UNIT 1 DEVELOPMENT OF INFORMATION SYSTEMS

Structure

1.0 INTRODUCTION

An integrated set of components for collecting, storing, processing, and communicating information is called information system. Business firms, other organizations, and individuals in contemporary society rely on information systems to manage their operations, compete in the marketplace, supply services, and augment personal lives. For instance, modern corporations rely on computerized information systems to process financial accounts and manage human resources; municipal governments rely on information systems to provide basic services to its citizens; and individuals use information systems to study, shop, bank, and invest.

As major new technologies for recording and processing new capabilities have appeared, the invention of movable type in the mid-15th century and the creation of the portable typewriter at the end of the 19th century are but two examples. Each of these inventions led to a profound revolution in the ability to record and disseminates information. The first large-scale mechanized information system was Herman Hollerith's census tabulator. Invented to process the 1890 U.S. census, Hollerith's machine represented a major step in automation, as well as an inspiration to develop computerized information systems. Beginning in the 1970s, personal computers brought some of the advantages of information systems to small businesses and to individuals, and the invention of the World Wide Web in the early 1990s accelerated the creation of an open global computer network. This was accompanied by a dramatic growth in digital human communications (e-mail and electronic conferences), delivery of products (software, music, and movies), and business transactions (buying, selling, and advertising on the Web). As information systems have enabled more diverse human activities, they have exerted a profound influence over society. Many transaction processing systems support electronic commerce over the Internet.

Among these are systems for on-line shopping, banking, and securities trading. Other systems deliver information, educational services, and entertainment on demand. Yet other systems serve to support the search for products with desired attributes, price discovery (for example, via an auction), and delivery of products in an electronic form (software, music, movies, or greeting cards). A growing array of specialized services and information-based products are offered by various organizations on the Web, as an infrastructure for electronic commerce is emerging on a global scale.

1.1 OBJECTIVES

After studying this unit, you should be able to:

- define an Information System;
- know the steps for the development of a typical Information System;
- know about the maintenance of an Information System;
- know about updating an Information System; and
- know about the tools for development of an Information System.

Information System (IS) refers to a system of people, data records and activities that process the data and information in an organization, and it includes the organization's manual and automated processes.

Computer-based information system refers to the specific application software that is used to store data records in a computer system and automates some of the information-processing activities of the organization.

The following are some of the characteristics of an Information System:

- It should be a high quality system that meets or exceeds customer expectations.
- It should reach completion within time and cost estimates.
- It should work effectively and efficiently in the current and planned Information Technology infrastructure.
- It should be inexpensive to maintain and cost-effective.

Example 1: An individual or other business may submit a custom order over the Web that automatically initiates “just-in-time” production to the customer’s exact specifications through an approach known as mass customization. This involves sending orders to the firm’s warehouses and suppliers to deliver materials just in time for a custom-production run. Finally, financial accounts are updated accordingly, and billing is initiated. Along with helping to integrate a firm’s own value chain, transaction processing systems can also serve to integrate an organization’s overall supply chain. This includes all of the various firms involved in designing, marketing, producing, and delivering the goods and services—from raw materials to final delivery.

Example 2: Purchasing an item at a Wal-Mart store generates more than a cash register receipt; it also automatically sends a restocking order to the appropriate supplier. Suppliers can also access a retailer's inventory database over the Web to schedule efficient and timely deliveries. Thus, inter-organizational information systems are essential to supply chain management.

1.2 DEVELOPMENT STEPS OF A TYPICAL INFORMATION SYSTEM

The Systems Development Life Cycle (SDLC) or Software Development Life Cycle in Software Engineering is the process of creating or altering systems, and the models and methodologies that people use to develop these systems. These methodologies form the framework for planning and controlling the creation of an information system. SDLC is any logical process used by a Systems Analyst to develop an information system, including requirements analysis, validation, training, and user ownership.

The following are various steps of an Information System development:

- Requirements analysis
- Design
- Coding
- Testing
- Maintenance

Requirements Analysis

The first essential or vital thing required for any software development is system. Also the system requirement may vary based on the software product that is going to get developed. So a careful analysis has to be made about the system requirement needed for the development of the product. After the analysis and design of the system requirement phase the system required for the development would be complete and the concentration can be on the software development process.

