primary methods, which are used for the analysis of the scope of a project in SDLC.*

**Answer**

Two primary methods, which are used for the analysis of the scope of a project in SDLC are given as follows:

- **Reviewing Internal Documents:** The analysts conducting the investigation first try to learn about the organization involved in, or affected by, the project. For example, to review an inventory system proposal, an analyst may try to know how the inventory department operates and who are the managers and supervisors. Analysts can usually learn these details by examining organization charts and studying written operating procedures.

- **Conducting Interviews:** Written documents tell the analyst how the systems should operate, but they may not include enough details to allow a decision to be made about the merits of a systems proposal, nor do they present users' views about current operations. To learn these details, analysts use interviews. Interviews allow analysts to know more about the nature of the project request and the reasons for submitting it. Usually, preliminary investigation interviews involve only management and supervisory personnel.

Question 12

*What are the major objectives of system requirements analysis phase in the SDLC?*

**Answer**

Major objectives of system requirements analysis phase in the SDLC are given as follows:

- To identify and consult stake owners to determine their expectations and resolve their conflicts;
- To analyze requirements to detect and correct conflicts and determine priorities;
- To gather data or find facts using tools like interviewing, research/document collection, questionnaires, observation;
- To verify that the requirements are complete, consistent, unambiguous, verifiable, modifiable, testable and traceable;
• To model activities such as developing models to document Data Flow Diagrams, Entity-Relationship Diagrams; and
• To document activities such as interview, questionnaires, reports etc. and development of a system (data) dictionary to document the modeling activities.

**Question 13**

*If you are the Project Manager of a Software Company with the responsibility for developing a break-through product, combining state of the art hardware and software; will you opt for prototyping as a process model for a product meant for the intensely competitive entertainment market?*

**Answer**

Prototyping as a process model will be inappropriate and hence inadvisable for the following reasons:

• Prototyping requires user involvement. Here, users are consumers of the product who are diffused and may not be inclined to join in.
• When we try to test the product with the involvement of customers, confidential or critical information might get leaked to the competitors on our line of thinking. The element of surprise and also the opportunity to capture the market will be lost.
• Prototyping requires significant time for experimenting. Since the product is meant for the intensely competitive entertainment market, the project manager may not have that much time to experiment, and the competitor may capture the market by entering the market in advance.

**Question 14**

*Describe briefly four categories of major tools that are used for system development.*

**Answer**

The major tools used for system development can be grouped into four categories based on the systems features each document has. These are:

• Components and flows of a system,
• User interface,
• Data attributes and relationships, and
• Detailed system process.

Each of these categories is briefly discussed below:

• **System components and flows**: For system analysts, these tools are helpful to document the data flow among the major resources and activities of an information system. System flow charts are typically used to show the flow of data media as they are
Information Systems Control and Audit

processed by hardware devices and manual activities. A system component matrix provides a matrix framework to document the resources used, the activities performed and the information produced by information system. A data flow diagram uses a few simple symbols to illustrate the flow of data among external entities.

- **User interface**: Designing the interface between end users and the computer system is a major consideration of system analysts while designing the new system. Layout forms and screens are used to construct the formats and contents of input / output media and methods. Dialogue flow diagrams analyze the flow of dialogue between computers and people. They document the flows among different display screens generated by alternative end user responses to menus and prompts.

- **Data attributes and relationships**: These tools are helpful to define, catalogue and design the data resources in information systems. A data dictionary catalogs the description of attributes of all data elements and their relationship to each other as well as to external systems. Entity – relationship diagrams are also used to document the number and type of relationship among entities in a system. File layout forms document the type, size, and names of the data elements in a system. Grid charts help in identifying the use of each type of data element in input / output or storage media of a system.

- **Detailed system process**: These tools are used to help the programmer to develop detailed procedures and processes required in the design of a computer program. Decision trees and decision tables use a network or a tabular form to document the complex conditional logic involved in choosing among the information processing alternatives in a system. Structure charts document the purpose, structure and hierarchical relationships of the modules in a program.

**Question 15**

**Bring out the reasons as to why organizations fail to achieve their Systems Development Objectives?**

**Answer**

Following are the major reasons due to which organizations fail to achieve their system development objectives:

(i) **User Related Issues**: It refers to those issues where user/customer is reckoned as the primary agent. Some of the aspects with regard to this problem are mentioned as follows:

- **Shifting User Needs**: User requirements for IT are constantly changing. As these changes accelerate, there will be more requests for Information systems development and more development projects. When these changes occur during a development process, the development team faces the challenge of developing systems whose very purpose might change after the development process began.

