BA4106 **INFORMATION MANAGEMENT** **L T P C**
3 0 0 3

COURSE OBJECTIVES:

- To understand the importance of information in business
- To know about the recent information systems and technologies.

UNIT I INTRODUCTION **9**

Data, Information, Information System, evolution, types based on functions and hierarchy, Enterprise and functional information systems.

UNIT II SYSTEM ANALYSIS AND DESIGN **10**

System development methodologies, Systems Analysis and Design, Data flow Diagram (DFD), Decision table, Entity Relationship (ER), Object Oriented Analysis and Design(OOAD), UML diagram.

UNIT III DATABASE MANAGEMENT SYSTEMS **8**

DBMS – types and evolution, RDBMS, OODBMS, RODBMS, Data warehousing, Data Mart, Data mining.

UNIT IV INTEGRATED SYSTEMS, SECURITY AND CONTROL **9**

Knowledge based decision support systems, Integrating social media and mobile technologies in Information system, Security, IS Vulnerability, Disaster Management, Computer Crimes, Securing the Web.

UNIT V NEW IT INITIATIVES **9**

Introduction to Deep learning, Big data, Pervasive Computing, Cloud computing, Advancements in AI, IoT, Block chain, Crypto currency, Quantum computing

TOTAL: 45 PERIODS

COURSE OUTCOMES:

1. Learn the basics of data and information system.
2. Understand the system development methodologies.
3. Understand database management system and its types.
4. Learn the various technologies in information system and its security.
5. Gains knowledge on effective applications of information systems in business.

REFERENCES:

1. Robert Schultheis and Mary Sumner, Management Information Systems – The Manager' s View, Tata McGraw Hill, 2008.
2. Kenneth C. Laudon and Jane P Laudon, Management Information Systems – Managing the Digital Firm, 15 th edition, 2018.
3. Panneerselvam. R, Database Management Systems, 3rd Edition, PHI Learning, 2018.

CHAPTER 1

INTRODUCTION TO INFORMATION MANAGEMENT

Data

Data is defined as facts or figures, or information that's stored in or used by a computer. An example of data is information collected for a research paper. the quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media.

Information

Information is a stimulus that has meaning in some context for its receiver. When information is entered into and stored in a computer, it is generally referred to as data. After processing (such as formatting and printing), output data can again be perceived as information.

Information (shortened as info or info.) is that which informs, i.e. that from which data can be derived. At its most fundamental, information is any propagation of cause and effect within a system. Information is conveyed either as the content of a message or through direct or indirect observation of something. That which is perceived can be construed as a message in its own right, and in that sense, information is always conveyed as the content of a message. Information can be encoded into various forms for transmission and interpretation. For example, information may be encoded into signs, and transmitted via signals.

These are difficult times for all organizations of all sizes and in all sectors. On the one hand, customers have ever-increasing expectations in terms of the speed and quality of service they expect and, on the other resources are continually under pressure.

This document sets out how effective information and records management can help any organization to move forward in this challenging environment through,

- achieving cost and efficiency savings;
- making best use of information assets and
- Taking advantage of the opportunities offered by new technologies.

Intelligence

Intelligence has been defined in many different ways such as in terms of one's capacity for logic, abstract thought, understanding, self-awareness, communication, learning, emotional knowledge, memory, planning, creativity and problem solving.

Knowledge

Knowledge is a familiarity, awareness or understanding of someone or something, such as facts, information, descriptions, or skills, which is acquired through experience or education by perceiving, discovering, or learning. Knowledge can refer to a theoretical or practical understanding of a subject.

Importance:

- Learning Better
- Setting Goals As You Learn
- Learn Complex Things Faster
- Knowledge Helps You Solve Problems
- Understanding Yourself

Information Technology (IT)

Information technology (IT) is the application of computers and telecommunications equipment to store, retrieve, transmit and manipulate data, often in the context of a business or other enterprise.

Need

- Education is a lifelong process therefore anytime anywhere access to it is the need
- Information explosion is an ever increasing phenomena therefore there is need to get access to this information
- Education should meet the needs of variety of learners and therefore IT is important in meeting this need
- It is a requirement of the society that the individuals should posses technological literacy
- We need to increase access and bring down the cost of education to meet the challenges of illiteracy and poverty-IT is the answer

Importance

- access to variety of learning resources
- immediacy to information
- anytime learning
- anywhere learning
- collaborative learning
- multimedia approach to education
- authentic and up to date information
- access to online libraries
- teaching of different subjects made interesting
- educational data storage
- distance education
- access to the source of information
- Multiple communication channels-e-mail, chat, forum, blogs, etc.

- access to open courseware
- better accesses to children with disabilities
- reduces time on many routine tasks

Information system

An information system (IS) is a system composed of people and computers that processes or interprets information. The term is also sometimes used in more restricted senses to refer to only the software used to run a computerized database or to refer to only a computer system.

Importance

1. To control the creation and growth of records

Despite decades of using various non-paper storage media, the amount of paper in our offices continues to escalate. An effective records information system addresses both creation control (limits the generation of records or copies not required to operate the business) and records retention (a system for destroying useless records or retiring inactive records), thus stabilizing the growth of records in all formats.

2. To reduce operating costs

Recordkeeping requires administrative dollars for filing equipment, space in offices, and staffing to maintain an organized filing system (or to search for lost records when there is no organized system).It costs considerably less per linear foot of records to store inactive records in a Data Records Center versus in the office and there is an opportunity to effect some cost savings in space and equipment, and an opportunity to utilize staff more productively - just by implementing a records management program.

3. To improve efficiency and productivity

Time spent searching for missing or misfiled records are non-productive. A good records management program (e.g. a document system) can help any organization upgrade its recordkeeping systems so that information retrieval is enhanced, with corresponding improvements in office efficiency and productivity. A well designed and operated filing system with an effective index can facilitate retrieval and deliver information to users as quickly as they need it.

Moreover, a well managed information system acting as a corporate asset enables organizations to objectively evaluate their use of information and accurately lay out a roadmap for improvements that optimize business returns.

4. To assimilate new records management technologies

A good records management program provides an organization with the capability to assimilate new technologies and take advantage of their many benefits. Investments in new computer systems whether this is financial, business or otherwise, don't solve filing problems unless current manual recordkeeping or bookkeeping systems are analyzed (and occasionally, overhauled) before automation is applied.

5. To ensure regulatory compliance

In terms of recordkeeping requirements, China is a heavily regulated country. These laws can create major compliance problems for businesses and government agencies since they can be difficult to locate, interpret and apply. The only way an organization can be reasonably sure that it is in full compliance with laws and regulations is by operating a good management information system which takes responsibility for regulatory compliance, while working closely with the local authorities. Failure to comply with laws and regulations could result in severe fines, penalties or other legal consequences.

6. To minimize litigation risks

Business organizations implement management information systems and programs in order to reduce the risks associated with litigation and potential penalties. This can be equally true in Government agencies. For example, a consistently applied records management program can reduce the liabilities associated with document disposal by providing for their systematic, routine disposal in the normal course of business.

7. To safeguard vital information

Every organization, public or private, needs a comprehensive program for protecting its vital records and information from catastrophe or disaster, because every organization is vulnerable to loss. Operated as part of a good management information system, vital records programs preserve the integrity and confidentiality of the most important records and safeguard the vital information assets according to a "Plan" to protect the records. This is especially the case for financial information whereby ERP (Enterprise Resource Planning) systems are being deployed in large companies.

8. To support better management decision making

In today's business environment, the manager that has the relevant data first often wins, either by making the decision ahead of the competition, or by making a better, more informed decision. A good management information system can help ensure that managers and executives have the information they need when they need it.

Likewise, implementing a good ERP system to take account of all the business‘ processes both financial and operational will give an organization more advantages than one who was operating a manual based system.

9. To preserve the corporate memory

An organization's files, records and financial data contain its institutional memory, an irreplaceable asset that is often overlooked. Every business day, you create the records, which could become background data for future management decisions and planning.

10. To foster professionalism in running the business

A business office with files, documents and financial data askew, stacked on top of file cabinets and in boxes everywhere, creates a poor working environment. The perceptions of customers and the public, and "image" and "morale" of the staff, though hard to quantify in cost-benefit terms, may be among the best reasons to establish a good management information system.

Evolution

The first business application of computers (in the mid- 1950s) performed repetitive, high-volume, transaction-computing tasks. The computers‖ crunched numbers‖ summarizing and organizing transactions and data in the accounting, finance, and human resources areas. Such systems are generally called transaction processing systems (TPSs).

Management Information Systems (MISs): these systems access, organize, summarize and display information for supporting routine decision making in the functional areas.Office Automation Systems (OASs): such as word processing systems were developed to support office and clerical workers.

Decision Support Systems: were developed to provide computer based support for complex, non routine decision. „ End- user computing: The use or development of information systems by the principal users of the systems‘ outputs, such as analysts, managers, and other professionals.

Intelligent Support System (ISSs): Include expert systems which provide the stored knowledge of experts to non experts, and a new type of intelligent system with machine- learning capabilities that can learn from historical cases. „ Knowledge Management Systems: Support the creating, gathering, organizing, integrating and disseminating of organizational knowledge.

Data Warehousing: A data warehouse is a database designed to support DSS, ESS and other analytical and end-user activities. „ Mobile Computing: Information systems that support employees who are working with customers or business partners outside the physical boundaries of their company; can be done over wire or wireless networks.

Kinds of Information Systems

- Organizational Hierarchy
- Organizational Levels
- Information Systems

Four General Kinds of IS

- Operational-level systems
    - Support operational managers by monitoring the day-to-day‘s elementary activities and transactions of the organization. e.g. TPS.
- Knowledge-level systems
    - Support knowledge and data workers in designing products, distributing information, and coping with paperwork in an organization. e.g. KWS, OAS
- Management-level systems
    - Support the monitoring, controlling, decision-making, and administrative activities of middle managers. e.g. MIS, DSS
- Strategic-level systems
    - Support long-range planning activities of senior management. e.g. ESS
- Executive Support Systems (ESS)
- Management Information Systems (MIS)
- Decision Support Systems (DSS)
- Knowledge Work Systems (KWS)
- Office Automation Systems (OAS)
- Transaction Processing Systems (TPS)

Transaction Processing Systems (TPS)

Computerized system that performs and records the daily routine transactions necessary to conduct the business; these systems serve the operational level of the organization

- TYPE: Operational-level
- INPUTS: transactions, events
- PROCESSING: updating
- OUTPUTS: detailed reports
- USERS: operations personnel, supervisors
- DECISION-MAKING: highly structured

EXAMPLE: payroll, accounts payable

Office Automation Systems (OAS)

Computer system, such as word processing, electronic mail system, and scheduling system, that is designed to increase the productivity of data workers in the office.

- TYPE: Knowledge-level
- INPUTS: documents, schedule
- PROCESSING: document management, scheduling, communication
- OUTPUTS: documents; schedules
- USERS: clerical workers

EXAMPLE: document imaging system

Knowledge Work Systems (KWS)

Information system that aids knowledge workers in the creation and integration of new knowledge in the organization.

- TYPE: Knowledge-level
- INPUTS: design specifications
- PROCESSING: modelling
- OUTPUTS: designs, graphics
- USERS: technical staff; professionals

EXAMPLE: Engineering workstations

Decision Support Systems (DSS)

Information system at the management level of an organization that combines data and sophisticated analytical models or data analysis tools to support semi-structured and unstructured decision making.

- TYPE: Management-level
- INPUTS: low volume data
- PROCESSING: simulations, analysis
- OUTPUTS: decision analysis
- USERS: professionals, staff managers
- DECISION-MAKING: semi-structured

EXAMPLE: sales region analysis

Management Information Systems (MIS)

Information system at the management level of an organization that serves the functions of planning, controlling, and decision making by providing routine summary and exception reports.

- TYPE: Management-level
- INPUTS: high volume data
- PROCESSING: simple models
- OUTPUTS: summary reports
- USERS: middle managers
- DECISION-MAKING: structured to semi-structured

EXAMPLE: annual budgeting

Executive Support Systems (ESS)

Information system at the strategic level of an organization that address unstructured decision making through advanced graphics and communications.

TYPE: Strategic level

- INPUTS: aggregate data; internal and external
- PROCESSING: interactive
- OUTPUTS: projections
- USERS: senior managers
- DECISION-MAKING: highly unstructured

EXAMPLE: 5 year operating plan

Classification of IS by Organizational Structure

- Departmental Information Systems
- Enterprise Information System
- Inter-organizational Systems
  - NYCE
  - SABRE or APOLLO

Classification of IS by Functional Area

- The accounting information system
- The finance information system
- The manufacturing (operations, production) information system
- The marketing information system
- The human resources information system

System development methodologies

Introduction

A system development methodology refers to the framework that is used to structure, plan, and control the process of developing an information system. A wide variety of such frameworks have evolved over the years, each with its own recognized strengths and weaknesses. One system development methodology is not necessarily suitable for use by all projects. Each of the available methodologies is best suited to specific kinds of projects, based on various technical, organizational, project and team considerations. CMS has considered each of the major prescribed methodologies in context with CMS‘ business, applications, organization, and technical environments. As a result, CMS requires the use of any of the following linear and iterative methodologies for CMS systems development, as appropriate.

Basic Principles:

1. Project is divided into sequential phases, with some overlap and splashback acceptable between phases.
2. Emphasis is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.
3. 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:

1. Ideal for supporting less experienced project teams and project managers, or project teams whose composition fluctuates.
2. The orderly sequence of development steps and strict controls for ensuring the adequacy of documentation and design reviews helps ensure the quality, reliability, and maintainability of the developed software.
3. Progress of system development is measurable.
4. Conserves resources.

Weaknesses:

1. Inflexible, slow, costly and cumbersome due to significant structure and tight controls.
2. Project progresses forward, with only slight movement backward.
3. Little room for use of iteration, which can reduce manageability if used.
4. Depends upon early identification and specification of requirements, yet users may not be able to clearly define what they need early in the project.
5. Requirements inconsistencies, missing system components, and unexpected development needs are often discovered during design and coding.
6. Problems are often not discovered until system testing.

7. System performance cannot be tested until the system is almost fully coded, and under-capacity may be difficult to correct.
8. Difficult to respond to changes. Changes that occur later in the life cycle are more costly and are thus discouraged.
9. Produces excessive documentation and keeping it updated as the project progresses is time-consuming.
10. Written specifications are often difficult for users to read and thoroughly appreciate.
11. Promotes the gap between users and developers with clear division of responsibility.