Design

Analysis and Design are very crucial in the whole development cycle. Any glitch in the design phase could be very expensive to solve in the later stage of the software development. Much care is taken during this phase. Analysis is made on the design of the system that is going to be developed. In other words database design, the design of the architecture chosen, functional specification design, low level design documents, high level design documents and so on takes place. Care must be taken to prepare these design documents because the next phases namely the development phase is based on these design documents. If a well structured and analyzed design document

is prepared it would reduce the time taken in the coming steps namely development and testing phases of the software development life cycle. .

Coding

This is the phase where actual development of the system takes place. That is based on the design documents prepared in the earlier phase code is written in the programming technology chosen. After the code is developed generation of code also takes place in this phase. In other words, the code is converted into executables in this phase after code generation.

Programming tools like compilers, interpreters, debuggers etc. are used to generate the code. Different high level programming languages like C, C++, Pascal, and Java are used for coding. With respect to the type of application, the right programming language is chosen.

Testing

A software or system which is not tested would be of poor quality, because this is the phase where system developed would be tested and reports are prepared about bugs or errors in system. To do this testing phase there are different levels and methods of testing like unit testing, system test and so on. Based on the need the testing methods are chosen and reports are prepared about bugs. After this process the system again goes to development phase for correction of errors and again tested. This process continues until the system is found to be error free. To ease the testing process debuggers or testing tools are also available. Once the code is generated, the software program testing begins. Different testing methodologies are available to unravel the bugs that were committed during the previous phases. Different testing tools and methodologies are already available. Some companies build their own testing tools that are tailor made for their own development operations.

Maintenance

The software will definitely undergo change once it is delivered to the customer. There can be many reasons for this change to occur. Change could happen because of some unexpected input values into the system. In addition, the changes in the system could directly affect the software operations. The software should be developed to accommodate changes that could happen during the post implementation period.

Let us discuss each of these to have an overview by an example of a company called **ABC** which has head office in New Delhi and large number of branch offices. ABC has over 9,000 consultants in 71 offices across 51 countries. They had a problem to track leaves of employees and to ensure accuracy and reduce absenteeism, Leave Information of employees was automated.

The objectives of the system are to ensure accuracy and reduce absenteeism, automate leave tracking and manage the system.
The following are some of the requirements of the Information System. Human resource managers should be able to:

- Read, write and update masters (leave, holiday, location, Designation, department).
- Assigning rights to managers to approve leaves.
- Reallocate leaves, carry forward leaves, adjust leaves.
- Receive e-mail for approved leaves.
- Designing, editing, referring department list, designation list, location list , leave master.
- Searching employee of any location, any designation and of any department.
- Changing password.
- Generate reports.

Help employee in:

- Applying for leave
- Changing password
- Referring their rights
- Receiving approval of leaves through email

Help managers in:

- Applying for leave
- Approving leaves
- Referring his leaves and leaves of whom he approved.
- Changing password

Technical Survey

The next step is to decide the platform on which it needs to be applied. Various technologies available are to be surveyed and appropriate decision needs to be taken. Of course, feasibility study is done after knowing "What is to be done?" Is it possible to do "What is to be done?" There were different categories under feasibility study. They are economical, technical and operational feasibility studies.

It is beyond the scope of this unit to describe all the phases of SDLC in detail. The following are different modules of the Leave Management System:

- Leave Master,
- Leave adjustment,
- Leave approval,
- Consolidated Leave report,
- Personal information,

- Department information,
- Holiday master etc.

The modules may change from system to system. Also, appropriate user interface needs to be designed.

Database Design

A database is a collection of inter-related data stored with minimum redundancy to serve many applications. The primary objectives of Database design should be less response time to various queries, more information at low cost, controlled redundancy, clarity and ease of use, accuracy and fast recovery. The organization of data in the database aims to achieve data integrity and data independence. During the design of the database, utmost care needs to be taken so that the objectives are fulfilled.

Coding

Coding should facilitate the identification and retrieval of items of information. The code should be simple and easily understandable.

The code should be adequate for present and anticipated data processing for machine and human use. Care needs to be taken to minimize the clerical effort and computer time required to continue operation.

Testing

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say that the testing is a process of executing a program with the intent of finding an error. A successful test is one that uncovers an as yet undiscovered error. A good test case is one that has a high probability of finding error, if it exists. Tests may be inadequate to detect possibly present errors. Software more or less should conform to the quality and reliability standards.

The different levels of testing are Unit testing , Integration testing, System testing and Acceptance testing.

