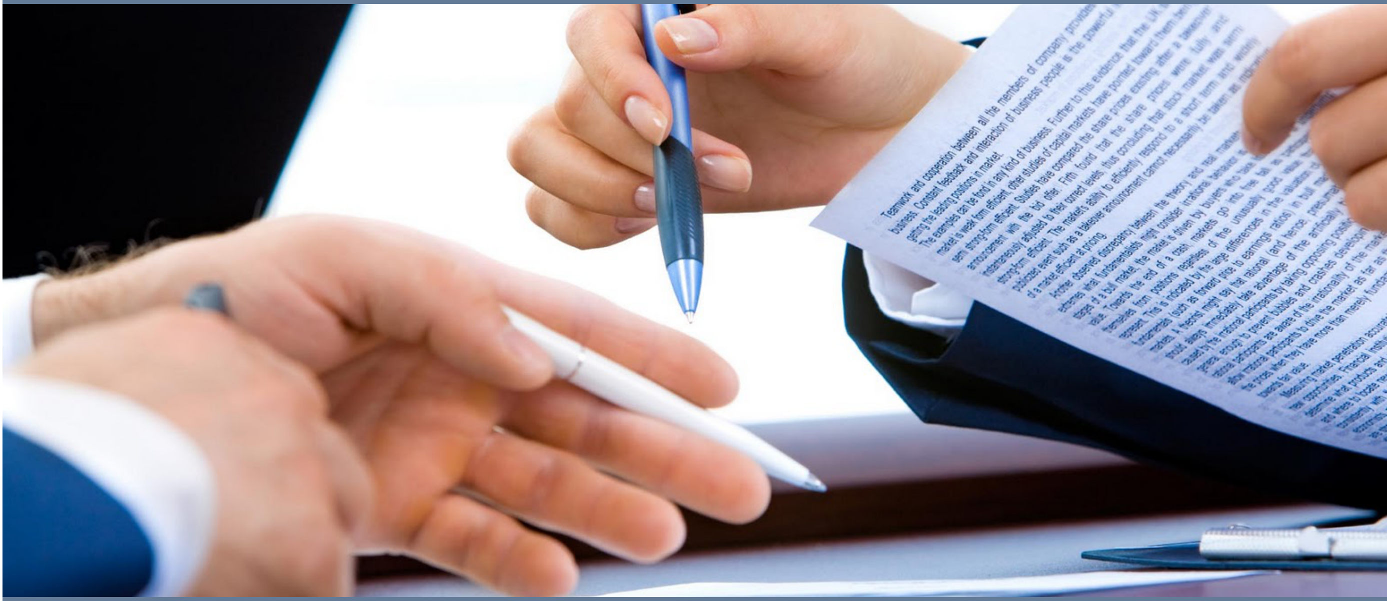
MODULE V

ISA

INFORMATION SYSTEMS AUDIT 2.0 COURSE

SYSTEMS DEVELOPMENT: ACQUISITION, MAINTENANCE AND IMPLEMENTATION



BACKGROUND MATERIAL

Committee on Information Technology
The Institute of Chartered Accountants of India
(Set up by an Act of Parliament)
New Delhi

Background Material On Information Systems Audit 2.0 Course

Module 5 : Systems Development : Acquisition, Maintenance and Implementation (14%)

The Institute of Chartered Accountants of India

(Set up by an Act of Parliament)

New Delhi

Note: There are six other modules which form part of ISA Background Material

Revised Edition	:	July 2015
Committee/Department	:	Committee of Information Technology
E-mail	:	cit@icai.org
Website	:	www.icai.org/http://cit.icai.org
Price	:	₹ 750/- (For Module - I to Module VII, Including DVD)
ISBN	:	978-81-8441-335-9
Published by	:	The Publication Department on behalf of the Institute of Chartered Accountants of India, ICAI Bhawan, Post Box No. 7100, Indraprastha Marg, New Delhi-110 002.
Printed by	:	Finesse Graphics & Prints Pvt. Ltd. Tel.: 022-4036 4600 Fax : 022-2496 2297

Foreword

Information technology (IT) plays a vital role in supporting the activities of any organisation. The growth and change that has come about as a result of developments in technology have important implications. At the same time the increasing use of IT has also led to e-crimes like cyber warfare, hacking, data thefts, DDoS (Distributed Denial of Service) and other computer related frauds. Subsequently, there are various e-Governance, regulatory and compliance issues which are required to be looked into. These technological changes have put more focus on the role performed by Chartered Accountants, especially in the field of Information Systems Audit.

For Chartered Accountants there exist opportunities in Auditing and Assurance as well as consulting areas. Chartered Accountants with their expertise in data and indepth understanding of systems and process functions are uniquely suited for providing consulting in control implementation of IT enabled services as well as review of the same. IT by default rather than by design has become critically relevant for CA firms.

The Committee on Information Technology (CIT) of the Institute of Chartered Accountants of India (ICAI) was established to identify the emerging professional opportunities in the IT sector. It has also been conducting post qualification course on Information Systems Audit thus providing vast opportunities to Chartered Accountants. In view of the dynamism of the sector, a revised edition of the background material for the post qualification course on Information Systems Audit is being brought up by the CIT.

The background material contains various practical aspects, new technologies along with case studies related to Information Systems Audit, which will make this a great learning guide. I appreciate the efforts put in by CA. Rajkumar S. Adukia, Chairman, CA. Atul Kumar Gupta, Vice Chairman, other members and officials of CIT and faculty for bringing out the revised background material.

I hope that it will be a useful learning material and will assist the members in understanding the nuances of the Information Systems Audit. I wish our members great success in the field of Information Systems Audit.

Best Wishes

CA. Manoj Fadnis
President, ICAI

Preface

Information Technology has now emerged as the Business Driver of choice by Enterprises and Government Departments to better manage their operations and offer value added services to their clients/citizens. We now find increasing deployment of IT by enterprises and governments alike in geometric progression.

While the increasing deployment of IT has given immense benefits to enterprises and government departments, there have been increasing concerns on the efficiency and effectiveness of the massive investments made in IT, apart from the safety and security of Information Systems themselves and data integrity. As enterprises are increasingly getting dependent on IT Resources to manage their core business functionality, there are also concerns of Business Continuity.

It is a matter of immense pleasure for me that the Committee on Information Technology of the Institute has come out with the updated ISA Course 2.0 to equip members with unique body of knowledge and skill sets so that they become Information Systems Auditors (ISAs) who are technologically adept and are able to utilise and leverage technology to become more effective in their work and learn new ways that will add value to clients, customers and employers. This will also meet the increasing need of CAs with solid IT skills that can provide IT enabled services through consulting/assurance in the areas of designing, integrating and implementing IT Solutions to meet enterprise requirements.

The updated course material has taken into consideration the latest curriculum of similar professional courses and the recent/emerging developments in the field of Information Technology and IS Auditing and has been updated taking into consideration all the suggested changes and encompasses existing modules, contents and testing methodology.

The specific objectives of the updated ISA course 2.0 is: "To provide relevant practical knowledge and skills for planning and performing various types of assurance or consulting assignments in the areas of Governance, Risk management, Security, Controls and Compliance in the domain of Information Systems and in an Information Technology environment by using relevant standards, frameworks, guidelines and best practices."

The updated ISA Course 2.0 has a blend of training and includes e-Learning, facilitated e-Learning, hands on training, project work in addition to class room lectures. This background material also includes a DVD which has e-Learning lectures, PPTs and useful checklists. The focus is to ensure that practical aspects are covered in all the modules as relevant. I am sure the updated ISA course 2.0 will be very beneficial to the members and enable them to offer IT assurance and advisory services.

I am sure that this updated background material on Information Systems Audit Course 2.0 would be of immense help to the members by enhancing efficiency not only in providing compliance, consulting and assurance services but also open out new professional avenues in the areas of IT Governance, assurance, security, control and assurance services.

Information Technology is a dynamic area and we have to keep updating our auditing methodologies and skill-sets in tune with emerging technologies. We hope this updated ISA 2.0 course is a step in this direction. We welcome your comments and suggestions.

CA. Rajkumar S. Adukia
Chairman
Committee on Information Technology

Table of Contents

SYSTEMS DEVELOPMENT: ACQUISITION, MAINTENANCE AND IMPLEMENTATION SECTION 1: OVERVIEW

CHAPTER 1: SYSTEM DEVELOPMENT LIFE CYCLE (SDLC) INTRODUCTION AND CONCEPTS

Learning objectives 3

1.1 Introduction 3

1.1.1 Objectives of SDLC approach 4

1.1.2 Steps of Systems development 4

1.1.3 Organisation of chapters 5

1.2 Concepts of SDLC 6

1.2.1 When is SDLC initiated? 6

1.2.2 What is SDLC? 7

1.2.3 Business Application System 7

1.3 Typical phases of SDLC 8

Phase 1: Feasibility Study 9

Role of IS Auditor in project initiation and feasibility study phase 9

Phase 2: Requirements Definition 9

Role of IS Auditor in requirements definition phase 9

Phase 3: System Analysis 10

Role of IS Auditor in system analysis phase 10

Phase 4: Design 10

Role of IS Auditor in detailed design phase 10

Phase 5: Development 11

Some key aspects of development 11

Role of IS Auditor in development phase 12

Phase 6: Testing 12
Role of IS Auditor in testing phase 13
Phase 7: User acceptance testing (UAT) or final testing 13
Phase 8: Implementation 14
Role of IS Auditor in implementation phase 14
Phase 9: Support and operations 14
Role of IS Auditor in maintenance and post implementation phase 15
1.4 Changes in SDLC phases 15
Introduction of a decision on Make or Buy 16
Phase 3B: Outsource the application development 16
Phase 3C: Select and Purchase software available in market 16
Phase 4B: Requirement finalisation 17
Phase 4C: Request for Proposal (RFP) 17
Phase 5B: Contract and SLA 17
Phase 5C: Purchase 18
Phase 6C: Configuration (Purchased systems) 18
Changes in Phase 7: UAT 18
1.5 Secure SDLC 18
1.6 Risk associated with SDLC and mitigation planning 20
1.7 Roles and responsibilities of SDLC 21
1.8 Summary 23
1.9 Questions 23
1.10 Answers and Explanations 24

CHAPTER 2: INITIATING SDLC

Learning Objectives 27

2.1 Introduction 27

2.1.1 Drivers, pain points and triggers 27

Pain points 27

Triggers 28

2.2 Feasibility study 28

2.2.1 Technical Feasibility 28

2.2.2 Economic Feasibility 29

2.2.3 Schedule or Time Feasibility 30

2.2.4 Resources Feasibility 30

2.2.5 Operational Feasibility 31

2.2.6 Social Feasibility 31

2.2.7 Compliance Feasibility 31

2.2.8 Internal Controls 31

2.3 Organisational methods for benefit realisation 32

2.4 Business case development 33

2.5 Business Requirements Identification 34

2.5.1 Techniques for requirements gathering 34

1. Understanding Requirements 34

2. Study of History, Structure and Culture 34

3. Study of Information Flows 34

2.5.2 Types of requirements 35

Functional requirements 35

Non-functional requirements 35

Requirements Engineering (RE) Process 36

2.6 Summary ... 37
2.7 Questions ... 37
2.8 Answers and Explanations ... 38

CHAPTER 3: PROJECT MANAGEMENT FOR SDLC

Learning Objectives ... 39
3.1 Introduction ... 39
3.2 Project management frameworks ... 39
3.3 Key concepts of project management ... 40
3.4 Programme and project management and organisation ... 41
3.4.1 Portfolio/Programme Management ... 41
3.4.2 Programme/Project management Organisation forms ... 43
3.5 Project Initiation ... 43
3.5.1 Project management methodology ... 44
3.5.2 Project Context and Environment ... 45
3.5.3 Project Communication and Culture ... 45
3.5.4 Project Objectives ... 45
3.5.5 Project Management Practices ... 46
3.6 Project Planning ... 46
3.7 Project Controlling ... 47
3.7.1 Management of Scope ... 48
3.7.2 Resource management ... 48
3.7.3 Project risk management standards and methods ... 49
Risk in project management ... 49
Risk management process ... 49
3.8 Project Closing ... 50
3.9 Roles and responsibilities ... 51

3.9.1 Steering committee 51
Role of project steering committee 51
3.9.2 Project Sponsor 51
3.9.3 Project Manager 52
3.9.4 Senior management 52
3.9.5 Business management 52
3.9.6 Systems development project team 53
3.9.7 Business function representatives/domain specialists 53
3.9.8 Security Officer 53
3.9.9 IS Auditor 53
Role of IS Auditor in SDLC 53
3.9.10 Quality Assurance (QA) 54
3.9.11 Technology Specialist 54
3.9.12 Systems Analyst 55
3.9.13 Programmers/Developers 55
3.9.14 Testers 55
3.9.15 Documentation Specialist 55
3.9.16 Database Administrator (DBA) 55
3.10 SDLC Project management techniques and tools 55
3.10.1 Computer Aided Software Engineering (CASE) tools 55
Code Generators 56
Development environments and non-procedural languages 56
3.10.2 Software Size Estimation 56
Source lines of code (SLOC) 57
Function Point Analysis (FPA) 57
FPA Feature Points 57

Cost Budgets 58
A. Programme Evaluation Review Technique (PERT) 58
B. Critical Path Methodology 59
C. Gantt Charts 61
3.11 Summary 61
3.12 Questions 62
3.13 Answers and Explanations 63

CHAPTER 4: DIFFERENT MODELS AND METHODS FOR SDLC

Learning objectives 65
4.1 Introduction 65
4.2 SDLC Models 66
4.2.1 Waterfall model 66
Verification and validation – a variant of waterfall model 68
4.2.2 Prototyping methodology 68
Generic phases of model 69
4.2.3 Spiral Model 71
Key characteristics 72
4.2.4 The Incremental Model 73
4.3 SDLC Methodologies 75
4.3.1 Rapid Application Development (RAD) 75
4.3.2 Agile software development methodology 77
Key characteristics of agile processes 78
Key features of agile methodologies 79
4.3.3 Software reengineering and reverse engineering 80
What is Software reengineering 80
Reverse Engineering 81

4.3.4 Object Oriented Software Development (OOSD) 81
Advantages of OOSD 82
4.3.5 Component based development 83
Advantages of component-based development are: 83
Disadvantages 84
4.3.6 Web-based application development 84
4.4 Summary 85
4.5 Questions 85
4.6 Answers and Explanations 86

CHAPTER 5: SYSTEM ACQUISITION FRAMEWORK

Learning objectives 89
5.1 Introduction 89
5.2 Requirements Analysis 90
5.3 Product selection 91
5.4 Request for proposal 93
5.5 Vendor selection criteria/process 94
5.5.1 Service Level Agreements (SLAs) 94
5.5.2 Vendor monitoring 95
5.6 Summary 96
5.7 Questions 97
5.8 Answers and Explanations 98

CHAPTER 6: IMPLEMENTATION AND MAINTENANCE

Learning objectives 99
6.1 Introduction 99
6.2 Testing 100

6.2.1 Unit Testing 100
6.2.2 Integration Testing 102
6.2.3 Regression Testing 103
6.2.4 System Testing 103
6.2.5 Final Testing 104
6.2.6 Other types of Testing 105
6.3 Implementation 106
6.3.1 Implementation Strategies 107
6.3.2 Preparing for implementation 109
6.3.3 Training 109
6.3.4 Conversion 110
6.4 Change management process 111
6.4.1 Emergency Changes 113
6.4.2 Implementing Changes into Production 113
6.4.3 Segregation of Duties 114
6.4.4 Configuration Management 115
6.5 Summary 116
6.6 Questions 117

CHAPTER 7: TRENDS IN TECHNOLOGY IMPACTING SDLC

Learning objective 119
7.1 Introduction 119
7.2 Technology Trends 119
7.2.1 Virtualisation 119
Characteristics of virtualisations affecting SDLC 119
7.2.2 Cloud computing and sourcing options 120
Auditing Cloud services 121

7.2.3 Mobile Computing 122
Mobile devices and applications development 122
7.2.4 Bring Your Own Device (BYOD) 123
Benefits of BYOD 123
Risks of BYOD 123
7.2.5 Big Data and Data analytics 123
Big Data 123
Data analytics 125
7.3 Summary 126
7.4 Questions 127
7.5 Answers and Explanations 128

CHAPTER 8: SDLC REVIEWS AND AUDIT

Learning objectives 129
8.1 Introduction 129
8.2 Role of IS Auditor in SDLC 129
8.2.1 IS Auditor as Team member 129
8.2.2 Mid-project reviews 130
8.2.3 Post implementation review 131
8.3 Summary 132
8.3.1 Key factors to be considered by IS Auditor in SDLC process 132
8.4 Questions 133
8.5 Answers and Explanations 134

SECTION 2: APPENDIX 135

INTRODUCTION TO BACKGROUND MATERIAL

Need for DISA 2.0 Course

Enterprises today in the rapidly changing digital world are inundated with new demands, stringent regulations and risk scenarios emerging daily, making it critical to effectively govern and manage information and related technologies. This has resulted in enterprise leaders being under constant pressure to deliver value to enterprise stakeholders by achieving business objectives. This has made it imperative for management to ensure effective use of information using IT. Senior management have to ensure that the investments and expenditure facilitate IT enabled change and provide business value. This can be achieved by ensuring that IT is deployed not only for supporting organisational goals but also to ensure compliance with internally directed and externally imposed regulations. This dynamic changing business environment impacted by IT provides both a challenge and opportunity for chartered accountants to be not only assurance providers but also providers of advisory services.

The updated ISA course 2.0 has been designed for CAs to provide IT enabled services with the required level of confidence so that management can have trust in IT and IT related services. The ISA course 2.0 builds on the existing core competencies of CAs and provides the right type of skills and toolsets in IT so that CAs can start exploring the immense potential of this innovative opportunity. A key component of this knowledge base is the use of globally accepted good practices and frameworks and developing a holistic approach in providing such services. The background material has been designed with practical perspective of using such global best practices.

Need for updation to DISA 2.0 course

The need for DISA course updation has been extensively discussed considering the objectives and utility of the course. It was decided to update the contents based on suggestions received considering the latest developments in the field of IT and IS Auditing. The updated course has revised modules with key areas of learning as practically relevant for CAs which will enable them to be more effective in their practice for regular compliance audits and also enable to provide IT assurance or consulting services. The updated syllabus has also considered the IT knowledge acquired by the latest batch of CA students who have studied IT in IPCC and Final and have also gone through practical IT trainings. A bridge DISA course is expected to be developed to help existing DISAs to update their knowledge and skills as per the latest course.

Objective of updated DISA Course

The objective of the updated DISA course 2.0 is to equip CAs with a unique body of knowledge and skill-sets so that they can become Information Systems Auditors (ISAs) who are technologically adept and are able to utilise and leverage technology to become more effective in their work and learn new ways and thus add value to their clients or employers. The updated DISA 2.0 course will also meet the increasing market need of CAs with solid IT skills who can provide consulting/assurance in the areas of designing, integrating and implementing IT Solutions to meet enterprise requirements. The updated syllabus of the DISA Course 2.0 has been prepared based on inputs from senior faculty and has undergone numerous reviews over a period of more than two years. The latest curriculum of similar professional courses and the recent/emerging developments in the field of IT and IS Auditing were also referred in updating the course.

Objective of updated DISA Course Material

The primary objective of the updated study material for DISA course is to ensure that DISAs are well versed with the latest IT concepts and practice in the areas of Governance of Enterprise IT, GRC, Assurance, risk, security and controls. The study material has a companion DVD which includes all the reading material and supplementary reference materials and checklists in soft copy. The DVD also includes the e-Learning content available as on date. All the contents in the DVD are presented and linked to aid in easy access of required material. Hence, the DVD and background material will be useful not only as a reading material for passing the DISA exam but also as a reference material for providing IT assurance and consulting services. The sample checklists given in the material can be customised based on scope, objectives of the assignment and considering the nature of business and the technology platform or the enterprise architecture.

Reading of this material is not a one-time exercise but has to be repeated and supplemented with other relevant material and research on the internet. As IT is a rapidly changing area, the material will be updated regularly. Although technology and the services provided using technology undergo rapid changes, the key concepts and requirements for risks, security and control will always remain whether it was the main-frame environment earlier or the mobile computing environment now. Hence, the need for audit and IS audit will always remain.

Use of structured approach

The updated syllabus has been developed by using process oriented structured approach based on the bloom taxonomy of learning and other global best practices. This covers the process/ guidelines to be adapted in development of updated study material.

Overall Objectives

The IT knowledge and skills acquired in the DISA course would enable DISAs to be more effective in using IT for auditing in a computerised environment in existing domains of compliance, consulting and assurance services. The overall objective of the DISA course 2.0 is: **"To provide relevant practical knowledge and skills for planning and performing various types of assurance or consulting assignments in the areas of Governance, Risk management, Security, Controls and Compliance in the domain of Information Systems and in an Information Technology environment by using relevant standards, frameworks, guidelines and best practices."**

Course Coverage

The DISA Course will provide basic understanding of how information technology is used and deployed. It facilitates understanding of how an IS Auditor is expected to analyse, review, evaluate and provide recommendations on identified control weaknesses in different areas of technology deployment. However, it is to be noted that the DISA course is not oriented towards teaching fundamentals of technology. The DISA course is conducted through a good blend of e-learning (online and facilitated), classroom training, hands-on training with practical case studies and project work to ensure practical application of knowledge. The DISA course combines technology, information assurance and information management expertise that enables a DISA to become trusted Information Technology advisor and provider of IS Assurance services. The DISA with the unique blend of knowledge would serve as the "bridge" between business and technology leveraging the CA's strategic and general business skills. The class room training has been supplemented with hands on training. Aspiring DISAs need to remember that the class room training is not expected to be comprehensive but as aid to facilitate understanding. Considering the extensive coverage of the course, duration and the diverse level of participants, the faculty will not be able to cover the material indepth. **Please read the background materials of the specific modules prior to attending the classes to derive maximum benefit from the class room training.**

DISA Certification

DISA Certification through judicious blend of theoretical and practical training provides CAs with better understanding of IT deployment in enterprises which will enable them to be more effective not only in auditing in a computerised environment covering traditional areas of financial/ compliance audits but also in offering IT enabled services. The DISA exam is designed to assess and certify CAs for conducting IS Audit. After successfully completing the course, the DISA candidates are expected to have required knowledge and skills to perform various assurance and consulting assignments relating to Governance, Risk management, Security, Controls and Compliance in the domain of Information Systems, Information Technology and related areas.

DISA Course : Basic competency requirements

After successful completion of the course, the DISA candidates will have conceptual clarity and will demonstrate basic competency in the following key areas:

- Overall understanding of information system and technology: concepts and practice
- Risks of deployment of information system and technology
- Features and functionalities of security and controls of IT components and IT environment.
- Controls which could be implemented using the security features and functionalities so as to mitigate the risks in the relevant IT components and environments.
- Recommend IT risk management strategy as appropriate.
- Apply appropriate strategy, approach, methodology and techniques for auditing technology using relevant IS Audit standards, guidelines and procedures and perform IS Assurance and consulting assignments.

Modules of the DISA Course

The updated ISA certification is granted exclusively to CAs who demonstrate considerable expertise in domain areas of IT Governance, Security, Control and assurance through their knowledge, skills and experience The primary purpose of the ISA exam is to test whether the candidate has the requisite knowledge and skills to apply IS assurance principles and practices in the following modules:

No.	Name of Module	(%) Q's
1	Primer on Information Technology, IS Infrastructure and Emerging Technologies	20
2	Information Systems Assurance Services	13
3	Governance and Management of Enterprise Information Technology, Risk Management and Compliance Reviews	13
4	Protection of Information Systems Infrastructure and Information Assets	20
5	Systems Development: Acquisition, Maintenance and Implementation.	14
6	Business Applications Software Audit	13
7	Business Continuity Management	7

Learning Objectives

The DISA course is not expected to be an in-depth comprehensive coverage of different aspects of IT such as computer hardware, operating system, network, databases, application software, etc. but is focused on training on how to review IT controls and provide assurance on secure technology deployment.

The key learning objectives are:

1. Demonstrate understanding of functioning of key components of existing and emerging information technology and their practical deployment.
2. Provide IS assurance or IT Enabled services and perform effective audits in a computerised environment by using relevant standards, guidelines, frameworks and best practices.
3. Evaluate structures, policies, procedures, practices, accountability mechanisms and performance measures for ensuring Governance and management of Information Technology, risk management and compliance as per internal and external stakeholder requirements.
4. Provide assurance, consulting or compliance services to confirm that enterprise has appropriate security and controls to mitigate risks at different layers of technology as per risk management strategy.
5. Provide assurance or consulting services that the management practices relating to systems development: acquisition, maintenance and implementation are appropriate to meet enterprise strategy and requirements.

6. Provide assurance or consulting services to validate whether required controls have been designed, configured and implemented in the application software as per enterprise and regulatory requirements and provide recommendations for mitigating control weaknesses as required.
7. Provide assurance or consulting services to confirm whether the Business continuity management strategy, processes and practices meet enterprise requirements to ensure timely resumption of IT enabled business operations and minimise the business impact of a disaster.
8. Plan and perform IS assurance or consulting assignments by applying knowledge learnt by presenting project assignment relating to allotted case study to confirm understanding.

Skill Levels

The updated syllabus provides specific skills in each of the three categories of skill areas. The suggested skill levels ensure that the updated syllabus through all the modules has right blend of concepts and practice. The skill levels will be considered by the authors of study material and also in testing methodology through the eligibility tests and assessment test.

Weightage and category of skills

No.	Skills Category	Weightage (%)
1	Knowledge and Understanding	30 to 40
2	Application of the Body of Knowledge	55 to 60
3	Written communication	5 to 10

Summary of revised DISA Training

No.	Mode of Training	Weightage (%)
1	e-Learning Online (self)	12
2	e-Learning facilitated (lectures)	12
3	Classroom Training (lectures)	42
4	Hands-on Training (on laptop)	24
5	Project Work (self in groups)	10
	Total	**100**

Key highlights of DISA training

DISA Training includes e-Learning, hands on Training, project work in addition to classroom lectures.

- Candidates will have to successfully complete e-learning mode before joining classroom training.
- The training in classroom and hands-on training will follow the order in sequential order of the modules. This includes an inter-mix of classroom lectures and hands-on training. The hands-on training pre-supposes and builds on understanding of concepts of the classroom lectures.
- The training includes mandatory e-Learning of 12 hours for Module-1 and 6 hours for Module-2 and passing in the online test is mandatory and part of the eligibility score.
- Module-4 will have class room lectures of 2 days and hands on training of 2 days. Module-6 will have hands on training of 2 days. **Supplementary e-Learning Lectures covering Modules 4 and 6 are also included.** These will be added in due course and will be made available through DVD or online.
- **Hands on training for Module 4 and 6 will be conducted by the experienced faculty at same venue as class rooms with all participants performing exercises on their own laptops with pre-loaded software and sample/test data as specified in advance.**

DISA 2.0 Course Background Material

The DISA Course 2.0 Background Material is intended to assist in preparing for the DISA exam. The material is a one source of preparation for the exam, but should not be considered as the only source nor should it be viewed as a comprehensive collection of all the information that is required to pass the exam. Participants are encouraged to supplement their learning by using and researching the references provided in the material.

DISA 2.0 Course DVD

The Reading material for the DISA 2.0 course includes a DVD which is comprehensive collection of educational material for revised DISA Course 2.0. This DVD will aid self-learning and includes Background Material, Reference Material, e-Lectures, PowerPoint Presentations, Podcasts/MP3 Files and Self-Assessment Quiz (). This DVD is designed to be supplementary to the background material. It has to be used for self-learning and also as a training aide for the DISA Course 2.0 and DISA candidates are strongly advised to use this for studying for the ISA course.

Standard PPTs for each of the modules of the DISA 2.0 course have been prepared by the authors based on the background material. These are provided in the DVD only and are expected to serve as reference material during the class. Additional references materials and checklists of the course are only included in the DVD. The PPTs may be customised or updated by the faculty as required. Participants are encouraged to copy the DVD contents in their laptops and use this as reference in the classroom training.

Feedback and updates

We compliment you on choosing to join the DISA 2.0 Course and wish you a great learning experience. Please make best use of the material and the training. **Please note that the training is expected to supplement your reading of the material prior to attending the course.** Please participate actively in the training to make the best use of the training The material will be useful to you not only to aid you in preparing for exam but also for providing services in the area of Governance, Assurance and consulting.

Please note that the **background material has been contributed by practising professionals who have shared their expertise and reflects different writing styles of the authors.**

Please provide your feedback on areas of improvement of the course and the reading material in the specified format so that further improvements can be made. Please email your feedback or queries to: **isa@icai.in.** Please visit CIT portal http://cit.icai.org/ for the latest updates of the DISA course. We wish you a great learning experience and a rewarding career as an IS Auditor.

Committee of Information Technology, ICAI

The course material includes references to some specific companies, hardware or software. This reference is only for educational purposes and is not in any way endorsement of the company or products. All copyrights are acknowledged and belong to the rightful owners.

Module 5:

Systems Development: Acquisition, Maintenance and Implementation (14%)

Section 1: Overview

SECTION 1: CONTENTS
CHAPTER 1: SYSTEM DEVELOPMENT LIFE CYCLE (SDLC) INTRODUCTION AND CONCEPTS

Learning objectives

After completion of this chapter you should have conceptual clarity on the basic concepts of System Development Life Cycle (SDLC), how SDLC has changed over a period of time due to changes in technology and environment and how this has led to inclusion of additional phases in SDLC. This chapter covers:

- Traditional SDLC phases and overview of the main activities;
- Additional phases due to availability of outsourcing and generic customisable software; and
- Steps added in different phases due to security requirements (Secure SDLC or SSDLC).

1.1 Introduction

In the present era, the critical need for Information Technology (IT) can be understood from the need to plan and develop safe, secure, and reliable system solutions using information systems which form the backbone for developing innovative product offerings and services. Information systems also play a key role in performing short and long term management functions and activities. There is also greater need to ensure appropriate level of security when developing information systems so as to establish appropriate privacy and protection practices and to develop acceptable implementation strategies for these practices. The SDLC methodology is designed to satisfy these needs by establishing procedures, and practices governing the initiation, definition, design, development, deployment, operation, maintenance, enhancement, and retirement of automated information systems.

If information is the currency of the current century then information systems provide the edifice for designing and deploying information technology for implementing organisation processes which improve efficiency and effectiveness of business functions. Organisations use Information systems to perform various business functions such as processing of business transactions, providing information for decision-making, resolve recurring issues, assisting senior management in designing strategy, linking information with corporate data. Applications systems represent the synergy of human, IT hardware and software. IT has been continuously developing at a rapid pace but the most important aspect for use of IT is human know-how and the use of ideas to harness IT to perform the organisation processes, tasks and activities. The process of developing application software to achieve functional objectives is called ‘system development’. Due to dynamic changes in requirements, these systems may require regular update and maintenance to address these changes and finally at a point in time, these must be replaced, i.e. retired with new system. Hence, it is said to follow a life cycle.

1.1.1 Objectives of SDLC approach

- Deliver quality systems which meet or exceed customer expectations and within cost estimates.
- Develop quality systems using an identifiable, measurable, and repeatable process.
- Establish an organisational and project management structure with appropriate levels of authority to ensure that each system development project is effectively managed throughout its life cycle.
- Identify and assign the roles and responsibilities of all affected parties including functional and technical managers throughout SDLC.
- Ensure that system development requirements are well defined and subsequently satisfied.
- Provide visibility to the functional and technical managers for major system development resource requirements and expenditure.
- Establish appropriate levels of management authority to provide timely direction, co-ordination, control, review, and approval of the system development project.
- Ensure accountability structure and mechanism for project management.
- Ensure that projects are developed within current and planned IT infrastructure.
- Identify project risks early and manage them before they become problems.

1.1.2 Steps of Systems development

An information system undergoes following steps of development: (Figure 1.1)

- Conceptualisation
- Feasibility
- Design and development
- Testing
- Implementation
- Use
- Retirement (replacement)

The stages from conception to retirement of an information system are said to undergo a life cycle. With availability of readymade products and availability of outsourcing options, the design, development and testing phases have undergone changes.

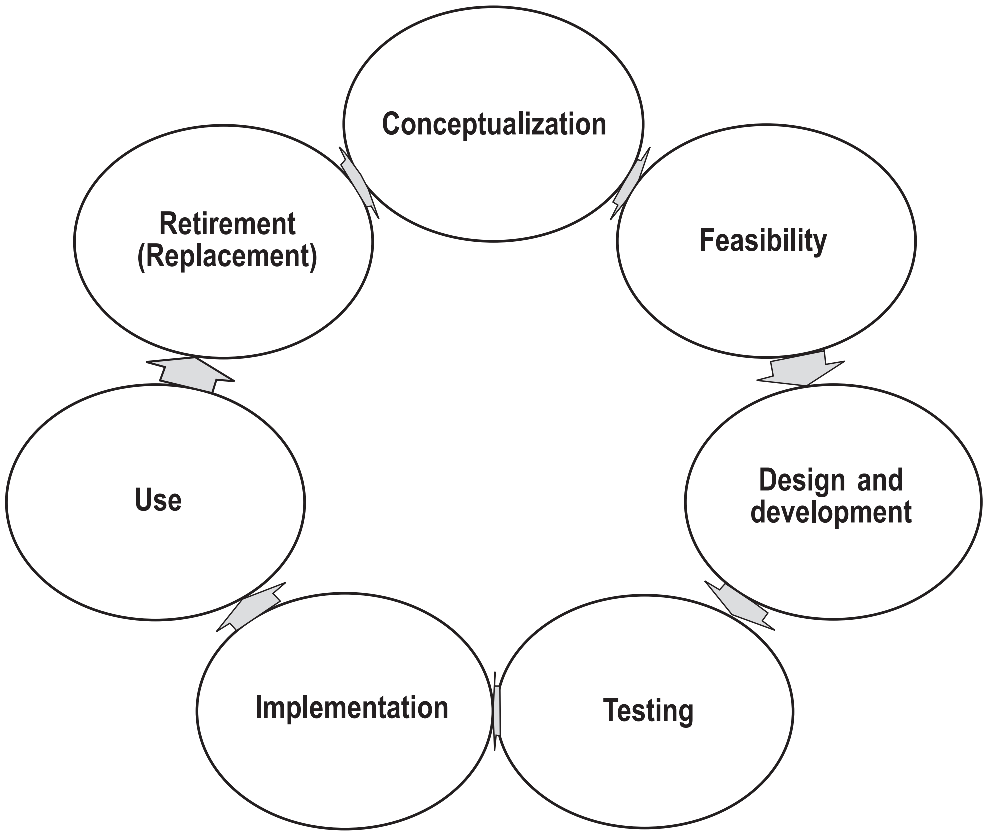


Figure 1.1: Application life cycle

Current trends and threats scenarios have impacted System Development leading to consideration of security in the various phases of SDLC. This is referred to as Secure SDLC and involves additional step in each phase of SDLC requiring consideration of security aspects in each of the phases. This chapter covers key activities of each phase and provides overview of each of the steps. The detailed explanation of the phases and steps are covered in ensuing chapters.

1.1.3 Organisation of chapters

The chapters (1 to 6) in Section 2 of this module are organised as per the logical sequence in which an SDLC project is typically executed in an organisation.