Situations where most appropriate:

1. Project is for development of a mainframe-based or transaction-oriented batch system.
2. Project is large, expensive, and complicated.
3. Project has clear objectives and solution.
4. Pressure does not exist for immediate implementation.
5. Project requirements can be stated unambiguously and comprehensively.
6. Project requirements are stable or unchanging during the system development life cycle.
7. User community is fully knowledgeable in the business and application.
8. Team members may be inexperienced.
9. Team composition is unstable and expected to fluctuate.
10. Project manager may not be fully experienced.
11. Resources need to be conserved.
12. Strict requirement exists for formal approvals at designated milestones.

Situations where least appropriate:

1. Large projects where the requirements are not well understood or are changing for any reasons such as external changes, changing expectations, budget changes or rapidly changing technology.
2. Web Information Systems (WIS) primarily due to the pressure of implementing a WIS project quickly; the continual evolution of the project requirements; the need for experienced, flexible team members drawn from multiple disciplines; and the inability to make assumptions regarding the users‘ knowledge level.
3. Real-time systems.
4. Event-driven systems.
5. Leading-edge applications.

Prototyping

Basic Principles

1. Not a standalone, complete development methodology, but rather an approach to handling selected portions of a larger, more traditional development methodology (i.e., Incremental, Spiral, or Rapid Application Development (RAD)).

2. Attempts to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process.
3. User is involved throughout the process, which increases the likelihood of user acceptance of the final implementation.
4. Small-scale mock-ups of the system are developed following an iterative modification process until the prototype evolves to meet the users‘ requirements.
5. While most prototypes are developed with the expectation that they will be discarded, it is possible in some cases to evolve from prototype to working system.
6. A basic understanding of the fundamental business problem is necessary to avoid solving the wrong problem.

Strengths:

1. ―Addresses the inability of many users to specify their information needs, and the difficulty of systems analysts to understand the user‘s environment, by providing the user with a tentative system for experimental purposes at the earliest possible time.‖ (Janson and Smith, 1985)
2. ―Can be used to realistically model important aspects of a system during each phase of the traditional life cycle.‖
3. Improves both user participation in system development and communication among project stakeholders.
4. Especially useful for resolving unclear objectives; developing and validating user requirements; experimenting with or comparing various design solutions; or investigating both performance and the human computer interface.
5. Potential exists for exploiting knowledge gained in an early iteration as later iterations are developed.
6. Helps to easily identify confusing or difficult functions and missing functionality.
7. May generate specifications for a production application.
8. Encourages innovation and flexible designs.
9. Provides quick implementation of an incomplete, but functional, application.

Weaknesses:

1. Approval process and control is not strict.
2. 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.
3. Requirements may frequently change significantly.
4. Identification of non-functional elements is difficult to document.
5. Designers may prototype too quickly, without sufficient up-front user needs analysis, resulting in an inflexible design with narrow focus that limits future system potential.
6. Designers may neglect documentation, resulting in insufficient justification for the final product and inadequate records for the future.
7. Can lead to poorly designed systems. Unskilled designers may substitute prototyping for sound design, which can lead to a ―quick and dirty system‖ without global consideration of the

integration of all other components. While initial software development is often built to be a ―throwaway‖, attempting to retroactively produce a solid system design can sometimes be problematic.

8. Can lead to false expectations, where the customer mistakenly believes that the system is ―finished‖ when in fact it is not; the system looks good and has adequate user interfaces, but is not truly functional.
9. Iterations add to project budgets and schedules, thus the added costs must be weighed against the potential benefits. Very small projects may not be able to justify the added time and money, while only the high-risk portions of very large, complex projects may gain benefit from prototyping.
10. Prototype may not have sufficient checks and balances incorporated.

Situations where most appropriate:

1. Project is for development of an online system requiring extensive user dialog, or for a less well-defined expert and decision support system.
2. Project is large with many users, interrelationships, and functions, where project risk relating to requirements definition needs to be reduced.
3. Project objectives are unclear.
4. Pressure exists for immediate implementation of something.
5. Functional requirements may change frequently and significantly.
6. User is not fully knowledgeable.
7. Team members are experienced (particularly if the prototype is not a throw-away).
8. Team composition is stable.
9. Project manager is experienced.
10. No need exists to absolutely minimize resource consumption.
11. No strict requirement exists for approvals at designated milestones.
12. Analysts/users appreciate the business problems involved, before they begin the project.
13. Innovative, flexible designs that will accommodate future changes are not critical.

Situations where least appropriate:

1. Mainframe-based or transaction-oriented batch systems.
2. Web-enabled e-business systems.
3. Project team composition is unstable.
4. Future scalability of design is critical.
5. Project objectives are very clear; project risk regarding requirements definition is low.

Incremental

Basic Principles

Various methods are acceptable for combining linear and iterative system development methodologies, with the primary objective of each being to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process:

1. 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; OR
2. Overall requirements are defined before proceeding to evolutionary, mini-Waterfall development of individual increments of the system, OR
3. The initial software concept, requirements 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).

Strengths:

1. Potential exists for exploiting knowledge gained in an early increment as later increments are developed.
2. 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.
3. Stakeholders can be given concrete evidence of project status throughout the life cycle.
4. Helps to mitigate integration and architectural risks earlier in the project.
5. Allows delivery of a series of implementations that are gradually more complete and can go into production more quickly as incremental releases.
6. Gradual implementation provides the ability to monitor the effect of incremental changes, isolate issues and make adjustments before the organization is negatively impacted.

Weaknesses:

1. When utilizing a series of mini-Waterfalls for a small part of the system before moving on to the next increment, there is usually a lack of overall consideration of the business problem and technical requirements for the overall system.
2. Since some modules will be completed much earlier than others, well-defined interfaces are required.
3. Difficult problems tend to be pushed to the future to demonstrate early success to management.

Situations where most appropriate:

1. Large projects where requirements are not well understood or are changing due to external changes, changing expectations, budget changes or rapidly changing technology.
2. Web Information Systems (WIS) and event-driven systems.
3. Leading-edge applications.

Situations where least appropriate:

1. Very small projects of very short duration.
2. Integration and architectural risks are very low.
3. Highly interactive applications where the data for the project already exists (completely or in part), and the project largely comprises analysis or reporting of the data.

Spiral

Basic Principles:

1. Focus is on risk assessment and on minimizing project risk by breaking a project into smaller segments and providing more ease-of-change during the development process, as well as providing the opportunity to evaluate risks and weigh consideration of project continuation throughout the life cycle.

2. ―Each cycle involves a progression through the same sequence of steps, for each portion of the product and for each of its levels of elaboration, from an overall concept-of-operation document down to the coding of each individual program.‖ (Boehm, 1986)

3. Each trip around the spiral traverses four basic quadrants: (1) determine objectives, alternatives, and constraints of the iteration; (2) evaluate alternatives; identify and resolve risks; (3) develop and verify deliverables from the iteration; and (4) plan the next iteration. (Boehm, 1986 and 1988)

4. Begin each cycle with an identification of stakeholders and their win conditions, and end each cycle with review and commitment. (Boehm, 2000)

Strengths:

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

Weaknesses:

1. Challenging to determine the exact composition of development methodologies to use for each iteration around the Spiral.
2. Highly customized to each project, and thus is quite complex, limiting reusability.
3. A skilled and experienced project manager is required to determine how to apply it to any given project.
4. There are no established controls for moving from one cycle to another cycle. Without controls, each cycle may generate more work for the next cycle.
5. There are no firm deadlines. Cycles continue with no clear termination condition, so there is an inherent risk of not meeting budget or schedule.
6. Possibility exists that project ends up implemented following a Waterfall framework.

Situations where most appropriate:

1. Real-time or safety-critical systems.
2. Risk avoidance is a high priority.
3. Minimizing resource consumption is not an absolute priority.
4. Project manager is highly skilled and experienced.
5. Requirement exists for strong approval and documentation control.
6. Project might benefit from a mix of other development methodologies.
7. A high degree of accuracy is essential.
8. Implementation has priority over functionality, which can be added in later versions.

Situations where least appropriate:

1. Risk avoidance is a low priority.
2. A high degree of accuracy is not essential.
3. Functionality has priority over implementation.
4. Minimizing resource consumption is an absolute priority.

Rapid Application Development (RAD)

Basic Principles

1. Key objective is for fast development and delivery of a high quality system at a relatively low investment cost.
2. Attempts to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process.
3. 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. These tools may include 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.
4. Key emphasis is on fulfilling the business need, while technological or engineering excellence is of lesser importance.
5. 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.
6. 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.
7. Active user involvement is imperative.
8. Iteratively produces production software, as opposed to a throwaway prototype.

9. Produces documentation necessary to facilitate future development and maintenance.
10. Standard systems analysis and design techniques can be fitted into this framework.

Strengths:

1. The operational version of an application is available much earlier than with Waterfall, Incremental, or Spiral frameworks.
2. Because RAD produces systems more quickly and to a business focus, this approach tends to produce systems at a lower cost.
3. Engenders a greater 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 developers are seen as gaining more satisfaction from producing successful systems quickly.
4. Concentrates on essential system elements from user viewpoint.
5. Provides the ability to rapidly change system design as demanded by users.
6. Produces a tighter fit between user requirements and system specifications.
7. Generally produces a dramatic savings in time, money, and human effort.

Weaknesses:

1. More speed and lower cost may lead to lower overall system quality.
2. Danger of misalignment of developed system with the business due to missing information.
3. Project may end up with more requirements than needed (gold-plating).
4. Potential for feature creep where more and more features are added to the system over the course of development.
4. Potential for inconsistent designs within and across systems.
5. Potential for violation of programming standards related to inconsistent naming conventions and inconsistent documentation.
6. Difficulty with module reuse for future systems.
7. Potential for designed system to lack scalability.
8. Potential for lack of attention to later system administration needs built into system.
9. High cost of commitment on the part of key user personnel.
10. Formal reviews and audits are more difficult to implement than for a complete system.
11. Tendency for difficult problems to be pushed to the future to demonstrate early success to management.

Situations where most appropriate:

1. Project is of small-to-medium scale and of short duration (no more than 6 man-years of development effort).
2. Project scope is focused, such that the business objectives are well defined and narrow.
3. Application is highly interactive, has a clearly defined user group, and is not computationally complex.
4. Functionality of the system is clearly visible at the user interface.

5. Users possess detailed knowledge of the application area.
6. Senior management commitment exists to ensure end-user involvement.
7. Requirements of the system are unknown or uncertain.
8. It is not possible to define requirements accurately ahead of time because the situation is new or the system being employed is highly innovative.
9. Team members are skilled both socially and in terms of business.
10. Team composition is stable; continuity of core development team can be maintained.
11. Effective project control is definitely available.
12. Developers are skilled in the use of advanced tools.
13. Data for the project already exists (completely or in part), and the project largely comprises analysis or reporting of the data.
14. Technical architecture is clearly defined.
15. Key technical components are in place and tested.
16. Technical requirements (e.g., response times, throughput, database sizes, etc.) are reasonable and well within the capabilities of the technology being used. Targeted performance should be less than 70% of the published limits of the technology.
17. Development team is empowered to make design decisions on a day-to-day basis without the need for consultation with their superiors, and decisions can be made by a small number of people who are available and preferably co-located.

Situations where least appropriate:

1. Very large, infrastructure projects; particularly large, distributed information systems such as corporate-wide databases.
2. Real-time or safety-critical systems.
3. Computationally complex systems, where complex and voluminous data must be analyzed, designed, and created within the scope of the project.
4. Project scope is broad and the business objectives are obscure.
5. Applications in which the functional requirements have to be fully specified before any programs are written.
6. Many people must be involved in the decisions on the project, and the decision makers are not available on a timely basis or they are geographically dispersed.
7. The project team is large or there are multiple teams whose work needs to be coordinated.
8. When user resource and/or commitment is lacking.
9. There is no project champion at the required level to make things happen.
10. Many new technologies are to be introduced within the scope of the project, or the technical architecture is unclear and much of the technology will be used for the first time within the project.

Functional Information System (FIS)

Supports a functional area by increasing its internal effectiveness and efficiency. Typically found for:

- Finance (FIN): provide internal and external professional access to stock, investment and capital spending information.
- Accounting (ACC): similar to financial MIS more related to invoicing, payroll, receivables.
- Marketing (MKT): pricing, distribution, promotional, and information by customer and salesperson.
- Operations (OPS): regular reports on production, yield, quality, inventory levels. These systems typically deal with manufacturing, sourcing, and supply chain management.
- Human Resources Management (HR): employees, benefits, hiring‘s, etc.

A summary of capabilities of a FIS are organized by functional area in the following chart:

- From the pyramid Each vertical section represents a functional area of the organization, and thus a vertical view can be compared to a functional view of the organization
- Information systems can be designed to support the functional areas or traditional departments

such as, accounting, finance, marketing, human resources, and manufacturing, of an organization

- Such systems are classified as _functional information systems‘. Functional information systems typically follow the organizational structure
- Functional information systems are typically focused on increasing the efficiency of a particular department or a functional area.
- One disadvantage of functional systems is that although they may support a particular functional area effectively, they may be incompatible to each other (NO interaction between internal systems).
- Such systems, rather than aiding organizational performance will act as inhibitors to an organization's development and change.
- Organizations have realized that in order to be agile and efficient they need to focus on organizational processes
- A process may involve more than one functional area.
- Some Information Systems are cross-functional
- Example: A TPS can affect several different business areas: Accounting, Human Resources, Production, etc.
- Some Information Systems concentrate on one particular business area (Accounting for example)
- These systems are:
    - Marketing Systems
    - Manufacturing Systems
    - Human Resource Systems
    - Accounting Systems
    - Financial Management Systems

DSS

A Decision Support System (DSS) is a computer-based information system that supports business or organizational decision-making activities.

DSSs serve the management, operations, and planning levels of an organization (usually mid and higher management) and help to make decisions, which may be rapidly changing and not easily specified in advance (Unstructured and Semi-Structured decision problems). Decision support systems can be either fully computerized, human or a combination of both.

Decision support systems generally involve non-programmed decisions. Therefore; there will be no exact report, content or format for these systems. Reports are generated on the fly.