1.3 TOOLS FOR DEVELOPMENT OF IS

In this section, our focus will be on tools which support the development of IS. We briefly characterize tools in terms of how they support different phases and tasks of software development. Secondly, we describe relationships between methods and tools in more detail through the concept of method-tool companionship. This allows us to explain how tools can support modeling techniques. We seek to apply Meta models in specifying modeling techniques enacted by software development tools. Thus, it is possible to describe the underlying elements of methods (i.e. a Meta model) on which these tools are based. This focus also means that we believe that the use of Meta modeling in local method development is most beneficial when related to

customization of tools. Naturally, meta modeling can be applied for reasons other than local method development but local method development aiming only to specify and compare methods takes us only half-way, because the usefulness of a method is realized only when it is applied. Using Meta models without considering their support in tools would be the same as designing an IS without implementing it.

Since the 1970's numerous attempts have been made to support methods via computer tools (i.e., software applications) .Technological developments have lead to a large number of tools that cover nearly all tasks of software development. At the same time, the term CASE (Computer-Aided Software Engineering) has been extended to denote all types of computer tools from business modeling and requirements gathering to implementation.

The concept of CASE is broad and it includes compilers, project management tools, and editors. These modeling tools are usually used to support early phases of software development. As already mentioned, the term "method" is restricted to mean that part of the method knowledge that is possible to capture into a formalized part of a tool. On the method side, process maps, workflow models, task structures and action diagrams are applied . On the tool side, computing power is applied, for example, to benchmark, compare, and simulate business processes through models. GDSS (Group Decision Support Systems), CSCW (Computer Supported Cooperative Work) and requirements engineering tools can be used in gathering information and organizing it into a structured format so that it can be used in later phases of software development. The methodical aspects of these tools rely on brain-storming, interviews and cooperation. In the system analysis and design phases, CASE tools support methods such as conceptual data modeling (ER models and derivatives) and structured analysis and design (e.g., data flow diagrams, decomposition diagrams and state transition diagrams). Most CASE products, now-a-days, focus on supporting object-oriented methods, and recently tool support has been extended towards business modeling.

The relationship between methods and tools is most obvious in the construction phase: program code written in a high-level language is compiled into machine code. During construction and maintenance, computer aided tools can support version control, configuration management, and reverse engineering.

Method-tool Companionship

Though the technical realization of the companionship between tools and methods can vary, the need to integrate tools and methods is obvious. On the one hand, tools mechanize operations prescribed by methods by storing system representations, transforming representations from one type of model to another, and displaying representations in varying forms. On the other hand, tools empower users by enhancing correctness checking and analytical power, by freeing them from tedious documentation tasks, and by providing multi-user coordination (access and version control). The companionship between tools and methods has also evolved in response to technical innovations. These require extensions to existing methods or entirely new types of methods to support their development.

CASE tools do not provide the same level of support for all types of method knowledge. For example, there are more tools that support model building, representation and checking than there are tools that guide processes or provide group support. Naturally, some aspects of methods lend themselves more easily to automation than others. Nevertheless some method knowledge need to be present in an ISD tool. The presence of methods can also be viewed using CASE tool support functionality, i.e., each type of functionality necessitates different method knowledge.

Abstraction deals with CASE tool support for capturing and representing aspects of object systems. **Checking** of system descriptions is needed to ensure that models are syntactically consistent with method knowledge. **Form conversion** deals with transforming results from one phase or task to another, e.g., analysis models to design models. **Review** deals with semantic validity of system descriptions, whereas checking focuses on syntactic properties of the model.

Remarks on modeling tool support

In the majority of current CASE tools method integration has been implemented only partially. Tool developers have concentrated more on producing technical solutions such as repositories and intelligent knowledge-based support in their products, while the methodical part has been given a lower priority. Hardly any CASE tool developers have introduced methods which have been developed especially for CASE environments .Furthermore, methods which have been coded as a part of a tool, what we call method-dependent CASE, do not allow the further development or extension of methods according to the situation specific needs. We believe that this technically-driven development of CASE has partly led to the rigidity and weak support of users' native methods.