Chapter 1 provides the basic information about SDLC that is required for IS auditor and project manager involved in SDLC activities. The chapter tries to answer why it is called life cycle? What are typical activities or phases through which development happens? These activities have undergone changes over a period of time and hence knowledge about these changes and security requirements in SDLC are covered.

Chapter 2 focuses on actions organisations need to take relating to when to initiating SDLC project. This covers initial phases of SLDC including feasibility study, requirement definition, expected benefits to organisation and developing business case for SDLC project. Decision about execution of next phases is based on business case and hence it necessary to understand these activities. Phase 1 feasibility study and Phase 2 requirement definition of typical SDLC phases (discussed in Chapter 1) are covered in this Chapter (2).

Chapter 3 provides the basic knowledge on project management with focus on SDLC. Once the organisation has decided to initiate project for automated solution using SDLC, IS auditor may be involved in reviewing this. Hence, IS auditor needs to have knowledge about SDLC project management activities. The details of Phases 3 and 4 of a typical SDLC which were discussed in chapter 1 are explained. This chapter does not cover general project management concepts but focuses on project management practices required for executing SDLC project.

Chapter 4 introduces models and methods used for system design and development that go together. To ensure that final system meets business requirements, project manager and system analyst has to choose the right development method for new application. IS auditor has to know key characteristics, strength and weaknesses of these methods. All these methods have the common objective of application development with many steps being similar.

Chapter 5 discusses software acquisition and outsourcing of development. In case organisation has decided to purchase the software then phases 3, 4 and 5 of typical SDLC are not applicable for the organisation. However, the accountability of actions cannot be outsourced. Hence, the organisation has to provide necessary inputs on controls to be implemented in the application.

Chapter 6 provides information on testing, UAT, implementation and support and operation activities which are covered in phases 6 to 9. Once the application is ready to be implemented whether it is developed or acquired, it must be implemented across the organisation using steps outlined in phases 6 to 9 to derive the benefits.

Chapter 7 discusses what IS auditor has to know about new trends in IT deployment and its impact on SDLC.

Chapter 8 provide basic and high-level information on auditing of SDLC project.

1.2 Concepts of SDLC

1.2.1 When is SDLC initiated?

The need for business development or acquisition of new applications may arise to due to following situations:

- New service delivery opportunity that relates to a new or existing business process (e.g. e-commerce);
- Issues and problems with an existing systems/business process (complaints from customers/users);
- Change in strategic focus leading to an opportunity that will provide benefits to the organisation (Mergers and acquisitions, or new service delivery channels like ATM for banks);
- New opportunity due to advancement of existing technology or availability of new technology (e.g. use of mobile technology for banking services); and

- Use of automation by competitors to enhance quality of services.

All of these situations directly affect the business drivers. Business drivers can be defined as the attributes of a business function (service delivery) that arise out of strategic objectives to enhance targets and goals of business function to achieve the strategic business goals. In other words business objectives defined by strategy gets translated into drivers for business operations which require new application software or upgrading of existing application software. This results in initiating an SDLC project.

Business application system/software is designed to support a specific organisational service, function or process, such as inventory management, payroll, market analysis or e-commerce. The goal of an application system is to turn data into information. System development involves developing or acquiring and maintaining application systems which are used for various day-to-day business activities. These business activities are called as Business Processes and they process data of relevant business processes.

1.2.2 What is SDLC?

SDLC refers to the process of examining a business situation with the intent of improving it through better procedures and methods. This is required when there is need to change business processes due to requirements arising out of customers/stakeholders expectations and business strategy. These changes are generally attributed to need to automate the service delivery using information and related technology. System development involves developing or acquiring and maintaining application systems that are used for various day-to-day business process activities. Generally these system processes data of business transactions.

A standard set of steps used for developing systems is called a SDLC. SDLC generally uses various methods depending upon the type and nature of application. For example a batch processing application that processes historical data to generate reports for management's information may use a model where activities are performed one after another (waterfall model), where as if there is no clear understanding of what functions can be automated, an iterative (spiral) model may be used. (These models are discussed in detail in Chapter 4). Similarly, depending upon the availability of skilled resources, the development team may adopt different methodology for developing software.

1.2.3 Business Application System

Business application system, also called application software, is designed to support a specific organisational function or process, such as inventory management, payroll, or market analysis. The goal of application system is to turn data into information. For example, software developed for the inventory department at a bookstore may keep track of the number of books in stock for the latest bestseller. Application system for the payroll department may keep track of the changing pay scales of the employees.

System development involves developing or acquiring and maintaining application systems which are used for various day-to-day business activities. These business activities are called as business processes and they process data. The effective management and control of this system development is critical as the business systems process and control information assets of the organisation. The use of a standard set of steps to develop and support business applications is called Systems Development Methodology.**It is important to remember that the terms SDLC and SDM are often used interchangeably.**

1.3 Typical phases of SDLC

Typically a SDLC consists of phases as shown in figure 1.2. Although these phases are common for any SDLC model, the way these are executed vary for each model. Although figure 1.2 is titled as 'SDLC'; this does not cover the last two phases shown in figure 1.1 i.e. using the application and retirement or replacement of application. System Development life cycle is a sequence of activities performed by group of users and IT development experts. These set of activities are grouped together to form phases and generally each phase has predetermined set of deliverables and/or a milestone to be reached.

The number of phases might vary for each SDLC project depending upon milestones and/or deliverables. For example: if an organisation is developing a software using internal development team, it may not lay much emphasis on user acceptance testing (UAT) and may club this activity with testing phase, whereas in case of outsourced development or acquired software, user acceptance testing (UAT) is a major milestone and has predefined deliverables that are signed off. In the diagram below, considering the criticality of outsourced applications, UAT has been shown as a separate phase

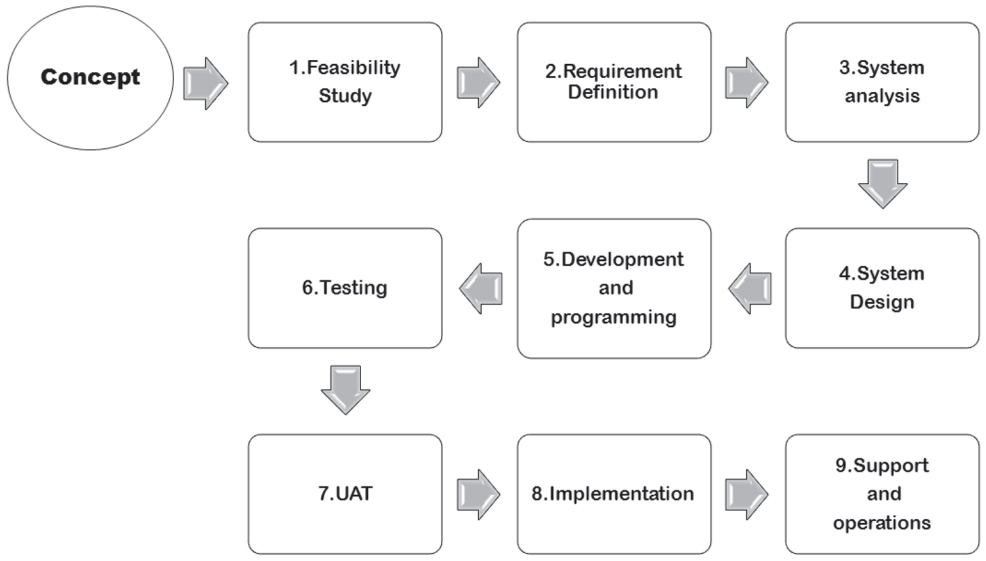


Figure 1.2: Typical phases of SDLC process

The concept or idea is starting point where automation of business function using information system is considered from business benefit perspective. For example a bank may decide to launch a new service delivery channel like internet banking, or a store may provide on-line shopping service through internet or an airline may implement a system via information kiosk for user enabled check-in and web check-in for reducing manual process. Once the idea is considered it passes through SDLC phases till it is materialised. Sometimes the concept requires further investigation (preliminary investigation) before SDLC project is initiated. Preliminary investigation is not considered part of typical SDLC phases but some organisations may include it in SDLC.

Phase 1: Feasibility Study

The feasibility study is based on technical, economical and social aspects and this helps in determining strategic benefits of using system. These benefits can be either in productivity gains or in future cost avoidance. The study has to also identify and quantify the cost savings and estimate the probable return on investment. This information is used to building a business case covering both tangible as well as intangible factors such as readiness of the business users and maturity of the business processes. The business case provides justification for proceeding to the next phase and is also used for reviewing progress or evaluating success of SDLC project.

Detailed steps of feasibility study are discussed in Chapter 2. The feasibility study shall be different for different application depending upon the expected benefits for the organisation. For example if an organisation intends to implement an application that is already implemented by various organisation of similar type, it may use a generic feasibility study and focus only on the benefits to the organisation, such as providing internet banking services or mobile banking services.

Role of IS Auditor in project initiation and feasibility study phase

- Review the documentation for the reasonableness.
- Review cost justification/benefits with schedule of when the anticipated benefits will be realised.
- Identify if the business needs used to justify the system actually exist.
- Justification for going for a development or acquisition.
- Review the alternate solutions for reasonableness.
- Review the reasonableness of the chosen solution.

Phase 2: Requirements Definition

This phase involves preparing the statement of intent explaining the problem or the need for new application to provide functional, service and quality requirements of the solution system. The user needs to be actively involved in requirements definition. This involves:

- Studying needs of the users
- Obtaining inputs from employees and managers on their expectations
- Determining information requirements of the users

Several fact-finding techniques and tools such as questionnaires, interviews, observing decision-maker behaviour and their office environment etc. are used for understanding the requirements.

Role of IS Auditor in requirements definition phase

- Identify the affected users and the key team members on the project to verify that they are having an appropriate representation.
- Review detailed requirements definition document and verify its accuracy and completeness through interviews with the affected and requested user departments.
- Review existing data flow diagrams and other related specifications like forms, data descriptions, output formats, etc., to ensure that they cover the user requirements.

Phase 3: System Analysis

This refers to the process of gathering and interpreting facts, diagnosing problems, and using the information to recommend improvements to the system. Before arriving at new design, one must thoroughly understand existing process/system and map them against new requirements to understand changes and rationale for changes. Analysis is also important to decide upon system design approach. Traditional system development generally adopt a data oriented approach since it had been focused on processing and presenting of business data., However, due to extensive use of technology in modern organisations, the focus now is more on service oriented approach where the objective of the system is to provide services using data models.

Role of IS Auditor in system analysis phase

- Verify that management has approved the initiation of the project and the cost.
- In case of acquisition, determine that an appropriate number of vendors have been given proposals to cover the true scope of the project and requirements of the users.
- Determine whether the application is appropriate for the user of an embedded audit routine and if so request may be made to incorporate the routine in conceptual design of the system.

Phase 4: Design

This phase takes primary inputs from *phase 1 requirement definition*. Based on the requirements identified, the team may need to finalise requirements by multiple user interactions and establishes a specifications baseline for development of system and sub system.

These specifications describe:

- Parts of the system
- How they interface
- How the system need to be implemented
- Type of hardware, operating system and other software
- Network facilities
- Programme and database specifications
- Security considerations

Additionally, a formal change control process is established to prevent uncontrolled entry of new requirements during development process

Role of IS Auditor in detailed design phase

- Review system flowcharts for adherence to the general design
- Review input, processing and output controls have been appropriately included in the system
- Assess adequacy of the audit trails which provide traceability and accountability
- Verify key calculations and processes for correctness and completeness

- Interview users to ascertain their level understanding of the system design, input to the system, screen formats and output reports
- Verify that system can identify erroneous data correctly and can handle invalid transactions
- Review conceptual design to ensure the existence of appropriate controls
- Review quality assurance and quality control results of programmes are developed
- Verify the design for its completeness and correctness and it meets the defined requirements
- Verify that functional data created during requirement phase is complete and test plans are developed

Phase 5: Development

Use the design specifications to begin programming and formalising supporting operational processes of the system. After the system design details are resolved, the resources needs such as specific type of hardware, software, and other services are determined. The choices depend on many factors such as time, cost and availability of skilled resources programmers and testers. The analyst works closely with the programmers. During this phase, the analyst also works with users to develop required documentation for software, including various procedure manuals.In the development phase, the design specifications are converted into a functional system that will work in planned system environment. Application programmes are written, tested and documented. Finally this results in development of a fully functional and documented system. A very well coded application programme should have the following characteristics:

- **Reliability:** It refers to the consistency with which a programme operates over a period of time. However, poor setting of parameters and hard coding of some data subsequently could result in the failure of a programme after some time.
- **Robustness:** It refers to the applications' strength to perform operations in adverse situations by taking into account all possible inputs and outputs of a programme considering even the least likely situations.
- **Accuracy:** It refers not only to 'what programme is supposed to do', but also the ability to take care of 'what it should not do'. The second part is of great interest for quality control personnel and auditors.
- **Efficiency:** It refers to the performance per unit cost with respect to relevant parameters and it should not be unduly affected with the increase in input values.
- **Usability:** It refers to a user-friendly interface and easy-to-understand internal/external documentation.
- **Readability:** It refers to the ease of maintenance of programme even in the absence of the programme developer.

Some key aspects of development

1. **Program Coding Standards:** The logic of the programme outlined in the flowcharts is converted into programme statements or instructions. For each language, there are specific rules concerning format and syntax. Syntax means vocabulary, punctuation and

grammatical rules available in the language manuals that the programmer has to follow strictly and pedantically. Different programmers may write a programme using different sets of instructions but each giving the same results. This might create a problem for changes to be done to the programme which has been written by another programmer. Therefore, the coding standards are to be defined so as to serve as a method of communication between teams, amongst the team members and users resulting in better controls. Coding standards minimize the system development issues due to programmer turnover. These standards provide simplicity, interoperability, compatibility, efficient utilisation of resources and reduce processing time.

2. **Programming Language:** Depending upon the development approach, the analyst decides the programming language to be used. Application programmes 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. The programming languages commonly used are:
   - High level general purpose programming languages such as COBOL and C;
   - Object oriented languages such as C++, JAVA etc.;
   - Scripting language such as JavaScript, VBScript; and
   - Decision Support or Logic Programming languages such as LISP and PROLOG.

The choice of a programming language may depend on various pertinent parameters. In general, language selection may be made on the basis of application area; algorithmic complexity; environment in which software has to be executed; performance consideration; data structure complexity; knowledge of System Development staff; and capability of in-house staff for maintenance.

Role of IS Auditor in development phase

- Ensure that documentation is complete
- Review QA report on adopting coding standards by developers
- Review the testing and bugs found are reported and sent for rework to developers

Phase 6: Testing

Before the information system can be used, it must be tested. Systems testing are done at various stages during development till implementation. There are primarily two types of testing:

1. Quality assurance testing includes unit testing, interface testing, integration testing and peer reviews.
2. User acceptance testing (UAT) also known as final acceptance testing (next phase).

Testing primarily focuses on ensuring that the software does not fail i.e. it will run according to its specifications and in the way users expect. Special test data are input for processing and the results are examined against predetermined output. If it is found satisfactory, it is eventually tested with actual data from the current system. *(Testing is discussed in more detail in Chapter 6)*

Role of IS Auditor in testing phase

- Review the test plan for completeness and correctness.
- Review whether relevant users have participated during testing phase.
- Review error reports for their precision in recognising erroneous data and for resolution of errors.
- Verify cyclical processes for correctness(example: year-end process, quarter-end process)
- Interview end-users of the system for their understanding of new methods, procedures and operating instructions.
- Review the system and end-user documentation to determine its completeness and correctness.
- Review whether reconciliation of control totals and converted data has been performed to verify the integrity of the data after conversion.
- Review all parallel testing results.
- Test the system randomly for correctness.
- Review unit test plans and system test plans to determine that tests for internal control are addressed.
- Verify that the system security is functioning as designed by developing and executing access tests.
- Ensure test plans and rest results are maintained for reference and audit

Phase 7: User acceptance testing (UAT) or final testing

Establishes the actual operation of the new information system, with the final iteration of user acceptance testing and user sign-off. Organisation may consider going for a certification and accreditation process to assess the effectiveness of the business application. This provides assurance to the management about:

1. Mitigating risks to an appropriate level.
2. Providing accountability over the effectiveness of the system in meeting objectives.
3. Establishing an appropriate level of internal control.

UAT supports the process of ensuring that the system is production-ready and satisfies all documented requirements. The methods include:

- Definition of test strategies and procedures.
- Design of test cases and scenarios.
- Execution of the tests.
- Utilisation of the results to verify system readiness.

Acceptance criteria are defined so that a deliverable satisfies the predefined needs of the user. A UAT plan must be documented for the final test of the completed system. The tests are written from a user perspective and should test the system in a manner as close to production as possible. For example, tests may be based around typical pre-defined, business process scenarios. If new business processes have been developed to accommodate the new or modified system they should also be tested at this point. A key aspect of testing should also include testers seeking to verify that supporting processes integrate into the application in an acceptable manner. Successful completion would generally enable a project team to hand over a complete integrated package of application and supporting procedures.

Ideally, UAT should be performed in a secure testing or staging environment. A secure testing environment where both source and executable code are protected helps to ensure that unauthorized or last-minute changes are not made to the system without going through the standard system maintenance process. The nature and extent of the tests will depend on the magnitude and complexity of the system change.

Test plan, data and results are to be maintained for subsequent audits.

Phase 8: Implementation

This involves roll out of the application which has been developed or acquired for the business function based on the current state. The approach for implementation will be decided based on this state. One of the following approaches may be adopted: *(This is discussed in more detail in Chapter 6)*

1. **Cut-off:** Where old system/process is discontinued and new application is made live (operational).
2. **Phased implementation:** Where new application is started in logical phases for different functions.
3. **Pilot:** Where a part function is implemented using new application and based on result either phased or cut-off approach is followed.
4. **Parallel:** Where both the old and new system run simultaneously and based on problem resolution and reliability of processing by the new system, the old system is discontinued.

Role of IS Auditor in implementation phase

- Ensure test plans, test data and rest results are maintained for reference and audit.
- Determine that the formal acceptance has been signed by the project development team, user management, quality assurance, security professional or auditor.
- Verify that the system has been installed according to the organisation's change control procedures.
- Review programmed procedure used for scheduling and running the system along with the system parameters are used in executing the production schedule.
- Review all system documentation to ensure its completeness and verify whether all recent updates from the testing phase have been incorporated.
- Verify that data conversion is correct and complete and is confirmed by the respective user departments before the system is implemented and final user sign-off is obtained.

Phase 9: Support and operations

This is the post-implementation stage following the successful implementation of a new or extensively modified system. This requires, implementation of a formal process that:

1. Provides support and assistance to users in smooth operations and end-user management.
2. There is a mechanism to record, review and implement deficiencies and future changes required.
3. Assess adequacy of the system and projected ROI measurements as per business case.
4. Update project management process based on lessons learned and recommendations for future projects regarding system development.

Role of IS Auditor in maintenance and post implementation phase

Sufficient time should be allowed before post implementation review for the system to stabilize in the live environment. Then only there may be significant problems would have been surfaced.

- Determine that the systems objective requirements were achieved
- Determine if the cost benefits identified in the feasibility study are being measured.
- Review the required controls have built into the system to ensure that they are operating as designed.
- Review error logs to determine if there is any resource or operating problems inherent with the system. Logs may indicate the inappropriate planning or testing of the system prior to implementation.
- Review input and output control balances and reports to verify that system is processing data correctly and completely.
- Evaluate adequacy of procedures for authorising, prioritising and tracking system changes.
- Identify system changes and verify that appropriate authorisation was given to make the change in accordance with organisational standards.
- Review permanent programme documentation to ensure that evidence (audit trail) is retained regarding programme changes.
- Evaluate adequacy of the security access restrictions over production source and executable modules.
- Evaluate adequacy of the organisation's procedures for dealing "emergency" programme changes.
- Evaluate the adequacy of the security access restrictions over the use of the "emergency" logon-Ids.
- Verify existence and adequacy of the records for system changes.
- Evaluate adequacy of the access protection of the maintenance records.

1.4 Changes in SDLC phases

With changing technology and outsourcing practices, the concept of in-house development is slowly phasing out. Organisations prefer outsourcing to get benefit of skills and experience and this also helps management in focusing on core business. These changes in environment require introduction of additional phases to traditional typical SDLC phases.

Introduction of a decision on Make or Buy

Organisations may choose either to develop the software in-house or may opt for buying an available product from market. This is based on analysis performed during first 2 phases, which takes into consideration 'what is being done in industry Based on this information, organisation may choose one of the following three options:

1. Develop software using in-house resources.
2. Hire a third party vendor to develop the software (i.e. outsource software development)
3. Purchase customisable software that is available in market.

Depending upon the decision, organisation follows different path for phases 3 to 6. Figure 1.3 shows these phases for all three decisions. *(Some aspects of phases 3B and 3C are discussed in more details in Chapter 5)*

If an organisation is purchasing or outsourcing the software development, they still need to follow SDLC. Although SDLC refers to Software development, the phases 1, 2, 7, 8 and 9 remains the responsibility of organisation. In case organisations miss out any of the steps, impact will be higher. For example, if organisation misses out identifying key requirements, the purchased or outsourced software may not provide required functionality. Further, making changes after implementation is always costly and time consuming. Similarly if testing is not performed meticulously, software implemented may have bugs.

Phase 3B: Outsource the application development

Organisation may hire a third party firm having necessary skills and experience to develop software specifically for organisation's use. The third party typically follows the phases 3 to 6 for actual development of software. However, the organisation that hires the firm follows phases 3B to 6B. In this situation hiring organisation focuses on requirements for automating the functional processes rather than development model and methodology, but may indicate or stipulate fully or partially preferences. Organisation may follow standard process for selection of third party based on requirements and risk assessment.

Phase 3C: Select and Purchase software available in market

The organisation may select and purchase standard software available in the market (For example, SAP or Core Banking solution or standard accounting software) and configure it as per organisation's requirements. In this case the purchaser does not have control on phases 3 to 6 and must follow entirely different steps to ensure that objectives are achieved.

These phases are described as 3C to 6C. Organisations generally map the available products against the requirements and identify the gaps based on the following criteria:

- If required functionality is not available, can this be configured?
- If required functionality cannot be configured, is it possible to implement a work around?
- Is additional functionality available which was not considered in the requirements?

Based on gap analysis and subsequent availability of maintenance and support (Phase 9) organisation selects the software to be purchased.

Phase 4B: Requirement finalisation

Based on vendor risk assessment, an RFP is forwarded to select vendors along with functional, operational, hardware, software, support and maintenance requirements. Once a vendor is selected, vendor may want to finalise and baseline the requirements so as to ensure that the efforts and associated commercials are appropriate. This is very important since there is always a possibility of changing the requirements during development. Many a times these changes may require rework and efforts might go waste, affecting commercials and project timelines. Hence it is better for both to agree upon the scope and method to deal with changes required when the project is in development stage itself. The organisation floating the RFP is responsible for deciding requirements.

Phase 4C: Request for Proposal (RFP)

Based on the gap analysis results, RFP is forwarded to the suppliers of system to be purchased. In addition to the functionality requirements, it also includes operational, support and technical requirements, and these, together with considerations of the suppliers' financial viability and provision for escrow, will be used to select the system that best meets the organisation's total requirements.

Phase 5B: Contract and SLA

Based on agreed upon requirements a contract containing service levels is drafted and executed by organisation and vendor. This contract generally covers project timelines, deliverables, monitoring development progress, change management (method to handle scope creep), vendor's responsibility for support and maintenance, ownership of software design, data and source code and method for arbitration in case of deputes.

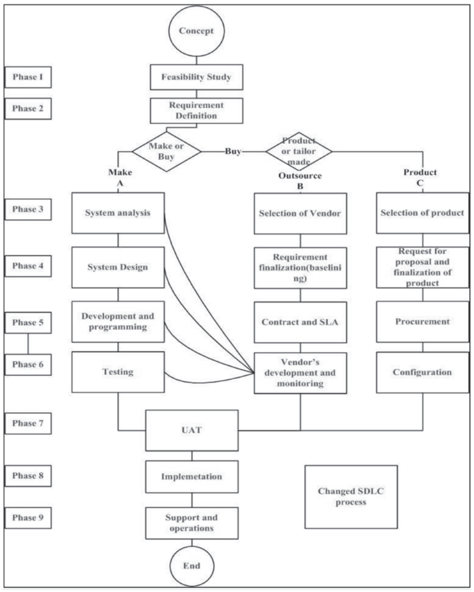


Figure 1.3: Changed SDLC phases

Phase 5C: Purchase

A formal contract for purchase of software is executed which may include cost of software, usage of licences, implementation support, configuration support, user training, helpdesk and maintenance levels to be provided by the vendor.

Phase 6B: Vendor development and monitoring

During this phase vendor executes development phases described in 3A, 4A, 5A and 6A. Organisation should implement a mechanism to monitor development progress to ensure that project is on time. This includes appointing a single point of contact and reporting mechanism by vendor.

Phase 6C: Configuration (Purchased systems)

This involves configuring the system (if it is a packaged system) to tailor it to the organisation's requirements. This is best done through the configuration of system control parameters, rather than changing programme code. Modern software packages are extremely flexible, making it possible for one package to suit many organisations simply by switching functionality on or off and setting parameters in tables. There may be a need to build interface programmes that will connect the acquired system with existing programmes and databases.

Note: Outsourcing development or purchasing and configuring software might reduce the software development efforts, but introduces vendor management and monitoring efforts and also managing risks associated with outsourcing.

Changes in Phase 7: UAT

Although packaged systems are tested by the vendor prior to distribution, these systems and any subsequent changes should be tested thoroughly by the end user and the system maintenance staff. These supplemental tests will help ensure that programmes function as designed by the vendor and the changes do not interact adversely with existing systems. In the case of acquired software, after attending to the changes during testing by the vendor, the accepted version should be controlled and used for implementation. In the absence of controls, the risk of introducing malicious patches/Trojan horse programmes is very high.

Changes in Phase 9: Support and operations from vendor need to be part of contract executed with vendor.

1.5 Secure SDLC

Earlier security was an afterthought for SDLC; normally developers used to check the security related aspects through penetration testing, which requires a lot of rework. For example, if a security related vulnerability, bug or flaw is detected after development then correction of the same will require re-examining all the aspects starting from requirements till coding. This entire exercise will increase the cost and efforts of the project, which sometimes may create a typical situation both for the client and development company. To overcome this issue, latest research studies suggest that the security should be incorporated right from the beginning in the SDLC. Information security trends indicate that embedding security within application development helps in addressing various issues may arise subsequently. For example, when multiple users are expected to access application hosted at central location from different nodes, the application should be able to provide access depending upon the function the specific users has to perform. This requires developers have to design role definition and providing functionality for assigning these roles to different users as defined.

Another example can be in case the application is developed using web based technologies and users are expected to access it using different browsers (like internet explorer, Google chrome etc.), application may not depend upon users to secure their browsers, but embed security within application. In case application is hosted on internet, it is subject to various application level attacks that need to be closed by adopting secure development and coding practices.

The following table describes the additional steps that need to be added to the traditional SDLC phases to make it secure SDLC.

Table 1.1: Security steps in various phases of SDLC

SDLC Phase	Security Steps
Requirement Definition	To identify security requirements including compliance for privacy and data loss. To determine risks associated with security and prepare mitigation plan. To train users on identification and fixing of security bugs.
Design Phase	To ensure security requirements are considered during design phase e.g. access controls for privacy sensitive data. To identify possible attacks and design controls e.g. implementing least privilege principle for sensitive data, and apply layered principle for modules.
Development Phase	To develop and implement security coding practices such as input data validation and avoiding complex coding. To train developers on security coding practices.
Testing Phase	To review code for compliance of secure coding practices. To develop test cases for security requirement testing. To ensure security requirements are tested during testing. To test application for identified attacks.
Implementation Phase	To analyse all functions and interfaces are secured. To perform security scan of application after implementation.
Maintenance Phase	To monitor for vulnerabilities on a continuous basis, To issue the patches for fixing the reported vulnerabilities, accordingly, To evaluate the effectiveness of countermeasures periodically.

1.6 Risk associated with SDLC and mitigation planning

Organisation requiring new software or changing the software initiate a project. The risks of SDLC are:

1. Risks associated with project management (discussed in Chapter-3: project management)
2. Risks associated with SDLC: It is necessary to know these risks prior to undertaking SDLC projects. The objectives are to:
    - Identify risks based on business requirements
    - Discovering methods to respond to risks (accept, avoid, mitigate, transfer)
    - Accepting residual risk and going ahead with the project

Some of the SDLC risks (apart from project risks, which are discussed separately) are:

- Inaccurate requirements definition.
- Inappropriate selection of platform (hardware, operating system, database).
- Compromise on quality and testing resulting in bugs after implementation.
- Missing or inadequate or incomplete documentation.
- Absence of skilled resources.
- Scope creep (changing requirements).
- Poor coding techniques.
- Inadequate QA and QC (including testing inadequacies)
- Lack of proper change control and controls over promotion into production
- Inappropriate processes for development.
- Technical vulnerabilities and how these could materialise and are to be controlled or addressed.

In order to mitigate risks on time, it is best to perform risk assessment during each phase of SDLC. In case of outsourcing and/or purchased software the risk associated with outsourcing and vendor management have to be managed by the organisation.

Mitigation of risks identified need to be planned during project planning and implemented during each phase of SDLC. Possible mitigation plans for risks listed above are given in Table 1.2 below.

Table 1.2: SDLC Risk and Mitigation Plan

Risk	Function
Inaccurate requirements definition	Select prototyping model to finalise the requirements. Meet various stakeholders to confirm identified requirements.
Inappropriate selection of platform (hardware, operating system, database etc.)	Understand technical and performance requirements (e.g. user response time, number of transactions per second etc.) and determine technical specification of proposed platform. In case platform specifications are available (existing hardware); ensure that application development can address requirements. Understand organisation baseline for infrastructure and incorporate in design.

Compromise on quality and testing resulting in bugs after implementation	Ensure standard coding practices are adopted. Provide enough time for building test cases to cover all function, performance and security requirements. Build test cases along with design.
Missing or Inadequate or incomplete documentation	Ensure completion of documentation along with design and development. Standardisation of coding practice helps in this process Ensure documentation experts and technical writers are part of team.
Absence of skilled resources	Consider outsourcing or hiring skilled resources on contract.
Scope creep (changing requirements)	Perform scope base lining. Introduce change management process to evaluate and adopt changes in requirements.
Poor coding techniques or inadequate documentation	Develop and implement standard coding practices.
Inadequate QA and QC (including testing inadequacies)	Plan QA process along with project and development plan. Implement standard coding practices. Implement performance monitoring process.
Lack of proper change control and controls over promotion into production	Perform scope base lining. Introduce change management process to evaluate and adopt changes in requirements.
Inappropriate processes for development	Understand requirements and select appropriate development method suitable for identified requirements.
Technical vulnerabilities	Ensure integration and system testing is performed on target platforms.

1.7 Roles and responsibilities of SDLC

There are certain standard functions (not designations) performed by individual resources during the SDLC process. These roles may be combined, especially for small organisations and some roles may be performed by one individual. Under such cases, IS Auditor has to evaluate conflicts between the roles (segregation of duties). The roles are listed here in brief: (Table 1.3)

Table 1.3: SDLC Roles and Responsibilities

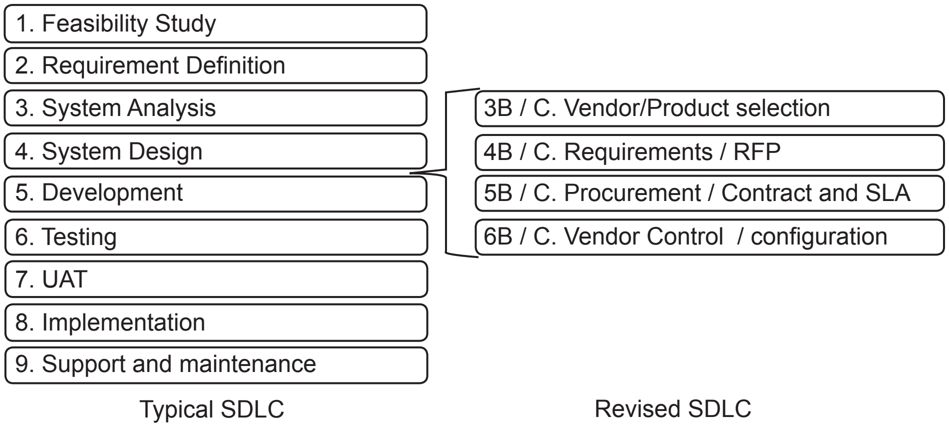
Role	Function
Steering Committee	A steering committee is set up to approve, supervise and direct IT projects, including SDLC. This committee provides overall direction and monitors progress of all IT projects including SDLC projects.
Program Manager	Responsible for all projects that are associated with large IT related programme. E.g. Implementation of ERP across many locations of an organisation. A programme consists of multiple projects.

Role	Function
Project Manager	A project manager is normally responsible for more than one project and liaisons with the client or the affected functions. This is a middle management function. The Project Manager is responsible for delivery of the project within the time and budget.
Systems Analyst	The system analyst also has a responsibility to understand existing problem/system/data flow and new requirements. System analysts convert the user's requirements in the system requirements to design new system.
Module Leader/ Team Leader	A project is divided into several manageable modules and the development responsibility for each module is assigned to module leaders.
Programmers	Programmers convert design into programmes by coding using programming language. They are also referred to as coders or developers.
Database Administrator (DBA)	The data in a database environment has to be maintained by a specialist in database administration so as to support the application programme. The database administrator handles multiple projects; and ensures the integrity and security of information stored in the database.
Quality Assurance Team	This team checks compliance with the SDLC related standards (Documentation, Coding standards, Security requirements etc.) set by the organisation, by project teams on a periodic basis.
Testers	Testers are junior level quality assurance personnel attached to a proj¬ect. They test programmes and sub-programmes as per the plan given by the module / project leaders and prepare test reports.
Domain Specialist	Whenever a project team has to develop an application in a field that's new to them, they take the help of a domain specialist, who understands the business requirements and helps system analyst, designer and programmer in understanding the requirements.
Technology Specialist	IT is developing so rapidly that even IT professionals find it difficult to keep track of all developments, let alone develop expertise. This has resulted in experts in specific technology areas, such as Microsoft technology, Web-enablement and the like.
Documentation Specialist	These professionals are responsible for the creation of user manuals and other documentation.
IS Auditor	As a member of the team, IS auditor can ensure that controls required for process for which application is being developed are included in requirements. IS auditor would be involved in design stage to ensure that the required controls have been included at the design stage.

1.8 Summary

SDLC is an essential aspect of automating business processes using information technology. It has been evolving with changing technology and global proliferation of computers. Today's business heavily depends on IT and any problem faced has multi-fold repercussions. Controlling SDLC process helps organisations in mitigating risks associated with implementation and use of IT. An IS auditor must be aware of phases and key steps of each of the SDLC phases. Although there are various models and methods (Discussed in detail in chapter 4) the basic process discussed in this chapter are common for any model or method used. An IS auditor while auditing SDLC process is not expected to be an expert in all technologies but should always focus on associated risks, assessment of these risks and assess whether the implemented solutions are as per the business objectives.