Attributes of a DSS

- Adaptability and flexibility
- High level of Interactivity

- Ease of use
- Efficiency and effectiveness
- Complete control by decision-makers.
- Ease of development
- Extendibility
- Support for modeling and analysis
- Support for data access
- Standalone, integrated and Web-based

Characteristics of a DSS

- Support for decision makers in semi structured and unstructured problems.
- Support for managers at various managerial levels, ranging from top executive to line managers.
- Support for individuals and groups. Less structured problems often requires the involvement of several individuals from different departments and organization level.
- Support for interdependent or sequential decisions.
- Support for intelligence, design, choice, and implementation.
- Support for variety of decision processes and styles
- DSSs are adaptive over time.

Benefits of DSS

- Improves efficiency and speed of decision making activities
- Increases the control, competitiveness and capability of futuristic decision making of the organization
- Facilitates interpersonal communication
- Encourages learning or training
- Since it is mostly used in non-programmed decisions, it reveals new approaches and sets up new evidences for an unusual decision
- Helps automate managerial processes

Components of a DSS

Following are the components of the Decision Support System:

- Database Management System (DBMS): To solve a problem the necessary data may come from internal or external database. In an organization, internal data are generated by a system such as TPS and MIS.External data come from a variety of sources such as newspapers, online data services, databases (financial, marketing, human resources).

- Model Management system: It stores and accesses models that managers use to make decisions. Such models are used for designing manufacturing facility, analyzing the financial health of an organization. Forecasting demand of a product or service etc.

  Support Tools: Support tools like online help; pull down menus, user interfaces, graphical analysis, error correction mechanism, facilitates the user interactions with the system.

Classification of DSS

There are several ways to classify DSS. Hoi Apple and Whinstone classify DSS in following:

- Text Oriented DSS: It contains textually represented information that could have a bearing on decision. It allows documents to be electronically created, revise and viewed as needed
- Database Oriented DSS: Database plays a major role here; it contains organized and highly structured data.
- Spreadsheet Oriented DSS: it contains information in spread sheets that allows create, view, modify procedural knowledge and also instruct the system to execute self-contained instructions. The most popular tool is Excel and Lotus 1-2-3.
- Solver Oriented DSS: it is based on a solver, which is an algorithm or procedure written for performing certain calculations and particular program type.
- Rules Oriented DSS: It follows certain procedures adopted as rules.
- Rules Oriented DSS: Procedures are adopted in rules oriented DSS. Export system is the example.
- Compound DSS: It is built by using two or more of the five structures explained above

Types of DSS

- Status Inquiry System: helps in taking operational management level or middle level management decisions, for example daily schedules of jobs to machines or machines to operators.
- Data Analysis System: needs comparative analysis and makes use of formula or an algorithm, for example cash flow analysis, inventory analysis etc.
- Information Analysis System: In this system data is analyzed and the information report is generated. For example, sales analysis, accounts receivable systems, market analysis etc.
- Accounting System: keep tracks of accounting and finance related information, for example, final account, accounts receivables, accounts payables etc. that keep track of the major aspects of the business.
- Model Based System: simulation models or optimization models used for decision- making used infrequently and creates general guidelines for operation or management.

EIS

Executive support systems are intended to be used by the senior managers directly to provide support to non-programmed decisions in strategic management.

These information are often external, unstructured and even uncertain. Exact scope and context of such information is often not known beforehand.

This information is intelligence based:

- Market intelligence
- Investment intelligence
- Technology intelligence

Examples of Intelligent Information

Following are some examples of intelligent information, which is often source of an ESS:

- External databases
- Technology reports like patent records etc.
- Technical reports from consultants
- Market reports
- Confidential information about competitors
- Speculative information like market conditions
- Government policies
- Financial reports and information

Advantages of ESS

- Easy for upper level executive to use
- Ability to analyze trends
- Augmentation of managers' leadership capabilities
- Enhance personal thinking and decision making
- Contribution to strategic control flexibility
- Enhance organizational competitiveness in the market place
- Instruments of change
- Increased executive time horizons.
- Better reporting system
- Improved mental model of business executive
- Help improve consensus building and communication
- Improve office automation
- Reduce time for finding information

- Detail examination of critical success factor
- Better understanding
- Time management
- Increased communication capacity and quality

Disadvantage of ESS

- Functions are limited
- Hard to quantify benefits
- Executive may encounter information overload
- System may become slow
- Difficult to keep current data
- May lead to less reliable and insecure data
- Excessive cost for small company

KMS

All the systems we are discussing here come under knowledge management category. A knowledge management system is not radically different from all these information systems, but it just extends the already existing systems by assimilating more information.

As we have seen data is raw facts, information is processed and/or interpreted data and knowledge is personalized information.

What is knowledge?

- personalized information
- state of knowing and understanding
- an object to be stored and manipulated
- a process of applying expertise
- a condition of access to information
- potential to influence action

Sources of Knowledge of an Organization

- Intranet
- Data warehouses and knowledge repositories
- Decision support tools
- Groupware for supporting collaboration
- Networks of knowledge workers

Purpose of a KMS

- Improved performance
- Competitive advantage
- Innovation
- Sharing of knowledge
- Integration
- Continuous improvement by:
    - Driving strategy
    - Starting new lines of business
    - Solving problems faster
    - Developing professional skills
    - Recruit and retain talent

Activities in Knowledge Management

- Start with the business problem and the business value to be delivered first.
- Identify what kind of strategy to pursue to deliver this value and address the KM problem
- Think about the system required from a people and process point of view.
- Finally, think about what kind of technical infrastructure are required to support the people and processes.
- Implement system and processes with appropriate change management and iterative staged release.

GIS

A geographic information system (GIS) is a computer system designed to capture, store, manipulate, analyze, manage, and present all types of spatial or geographical data.

GIS techniques and technology

Modern GIS technologies use digital information, for which various digitized data creation methods are used. The most common method of data creation is digitization, where a hard copy map or survey plan is transferred into a digital medium through the use of a CAD program, and geo-referencing capabilities. With the wide availability of ortho-rectified imagery (both from satellite and aerial sources), heads-up digitizing is becoming the main avenue through which geographic data is extracted. Heads-up digitizing involves the tracing of geographic data directly on top of the aerial imagery instead of by the traditional method of tracing the geographic form on a separate digitizing tablet (heads-down digitizing).

Data representation

GIS data represents real objects (such as roads, land use, elevation, trees, waterways, etc.) with digital data determining the mix. Real objects can be divided into two abstractions: discrete objects (e.g., a

house) and continuous fields (such as rainfall amount, or elevations). Traditionally, there are two broad methods used to store data in a GIS for both kinds of abstractions mapping references: raster images and vector. Points, lines, and polygons are the stuff of mapped location attribute references. A new hybrid method of storing data is that of identifying point clouds, which combine three-dimensional points with RGB information at each point, returning a "3D color image". GIS thematic maps then are becoming more and more realistically visually descriptive of what they set out to show or determine.

Data capture

Example of hardware for mapping (GPS and laser rangefinder) and data collection (rugged computer). The current trend for geographical information system (GIS) is that accurate mapping and data analysis are completed while in the field. Depicted hardware (field-map technology) is used mainly for forest inventories, monitoring and mapping.

Data capture entering information into the system consumes much of the time of GIS practitioners. There are a variety of methods used to enter data into a GIS where it is stored in a digital format.

Existing data printed on paper or PET film maps can be digitized or scanned to produce digital data. A digitizer produces vector data as an operator traces points, lines, and polygon boundaries from a map. Scanning a map results in raster data that could be further processed to produce vector data.

A GIS was used to register and combine the two images to render the three-dimensional perspective view looking down the San Andreas Fault, using the Thematic Mapper image pixels, but shaded using the elevation of the landforms. The GIS display depends on the viewing point of the observer and time of day of the display, to properly render the shadows created by the sun's rays at that latitude, longitude, and time of day.

GIS data mining

GIS or spatial data mining is the application of data mining methods to spatial data. Data mining, which is the partially automated search for hidden patterns in large databases, offers great potential benefits for applied GIS-based decision making. Typical applications including environmental monitoring. A characteristic of such applications is that spatial correlation between data measurements requires the use of specialized algorithms for more efficient data analysis.

International information systems

International information systems (IIS) technology is a field where academic research is sparse. These contrasts starkly with a growing concern of practitioners who have come to regard IIS as a double threat: they are often vitally critical for the globally oriented firm, but at the same time they are perceived as difficult and risky. The areas of importance for practitioners are less well researched than others. A theory building methodology is discussed and recommended for an initial research project.

- Global business drivers are general cultural factors and specific business factors
- Global culture, created by TV and other global media (e.g., movies) permit cultures to develop common expectations about right and wrong, desirable and undesirable, heroic and cowardly
- A global knowledge base--strengthened by educational advances in Latin America, China, southern Asia, and eastern Europe--also affects growth
- Particularism, making judgments and taking action based on narrow or personal features, rejects the concept of shared global culture
- Transborder data flow is the movement of information across international boundaries in any form
- National laws and traditions create disparate accounting practices in various countries, impacting how profits and losses are analyzed

CHAPTER 2

SYSTEM ANALYSIS AND DESIGN

System Analysis and Design

Term system is derived from the Greek word _Systema‘ which means an organized relationship among functioning units or components.

A system is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective.

Characteristics of a System

- Organization
- Interaction
- Interdependence
- Integration
- Central Objective

Elements of a System

- Outputs and Inputs
- Processor
- Control
- Feedback
- Environment
- Boundaries and Interface

Types of System

Physical – These are tangible entities that may be static or dynamic in operation. For example- parts of a computer center are the desks, chairs etc. that facilitate operation of the computer. They are static and a programmed computer is dynamic.

Abstract System – These are conceptual or non physical entities. For example- the abstract conceptualization of physical situations. A model is a representation of a real or planned system. A model is used to visualize relationships.

Deterministic System – It operates in a predictable manner and the interaction between parts is known with certainty. For example: Two molecules of hydrogen and one molecule of oxygen make water.

Probabilistic System – It shows probable behavior. The exact output is not known. For example: weather forecasting, mail delivery.

Social System- It is made up of people. For example: social clubs, societies

Human Machine System- When both human and machines are involved to perform a particular a particular task to achieve a target. For example: - Computer.

Machine System- Where human interference is neglected. All the tasks are performed by the machine.

Natural System- The system which is natural. For example- Solar system, Seasonal System.

Manufactured System- System made by man is called manufactured system. For example- Rockets, Dams, and Trains.

Permanent System- Which persists for long time. For example- policies of business.

Temporary System- Made for specified time and after that they are dissolved. For example- setting up DJ system.

Adaptive System- responds to change in the environment in such a way to improve their performance and to survive. For example- Human beings, animals.

Non Adaptive System-The system which doesn‘t respond to the environment. For example- Machines

Open System – It has many interfaces with its environment. It interacts across its boundaries, it receives inputs from and delivers outputs to the outside world. It must adapt to the changing demands of the user.

Closed System – It is isolated from the environmental influences. A completely closed system is rare.

CASE Tools

CASE tools stand for Computer Aided Software Engineering tools As the name implies they are computer based programs to increase the productivity of analysts They permit effective communication with users as well as other members of the development team. They integrate the development done during each phase of a system life cycle. They assist in correctly assessing the effects and cost of changes so that maintenance cost can be estimated.

Available CASE tools

- Commercially available systems provide tools for each phase of the system development life cycle. A typical package is Visual Analyst which has several tools integrated together.
- Tools are also in the open domain which can be downloaded and used. They do not usually have very good user interfaces.
- System requirements specification documentation tool
- Data flow diagramming tool
- System flow chart generation tool
- Data dictionary creation
- Formatting and checking structured English process logic
- Decision table checking

- Screen design for data inputting
- Form design for outputs.
- E-R diagramming
- Data base normalization given the dependency information

Uses

- Improve productivity of their software engineers
- Reduce time to develop applications
- Improve documentation
- Automate system analysis

Disadvantages

- Some tools are expensive
- All software engineers need to be trained to use these tools
- A lot of time is wasted in using the tools
- Software developed using CASE tools are of poor quality

Advantages

- they integrate the development done during each phase of system development
- they permit effective communication with users
- they are useful as communication aids with users of the system

System flowchart

System flowcharts are a way of displaying how data flows in a system and how decisions are made to control events.

To illustrate this, symbols are used. They are connected together to show what happens to data and where it goes. The basic ones include:

Symbols used in flow charts

Note that system flow charts are very similar to data flow charts. Data flow charts do not include decisions, they just show the path that data takes, where it is held, processed, and then output.

Using system flowchart ideas in this system flowchart is a diagram for a 'cruise control' for a car. The cruise control keeps the car at a steady speed that has been set by the driver.

The flowchart shows what the outcome is if the car is going too fast or too slow. The system is designed to add fuel, or take it away and so keep the car's speed constant. The output (the car's new speed) is then fed back into the system via the speed sensor.

Other examples of uses for system diagrams include:

- aircraft control
- central heating

Input and output

For the system to work there is an input and an output. The process is taking the input and doing something with it - modifying it in some way - and producing an output.

In a computer system the processing will be done by a microprocessor of some kind.

Feedback is the output fed back to the input. The cruise control flowchart is an example of negative feedback because the speed is always kept at the same value. Positive feedback would push the speed away from the desired value.

Examples of inputs

- keyboard
- mouse
- microphone
- scanner
- camera
- pressure sensor

Examples of outputs

- printers
- speakers
- motors
- monitors
- heaters
- electromagnets
- bulbs/LEDs

Decision Tables

A decision table is a good way to deal with combinations of things (e.g. inputs). This technique is sometimes also referred to as a ‘cause-effect‘ table. The reason for this is that there is an associated logic diagramming technique called ‘cause-effect graphing‘ which was sometimes used to help derive the decision table.

- Decision tables provide a systematic way of stating complex business rules, which is useful for developers as well as for testers.
- Decision tables can be used in test design whether or not they are used in specifications, as they help testers explore the effects of combinations of different inputs and other software states that must correctly implement business rules.
- It helps the developers to do a better job can also lead to better relationships with them. Testing combinations can be a challenge, as the number of combinations can often be huge.

If you do not have a systematic way of selecting combinations, an arbitrary subset will be used and this may well result in an ineffective test effort.