- **Resistance to Change**: People have a natural tendency to resist change, and
information systems development projects signal changes - often radical - in the workplace. When personnel perceive that the project will result in personnel cutbacks, threatened personnel will dig in their heels, and the development project is doomed to failure.

- **Lack of User Participation:** Often users do not participate in the development stage because they are preoccupied with their existing work, or do not understand the benefits of the new system. User apathy ‘I have nothing to gain if I participate’ is also a reason.

- **Inadequate Testing and User Training:** Often systems are not tested due to lack of time and rush to introduce the new system or because problems were not envisaged at the development stage. Inadequate user training may be a result of poor project planning, or lack of training techniques, or because user management does not release personnel for training due to operational pressure.

(ii) **Developer Related Issues:** It refers to the issues and challenges with regard to developers. Some of the critical bottlenecks are mentioned as follows:

- **Lack of Standard Project Management and System Development Methodologies:** Some organizations do not formalize their project management and system development methodologies, thereby making it very difficult to consistently complete projects on time or within budget.

- **Overworked or Under-Trained Development Staff:** In many cases, system developers lack sufficient educational background and requisite state of the art skills. Furthermore, many companies do little to help their development personnel stay technically sound, and often a training plan and training budget do not exist.

(iii) **Management Related Issues:** It refers to the bottlenecks with regard to organizational set up, administrative and overall management to accomplish the system development goals. Some of such bottlenecks are mentioned as follows:

- **Lack of Senior Management Support and Involvement:** Developers and users of information systems watch senior management to determine ‘which systems development projects are important’ and act accordingly by shifting their efforts away from any project, which is not receiving management attention. In addition, management may not allocate adequate resources, as well as budgetary control over use of resources, assigned to the project.

- **Development of Strategic Systems:** Because strategic decision making is unstructured, the requirements, specifications, and objectives for such development projects are difficult to define.

(iv) **New Technologies:** When an organization tries to create a competitive advantage by applying advance technologies, it generally finds that attaining system development objectives is more difficult because personnel are not as familiar with the technology.
In order to overcome these aforementioned issues, organizations must execute a well-planned systems development process efficiently and effectively. Accordingly, a sound system development team is inevitable for project success.

**Question 16**

*Discuss major characteristics of a good coded program in brief.*

**Answer**

A good coded program should have the following characteristics:

- **Reliability**: It refers to the consistency with which a program operates over a period of time. However, poor setting of parameters and hard coding of some data could result in the failure of a program after some time.

- **Robustness**: It refers to the applications’ strength to uphold its operations in adverse situations by taking into account all possible inputs and outputs of a program even in case of least likely situations.

- **Accuracy**: It refers not only to ‘what program is supposed to do’, but should also take care of ‘what it should not do’. The second part becomes more challenging for quality control personnel and auditors.

- **Efficiency**: It refers to the performance per unit.

- **Usability**: It refers to a user-friendly interface and easy-to-understand internal/external documentation.

- **Maintainability**: It refers to the ease of maintenance of program even in the absence of the program developer and includes narrations in the source code.

**Question 17**

*What is Unit Testing? Explain five categories of tests that a programmer typically performs on a program unit.*

**Answer**

**Unit Testing**: Unit testing is a software verification and validation method in which a programmer tests if individual units of source code are fit for use. A unit is the smallest testable part of an application, which may be an individual program, function, procedure, etc. or may belong to a base/super class, abstract class or derived/child class.

Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended. The goal of unit testing is to isolate each component of the program and show that they are correct. A unit test provides a strict, written contract that the piece of code must satisfy.

There are five categories of tests that a programmer typically performs on a program unit. Such typical tests are described as follows:
• **Functional Tests**: Functional Tests check ‘whether programs do, what they are supposed to do or not’. The test plan specifies operating conditions, input values, and expected results, and as per this plan, programmer checks by inputting the values to see whether the actual result and expected result match.

• **Performance Tests**: Performance Tests should be designed to verify the response time, the execution time, throughput, primary and secondary memory utilization and the traffic rates on data channels and communication links.

• **Stress Tests**: Stress testing is a form of testing that is used to determine the stability of a given system or entity. It involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results. These tests are designed to overload a program in various ways. The purpose of a stress test is to determine the limitations of the program. For example, during a sort operation, the available memory can be reduced to find out whether the program is able to handle the situation.

• **Structural Tests**: Structural Tests are concerned with examining the internal processing logic of a software system. For example, if a function is responsible for tax calculation, the verification of the logic is a structural test.

• **Parallel Tests**: In Parallel Tests, the same test data is used in the new and old system and the output results are then compared.