SDLC phases discussed above are represented as below for easy reference.


1. Feasibility Study
2. Requirement Definition
3. System Analysis
4. System Design
5. Development
6. Testing
7. UAT
8. Implementation
9. Support and maintenance
3B / C. Vendor/Product selection
4B / C. Requirements / RFP
5B / C. Procurement / Contract and SLA
6B / C. Vendor Control / configuration
Typical SDLC
Revised SDLC

1.9 Questions

1. System development life cycle (SDLC) primarily refers to the process of:
   - A. Developing IT based solution to improve business service delivery.
   - B. Acquiring upgraded version of hardware for existing applications.
   - C. Redesigning network infrastructure as per service provider's needs.
   - D. Understanding expectations of business managers from technology.
2. Organisations should adopt programming/coding standards mainly because, it:
   - A. Is a requirement for programming using high level languages.
   - B. Helps in maintaining and updating system documentation.
   - C. Is required for security and quality assurance function of SDLC.
   - D. Has been globally accepted practice by large organisations.

3. Which of the following is main reason to perform User Acceptance Testing (UAT)?
   - A. To train and educate users on features of new solution.
   - B. To confirm from users that solution meets requirements.
   - C. To complete formality of sign-off to mark end of project.
   - D. To finalise the implementation plan for new IT solution.
4. An organisation decided to purchase a configurable application product instead of developing in-house. Outcome of which of the following SDLC phase helped organisation in this decision?
   - A. Requirement definition
   - B. Feasibility Study
   - C. System analysis
   - D. Development phase
5. In which of the following phases of SDLC, controls for security must be considered FIRST?
   - A. Requirement definition
   - B. Feasibility study
   - C. System design
   - D. Implementation
6. IS auditor has been part of SDLC project team. Which of the following situation does not prevent IS auditor from performing post implementation review? The IS Auditor has:
   - A. Designed the security controls.
   - B. Implemented security controls.
   - C. Selected security controls.
   - D. Developed integrated test facility.

1.10 Answers and Explanations

1. A. SDLC primarily focuses on identifying IT based solution to improve business processes delivering services to customers. Other activities may be part of SDLC however, these are IT projects not SDLC projects.

2. C. Adopting coding standards helps organisation in ensuring quality of coding and in minimizing the errors. It also helps in reducing obvious errors which may lead to vulnerabilities in application. A is not true since it is required for all languages, B is partially true but is not main reason. D is not main reason.

3. B. UAT is mainly conducted to confirm from the users and application owners that application meets their requirements. Sign-off is a formality to be completed only if requirements are met. Training and implementation planning are different activities which are not dependent on UAT.

4. B. Make or buy decision is the outcome of feasibility study where technical, economical and social feasibilities are considered.

5. A. Security requirements must be considered during requirement definition. However, the nature of controls to be implemented for security must be considered first during design phase. This will ensure that necessary security controls are built while developing application. Security controls are implemented and not designed during implementation phase.

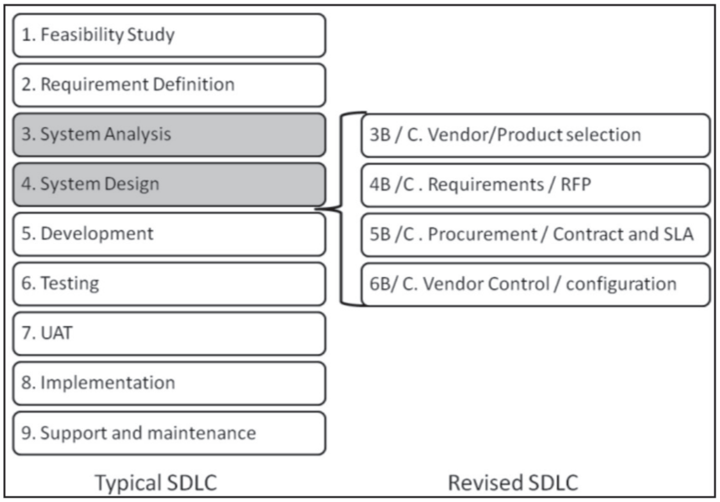
6. D. Active role of IS auditor in design and development of controls affects the independence. Hence, IS auditor cannot perform review or audit of the application system. However, developing integrated test facility within application is not a control, but a facility to be used by auditors in future. Hence, this does not impact independence of IS auditor.

CHAPTER 2: INITIATING SDLC

Learning Objectives

This chapter outlines the steps an organisation follows while initiating a SDLC project. These steps involve understanding of pain points and triggers, conducting feasibility study, analysing benefit realisation, preparing business case for management approval and finalising requirements. These steps help organisations in deciding whether to develop or acquire the software solution.

2.1 Introduction


1. Feasibility Study
2. Requirement Definition
3. System Analysis
4. System Design
5. Development
6. Testing
7. UAT
8. Implementation
9. Support and maintenance
3B / C. Vendor/Product selection
4B /C . Requirements / RFP
5B /C . Procurement / Contract and SLA
6B/ C. Vendor Control / configuration
Typical SDLC
Revised SDLC

This chapter primarily provides information on phase 1 Feasibility study and Phase 2 requirement definition, of typical SDLC (shadowed area). It also discusses the relevant information required to execute these phases within organisation.

Feasibility study focuses on analysing the economic, technical and social aspects of new IT based solution based on requirement definitions and the benefits to be derived from such implementation.

2.1.1 Drivers, pain points and triggers

Use of information technology has prompted organisations for automating data intensive service delivery processes. Initiatives for automation are considered based on business priorities and requirements. These priorities that prompt for selection of automation initiatives are called drivers for initiating SDLC. When there are multiple requests for initiating SDLC projects, organisations may need to prioritise based on the pain points, triggers and the expected benefit realisation.

Pain points

Given below are some of the pain points an organisation may be facing which lead to initiating SDLC process:

- Increasing customer complaints related to service delivery.
- Decreasing level of user satisfaction.
- Loss of business opportunities.
- Problems with existing technology.

- Major changes required in current application.
- Obsolescence of technology affecting service delivery.

Triggers

Pain points and triggers prompt organisations to initiate a SDLC process. Examples of such triggers are:

- Competitors using technology for providing better products or services.
- Increasing market demands.
- Strategic policy changes targeting business growth.
- Availability of technology facilitating new service delivery channels.
- Acquisitions and mergers requiring integration of technology.

2.2 Feasibility study

Feasibility Study refers to evaluating the proposed solution for automating the business process using information technology. It is a process to arrive at a decision, whether the proposed IT solution will deliver business value to the organisation through the IT enabled investments. A feasibility study is generally carried out using various parameters such as technology, direct and indirect investments, expected value materialization period, social acceptability, compliance requirements etc. The feasibility study of a system considers following dimensions:

- **Technical:** Availability of technology for automation.
- **Economic:** Required investments against expected benefits and time required for return on investments.
- **Schedule/Time:** The time required to implement solutions or to go to market.
- **Resources:** Availability of resources like skilled personnel, technology, power and if not available, can they be made available and at what cost?
- **Operational:** The expected improvement in efficiency of service delivery to customers and stakeholders after implementing solution.
- **Social:** Perceived impact on work-life balance on employees.
- **Compliance:** Evaluation benefits of adverse impact on compliance requirements due to proposed solution.

2.2.1 Technical Feasibility

This refers to review of availability of technology and impact of expected obsolescence in future. For example, an organisation which currently deputes its employees to various training courses now proposes to automate training solutions and provide them via intranet portal hosted in-house. The technical feasibility includes evaluation of the following factors:

- Can the solution work on existing infrastructure or does organisation need to acquire new hardware or software? If currently the organisation is not using an automated solution, they may have to invest in acquiring technology and solution.

- Will the proposed system provide adequate responses to inquiries, regardless of the number or location of users? Currently there are many organisations that have deployed such solutions and hence we can conclude that the technical solutions can be made available to meet the response requirements.
- Is system scalable and can it handle the expected business and data growth? There are multiple training courses and those can be deployed using scalable infrastructure.
- Does the technology offer adequate security? Those requirements need to be considered while developing or acquiring solution. However since many organisations have already implemented similar solution, the required security can be embedded.

Some of the technical issues which are to be considered are given in the Table 2.1 below.

Table 2.1: Technical Issues

Design Considerations	Design Alternatives
Communications Channel configuration	Point to point, line sharing, store and forward (multi-drop)
Communications Channel	Telephone lines, coaxial cable, fibre optics, microwave, or satellite
Communications network	Centralised, decentralised, distributed, or local area
Computer programmes	Independent vendor or in-house
Data storage medium	Tape, hard disk, or hard copy
Data storage structure	Files or database
File organisation and access	Direct access or sequential files
Input medium	Keying, OCR, MICR, POS, EDI, or voice recognition
Operations	In-house or outsourcing
Output frequency	Instantaneous, hourly, daily, weekly, or monthly
Output medium	CRT, hard copy, voice, or turn-around document
Output scheduling	Pre-determined times or on demand
Printed output	Pre-printed forms or system-generated forms
Processor	Micro, mini, or mainframe
Transaction processing	Batch or online
Update frequency	Instantaneous, hourly, daily, weekly, or monthly

2.2.2 Economic Feasibility

Use of IT requires financial investments. Economic or financial feasibility refers to evaluation of investments and operating costs as against expected return on investment (qualitative or quantitative). Qualitative benefits refer to better service levels, improved efficiency, increased customer satisfaction etc. Quantitative feasibility refers to direct financial returns like increase in customer base resulting in increased revenue, growth in business, new opportunities etc.

Considering the case of previous example the organisation may not be able to depute all employees for training, however the new solution offers following benefits:

1. 100% of employees can be trained (Improved Compliance) (Qualitative)
2. Employee need not to be deputed for external training (Quantitative)
3. Employees can undergo training as per their convenience (improved productivity) (Partially quantitative)
4. Improved skills and motivation of employees (Qualitative)

These benefits can be weighed against the investments for:

1. Infrastructure (Direct quantitative)
2. Outsourced development (Quantitative)
3. Content development by employees (Partially quantitative)

By comparing both the benefits, cost benefits analysis can be done and then ROI can be calculated.

In short economic feasibility provides answers to the following:

- The cost of conducting a full systems development/acquisition, implementation and operation.
- The cost of hardware and software for the class of applications being considered.
- The benefits derived from new application such as improved efficiency, reduced costs, business growth, and customer and user satisfaction.
- Opportunity cost, i.e. if nothing changes (i.e. the proposed system is not developed/ acquired).

2.2.3 Schedule or Time Feasibility

This refers to the expected time required for the new system to become operational. For example, in the above example, if the contents design and development can be run in parallel, it might take say 6 months, the system design might require one month and the testing and deployment may need another 2 months i.e. totally, the organisation have to wait for 9 months to make the solution operational and start deriving benefits.

2.2.4 Resources Feasibility

This focuses on availability of technical and skilled human resources required for developing/ acquiring and implementing the required solutions. It may also focus on sourcing options and costs thereof. In the example since development shall be outsourced, organisation need to consider the availability of content developers and ensure that they are available when required. In case of non-availability of content developers the project might be delayed. The project shall also be impacted if outsourced vendor fails to provide adequate resources to develop required application in time.

2.2.5 Operational Feasibility

This refers to establishing a process while implementing new solution that will provide ease of operation and also improve efficiency of operations. Some of the questions, which help in testing the operational feasibility are:

- Is there improvement in service efficiency? For example: customers are getting services more efficiently than earlier.
- Is there mechanism that ensures support to users and provides solutions for new system?
- Have the users been involved in planning and development of the project to provide inputs on operations?
- Will the proposed system cause harm? Will it produce poorer results in any respect or area? Will loss of control result in any areas? Will accessibility of information be lost?
- Will individual performance improve after implementation than before?

In the above example, the existing training involves identification of employees, deputing them, providing alternate resource during their absence, cost for training material and trainer. The new process shall not require these operations, however there will be new operations of communicating employees about training, maintaining contents, monitoring that employees attend training on portal, train users on using new system.

2.2.6 Social Feasibility

It is a concern associated with the views of stakeholders like management, workers, employees, customers and suppliers about the use of new solution and acceptance by them. Non-acceptance by majority stakeholders might result in not achieving desired results. In order to ensure acceptance an appropriate change management process involving stakeholders might help. For example the new training system in above example may not be acceptable by employees since it is depriving them a kind of paid holiday from daily routine for attending training. Management may consider offering another incentive to make new solution acceptable.

2.2.7 Compliance Feasibility

Any change in process must not affect the compliance level of the organisations. Any organisation may have to comply with legal, regulatory, contractual and internal compliances. Compliance feasibility is concerned with whether there will be any conflict between a newly proposed system and the organisation's compliance obligations. For example, a revised training system should comply with all applicable statutes about financial and statuary reporting requirements, as well as the company's contractual obligations.

2.2.8 Internal Controls

Automation might require changes in existing internal control environment. These aspects must be considered while considering the feasibility aspects as the assurance on internal controls is accountability of management. It is necessary to inform management about the changes in internal controls along with compliance requirements and associated risks.

2.3 Organisational methods for benefit realisation

After completing feasibility study, the next step is preparing business case and presenting it to management for approval. Assessment of the benefits realisation is a key component of business case. This is extended aspect of economic feasibility where the granular analysis of qualitative and quantitative benefits is considered. This section discusses about the various aspects of benefit realisation.

Benefits realisation of projects considers major factors such as cost, quality, development/ delivery time, reliability and dependability. Project sponsor perform a comprehensive study and evaluate which factors are qualifying and then compare those factors with strengths, weaknesses opportunities, associated risks and competencies of services available to develop/acquire, implement and maintain systems.

Benefits Realisation Techniques

Business benefits from SDLC projects are determined based on the changes in strategy, business objectives, changes in technology etc. The realisation of benefits has become a more complex task due to evolution of IT applications, automation of work using embedded technology, business transformation through information management. Benefits do not just happen when the new technology is delivered; they occur throughout the business cycle. A planned approach to benefits realisation with long-term cycles that consider the total benefits and total costs throughout the life of the new system. Although it is possible that benefits may not come exactly as planned, organisations have to keep checking and adjusting the strategies.

Benefits realisation is a continuous process that must be managed just like any business process and this requires organisations to:

- Describe benefits realisation, for example customer growth, revenue growth due to technology.
- Assign measure and target for each aspect i.e. percentage and number to be achieved/ expected.
- Establish a tracking/measuring process for expected benefits i.e. have we achieved expected numbers? If not what needs to be done? Were the assumptions correct?
- Document the assumptions against which the benefits are assessed.
- Define responsibilities for realisation i.e. which organisational resources can be diverted due to automation to achieve benefit realisation.
- Validate benefits predicted against current data, past achievement, industry experience and so on.
- Plan the benefits to be realised. For example automation is expected to achieve growth, but customers are not aware of changes in service process/levels.

Organisation wide benefits realisation studies should be aggregated and lessons learned should be used to fine-tune the organisation benefit realisation process.

Benefits realisation often includes a post implementation review within 6-18 months after the implementation of systems. Time must be allowed for initial technical problems to be resolved and for the project benefits to accrue as users become familiar with the new processes and procedures.

Benefits realisation must be part of management practice of projects. There must be business sponsorship of projects. Project governance structures should involve the project and the functional line organisation. All governance decisions about the project should be driven through the business case, and there must be periodic review of benefits.

IS auditor should understand how the business defines and monitors return on investment (ROI) for SDLC projects. Not achieving expected benefits or failure to monitor and take corrective actions in meeting ROI objectives, may point to weakness in benefit realisation process and related project management practices

2.4 Business case development

A business case is normally derived from the benefit realisation plan and feasibility study. A business case provides the information required for an organisation to decide whether the SDLC project should be undertaken and if approved, becomes the basis for a project execution and assessment. An IT project can have different scope and objectives. A project may be initiated for development of a new application system (SDLC) or for an investment in new infrastructure (partially touch SDLC), or for major modification in existing application and technology (Full or partial SDLC). A business case identifies and recognises the achievement of business benefits to be derived from project.

Depending on the size of IT investments, organisations may choose to prepare business case at high level for programme and proceed further to develop business cases for underlying projects. The initial business case would normally be derived from a feasibility study. The feasibility study will normally scope the problem, identify and explore a number of solutions and make a recommendation on what action to take. Part of the work in identifying options requires outlining and calculation of benefits. A business case contains a comparison of costs and business benefits for different options for proposed solutions.

The business case should contain sufficient details to describe the justification for the project along with the reasons and should provide justification for the question, "Why the project should be approved? "The business case is a key element of the decision making process throughout the life cycle of project. If at any stage the business case is thought to no longer be valid, through increased costs or through reduction in the anticipated benefits, the project sponsor or steering committee shall refer to business case to arrive at a decision on whether the project should be terminated or should continue. If the business case changes during the course of an IT project, the project should be reapproved through the departmental planning and approval process.

While performing audit of benefit realisation IS auditor has to understand and validate the methods identified and adopted by the organisation for determining benefit realisation and also the process for monitoring the benefits through put project life cycle. **These reviews happen after achieving predefined milestones (i.e. mid-term project review) and post-implementation review is generally conducted after 6 (or more) months.**

2.5 Business Requirements Identification

Once the business case is approved, the project is initiated. The primary part of this process is to prepare detailed requirements analysis. This is particularly required in case of acquiring of the software or outsourcing of software development. In between approval for business case and initiating the project, organisation should consider undertaking the detailed requirement analysis.

If it is decided to outsource the development, the vendor selected may require detailed function requirement specifications in order to arrive at efforts required and hence the cost. If organisation does not perform this step, the vendor shall perform it, to arrive at appropriate project costing. This section describes the various aspects of such analysis.

2.5.1 Techniques for requirements gathering

1. Understanding Requirements

The following are major activities under this section:

- Identifying stakeholder expectations. Stakeholders generally are not aware about the type of information required and may end up in providing high-level requirements. For example, for a payroll system, stakeholders may expect employee appraisals to be part of system but may assume that it might be considered. However since employee appraisal requirements are generally classified under HR Management project manager may ignore them.
- Analysing requirements by discussing with stakeholder to detect and correct gaps in understanding and trim down extended expectations and determine priorities.
- Verifying that requirements are complete, consistent, unambiguous, verifiable, modifiable, testable and traceable.
- Resolving conflicts between the requirements set and the resources that are available.
- Identifying how the system needs to interact with its environment. For example, in a financial accounting system, requirement may state that the system must scan the supporting bills/documents and attach it to the financial transaction. Or in a payroll system an interface with internet banking may be required to be built to generate data for direct credit to employee accounts.

2. Study of History, Structure and Culture

The study of the history of systems in an organisation gives an idea about the types of systems that have been extremely useful, issues that have not been addressed over a period and new issues that require attention. It is essential to understand organisational structure and culture as the solutions that are not consistent with the culture often fail.

3. Study of Information Flows

In designing a new system, it is necessary to study the existing system. This involves study of the business processes, underlying activities, actors that perform these activities and resources required. This provides information on the essential aspects and controls while designing the new system and also helps in identifying changes required in existing process, without affecting internal control/compliance mechanism. Sometimes it is possible to collect user's expectation and requirements; however sometimes it also provides insight on what else is required to be done (e.g. prototyping) during project design phase to ensure that user's requirements are collected properly. The information gathering is a series of activities such as:

- Meetings with users at various levels
- Observation of users at work
- Forms, documents, registers being handled in the system
- Verbal interactions done by users to collect and disseminate the information

2.5.2 Types of requirements

Software solution has two types of requirements viz. functional requirements and non-functional requirements. Many times functional requires are prepared and presented by users however non-functional requirements has to be derived based on the analysis of functional requirements. A non-functional requirement consists of technical requirements, performance requirements, operational requirements and security requirements.

Functional requirements

These are primarily requirements related to service to be provided by business to customers. For example if a tour operator intends to offer online booking service to its customers, the list of functions that can be offered online shall be described in functional requirements, say user interface requirements and may include feature that once user visits home page there need to be an options for information on tours available, booking options with calendar, availability of seats, payment options, collecting and storing information on booking, forwarding e-mails and SMS on confirmation. Operational requirements may include back end operations of ensuring and confirming booking, arranging travel and stay as required, building tie-ups with travel agents, hotel managements and establishing communication process. However the business users may not be able to comprehend the technical requirements or define performance requirements beyond 'within what time the customer must get confirmation and communication'.

Non-functional requirements

These requirements cover what technology to be used and why? What are performance expectations e.g. CPU usage, number users, expected number of hits, user experience in getting next screen after click, designing frames so that user need not have to click multiple times etc. These requirements are generally captured during interview and information gathering. For example in above example if the tour operator expects abound 1,000 bookings per day, there could be high number of hits, since all hits may not materialize in booking. The implemented technology must support (e.g. multiple servers with load balancing) expected high number of hits or else the application may fail to serve. Similarly user may expect response from tour operator for any query or booking with in specific time and the application should be able to (if automated) respond within that time or should be able to schedule response or generate alert for manual response. The technical requirements may also include providing personal communication using telephone or VOIP. In such a scenario, the technical requirements need to be defined to meet requirements. Once the requirements are defined the organisation may be able to decide on the right option which could be: in-house development or acquisition of solution or outsourcing the development.

Requirements Engineering (RE) Process

RE is one of the most important areas of System Development. During this stage, the customer and Requirement Engineers come to an agreement as to 'what constitutes the software to be developed'. This is a critical stage because anything that is (or is not) resolved at this time, will be carried down through the rest of the System Development lifecycle. Good RE process is therefore essential for successful system development. RE process have several phases, which facilitate to understand the customers' needs, define the system requirements and constraints, analyse them, evaluate their feasibility, determine customers' real need, validate requirements specification and manage the requirements. Basically, there are five major phases of RE, which are explained here:

1. **Elicitation:** The RE process is normally considered as the process of finding out 'what are the real needs of the customers as well as of the system'. It also includes activities to explore 'how the software can meet the stakeholders' goals' and 'what alternatives might exist'.
2. **Analysis and Negotiation:** This phase consists of a set of activities aimed to discover problems within the system requirements and achieve agreement on changes to satisfy all system stakeholders. If an analyst discovers some problems with the requirements during the analysis phase, such requirements are referred back to the elicitation phase. This process is related to the requirements that are incomplete, ambiguous and/or conflicting. Negotiation part is known as 'the process of discussing conflicts in requirements and finding some compromise which all of the stakeholders can live with'. The principle of this process should be objective, where the judgments and the compromise for the system requirements should be based on technical and organisational needs. All the conflict requirements identified during the analysis process should be negotiated and discussed individually with the stakeholders in order to resolve the conflicts.
3. **Documentation:** Once the requirements have been analysed, it is important to record them in order to make them formal through proper specification mechanism. During this phase, the team organizes the requirements in such a way that ascertains their clarity, consistency, and traceability etc. This phase is extremely important because often 'the document produced during specification is what the rest of the development stages will be based upon'.
4. **Validation:** This phase ensures that models and documentation accurately express the stakeholders' needs along with checking the final draft of requirements document for conflicts, omissions and deviations from different standards.
5. **Management:** This phase of requirements lifecycle is similar to the maintenance of a software system. Some of the most important maintenance tasks during this phase include the updating of the requirements as well as the degree of evolution support that the approach provides.

2.6 Summary

Once the SDLC project is initiated it is critical to ensure that:

- Feasibility study is complete and based on realistic and plausible assumptions.
- Benefit realisation covers both qualitative and quantitative analysis of expected benefits from the project.
- Benefit realisation process is defined and contains measurable achievements. It also contains what needs to be measured during project execution.
- Business case is prepared and approved and covers sufficient details so as to form a project baseline document.

The next step is to execute the project. For this the organisation must have a uniform project management practices, methodology and process. The next chapter covers the various steps required for project execution.

2.7 Questions

1. An organisation has implemented an IT based solutions to support business function. Which of the following situation shall indicate the need to initiate SDLC project?
   A. Vendor has launched a new hardware which is faster
   B. Organisations has unused surplus budget for IT
   C. Regulators have requested additional reports from business
   D. Competitor has launched an efficient IT based service
2. A “Go or No Go” decision for SDLC project is primarily based on:
   A. Feasibility Study
   B. Business case
   C. Budget provision
   D. Market situation
3. Which of the following is the primary reason for organisation to outsource the SDLC project? Non-availability of:
   A. Skilled resources
   B. Budgetary approvals
   C. Security processes
   D. Infrastructure
4. Which of the following is an example of addressing social feasibility issue in SDLC project?
   A. Organisation decides to use existing infrastructure
   B. Beta version of application is made available to users
   C. Configuration of purchased software requires more cost
   D. Allowing employees to access social media sites

5. Which if the following is not an indicator to assess benefit realisation for internal application software developed in-house?
   - A. Increase in number of customers because of new application
   - B. Decrease in audit findings related to regulatory non-compliance
   - C. Reduced number of virus attacks after implementing new software
   - D. Increase in productivity of employees after implementation
6. Which of the following requirements for an application to be developed for use by human resource department are non-functional requirements? The application should:
   - A. Capture the employee data at the time of hiring
   - B. Provide option to all only to edit own information
   - C. Capture performance details required for appraisals
   - D. Use relational database as backend to store data

2.8 Answers and Explanations

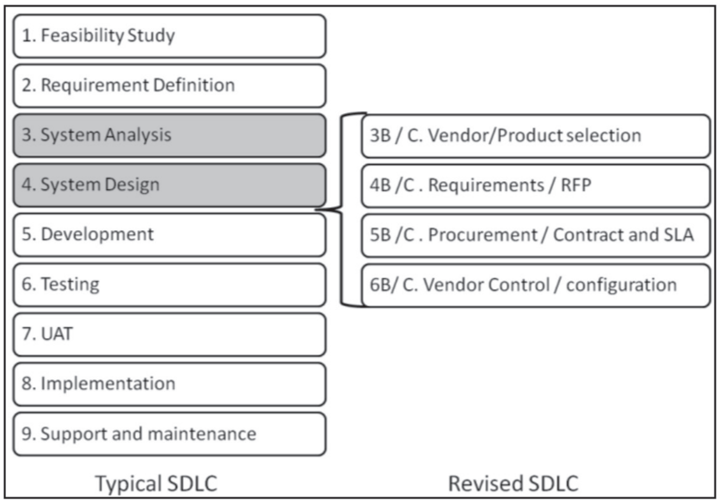
1. D. When a competitor launches new IT based efficient service, it becomes necessary for management to consider the impact in market place and in order to remain in competition should provide similar or better services. Options A and C may not require SDLC since it can be adopted with change management process. B may help in deciding for D, but is not the reason for initiating SDLC project.
2. C. Business case is a document that narrates all aspects including benefit realisation, cost and effort estimates, outcome of feasibility study, available budget. That helps management in decision on the need of the SDLC project.
3. A. Non availability of skilled resources required for application development is primary reason for outsourcing the SDLC project. Other reasons can be addressed. i.e. (B) budget can be made available, (C) security processes can be established. (D) Infrastructure can be acquired, depending upon design of new application and hence it is not a reason.
4. B. In order to ensure the acceptability by users, beta version of solution is made available to users. Based on feedback changes are made so that the solution can be socialized. Option A addresses technical feasibility, Option C addresses economic feasibility. Option D addresses IT policy that has nothing to do with SDLC.
5. C. Since the application is for internal use and developed in-house it has nothing to do with reduction in virus attacks. This can stet benefit realisation for anti-virus solution.
6. D. Specification of technology to be used are non-functional requirements.

CHAPTER 3: PROJECT MANAGEMENT FOR SDLC

Learning Objectives

This chapter focuses on proving insights on System Development project management. This includes initiation of programme/project, establishing project management methodology, defining objective, project risk management, planning, resource management, monitoring and controlling project, managing changes, closing project and tools and techniques required for software project management.

3.1 Introduction


1. Feasibility Study
2. Requirement Definition
3. System Analysis
4. System Design
5. Development
6. Testing
7. UAT
8. Implementation
9. Support and maintenance
3B / C. Vendor/Product selection
4B /C . Requirements / RFP
5B /C . Procurement / Contract and SLA
6B/ C. Vendor Control / configuration
Typical SDLC
Revised SDLC

This chapter primarily provides information on phase 3: System Analysis and Phase 4: System Design of typical SDLC (shadowed area). However the detailed discussions on methods of system analysis and design are also discussed in Chapter 4.

This chapter provides inputs on basic understanding of project management as executing SDLC is a project till new solution is operational and organisation starts deriving benefits.

IS auditor primarily should be looking for control aspects in analysis and design hence the focus is on project management where the controls aspects are planned at the design stage.

3.2 Project management frameworks

Project is initiated once it is approved. In order to ensure that the project is successful, i.e. it meets its predefined objectives and delivers value, the organisation must adopt effective and efficient project management practices. Without project management practices, tools and control frameworks, it is not possible to manage all the relevant aspects like planning, scheduling, resource management, risk management, sizing and estimation of efforts, milestone achievements, quality, deliverables and budget monitoring, of a large project. IS auditor must understand the need for a project management framework within the organisation, and associated elements required to establish a standard methodology. This chapter covers the various project management practices and how these are executed within the organisation.

There are many approaches for project management defined by various professional bodies. The most commonly used approaches are:

- Project Management Body of Knowledge (PMBOK®) version 5, i.e. IEEE standard 1490 from the Project Management Institute (PMI),
- Projects in a Controlled Environment (PRINCE2™) from the Office of Government Commerce (OGC) in the UK, and the International Project Management Association (IPMA).

Since there are significant differences in scope, content and wording in each of these standards, an auditor has to become familiar with the standard adopted by auditee organisation, prior to involvement in project. Although each project management approach has its own pros and cons, several elements are common across all project management methodologies. Some are focused on software development, others have a more general approach; some concentrate on a holistic and systemic view, others provide a very detailed workflow including templates for document creation

3.3 Key concepts of project management

Project is a temporary endeavour undertaken to generate defined outcome (like creating a service or product). Temporary need not be short, what it means is it has predefined beginning and end. For example a project can be initiated to build a housing complex, that is completed once tenants have occupied it, or it can be for building infrastructure, or designing a new product, e.g. Tata motors designing the Nano car. The project is closed once the expected outcome is delivered or results are achieved or if the project becomes technically or economically unviable. In short projects can be initiated for any reason. Developing Software and deploying it can be a project or depending upon size, it can be group of projects.

A project management refers to the practice of managing a project. Management may not want to initiate a project as it involves providing resources and waiting till the end to get deliverables. Management has to monitor the progress of project and intervene periodically to ensure that the project finally achieves the defined objectives.

Project management practice is a set of multiple processes grouped in five major process groups:

1. Project initiation
2. Project planning
3. Project execution
4. Project controlling and monitoring
5. Project closing

Although these processes are grouped they are not executed in sequence, Process groups under project planning, project execution and controlling are executed in iteration. (Figure 3.1)

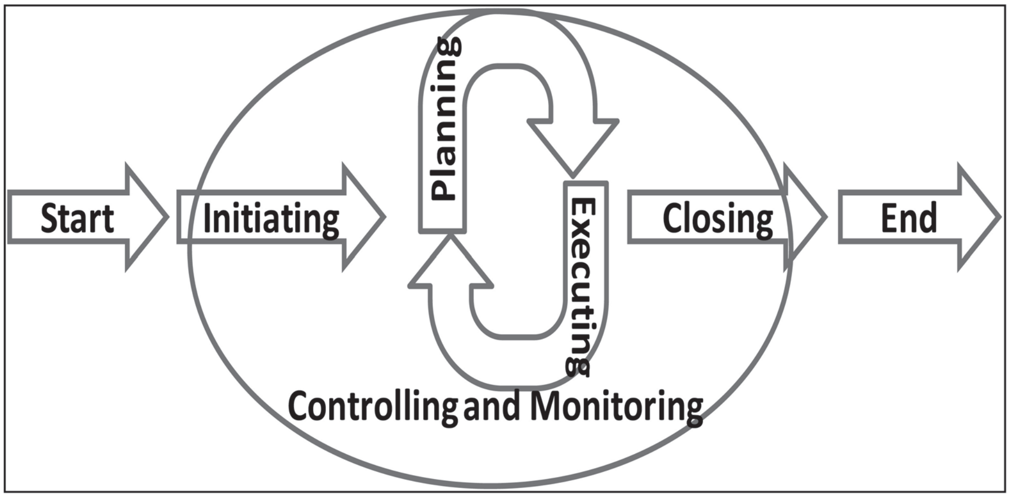


Figure 3.1: Project process groups

Each group consists of various processes, however all processes may not be applicable to all projects.

Project initiation group consists of mainly processes related to developing project charter based on scope of project. In SDLC project it is business case (discussed in previous chapter), Identifying beneficiaries and stakeholders of project.

Project planning consists of processes related to developing project execution plan, finalising requirements, defining work breakdown structure and modules to be developed, estimating efforts and cost, resource planning, risk management, procurement planning and plan for communications with stakeholders.

Project execution consists of processes related to direct project teams, ensuring quality assurance and testing, managing requirements and changes in requirements, ensuring timely procurements and manage resources.

Project controlling and monitoring consists of processes related to monitoring risks, scope creeps, quality of deliverables, costs and budgets, performance reporting.

Project closing has processes for handing over deliverables or terminating project.

SDLC project management is further discussed in ensuing sections.

3.4 Programme and project management and organisation

3.4.1 Portfolio/Programme Management

A programme is a group of projects and/or time-bound tasks that are linked together through common objectives. It may share a common budget and can have intertwined schedules. Like projects, programmes have a limited time frame (start and end date), predetermined budget, defined deliverables/outcomes and sometimes organisational boundaries. A programme is more complex than a project and many times consists of multiple projects. For example implementing

ERP at all plants can be a programme consisting of multiple projects for implementing ERP at each plant. (Figure 3.2 explains the relationship of portfolio, programmes and projects)

A portfolio is group of all projects/programmes (related or unrelated) being carried out in an organisation at a given point in time.

A project/programme management office (popularly referred as PMO) controls and manages Portfolios, programmes and projects. PMO also governs the processes of project management but not involved in management of project content.