In our opinion, the promise of CASE tools does not lie in the long run in the automated support of old "pen and paper" methods, but in innovative and new uses of computer based methods. Against this backdrop the surprisingly slow diffusion of CASE tools is also more understandable. Research into introducing CASE in an organization reveals that the main problems in the introduction are not the technical changes, but the methodical and cultural changes which the use of the new tool will inevitably cause These observations are obvious, because the effective use of CASE tools is not possible without an adequate experience and knowledge of method use . Introducing method-dependent CASE tools causes changes in the way of working and in the use of methods. Limited possibilities to adapt the tool into an organization's own standards have often led to growing dissatisfaction among users. In contrast to the tool-driven approach, one should select tools so that they fit into the local domain and ISD situations. Several studies of CASE tools speculate that tool development will lead to method-independent CASE tools, instead of tool-driven development. In the same vein, examines several alternative strategies for selecting CASE tools and introduces seven possible ways to exploit CASE. Four of these, building your own CASE tool, ordering your own CASE tool, integrating several tools and experimentations with research prototypes, allow the adaptation of organizations'

methods with the tools. Whereas these researchers have pointed out the demand for flexible CASE support, the technological point of view has still been dominant. Therefore, the opportunities for flexibility in CASE-supported ISD is still at most modest. This problem is discussed from the viewpoint of tool adaptation .

1.4 MAINTENANCE OF INFORMATION SYSTEMS

Maintenance can be classified as:

- Corrective maintenance
- Adaptive maintenance
- Perfective maintenance

Corrective maintenance means repairing processing or performance failures or making changes because of previously uncorrected problems or false assumptions.

Adaptive maintenance means changing the program function according to changing needs.

Perfective maintenance means enhancing the performance or modifying the program(s) to respond to the user's additional or changing needs. Maintenance is actually the implementation of the post implementation review plan.

As important as it is, many programmers and analysts are reluctant to perform or identify themselves with the maintenance effort. There are psychological, personality and professional reasons for this. In any case, a first class effort must be made to ensure that software changes are made properly and in time to keep the system in tune with user specifications.

Maintenance is expensive. One way to reduce maintenance costs is through maintenance management and software modification audits. Software modification consists of program rewrites system level updates, and re-audits of low ranking programs to verify and correct the soft spots. The outcome should be more reliable software, a reduced maintenance backlog, and higher satisfaction and morale among the maintenance staff.

1.5 UPDATING INFORMATION SYSTEMS

The evaluation phase ranks vendors proposals and determines the one best suited. Evaluation of the system is performed to identify its strengths and weaknesses. Evaluation can be done along following dimensions:

- **Operational Evaluation:** Assessment of the manner in which the system functions, including case of use, response time, overall reliability and level of utilization.
- **Organizational Impact:** Identification and measurement of benefits to the organization in such areas as financial concerns, operational efficiency and competitive impact.
- **User Manager Assessment:** Evaluation of the attitudes of Managers within the organization, as well as end-users.
- **Development Performance:** Evaluation of the development process in accordance with such yardsticks as overall development time and effort, conformance to budgets and standards and other project management criteria.

1.6 SUMMARY

Information systems bring new options to the way companies interact, the way organizations are structured, and the way workplaces are designed. In general, use of network-based information systems can significantly lower the costs of communication among workers and firms and enhance coordination on collaborative projects. This has led many organizations to concentrate on their core competencies and to outsource other parts of their value chain to specialized companies. The capability to communicate information efficiently within a firm has also led to the deployment of flatter organizational structures with fewer hierarchical layers.

Nevertheless, information systems do not uniformly lead to higher profits. Success depends on both the skill with which information systems are deployed and the availability of other assets. In particular, "virtual" organizations have emerged that do not rely on physical offices and standard organization charts. Two notable forms are a network organization and a cluster organization.

In a network organization, long-term corporate partners supply goods and services to and through a central firm. Together, a network of small companies can present the appearance of a large corporation. Indeed, at the core of such an organization may be nothing more than a single entrepreneur supported by only a few employees. Thanks to information systems, product specifications in an electronic form can be modified during computerized video conferences between employees throughout an organization—after which supplies can be secured and distribution coordinated, using automatic electronic forms as sales orders are received. Wide area networks, and the Internet in particular, help partnering organizations to facilitate the interaction of widely dispersed business units.

Team members, who are often widely dispersed around the globe, are greatly assisted in their work by the use of corporate intranets and groupware.

Information systems built around portable computers, mobile devices, and groupware have enabled employees to work not just outside the corporate offices but virtually anywhere. "Work is the thing you do, not the place you go to," has become the slogan

of the emerging new workplace. Virtual workplaces include home offices, regional work centres, customers' premises, and mobile offices of people. Employees who work in virtual workplaces outside their company's premises are known as telecommuters.

1.7 FURTHER READINGS

- Fundamentals of Information Systems by Ralph Stair and George Reynolds; Course Technology(Publisher);2008
- Information Systems Today: Managing in the Digital World, Third Edition by Leonard Jessup and Joseph Valacich; Prentice Hall;2007
- http://ocw.mit.edu
- http://www.egyankosh.ac.in