Three parts

- Condition rows (stubs)
  - Lists condition relevant to decision
- Action rows (stubs)
  - Actions that result from a given set of conditions
- Rules
  - Specify which actions are to be followed for a given set of conditions

Uses of decision tables

- Powerful visualisation
- Compact and structured presentation
- Preventing errors is easier
- Avoid incompleteness and inconsistency
- Modular knowledge organisation
- Group related rules into single table
- Combine tables to achieve decision

Decision Table Methodology

1. Identify Conditions & Values:
   Find the data attribute each condition tests and all of the attribute's values.

2. Identify Possible Actions:
   Determine each independent action to be taken for the decision or policy.

3. Compute Max Number of Rules:
   Multiply the number of values for each condition data attribute by each other.

4. Enter All Possible Rules:
   Fill in the values of the condition data attributes in each numbered rule column.

5. Define Actions for each Rule:
   For each rule, mark the appropriate actions with an X in the decision table.

6. Verify the Policy Review completed decision table with end-users.
7. Simplify the Table Eliminate and/or consolidate rules to reduce the number of columns.

TABLE Empty decision table

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:				
Term of loan has been Entered:				

Next we will identify all of the combinations of True and False. With two conditions, each of which can be true or false, we will have four combinations (two to the power of the number of things to be combined). Note that if we have three things to combine, we will have eight combinations, with four things, there are 16, etc. This is why it is good to tackle small sets of combinations at a time. In order to keep track of which combinations we have, we will alternate True and False on the bottom row, put two True‘s and then two Falses on the row above the bottom row, etc., so the top row will have all True‘s and then all Falses (and this principle applies to all such tables).

TABLE Decision table with input combinations:

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been entered:	T	F	T	F

In the next step we will now identify the correct outcome for each combination. In this example, we can enter one or both of the two fields. Each combination is sometimes referred to as a rule.

TABLE Decision table with combinations and outcomes:

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been entered:	T	F	T	F
Actions/Outcomes				
Process loan amount:	Y	Y		
Process term:	Y		Y	

We could assume that this combination should result in an error message, so we need to add another action. This highlights the strength of this technique to discover omissions and ambiguities in specifications. It is not unusual for some combinations to be omitted from specifications; therefore this is also a valuable technique to use when reviewing the test basis.

TABLE Decision table with additional outcomes:

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been entered:	T	F	T	F
Actions/Outcomes				
Process loan amount:	Y	Y		
Process term:	Y		Y	
Error message:				Y

Now, we make slight change in this example, so that the customer is not allowed to enter both repayment and term. Now the outcome of our table will change, because there should also be an error message if both are entered.

TABLE Decision table with changed outcomes:

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been entered:	T	F	T	F
Actions/Outcomes				
Process loan amount:		Y		
Process term:			Y	
Error message:	Y			Y

You might notice now that there is only one _Yes‘ in each column, i.e. our actions are mutually exclusive – only one action occurs for each combination of conditions. We could represent this in a different way by listing the actions in the cell of one row.

TABLE Decision table with outcomes in one row:

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been entered:	T	F	T	F
Actions/Outcomes:				
Result:	Error message	Process loan amount	Process term	Error message

Data Flow Diagram (DFD)

The Data Flow Diagram (DFD) is a graphical representation of the flow of data through an information system. It enables you to represent the processes in your information system from the viewpoint of data. The DFD lets you visualize how the system operates, what the system accomplishes and how it will be implemented, when it is refined with further specification.

Data flow diagrams are used by systems analysts to design information-processing systems but also as a way to model whole organizations. You build a DFD at the very beginning of your business process modeling in order to model the functions your system has to carry out and the interaction between those functions together with focusing on data exchanges between processes. You can associate data with conceptual, logical, and physical data models and object-oriented models.

There are two types of DFDs, both of which support a top-down approach to systems analysis, whereby analysts begin by developing a general understanding of the system and gradually break components out into greater detail:

- Logical data flow diagrams - are implementation-independent and describe the system, rather than how activities are accomplished.
- Physical data flow diagrams - are implementation-dependent and describe the actual entities (devices, department, people, etc.) involved in the current system.

DFDs can also be grouped together to represent a sub-system of the system being analyzed.

Power Designer support for DFD

- Support for the Gane & Sarson and Yourdon notations, which you choose between by selecting Tools > Model Options.

- Data Flow Diagram specific validation rules (F4) – Power Designer may perform automatic corrections to your model or output errors and warnings that you will have to correct manually.

Entity Relationship

In software engineering, an entity–relationship model (ER model) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities (things) and the relationships that can exist among them, and databases.

Entity–Relationship modeling

Two related entities

- An entity with an attribute
- A relationship with an attribute

Primary key

An entity may be defined as a thing capable of an independent existence that can be uniquely identified. An entity is an abstraction from the complexities of a domain. When we speak of an entity, we normally speak of some aspect of the real world that can be distinguished from other aspects of the real world.

An entity is a thing that exists either physically or logically. An entity may be a physical object such as a house or a car (they exist physically), an event such as a house sale or a car service, or a concept such as a customer transaction or order (they exist logically as a concept). Although the term entity is the one most commonly used, following Chen we should really distinguish between an entity and an entity-type. An entity-type is a category. An entity, strictly speaking, is an instance of a given entity-type. There are usually many instances of an entity-type. Because the term entity-type is somewhat cumbersome, most people tend to use the term entity as a synonym for this term.

Entities can be thought of as nouns. Examples: a computer, an employee, a song, a mathematical theorem.

A relationship captures how entities are related to one another. Relationships can be thought of as verbs, linking two or more nouns. Examples: an owns relationship between a company and a computer, a supervises relationship between an employee and a department, a performs relationship between an artist and a song, a proved relationship between a mathematician and a theorem.

Entities and relationships can both have attributes. Examples: an employee entity might have a Social Security Number (SSN) attribute; the proved relationship may have a date attribute.

Every entity (unless it is a weak entity) must have a minimal set of uniquely identifying attributes, which is called the entity's primary key.

Entity–relationship diagrams don't show single entities or single instances of relations. Rather, they show entity sets (all entities of the same entity type) and relationship sets (all relationships of the same relationship type). Example: a particular song is an entity. The collection of all songs in a database is an entity set. The eaten relationship between a child and her lunch is a single relationship. The set of all such child-lunch relationships in a database is a relationship set. In other words, a relationship set corresponds to a relation in mathematics, while a relationship corresponds to a member of the relation.

Object Oriented Analysis and Design (OOAD)

Object-oriented analysis and design (OOAD) is a popular technical approach to analyzing, designing an application, system, or business by applying the object-oriented paradigm and visual modeling throughout the development life cycles to foster better stakeholder communication and product quality.

According to the popular guide Unified Process, OOAD in modern software engineering is best conducted in an iterative and incremental way. Iteration by iteration, the outputs of OOAD activities, analysis models for OOA and design models for OOD respectively, will be refined and evolve continuously driven by key factors like risks and business value

The software life cycle is typically divided up into stages going from abstract descriptions of the problem to designs then to code and testing and finally to deployment. The earliest stages of this process are analysis and design. The analysis phase is also often called "requirements acquisition".

The Waterfall Model

OOAD is conducted in an iterative and incremental manner, as formulated by the Unified Process.

In some approaches to software development known collectively as waterfall models the boundaries between each stage are meant to be fairly rigid and sequential. The term "waterfall" was coined for such methodologies to signify that progress went sequentially in one direction only, i.e., once analysis was complete then and only then was design begun and it was rare (and considered a source of error) when a design issue required a change in the analysis model or when a coding issue required a change in design.

The alternative to waterfall models is iterative models. This distinction was popularized by Barry Boehm in a very influential paper on his Spiral Model for iterative software development. With

iterative models it is possible to do work in various stages of the model in parallel. So for example it is possible and not seen as a source of error to work on analysis, design, and even code all on the same day and to have issues from one stage impact issues from another. The emphasis on iterative models is that software development is a knowledge-intensive process and that things like analysis can't really be completely understood without understanding design issues, that coding issues can affect design, that testing can yield information about how the code or even the design should be modified, etc.

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.

The object-oriented paradigm emphasizes modularity and re-usability. The goal of an object-oriented approach is to satisfy the "open closed principle". A module is open if it supports extension. If the module provides standardized ways to add new behaviors or describe new states. In the object-oriented paradigm this is often accomplished by creating a new subclass of an existing class. A module is closed if it has a well defined stable interface that all other modules must use and that limits the interaction and potential errors that can be introduced into one module by changes in another. In the object-oriented paradigm this is accomplished by defining methods that invoke services on objects. Methods can be either public or private, i.e., certain behaviors that are unique to the object are not exposed to other objects. This reduces a source of many common errors in computer programming.

The software life cycle is typically divided up into stages going from abstract descriptions of the problem to designs then to code and testing and finally to deployment. The earliest stages of this process are analysis and design. The distinction between analysis and design is often described as "what vs. how". In analysis developer‘s work with users and domain experts to define what the system is supposed to do. Implementation details are supposed to be mostly or totally (depending on the particular method) ignored at this phase. The goal of the analysis phase is to create a functional model of the system regardless of constraints such as appropriate technology. In object-oriented analysis this is typically done via use cases and abstract definitions of the most important objects. The subsequent design phase refines the analysis model and makes the needed technology and other implementation choices. In object-oriented design the emphasis is on describing the various objects, their data, behavior, and interactions. The design model should have all the details required so that programmers can implement the design in code.

Object-oriented analysis

The purpose of any analysis activity in the software life-cycle is to create a model of the system's functional requirements that is independent of implementation constraints.

The main difference between object-oriented analysis and other forms of analysis is that by the object-oriented approach we organize requirements around objects, which integrate both behaviors (processes) and states (data) modeled after real world objects that the system interacts with. In other or

traditional analysis methodologies, the two aspects: processes and data are considered separately. For example, data may be modeled by ER diagrams, and behaviors by flow charts or structure charts.

The primary tasks in object-oriented analysis (OOA) are:

- Find the objects
- Organize the objects
- Describe how the objects interact
- Define the behavior of the objects
- Define the internals of the objects

Common models used in OOA are use cases and object models. Use cases describe scenarios for standard domain functions that the system must accomplish. Object models describe the names, class relations (e.g. Circle is a subclass of Shape), operations, and properties of the main objects. User-interface mockups or prototypes can also be created to help understanding.

Object-oriented design

During object-oriented design (OOD), a developer applies implementation constraints to the conceptual model produced in object-oriented analysis. Such constraints could include the hardware and software platforms, the performance requirements, persistent storage and transaction, usability of the system, and limitations imposed by budgets and time. Concepts in the analysis model which is technology independent are mapped onto implementing classes and interfaces resulting in a model of the solution domain, i.e., a detailed description of how the system is to be built on concrete technologies.

Important topics during OOD also include the design of software architectures by applying architectural patterns and design patterns with object-oriented design principles.

Object-oriented modeling

Object-oriented modeling (OOM) is a common approach to modeling applications, systems, and business domains by using the object-oriented paradigm throughout the entire development life cycles. OOM is a main technique heavily used by both OOA and OOD activities in modern software engineering.

Object-oriented modeling typically divides into two aspects of work: the modeling of dynamic behaviors like business processes and use cases, and the modeling of static structures like classes and components. OOA and OOD are the two distinct abstract levels (i.e. the analysis level and the design level) during OOM. The Unified Modeling Language (UML) and Sys ML are the two popular international standard languages used for object-oriented modeling.

The benefits of OOM are:

- Efficient and effective communication

Users typically have difficulties in understanding comprehensive documents and programming language codes well. Visual model diagrams can be more understandable and can allow users and stakeholders to give developers feedback on the appropriate requirements and structure of the system. A key goal of the object-oriented approach is to decrease the "semantic gap" between the system and the real world, and to have the system be constructed using terminology that is almost the same as the stakeholders use in everyday business. Object-oriented modeling is an essential tool to facilitate this.

- Useful and stable abstraction

Modeling helps coding. A goal of most modern software methodologies is to first address "what" questions and then address "how" questions, i.e. first determine the functionality the system is to provide without consideration of implementation constraints, and then consider how to make specific solutions to these abstract requirements, and refine them into detailed designs and codes by constraints such as technology and budget. Object-oriented modeling enables this by producing abstract and accessible descriptions of both system requirements and designs, i.e. models that define their essential structures and behaviors like processes and objects, which are important and valuable development assets with higher abstraction levels above concrete and complex source code.

UML Diagrams

The elements are like components which can be associated in different ways to make complete UML pictures which is known as diagram. So it is very important to understand the different diagrams to implement the knowledge in real life systems.

Any complex system is best understood by making some kind of diagrams or pictures. These diagrams have a better impact on our understanding. So if we look around then we will realize that the diagrams are not a new concept but it is used widely in different form in different industries.

We prepare UML diagrams to understand a system in better and simple way. A single diagram is not enough to cover all aspects of the system. So UML defines various kinds of diagrams to cover most of the aspects of a system.

You can also create your own set of diagrams to meet your requirements. Diagrams are generally made in an incremental and iterative way.

There are two broad categories of diagrams and then are again divided into sub-categories:

- Structural Diagrams
- Behavioral Diagrams

Structural Diagrams

The structural diagrams represent the static aspect of the system. These static aspects represent those parts of a diagram which forms the main structure and therefore stable.

These static parts are represents by classes, interfaces, objects, components and nodes. The four structural diagrams are:

- Class diagram
- Object diagram
- Component diagram
- Deployment diagram

Class Diagram

Class diagrams are the most common diagrams used in UML. Class diagram consists of classes, interfaces, associations and collaboration. Class diagrams basically represent the object oriented view of a system which is static in nature. Active class is used in a class diagram to represent the concurrency of the system. Class diagram represents the object orientation of a system. So it is generally used for development purpose. This is the most widely used diagram at the time of system construction.

Object Diagram

Object diagrams can be described as an instance of class diagram. So these diagrams are more close to real life scenarios where we implement a system. Object diagrams are a set of objects and their relationships just like class diagrams and also represent the static view of the system. The usage of object diagrams is similar to class diagrams but they are used to build prototype of a system from practical perspective.

Component Diagram

Component diagrams represent a set of components and their relationships. These components consist of classes, interfaces or collaborations. So Component diagrams represent the implementation view of a system. During design phase software artifacts (classes, interfaces etc) of a system are arranged in different groups depending upon their relationship. Now these groups are known as components. Finally, component diagrams are used to visualize the implementation.

Deployment Diagram

Deployment diagrams are a set of nodes and their relationships. These nodes are physical entities where the components are deployed. Deployment diagrams are used for visualizing deployment view of a system. This is generally used by the deployment team.