The programme management includes management of:

- Programme scope
- Programme financials (costs, resources, cash flow, etc.)
- Programme schedules
- Programme objectives and deliverables
- Programme context and environment
- Programme communication and culture
- Programme organisation

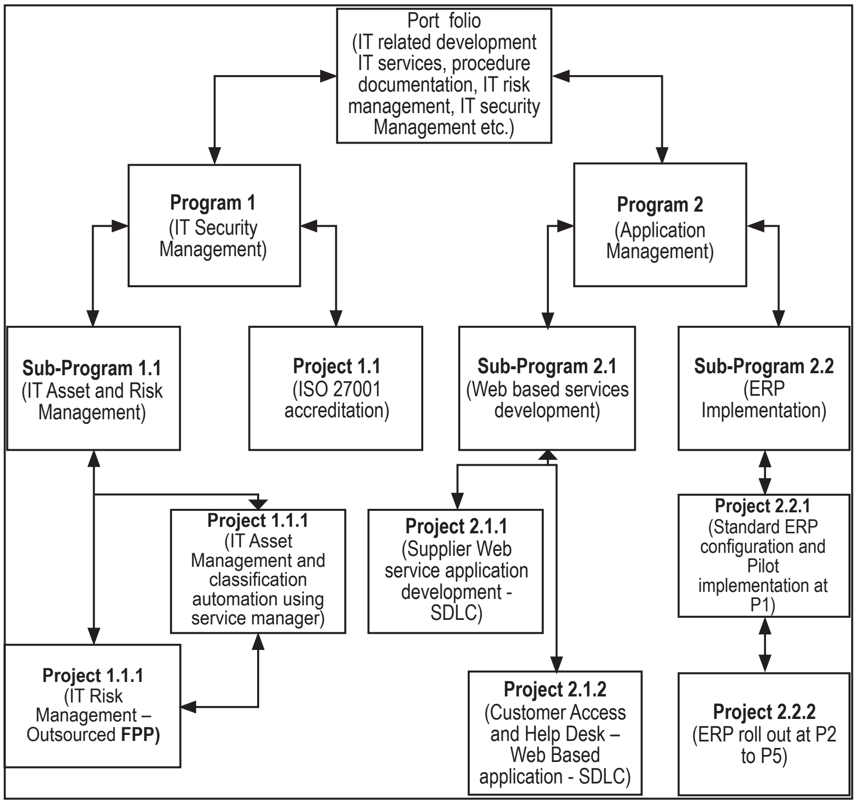

Port folio (IT related development IT services, procedure documentation, IT risk management, IT security Management etc.)
Program 1 (IT Security Management)
Program 2 (Application Management)
Sub-Program 1.1 (IT Asset and Risk Management)
Project 1.1 (ISO 27001 accreditation)
Sub-Program 2.1 (Web based services development)
Sub-Program 2.2 (ERP Implementation)
Project 1.1.1 (IT Asset Management and classification automation using service manager)
Project 2.1.1 (Supplier Web service application development - SDLC)
Project 2.2.1 (Standard ERP configuration and Pilot implementation at P1)
Project 1.1.1 (IT Risk Management – Outsourced FPP)
Project 2.1.2 (Customer Access and Help Desk – Web Based application - SDLC)
Project 2.2.2 (ERP roll out at P2 to P5)

Fig 3.2: Portfolio, programme and project

3.4.2 Programme/Project management Organisation forms

A project may be considered as a group of complex tasks executed by a temporary organisation. However depending upon the nature of business, from a project management perspective, organisations can be categorised as follows:

- **Functional organisation that is influenced by the projects:** These are business organisations that are involved in production of goods and services. Projects are undertaken to support the functional activities. For example, a manufacturing organisation may want to automate administrative processes (like finance, HR, pay roll etc.) using IT. In such organisations, project manager has only a staff function without formal management authority. The project manager is only allowed to advise peers and team members as to which activities should be completed. In such organisation project team consist of staff that report to functional manager, except for the purpose of project activities assigned, reports to project manager.
- **Projectile organisation:** These are pure project organisations that execute projects. For example, an infrastructure development organisation or consulting organisations that executes projects. In such organisations project manager has formal authority over those taking part in the project. Often, this is bolstered by providing a special working area for the project team that is separated from their normal office space.
- **Matrix project organisation:** The organisation that provides product and services and also executes projects. Most IT companies fall under such categories where these organisations undertake project to manage business functions for other organisations and also executes projects for customer organisation. In such organisation, management authority is shared between the project manager and the department heads

IS auditor has to understand these organisational forms and their implications on controls in SDLC project management activities.

3.5 Project Initiation

Whenever a business entity decides (i.e. stakeholders in the business or senior management) to undertake computerisation, a project will have to be initiated. Some examples of a formal project initiation are:

- A new business application is required to be developed to address a new or existing business process e.g. HR management system, billing system, order processing etc.
- Adoption of a new technology invented or available becomes advantageous to the business e.g. Internet based advertising for an advertising company.
- The application software to be developed, is expected to rectify the present problem related to existing business e.g. computerisation of college admissions.
- The application software to be developed, is expected to rectify the present problem related to existing technology e.g. migrating from text-based computerized system to GUI based system as in case of old COBOL/XBASE based distributed banking to RDBMS based Core Banking system.

A project may be initiated from any part of the organisation, including the IS department. A project is time bound, with specific start and end dates, a specific objective and a set of predetermined deliverables. Once a project is initiated a project sponsor and project manager is appointed to execute the further activities including gathering the information required to gain approval for the project to be created. This will often be compiled into terms of reference or a project charter that states the objective of the project, the stakeholders in the system to be produced, and the project manager and sponsor. Approval of a project initiation or project request is authorisation for a project to begin.

During project initiation, the project manager performs several activities that assess the size, scope, and complexity of the project, and establishes procedures to support subsequent activities. The major activities to be performed in the project initiation are:

- **Establishing the project initiation team:** This activity involves organising an initial core of project team members to assist in accomplishing the project initiation activities.
- **Establishing a relationship with the customer:** A thorough understanding of the customer builds stronger partnerships and higher levels of trust.
- **Establishing the project initiation plan:** This step defines the activities required to organise the initiation team while it is working to define the scope of the project.
- **Establishing management procedures:** Successful projects require the development of effective management procedures.
- **Establishing the project management environment and project workbook:** The focus of this activity is to collect and organize the tools that will be used while managing the project and to construct the project workbook. For example, most diagrams, charts, and system descriptions provide much of the project workbook contents. Thus, the project workbook serves as a repository for all project correspondence, inputs, outputs, deliverables, procedures, and standards established by the project team.

Many organisations that follow standard process for project management prepare a formal Project Initiation Report that is presented to senior management or Board of Directors. Once accepted this becomes formal charter for the project and triggers next phases of SDLC.

3.5.1 Project management methodology

IT projects are divisible into pre-defined phases *(SDLC phases are described in Chapter 1 and various methods and models are discussed in Chapter 4)*. The project management process begins with the project charter and ends with the closure of the project. Since the project management can be complex, like other business processes, it also requires a standardised approach. Organisations may adopt standard processes prescribed by globally accepted standards developed by organisations like PMI or can define a project management process within organisation based on such prescribed standards.

Organisations following a standard project management process have higher possibility of completing projects in time, within budget and deliverables meeting with expected quality. The following section explains general project management practices used by various organisations.

3.5.2 Project Context and Environment

Organisation may be running several projects at the same time. These projects need not be SDLC projects or IT projects. At organisation level the relationships between these projects have to be established to identify common objectives for the business, which is a function of a project portfolio management and/or a programme management. This helps in consolidating common activities (e.g. identify and manage risks) and managing of common resource requirements. (Refer Figure 3.1)

A context of the project may be determined based on:

- Importance of the project deliverables to organisation's objectives
- Connection between the organisation's strategy and the project outcome
- Relationship with other projects
- Priority based on the business case

In addition while considering the time context of the project, following aspects must be considered:

- Start and end time of the project, particularly if it is expected that the outcome of project has linkages to other projects.

The objective is to determine whether all relevant environments for the project, which will have a significant influence on overall project planning and project success, have been considered.

3.5.3 Project Communication and Culture

Success of project depends upon timely communication with stakeholders and affected parties. This can be achieved by:

- One-on-one meetings
- Kick-off meetings
- Project start workshops
- Periodic reporting

Communication helps in obtaining co-operation from all team members and buy-in from stakeholders. One of the major activities for project manager during execution of project is to develop and execute communication plan so as to inform issues, concerns, if any and to report project progress.

3.5.4 Project Objectives

Primary objective of project is to deliver the defined outcome/deliverables/product in time, within budget and of desired quality. The measurement of success depends upon clearly defining results that are specific, measurable, attainable, realistic and timely (SMART). The main objectives of project are always directly coupled with business expectations. Additional objectives are objectives that are not directly related to the main results of the project but may contribute to project success (e.g., business unit reorganisation in a System Development project).

A commonly accepted approach to define project objectives is to start with a work breakdown structure (WBS) with each work module having its own objectives derived from main objectives. The WBS represents the project in terms of manageable and controllable units of work and forms the baseline for cost and resource planning. Detailed specifications regarding the WBS can be used to develop work packages (WP). Each WP must have a distinct owner and a list of main objectives, and may have a list of additional objectives. The WP specifications should include dependencies on other WPs and a definition of how to evaluate performance and goal achievement.

A task list is a list of actions to be carried to complete each work package and includes assigned responsibilities and deadlines. The task list aids the individual project team members in operational planning and scheduling, that when merged together forms a project schedule. Project schedules are work documents containing the start and finish dates, percentage completed, task dependencies, and resource names of individuals planned to work on tasks.

3.5.5 Project Management Practices

Every organisation uses project/programme to implement new concepts, changes, business strategies etc. Collective knowledge of executing projects may be used to execute the projects; however it is a prudent for the organisation to adopt a standard project management practice, across entire organisation. Many organisations prefer to adopt the practices based on global standards/best practices e.g. PMBOK, Prince2 etc.

Project management is the application of knowledge, skills, tools and techniques to a broad range of activities to achieve organisational objectives. For example: meeting user requirements by developing/acquiring new software within budget and timelines. Project management practices consist of defined processes for initiating, planning, executing, controlling and closing a project.

A successful project planning is a risk-based management process that is iterative in nature. Project management practices for SDLC projects also provide standards for systematic quantitative and qualitative approaches to software size estimating, scheduling, allocating resources and measuring productivity. There are numerous project management tools available (e.g. MS project) that can be adapted to implement techniques to assist the project manager in controlling the time and resources utilised during execution of project.

3.6 Project Planning

To plan and control SDLC projects, project manager needs to determine:

- The various project tasks and management tasks that need to be performed to develop/ acquire and implement business application system.
- The order in which these tasks should be performed.
- The estimated duration for each task.
- The priority of each task.
- The IT resources, available, transferred/loaned or to be acquired to perform these tasks.
- Budget or costing for each of these tasks. This can be notional for internal resources and monetary for outsourced projects.

In complex projects the planning is dynamic and has to be reviewed/adjusted at the beginning and end of each project phase. This is to ensure that resources are available, quality of work during earlier phase has been as expected (i.e. no rework is required, or if required adjusting the plan by considering the delay and so on.)

There are some techniques like programme evaluation review technique (PERT), critical path method (CPM), Gantt chart etc., that are useful in creating and monitoring project plan. The major activities which are performed during project planning are:

- Measure the development efforts. (Different software sizing techniques are discussed in section 3.8.)
- Another activity is to identify resources (e.g., people with requisite skills, development tools, facilities) for software development.
- Budgeting is next activity. Although overall budget for the project has been allocated at high-level during business case development, project manager need to prepare granular budget for monitoring. This is done by considering the cost for each resource and their expected use. For example group of testing professionals might be required in the project, however they need not be available from the beginning of project and thus can be inducted (On boarded) at a later date thus optimising the cost associated with their release from another project.
- Scheduling and establishing the time frame is another activity. While budgeting involves adding up the cost for human and machine resource usage involved in each task, scheduling involves establishing when these resources are required in the project. This is achieved by arranging tasks according to:
  - The logical sequential and parallel tasks relationship and determining earliest start date.
  - Based on estimated efforts (section 3.7) for each resource arriving at latest expected finish date.
  - Schedules are presented using PERT, CPM diagrams and Gantt Charts. *(Discussed in section 3.7.)*

3.7 Project Controlling

The controlling activities of a project include:

- Management of scope
- Monitoring of resource usage
- Risk management.

It is critical to ensure that new requirements for the project are documented and, if approved, appropriate resources are allocated. Control of changes during a project ensures that projects are completed meeting stakeholder requirements of: time, use of funds and quality objectives. Stakeholder satisfaction should be addressed with effective and accurate requirements capture, proper documentation, baselining and skilled steering committee activity.

During mid-term project review IS auditor should focus on project planning and controlling activities to ensure that these are not deviating from primary objectives of the project.

3.7.1 Management of Scope

Quite often, it is noticed that the majority of the SDLC projects suffer from "Scope creep". This happens particularly when the requirement analysis is incomplete or the dynamic nature of business environment forces users to include these requirements and all these requirements cannot be put on hold. The scope creep affects the project planning seriously. This can be controlled by:

- Baselining the requirements before project planning.
- Establishing process for change management to decide which requirements must be included during development and how it is affecting the project time, cost, quality and outcome. Change management process must define who can request for change, how a formal change request be made, what it should contain and the reasons for the change. For complex deliverables, it is best to document the work breakdown structure.
- The project manager then assesses the impact of change request on project activities, schedule and budget.
- A change advisory board is appointed to evaluate change requests and decide on approving changes.
- If the change is accepted, the project manager should update the project plan.
- The updated project plan must be formally confirmed by the project sponsor—accepting or rejecting the recommendation of the change advisory board.

3.7.2 Resource management

Monitoring resource usage in project execution is the process to control budget and ensure that cost plan is on track .Budget and project plan assumes certain productivity of resources. For example, if a programme development is expected and hence planned to take 16 person-hours, then it is supposed that the resource being deployed is capable of finishing that task in 16 person-hours with expected quality level. (Using coding standards might help in improving productivity). Whether this is actually happening can be verified using earned value analysis (EVA).

Earned Value Analysis consists of comparing expected budget till date, actual cost, estimated completion date and actual completion at regular intervals during the project. In above example the programme development is expected to take two working days, with eight hours spent each day. At the end of first day cost is as per budget but EVA cannot be determined unless 50% or more work has been completed. Alternate is to get information on how much time is required to complete remaining programme. If the answer is 8 hours the project is on track, if it is less resource might be idle and if it is more, the project might be delayed. In short at the end of first day resource spent is according to budget, but the "earned value" will be based on remaining time to complete the task. (Figure 3.3)

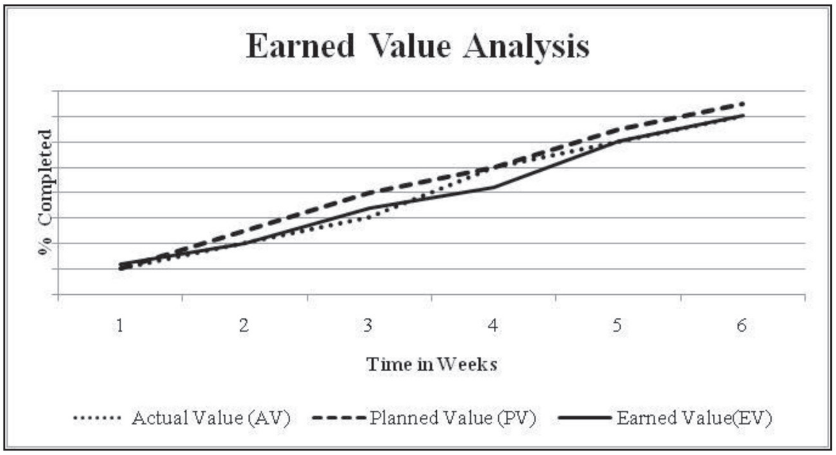


Figure 3.3: Earned Value Analysis

3.7.3 Project risk management standards and methods

Project management practices require the project manager must adopt the risk management frameworks. PMBOK of PMI specifies following activities for project manager:

Project Planning Phase

- Plan Risk
- Identify Risk,
- Qualitative analyses of risks
- Quantitative analysis of risk
- Plan risk response

Project monitoring phase

- Control risks

Risk in project management

Risk is defined as a possible negative event or condition that would disrupt relevant aspects of the project. There are two main categories of project risk: the category that impacts the business benefits (and therefore endangers the reasons for the project's very existence) and the category that impacts the project itself. The project sponsor is responsible for mitigating the first category of risk and the project manager is responsible for mitigating the second category. The risk management process consists of five steps that are repeatedly executed during a project. Phase-end milestones are a good anchor point in time at which to review and update the initial risk assessments and related mitigations.

Risk management process

- **Identify risk:** Perform a brainstorming session with your team and create an inventory of possible risk.

- **Assess and evaluate risk:** Quantify the likelihood (expressed as a percentage) and the impact of the risk (expressed as an amount of money). The "insurance policy" (total impact) that needs to be in the project budget is calculated as the likelihood multiplied by the impact.
- **Manage risk:** Create a risk management plan, describing the strategy adopted and measures to deal with the risk. Generally, the more important the risk, the more budget should be made available for countermeasures. Countermeasures could include prevention, detection and damage control/reconstruction activities. Any risk can be mitigated, avoided, transferred or accepted depending on its severity, likelihood and cost of countermeasures and the organisation's policy.
- **Monitor risk:** Discover risk that materialises, and act accordingly.
- **Evaluate the risk management process:** Review and evaluate the effectiveness and costs of the risk management process

IS auditor has to focus on the risk management process as it provides detailed insight on the effectiveness of project management.

3.8 Project Closing

Projects should be formally closed to provide accurate information on project results, improve future projects and allow an orderly release of project resources. The closure process should determine whether project objectives were met or excused, and should identify lessons learned to avoid mistakes and encourage repetition of good practices. Project closure is to be planned in two situations:

1. Project deliverables are completed and are ready to be implemented

- The project sponsor should be satisfied that the system produced is acceptable and ready for implementation/delivery.
- Custody of contracts may need to be assigned, and documentation archived or passed on to those who will need it.
- Survey the project team, development team, users and other stakeholders to identify any lessons learned that can be applied to future projects.
- Achievement of objectives of project and performance fulfilment, adherence to the schedule, costs, and quality of the project.
- Post project review in which lessons learned and an assessment of project management processes used are documented.
- Release of project teams either to other projects or line functions.

2. Project is suffering from risk materialisation and has to be terminated

These are generally exceptional situations like changes in functional requirements, obsolescence of planned technology, availability of new technology, unforeseen budget constraints, strategy changes etc. In rare cases the project may have to be terminated due to non-performance of project teams. In such situations closure of project may have to be planned depending upon the status of project. For example, based on project planning, organisation may have placed order for required software and hardware or might have made some changes to its existing infrastructure. These need to be undone or planned so as to minimise the impact on organisation.

IS auditor conducting review after project closure has to consider the overall project execution on various parameters such as objectives achieved, time overrun, cost overrun, quality of deliverables? If the review is being done immediately after implementation, IS auditor may also review the challenges faced by the users and the resolution methods.

Achieving business objectives must be the focus of project review. Accordingly, the auditor may review and comment on budget and time overrun situations.

3.9 Roles and responsibilities

The various roles and responsibilities of groups/individuals that are associated with the project and programme management for SDLC project are described below:

3.9.1 Steering committee

Project steering committee provides overall direction and monitors the project execution this is assured by representation of major stakeholders. The project steering committee is ultimately responsible for all deliverables, project costs and schedules.

This committee should be comprised of senior representatives having authority for decision from business areas likely to be impacted by the proposed system or change. Mostly project sponsor will chair the steering committee. The project manager is member of steering committee.

Role of project steering committee

Project steering committee performs the following functions:

- Reviews project progress periodically (fortnightly or monthly or as required)
- Serves as co-ordinator and advisor to project. Members of the committee should be available to make user-related decisions about system and programme design.
- Takes corrective action based on reviews. The committee should evaluate progress and take action or make recommendations to resolve project issues related to budget, schedules, resources, and scope and project objectives.
- Assess risks and decide upon mitigation plan. Also resolve issues that are escalated and cannot be resolved at the project level.
- Take decision on and if required recommend the project be halted or discontinued.
- Works closely with the project manager to define project success factors and metrics in measurable and quantifiable terms.

3.9.2 Project Sponsor

Head of business function or senior management (generally who has highest stake in benefit realisation from project) is designated as project sponsor who provides funding and assumes overall ownership and accountability of the project. Project sponsor is responsible for providing funding and budget for the project execution.

3.9.3 Project Manager

A project manager should be identified and appointed by the IS steering committee. The project manager, who need not be an IS staff member, should be given complete operational control over the project and be allocated the appropriate resources, including IS professionals and other staff from user departments, for the successful completion of the project. A project manager is appointed for execution of project. The project manager can be from the user department, or from IS department or hired separately to handle the project.

Primary functions of Project manager are:

- Provide day-to-day management and leadership
- Ensure that project activities are in line with predetermined objectives
- Involve affected departments
- Follow organisation's project management standards
- Ensure expected quality of deliverables
- Resolve conflicts
- Monitor and controls costs, schedules and associated risks

3.9.4 Senior management

Demonstrates commitment to the project and approves the necessary resources to complete the project. This commitment from senior management helps ensure involvement by those needed to complete the project. Generally senior management representative is appointed on steering committee.

3.9.5 Business management

Business management, most of the times, assumes ownership of the project and resulting system, allocates qualified representatives to the team, and actively participates in business process redesign, system requirements definition, test case development, acceptance testing and user training. Business management should review and approve system deliverables as they are defined and implemented.

Business management is concerned particularly with the following questions:

- Are the required functions available in the software?
- How reliable is the software?
- How efficient is the software?
- Is the software easy to use?
- How easy is it to transfer or adapt old data from pre-existing software to this environment?
- How easy is it to transfer the software to another environment?
- Is it possible to add new functions?
- Does it meet regulatory requirements?

3.9.6 Systems development project team

System development team consist of System analyst, Developers, testing professionals, control consultants (IS Auditor), hardware and network consultants, The team members completes assigned tasks, communicates effectively with users by actively involving them in the development process, works according to local standards and advises the project manager of necessary project plan deviations.

3.9.7 Business function representatives/domain specialists

Consists of subject matter experts (SME) that provides inputs to developers and system analysts on requirements, business related controls, and sometime approves the low level design specifications.

3.9.8 Security Officer

Ensures that system controls and supporting processes provide an effective level of protection, based on the data classification set in accordance with corporate security policies and procedures; consults throughout the life cycle on appropriate security measures that should be incorporated into the system; reviews security test plans and reports prior to implementation; evaluates security-related documents developed in reporting the system's security effectiveness for accreditation; and periodically monitors the security system's effectiveness during its operational life.

3.9.9 IS Auditor

The IS auditor can be part of SDLC project team as consultant for internal controls or for the review of the project activities. They may also provide an independent, objective review to ensure appropriate level of commitment of the responsible parties. Is auditor has to understand the systems development; acquisition and maintenance methodologies used by the organisation and identify potential vulnerabilities. If auditor observes control weakness either as a result of review due to organisational structure or the software methods used, or weakness in process execution, it is the IS auditor's role to advise the project team and senior management of the deficiencies in project management and provide recommendations for improvement.

Role of IS Auditor in SDLC

Throughout the project management process, IS Auditor should analyse the associated risks and exposures inherent in each phase of SDLC. He should assure that appropriate control mechanisms are in place to minimise the risks in a cost effective manner. While reviewing SDLC various phases as well as attend the project team meetings offering advice through Software Development process. He will also assess the project development team's ability to produce key deliverables by the promised dates. Adequate and complete documentation of all phases should be collected and reviewed by processes, IS Auditor is expected to obtain necessary and available documentation from the IS Auditor. The specific areas of review are:

- Understand standards adopted and followed by the organisation through the process of inquiry, observation and documentation review
- To determine significant phases for the various size and type

- To assess efficiency and effectiveness of each function to satisfy the users goals and organisation objectives
- To test methodology adopted and determine compliance with the organisation standards by reviewing the documentation produced
- To evaluate controls designed for compliance with internal control principles and standards
- To determine compliance with common security, auditability and change control standards

If IS auditor is part of project team not for performing an audit, but is participating on the project in an advisory role then depending on the level of involvement, IS auditor may become ineligible to perform audits of the application when it becomes operational.

3.9.10 Quality Assurance (QA)

Quality assurance function consists of following key activities:

1. Develop test plan and test the code
2. Review and ensure that project documentation is complete
3. Review deliverables of the project

The objective is to ensure that the quality of the project by measuring the adherence by the project staff to the organisation's standard methodology of System Development life cycle (SDLC), advise on deviations, and propose recommendations for process improvements or greater control points when deviations occur.

Specific objectives of the QA function include:

- Ensuring the active and co-ordinated participation by all relevant parties in the revision, evaluation and dissemination, and application of standards, management guidelines and procedures
- Ensuring compliance with the agreed on systems development methodology
- Reviewing and evaluating large system projects at development milestones, and making appropriate recommendations for improvement
- Establishing, enhancing and maintaining a stable, controlled environment for the implementation of changes within the production software environment
- Defining, establishing and maintaining a standard, consistent and well-defined testing methodology for applications
- Reporting to management on systems that are not performing as defined or designed

3.9.11 Technology Specialist

IT is developing so rapidly that even IT professionals find it difficult to keep track of all developments, let alone develop expertise. This has resulted in experts in specific technology areas, such as Microsoft technology, Web-enablement and the like.

3.9.12 Systems Analyst

The system analyst also has a responsibility to understand existing problem/system/data flow and new requirements. System analysts convert the user's requirements in the system requirements to design new system.

3.9.13 Programmers/Developers

Programmers convert design into programmes by coding using programming language. They are also referred to as coders or developers.

3.9.14 Testers

Testers are junior level quality assurance personnel attached to a project. They test programmes and sub programmes as per the plan given by the module / project leaders and prepare test reports.

3.9.15 Documentation Specialist

These professionals are responsible for the creation of user manuals and other documentation.

3.9.16 Database Administrator (DBA)

The data in a database environment has to be maintained by a specialist in database administration so as to support the application programme. The database administrator handles multiple projects; and ensures the integrity and security of information stored in the database.

3.10 SDLC Project management techniques and tools

System development process may be associated with various automated tools that help in improving productivity and maintaining record and documentation of application being developed. Tools that help in improving productivity include code generators, development environments (also referred to as developer's workbench) like visual studio and Computer-Aided Software Engineering (CASE) applications that help in documenting the SDLC process. In addition project managers may use project management tools like MS Project. This section provides information about these tools. This section covers following three areas:

1. CASE tools
2. Software size estimation covering various techniques used like LOC, FPA analysis etc.
3. Project controlling tools like PERT, CPM and Gantt Charts.

3.10.1 Computer Aided Software Engineering (CASE) tools

SDLC requires collecting, organising and presenting information required at application systems and programme level. This involves building data flows, documenting design of application system, identifying modules/functions/programme required to be developed and sometimes developing prototypes to capture requirements. These are essential but time consuming processes that are required for developing, using and maintaining computer applications.

Although IS auditor is not expected to have detailed knowledge of how to use CASE tools, they may have to learn how to use CASE tools for effective audit of SDLC project, as required.

Computer-aided software engineering (CASE) is automated tools that aid in the software development process. Their use may include tools for capturing and analysing requirements, software design, code generation, testing, document building and other software development activities.

Code Generators

Code generators are tools that are part of CASE tools or development environment like visual studio. These tools generate programme source code based on parameters provided. These products significantly reduce the development (particularly coding) time; however maintaining or changing these programmes might be painful and time consuming.

Development environments and non-procedural languages

Developer's workbench: Provides environment to developer for editing, simulating code, temporary storage, file management and sometime code generation. It may also provide Software facilities that include the ability to design or paint retrieval screen formats, develop computer-aided training routines or help screens, and produce graphical outputs. It is often referred to as an integrated development environment (IDE).

Non-procedural languages: These are event driven and make extensive use of object-oriented programming concepts such as objects, properties and methods. These languages cannot perform data intensive or online operations however are best suited to provide environment to end user for generating their own views and report required for data analysis and decision making. These languages provide environmental independence (portability) across computer architectures, operating systems and telecommunications monitors. These languages generally have simple language subsets that can be used by less-skilled users.

These languages are classified in the following ways:

- **Query and report generators :** These languages can extract and produce reports and sometimes can access database records, produce complex online outputs.
- **Embedded database languages** are more user-friendly but also may lead to applications that are not integrated well with other production applications.
- **Relational database languages** are usually an optional feature on a vendor's DBMS. These allow the applications developer to make better use of the DBMS product, but they often are not end-user-oriented.

3.10.2 Software Size Estimation

Once the work breakdown structure is completed and SDLC methodology (discussed in Chapter 4) is finalised project manager must perform Software size estimation, i.e. determining the physical size of application (number of programmes, modules, reusable function/modules etc.). This helps the project manager in deciding resource and skills requirements, to judge the time and cost required for development, and to compare the total effort required by the resources.

Source lines of code (SLOC)

Traditionally, particularly when COBOL like languages was used, software sizing used to be performed using number of source lines of code (SLOC). However it does not work well for complex systems using different types of programmes and automated tools like source code generators. This puts limitation on planning for cost, schedule and quality metrics.

With new technologies multi-point estimations techniques were developed that now uses diagrams, objects, spreadsheet cells, database queries and graphical user interface (GUI) widgets. These technologies are more closely related to functionality that needs to be created rather than lines of code.

Function Point Analysis (FPA)

The function point analysis (FPA) technique has evolved over the years and is widely used for estimating complexity in developing large business applications. The results of FPA are a measure of the size of an information system based on the number and complexity of the inputs, outputs, files, interfaces and queries with which a user views and interacts with the data. This is an indirect measure of software size and the process of development. It is based on the number and complexity of inputs, outputs, files, interfaces and queries.

Function points (FPs) are computed by considering various parameters like number of users, number of inputs, number of outputs, expected user actions, data elements to be processed and external interfaces to determine whether a particular module/programme is simple, average or complex. This information is used to compute function point using an algorithm that takes into account complexity adjustment values (i.e., rating factors) based on responses to questions related to reliability, criticality, complexity, reusability, changeability and portability.

Function Points (FP) derived from this equation are then used as a measure for cost, schedule, productivity and quality metrics (e.g., productivity = FP/person-month, quality = defects/FP, and cost = monetary value/FP).

IS auditor should be familiar with the use of Function Point Analysis. However, IS Auditors are not expected to be experts in this technique.

FPA Feature Points

A slightly different approach for function point analysis for system software such as operating systems, telephone switching systems, etc. was developed. To differentiate from FPA it is called “Feature Points." It is used for software that has well-defined algorithms like systems software, embedded software, real time software, CAD, Artificial intelligence and some traditional MIS software.

In web-enabled applications, the development effort depends on the number of forms, number of images; type of images (static or animated), features to be enabled, interfaces and cross-referencing that is required. Thus, from the point of view of web applications, the effort would include all that is mentioned under function point estimation, plus the features that need to be enabled for different types of user groups. The measurement would involve identification or listing of features, access rules, links, storage, etc.

Cost Budgets

Cost estimates of a SDLC project are based on the amount of effort likely to be required to carry out each task. The estimates for each task contain one or more of the following elements:

- Person-hours for all types of resources e.g. system analyst, programmers, support staff, Testing teams etc. (Pl. refer Section 3.9 Roles and Responsibilities)
- Infrastructure (Hardware, software, networks etc.), other specialized software, if any and communication equipment)
- Other costs such as third-party services, automation tools required for the project, consultant or contractor fees, training costs, etc.

Based on estimates following steps are used in arriving at cost budget:

- Prepare estimate of human and machine effort by for all tasks.
- Determine hourly rate for each type of person-hours and arrive total person cost.

3. Project Controlling tools and techniques

Project manager uses various tools and techniques to control the project. The graphical techniques used to represent schedule are:

A. Project evaluation review technique (PERT)
B. Critical path method (CPM)
C. Gantt Chart

A. Programme Evaluation Review Technique (PERT)

PERT is a technique to estimate the efforts and time required to complete the work/task described by work breakdown structure (WBS). Project manager lists out the major tasks and arrives at three different duration estimates of each activity. The three estimates are then used to derive single estimate applying a mathematical formula. PERT is often used in projects with uncertainty about the duration.

Table 3.1 illustrates one such formula for a hypothetical project where activities are named from A to L. The first is the most optimistic time (if everything went well) and the third is the pessimistic or worst-case scenario. The second is the most likely scenario.

This estimate is based on experience attained from projects that are similar in size and scope. To calculate the PERT time estimate for each given activity, the following calculation is applied:

[Optimistic + Pessimistic + 4(most likely)]/6

Table 3.1: PERT table

Activity	Optimistic Estimate	Pessimistic Estimate	Most Likely Estimate	Final Estimate
A	2	6	4	4
B	4	10	7	7
C	4	14	9	9
D	7	19	10	11
E	2	6	4	4
F	1	5	3	3
G	2	8	5	5
H	3	9	6	6
I	2	6	4	4
J	1	5	3	3
K	2	6	4	4
L	1	3	2	2

Figure 3.4 illustrates use of the PERT network management technique. (Each circle represents milestones and the arrow represents activities. Number after activity shows the number of days required to complete the activity.)

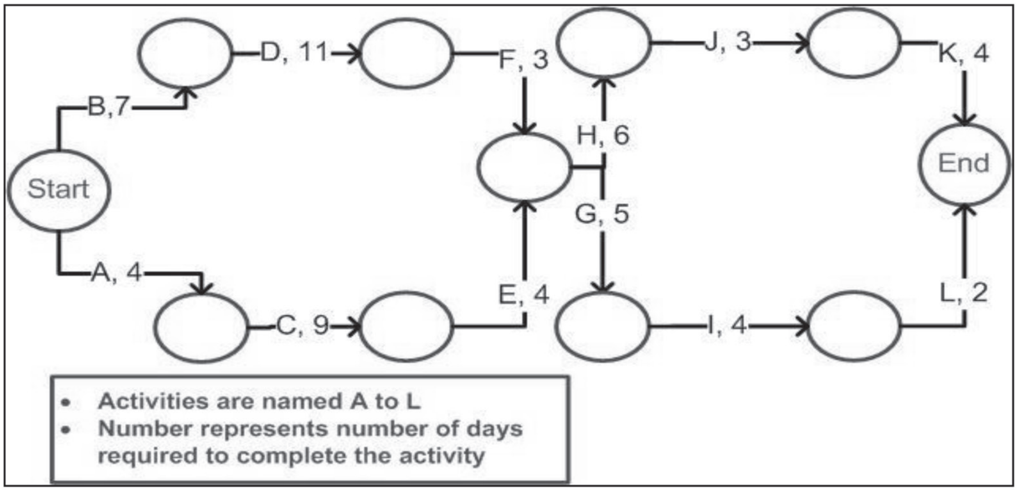

Start
B,7
D, 11
F, 3
J, 3
K, 4
H, 6
G, 5
End
A, 4
C, 9
E, 4
I, 4
L, 2
Activities are named A to L
Number represents number of days required to complete the activity

Figure 3.4: PERT Diagram

Some project managers show all estimates in the PERT diagram.

B. Critical Path Methodology

All project schedules have a critical path. Activities of a project are in sequence or independent or parallel, a project can be represented as a network of activities is shown in PERT diagram. (Some project managers refer to PERT diagram as PERT network).

A path through the network is any set of successive activities which go from the beginning to the end of the project. Associated with each activity in the network is a single number that represents estimates the amount of time that the activity will require to complete.

The critical path is the sequence of activities whose sum of activity time is highest than that for any other path through the network. Critical paths represents the shortest possible project completion time, if everything goes according to schedule. In other words, delay in completing any activity on critical path delays the overall project.

Activities which are not in the critical path have slack time, i.e. delay in performing these activities may not affect the overall project schedule. Activities on critical path have zero slack time.