**Question 18**

*Explain the following testing techniques:*

(i) **Black Box Testing**

(ii) **White Box Testing**

(iii) **Gray Box Testing**

**Answer**

(i) **Black Box Testing**: Black Box Testing takes an external perspective of the test object, to derive test cases. These tests can be functional or non-functional, though usually functional. The test engineer has no prior knowledge of the test object’s internal structure. The test designer selects typical inputs including simple, extreme, valid and invalid input-cases and executes to obtain assurance or uncover errors.

This method of test design is applicable to all levels of software testing i.e. unit, integration, functional testing, system and acceptance. The higher the level, the box is bigger and more complex, and the more one is forced to use black box testing to simplify. While this method can uncover unimplemented parts of the specification, one cannot be sure that all existent paths are tested. If a module performs a function, which it is not supposed to, the black box test may not identify it.

(ii) **White Box Testing**: It uses an internal perspective of the system to design test cases based on internal structure. It requires programming skills to identify all paths through the
5.28 Information Systems Control and Audit

software. The tester chooses test case inputs to select paths through the code and determines the appropriate outputs. It is applicable at the unit, integration and system levels of the testing process, it is typically applied to the unit. While it normally tests paths within a unit, it can also test paths between units during integration, and between subsystems during a system level test. After obtaining a clear picture of the internal workings of a product, tests can be conducted to ensure that the internal operation of the product conforms to specifications and all the internal components are adequately exercised.

(iii) **Gray Box Testing:** It is a software testing technique that uses a combination of black box testing and white box testing. In gray box testing, the tester applies a limited number of test cases to the internal workings of the software under test. For the remaining part of the software one takes a black box approach in applying inputs to the software under test and observing the outputs.

**Question 19**

*Explain different changeover strategies used for conversion from old system to new system.*

**Answer**

Different changeover strategies used for conversion from old system to new system are given as follows:

- **Direct Implementation / Abrupt Change-Over:** This is achieved through an abrupt takeover – an all or no approach. With this strategy, the changeover is done in one operation, completely replacing the old system in one go. Fig 5.1 (i) depicts Direct Implementation, which usually takes place on a set date, often after a break in production or a holiday period so that time can be used to get the hardware and software for the new system installed without causing too much disruption.

![Fig. 5.1 (i): Direct Changeover](image)

- **Phased Changeover:** With this strategy, implementation can be staged with conversion to the new system taking place gradually. For example, some new files may be converted and used by employees whilst other files continue to be used on the old system i.e. the new is brought in stages (phases). If a phase is successful then the next phase is started, eventually leading to the final phase when the new system fully replaces the old one as shown in Fig. 5.1(ii).
• **Pilot Changeover:** With this strategy, the new system replaces the old one in one operation but only on a small scale. Any errors can be rectified or further beneficial changes can be introduced and replicated throughout the whole system in good time with least disruption. For example - it might be tried out in one branch of the company or in one location. If successful the pilot is extended until it eventually replaces the old system completely. Fig. 5.1 (iii) depicts Pilot Implementation.

• **Parallel Changeover:** This is considered as a secure method with both systems running in parallel over an introductory period. The old system remains fully operational while the new systems come online. With this strategy, the old and the new system are both used alongside each other, both being able to operate independently. If all goes well, the old system is stopped and new system carries on as the only system. Fig. 5.1(iv) shows parallel implementation.

**Question 20**

Discuss briefly, various activities that are involved for successful conversion with respect to a computerized information system.
Activities involved in conversion: Conversion includes all those activities which must be completed to successfully convert from the existing manual system to the computerized information system. Fundamentally these activities can be classified as follows:

- Procedure conversion;
- File conversion;
- System conversion;
- Scheduling personnel and equipment; and
- Alternative plans in case of equipment failure.

These are briefly discussed as follows:

- Procedure conversion: Operating procedures should be completely documented for the new system. This applies to both computer operations and functional area operations. Before any parallel or conversion activities can start, operating procedures must be clearly spelled out for personnel in the functional areas undergoing changes. Information on input, data files, methods, procedures, outputs, and internal controls must be presented in clear, concise and understandable terms for the average reader. Written operating procedures must be supplemented by oral communication during the training sessions on the system change. Brief meetings must be held when changes are taking place in order to inform all operating employees of any changes initiated. Revisions to operating procedures should be issued as quickly as possible. These efforts enhance the chances of successful conversion.

Once the new system is completely operational, the system implementation group should spend several days checking with all supervisory personnel about their respective areas.