Behavioral Diagrams

Any system can have two aspects, static and dynamic. So a model is considered as complete when both the aspects are covered fully. Behavioral diagrams basically capture the dynamic aspect of a system. Dynamic aspect can be further described as the changing/moving parts of a system.

UML has the following five types of behavioral diagrams:

- Use case diagram
- Sequence diagram
- Collaboration diagram
- State chart diagram
- Activity diagram

Use case Diagram

Use case diagrams are a set of use cases, actors and their relationships. They represent the use case view of a system. A use case represents a particular functionality of a system. So use case diagram is used to describe the relationships among the functionalities and their internal/external controllers. These controllers are known as actors.

Sequence Diagram

A sequence diagram is an interaction diagram. From the name it is clear that the diagram deals with some sequences, which are the sequence of messages flowing from one object to another. Interaction among the components of a system is very important from implementation and execution perspective. So Sequence diagram is used to visualize the sequence of calls in a system to perform a specific functionality.

Collaboration Diagram

Collaboration diagram is another form of interaction diagram. It represents the structural organization of a system and the messages sent/received. Structural organization consists of objects and links. The purpose of collaboration diagram is similar to sequence diagram. But the specific purpose of collaboration diagram is to visualize the organization of objects and their interaction.

State chart Diagram

Any real time system is expected to be reacted by some kind of internal/external events. These events are responsible for state change of the system. State chart diagram is used to represent the event driven state change of a system. It basically describes the state change of a class, interface etc. State chart diagram is used to visualize the reaction of a system by internal/external factors.

Activity Diagram

Activity diagram describes the flow of control in a system. So it consists of activities and links. The flow can be sequential, concurrent or branched. Activities are nothing but the functions of a system. Numbers of activity diagrams are prepared to capture the entire flow in a system. Activity diagrams are used to visualize the flow of controls in a system. This is prepared to have an idea of how the system will work when executed.

CHAPTER -3

DATABASE MANAGEMENT SYSTEMS

DBMS- Data Base Management System

Database is collection of data which is related by some aspect. Data is collection of facts and figures which can be processed to produce information. Name of a student, age, class and her subjects can be counted as data for recording purposes.

Mostly data represents recordable facts. Data aids in producing information which is based on facts. For example, if we have data about marks obtained by all students, we can then conclude about toppers and average marks etc.

A database management system stores data, in such a way which is easier to retrieve, manipulate and helps to produce information.

Characteristics

Traditionally data was organized in file formats. DBMS was all new concepts then and all the research was done to make it to overcome all the deficiencies in traditional style of data management. Modern DBMS has the following characteristics:

- Real-world entity: Modern DBMS are more realistic and uses real world entities to design its architecture. It uses the behavior and attributes too. For example, a school database may use student as entity and their age as their attribute.

- Relation-based tables: DBMS allows entities and relations among them to form as tables. This eases the concept of data saving. A user can understand the architecture of database just by looking at table names etc.

- Isolation of data and application: A database system is entirely different than its data. Where database is said to active entity, data is said to be passive one on which the database works and organizes. DBMS also stores metadata which is data about data, to ease its own process.

- Less redundancy: DBMS follows rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Following normalization, which itself is a mathematically rich and scientific process, make the entire database to contain as less redundancy as possible.

- Consistency: DBMS always enjoy the state on consistency where the previous form of data storing applications like file processing does not guarantee this. Consistency is a state where every relation in database remains consistent. There exist methods and techniques, which can detect attempt of leaving database in inconsistent state.

- Query Language: DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and different filtering options, as he or she wants. Traditionally it was not possible where file-processing system was used.

- ACID Properties: DBMS follows the concepts for ACID properties, which stands for Atomicity, Consistency, Isolation and Durability. These concepts are applied on transactions, which manipulate data in database. ACID properties maintains database in healthy state in multi-transactional environment and in case of failure.

- Multiuser and Concurrent Access: DBMS support multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when they attempt to handle same data item, but users are always unaware of them.

- Multiple views: DBMS offers multiples views for different users. A user who is in sales department will have a different view of database than a person working in production department. This enables user to have a concentrate view of database according to their requirements.

- Security: Features like multiple views offers security at some extent where users are unable to access data of other users and departments. DBMS offers methods to impose constraints while entering data into database and retrieving data at later stage. DBMS offers many different levels of security features, which enables multiple users to have different view with different features. For example, a user in sales department cannot see data of purchase department is one thing, additionally how much data of sales department he can see, can also be managed. Because DBMS is not saved on disk as traditional file system it is very hard for a thief to break the code.

Users

DBMS is used by various users for various purposes. Some may involve in retrieving data and some may involve in backing it up. Some of them are described as follows:

- Administrators: A bunch of users maintain the DBMS and are responsible for administrating the database. They are responsible to look after its usage and by whom it should be used. They create users access and apply limitation to maintain isolation and force security. Administrators also look after DBMS resources like system license, software application and tools required and other hardware related maintenance.

- Designer: This is the group of people who actually works on designing part of database. The actual database is started with requirement analysis followed by a good designing process. They people keep a close watch on what data should be kept and in what format. They identify and design the whole set of entities, relations, constraints and views.

- End Users: This group contains the persons who actually take advantage of database system. End users can be just viewers who pay attention to the logs or market rates or end users can be as sophisticated as business a analyst who takes the most of it.

DBMS - Architecture

The design of a Database Management System highly depends on its architecture. It can be centralized or decentralized or hierarchical. DBMS architecture can be seen as single tier or multi tier. n-tier architecture divides the whole system into related but independent n modules, which can be independently modified, altered, changed or replaced.

In 1-tier architecture, DBMS is the only entity where user directly sits on DBMS and uses it. Any changes done here will directly be done on DBMS itself. It does not provide handy tools for end users and preferably database designer and programmers use single tier architecture.

If the architecture of DBMS is 2-tier then must have some application, which uses the DBMS. Programmers use 2-tier architecture where they access DBMS by means of application. Here application tier is entirely independent of database in term of operation, design and programming.

3-tier architecture

Most widely used architecture is 3-tier architecture. 3-tier architecture separates it tier from each other on basis of users. It is described as follows:

- Database (Data) Tier: At this tier, only database resides. Database along with its query processing languages sits in layer-3 of 3-tier architecture. It also contains all relations and their constraints.

- Application (Middle) Tier: At this tier the application server and program, which access database, resides. For a user this application tier works as abstracted view of database. Users are unaware of any existence of database beyond application. For database-tier, application tier is the user of it. Database tier is not aware of any other user beyond application tier. This tier works as mediator between the two.

- User (Presentation) Tier: An end user sits on this tier. From a users aspect this tier is everything. He/she doesn't know about any existence or form of database beyond this layer. At this layer multiple views of database can be provided by the application. All views are generated by applications, which reside in application tier.

Multiple tier database architecture is highly modifiable as almost all its components are independent and can be changed independently.

DBMS - Data Models

Data model tells how the logical structure of a database is modeled. Data Models are fundamental entities to introduce abstraction in DBMS. Data models define how data is connected to each other and how it will be processed and stored inside the system.

The very first data model could be flat data-models where all the data used to be kept in same plane. Because earlier data models were not so scientific they were prone to introduce lots of duplication and update anomalies.

Entity-Relationship Model

Entity-Relationship model is based on the notion of real world entities and relationship among them. While formulating real-world scenario into database model, ER Model creates entity set, relationship set, general attributes and constraints. ER Model is best used for the conceptual design of database.

ER Model is based on:

- Entities and their attributes
- Relationships among entities
- Entity

  An entity in ER Model is real world entity, which has some properties called attributes. Every attribute is defined by its set of values, called domain.

  For example, in a school database, a student is considered as an entity. Student has various attributes like name, age and class etc.

- Relationship

  The logical association among entities is called relationship. Relationships are mapped with entities in various ways. Mapping cardinalities define the number of association between two entities.

  Mapping cardinalities:

  - one to one
  - one to many
  - many to one
  - many to many

Relational Model

The most popular data model in DBMS is Relational Model. It is more scientific model then others. This model is based on first-order predicate logic and defines table as an n-ary relation.

The main highlights of this model are:

- Data is stored in tables called relations.
- Relations can be normalized.
- In normalized relations, values saved are atomic values.
- Each row in relation contains unique value
- Each column in relation contains values from a same domain.

DBMS - Data Schemas

Database schema

Database schema skeleton structure of and it represents the logical view of entire database. It tells about how the data is organized and how relation among them is associated. It formulates all database constraints that would be put on data in relations, which resides in database.

A database schema defines its entities and the relationship among them. Database schema is a descriptive detail of the database, which can be depicted by means of schema diagrams. All these activities are done by database designer to help programmers in order to give some ease of understanding all aspect of database.

Database schema can be divided broadly in two categories:

- Physical Database Schema: This schema pertains to the actual storage of data and its form of storage like files, indices etc. It defines the how data will be stored in secondary storage etc.

- Logical Database Schema: This defines all logical constraints that need to be applied on data stored. It defines tables, views and integrity constraints etc.

Database Instance

It is important that we distinguish these two terms individually. Database schema is the skeleton of database. It is designed when database doesn't exist at all and very hard to do any changes once the database is operational. Database schema does not contain any data or information.

Database instances, is a state of operational database with data at any given time. This is a snapshot of database. Database instances tend to change with time. DBMS ensures that its every instance (state)

must be a valid state by keeping up to all validation, constraints and condition that database designers has imposed or it is expected from DBMS itself.

HDBMS Hierarchical Database Management System

A hierarchical database model is a data model in which the data is organized into a tree-like structure. The data is stored as records which are connected to one another through links. A record is a collection of fields, with each field containing only one value. The entity type of a record defines which fields the record contains.

Example of a hierarchical model

A record in the hierarchical database model corresponds to a row in the relational database model and an entity type corresponds to a table.

The hierarchical database model mandates that each child record has only one parent, whereas each parent record can have one or more child records. In order to retrieve data from a hierarchical database the whole tree needs to be traversed starting from the root node. This model is recognized as the first database model created by IBM in the 1960s.

The Hierarchical Data Model is a way of organizing a database with multiple one to many relationships. The structure is based on the rule that one parent can have many children but children are allowed only one parent. This structure allows information to be repeated through the parent child relations created by IBM and was implemented mainly in their Information Management System.

Advantages

The model allows easy addition and deletion of new information. Data at the top of the Hierarchy is very fast to access. It was very easy to work with the model because it worked well with linear type data storage such as tapes. The model relates very well to natural hierarchies such as assembly plants and employee organization in corporations. It relates well to anything that works through a one to many relationships. For example; there is a president with many managers below them, and those managers have many employees below them, but each employee has only one manager.

Disadvantages

This model has many issues that hold it back now that we require more sophisticated relationships. It requires data to be repetitively stored in many different entities. The database can be very slow when searching for information on the lower entities. We no longer use linear data storage mediums such as tapes so that advantage is null. Searching for data requires the DBMS to run through the entire model from top to bottom until the required information is found, making queries very slow. Can only model one to many relationships, many to many relationships are not supported. Clever manipulation of the model is required to make many to may relationships.

NDBMS-Network Database Management System

Network Database: A network databases are mainly used on large digital computers. It more connections can be made between different types of data, network databases are considered more efficiency It contains limitations must be considered when we have to use this kind of database. It is Similar to the hierarchical databases; network databases.

Network databases are similar to hierarchical databases by also having a hierarchical structure. A network database looks more like a cobweb or interconnected network of records.

In network databases, children are called members and parents are called occupier. The difference between each child or member can have more than one parent. The Approval of the network data model similar with the esteem of the hierarchical data model. Some data were more naturally modeled with more than one parent per child. The network model authorized the modeling of many-to-many relationships in data.

The network model is very similar to the hierarchical model really. Actually the hierarchical model is a subset of the network model. However, instead of using a single-parent tree hierarchy, the network model uses set theory to provide a tree-like hierarchy with the exception that child tables were allowed to have more than one parent. It supports many-to-many relationships.

RDBMS-Relational Database Management System

In relational databases, the relationship between data files is relational. Hierarchical and network databases require the user to pass a hierarchy in order to access needed data. These databases connect to the data in different files by using common data numbers or a key field. Data in relational databases is stored in different access control tables, each having a key field that mainly identifies each row. In the relational databases are more reliable than either the hierarchical or network database structures. In relational databases, tables or files filled up with data are called relations designates a row or record, and columns are referred to as attributes or fields.

Relational databases work on each table has a key field that uniquely indicates each row, and that these key fields can be used to connect one table of data to another.

The relational database has two major reasons

1. Relational databases can be used with little or no training.
2. Database entries can be modified without specify the entire body.

Properties of Relational Tables

In the relational database we have to follow some properties which are given below.

- It's Values are Atomic
- In Each Row is alone.

- Column Values are of the same thing.
- Columns are undistinguished.
- Sequence of Rows is Insignificant.
- Each Column has a common Name.

OODBMS – Object oriented Database Management System

In this Model we have to discuss the functionality of the object oriented Programming .It takes more than storage of programming language objects. Object DBMS's increase the semantics of the C++ and Java .It provides full-featured database programming capability, while containing native language compatibility. It adds the database functionality to object programming languages. This approach is the analogical of the application and database development into a constant data model and language environment. Applications require less code, use more natural data modeling, and code bases are easier to maintain. Object developers can write complete database applications with a decent amount of additional effort.

The object-oriented database derivation is the integrity of object-oriented programming language systems and consistent systems. The power of the object-oriented databases comes from the cyclical treatment of both consistent data, as found in databases, and transient data, as found in executing programs.
Object-oriented databases use small, recyclable separated of software called objects. The objects themselves are stored in the object-oriented database. Each object contains of two elements:

1. Piece of data (e.g., sound, video, text, or graphics).
2. Instructions or software programs called methods, for what to do with the data.

Disadvantage of Object-oriented databases

1. Object-oriented databases have these disadvantages.
2. Object-oriented database are more expensive to develop.
3. In the Most organizations are unwilling to abandon and convert from those databases.

The benefits to object-oriented databases are compelling. The ability to mix and match reusable objects provides incredible multimedia capability.

Query Processing

Upper levels of the data integration problem

- How to construct mappings from sources to a single mediated schema
- How queries posed over the mediated schema are reformulated over the sources

Basic Steps in Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation

Parsing and translation

Translate the query into its internal form. This is then translated into relational algebra. Parser checks syntax, verifies relations.