The PERT diagram shown in figure 3.4 has following 4 paths:

1. A – C – E – G - I – L
2. A – C – E –H – J – K
3. B – D – F – H – J – K
4. B – D – F – G – I – L

Using the time estimates the total time require for each path is 28, 30, 34 and 32 days respectively. Third path hence is Critical Path. (Shown by thick arrows in Figure 3.5

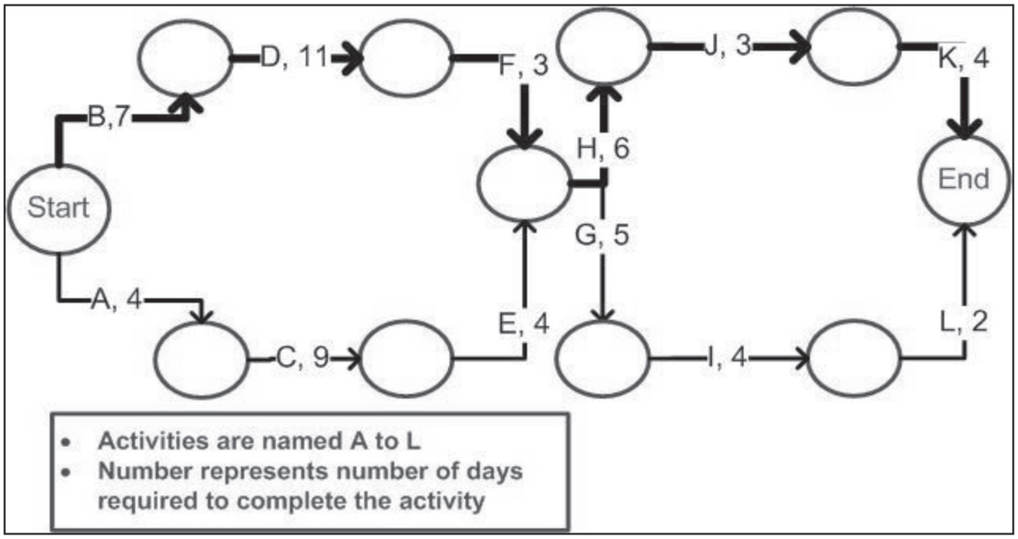

Start
B,7
D, 11
F, 3
J, 3
K, 4
H, 6
End
G, 5
A, 4
C, 9
E, 4
I, 4
L, 2
• Activities are named A to L
• Number represents number of days required to complete the activity

Figure 3.5: Critical Path Method (CPM)

Project manager can use the slack time on non-critical path for scheduling resources optimally, since slack time provides flexibility to start activity late than scheduled start date. The slack times for a project are computed by working forward through path, computing the earliest possible completion time for each activity, until the earliest possible completion time for the total project is found. Then by working backward through the network, the latest completion time for each activity is found, the slack time computed and the critical path identified.

Most CPM packages facilitate the analysis of resource utilisation per time unit (e.g., day, week, etc.) and resource levelling, which is a way to level off resource peaks and valleys.

C. Gantt Charts

Gantt charts are aid for scheduling activities/tasks needed to complete a project. These charts show details related to activities calculated during PERT and CPM. The charts also show which activities are in progress concurrently and which activities must be completed sequentially. Gantt charts may reflect the resources assigned to each task and by what per cent allocation. The charts aid in identifying activities that have been completed early or late. Progress of the entire project can be tracked from the Gantt chart. Gantt charts can also be used to track the milestones for the project.

Figure 3.6 Provides example of Gantt chart.

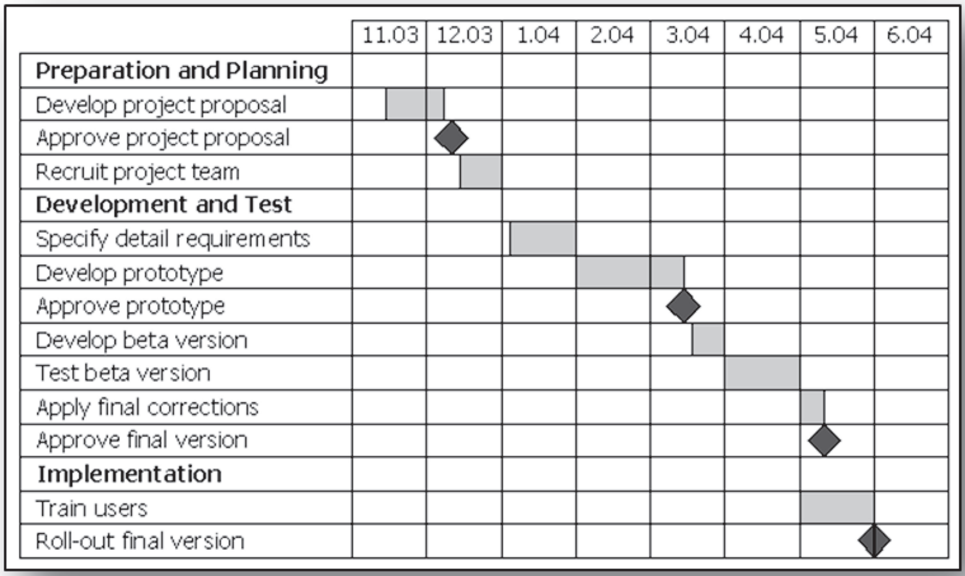


Figure 3.6: Gantt chart

3.11 Summary

Every project has unique success criteria based on the expectations of stakeholders. Generally success criteria are measurable and manageable such as cost, time and scope. However some criteria, such as meeting business needs, are subjective but essential. The project sponsor is a key stakeholder who defines such success criteria. The project team should capture project requirements and document them at the initial stage to complete the project successfully. Activity of capturing requirements is usually difficult because it involves subjective decisions and extensive interaction between users and developers. Requirements should be formally approved and then frozen (baselined) to prevent scope creep. Success criteria allow the project manager to focus on managing risks that can affect desirable outcome and successful completion of the project.

IS auditor should review adequacy of the following project management activities:

- **Levels of oversight by project committee/board**
- **Risk management methods within the project**
- **Issue management**
- **Cost management**
- **Processes for planning and dependency management**
- **Reporting processes to senior management**
- **Change control processes**
- **Stakeholder management involvement**
- **Sign-off process**
- **Adequate documentation of all phases of the SDLC process such as:**
  - o **Availability of clearly defined objectives on what is to be accomplished during each phase.**
  - o **Key deliverables of each phase with project personnel assigned direct responsibilities for these deliverables.**
  - o **A project schedule with highlighted dates for the completion of key deliverables.**
  - o **An economic forecast for each phase, defining resources and the cost of the resources required to complete the phase.**

3.12 Questions

1. Who among the following is responsible for ongoing facilitation of a SDLC project?
   A. Project sponsor
   B. Project manager
   C. Steering committee
   D. Board of directors
2. A multi-national organisation has decided to implement an ERP solution across all geographical locations. The organisation shall initiate a:
   A. Project
   B. Program
   C. Portfolio
   D. Feasibility study
3. Which of the following primarily helps project manager in mitigating the risk associated with change in scope of software development project?
   A. Change management process
   B. Use of prototyping
   C. Revising effort estimates
   D. Baselining requirements

4. Monitoring which of the following aspect of SDLC project shall help organisation in benefit realisation over sustained period of time? The project adhering to:
   A. Quality
   B. Budget
   C. Schedule
   D. Methodology
5. Which of the following tools and techniques primarily help in improving productivity of SDLC project team members?
   A. Use of standard methodology
   B. Software sizing using FPA
   C. Developers' workbench
   D. Appropriate HR policies
6. While performing mid-term review of SDLC project, the IS auditor primarily focuses on:
   A. Project risk management process
   B. Adherence to the schedule
   C. Reviewing minutes of steering committee meeting
   D. Cost management is as per budget

3.13 Answers and Explanations

1. A. Project sponsor is a stakeholder having maximum interest/stake in the success of project and is primary responsibility is to coordinate with various stakeholders for project success. Project manager is responsible for executing the project activities. Steering committee monitors project progress but is not ongoing activity. Board of director provides direction.
2. B. Considering the spread of organisation the organisation shall initiate a programme for implementing ERP, consisting of different project for each location. The programme shall be part of IT programme portfolio of organisation. Since the decision has been made, feasibility study either has been completed or shall be initiated as part of programme.
3. D. Scope creep of continued changes in requirements during SDLC project is most common risk. If not properly handled the project may be delayed and benefit realisation from the project shall be affected. The project manager therefore, must freeze the scope by base-lining requirements. Any change after base-lining shall follow change management process. Change management process without base-lining may not help. Project manager may or may not use prototyping for freezing the requirements. Revised effort estimates are applicable after change is approved.
4. A. Quality is most important aspect for SDLC project, since it minimises errors that can impact operations.
5. C. Automated tools help team in improving productivity as these tools help in managing mundane and structure activities and developers can focus on core activities. Developers'

workbench provides various functions that help in improving productivity. Use of standards help in following uniform methods and reducing rework. Software sizing is useful in monitoring productivity. HR policies may help in motivating team but it is secondary.

6. A. Auditor should primarily focus on risk management that will provide inputs on events that has impact on all aspects of project. Options B, C and D help in confirming the findings from review of risk management process.

CHAPTER 4: DIFFERENT MODELS AND METHODS FOR SDLC

Learning objectives

This chapter provides information on various software development models and methods. A project manager may adopt these models/methods depending upon functional and non-functional requirements to ensure the timely deliverables. The method selected is dependent on cost, time, quality and business requirements of the project.

4.1 Introduction

This chapter mainly covers basic understanding of system design and development methodologies being used while executing SDLC project. A Project Manager and system analysis depending on requirements selects one or more software development methods and/or models for developing software. Execution of these models primarily focuses on converting requirements into a system design and to develop the solution. This chapter provides information of various models and methods that might be required for executing phases 4 and 5 of SDLC. IS auditor primarily should be looking for control aspects associated with development methodology.

Software development undergoes different phases as discussed in Chapter 1. However the execution of these phases vary based on the requirements, particularly phases 3 to 6, i.e. analysis, design, development and testing. This chapter discusses the various models and methods used for software development. These models have been evolving due to changes in:

1. **Technology:** Few examples of impact of Technological advancement on SDLC process are:
    a. Availability of faster processor and network has created a requirement of client server and web based application where application and data are segregated and hosted at one central location.
    b. Separation of application from platform (OS, Hardware etc.). Organisations needed software that can run on any platform/OS, which has enabled development of platform independent systems using JAVA (or similar) development environments.
2. **Business requirements:** Spread and growth of business expects faster delivery of application/solutions from SDLC project. For example:
    a. Dynamic changes in functional requirements due to customer focus and competition has reduced turnaround time in change management. This has been enabled by object oriented methodology which facilitates development of reusable and independent programme modules.
    b. Reduced time between conceptualisation and go-to-market due to customer focused service approach. This has created the need for Rapid Application Development (RAD) and Agile Development Methodology.
3. **Availability of outsourcing for software development:** This has prompted changes in the SDLC methodology, for example:

a. **Requirement finalisation:** Due to commercial aspects associated with outsourcing, requirement base lining using the prototype model has become the most popular model.

b. **Faster delivery of product:** This has enabled development teams to modify SDLC models e.g. waterfall model was changed to verification and validation model, spiral model and incremental model.

In this chapter, we will discuss salient features of various traditional SDLC models and methodologies such as:

1. Waterfall model
2. Spiral Model
3. Incremental model
4. Prototyping model

Further, we will also discuss some important methodologies that have evolved and are currently being used. The following six methodologies are explained here with strengths and weaknesses of each of them:

1. Rapid Application development (RAD)
2. Agile development
3. Re-engineering and reverse engineering
4. Object oriented development
5. Component based development
6. Web application development

4.2 SDLC Models

4.2.1 Waterfall model

The waterfall approach is a traditional development approach in which each phase is executed in sequence or in linear fashion. These phases include requirements analysis, specifications and design requirements, coding, final testing, and release. Fig. 4.1 shows representative model of this method. When the traditional approach is applied, an activity is undertaken only when the prior step is completed.

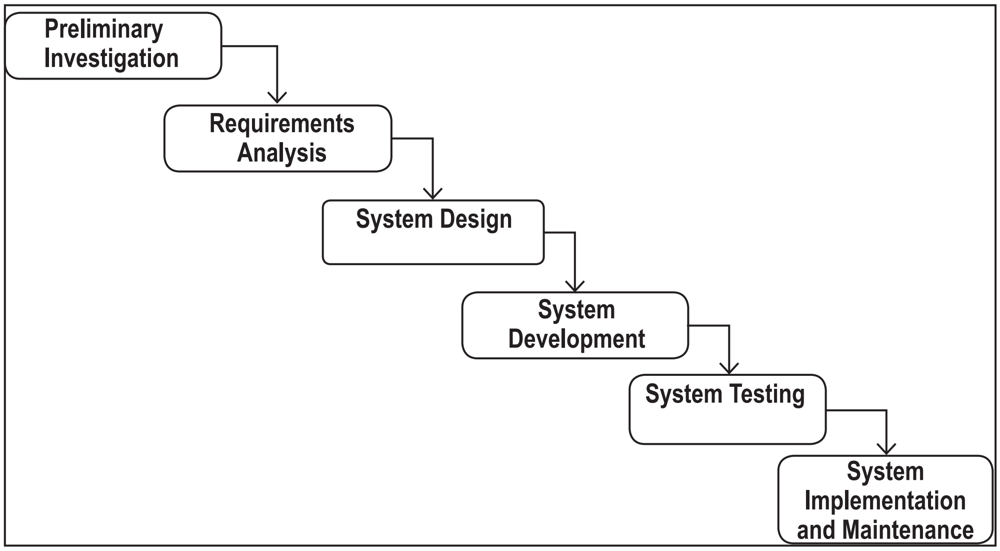


Fig. 4.1: Waterfall Approach

The characterizing features of this model have influenced the development community in big way. Some of the key characteristics are:

- 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 the 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.

Strengths

- It is ideal for supporting less experienced project teams and project managers or project teams, whose composition fluctuates.
- The orderly sequence of development steps and design reviews help to ensure the quality, reliability, adequacy and maintainability of the developed software.
- Progress of system development can be tracked and monitored easily.
- It enables to conserve resources.

Weaknesses

- It is criticised 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 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 discovered during design and coding are most difficult to handle.
- 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.
- Written specifications are often difficult for users to read and thoroughly appreciate.
- It promotes the gap between users and developers with clear vision of responsibility.

Verification and validation – a variant of waterfall model

In order to get benefits and reduce the inflexibility verification and validation variant was introduced. Figure 4.2 shows the changes made in the waterfall model. The model has introduced validation and verification at each stage (shown by dotted line). This model minimises some weaknesses in waterfall model by retaining strengths.

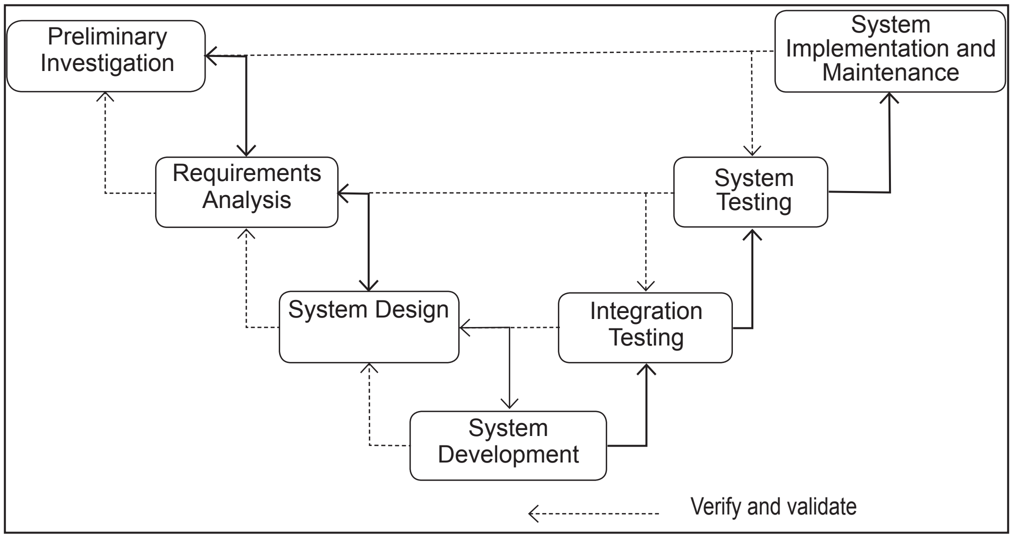


Figure 4.2: Variant of waterfall model

4.2.2 Prototyping methodology

The traditional approach sometimes may take long time to analyse, design and implement a system. More so, many a times we know a little about the system until and unless we go through its working phases, which are not available. In order to avoid such bottlenecks and overcome the issues, organisations are increasingly using prototyping techniques to develop smaller systems such as DSS, MIS 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 may be 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 the 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 it is scrapped. If it is scrapped, the knowledge gained from building the prototype is used to develop the real system.

Prototyping can be viewed as a series of four steps, symbolically depicted in Fig. 4.5 wherein implementation and maintenance phases followed by full-blown developments take place once the prototype model is tested and found to be meeting user's requirements.

Generic phases of model

- **Identify Information System Requirements:** In traditional approach, the system requirements are to be identified before the development process starts. However, under prototype approach, the design team needs only fundamental system requirements to build the initial prototype, the process of determining them can be less formal and time-consuming than when performing traditional systems analysis.
- **Develop the Initial Prototype:** The designers create an initial base model and give little or no consideration to internal controls, but instead emphasise system characteristics such as simplicity, flexibility, and ease of use. These characteristics enable users to interact with tentative versions of data entry display screens, menus, input prompts, and source documents. The users also need to be able to respond to system prompts, make inquiries of the information system, judge response times of the system, and issue commands.
- **Test and Revise:** After finishing the initial prototype, the designers first demonstrate the model to users and then give it to them to experiment and ask users to record their likes and dislikes about the system and recommend changes. Using this feedback, the design team modifies the prototype as necessary and then resubmits the revised model to system users for re-evaluation. Thus iterative process of modification and re-evaluation continues until the users are satisfied.
- **Obtain User Signoff of the Approved Prototype:** Users formally approve the final version of the prototype, which commits them to the current design and establishes a contractual obligation about what the system will, and will not, do or provide. Prototyping is not commonly used for developing traditional MIS and batch processing type of applications such as accounts receivable, accounts payable, payroll, or inventory management, where the inputs, processing, and outputs are well known and clearly defined.

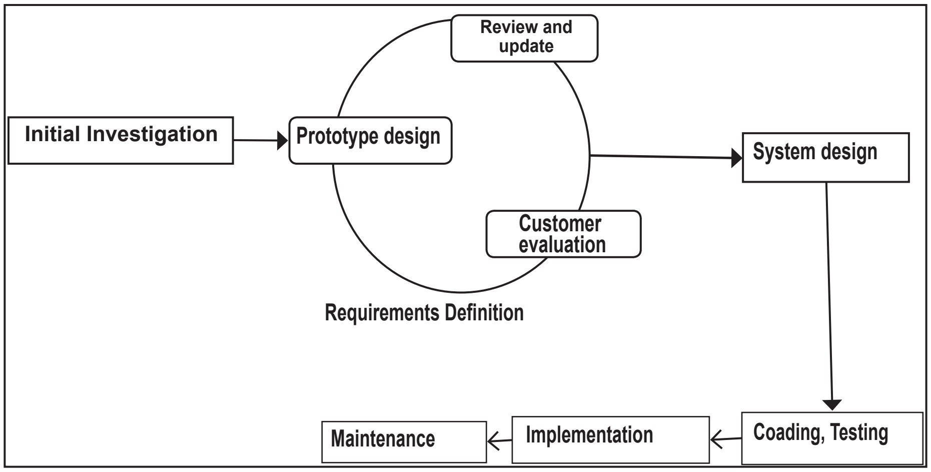


Fig. 4.3: Prototyping Model

Strengths

- It improves both user participation in system development and communication among project stakeholders.
- It is especially useful for resolving unclear objectives and requirements; 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 these users' needs and requirements than does the 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 the traditional systems development approach is employed.

Weaknesses

- Approval process and control are not formal.

- Incomplete or inadequate problem analysis may occur whereby only the most obvious and superficial needs will be addressed, resulting in current inefficient practices being easily built into the new system.
- Requirements may frequently change significantly.
- Identification of non-functional elements is difficult to document.
- Designers may prototype too quickly, without sufficient upfront user needs analysis, resulting in an inflexible design with narrow focus that limits future system potential.
- Prototype may not have sufficient checks and balances incorporated.
- Prototyping can only be successful if the system users are willing to devote significant time in experimenting with the prototype and provide the system developers with change suggestions. The users may not be able or willing to spend the amount of time required under the prototyping approach.
- The interactive process of prototyping causes the prototype to be experimented with quite extensively. Because of this, the system developers are frequently tempted to minimise the testing and documentation process of the ultimately approved information system. Inadequate testing can make the approved system error-prone, and inadequate documentation makes this system difficult to maintain.
- Prototyping may cause behavioural problems with system users. These problems include dissatisfaction by users if system developers are unable to meet all user demands for improvements as well as dissatisfaction and impatience by users when they have to go through too many interactions of the prototype.

In spite of above listed weaknesses, to some extent, systems analysis and development has been greatly improved by the introduction of prototyping. Prototyping enables the user to take an active part in the systems design, with the analyst acting in an advisory role. Prototyping makes use of the expertise of both the user and the analyst, thus ensuring better analysis and design, and prototyping is a crucial tool in that process.

Prototype has one major drawback that many a times users do not realise that prototype is not actual system or code but is just a model whereas users may think that the system is ready. Actual development starts only after the prototype is approved. Hence, the actual system may require time before it is ready for implementation and use. In the meantime, users may get restless and wonder why there is so much delay.

4.2.3 Spiral Model

The Spiral model is a repetitive software development process combining elements of both design and prototyping within each of the iterations. It combines the features of the prototyping model and the waterfall model (given in Fig. 4.3). Initially spiral model was intended for large, expensive and complicated projects like Game development because of size and constantly shifting goals of large projects. Spiral model when defined was considered and best model and further models were developed using spiral models.

Key characteristics

- Spiral model is an iterative model where each iteration helps in optimising the intended solution.
- The new system requirements are defined in as much detail as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
- A preliminary design is created for the new system during initial iterations. This phase is the most important part of "Spiral Model" in which all possible alternatives that can help in developing a cost effective project are analysed and strategies are decided to use them. This phase has been added specially in order to identify and resolve all the possible risks in the project development. If risks indicate any kind of uncertainty in requirements, prototyping may be used to proceed with the available data and find out possible solution in order to deal with the potential changes in the requirements.
- A first prototype of the new system in constructed from the preliminary design during first iteration. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
- A second prototype is evolved during next iteration by a fourfold procedure by evaluating the first prototype in terms of its strengths, weaknesses, and risks; defining the requirements of the second prototype; planning and designing the second prototype; and constructing and testing the second prototype.

Strengths

- Enhances the risk avoidance.
- Useful in helping for optimal development of a given software iteration based on project risk.
- Incorporates 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.

Weaknesses

- It is challenging to determine the exact composition of development methodologies to use for each of the iterations around the Spiral.
- A skilled and experienced project manager is required to determine how to apply it to any given project.
- Sometimes there are no firm deadlines, cycles continue till requirements are clearly identified. Hence has an inherent risk of not meeting budget or schedule.

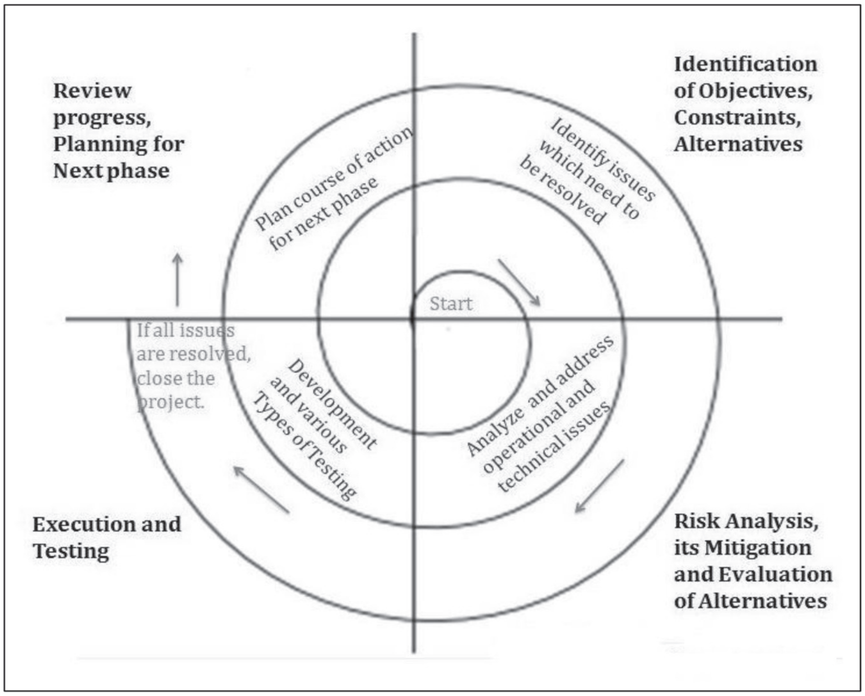

Review progress, Planning for Next phase
Identification of Objectives, Constraints, Alternatives
Plan course of action for next phase
Identify issues which need to be resolved
Start
If all issues are resolved, close the project.
Development and various Types of Testing
Analyze and address operational and technical issues
Execution and Testing
Risk Analysis, its Mitigation and Evaluation of Alternatives

Fig. 4.4: Spiral Model

4.2.4 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. It is pictorially depicted in Fig. 4.4.

The product is decomposed into a number of components, each of which are designed and built separately (termed as builds). Each component is delivered to the client when it is complete. This allows partial utilisation of product and avoids a long development time. It also creates a large initial capital outlay with the subsequent long wait avoided. This model of development also helps to ease the traumatic effect of introducing completely new system all at once. A few pertinent features are listed as follows:

- A series of mini-waterfalls are performed, where all phases of the waterfall development model are completed for a small part of the system, before proceeding to the next increment.
- Overall requirements are defined before proceeding to evolutionary, mini–Waterfall development of individual increments of the system.
- The initial software concept, requirement analysis, and design of architecture and system core are defined using the Waterfall approach, followed by iterative Prototyping, which culminates in installation of the final prototype (i.e. working system).

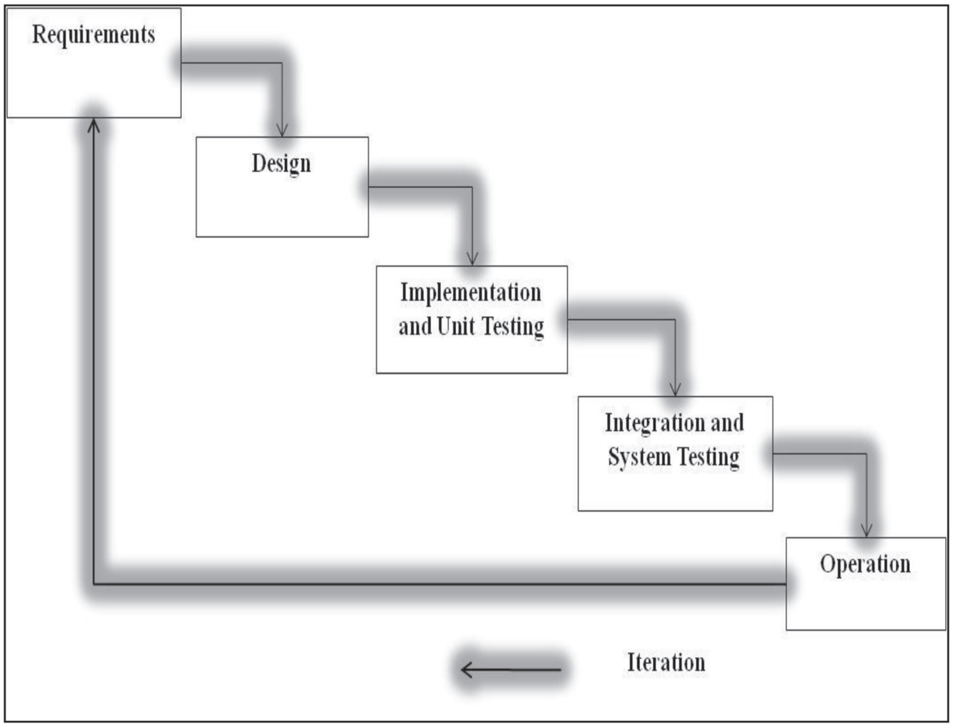

Requirements
Design
Implementation and Unit Testing
Integration and System Testing
Operation
Iteration

Fig. 4.5: Incremental Model

Strengths:

- Potential exists for exploiting knowledge gained in an early increment as later increments are developed.
- Moderate control is maintained over the life of the project through the use of written documentation and the formal review and approval/signoff by the user and information technology management at designated major milestones.
- Stakeholders can be given concrete evidence of project status throughout the life cycle.
- It is more flexible and less costly to change scope and requirements.
- It helps to mitigate integration and architectural risks earlier in the project.
- It allows the delivery of a series of implementations that are gradually more complete and can go into production more quickly as incremental releases.
- Gradual implementation provides the ability to monitor the effect of incremental changes, isolated issues and make adjustments before the organisation is negatively impacted.

Weaknesses:

- When utilising a series of mini-waterfalls for a small part of the system before moving onto the next increment, there is usually a lack of overall consideration of the business problem and technical requirements for the overall system.
- Each phase of an iteration is rigid and do not overlap each other.

- Problems may arise pertaining to system architecture because not all requirements are gathered up front for the entire software life cycle.
- Since some modules will be completed much earlier than others, well-defined interfaces are required.
- It is difficult to demonstrate early success to management.

4.3 SDLC Methodologies

4.3.1 Rapid Application Development (RAD)

RAD refers to a type of software development methodology, which uses minimal planning in favour of rapid prototyping. (Figure 4.6) 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. The key features are:

- Key objective is fast development and delivery of a high quality system at a relatively low investment cost,
- Attempts to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process.
- Aims to produce high quality systems quickly, primarily through the use of iterative Prototyping (at any stage of development), active user involvement, and computerized development tools like Graphical User Interface (GUI) builders, Computer Aided Software Engineering (CASE) tools, Database Management Systems (DBMS), Fourth generation programming languages, Code generators and object-oriented techniques.
- Key emphasis is on fulfilling the business need while technological or engineering excellence is of lesser importance.
- Project control involves prioritizing development and defining delivery deadlines or “time boxes.” If the project starts to slip, emphasis is on reducing requirements to fit the time box, not in increasing the deadline.
- Generally includes Joint Application Development (JAD), where users are intensely involved in system design, either through consensus building in structured workshops, or through electronically facilitated interaction.
- Active user involvement is imperative.
- Iteratively produces production software, as opposed to a throwaway prototype.
- Produces documentation necessary to facilitate future development and maintenance.
- Standard systems analysis and design techniques can be fitted into this framework.

Strengths

- The operational version of an application is available much earlier than with Waterfall, Incremental, or Spiral frameworks.
- Because RAD produces systems more quickly and to a business focus, this approach tends to produce systems at lower cost.

- Quick initial reviews are possible.
- Constant integration isolates problems and encourages customer feedback.
- It holds a great level of commitment from stakeholders, both business and technical, than Waterfall, Incremental, or spiral frameworks. Users are seen as gaining more of a sense of ownership of a system, while developer are seen as gaining more satisfaction from producing successful systems quickly.
- It concentrates on essential system elements from user viewpoint.
- It provides for the ability to rapidly change system design as demanded by users.
- It leads to a tighter fit between user requirements and system specifications.

Weaknesses

- Fast speed and lower cost may affect adversely the system quality.
- The project may end up with more requirements than needed (gold-plating).
- Potential for feature creep where more and more features are added to the system during development.
- It may lead to inconsistent designs within and across systems.
- It may call for violation of programming standards related to inconsistent naming conventions and inconsistent documentation,
- It may call for lack of attention to later system administration needs built into system.
- Formal reviews and audits are more difficult to implement than for a complete system.
- Tendency for difficult problems to be pushed to the future to demonstrate early success to management.
- As some modules are completed much earlier than others, well–defined interfaces are required.

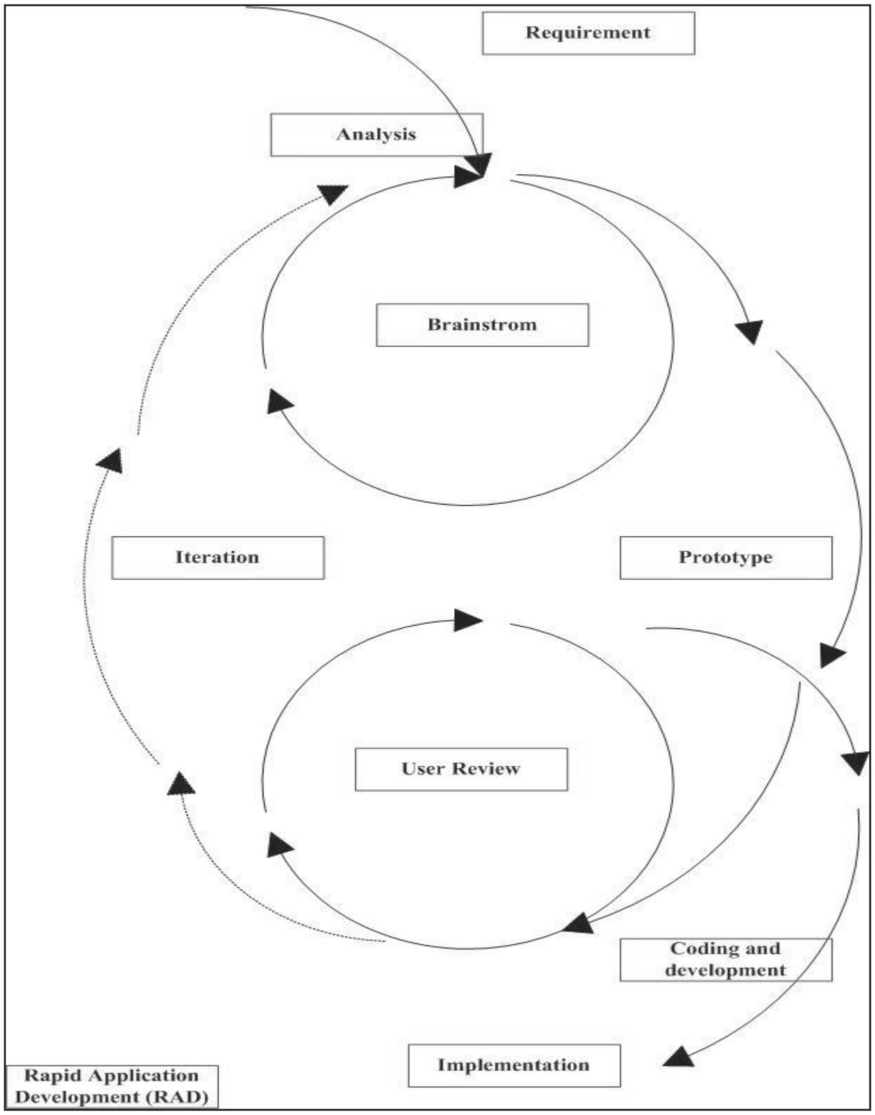


Figure 4.6: RAD

4.3.2 Agile software development methodology

The term "agile development" refers to a family of similar development processes that adopt a non-traditional way of developing complex systems. The term "agile" refers to characteristic of processes that are designed to flexibly handle changes to the system being developed. Scrum is the first project management approach that fits well with other agile techniques. Other agile processes that have since emerged such as Extreme Programming (XP), Crystal, Adaptive Software Development, Feature Driven Development, and Dynamic Systems Development Method.