- File conversion: Because large files of information must be converted from one medium to another, this phase should be started long before programming and testing are completed. The cost and related problems of file conversion are significant whether they involve on-line files (common data base) or off-line files. Present manual files are likely to be inaccurate and incomplete where deviations from the accepted formats are common. Computer generated files tend to be more accurate and consistent.

In order for the conversion to be as accurate as possible, file conversion programs must be thoroughly tested. Adequate controls, such as record counts and control totals, should be the required output of the conversion program. The existing computer files should be kept for a period of time until the new system perform in stable manner. This is necessary in case the files must be reconstructed from scratch after a “bug” is discovered later in the conversion routine.

- System conversion: After on-line and off-line files have been converted and the reliability of the new system has been confirmed for a functional area, daily processing can be shifted from the existing information system to the new one. A cut-off point is established so that data base and other data requirements can be updated to the cutoff
point. All transactions initiated after this time are processed on the new system. System development team members should be present to assist and to answer any questions that might develop. Consideration should be given to operating the old system for some more time to permit checking and balancing the total results of both systems. All differences must be reconciled. If necessary, appropriate changes are made to the new system and its computer programs. The old system can be dropped as soon as the data processing group is satisfied with the new system's performance.

- **Scheduling personnel and equipment:** Scheduling data processing operations of a new information system for the first time is a difficult task for the system manager. As users become more familiar with the new system, however, the job becomes more routine.

Before the new design project is complete, it is often necessary to schedule the new equipment. Some programs will be operational while others will be in various stages of compiling and testing. Since production runs tend to push aside new program testing, the system manager must assign ample time for all individuals involved. Schedules should be set up by the system manager in co-ordination with departmental managers of operational units serviced by the equipment.

Just as the equipment must be scheduled for its maximum utilization, so must be personnel who operate the equipment. It is also imperative that personnel who enter input data and handle output data be included in the data processing schedule. Otherwise, data will not be available when the equipment needs it for processing.

- **Alternative plans in case of equipment failure:** Alternative-processing plans must be implemented in case of equipment failure. Who or what caused the failure is not as important in case of equipment failure as the fact that the system is down. Priorities must be given to those jobs critical to an organization, such as billing, payroll, and inventory. Critical jobs can be performed manually until the equipment is set right.

Documentation of alternative plans is the responsibility of the computer section and should be fully covered by the organization’s systems and procedures manual. It should state explicitly what the critical jobs are, how they are to be handled in case of equipment failure, where compatible equipment is located, who will be responsible for each area during downtime and what deadlines must be met during the emergency. A written manual of procedures concerning what steps must be undertaken will help to overcome the unfavorable situation. Otherwise, panic will result in use of the least efficient methods when time is of the essence.

**Question 21**

*Explain corrective and adaptive maintenance in brief.*

**Answer**

**Corrective Maintenance:** Corrective maintenance deals with fixing bugs in the code or defects found during execution. A defect can result from design errors, logic errors, coding errors, data processing and system performance errors. The need for corrective maintenance
is usually initiated by bug reports drawn up by end users. Examples of corrective maintenance include correcting a failure to test for all possible conditions or correcting a failure to process the last record in a file.

**Adaptive Maintenance:** Adaptive maintenance consists of adapting software to changes in the environment, such as the hardware or the operating system. The term environment in this context refers to the totality of all conditions and influences, which act from outside upon the system, for example, business rule, government policies, work patterns, software and hardware operating platforms. The need for adaptive maintenance can only be recognized by monitoring the environment.

### Exercise

1. What is waterfall model of system development? Also discuss its major strengths.
2. What is Rapid Application Development? Discuss its strengths and weaknesses in brief.
3. Agile methodology is one of the popular approaches of system development. What are the weaknesses of this methodology in your opinion?
4. What do you understand by feasibility study? Explain various types of feasibility studies in detail.
5. System Analysts use various fact-finding techniques for determining the needs/requirements of a system to be developed. Explain these techniques in brief.
6. What do you understand by “Requirement analysis”? What is the significance of analyzing the present system and how is it carried out? Explain briefly.
7. What is SDLC? Explain the key activities performed in the Requirements Analysis phase.
8. Discuss the roles of the following with reference to SDLC:
   (i) Steering Committee
   (ii) System Analyst
   (iii) Database Administrator
   (iv) IS Auditor
9. Discuss Final Acceptance Testing in brief.
10. Write short notes on the following:
   (i) Data Dictionary
   (ii) Static Testing
   (iii) Regression Testing
   (iv) System Testing
   (v) Preventive Maintenance
   (vi) Parallel Running Implementation
   (vii) Weaknesses of Incremental Model
   (viii) Auditors' involvement in development work