Evaluation

The query-execution engine takes a query-evaluation plan, executes that plan, and returns the answers to the query.A relational algebra expression may have many equivalent expressions. Each relational algebra operation can be evaluated using one of several different algorithms. Correspondingly, a relational-algebra expression can be evaluated in many ways. Annotated expression specifying detailed evaluation strategy is called an evaluation-plan.

Query Optimization

Amongst all equivalent evaluation plans choose the one with lowest cost. Cost is estimated using statistical information from the database catalog.

SQL

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in relational database.

SQL is the standard language for Relation Database System. All relational database management systems like MySQL, MS Access, and Oracle, Sybase, Informix, postgres and SQL Server use SQL as standard database language.

Why SQL?

- Allows users to access data in relational database management systems.
- Allows users to describe the data.
- Allows users to define the data in database and manipulate that data.
- Allows to embed within other languages using SQL modules, libraries & pre-compilers.
- Allows users to create and drop databases and tables.
- Allows users to create view, stored procedure, functions in a database.
- Allows users to set permissions on tables, procedures, and views

History

- 1970 -- Dr. Edgar F. "Ted" Codd of IBM is known as the father of relational databases. He described a relational model for databases.
- 1974 -- Structured Query Language appeared.
- 1978 -- IBM worked to develop Codd's ideas and released a product named System/R.
- 1986 -- IBM developed the first prototype of relational database and standardized by ANSI. The first relational database was released by Relational Software and its later becoming Oracle.

SQL Process

When you are executing an SQL command for any RDBMS, the system determines the best way to carry out your request and SQL engine figures out how to interpret the task.

There are various components included in the process. These components are Query Dispatcher, Optimization Engines, Classic Query Engine and SQL Query Engine, etc. Classic query engine handles all non-SQL queries but SQL query engine won't handle logical files.

SQL Commands

The standard SQL commands to interact with relational databases are CREATE, SELECT, INSERT, UPDATE, DELETE and DROP. These commands can be classified into groups based on their nature.

DDL - Data Definition Language

Command	Description
CREATE	Creates a new table, a view of a table, or other object in database
ALTER	Modifies an existing database object, such as a table.
DROP	Deletes an entire table, a view of a table or other object in the database.

DML - Data Manipulation Language

Command	Description
SELECT	Retrieves certain records from one or more tables
INSERT	Creates a record
UPDATE	Modifies records
DELETE	Deletes records

DCL - Data Control Language

Command	Description
GRANT	Gives a privilege to user
REVOKE	Takes back privileges granted from user

Concurrency Management

In a multiprogramming environment where more than one transactions can be concurrently executed, there exists a need of protocols to control the concurrency of transaction to ensure atomicity and isolation properties of transactions.

Concurrency control protocols, which ensure serializability of transactions, are most desirable. Concurrency control protocols can be broadly divided into two categories:

- Lock based protocols
- Time stamp based protocols

Lock based protocols

Database systems, which are equipped with lock-based protocols, use mechanism by which any transaction cannot read or write data until it acquires appropriate lock on it first. Locks are of two kinds:

- Binary Locks: a lock on data item can be in two states; it is either locked or unlocked.

- Shared/exclusive: this type of locking mechanism differentiates lock based on their uses. If a lock is acquired on a data item to perform a write operation, it is exclusive lock. Because allowing more than one transactions to write on same data item would lead the database into an inconsistent state. Read locks are shared because no data value is being changed.

Types lock protocols

- Simplistic

  Simplistic lock based protocols allow transaction to obtain lock on every object before 'write' operation is performed. As soon as 'write' has been done, transactions may unlock the data item.

- Pre-claiming

  In this protocol, a transactions evaluations its operations and creates a list of data items on which it needs locks. Before starting the execution, transaction requests the system for all locks it needs beforehand. If all the locks are granted, the transaction executes and releases all the locks when all its operations are over. Else if all the locks are not granted, the transaction rolls back and waits until all locks are granted.

- Two Phase Locking - 2PL

  This locking protocol is divides transaction execution phase into three parts. In the first part, when transaction starts executing, transaction seeks grant for locks it needs as it executes. Second part is where the transaction acquires all locks and no other lock is required. Transaction keeps executing its operation. As soon as the transaction releases its first lock, the third phase starts. In this phase a transaction cannot demand for any lock but only releases the acquired locks.

  Two phase locking has two phases, one is growing; where all locks are being acquired by transaction and second one is shrinking, where locks held by the transaction are being released.To claim an exclusive (write) lock, a transaction must first acquire a shared (read) lock and then upgrade it to exclusive lock.

- Strict Two Phase Locking

  The first phase of Strict-2PL is same as 2PL. After acquiring all locks in the first phase, transaction continues to execute normally. But in contrast to 2PL, Strict-2PL does not release lock as soon as it is no more required, but it holds all locks until commit state arrive. Strict-2PL releases all locks at once at commit point.

Time stamp based protocols

The most commonly used concurrency protocol is time-stamp based protocol. This protocol uses either system time or logical counter to be used as a time-stamp.

Lock based protocols manage the order between conflicting pairs among transaction at the time of execution whereas time-stamp based protocols start working as soon as transaction is created.

Every transaction has a time-stamp associated with it and the ordering is determined by the age of the transaction. A transaction created at 0002 clock time would be older than all other transaction, which come after it. For example, any transaction 'y' entering the system at 0004 is two seconds younger and priority may be given to the older one.

In addition, every data item is given the latest read and write-timestamp. This lets the system know, when last read was and write operation made on the data item.

Time-stamp ordering protocol

The timestamp-ordering protocol ensures serializability among transaction in their conflicting read and writes operations. This is the responsibility of the protocol system that the conflicting pair of tasks should be executed according to the timestamp values of the transactions.

- Time-stamp of Transaction Ti is denoted as TS (T_i).
- Read time-stamp of data-item X is denoted by R-timestamp(X).
- Write time-stamp of data-item X is denoted by W-timestamp(X).

Timestamp ordering protocol works as follows:

- If a transaction Ti issues read(X) operation:
  - If TS(Ti) < W-timestamp(X)
    - Operation rejected.
  - If TS(Ti) >= W-timestamp(X)
    - Operation executed.
  - All data-item Timestamps updated.
- If a transaction Ti issues write(X) operation:
  - If TS(Ti) < R-timestamp(X)
    - Operation rejected.
    - If TS(Ti) < W-timestamp(X)
    - Operation rejected and Ti rolled back.
  - Otherwise, operation executed.

Data warehouse

- Data warehouse is data management and data analysis
- Goal: is to integrate enterprise wide corporate data into a single reository from which users can easily run queries

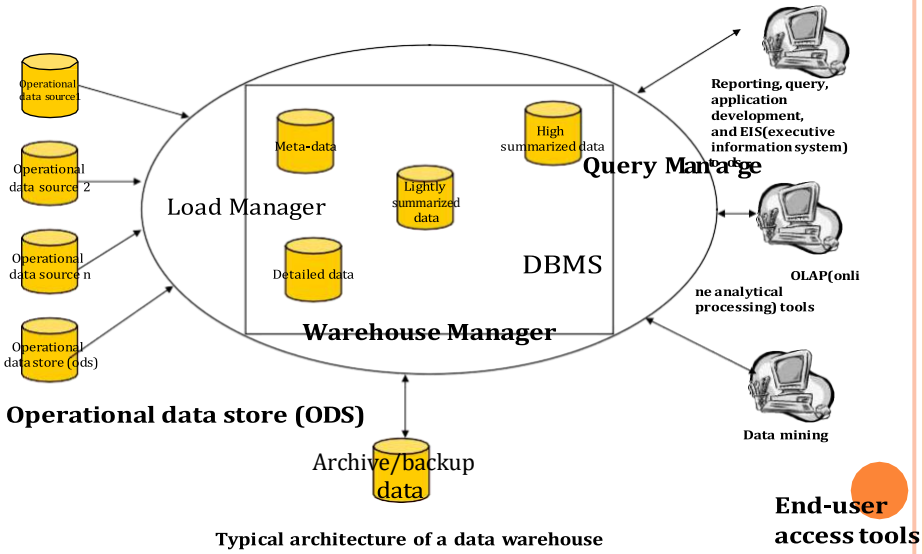
Benefits

- The major benefit of data warehousing are high returns on investment.
- Increased productivity of corporate decision-makers

Problems

- Underestimation of resources for data loading
- Hidden problems with source systems
- Required data not captured
- Increased end-user demands
- Data homogenization
- High demand for resources

ARCHITECTURE:



Typical architecture of a data warehouse

- Data ownership
- High maintenance
- Long-duration projects
- Complexity of integration

Main components

- Operational data sources→for the DW is supplied from mainframe operational data held in first generation hierarchical and network databases, departmental data held in proprietary file systems, private data held on workstaions and private serves and external systems such as the Internet, commercially available DB, or DB assoicated with and organization‘s suppliers or customers.
- Operational datastore(ODS)→is a repository of current and integrated operational data used for analysis. It is often structured and supplied with data in the same way as the data warehouse, but may in fact simply act as a staging area for data to be moved into the warehouse.
- query manager→also called backend component, it performs all the operations associated with the management of user queries. The operations performed by this component include directing queries to the appropriate tables and scheduling the execution of queries
- end-user access tools→can be categorized into five main groups: data reporting and query tools, application development tools, executive information system (EIS) tools, online analytical processing (OLAP) tools, and data mining tools.

Data flow

- Inflow- The processes associated with the extraction, cleansing, and loading of the data from the source systems into the data warehouse.
- upflow- The process associated with adding value to the data in the warehouse through summarizing, packaging , packaging, and distribution of the data.
- downflow- The processes associated with archiving and backing-up of data in the warehouse.

Tools and Technologies

The critical steps in the construction of a data warehouse:

- Extraction
- Cleansing
- Transformation

after the critical steps, loading the results into target system can be carried out either by separate products, or by a single, categories:

- code generators
- database data replication tools
- dynamic transformation engines

For the various types of meta-data and the day-to-day operations of the data warehouse, the administration and management tools must be capable of supporting those tasks:

- Monitoring data loading from multiple sources
- Data quality and integrity checks
- Managing and updating meta-data
- Monitoring database performance to ensure efficient query response times and resource utilization
- Auditing data warehouse usage to provide user chargeback information
- Replicating, subsetting, and distributing data
- Maintaining effient data storage management
- Purging data;
- Archiving and backing-up data
- Implementing recovery following failure

Data Mart

A data mart is a simple form of a data warehouse that is focused on a single subject (or functional area), such as sales, finance or marketing. Data marts are often built and controlled by a single department within an organization. Given their single-subject focus, data marts usually draw data from only a few sources. The sources could be internal operational systems, a central data warehouse, or external data.[1]

Dependent and Independent Data Marts

There are two basic types of data marts: dependent and independent. The categorization is based primarily on the data source that feeds the data mart. Dependent data marts draw data from a central data warehouse that has already been created. Independent data marts, in contrast, are standalone systems built by drawing data directly from operational or external sources of data, or both.

The main difference between independent and dependent data marts is how you populate the data mart; that is, how you get data out of the sources and into the data mart. This step, called the Extraction-Transformation-and Loading (ETL) process, involves moving data from operational systems, filtering it, and loading it into the data mart.With dependent data marts, this process is somewhat simplified because formatted and summarized (clean) data has already been loaded into the central data warehouse. The ETL process for dependent data marts is mostly a process of identifying the right subset of data relevant to the chosen data mart subject and moving a copy of it, perhaps in a summarized form.

With independent data marts, however, you must deal with all aspects of the ETL process, much as you do with a central data warehouse. The number of sources is likely to be fewer and the amount of data associated with the data mart is less than the warehouse, given your focus on a single subject.The motivations behind the creation of these two types of data marts are also typically different.

Steps in Implementing a Data Mart

Simply stated, the major steps in implementing a data mart are to design the schema, construct the physical storage, populate the data mart with data from source systems, access it to make informed decisions, and manage it over time.

- Designing
- Constructing
- Populating
- Accessing
- Managing

1. Designing

The design step is first in the data mart process. This step covers all of the tasks from initiating the request for a data mart through gathering information about the requirements, and developing the logical and physical design of the data mart. The design step involves the following tasks:

- Gathering the business and technical requirements
- Identifying data sources
- Selecting the appropriate subset of data
- Designing the logical and physical structure of the data mart

2. Constructing

This step includes creating the physical database and the logical structures associated with the data mart to provide fast and efficient access to the data. This step involves the following tasks:

- Creating the physical database and storage structures, such as tablespaces, associated with the data mart
- Creating the schema objects, such as tables and indexes defined in the design step
- Determining how best to set up the tables and the access structures
-

3. Populating

The populating step covers all of the tasks related to getting the data from the source, cleaning it up, modifying it to the right format and level of detail, and moving it into the data mart. More formally stated, the populating step involves the following tasks:

- Mapping data sources to target data structures
- Extracting data
- Cleansing and transforming the data
- Loading data into the data mart
- Creating and storing metadata

4. Accessing

The accessing step involves putting the data to use: querying the data, analyzing it, creating reports, charts, and graphs, and publishing these. Typically, the end user uses a graphical front-end tool to submit queries to the database and display the results of the queries. The accessing step requires that you perform the following tasks:

- Set up an intermediate layer for the front-end tool to use. This layer, the metalayer, translates database structures and object names into business terms, so that the end user can interact with the data mart using terms that relate to the business function.
- Maintain and manage these business interfaces.
- Set up and manage database structures, like summarized tables, that help queries submitted through the front-end tool execute quickly and efficiently.

5. Managing

This step involves managing the data mart over its lifetime. In this step, you perform management tasks such as the following:

- Providing secure access to the data
- Managing the growth of the data
- Optimizing the system for better performance
- Ensuring the availability of data even with system failures

Data Mart issues

- Data mart functionality→the capabilities of data marts have increased with the growth in their popularity
- Data mart size→the performance deteriorates as data marts grow in size, so need to reduce the size of data marts to gain improvements in performance
- Data mart load performance→two critical components: end-user response time and data loading performance→to increment DB updating so that only cells affected by the change are updated and not the entire MDDB structure.

CHAPTER-4

SECURITY, CONTROL AND REPORTING

Information security

Information security, sometimes shortened to Info Sec, is the practice of defending information from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction. It is a general term that can be used regardless of the form the data may take (electronic, physical, etc.)