Key characteristics of agile processes

- Use of small, time-boxed subprojects or iterations where each iteration forms basis for planning next iteration.
- Re-planning the project at the end of each iteration (referred to as a "sprint" in Scrum), including reprioritizing requirements, identifying any new requirements and determining within which release delivered functionality should be implemented
- Relatively greater reliance, compared to traditional methods, on the knowledge in people's heads (tacit knowledge), as opposed to external knowledge that is captured in project documentation
- A heavy influence on mechanisms to effectively disseminate tacit knowledge and promote teamwork. Therefore, teams are kept small in size, comprise both business and technical representatives, and are located physically together. Team meetings to verbally discuss progress and issues occur daily, but with strict time limits.
- At least some of the agile methods stipulate pair-wise programming (two persons code the same part of the system) as a means of sharing knowledge and as a quality check.
- A change in the role of the project manager, from one primarily concerned with planning the project, allocating tasks and monitoring progress to that of a facilitator and advocate. Responsibility for planning and control is delegated to the team members.

Exhibit

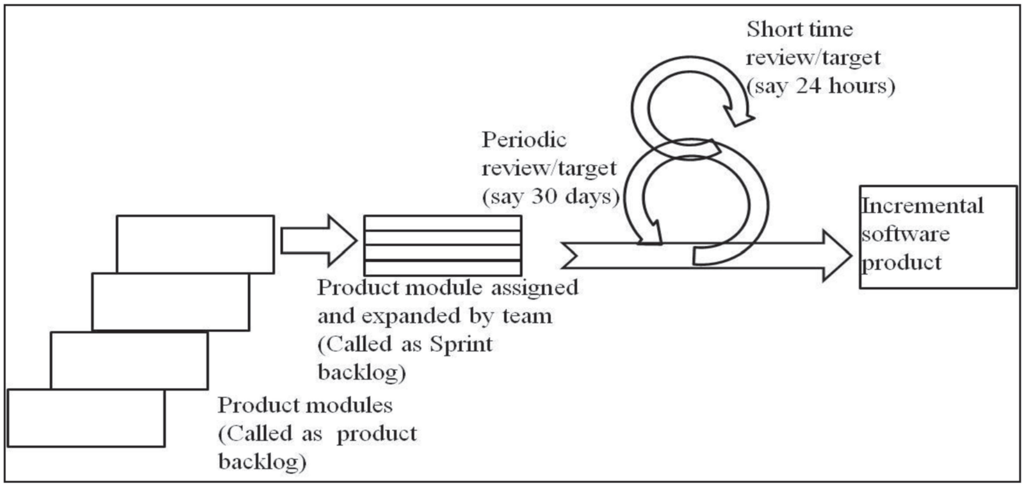

Short time review/target (say 24 hours)
Periodic review/target (say 30 days)
Incremental software product
Product module assigned and expanded by team (Called as Sprint backlog)
Product modules (Called as product backlog)

Figure 4.7: Agile (Sprint) review cycle

The agile methodology may be considered as iterative and *incremental* development, where requirements and solutions evolve through collaboration between self-organising, 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.

Key features of agile methodologies

- Customer satisfaction by rapid delivery of useful software;
- Welcome changing requirements, even late in development;
- Working software is delivered frequently (weeks rather than months);
- Working software is the principal measure of progress;
- Sustainable development, able to maintain a constant pace;
- Close, daily co-operation between business people and developers;
- Face-to-face conversation is the best form of communication (co-location);
- Projects are built around motivated individuals, who should be trusted;
- Continuous attention to technical excellence and good design;
- Simplicity; Self-organizing teams; and
- Regular adaptation to changing circumstances.

Strengths

- Agile methodology has the concept of an adaptive team, which enables to respond to the 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 a 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

- In case of some software deliverables, especially the large ones, it is difficult to assess the efforts required at the beginning of the System Development life cycle.
- There is lack of emphasis on necessary designing and documentation due to time management and generally is left out or incomplete.
- Agile increases potential threats to business continuity and knowledge transfer due to verbal communication and weak documentation.
- Agile requires more re-work and due to the lack of long-term planning and the lightweight approach to architecture.
- The project can easily get taken off track if the customer representative is not clear about the requirements and final outcome.
- Agile lacks the attention to outside integration.

4.3.3 Software reengineering and reverse engineering

Software reengineering

Reengineering as name suggest is a process of updating an existing system by reusing design and programme components. Although it updates existing software it is used in case of major changes in existing system, it differs from change management due to the extent of changes. These changes typically prompts for new software development project, however as interim solution a reengineering project is initiated. A number of tools are now available to support this process.

What is software reengineering?

- Restructuring or rewriting part or all of a system without changing its functionality
- Applicable when some (but not all) subsystems of a larger system require frequent maintenance
- Reengineering involves putting in the effort to make it easier to maintain
- The reengineered system may also be restructured and re-documented

When to reengineer?

- When system changes are confined to one subsystem, the subsystem needs to be reengineered
- When hardware or software support becomes obsolete
- When tools to support restructuring are readily available
- When some business processes or functions are reengineered

Software Reengineering Activities

- **Inventory analysis –** Listing and identifying active software applications and components required by business. The attributes of applications can be criticality, longevity, current maintainability. This helps in identifying reengineering criteria.
- **Document restructuring –** Identify documentation for identified applications/modules. (Identifying poor or weak documentation for re-documentation sometimes considered as reengineering activity)
- **Design recovery –**Identify the design of the application module to be reengineered. (In case it is not available it may have to be built based on code or use reverse engineering method)
- **Reverse engineering –** Process of design recovery – analysing a programme in an effort to create a representation of the programme at some abstraction level higher than source code
- **Code restructuring –** Source code is analysed and violations of structured programming practices are noted and repaired, the revised code also needs to be reviewed and tested
- **Data restructuring –** Usually requires full reverse engineering, current data architecture is dissected and data models are defined, existing data structures are reviewed for quality
- **Forward engineering –** Also called reclamation or renovation, recovers design information from existing source code and uses this information to reconstitute the existing system to improve its overall quality and/or performance.

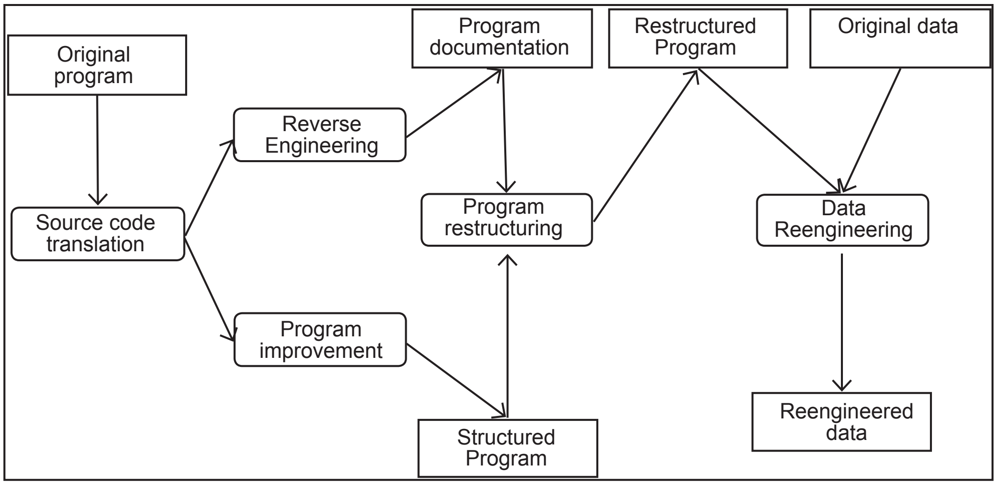


Figure 4.8: Reengineering and reverse Engineering

Reverse Engineering

Reverse engineering is the process of studying and analysingan application, a software application or a product to see how it functions and to use that information to develop a similar system. This process can be carried out in several ways:

- Decompiling object or executable code into source code and using it to analyse the programme
- Black box testing the application to be reverse-engineered to unveil its functionality

The major advantages of reverse engineering are:

- Faster development and reduced SDLC duration
- The possibility of introducing improvements by overcoming the reverse-engineered application drawback

The IS auditor should be aware of following risks:

- **Software licence agreements often contain clauses prohibiting the licensee from reverse engineering the software so that any trade secrets or programming techniques are not compromised.**
- **Decompilers are relatively new tools with functions that depend on specific computers, operating systems and programming languages. Any change in one of these components may require developing or purchasing a new decompiler.**

4.3.4 Object Oriented Software Development (OOSD)

OOSD differs from traditional SDLC approach which considers data separately from the procedures that act on them (e.g., programme and database specifications). OOSD is the process of solution specification and modelling where data and procedures can be grouped into an entity known as an object. An object's data are referred to as its attributes and its functionality is referred to as its methods. Proponents of OOSD claim the combination of data and functionality is aligned with how humans conceptualise everyday objects.

Objects usually are created from a general template called a class. The template contains the characteristics of the class without containing the specific data that need to be inserted into the template to form the object.

Classes are the basis for most design work in objects. Classes are either super-classes (i.e., root or parent classes) with a set of basic attributes or methods, or subclasses which inherit the characteristics of the parent class and may add (or remove) functionality as required. In addition to inheritance, classes may interact through sharing data, referred to as aggregate or component grouping, or sharing objects.

Aggregate classes interact through messages, which are requests for services from one class (called a client), to another class (called a server). The ability of two or more objects to interpret a message differently at execution, depending on the superclass of the calling object, is termed polymorphism.

For example consider a car owned by you as an object. The object is complete in itself and all necessary data (components and specifications) are embedded into object. You can use this object for specific use it has been designed for. However there are different objects either having similar data (same model, same company) or different data (Different model, different companies etc.) These all objects belong to class cars. All object cars have common attributes (i.e. steering, gear, break, wheels etc.) that are inherited from class cars (or may be from superclass vehicles). One can modify car by keeping basic common attributes and add few more functions to it. (Polymorphism)

There are many programming languages that are used for developing object oriented systems. To realise the full benefits of using object-oriented programming, it is necessary to employ object-oriented analysis and design approaches. Dealing with objects should permit analysts, developers and programmers to consider larger logical chunks of a system and clarify the programming process. Although it is possible to do object-oriented development using a waterfall model in practice most object-oriented systems are developed with an iterative approach. As a result in object-oriented processes "analysis and design" are often considered at the same time. OOSD being programming method, a particular programming language, or use of a particular programming technique, does not imply or require use of a particular software development methodology.

Advantages of OOSD

- The ability to manage an unrestricted variety of data types
- Provision of a means to model complex relationships
- The capacity to meet the demands of a changing environment

A significant development in OOSD has been the decision by some of the major players in object-oriented development to join forces and merge their individual approaches into a unified approach using the Unified Modelling Language (UML). UML is a general-purpose notational language for specifying and visualizing complex software for large object-oriented projects. This signals a maturation of the object-oriented development approach. While object-orientation is not yet pervasive, it can accurately be said to have entered the computing mainstream.

Applications that use object-oriented technology are:

- Web applications
- E-business applications
- CASE for software development
- Office automation for e-mail and work orders
- Artificial intelligence
- Computer-aided manufacturing (CAM) for production and process control

4.3.5 Component based development

Component-based development is an outgrowth of object-oriented development. Component-based development is in fact assembling packages of executable software that make their services available through defined interfaces. These packages also called as enabling pieces of programmes are called objects. These objects are independent of programming languages or operating system. The basic types of components are:

- **In-process client components:** These components must run from within defined programme (called as 'container') such as a web browser; they cannot run on their own.
- **Stand-alone client components:** Applications (like Microsoft's Excel and Word) that work as service.
- **Stand-alone server components:** Processes running on servers that provide services in standardised way. These are initiated by remote procedure calls or some other kind of network call. Technologies supporting this include Microsoft's Distributed Component Object Model (DCOM), Object Management Group's Common Object Request Broker Architecture (CORBA) and Sun's Java through Remote Method Invocation (RMI).
- **In-process server components:** These components run on servers within containers. Examples include Microsoft's Transaction Server (MTS) and Sun's Organisation Java Beans (EJB)

A number of different component models have emerged. E.g. Microsoft's Component Object Model (COM). MTS when combined with COM allows developers to create components that can be distributed in the Windows environment. COM is the basis for ActiveX technologies, with ActiveX Controls being among the most widely used components. Alternative component models include the CORBA Component Model and Sun's EJB.

COM/DCOM, CORBA and RMI are sometimes referred to as distributed object technologies or also termed middleware. (Middleware is a broad term, but a basic definition is software that provides run-time services where by programmes/objects/components can interact with one another).

Visual tools are now available for designing and testing component-based applications. Components play a significant role in web-based applications.

Advantages of component-based development are:

- This reduces development time as application system can be assembled from prewritten components and only code for unique parts of the system needs to be developed.

- Improves quality by using prewritten and tested components.
- Allows developers to focus more strongly on business functionality.
- Promotes modularity by encouraging interfaces between discrete units of functionality.
- Simplifies reuse and avoids need to be conversant with procedural or class libraries.
- Combining and allowing reusable code to be distributed in an executable format—i.e., no source is required.
- Reduces development cost as less effort is required for designing and building the software.
- Supports multiple development environments due to platform independent components.
- Allows a satisfactory compromise between build and buy options i.e. instead of buying a complete solution, it could be possible to purchase only needed components and incorporate these into a customised system.

Disadvantages

- Attention to software integration should be provided continuously during the development process.
- If system requirements are poorly defined or the system fails to adequately address business needs, the project will not be successful.

4.3.6 Web-based application development

Web based development has modified client-server architecture and made it more light weight and easy to implement. It has eliminated the need to implement client module on end-user's desktop, and is delivered via internet based technologies. User need to know URL to access the application and if need be the same is delivered on users' desktop or executed from web server. The technology can be deployed within organisation also. For example many organisations have implemented internal user services using intranet portal, which essentially uses internet (web) based technologies.

Historically, software written in one language on a particular platform has used a dedicated application programming interface (API). The use of specialised APIs has caused difficulties in integrating software modules across platforms. Component based technologies such as CORBA and COM that use remote procedure calls (RPCs) have been developed to allow real-time integration of code across platforms. However, using these RPC approaches for different APIs still remains complex. Web-based application development is designed to further facilitate and standardise code module and programme integration.

Web-based application development enables users to avoid the need to perform redundant computing tasks with redundant code. For example installing client on all users after making changes or change of address notification from a customer need not be updated separately in multiple databases. For example entering and maintaining same data in contact management, accounts receivable etc. Web application development though is different than traditional developments (e.g. users test and approve the development work), but the risks of application development remain the same.

With web-based application development, an XML language known as Simple Object Access Protocol (SOAP) is used to define APIs. SOAP will work with any operating system and programming language that understands XML. SOAP is simpler than using the more complex RPC-based approach, with the advantage that modules are coupled loosely so that a change to one component does not normally require changes to other components.The second key component of web development is the Web Services Description Language (WSDL), which is also based on XML. WSDL is used to identify the SOAP specification that is to be used for the code module API and the formats of the SOAP messages used for input and output to the code module. The WSDL is also used to identify the particular web service accessible via a corporate intranet or across the Internet by being published to a relevant intranet or Internet web server.

4.4 Summary

Software development is the key to automate business processes using information technology. With changing technologies the SDLC models and methods have undergone many changes. However, the basic risks of SDLC still exist and these are:

- Poor coding techniques
- Inadequate documentation
- Inadequate QC and QA (including testing inadequacies)
- Lack of proper change control and controls over promotion into production
- Inappropriate processes for development processes and deliverables
- Technical vulnerabilities, and how these could materialize and be controlled/addressed

An IS auditor may not have to master all the development methods but must focus on a risk-based approach in the assessment of SDLC process. The approach should be to identify the business goals and supporting IT goals related to the development and based on these identify the risks. During SDLC audit the focus has to be on application development risks and associated business risks. Some of these controls may look similar for all application development activity. However, these will have to be relevant to the development activity that is taking place in the area which are the subject of audit.

4.5 Questions

1. A SDLC project for updating existing application has been initiated, however project manager has realized that the documentation has not been updated and source code is not available. Which of the following method shall help the project manager?
   A. Business process reengineering
   B. Reverse engineering
   C. Component based development
   D. Agile development method
2. Reusing already developed programmes helps project manager in improving productivity. Which of the following methodology is outcome of this concept?

A. Agile development methodology
B. Object oriented software development
C. Component based development
D. Rapid application development

3. Which of the following is a major weakness of agile development methodology?
A. High dependence of user interaction
B. Reduced focus on recording design
C. Reduced size of development team
D. Project manager is in role of facilitator

4. The primary advantage of prototyping is that it
A. Increases user interaction
B. Helps in outsourcing decision
C. Used in rapid application development
D. Helps in finalising user requirements

5. Which of the following model addresses the weakness of waterfall model related to accommodating changes during development stage?
A. Spiral model
B. Incremental model
C. Validation and verification
D. Web development method

6. Which of the following activity is most important for IS auditor conducting mid-term review of SDLC?
A. Ensure risks associated with development method are controlled
B. Review appropriateness of development method
C. Extrapolate the completion time based on current state
D. Perform code review of programmes developed so far

4.6 Answers and Explanations

1. B. Reverse engineering helps in understanding existing system and therefore the primary requirements that can be reengineered to develop new application. Project manager may use any development method depending upon type of application to be developed and deployed.

2. C. Reusing already developed programme is most common in software development. However the development model based on this concept is called component based development where programme components are already developed and programmer may just use them directly. In Object oriented method Object and classes are reused after customizing. Agile and Rapid development are not primarily developed for reusability.

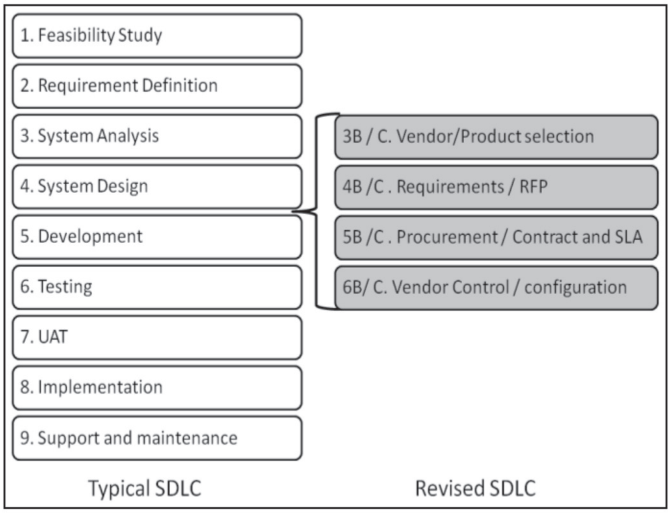
3. B. The major challenge faced by Agile development is weak documentation of design and related record. Since it heavily depends upon working in interaction with users, where small team of experts captures requirements and develops functional code that can be deployed.
4. D. Prototyping helps in finalising user requirements where users can review the prototype and confirm if it meets the requirements. This method is used in various models like rapid application development, agile development, spiral model etc.
5. C. Waterfall model requires finalisation of earlier phase before beginning next phase. This makes it difficult to accommodate subsequent changes. Validation and verification method helps in overcoming this weakness by introducing iteration during each phase.
6. A. IS auditor should ensure that objectives of organisation in initiating SDLC project are met by confirming that the risks associated with development are controlled.

CHAPTER 5: SYSTEM ACQUISITION FRAMEWORK

Learning objectives

This chapter focuses on steps required to be followed by the project manager if the organisation decide to acquire the software solution. This includes identifying different types of solutions available in market, the techniques on how to analyse them and select the right solution as required. In case organisation decides to outsource the development the steps required to be followed in selecting and monitoring the vendor are discussed. This chapter covers phases 3 B to 6B and 3C to 6C in changed SDLC described in chapter 1.

5.1 Introduction


1. Feasibility Study
2. Requirement Definition
3. System Analysis
4. System Design
5. Development
6. Testing
7. UAT
8. Implementation
9. Support and maintenance
3B / C. Vendor/Product selection
4B /C . Requirements / RFP
5B /C . Procurement / Contract and SLA
6B/ C. Vendor Control / configuration
Typical SDLC
Revised SDLC

This chapter provides information on Phases 3B to 6B covering software acquisition processes and 3C to 6C outsourcing of system development (shadowed area).

This chapter discusses the process generally followed by various organisations for software acquisition and outsourcing.

IS auditor primarily should be looking for control aspects associated with outsourcing to ensure that expected benefits can be derived.

Changes in technology have introduced additional phase in traditional SDLC such assoftware acquisition. A decision for acquiring software may be reached due to various reasons such as availability of commercial software, lack of skilled resources within organisation; the cost of in-house development is higher than benefits of acquiring, focus on core competency, etc. Software acquisition is the process that should occur after the requirements definition phase.

Depending on requirement there could be three cases:

1. **Generic products without customisation:** Software is available and can be implemented without customisation. These products are also known as Plug-and-play or COTS (Commercial of the shelf) for example MS Office, MS projects etc.
2. **Commercial product with customisation:** Software needs to be customised like ERP or core banking products or at lower level customisation like Tally.
3. **Outsourced development:** Ready-made software as required is not available. Hence, the organisation intends to outsource development activities based on cost benefit analysis.

1. **Software acquisition decision must be supported by users and users need to be actively involved in evaluation and selection process.**
2. **The feasibility study should contain documentation that supports the decision to acquire the software.**
3. **This chapter describes the acquisition process for application software. Acquisition of supporting software (like operating system, Database and Middleware) is not discussed here since acquisition of these supporting software depends upon primary application, organisational policies and internal standards., However, the processes discussed here can be followed for acquiring these products also.**

Table 5.1 shows the high-level steps in software acquisition process and their applicability for each of three cases mentioned above.

Software Acquisition Steps	Generic products without customization	Commercial product with customization	Outsourced development
Requirement analysis	General requirements	Detail of functionality	Highly granular requirement analysis is required
Gap Analysis	For record	Can be reduced by vendor	Not applicable, as the vendor shall be developing required functionalities
Selection of product	User involvement is necessary	User Involvement is Must	User Involvement is for specifying requirements
Request for Proposal (RFP)	May or May not required	Required	Required
Selection of vendor	General	Specific based on requirements and support	Specific based on requirements, Sustainability and skills
Service level Agreement (SLA)	may not required	Required for support	Essential since the product is specific to organization
User Acceptance (UAT)	may not required	Detailed UAT and sign off for configuration	Detailed UAT and Sig-off for functionalities
Implementation Support	may not required	Required	Required
Maintenance Support	On call	On demand	On going
Ownership of software	License to use	Licensed to use based on contract	Depends upon agreement, organization can own software
Vendor monitoring	Not applicable	Only for support service	Continuous monitoring

Table 5.1: Software acquisition – comparison of steps

5.2 Requirements Analysis

Defining detailed requirement is most essential in case of software acquisition, as selection of product or details of contract for outsourcing development, heavily depends on meeting the maximum or all requirements. This requires involvement of users in thorough and detailed understanding of the system, identifying areas that need to be covered to solve the problem and determination of monitoring requirements to select appropriate solution.

Requirement analysis focuses in achieving the following objectives:

- To consult user management (stakeholders) to determine their expectations.
- To analyse requirements to detect conflicts and determine priorities
- To gather data by various means to ensure requirement gathering is complete.
- To verify that the requirements are complete, consistent, unambiguous, verifiable, modifiable, testable and traceable.

The specific steps which help in achieving the objectives are:

Fact Finding: Application system focuses on two main types of requirements. The first one is service delivery and second one is operational requirements. These may include lower operational costs, better information for managers, smooth operations for users or better levels of services to customers. To assess these needs, the analysts often interact extensively with stakeholders, to determine 'detail requirements'. The fact-finding techniques/tools used by the system analyst include document verification, interviews, questionnaire and observation.

Analysis to understand Present process: Understanding present system and its related problems helps in confirming the requirements from new application/software. Generally this includes identifying rationale and objectives, inputs and data sources, decision points, desired outcomes from application, mandatory and discretionary controls.

Requirements for Proposed Systems: Analysis of functional area and process, the proposed expectations can be clearly defined considering the issues and objectives. The requirements should cover at the minimum the following:

- Outcome to different users from the application.
- Online processing capabilities and expected response time.
- Sources and nature (manual, online etc.) of data to be captured and processed.
- Methods and procedures that show the relationship of inputs and outputs to the database, utilise data communications as, when and where deemed appropriate.
- Work volumes and timings are considered for present and future including peak periods.

5.3 Product selection

Based on requirements, organisations may proceed to select suitable product to meet the requirements. The three options which are considered are:

- **Generic products without customisation:** Software is available and can be implemented without customisation.
- **Commercial product with customisation:** Software needs to be customised
- **Outsourced development:** Software is not available however organisation wishes to outsource development activities based on cost benefit analysis.

In case organisation opts for third option then it can initiate the process for RFP. In first two cases it is likely that more than one product/vendor fits the requirements with advantages and disadvantages. It is best to have adequate participation from various user groups is included when evaluating the product.

The organisations may consider following aspects for evaluating the software product:

- **The agenda-based presentations** are scripted business scenarios that are designed to show how the software will perform certain critical business functions. Vendors are typically invited to demonstrate their product and follow the sample business scenarios given to them to prepare.
- **Checklists or questionnaire:** A most simple but subjective method where criteria are listed as a check list or questions against which functional requirements for various

products. Organisation may get this information from vendors as part of request for proposal (RFP) or request for intent (RFI). These responses for different products are validated against requirements. For example, support service checklists may have parameters such as performance; system development, maintenance, conversion, training, back-up, proximity, hardware and software.

- **Point-Scoring Analysis (Functional gap analysis):** Point-scoring analysis provides an objective means of selecting software. This is performed by allotting weight-age for each requirement and then allotting score to the software that meets that requirement. Table 5.2 illustrates a Point Scoring Analysis list.

Table 5.2: Point Scoring Analysis List

Software Evaluation Criteria	Possible points	Software A	Software B	Software C
Does the software meet all mandatory specifications?	10	7	9	6
Will programme modifications, if any, be minimal to meet company needs?	10	8	9	7
Does the software contain adequate controls?	10	9	9	8
Is the performance (speed, accuracy, reliability, etc.) adequate?	10	7	9	6
Are other users satisfied with the software?	8	6	7	5
Is the software user-friendly?	10	7	8	6
Can the software be demonstrated and test-driven?	9	8	8	7
Does the software have an adequate warranty?	8	6	7	6
Is the software flexible and easily maintained?	8	5	7	5
Is online inquiry of files and records possible?	10	8	9	7
Will the vendor keep the software up-to-date?	10	8	8	7
Totals	**123**	**94**	**106**	**85**

- **Public Evaluation Reports:** Organisation may refer to independent agencies that evaluate various software products of different vendors and publish comparison along with rating based on various predefined parameters including survey of current users. (For example, magic quadrant for similar software product by Gartner, Forester etc.). This method has been frequently and usefully employed by several buyers in the past.

- **Proof of Concept (PoC) or Benchmarking Solutions:** Organisations may request vendor to provide a proof of concept (by implementing product in small pilot area within organisation) that the software meets the expected requirements. This helps organisation in evaluating best product that meets the requirements. This is particularly useful for products that has high-cost and requires high level of efforts that it may not be possible to roll back.

- **Acceptability by end users:** This is an important parameter to be considered many times software may be able meet maximum functional requirements but users may not accept it. In such situations user acceptability needs to be considered on priority.

5.4 Request for proposal

Once the software is shortlisted organisation may invite request for proposal from the selected vendors. In case it has been decided to outsource the development, organisation may also prepare a RFP. The invitation to respond to an RFP should be published/forwarded to appropriate vendors. Organisations have to carefully examine and compare the vendors' responses using an objective method such as a scoring and ranking methodology. Response from vendors may be evaluated using parameters listed in Table 5.3.

Table 5.3: Parameters for preparing RFP

Item	Description
Software and system requirements	Comparison of product functionalities against requirements based on score point criteria.
Customer References	Validate the vendor's claims about their product performance and timely completion of work by the vendor from the vendor's customers.
Vendor viability and financial stability	Evaluate the vendor's viability with reference to period for which the vendor is in operation, the period for which the desired product is being used by the existing customers and the vendor's financial stability on the basis of the market survey and the certification from the customers and on certain supporting documentation from the Vendor
Availability of complete and reliable documentation about the new software	Review the complete set of system documentation provided by the vendor prior to acquisition of the software. The documentation should be in detail and precise.
Vendor support	Evaluate what kind of support the vendor provides for the software like onsite maintenance, online updating of upgrades, onsite training to users, automatic new version notifications, 24 x 7 helpline etc.
Response time	Time taken for the vendor to respond and fix in case a problem is reported.
Source code availability	If the software is developed only for the concerned business, the vendor has no right to further sale. The source code must be given by Vendor. In other case, the source code should be deposited with a third party so that the same would be available if the vendor goes out business. The source with the programme updates and programmes fixes should be included as a part of escrow agreement.

Vendor's experience	Vendor having longer experience in the desired software is more acceptable.
A list of recent or planned enhancements with dates	This will involve review of the vendor's effort to keep the product current.
List of current customers	More the number of customers, greater are the acceptability of the product.
Acceptance testing of product	The vendor should allow the acceptance testing by the users to ensure that the product satisfies the system requirements of the business. This should be done before commitment of the purchase.

5.5 Vendor selection criteria/process

After the RFP responses have been examined, organisation may be able to short list vendors whose product satisfies most or all of the stated requirements in the RFP. In evaluating the best-fit solution and vendor against the given set of business requirements and conditions, a suitable methodology of evaluation should be adopted. The methodology should ensure objective, equitable and fair comparison of the products/vendors (e.g., a gap analysis to find out the differences between requirements and software, the parameters required to modify, cost of product against the benefits derived etc.). Many organisations follow lowest (cost) amongst equal (in meeting requirements) principle.

It is important to keep in mind the minimum and recommended requirements to use the software including:

- Required hardware such as memory, disk space, and server or client characteristics
- Operating system versions and patch levels supported
- Additional tools such as import and export tools
- Databases supported

While selecting the vendor users should concentrate on following criteria:

- **Reliability:** Are the vendor's deliverables (enhancements or fixes) dependable?
- **Commitment to service:** Is the vendor responsive to problems with its product? Does the vendor deliver on time?
- **Commitment to providing training, technical support and documentation for its product:** What is the level of customer satisfaction?

5.5.1 Service Level Agreements (SLAs)

Service Level Agreement (SLA) is formalisation of the outsourcing/acquisition activity. This activity follows the vendor selection process that selects the final vendor for supply/develop, implement and support the software product. During this phase organisation and vendor negotiate the terms of contract (SLA) and sign it. Appropriate legal counsel should review the contract prior to its signing. The contract should cover the following items, depending upon terms of requirements:

- Specific description of services, deliverables and their costs
- Commitment dates for deliverables and support
- Commitments for delivery of documentation, fixes, upgrades, new release notifications and training
- Commitments for data migration
- Arrangement for a software escrow agreement or deliverables of source code and system documentation
- Description of the support to be provided during installation/customisation
- Criteria for user acceptance
- Provision for a reasonable acceptance testing period, before the commitment to purchase is made
- Allowance for changes to be made by the purchasing company
- Terms of software maintenance
- Allowance for copying software for use in business continuity efforts and for test purposes
- Payment schedule linked to actual delivery dates
- Confidentiality clauses
- Data protection clauses
- Monitoring mechanism and measurement criteria along with reporting commitments. This is particularly important for development and maintenance activities.

Software Licences: Software licence is a contract that grants permission to use software to the organisation that is protected under legal system (like copyright law, patent law, trademark law and any other intellectual property rights), by paying agreed fee. Use of unlicensed software or violations of a licensing agreement expose organisations to possible litigation. Organisation and a software vendor should clearly describe the rights and responsibilities of the parties to the contract. The contracts should cover detail to provide assurances for performance, source code accessibility, software and data security, and other important issues.

5.5.2 Vendor monitoring

Outsourcing helps in improving speed and quality and reducing total costs. Outsourcing also introduces risks, particularly after signing the agreement since vendor becomes a single- or sole-source provider. Proper vendor management can reduce or prevent problems caused by picking a vendor that is unable to achieve the required solution or timescale and by ensuring that contracts address business needs and do not expose the business to unnecessary risk.

Managing the contract involves a monitoring of vendor activities to ensure that deployment efforts are controlled, measured and improved. This includes periodic status reporting, milestones and metrics as agreed with the vendor.

Critical success factors of vendor selection process

1. Involvement of users.
2. Priority to meeting requirements and not commercials.
3. If requirements meets then the lowest bid must be considered.
4. Organisation should maintain the documentation about vendor selection process.

The IS auditor has to understand the RFP and requirements specifications. The review covers:

- Security requirements
- Process of vendor selection
- Contents of the contract
- Monitoring of terms of contract
- Inclusion of provision of the right to audit the vendor

5.6 Summary

1. The feasibility study in relation to the software acquisition should contain documentation that supports the decision to acquire the software.
2. The decision is based upon various factors such as cost difference between development and acquisition, availability of the required software readily in the market, the time saved between development and acquisition.
3. A team comprising of technical supports staff and key users should be created to prepare Request for Proposal (RFP).
4. Organisation should carefully compare and examine the various responses of the Vendor to the RFP, prepare the gap analysis and shortlist the vendors on the basis of such evaluation.
5. The organisation should invite the short listed vendors to have presentation of the product and the processes. The user’s participation with feedback must be ensured in such presentation.
6. The last step in the acquisition process is the negotiation and signing of a legal contract with the selected vendor.
7. Managing and control on the implementation of the system is required with the regular status reports.

IS auditors are involved in the software acquisition process to determine whether an adequate level of security controls has been considered prior to any agreement being reached. If security controls are not part of the software, it may become difficult to ensure data integrity for the information that will be processed through the system. Risks involved with the software package include inadequate audit trails, password controls and overall security of the application. To ensure effective mitigation of these risks, IS auditor should ensure that these controls are built into the software application.

5.7 Questions

1. While auditing the software acquisition process the IS auditor PRIMARILY review which of the following to understand the benefits to the business?
   - A. System requirement specifications
   - B. Cost comparison of different products
   - C. Alignment of IT strategy with business
   - D. Vendor with lowest cost has been selected
2. While auditing outsourced software development the IS auditor primarily ensure that:
   - A. Vendor selected has experience in similar engagements
   - B. Organisation has followed established procurement process
   - C. Outcome of feasibility study has indicated for outsourcing
   - D. Ownership of design and source code has been established
3. Which of the following is primary objective of monitoring activities of vendor hired for software development?
   - A. Invoke the penalty clause in contract
   - B. Mitigate risk associated with performance
   - C. Ensure senior management satisfaction
   - D. Monitor third-party resource performance
4. While acquiring application software the organisation has finalized products as follows:

Product	Functional requirements coverage	User acceptability
A	96%	50%
B	88%	75%
C	80%	95%
D	69%	80%

   The organisation should select product:
   - A.
   - B.
   - C.
   - D.
5. Which of the following is a primary concern of management while considering acquisition of new software?
   - A. New application does not have long term operational issues.
   - B. Vendor for application may not provide source code.
   - C. User acceptance testing has been performed but not signed off.
   - D. Change management for application is not defined.