IT security

Sometimes referred to as computer security, Information Technology Security is information security applied to technology (most often some form of computer system). It is worthwhile to note that a computer does not necessarily mean a home desktop. A computer is any device with a processor and some memory. Such devices can range from non-networked standalone devices as simple as calculators, to networked mobile computing devices such as smart phones and tablet computers. IT security specialists are almost always found in any major enterprise/establishment due to the nature and value of the data within larger businesses. They are responsible for keeping all of the technology within the company secure from malicious cyber attacks that often attempt to breach into critical private information or gain control of the internal systems.

Information assurance

The act of ensuring that data is not lost when critical issues arise. These issues include but are not limited to: natural disasters, computer/server malfunction, physical theft, or any other instance where data has the potential of being lost. Since most information is stored on computers in our modern era, information assurance is typically dealt with by IT security specialists. One of the most common methods of providing information assurance is to have an off-site backup of the data in case one of the mentioned issues arises.

Governments, military, corporations, financial institutions, hospitals and private businesses amass a great deal of confidential information about their employees, customers, products, research and financial status. Most of this information is now collected, processed and stored on electronic computers and transmitted across networks to other computers.

Should confidential information about a business' customers or finances or new product line fall into the hands of a competitor or a black hat hacker, a business and its customers could suffer widespread, irreparable financial loss, not to mention damage to the company's reputation. Protecting confidential information is a business requirement and in many cases also an ethical and legal requirement. A key concern for organizations is the derivation of the optimal amount to invest, from an economics perspective, on information security.

For the individual, information security has a significant effect on privacy, which is viewed very differently in different cultures.

The field of information security has grown and evolved significantly in recent years. There are many ways of gaining entry into the field as a career. It offers many areas for specialization including securing network(s) and allied infrastructure, securing applications and databases, security testing, information systems auditing, business continuity planning and digital forensics, etc.

Definition

1. "Preservation of confidentiality, integrity and availability of information.

2. "The protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to provide confidentiality, integrity, and availability."

3. "Ensures that only authorized users (confidentiality) have access to accurate and complete information (integrity) when required (availability)."

4. "Information Security is the process of protecting the intellectual property of an organization."

5. "Information security is a risk management discipline, whose job is to manage the cost of information risk to the business."

Basic principles

The CIA triad of confidentiality, integrity, and availability is at the heart of information security. (The members of the classic Info Sec triad -confidentiality, integrity and availability - are interchangeably referred to in the literature as security attributes properties, security goals, fundamental aspects, information criteria, critical information characteristics and basic building blocks.) There is continuous debate about extending this classic trio. Other principles such as Accountability have sometimes been proposed for addition it has been pointed out that issues such as Non-Repudiation do not fit well within the three core concepts, and as regulation of computer systems has increased (particularly amongst the Western nations) Legality is becoming a key consideration for practical security installations.

In 1992 and revised in 2002 the OECD's Guidelines for the Security of Information Systems and Networks proposed the nine generally accepted principles: Awareness, Responsibility, Response, Ethics, Democracy, Risk Assessment, Security Design and Implementation, Security Management, and Reassessment. Building upon those, in 2004 the NIST's Engineering Principles for Information Technology Security proposed 33 principles. From each of these derived guidelines and practices.

Integrity

In information security, data integrity means maintaining and assuring the accuracy and consistency of data over its entire life-cycle. This means that data cannot be modified in an unauthorized or undetected manner. This is not the same thing as referential integrity in databases, although it can be viewed as a special case of consistency as understood in the classic ACID model of transaction processing. Integrity is violated when a message is actively modified in transit. Information security systems typically provide message integrity in addition to data confidentiality.

Availability

For any information system to serve its purpose, the information must be available when it is needed. This means that the computing systems used to store and process the information, the security controls used to protect it, and the communication channels used to access it must be functioning correctly. High availability systems aim to remain available at all times, preventing service disruptions due to power outages, hardware failures, and system upgrades. Ensuring availability also involves preventing denial-of-service attacks, such as a flood of incoming messages to the target system essentially forcing it to shut down.

Authenticity

In computing, e-Business, and information security, it is necessary to ensure that the data, transactions, communications or documents (electronic or physical) are genuine. It is also important for authenticity to validate that both parties involved are who they claim to be. Some information security systems incorporate authentication features such as "digital signatures", which give evidence that the message data is genuine and was sent by someone possessing the proper signing key.

4.1.4.5 Non-repudiation

In law, non-repudiation implies one's intention to fulfill their obligations to a contract. It also implies that one party of a transaction cannot deny having received a transaction nor can the other party deny having sent a transaction.

It is important to note that while technology such as cryptographic systems can assist in non-repudiation efforts, the concept is at its core a legal concept transcending the realm of technology. It is not, for instance, sufficient to show that the message matches a digital signature signed with the sender's private key, and thus only the sender could have sent the message and nobody else could have altered it in transit. The alleged sender could in return demonstrate that the digital signature algorithm is vulnerable or flawed, or allege or prove that his signing key has been compromised. The fault for these violations may or may not lie with the sender himself, and such assertions may or may not relieve the sender of liability, but the assertion would invalidate the claim that the signature necessarily proves authenticity and integrity and thus prevents repudiation.

Risk management

The Certified Information Systems Auditor (CISA) Review Manual 2006 provides the following definition of risk management: "Risk management is the process of identifying vulnerabilities and threats to the information resources used by an organization in achieving business objectives, and deciding what countermeasures, if any, to take in reducing risk to an acceptable level, based on the value of the information resource to the organization."

There are two things in this definition that may need some clarification. First, the process of risk management is an ongoing, iterative process. It must be repeated indefinitely. The business environment is constantly changing and new threats and vulnerabilities emerge every day. Second, the choice of countermeasures (controls) used to manage risks must strike a balance between productivity, cost, effectiveness of the countermeasure, and the value of the informational asset being protected.

Risk analysis and risk evaluation processes have their limitations since, when security incidents occur, they emerge in a context, and their rarity and even their uniqueness give rise to unpredictable threats. The analysis of these phenomena which are characterized by breakdowns, surprises and side-effects, requires a theoretical approach which is able to examine and interpret subjectively the detail of each incident.

Risk is the likelihood that something bad will happen that causes harm to an informational asset (or the loss of the asset). Vulnerability is a weakness that could be used to endanger or cause harm to an informational asset. A threat is anything (manmade or act of nature) that has the potential to cause harm.

The likelihood that a threat will use a vulnerability to cause harm creates a risk. When a threat does use a vulnerability to inflict harm, it has an impact. In the context of information security, the impact is a loss of availability, integrity, and confidentiality, and possibly other losses (lost income, loss of life, loss of real property). It should be pointed out that it is not possible to identify all risks, nor is it possible to eliminate all risk. The remaining risk is called "residual risk".

A risk assessment is carried out by a team of people who have knowledge of specific areas of the business. Membership of the team may vary over time as different parts of the business are assessed. The assessment may use a subjective qualitative analysis based on informed opinion, or where reliable dollar figures and historical information is available, the analysis may use quantitative analysis.

The research has shown that the most vulnerable point in most information systems is the human user, operator, designer, or other human The ISO/IEC 27002:2005 Code of practice for information security management recommends the following be examined during a risk assessment:

- security policy,
- organization of information security,

- physical and environmental security,
- communications and operations management,
- access control,
- information systems acquisition, development and maintenance,
- information security incident management,
- business continuity management, and
- Regulatory compliance.

In broad terms, the risk management process consists of:

1. Identification of assets and estimating their value. Include: people, buildings, hardware, software, data (electronic, print, and other), and supplies.
2. Conduct a threat assessment. Include: Acts of nature, acts of war, accidents, and malicious acts originating from inside or outside the organization.
3. Conduct a vulnerability assessment, and for each vulnerability, calculate the probability that it will be exploited. Evaluate policies, procedures, standards, training, physical security, quality control, technical security.
4. Calculate the impact that each threat would have on each asset. Use qualitative analysis or quantitative analysis.
5. Identify, select and implement appropriate controls. Provide a proportional response. Consider productivity, cost effectiveness, and value of the asset.
6. Evaluate the effectiveness of the control measures. Ensure the controls provide the required cost effective protection without discernible loss of productivity.

For any given risk, management can choose to accept the risk based upon the relative low value of the asset, the relative low frequency of occurrence, and the relative low impact on the business. Or, leadership may choose to mitigate the risk by selecting and implementing appropriate control measures to reduce the risk. In some cases, the risk can be transferred to another business by buying insurance or outsourcing to another business. The reality of some risks may be disputed. In such cases leadership may choose to deny the risk.

Security classification for information

An important aspect of information security and risk management is recognizing the value of information and defining appropriate procedures and protection requirements for the information. Not all information is equal and so not all information requires the same degree of protection. This requires information to be assigned a security classification.

The first step in information classification is to identify a member of senior management as the owner of the particular information to be classified. Next, develop a classification policy. The policy should describe the different classification labels, define the criteria for information to be assigned a particular label, and list the required security controls for each classification.

Some factors that influence which classification information should be assigned include how much value that information has to the organization, how old the information is and whether or not the information has become obsolete. Laws and other regulatory requirements are also important considerations when classifying information.

The Business Model for Information Security enables security professionals to examine security from systems perspective, creating an environment where security can be managed holistically, allowing actual risks to be addressed.

The type of information security classification labels selected and used will depend on the nature of the organization, with examples being:

- In the business sector, labels such as: Public, Sensitive, Private, and Confidential.
- In the government sector, labels such as: Unclassified, Sensitive But Unclassified, Restricted, Confidential, Secret, Top Secret and their non-English equivalents.
- In cross-sectoral formations, the Traffic Light Protocol, this consists of: White, Green, Amber, and Red.

All employees in the organization, as well as business partners, must be trained on the classification schema and understand the required security controls and handling procedures for each classification. The classification of a particular information asset that has been assigned should be reviewed periodically to ensure the classification is still appropriate for the information and to ensure the security controls required by the classification are in place and are followed in their right procedures.

Security governance

The Software Engineering Institute at Carnegie Mellon University, in a publication titled "Governing for Enterprise Security (GES)", defines characteristics of effective security governance. These include:

- An enterprise-wide issue
- Leaders are accountable
- Viewed as a business requirement
- Risk-based
- Roles, responsibilities, and segregation of duties defined
- Addressed and enforced in policy
- Adequate resources committed
- Staff aware and trained
- A development life cycle requirement
- Planned, managed, measurable, and measured
- Reviewed and audited

System testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

Testing the whole system

System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s).

Types of tests to include in system testing

The following examples are different types of testing that should be considered during System testing:

1. Graphical user interface testing

In software engineering, graphical user interface testing is the process of testing a product's graphical user interface to ensure it meets its written specifications. This is normally done through the use of a variety of test cases.

2. Usability testing

Usability testing is a technique used in user-centered interaction design to evaluate a product by testing it on users. This can be seen as an irreplaceable usability practice, since it gives direct input on how real users use the system.[1] This is in contrast with usability inspection methods where experts use different methods to evaluate a user interface without involving users.

3. Software performance testing

In software engineering, performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload.

Performance testing is a subset of performance engineering, an emerging computer science practice which strives to build performance into the implementation, design and architecture of a system.

4. Compatibility testing

Compatibility testing, part of software non-functional tests, is testing conducted on the application to evaluate the application's compatibility with the computing environment. Computing environment may contain some or all of the below mentioned elements:

- Computing capacity of Hardware Platform (IBM 360, HP 9000, etc.).
- Bandwidth handling capacity of networking hardware
- Compatibility of peripherals (Printer, DVD drive, etc.)
- Operating systems (Linux, Windows, Mac etc.)
- Database (Oracle, SQL Server, MySQL, etc.)
- Other System Software (Web server, networking/ messaging tool, etc.)
- Browser compatibility (Chrome, Firefox, Netscape, Internet Explorer, Safari, etc.)

5. Load testing

6.

Load testing is the process of putting demand on a system or device and measuring its response. Load testing is performed to determine a system‘s behavior under both normal and anticipated peak load conditions. It helps to identify the maximum operating capacity of an application as well as any bottlenecks and determine which element is causing degradation. When the load placed on the system is raised beyond normal usage patterns, in order to test the system's response at unusually high or peak loads, it is known as stress testing. The load is usually so great that error conditions are the expected result, although no clear boundary exists when an activity ceases to be a load test and becomes a stress test.

7. Volume testing

Volume Testing belongs to the group of non-functional tests, which are often misunderstood and/or used interchangeably. Volume testing refers to testing a software application with a certain amount of data. This amount can, in generic terms, be the database size or it could also be the size of an interface file that is the subject of volume testing. For example, if you want to volume test your application with a specific database size, you will expand your database to that size and then test the application's performance on it. Another example could be when there is a requirement for your application to interact with an interface file (could be any file such as .dat, .xml); this interaction could be reading and/or writing on to/from the file. You will create a sample file of the size you want and then test the application's functionality with that file in order to test the performance.

8. Stress testing

Stress testing (sometimes called torture testing) is a form of deliberately intense or thorough testing 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. Reasons can include:

- to determine breaking points or safe usage limits
- to confirm intended specifications are being met
- to determine modes of failure (how exactly a system fails)
- to test stable operation of a part or system outside standard usage

9. Security testing

Security testing is a process intended to reveal flaws in the security mechanisms of an information system that protect data and maintain functionality as intended. Due to the logical limitations of security testing, passing security testing is not an indication that no flaws exist or that the system adequately satisfies the security requirements.

Typical security requirements may include specific elements of confidentiality, integrity, authentication, availability, authorization and non-repudiation. Actual security requirements tested depend on the security requirements implemented by the system. Security testing as a term has a number of different meanings and can be completed in a number of different ways. As such a Security Taxonomy helps us to understand these different approaches and meanings by providing a base level to work from.

10. Scalability testing

Scalability Testing, part of the battery of non-functional tests, is the testing of a software application for measuring its capability to scale up or scale out in terms of any of its non-functional capability.

Performance, scalability and reliability are usually considered together by software quality analysts.

Scalability testing tools exist (often leveraging scalable resources themselves) in order to test user load, concurrent connections, transactions, and throughput of many internet services. Of the available testing services, those offering API support suggest that environment of continuous deployment also continuously test how recent changes may impact scalability.

11. Sanity testing

A sanity test or sanity check is a basic test to quickly evaluate whether a claim or the result of a calculation can possibly be true. It is a simple check to see if the produced material is rational (that the material's creator was thinking rationally, applying sanity).