6. Compliance with terms of licence for software purchased is primarily which type of compliance?

   A. Legal

   B. Contractual

   C. Regulatory

   D. Standard

5.8 Answers and Explanations

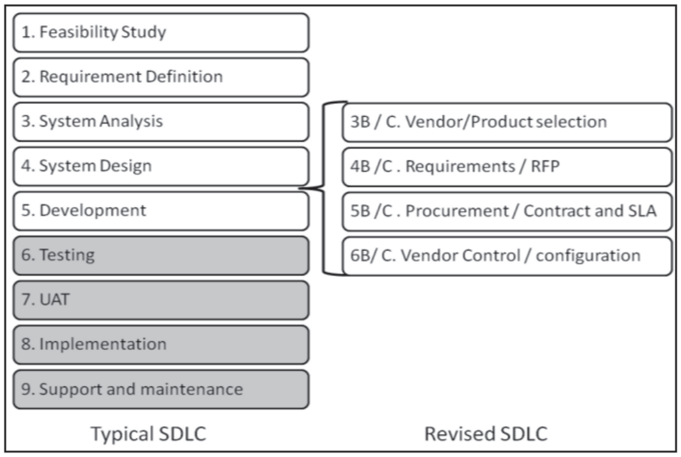
1. A. While acquiring software organisations must focus on requirement specifications so as to select most appropriate product. If the products does not meet requirement, it may not be considered.
2. D. While outsourcing the software development the IS auditor must ensure that the ownership of developed software is within the organisation. In absence of this provision organisation may lose the IPR. Other things are secondary.
3. B. The primary objective of monitoring vendor activities is that the non-performance by vendor should not impact the organisation's benefit realisation plan.
4. C. The effective requirements meeting for each product is A = 48% (96*.50), B = 66% (88*.75), C = 76 % (80*.95), D = 55% (69*.80)
5. D. Absence of change management process is a major hurdle in support and maintenance of application software. Other issues can be managed.
6. B. Terms of software licence are governed by the contract between software vendor and purchaser. Hence, it is a contractual compliance.

CHAPTER 6: IMPLEMENTATION AND MAINTENANCE

Learning objectives

This chapter focuses on providing information on implementation methodologies that organisation may adopt depending upon the nature and complexity of solution developed or acquired and nature of services offered by the solutions. The chapter also focuses on providing information on activities that need to be performed before and after implementation (maintenance of application).

6.1 Introduction


1. Feasibility Study
2. Requirement Definition
3. System Analysis
4. System Design
5. Development
6. Testing
7. UAT
8. Implementation
9. Support and maintenance
3B / C. Vendor/Product selection
4B /C . Requirements / RFP
5B /C . Procurement / Contract and SLA
6B/ C. Vendor Control / configuration
Typical SDLC
Revised SDLC

This chapter discusses Phases 6 to 9 of SDLC

Phase 6 is testing of developed or configured software.

Phase 7 is conducting user acceptance testing (UAT) and sign-off from users. (UAT is discussed in testing section.)

Phase 8 is implementation methods being used by various organisations.

Phase 9 is change management process required for support and maintenance activities

Application software developed or acquired requires appropriate process for implementing and maintenance in order to realise benefits effectively and efficiently. The best solution to ensure smooth implementation, configuration, change and release management processes need to be implemented. These processes should be focused on reliability, availability and security of production systems. Maintenance generally involves updating and changing IT systems (application and infrastructure) to suit to changing requirement of business. Every change must be assessed, planned, tested, approved, documented, communicated and carried out without any undesirable consequences and minimum downtime for business processes. The IS auditor should be aware of the processes for implementation and change management including controls to ensure segregation of duties between development, test and the production environment.

The change management processes discussed in Section 6.4 and configuration management process discussed section 6.5 are common processes used for implementation of changes as well as new application systems.

6.2 Testing

The success of information systems depends upon the quality of software that supports the system. Testing of software before deploying in production to ensure it delivers as per requirements is most essential aspect of quality apart from documentation, compliance with coding standards, version control discipline and user training.

Although testing phase comes much later in the life cycle, planning for testing starts with the commencement of System Development life cycle i.e. during requirement gathering phase. Testing is a process that focuses on correctness, completeness and quality of developed computer software. Testing should systematically uncover different classes of errors in a minimum amount of time with a minimum amount of efforts. The data collected through testing can also provide an indication of the software's reliability and quality. However, testing cannot show the absence of defect, it can only show that software defects are present. There are various types of testing performed during development of life cycle that are discussed in ensuing paragraphs.

6.2.1 Unit Testing

In computer programming, unit testing is a verification and validation method in which a programmer tests whether individual units of source code are fit for use. A unit is the smallest functional part of an application often called as module. It can be an individual programme, function, procedure, or may belong to a base/super class, abstract class or derived/child class.

Unit tests are typically written based on requirement specifications and run by testing professionals or software developers to ensure that code meets these requirements and behaves as intended. The goal of unit testing is to isolate each component of the programme 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 typically performed on a programme unit. Such typical tests are described as follows:

1. **Functional Tests:** Functional Tests check 'whether programmes 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. These test data values are prepared in advance for all possible permutations the data can acquire during live run. This can have two types:
   a. **Positive test:** Where tester collects the expected values the data can possess. Sometimes tester may use sanitized live data for testing.
   b. **Negative test:** Where tester provides value sets that data should not possess anytime. Here the programme should flash the error with suitable message.

For example, if data field is used to store amount and can acquire a value between 0 and 1 crore, positive test should provide expected results and negative test should flash error if values are beyond the specified range.

2. **Performance Tests:** Performance tests are designed to verify the expected performance criteria of programme.

   There are different performance parameters like response time (time required to receive input and deliver confirmation), execution time (processing of single data value should be less than 100 microseconds), throughput (1000 values must be processed in once second), primary (RAM/CPU) and secondary memory (Storage) utilisation and rate of traffic flow on data channels and communication links (number of messages per second).

3. **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 programme in various ways. The purpose of a stress test is to determine the limitations of the programme. (For example if access to web application is expected to be 10000 hits per second, if the programme can stand this load and how it behaves when load exceeds or during a sort/search operation, available memory can be reduced to find out whether the programme is able to handle the situation.).

4. **Structural Tests:** Structural Tests are concerned with examining the internal processing logic of a software system. Particularly when the programme is expected to behave differently depending upon value of data set. Programmer may code for known values and might forget to code for unknown values where programme might misbehave. For example tax calculation where depending upon value different rates is applied or if division operation is involved and data set gets value of zero, programme may terminate abruptly or go in loop without response.

5. **Parallel Tests:** These are applicable during change management or reengineering where the same test data is used in the new and old system and the output results are then compared.

Methods of Unit Testing

A. Static analysis testing and

B. Dynamic testing.

A. Static Testing: Static analysis tests are conducted on source programmes and do not normally require executions in operating conditions. Typical static analysis techniques include the following:

i. **Desk check:** This is done by the programmer to check the logical syntax errors, and deviation from coding standards. As name suggests programmer uses paper and pen to verify the logic of code by jotting down values of data sets and thinking like computer to arrive at possible values.

ii. **Structured Walk-through:** Desk check performed with team or peers who scan through the text of the programme and explanation try to uncover errors.

iii. **Code Inspection:** The programme is reviewed by a formal committee. Review is done with formal checklists.

B. Dynamic Analysis Testing: Such testing is normally conducted through execution of programmes in operating conditions. Typical techniques for dynamic testing and analysis include the following:

i. **Black Box Testing:** This testing primarily focuses on functionalities as specified in requirements. It assumes code set (programme/module/function etc.) as a black box without knowledge of internal structure, and focuses on input values and compares expected output for each value. The test designer selects typical inputs including simple, extreme, valid and invalid input-cases and executes to uncover errors.

This method of test design is applicable to all levels of software testing i.e. unit, integration, functional testing, system and acceptance. 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 perform, for example: running of a Trojan code, the black box test does not identify it.

ii. **White Box Testing:** This testing goes further than black box testing and also tests internal perspective of the code/system/function to design test cases. The tester chooses test case inputs to exercise paths through the code and determines the appropriate outputs. Since the tests are based on the actual implementation, if the implementation changes, the tests probably will need to change, too. 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. There are automated tools (like test tools/debuggers etc.) available to conduct this type of testing.

iii. **Grey Box Testing:** It is a technique that is in between uses black box testing and white box testing. Tester applies a limited number of test cases to the internal workings of the software under test. In the remaining part of the grey box testing, one takes a black box approach in applying inputs to the software under test and observing the outputs.

6.2.2 Integration Testing

Unit testing focuses on testing of different modules/functions and programmes that are small part of entire information system being developed. These modules are expected to work together to achieve objectives of information system. For example Internet banking is a system consisting of various functions like saving account management, time deposit management, loan account management, third-party fund transfer, standing instruction, getting statements of accounts etc. While developing programmes/functions, each service function is developed separately and tested in unit testing. Now it is necessary to test if these modules/functions work together seamlessly and communicate appropriately during execution. Objective is to evaluate the validity of integration of two or more components that pass information to one another. Integration testing puts together modules that have been unit tested and applies tests defined. There are two approaches for integration testing:

1. **Bottom-up Integration:** It is the traditional strategy used to integrate the components of a software system starting from smallest module/function/programme. It consists of unit testing, followed by sub-system testing. Bottom-up testing is easy to implement as at the time of module testing, tested subordinate modules are available. The disadvantage; however is that testing of major decision/control points is deferred to a later period. For example in above example of internet banking it will test communication between different modules using smallest level of module like saving bank account, fund transfer and then statement of accounts to ensure previous transaction reflects in statement, and so on, however it might not ensure the overall control on passing parameters required for session time out or inactive session

2. **Top-down Integration:** This starts with the main routine followed by the stubs being substituted for the modules which are directly subordinate to the main module. Considering above example, the testing will start from opening login screen and then login, then selecting function one by one. An incomplete portion of a programme code is put under a function (called stub) to allow the function. Here a stub is considered as black box and assumed to perform as expected, which is tested subsequently. Once the main module testing is complete, stubs are substituted with real modules one by one, and these modules are tested. This process continues till the atomic (smallest) modules are reached. Since decision-making processes are likely to occur in the higher levels of programme hierarchy, the top-down strategy emphasizes on major control decision points encountered in the earlier stages of a process and detects any error in these processes. The difficulty arises in the top-down method, because the high-level modules are tested with stubs and not with actual modules.

6.2.3 Regression Testing

It is a testing performed during change management when a function/module/programme is changed or added to existing software, in order to ensure that new/changed functions executes properly and integrates with other modules as expected. It is required since new data flow paths are established, new I/O may occur and new control logic is invoked. These changes may cause problems with functions that previously worked flawlessly. 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.

6.2.4 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 purpose of system testing is to ensure that the new or modified system functions properly. These test procedures are often performed in a non-production test environment. The types of testing that might be carried out with various other objectives described below:

1. **Recovery Testing:** This is the activity of testing 'how well the application is able to recover from crashes, hardware failures and other similar problems'. Recovery testing is the forced failure of the software in a variety of ways to verify that recovery is able to be perform properly, in actual failures
2. **Security Testing:** This is the process to determine that an Information System protects data and maintains functionality as intended. The three basic security concepts that need to be covered by security testing are–confidentiality, integrity, availability. In addition the software may further be tested for user management requirements require i.e. authentication, authorisation, and non-repudiation and log maintenance.
3. **Stress or Volume Testing:** Stress testing is a form of testing that is used to determine the stability of a given system or entity based on the requirements and expected data growth. It involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results. Stress testing may be performed by testing the application with large quantity of data during peak hours to test its performance.

4. **Performance Testing:** Software performance testing is performed on various parameters like response time, speed of processing, effectiveness use of a resources (RAM, CPU etc.), network, etc. This testing technique compares the new system's performance with that of similar systems using available industry benchmarks.

6.2.5 Final Testing

It is conducted if results of system testing are satisfactory and when the system is just ready for implementation. This testing is performed at two levels:

1. At technical level quality assurance testing is performed
2. At functional level user acceptance testing is performed.

Quality Assurance Testing (QAT): QAT focuses on conforming to the quality standards of the organisation accepted before development. It includes documented specifications, technology employed, use of coding standards, and the application meets the documented technical specifications and deliverables. QAT is performed primarily by the technical (IT) department. The participation of the end user is minimal and on request. QAT does not focus on functionality testing.

User Acceptance Testing (UAT): It is a user extensive activity and participation of functional user is a primary requirement for UAT. The objective of UAT is to ensure that the system is production-ready and satisfies all accepted (baselined) requirements. UAT is a formal process and may include:

i. Definition of test strategies and procedures
ii. Design of test cases and scenarios
iii. Execution of the tests
iv. Utilisation of the results to verify system readiness

Acceptance criteria defined along with requirement specifications includes that deliverables must satisfy the predefined needs of the user. A UAT plan must be documented for the final test of the completed system. The tests are written from a user’s perspective and should test the system in a manner as close to production as possible. For example, tests may be based around typical predefined, business process scenarios. If new business processes have been developed to accommodate the new or modified system they should also be tested at this point. A key aspect of testing should also include testers seeking to verify that supporting processes integrate into the application in an acceptable fashion. Successful completion would generally enable a project team to hand over a complete integrated package of application and supporting procedures.

1. **UAT is a stage in SDLC where end users finally accept the developed application system. This is required for all situations of acquiring software i.e. software developed in-house, or by outsourced team or purchased and configured by vendor. A formal sign-off generally marks end of development process.**
2. **UAT should be performed in a secure testing or staging environment where both source and executable code are protected to ensure that unauthorised or last-minute changes are not made to the system unless authorised and the standard change management process is followed. In the absence of controls, the risk of introducing unauthorised changes/malicious patches/Trojan horse programmes is very high.**

3. **Users should develop test cases or use data of live operations of a specified period to confirm whether the processing of data by new application is providing correct results, has required controls and the reports meet the management requirements.**

Many organisations expects report from IS auditor after tests are completed. The IS auditor should issue an opinion to management as to whether the system meets documented business requirements, has incorporated appropriate controls, and may be migrated to production. This report should also identify and explain the risk that the organisation might be exposed by implementing the system.

6.2.6 Other types of Testing

When any complex application/software is intended for general and wide spread use developers want to make sure that product delivers diverse requirements of general users, organisations may consider alpha and beta testing. For example Microsoft performs this type of testing on new product before making it available commercially.

Alpha Testing: This is the first stage, often performed by users within the organisation by the developers, to improve and ensure the quality/functionalities as per users' satisfaction.

Beta Testing: This is the second stage, generally performed after the deployment of the system. It is performed by the external users, during the real life execution of the project. It normally involves sending the product outside the development environment for real world exposure and receives feedback for analysis and modifications, if any.

Automated Testing: In software testing, automation of testing is performed using special software (separate from the software being tested) to control the execution of tests and the comparison of actual outcomes with predicted outcomes. Test automation can automate some repetitive but necessary tasks in a formalised testing process already in place, or add additional testing that would be difficult to perform manually.

Integrated Testing: Some organisations rely on integrated test facilities. Test data usually are processed in production-like systems. This confirms the behaviour of the new application or modules in real-life conditions. These conditions include peak volume and other resource-related constraints. In this environment, IS will perform their tests with a set of fictitious data whereas client representatives use extracts of production data to cover the most possible scenarios as well as some made-up data for scenarios that would not be tested by the production data.

Some organisations use a subset of production data in a test environment, such production data may be altered or scrambled to mask the confidential data. This is often the case where the acceptance testing is done by team members who, under usual circumstances, would not have access to such production data. These tools help in building test cases and also generate test data based on conditions. However, using production data may not help in identifying negative test cases.

Accreditation of software: Although it is not type of testing, many organisations insist on certification and accreditation of software. Generally this is done by the software development houses before taking product to market. In case of tailor-made software certification/accreditation should only be performed once the system is implemented and in operation for some time to produce the evidence needed for certification/accreditation processes. This process includes evaluating programme documentation and testing effectiveness. The process will result in a final decision for deploying the business application system.

Security testing: (Application scans and penetration testing) for information security issues, the evaluation process includes reviewing security plans, the risk assessments results along with response decision, and the evaluation of processes to be deployed. The result of security assessment focuses on measuring effectiveness of the security controls. Security testing provides assurance to the business owner.

Security testing of web application for identified external threats (like SQL injection, cross site scripting etc.) is necessary to ensure that the application can sustain an attack by the hacker who is trying to breach the security.

While reviewing testing process IS auditors focus on getting answers to following questions:

1. **Whether the test-suite prepared by the testers includes the actual business scenarios?**
2. **Whether test data used covers all possible aspects of system?**
3. **Whether CASE tools like 'Test Data Generators' have been used?**
4. **Whether test results have been documented?**
5. **Whether tests have been performed in their correct order?**
6. **Whether modifications needed based on test results have been done?**
7. **Whether modifications made have been properly authorised and documented?**

Testers generally perform black box testing (Penetration test) by trying to simulate attacks on hosted application. This is then followed by performing grey box and/or white box testing that includes code review to identify the issues in coding practices that might introduce the vulnerabilities in the application. These can be avoided by including secure coding practices in coding standard developed by the organisations.

6.3 Implementation

Application software developed shall be implemented once it is tested and UAT has been signed off. However the planning for implementation must start much earlier in SDLC, many times after feasibility study. Planning involves:

- Selecting Implementation Strategies
- Preparing for implementation
  - o Deciding on hardware and ordering (if required) in advance so as to be available in time
  - o Deciding on site where infrastructure to be made available
- Training
- Conversion of data to suit to the requirements of new application.

6.3.1 Implementation Strategies

Considering the nature of business operation appropriate implementation strategy must be decided, much earlier in SDLC. Generally it is decided once the design is finalized or in case of acquisition once the application is selected. Organisation can adopt one of the four strategies, which are described below:

Cut-off or 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. 6.1 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.

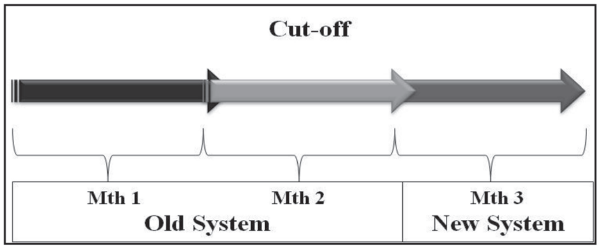


Fig. 6.1: Cut-off or Direct implementation

The challenge in cut-off implementation is that roll-back is most difficult and hence planning must be meticulously done. Also conversion activity must start well in advance and must be properly planned.

Phased Changeover: With this strategy, implementation can be staged with conversion to the new system taking place gradually. This is done based on business operations. For example, converting one function (e.g. marketing) on new system, wait for the same be stabilized and then take another function (Finance/HR/production etc.) 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. 6.2.

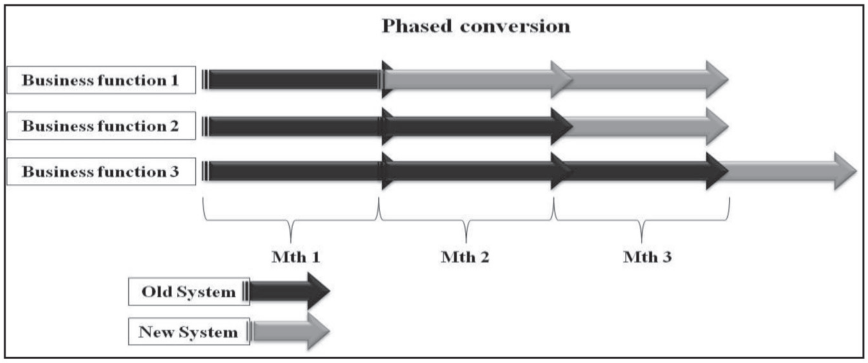


Fig. 6.2: Phased changeover

Phase changeover might require more time to implement however it helps in stabilising one function before starting another.

Pilot Changeover: With this strategy, the new system replaces the old one in one operational area or with smaller scale. Any errors can be rectified and new system is stabilised in pilot area, this stabilised system is replicated in operational areas throughout the whole system. For example converting banking operations to centralized systems are done at one branch and stabilised. The same process is replicated across all branches. Fig. 6.3 depicts Pilot Implementation.

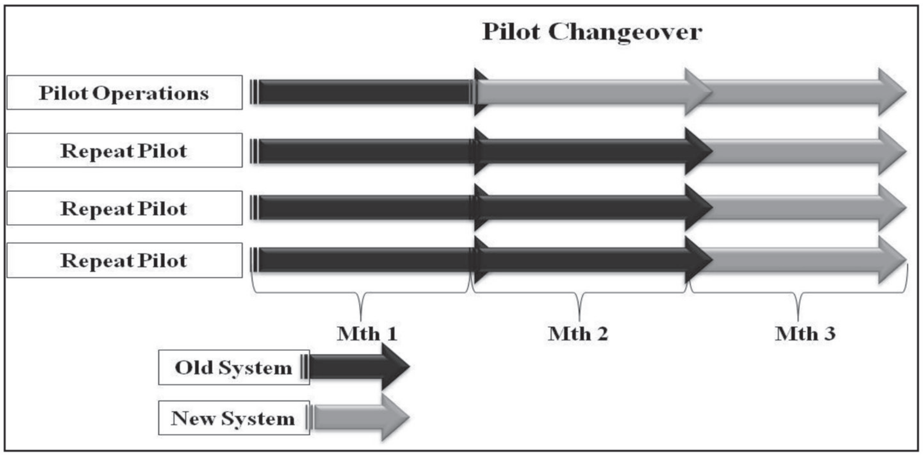


Fig. 6.3: Pilot Changeover

Advantage of pilot implementation is that issues and problems are identified and rectified during pilot run and a stabilised system is implemented thus saving cost and enabling faster implementation.

Parallel Changeover: This is considered the most secure method, time and resource consuming implementation. The new system is implemented, however the old system also continues to be operational. The output of new system is regularly compared with old system. If results matches over period of time and issues observed with new system are taken care of, the old system is discontinued. Fig. 6.4 shows parallel implementation.

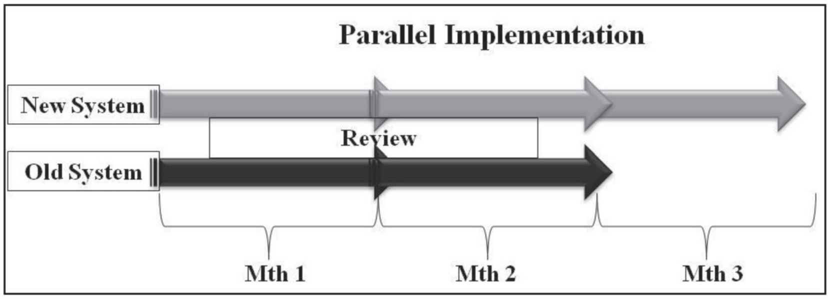


Fig. 6.4: Parallel Changeover

However it is costly and may not be feasible in large and complex systems, since all transactions must be processed twice. Many times users may not be conformable in duplicating the work.

6.3.2 Preparing for implementation

In order to finally deploy or implement the new system in the operating environment, several activities are undertaken. A fully functional as well as documented system is a prerequisite for implementation to begin. Moreover, many other issues like defect removal, maintenances, reengineering may need to be addressed to assure the desirable quality control of the system in operational environment.

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. Implementation includes all those activities that take place to convert from the old system to the new. The new system may be totally new, replacing an existing manual or automatic system. Some of the generic key activities involved in system Implementation are:

- Site preparation and hardware installation
- Conversion of data to the new system files
- Training of end users
- Completion of user documentation
- System changeover
- Post implementation review and evaluation

Site preparation and Installation: The hardware required to support the new system is selected prior to the implementation phase. The necessary hardware should be ordered in time to allow for installation and testing of equipment during the implementation phase. An installation checklist should be developed at this time with operating advice from the vendor and system development team. In situations, where people are not experienced in the installation of similar hardware/ platform/equipment, adequate time should be planned to allow completion of required activities.

Site Preparation: An appropriate location as required to provide an operating environment for system (temperature, humidity and dust control specifications) has to be prepared in time.

Installation of New Hardware/Software: Site preparation also includes receiving, installing and connecting the hardware and supporting software (like operating systems, middleware etc.). In case the hardware is available need to be commissioned for the new application as per design requirements.

Equipment Checkout: The equipment must be turned on for testing under normal operating conditions. Though the routine 'diagnostic tests' should be run by the vendor, the in-house implementation team should test the equipment functionalities in actual working conditions.

6.3.3 Training

Operational training to users about system must be planned along with implementation and before going live and depending upon the complexity of application. There are three ways to provide training:

1. **Simulated training:** Where a training instance of application is installed along with live environment and users are requested to explore the application. This type of training is

generally performed on-site.

2. **Classroom training:** This is controlled simulated training conducted in classroom and is facilitated by trainer.
3. **On the job training:** Conducted in live environment with appropriate guidance from experts.

The quality of training received by the personnel involved with the system in various capacities helps or hinders the successful implementation of information system. When a new system is acquired, which often involves new hardware and software, both users and computer professionals generally need some type of training.

Effectiveness of training can be measured based on changes in number of help desk requests before and after training.

6.3.4 Conversion

Conversion of data is most important activity while implementing new application system or when there is significant change in technology requiring conversion. The conversion activity involves converting data, procedures, documentation from old system to new system. Most important being data conversion.

Data Conversion: The requirement of data conversion depends upon the change. If the new application is replacing manual operation to automated operation it involves:

1. Capturing of data into electronic form
2. Verification of data
3. Uploading into database

In case change is from old system to new system, it involves:

1. Converting electronic data from old format to new format
2. Verification
3. Uploading into new database

Since data conversion is a type of input, controls on conversions are essential to ensure integrity of data. These controls generally include:

1. **Completeness check:** Using number of records, control totals, batch totals, hash totals. For example verifying number of employees record, checking trial balance before and after conversion etc.
2. **Accuracy check:** Manual verification or key verification (manual to electronic conversion)

Unauthorised changes during conversion are one of the sources of frauds.

Procedure Conversion: Changes in application systems may require changes in operating procedures and associated controls. Operating procedures should be carefully completed with sufficient documentation pertaining to operations on how to use the new system. It applies to both computer-operations and functional area operations. Before conversion activities can start, conversion procedures must be defined and personnel involved must be trained to cover input, data files, methods, procedures, output, and internal control.

For example, during manual operation in banking every transaction is verified before being posted to account and then the effect of transaction is reflected in general ledger. However in electronic banking system transactions are flagged with type of transaction and posted to general ledger. Hence verification of transaction is most essential in new system.

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. 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, matching and balancing the total results of both systems.

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 familiar with the new system, the job becomes easier to perform and becomes part of the routine work. Schedules should be set up by the system manager in conjunction with departmental managers of operational units serviced by the equipment. The master schedule for next period/ month should provide sufficient computer time to handle all required processing.

6.4 Change management process

Application maintenance refers to the process of managing changes in the application and IT triggered or prompted due to changes in processes, regulatory compliances, and strategic changes in business, technology changes and so on. Changes also arise due to issues, problems, incidents faced. In order to handle changes organisation should have defined process. This process generally includes:

1. **Raising change request:** Formal process for requesting change. Anyone can raise the change request with reason for the change, a cost justification analysis, if possible and the expected benefits of the change. An automated process for raising change requests helps in capturing all associated changes and maintaining record of change requests.
2. **Defining requirements:** Defining details of changes required, like functional changes, appearance changes, processing changes. (e.g. change in tax structure may require processing changes, if tax slabs are displayed then appearance changes and so on).
3. **Analysing requirements:** Getting answers to the questions such as: why change is required, when it should be effective, who needs it, where the changes are required, what programmes/modules/function is affected, how the changes will be carried out and so on.
4. **Impact analysis:** What will be the impact of changes on processes and other related programmes that interface with application that need to be changed or how changes in technology shall affect the processing.
5. **Approval of change:** Changes must be approved by the asset owners, i.e. application owners in case of application change and other stakeholders that might be impacted. Sometimes it is difficult to decide who has appropriate authority to approve change due to impact on multiple processes. To overcome such situations organisations form a change approval board or committee (CAB) consisting of representatives from multiple business functions.

6. **Prioritising the change requests:** This is required to resolve the conflict due to multiple change requests from different users.
7. **Carrying out changes:** System analyst shall review the changes and decide appropriate resources to carry out changes. Records of all programme changes should be maintained. Library management software may help in automating this process and also maintaining audit trail. The maintenance information usually consists of the programmer ID, time and date of change, project or request number associated with the change, and before and after images of the lines of code that were changed. This also helps in preventing and/or detecting unauthorised changes.
8. **System document maintenance:** All relevant system documentation updating sometimes is neglected area during change management. It is essential to ensure the effective utilization and future maintenance of a system, Documentation requiring revision may consist of programme and/or system flowcharts, programme narratives, data dictionaries, entity relationship models, data flow diagrams (DFDs), operator run books and end-user procedural manuals. In case of infrastructure changes, network diagrams. Data centre block diagrams, electrical and facility diagrams etc. are likely to undergo changes.
9. **Testing the changes:** Changes will be tested as per testing process (Please refer sub-section on testing). However for testing changes, following points must be considered:
   - o Existing functionalities are not affected by the change
   - o System performance is as expected
   - o Security vulnerabilities are not introduced

   This also includes conducting user acceptance testing and formal sign-off from users/ owners. (E-mail, electronic approval in automated system, document etc.)
10. **Releasing changes:** Changes shall be released to production once approved by stakeholders (UAT). Ensure fall-back procedures in place in case operations are affected due to change. Automation of this process shall help management in restricting one person requiring access to production, test and development environment.
11. **Review:** Post implementation/release review may be conducted.
12. **Record maintenance:** Change requests should be maintained in a format that will ensures that all changes associated with primary change requests are considered. This allows the management to easily track the changes to change requests. The process must be formal and maintain record of all approvals and rejections.

For acquired systems vendor may distribute periodic updates, patches or new version of the software. User and systems management should review such changes for impact and appropriateness before implementing.

While reviewing change management IS auditor should ensure that:

- **Access to source programme is restricted.**
- **Change requests are approved and record is maintained to trace and track the changes.**
- **Impact assessment is performed before approving changes.**

- **The change request should be documented in standard format covering at the minimum:**
  - o **Change specifications, benefit analysis developed and a target date.**
  - o **Change form has been reviewed and impact assessment is recorded.**
  - o **change request has been approved formally**
- **Verify records of changes for sample changes made and trace end-to-end (from request till closure) confirm that the changes are authorised, approved, and moved to production after UAT.**

Ensure segregation of duties during change management

6.4.1 Emergency Changes

In exceptional situations there may be need to make changes to production to resolve issues in time. This requirement can arise due to one or more reasons like:

- Events/incidents
- Short notice requirement changes (due to external incidents/events: terrorist attacks, natural disasters, etc.)
- Infrastructure failure
- Production issues due to unexpected data conditions

Procedures should focus to ensure that emergency changes can be performed without compromising the integrity of the system. Organisation should have a process for carrying out emergency changes. The process may consist of following steps:

1. Identify need for emergency change (process issue, incident/event etc.)
2. Determine activities involved. Generally it may involve providing all accesses to one person. A special user ID may be created with higher privileges for this purpose and all activities are logged and reviewed.
3. A post-facto change management process must be followed, so as to ensure consistency in documents, source-code library, network diagram etc. as applicable.

The IS auditor has to ensure that emergency changes are handled in a controlled manner.

6.4.2 Implementing Changes into Production

Changes are implemented into production environment once they are approved by the user management (UAT sign-off). The best practices suggest that this implementation should be done by independent team not involved in development or testing of changes. In case of client-server applications or distributed systems, such as point-of-sale systems, the process should be properly documented and implemented over a period of time to ensure:

- Conversion of data
- Training of users
- Support process for changes

- Rollback plan
- All points are updated

6.4.3 Segregation of Duties

Uncontrolled change management has risk associated with unauthorised changes. An unauthorised change occurs due to various reasons:

- Developer has access to production libraries containing programmes and data including object code
- User has not approved change or not aware of the change
- A change procedure has not formally established.
- A change was updated into production without user approval.
- The changes had not been reviewed or tested.
- Developer inserted extra logic for personal benefit (i.e., committed fraud).
- In case of vendor software, changes received were not tested.

In order to control unauthorised changes segregation of duties has to be implemented at organisation level. The typical segregation includes following controls at the minimum:

- Development, test and production environments to be physically separated.
- Developers' team, testing team and production user should not have access to other areas. I.e. developers should not have access to test and production and so on.
- Source code must be maintained by librarian. At least control must be in place to prevent or detect insertion of unauthorised code.
- A separate change control team or release team should be appointed to move source/object code from development to test and from test to production.

It may not be possible for some organisations to implement strict segregation of duties, in such situation appropriate compensating controls should be present to prevent or detect and correct unauthorised changes. Some such situations may arise due to:

1. The developer is also the operator due to small IT department. In this case user management required to ensure proper authorisation and monitoring of changes and upgrades made by the programmer.
2. Emergency changes to resolve the issues in production.
3. In case separate release team is not possible, compensating control of enabling user ID of user who moves changes from development to test and/or test to production only after approval and monitoring activities may work as compensating control.
4. Developers should not have write, modify or delete access to production data. Depending on the type of information in production, programmers may not have read-only access to personally identifiable information.

6.4.4 Configuration Management

Configuration management is refers to automated process organisations installed to maintain all information assets and work-flows required to maintain them. The backend of such system is a data base called a Configuration Management Data Base (CMDB); hence sometimes the system is also referred to as CMDB.

Configuration management system helps in maintaining information about system as collection of configuration items (CI). A CI can be a module/function/programme or database instance of IT asset associated with a system and referenced with an ID. Workflows around the CMDB consist of workflows for Change Management, configuration management etc. Change management requests must be formally documented and approved by a change control group within CMDB. CMDB then manages the change process via checkpoints, reviews and sign-off procedures that generates audit trails.

Configuration management sometimes may provide procedures throughout the software life cycle (from requirements analysis to maintenance) to identify, define and baseline software items in the system and thus provide a basis for problem management, change management and release management. However though it sounds easy, proper implementation of CMDB is a necessary requirement (which must follow SDLC process for acquired Software).

Software configuration management requires following tasks to be performed:

1. Develop configuration management plan
2. Baseline application and associated assets
3. Analyse results of configuration control
4. Develop monitoring of configuration status
5. Develop release procedures
6. Define and implement configuration control activities (such as identification and recording of change requests.)
7. Update the configuration status accounting database

A configuration management tool supports change and release management by supporting following activities:

1. Identification of items affected by a proposed change
2. Help in impact assessment by providing information
3. Recording configuration items affected by changes
4. Implementation of changes as per authorisation
5. Registering of configuration item changes when authorised changes and releases are implemented
6. Recording of original configuration to enable rollback if an implemented change fails
7. Preparing a release to avoid human errors and resource costs

6.5 Summary

Implementation of new application follows a similar process required for change and release management. Although change management primarily refers to the maintenance activity in SDLC phases, a large change may have to be initiated to implement change by implementing a SDLC project. In other words a change management process by itself may be considered as a mini-SDLC project. Controls required during implementation and change management are similar and IS auditor has to understand the process of controlling unauthorised changes in software whether it is being developed or configured or maintained.

6.6 Questions

1. Which of the following process during implementation of banking software is most critical to minimise the risk associated with frauds?
   A. Training and awareness
   B. Data conversion
   C. Hardware configuration
   D. Procedure conversion
2. Which of the following is a major concern for IS auditor while auditing implementation phase of SDLC project?
   A. Requirement of resilient infrastructure are captured during implementation
   B. Hardware acquisition has been delayed due to delay in procurement process
   C. Organisation has contracted multiple network service providers for connectivity
   D. Organisation has decided to use existing site for implementing new solution
3. Who among the following should be primarily responsible for approving changes to application system?
   A. Head of IT department
   B. Business process users
   C. Application administrator
   D. Affected business stakeholders
4. Which of the following method is most useful in implementing an ERP application at multiple locations of same organisation?
   A. Parallel
   B. Pilot
   C. Phased
   D. Cut off

5. An organisation has developed a web based application for the use of internal users to be hosted on intranet. Before finalising and making it live it was decided to make it available to users for providing feedback. This is an example of:
   - A. Internal audit
   - B. Alfa testing
   - C. Beta testing
   - D. User training
6. A major concern associated with using sanitized old production data for testing new application is that:
   - A. User may not provide sign off
   - B. Production data may be leaked
   - C. Integration testing cannot be performed
   - D. All conditions cannot be tested

6.7 Answers and Explanations

1. B. Fraudster manipulated the data being entered into system to commit fraud by exploiting vulnerabilities in the process. Hence, data conversion process must be monitored more closely as compared to other processes.
2. A. Requirement of resilient infrastructure must have been considered during requirement phase, since it affects software development as well as procuring redundant infrastructure. Capturing this requirement during implementation phase will have major impact on benefit realisation. Other options are not major concerns, but in fact C is an advantage.
3. D. Changes to application must be primarily approved by application owner, who is also business process owner. However change in application may also require change in dependent systems hence it is best to get approval from all. To facilitate this process generally change approval board is formed that represents all stake holders.
4. B. Pilot method is most useful, as organisation can implement application at one location and run for few days. This will help in finalising the basic configuration and also learn from post implementation issues. Once successful the same can be replicated at other locations.
5. C. Beta testing is making product available for user for feedback before launching.
6. D. Production data generally may not cover all paths the data can take and hence system cannot be tested for all possible cases. Data leakage is not a major concern since data is sanitised. Options A and C are not concerns.

CHAPTER 7: TRENDS IN TECHNOLOGY IMPACTING SDLC

Learning objective

This chapter provides information on impact of emerging trends on SDLC process. The emerging trends discussed are virtualisation, cloud computing, mobile computing, Big data and data analytics. These trends are impacting the decision on technology requirements.

7.1 Introduction

Changes in technology have direct and/or indirect impact on the SDLC phases. With various options available, organisations need to consider use of new technology for effective service delivery and IT must support this service delivery without affecting the security of information assets. In Module 1 we have seen various emerging technologies affecting use of information technology. In this chapter we will consider how some new technologies have impact on SDLC.

We will focus on following technologies:

- Virtualisation and cloud computing
- Mobile technology
- Big data and data analytics

7.2 Technology Trends

7.2.1 Virtualisation

Virtualisation refers to the creation of a virtual machine that acts like a real computer with an operating system. The software that helps in creating virtual machine on the host hardware is called a *hypervisor* or *Virtual Machine Manager or in short VMware.* In other words one physical server may host more than one virtual server that supports application. Instead of relying on the old model of "one server, one application" that leads to underutilisation of resource, virtual resources are dynamically applied to meet business needs.

Characteristics of virtualisations affecting SDLC

1. **Backup:** It is easier to backup virtual server. When the virtual server boots, it checks the configuration file and then reads from the simulated disk drive, which contains a simulated file-system holding server's operating system software and applications. A backup copy can be created any time when server is not running, by copying disk files, and configuration file. If the virtual server can be installed on another environment by making necessary changes to configuration file. (i.e. IP and MAC address). The development team needs to define backup procedures according to target environment.
2. **Evaluation of alternates:** Many large projects require evaluation of tools and software. Particularly if alternate solutions are being considered before finalising suitable solution.

Evaluation can be for acquisition of software from different vendors, reusable components or for prototype selection. One can build a series of virtual servers with different solutions or products. It is then easier to compare the requirements against predefined parameters.

3. **Virtualised target for developers:** Developers sometimes need to test their code before releasing to testing team; however the developers might be using desktop or laptop for development and might be different from target environment. Using virtualisation required target environment may be simulated on developers desktop effectively reducing errors found during final testing or operations.

7.2.2 Cloud computing and sourcing options

The "cloud" in cloud computing is defined as the set of hardware, networks, storage, services, and interfaces that combine to deliver aspects of computing as a service. Webmail service providers like Google, Hotmail, Yahoo etc. depends on cloud technology. Many of us who are using e-mail account with a web-based e-mail service provider like Gmail, Hotmail, Yahoo, etc. are already be cloud service users even without knowing it. You will notice that you are using a computer of any type which may be located anywhere, anytime to access your e-mails. The email programme is running on the server and all your e-mails including the software is stored on the cloud.

In cloud computing, services are provided over the Internet based technology by dynamically scalable and often virtualised resources. The impact of using 'cloud' technology for service delivery affects SDLC process, depending upon which and how the services are being used. The lists of requirements that must be considered are discussed below:

- Application on cloud uses platform independent web-based technology like Java, Net, XML, PHP etc. Deployment of services may happen in phased manner, the project manager may consider agile development method to develop and deploy services.
- Client is executed using internet browsers like internet explorer, Google chrome, Mozilla etc. and hence need to be tested for all known browsers. It is necessary to consider security while developing the software, users may or may not use security settings in their browsers. Also not all browsers offer same level of security settings.
- Web application security requirements need to be considered while designing and testing the application.
- Non-functional requirements of performance and response have to be considered while developing the software.
- Licensing issues for utilities and middleware are complex and should be considered. The challenge of licensing has two aspects:
  - o Who is responsible for procuring licences? In case organisation uses cloud services offered by service provider for infrastructure only (IaaS), procuring required licences is not the responsibility of service provider. However in case of PaaS and SaaS, service provider is responsible for procuring licences.
  - o Licensing policy of software owner is different and is governed by the contract between provider and purchaser. It is best to involve the software vendor in decision while procuring cloud services.

- In case of procuring SaaS, the software is provided by service provider to multiple clients. Procuring organisation is accountable for security of information processed and stored on cloud and hence the contract and monitoring of terms of contract need to be considered during acquisition phase of SDLC.
- Service provider while offering services must ensure that client organisations are complying with required legal and regulatory requirements and issues related to cross-border compliances are addressed in contract.

Auditing Cloud services

While performing IS audit where auditee has hired third party cloud service provider to host the services, auditor need to evaluate following aspects:

- Business case for using cloud services
- Third party selection process
- Provisions of service level agreement if it covers concerns of business
- Service monitoring mechanism that provides assurance to management
- Mechanism for assurance by service provider on controlling of risks associated
- Third-party independent audit of service provider

Auditor may consider following questions:

- How much security is enough?
- What is the Criticality of the application being hosted on cloud?
- What is Outsourcer's experience with SLA and vendor management?
- What are Country/regional regulations (for example SOX and Europe's data privacy laws), and Industry Regulations (for example GLBA and HIPAA)?
- Does your present security model need to be altered?
- What is the Cloud vendor's policy on vulnerability management - reporting (beyond basic 'Contact Us' links), commitment to following up, promptly responding to reports?
- Is there an independent auditor's report?
- What is the impact on the auditor when client has used cloud computing and the data to be audited is with the cloud service provider?
- What is the impact on compliances?
- What is the impact on security?
- Who owns the data?
- Where does the data reside?
- Who is responsible for compliance and what is the impact on compliance?
- What business continuity and disaster recovery measures are in place in the cloud infrastructure? Does the cloud provider have a backup in place?
- What is the potential implication of employees wanting to sabotage a successful cloud migration strategy?
- What are fall back arrangements on termination of contract?

7.2.3 Mobile Computing

Mobile computing is the name for using of portable devices with computing power connected to the network (mostly wireless), such as PDA, laptops, mobile phones, MP3 players, digital cameras, tablet PC and Palmtops etc. Mobile technology enables organisations to scale up process efficiency by offering access to data anytime, anywhere, anyhow based on access rules defined, on an online and real-time basis thus increasing productivity, customer service, employee satisfaction, etc. Use of mobile devices has inherent risks and these needs to be mitigated by implementing appropriate controls.

Some of the business risks are given below:

- Mobile devices may contain an organisation's sensitive information, like data obtained from business/customer applications, customer private information, corporate e-mails, and documents. The mobile computing devices may be used to provide input to business applications on wireless networks which can impact organisation data on a real-time basis. Loss or theft of organisation information/asset due to virus attacks or malware.
- Exposure due to Information interception through wireless sniffers/intrusion resulting in a loss or breach of sensitive data, privacy impacting organisation reputation and legal implications.
- Propagation of malware resulting in data leakage, data corruption and non-availability of required data.
- Physical damage to devices, data corruption, data leakage, interception of calls and possible exposure of sensitive information.
- Possibility of fraud through remote access and inability to prevent/detect it.
- Copying of critical organisation information by hackers using remote access.
- Lost devices or unauthorised access to unsecured devices allowing exposure of sensitive data, resulting in loss to the organisation, customers or employees

Organisation need to mitigate these risks by using standard risk mitigation options like identifying high-value data elements and providing security, implementing mobile computing security policy, implement specific controls for mobile devices like standard configuration of mobile devices, anti-virus software, user training, encryption etc.

Mobile devices and applications development

Many organisations provide services based on mobile technology. These services use special application developed to suit to mobile based delivery. New generation smart phones come with separate operating systems (e.g. android, iPod, Microsoft etc.) that run on these phones. Application need to be developed that can run on these technologies. Many such applications are available for free download and users may not be aware of existence of malware. Organisations that wish to provide mobile based services may initiate projects for developing applications for mobile delivery for existing services. (E.g. Mobile Banking).

This may be essential to ensure capturing of appropriate requirements and that security is built in the application. Organisation may not want to rely on security setting to be done by mobile users. In case new application is being developed which the organisation intends to deploy on mobile devices, SDLC process must capture these requirements before considering feasibility.

7.2.4 Bring Your Own Device (BYOD)

BYOD refers to the policy of permitting employees to bring personally owned mobile devices (laptops, tablets, and smart phones) to their workplace, and to use those devices to access privileged company information and applications.

Benefits of BYOD

- **Increased productivity:** Increased productivity comes from a user being more comfortable with their personal device, being an expert user makes navigating the device easier, increasing productivity
- **Employee satisfaction:** BYOD allows user to use the device they have selected as their own rather than one selected by the IT team. It also allows them to carry one device as opposed to one for work and one for personal. Also employee may wish to have devices that are more cutting edge and organisations may not replace these devices as often.
- **Cost savings for the company:** Cost savings can occur on the organisation need not provide mobile devices to employees.

Risks of BYOD

Risks of BYOD are same as risk of mobile computing; however it has higher risk associated with data breach. For example, if an employee uses a smart phone to access the company network and then loses that phone, untrusted parties could retrieve any unsecured data on the phone. Another type of security breach occurs when an employee leaves the company, they do not have to give back the device, so applications and other data may still be present on their device.

A key issue of BYOD which is often overlooked is BYOD's phone number problem, which raises the question of the ownership of the phone number. The issue becomes apparent when employees in sales or other customer-facing roles leave the organisation and take their phone number Customers calling the number will then potentially be calling competitors which can lead to loss of business.

To mitigate the risks organisation must have a BYOD policy that defines:

- Ownership of information on mobile device
- Which sensitive company information needs to be protected and how
- Education of employees on mobile security.

7.2.5 Big Data and Data analytics

Big Data

Big data is a term for the collection of large and complex data that cannot be processed using normal database management tools or traditional data processing applications. The challenges include capture, duration, storage, search, sharing, transfer, analysis, and visualization. Big data requires the use of new or exotic technologies to manage the volume of data. Technologies such as in-line de-duplication, automated tiering of data to get the most efficient usage patterns per kilowatt, and flash or solid-state drives for higher-end performance optimisation are expected to increase in importance over the next few years.

The data in big data comes from various sources. It is either historical data or the data collected over period of time that requires analysis or both. For example demographics of population with various attributes like gender, religion, ethnicity, age group, life expectancy, diseases history, birth rate, death rate and so many such attributes and parameter might run in terabytes. There can be another data set capturing consumptions and availability of essential elements like food, liquids, from various countries.

Managing Big data requires special framework technologies like NoSQL databases, Hadoop, Map Reduce. These technologies form the core of an open source software framework that supports the processing of large data sets across clustered systems. Primary challenge with big data is design architecture to store the data without affecting integrity (accuracy and completeness) of the data. This is important because most of the times it is stored in denormalised form.

Another challenge is maintaining confidentiality and privacy of the data stored. This is generally controlled by providing access to limited personnel based on need to do basis.

Some examples of big data projects are:

- Turning 12 terabytes of Tweets into improved product sentiment analysis
- Scrutinizing 5 million trade events created each day to identify potential fraud
- Monitoring 100's of live video feeds from surveillance cameras to identify security threats

Risks

Organisations must come to terms with the security challenges they introduce, for example:

- Big data requires data to be stored in denormalised form i.e. schema-less in distributed environments, where data from multiple sources can be joined and aggregated in arbitrary ways, make it challenging to establish access controls
- As the big data consist of high volume, high variety and high velocity data, it makes difficult to ensure data integrity
- Since it is aggregation of data from across the organisation, it also includes sensitive data
- Most existing data security and compliance approaches will not scale to handle big data security

Risk mitigation

Following controls need to be implemented to minimise the associated risks:

- **Sensitive data identification and classification:** Identification of sensitive data and their relationships with other data sets, so that the right security policies can be implemented.
- **Data access and change controls:** Defining and implementing policies regarding which users and applications can access or change data.
- **Real-time data activity monitoring and auditing:** Enabling logs for sensitive data access with time stamp for monitoring.
- **Data protection:** Consider encrypting confidential data (e.g. credit card details).
- **Vulnerability management:** Identifying weaknesses and initiating remedial action.

Audit questions

Some questions that helps auditor in understanding control implementation.

- Who are running specific big data requests?
- Are users authorised to make requests?
- What jobs users are running?
- Are users trying to download sensitive data or is the request part of a job requirement, for example, a marketing query and credit card details?

The rush for big data benefits is not an excuse for overlooking security.

Data analytics

Organisations use data analytics for various analyses that help management in arriving at right decision. These technologies are useful for analysis of big data, new analytic applications and pattern-based strategies. Organisations will focus on harnessing the power of information by using business intelligence and data analytics tools to monitor and improve performance and costs. IT may be required to focus on developing analytics that enable and track collaborative decision making.

Analytics mean the use of analysis, data, and systematic reasoning to make decisions. What kind of analysis? What kind of data? What kind of reasoning? There are no hard-and-fast answers. Any analytical process can provide a serious, systematic analysis. The most common is statistical analysis, in which data are used to make inferences about a population from a sample. Variations of statistical analysis can be used for a huge variety of decisions from knowing reasons of something that happened in the past, to predicting what may happen in the future. Statistical analysis can be powerful, but it’s often complex, and sometimes employs untenable assumptions about the data and the business environment.

Benefits of data analytics

- Help manage and steer the business. Analytics give managers tools to understand the dynamics of business, including how economic and marketplace shifts influence business performance.
- Know what’s really working. Rigorous analytical testing can establish whether management intervention is causing desired changes, or whether it’s simply the result of random statistical fluctuations.
- Leverage previous investments in IT and information to get more insight, faster execution, and more business value in many business processes.
- Cut costs and improve efficiency. Optimisation techniques can minimize asset requirements, and predictive models can anticipate market shifts and enable companies to move quickly to slash costs and eliminate waste.
- Manage risk. Greater regulatory oversight will require more precise metrics and risk management models.
- Anticipate changes in market conditions. You can detect patterns in the vast amount of customer and market data coming your way.
- Have a basis for improving decisions over time. If you are using clear logic and explicit supporting data to make a decision, you or someone else can examine the decision process more easily and try to improve it.

Risks

Most risks of Big data are also similar to risks of data analytics. However, there are additional risks such as:

- **Logic Errors:** These are related to not being able to collect and analyse appropriate data, or making incorrect assumption.
- **Process Errors:** These are associated with processing incorrect data, missing alternate analysis, incorrect treatment to data resulting in wrong decision.
- **Access Control Errors:** Data analytics and business intelligence tools help in developing analysis of unstructured data and deploy parameterised report generation for analysis of data. Care must be taken while providing access to report generation modules that the person is eligible to access the data. In absence of such control data breach may occur.

7.3 Summary

Changes in technology are definitely affecting the business environment. However every new technology has its own payload of risks associated. Organisations must ensure that appropriate risk management process is established while considering new technologies while developing solutions for service delivery.

When faced with the paradigm change and nature of services provided through new technologies, IS auditors should consider following key assurance areas:

- **Transparency:** Technology must demonstrate the existence of effective and robust security controls, assuring customers that their information is properly secured against unauthorised access, change and destruction. Is segregation of duties maintained? How are different customers' information segregated? What controls are in place to prevent, detect and react to breaches?
- **Privacy:** With privacy concerns growing across the globe it will be imperative to prove to existing and prospective customers that privacy controls are in place and demonstrate their ability to prevent, detect and react to breaches in a timely manner.
- **Compliance:** Most organisations today must comply with various laws, regulations and standards. New technology must ensure that information must be provided to authorities without compromising other information.
- **Trans-border information flow:** When information can be stored anywhere in the cloud, the physical location of the information can become an issue. Physical location dictates jurisdiction and legal obligation.
- The use of standards and frameworks will help businesses gain assurance around. At the time of writing, there are no publicly available standards for new technologies, however, existing standards should be consulted to address the relevant areas and businesses should look to adjust their existing control frameworks.

7.4 Questions

1. Organisations consider virtualisation primarily to:
   - A. Optimize resource utilisation
   - B. Eliminate hardware cost
   - C. Implement cloud application
   - D. Reduce licence requirements
2. Organisation is considering changing the hosting technology for application accessed by internal and external users. Which of the following options has highest security concerns?
   - A. Client-server technology by providing customised client to users
   - B. Private cloud hosted within organisation and accessed using web
   - C. Outsourced services provided by third party on public cloud
   - D. Establish own fibre optics network using service provider
3. An organisation hired third party service provider to provide Software as a service to internal users. Which of the following document IS auditor should review first?
   - A. Service level agreement signed with service provider
   - B. Business case document approved by the steering committee
   - C. Comparative analysis of requirements for different vendors
   - D. Feasibility study report containing outsourcing recommendations
4. Primary benefit of developing and providing application using mobile technology to customers of organisation is to:
   - A. Enhance security of data
   - B. Increase usage of technology
   - C. Stay ahead in competition
   - D. Enhance service delivery
5. Which of the following the organisation should consider first while allowing users to access organisation's data from their own device?
   - A. Formulating a policy for use of personal devices
   - B. Selection of encryption appropriate for all devices
   - C. Savings in cost required for acquiring devices
   - D. Define uniform configuration for personal device
6. Which of the following controls are most important for Big data application?
   - A. Data encryption
   - B. Data classification
   - C. Data access
   - D. Data analysis

7.5 Answers and Explanations

1. A. Virtualsation primarily helps in optimising resources by allowing consolidating hardware and storage for application running on diverse operating systems. Eventually it reduces costs, not eliminate, but that may not be primary objectives. Cloud technology used virtualisation to optimise resources. Virtualisation may not affect licence requirements.
2. C. Public cloud services provided by third party as highest security concerns.
3. B. Business case is the first document IS auditor should review to understand the reasons behind the decision. Business case generally covers feasibility study and requirement analysis. SLA and vendor selection can be reviewed later.
4. D. Using mobile application helps organisation to add service delivery channel that enables customers to get information anywhere. It may not enhance security, usage of technology may not be concern for organisation and although it may help in match the competition it may not help in leading in market.
5. A. Organisations considering BYOD, must first formulate policy for usage of such devices so as to protect the information of organisation. Other aspects may be considered based on policy.
6. B. Although most important control for Big data application are access controls since data is stored in denormalised form, access controls need to be implemented at data element level. This can be achieved by data classification. Also encryption controls can be considered for most sensitive data based on classification so that data analysis controls can be implemented effectively.

CHAPTER 8: SDLC REVIEWS AND AUDIT

Learning objectives

This chapter focuses on the process and method required to be followed by IS auditor while auditing the SDLC projects. It covers role of IS auditor as team member of SDLC project where auditor is consultant for including controls. It also covers mid project reviews and post implementation audit of SDLC project.

8.1 Introduction

Every organisation using IT has to adopt a SDLC process to develop and implement the software or to acquire, customise and implement the software. IS auditor has to understand the process each organisation follows and perform the SDLC process audits. The scope of audit might be focused on SDLC or it might be part of larger scope where SDLC processes need to be reviewed.

The IS auditors role in the software acquisition process is to determine whether adequate level of security controls has been considered. This is required to ensure data integrity of the information processed and controls like audit trails, password controls and overall security of the application. The above procedures should be more elaborate and systematic in case where the business is implementing ERP systems giving fully integrated corporate solutions like SAP, Oracle Financials, BAAN, PeopleSoft, JD Edwards etc.

8.2 Role of IS Auditor in SDLC

The IS auditor may be involved in SDLC process:

1. As a project team member as consultant to suggest controls
2. As IS auditor to review the SDLC project
3. As IS auditor to perform post implementation review

8.2.1 IS Auditor as Team member

The IS auditor can be associated with SDLC project as team member to perform following activities:

1. As a reviewer for information security requirements to be included in proposed system
2. As a consultant to suggest appropriate controls to be included in proposed solution as well as ensuring security during execution of project.

Generally a question arises about the independence when the auditor is required to perform independent review of SDLC project either mid-project review or post-implementation review. Ideally auditor should avoid performing review in such situations, however, in extreme cases auditor may perform reviews if the auditor’s role is that of consultant of controls and not involved in deciding the controls to be implemented. If auditor has decided and designed the controls it will not be appropriate for the auditor to perform reviews.

8.2.2 Mid-project reviews

The audit of SDLC project can have the following three primary objectives:

1. To provide an opinion on the efficiency, effectiveness, and economy of project management
2. To assess the extent to which proposed application provides for adequate audit trails to evaluate the controls to ensure the integrity of data processed and stored
3. To ascertain the effectiveness of controls embedded within application system.

In order to achieve these goals and provide assurance to management on progress of SDLC project, an IS auditor has to be part of project team to review the progress. IS Auditor typically performs following activities:

- Attend project and steering committee meetings to understand the status of project
- Examine project control documentation to assess possible risks and mitigation plan to ensure expected deliverables
- Interview project team and stakeholders to understand expectations and assess the expected deliverables.
- Ensure what project control standards are to be complied with, (such as a formal systems development process) and determining the extent to which compliance is being achieved.
- Examine system documentation such as functional specifications to arrive at an opinion on controls. The auditor's opinion will be based on the degree to which the system satisfies the general control objectives that any information system should meet. A list of such objectives should be provided to the auditor. The auditor should provide a list of the standard controls, over such operational concerns as response time, CPU usage, and random access space availability that the auditor has used as assessment criteria.
- An Auditor may adapt a rating system (for example a scale of 1 to 10, 10 being best) in order to give rating to the various phases of SDLC. For example while rating a Feasibility Study, an auditor can review Feasibility Study Report, interview the personnel, who have conducted feasibility study, understand deliverables and then depending on the content and quality of the Feasibility Study report and interviews, an auditor can arrive at a rating. (Say 7 if scale 1 to 10 is used). After deriving such a rating for all the phases, the auditor can form his/her overall opinion about the SDLC phases.

In order to audit technical work products (such as database design or physical design), auditor may seek opinion of technical experts. Some of the control considerations for an auditor include the following:

- Documented policy and procedures
- Established Project team with all infrastructure and facilities
- Developers/IT managers are trained on the procedures
- Appropriate approvals are being taken at identified mile-stones
- Development is carried over as per standards, functional specifications
- Separate test environment for development/ test/ production/test plans
- Design norms and naming conventions are as per standards and are adhered to

- Business owners testing and approval before system going live
- Version control on programmes
- Source Code is properly secured
- Adequate audit trails are provided in system; and
- Appropriateness of methodologies selected.

While auditing project management, auditor should consider following aspects:

- Project charter
- Roles and responsibilities of different groups/committees (e.g. Project steering committee)
- Adopted project management methodology
- Application Development Methodology/model
- Contractual terms with the vendors for purchased applications (E.g. SLAs)
- Contractual terms with the vendors for outsourced services (E.g. SLAs)
- Approvals and sign-offs by the Project steering committee for each SDLC stages
- Deliverables of each SDLC stages
- Minutes of relevant meetings
- Project tracking and reporting documentation
- Resource management
- Ongoing risk management
- Quality Control/Assurance
- Change management
- Data conversion/migration
- Application testing documentation
- Relevant legal, regulatory and policy aspects to be complied with, if any

8.2.3 Post implementation review

Post Implementation Review of SDLC project focuses on finding the answer for the question: "Did the organisation achieve what was required?" It also tries to find out the degree of success from the project, to the extent to which it has met its objectives. A Post-Implementation review should be scheduled some time after the solution has been deployed. Typical periods range from 6 months to 18 months, depending on the type of solution and its environment. The delay is basically to provide sufficient time to be able to measure the benefits from new solution and be able to compare them with business case.

There are two aspects of evaluation:

1. The system is operating as expected without operational issues
2. The user is satisfied with service delivered by the information system

Typical evaluation covers following aspects:

Development process: Evaluation of the development/acquisition and implementation process is primarily focuses on project execution and ensuring the system was developed on schedule, within budget and meets required quality. The review covers schedules and budgets established in advance against the records of actual performance and cost. However, IS auditor may or may not comment on delay in delivery or budget overrun depending upon the importance of project to the organisation presented in business case.

Operations: The review of issues identified post-implementation and their resolution. Also the benefits of resource utilization as predicted in business case compared against current utilisation. For example if the application is expected to support increasing number of concurrent users, are there any operational issues currently observed at lower scale of operations? Is there simulation testing performed to ascertain expected peak performance?

Information Security: System should also need to be evaluated for information security and privacy controls. This aspect of system evaluation is based on the security requirements documented during information gathering, security of infrastructure on which the application is hosted (e.g. hardware baselining, network security, access controls and vulnerability scanning). Evaluation may also include the availability aspect required for continuity (e.g. in case of high availability requirements redundant infrastructure in cluster or replication and readiness of alternate site, updating of BCP documents etc.)

Conversion process: During implementation it is possible that the data converted and migrated to new system might contain obvious errors resulting in erroneous processing or frauds. The conversion processes must be evaluated against the record maintained to ensure the accuracy, completeness and security of data.

IS Auditors should be able to determine if the expected benefits of the new system are realised and whether users are satisfied with the new system. IS Auditors may also review which of the SDLC phases have not met desired objectives and whether any corrective actions were taken.

If there are differences between expectations and actual results, auditors need to determine the reasons for the same. Such discrepancies may be due to incomplete user requirements or any other shortcomings.

8.3 Summary

Successful execution of SDLC project must be reviewed regularly as part of continuous improvement of organisation's project management practices. The IS auditor can play an important role in the process by providing inputs based on review of existing projects. IS Auditor in this case is performing the role of consultant rather than an auditor, as the objective is to implement the learning from review process.

8.3.1 Key factors to be considered by IS Auditor in SDLC process

- Understanding characteristics of system life cycle from initiation till retirement or replacement
- Understanding and improving project management practices for SDLC projects within organisation

- The need for adopting latest methodology for information Systems development/acquisition projects based on business focus
- Understand SDLC phases and activities undertaken in each phase and modifications required depending on the nature of deliverables (new application)
- Requirements analysis is a key component of every IT related project. Capturing appropriate requirements helps in improving time, cost and quality requirements of project. It also helps in minimising the scope creep situations.
- Building business case with expected benefits and associated risks forms the basis for evaluation of SDLC projects.
- Software Acquisition activities begins with feasibility Study. Although Software Acquisition is not a phase of traditional SDLC but is an important aspect as majority of companies are now buying readymade solutions.
- Outsourcing of SDLC project must be controlled by service level agreements and ongoing monitoring.
- Majority learning are identified in development and implementation phase, where user involvement must be highest possible.
- Mid-project review helps in making corrections in project execution to minimise risks
- Post-implementation review helps in ensuring benefit realisation and identifying learning form executed project.

8.4 Questions

1. The primary reason why post implementation review is not performed immediately after implementation is to ensure that auditor can review:
   - A. Change management process based on incidents
   - B. Measure changes customer satisfaction levels
   - C. Compare benefits realised with business case
   - D. Comment of conformance of implementation process
2. While conducting post implementation review of software implemented as IS auditor shall be MOST interested in which of the following metrics? Increasing number of:
   - A. Help desk calls for improving service delivery
   - B. Operational errors impacting service delivery
   - C. Change requests approved to add new service
   - D. Updates required in end user operations manual
3. While reviewing the prioritisation and co-ordination of SDLC projects and programme management, IS auditor shall FIRST ensure that:
   - A. Project selection framework is aligned with IT strategy
   - B. Risks are identified, monitored and mitigated in time
   - C. Projects are completed within stipulated time and budget
   - D. Project and programme related documentation is available

4. While reviewing role of senior management in an organisation's IT project and programme management, IS auditor FIRST ensures that the senior management has:
   - A. Allotted appropriate budget and required resources
   - B. Provided for monitoring of risks associated with projects
   - C. Defined key performance indicators for each IT project
   - D. Issued guidelines for implementing framework
5. Which of the following pair of activities IS auditor cannot perform as team member for software development project?
   - A. Conduct the midterm review and recommend controls
   - B. Implement controls and develop integrated test facility
   - C. Develop a control module and perform unit testing
   - D. Implement controls and perform post implementation review
6. Which of the following SDLC project related document shall best provide input on design of controls in application being developed?
   - A. Project business case and feasibility study
   - B. PERT, CPM diagrams and Gantt Charts
   - C. Application system detail design document
   - D. Detailed requirements specifications

8.5 Answers and Explanations

1. C. The primary reason for post-implementation review by IS Auditor is to ensure that the organisation can start realising benefits described in business case. Other options are not primary.
2. B. Increasing number of errors affecting service delivery after implementing new application indicates that the application contains bugs that are affecting service delivery and is a major concern. Other options are normal processes and may or may not affect the service delivery.
3. A. Organisation must have standard project selection framework as per organisation's IT strategy, since that is going to help steering committee in deciding the priority of project selection. Other activities are required for project and programme management.
4. D. Except option D other activities are not performed by senior management
5. D. Auditor cannot audit a project where auditor has been actively involved in implementation.
6. C. Detailed design document shall best provide information on controls that are being embedded in application being developed. Requirements shall also provide information. However, design shall ensure that requirements are captured in design.

SECTION 2: APPENDIX

IS auditor may use checklist to ensure that review is complete. In order to formalize the auditors' role and practices, a master checklist may be prepared. It is suggested that a master checklist for every audit must be prepared by considering the nature of engagement, type and culture of organisation, objectives of audit and expectation from auditor. Using one checklist for all audits have a risk that auditor may end up in concluding on inappropriate findings and auditor's report may not add value to the organisations.

This is a general checklist. For additional information readers may visit websites of Institute of Internal Auditors (www.theiia.org), ISACA (www.isaca.org) and similar organisations which have done exhaustive research in developing suitable audit programmes for various technologies and SDLC projects. The IS Auditor may add more columns (control description, documents inspected, evidence collected, tests performed, result of analysis, conclusion on design, conclusion effectiveness of controls and overall finding) to this checklist to convert it into an audit work paper document..

(Table 8.1) is a sample checklist:

Sl. No.	Checkpoints	SDLC Phase Remark
1.	Whether information system acquisition and / or development policy and procedure documented?	*(General questions covers essential information about control environment within organisation)*
2.	Whether system acquisition and / or development policy and procedure approved by the management?	General *(related to Phases 3 B to 6B)*
3.	Whether the policy and procedure cover the following: Problems faced in existing system and need for replacement Functionality of new IS Security needs Regulatory compliance Acceptance Criteria Proposed roles and responsibilities Transition/Migration to new IS Interfaces with legacy systems Post implementation review Maintenance arrangements.	General *(related all phases)*
4.	Whether policy and procedure documents are communicated/available to the respective users?	General
5.	Whether policy and procedure documents are reviewed and updated at regular intervals?	General
6.	Whether the organisation has evaluated requirement and functionalities of proposed application?	Phase 1 Feasibility Study Phase 2 requirement definition

Sl. No.	Checkpoints	SDLC Phase Remark
7.	Whether the organisation carried out feasibility study in respect of financial, operational and technical feasibility	Phase 1 Feasibility Study
8.	Whether Business case has been prepared listing the benefits against associated risks and approved by management?	Phases 1 and 2 Feasibility Study and requirement definition
9.	Whether selection of vendor and acquisition terms considers: Evaluation of alternative vendors Specification on service levels and deliverables Penalty for delays Escrow mechanism for Source codes Customisation Upgrades Regulatory Compliance Support and maintenance.	Phases 3B and 3C
10.	Whether the organisation has identified and assigned roles in development activities to appropriate stakeholders?	General
11.	Whether the organisation has a separate development, test and production environments?	General *(Mainly related to Phase 6 Testing, Phase 7 UAT and Phase 9 Support)*
12.	Whether the IS developed plan is prepared and approved by the management?	Phase 1 Feasibility study Phases 3/4 Analysis and Design
13.	Whether the testing of IS includes: Confirms the compliance to functional requirements Confirms the compatibility with IS infrastructure Identifies bugs and errors and addresses them by analysing root causes Escalating functionality issues at appropriate levels.	Phase 6 Testing
14.	Whether the adequate documentation for: Preserving test results for future reference Preparation of manuals like systems manual, installation manual, user manual Obtaining user sign off/acceptance	Phase 6 Testing

Sl. No.	Checkpoints	SDLC Phase Remark
15.	Whether the implementation covers the following? User Departments' involvement and their role User Training Acceptance Testing Role of Vendor and period of Support Required IS Infrastructure plan Risk involved and actions required to mitigate the risks Migration plan	Phase 8 Implementation
16.	If the development activities are outsourced, are the outsourcing activities evaluated based on following practices: What is the objective behind Outsourcing? What are the in-house capabilities in performing the job? What is the economic viability? What are the in-house infrastructure deficiencies and the time factor involved? What are the Risks and security concerns? What are the outsourcing arrangement and fall back method? What are arrangements for obtaining the source code for the software? Reviewing the capability and quality of software development activities by visit to vendor's premises? Review of progress of IS development at periodic intervals.	Phase 1 Feasibility Study and Phases 3C to 6C Phase 7 UAT
17.	Whether the organisation carried out a post implementation review of new IS?	General Phase 8 Implementation
18.	Whether a process exists for measuring vendors' performance against the agreed service levels?	Phase 6C and Phase 9 Support and maintenance
19.	Whether post implementation review results are documented?	Phase 8 Implementation

ISBN: 978-81-8441-335-9

₹ 750/- (For Modules I to VII) with DVD

http://cit.icai.org

www.icai.org

1 1 I S A 0 0 2 9

July/2015/P3000(Revised)