In arithmetic, for example, when multiplying by 9, using the divisibility rule for 9 to verify that the sum of digits of the result is divisible by 9 is a sanity test - it will not catch every multiplication error; however it's a quick and simple method to discover many possible errors

12. Smoke testing

In computer programming and software testing, smoke testing (also confidence testing, sanity testing is preliminary testing to reveal simple failures severe enough to reject a prospective software release. A subset of test cases that cover the most important functionality of a component or system is selected and run, to ascertain if crucial functions of a program work correctly. When used to determine if a computer program should be subjected to further, more fine-grained testing, a smoke test may be called an intake test.

For example, a smoke test may ask basic questions like "Does the program run?", "Does it open a window?", or "Does clicking the main button do anything?" The process aims to determine whether the application is so badly broken as to make further immediate testing unnecessary. As the book "Lessons Learned in Software Testing‖ puts it, "smoke tests broadly cover product features in a limited time ... if key features don't work or if key bugs haven't yet been fixed, your team won't waste further time installing or testing".

Error detection and correction

In information theory and coding theory with applications in computer science and telecommunication, error detection and correction or error control are techniques that enable reliable delivery of digital data over unreliable communication channels. Many communication channels are subject to channel noise, and thus errors may be introduced during transmission from the source to a receiver. Error detection techniques allow detecting such errors, while error correction enables reconstruction of the original data in many cases.

Implementation

Error correction may generally be realized in two different ways:

- Automatic repeat request (ARQ) (sometimes also referred to as backward error correction): This is an error control technique whereby an error detection scheme is combined with requests for retransmission of erroneous data. Every block of data received is checked using the error detection code used, and if the check fails, retransmission of the data is requested – this may be done repeatedly, until the data can be verified.
- Forward error correction (FEC): The sender encodes the data using an error-correcting code (ECC) prior to transmission. The additional information (redundancy) added by the code is used by the receiver to recover the original data. In general, the reconstructed data is what is deemed the "most likely" original data.

ARQ and FEC may be combined, such that minor errors are corrected without retransmission, and major errors are corrected via a request for retransmission: this is called hybrid automatic repeat-request (HARQ).

Error detection schemes

Error detection is most commonly realized using a suitable hash function (or checksum algorithm). A hash function adds a fixed-length tag to a message, which enables receivers to verify the delivered message by recomputing the tag and comparing it with the one provided.

There exists a vast variety of different hash function designs. However, some are of particularly widespread use because of either their simplicity or their suitability for detecting certain kinds of errors (e.g., the cyclic redundancy check's performance in detecting burst errors).

Random-error-correcting codes based on minimum distance coding can provide a suitable alternative to hash functions when a strict guarantee on the minimum number of errors to be detected is desired. Repetition codes, described below, are special cases of error-correcting codes: although rather inefficient, they find applications for both error correction and detection due to their simplicity.

Repetition codes

A repetition code is a coding scheme that repeats the bits across a channel to achieve error-free communication. Given a stream of data to be transmitted, the data is divided into blocks of bits. Each block is transmitted some predetermined number of times. For example, to send the bit pattern "1011", the four-bit block can be repeated three times, thus producing "1011 1011 1011". However, if this twelve-bit pattern was received as "1010 1011 1011" – where the first block is unlike the other two – it can be determined that an error has occurred.

Repetition codes are very inefficient, and can be susceptible to problems if the error occurs in exactly the same place for each group (e.g., "1010 1010 1010" in the previous example would be detected as correct). The advantage of repetition codes is that they are extremely simple, and are in fact used in some transmissions of numbers stations.

Parity bits

A parity bit is a bit that is added to a group of source bits to ensure that the number of set bits (i.e., bits with value 1) in the outcome is even or odd. It is a very simple scheme that can be used to detect single or any other odd number (i.e., three, five, etc.) of errors in the output. An even number of flipped bits will make the parity bit appear correct even though the data is erroneous.

Extensions and variations on the parity bit mechanism are horizontal redundancy checks, vertical redundancy checks, and "double," "dual," or "diagonal" parity (used in RAID-DP).

Checksums

A checksum of a message is a modular arithmetic sum of message code words of a fixed word length (e.g., byte values). The sum may be negated by means of a ones'-complement operation prior to transmission to detect errors resulting in all-zero messages.

Checksum schemes include parity bits, check digits, and longitudinal redundancy checks. Some checksum schemes, such as the Damm algorithm, the Luhn algorithm, and the Verhoeff algorithm, are specifically designed to detect errors commonly introduced by humans in writing down or remembering identification numbers.

Cyclic redundancy checks (CRCs)

A cyclic redundancy check (CRC) is a single-burst-error-detecting cyclic code and non-secure hash function designed to detect accidental changes to digital data in computer networks. It is not suitable for detecting maliciously introduced errors. It is characterized by specification of a so-called generator polynomial, which is used as the divisor in a polynomial long division over a finite field, taking the input data as the dividend, and where the remainder becomes the result.

Cyclic codes have favorable properties in that they are well suited for detecting burst errors. CRCs are particularly easy to implement in hardware, and are therefore commonly used in digital networks and storage devices such as hard disk drives.

Even parity is a special case of a cyclic redundancy check, where the single-bit CRC is generated by the divisor $x + 1$.

Cryptographic hash functions

The output of a cryptographic hash function, also known as a message digest, can provide strong assurances about data integrity, whether changes of the data are accidental (e.g., due to transmission errors) or maliciously introduced. Any modification to the data will likely be detected through a mismatching hash value. Furthermore, given some hash value, it is infeasible to find some input data (other than the one given) that will yield the same hash value. If an attacker can change not only the message but also the hash value, then a keyed hash or message authentication code (MAC) can be used for additional security. Without knowing the key, it is infeasible for the attacker to calculate the correct keyed hash value for a modified message.

Error-correcting codes

Any error-correcting code can be used for error detection. A code with minimum Hamming distance, d, can detect up to $d - 1$ errors in a code word. Using minimum-distance-based error-correcting codes for error detection can be suitable if a strict limit on the minimum number of errors to be detected is desired.

Codes with minimum Hamming distance d = 2 are degenerate cases of error-correcting codes, and can be used to detect single errors. The parity bit is an example of a single-error-detecting code.

An error-correcting code (ECC) or forward error correction (FEC) code is a system of adding redundant data, or parity data, to a message, such that it can be recovered by a receiver even when a number of errors (up to the capability of the code being used) were introduced, either during the process of transmission, or on storage. Since the receiver does not have to ask the sender for retransmission of the data, a back-channel is not required in forward error correction, and it is therefore suitable for simplex communication such as broadcasting. Error-correcting codes are frequently used in lower-layer communication, as well as for reliable storage in media such as CDs, DVDs, hard disks, and RAM.

Error-correcting codes are usually distinguished between convolution codes and block codes:

- Convolution codes are processed on a bit-by-bit basis. They are particularly suitable for implementation in hardware, and the Viterbi decoder allows optimal decoding.

- Block codes are processed on a block-by-block basis. Early examples of block codes are repetition codes, Hamming codes and multidimensional parity-check codes. They were followed by a number of efficient codes, Reed–Solomon codes being the most notable due to their current widespread use. Turbo codes and low-density parity-check codes (LDPC) are relatively new constructions that can provide almost optimal efficiency.

Shannon's theorem is an important theorem in forward error correction, and describes the maximum information rate at which reliable communication is possible over a channel that has a certain error probability or signal-to-noise ratio (SNR). This strict upper limit is expressed in terms of the channel capacity. More specifically, the theorem says that there exist codes such that with increasing encoding length the probability of error on a discrete memory less channel can be made arbitrarily small, provided that the code rate is smaller than the channel capacity. The code rate is defined as the fraction k/n of k source symbols and n encoded symbols.

The actual maximum code rate allowed depends on the error-correcting code used, and may be lower. This is because Shannon's proof was only of existential nature, and did not show how to construct codes which are both optimal and have efficient encoding and decoding algorithms.

Automatic repeat request (ARQ)

Automatic Repeat request (ARQ) is an error control method for data transmission that makes use of error-detection codes, acknowledgment and/or negative acknowledgment messages, and timeouts to achieve reliable data transmission. An acknowledgment is a message sent by the receiver to indicate that it has correctly received a data frame.

Usually, when the transmitter does not receive the acknowledgment before the timeout occurs (i.e., within a reasonable amount of time after sending the data frame), it retransmits the frame until it is either correctly received or the error persists beyond a predetermined number of retransmissions.

Three types of ARQ protocols are Stop-and-wait ARQ, Go-Back-N ARQ, and Selective Repeat ARQ.

ARQ is appropriate if the communication channel has varying or unknown capacity, such as is the case on the Internet. However, ARQ requires the availability of a back channel, results in possibly increased latency due to retransmissions, and requires the maintenance of buffers and timers for retransmissions, which in the case of network congestion can put a strain on the server and overall network capacity.

ARQ is used on shortwave radio data links in the form of ARQ-E or combined with multiplexing as ARQ-M.

Hybrid schemes

Hybrid ARQ is a combination of ARQ and forward error correction. There are two basic approaches:

- Messages are always transmitted with FEC parity data (and error-detection redundancy). A receiver decodes a message using the parity information, and requests retransmission using ARQ only if the parity data was not sufficient for successful decoding (identified through a failed integrity check).
- Messages are transmitted without parity data (only with error-detection information). If a receiver detects an error, it requests FEC information from the transmitter using ARQ, and uses it to reconstruct the original message.

The latter approach is particularly attractive on an erasure channel when using a rate less erasure code.

Applications

Applications that require low latency (such as telephone conversations) cannot use Automatic Repeat request (ARQ); they must use forward error correction (FEC). By the time an ARQ system discovers an error and re-transmits it, the re-sent data will arrive too late to be any good.

Applications where the transmitter immediately forgets the information as soon as it is sent (such as most television cameras) cannot use ARQ; they must use FEC because when an error occurs, the original data is no longer available. (This is also why FEC is used in data storage systems such as RAID and distributed data store).

Applications that use ARQ must have a return channel; applications having no return channel cannot use ARQ. Applications that require extremely low error rates (such as digital money transfers) must use ARQ. Reliability and inspection engineering also make use of the theory of error-correcting codes.

Information systems controls

Controls

Selecting proper controls and implementing those will initially help an organization to bring down risk to acceptable levels. Control selection should follow and should be based on the risk assessment. Controls can vary in nature but fundamentally they are ways of protecting the confidentiality, integrity or availability of information. ISO/IEC 27001:2005 has defined 133 controls in different areas, but this is not exhaustive. You can implement additional controls according to requirement of the organization. ISO 27001:2013 has cut down the number of controls to 113.

Administrative

Administrative controls (also called procedural controls) consist of approved written policies, procedures, standards and guidelines. Administrative controls form the framework for running the business and managing people. They inform people on how the business is to be run and how day-to-day operations are to be conducted. Laws and regulations created by government bodies are also a type of administrative control because they inform the business. Some industry sectors have policies, procedures, standards and guidelines that must be followed – the Payment Card Industry (PCI) Data Security Standard required by Visa and MasterCard is such an example. Other examples of administrative controls include the corporate security policy, password policy, hiring policies, and disciplinary policies.

Administrative controls form the basis for the selection and implementation of logical and physical controls. Logical and physical controls are manifestations of administrative controls. Administrative controls are of paramount importance.

Logical

Logical controls (also called technical controls) use software and data to monitor and control access to information and computing systems. For example: passwords, network and host-based firewalls, network intrusion detection systems, access control lists, and data encryption are logical controls.

An important logical control that is frequently overlooked is the principle of least privilege. The principle of least privilege requires that an individual, program or system process is not granted any more access privileges than are necessary to perform the task. A blatant example of the failure to adhere to the principle of least privilege is logging into Windows as user Administrator to read Email and surf the Web. Violations of this principle can also occur when an individual collects additional access privileges over time. This happens when employees' job duties change, or they are promoted to a new position, or they transfer to another department. The access privileges required by their new duties are frequently added onto their already existing access privileges which may no longer be necessary or appropriate.

Physical

Physical controls monitor and control the environment of the work place and computing facilities. They also monitor and control access to and from such facilities. For example: doors, locks, heating and air conditioning, smoke and fire alarms, fire suppression systems, cameras, barricades, fencing, security guards, cable locks, etc. Separating the network and workplace into functional areas are also physical controls.

An important physical control that is frequently overlooked is the separation of duties. Separation of duties ensures that an individual can not complete a critical task by himself. For example: an employee who submits a request for reimbursement should not also be able to authorize payment or print the check. An applications programmer should not also be the server administrator or the database administrator – these roles and responsibilities must be separated from one another.

- General controls
  - ✓ Govern design, security, and use of computer programs and security of data files in general throughout organization‘s information technology infrastructure.
  - ✓ Apply to all computerized applications

Combination of hardware, software, and manual procedures to create overall control environment

- Types of general controls
  - ✓ Software controls
  - ✓ Hardware controls
  - ✓ Computer operations controls
  - ✓ Data security controls
  - ✓ Implementation controls
  - ✓ Administrative controls

- Application controls
  - ✓ Specific controls unique to each computerized application, such as payroll or order processing
  - ✓ Include both automated and manual procedures
  - ✓ Ensure that only authorized data are completely and accurately processed by that application Include:
    - Input controls
    - Processing controls
    - Output controls

IS Vulnerability

In computer security, vulnerability is a weakness which allows an attacker to reduce a system's information assurance. Vulnerability is the intersection of three elements: a system susceptibility or flaw, attacker access to the flaw, and attacker capability to exploit the flaw. To exploit vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness. In this frame, vulnerability is also known as the attack surface.

Vulnerability management is the cyclical practice of identifying, classifying, remediating, and mitigating vulnerabilities. This practice generally refers to software vulnerabilities in computing systems.

A security risk may be classified as vulnerability. The use of vulnerability with the same meaning of risk can lead to confusion. The risk is tied to the potential of a significant loss. Then there are vulnerabilities without risk: for example when the affected asset has no value. Vulnerability with one or more known instances of working and fully implemented attacks is classified as an exploitable vulnerability a vulnerability for which can exploit exists. The window of vulnerability is the time from when the security hole was introduced or manifested in deployed software, to when access was removed, a security fix was available/deployed, or the attackers was disabled see zero-day attack.

A weakness of an asset or group of assets that can be exploited by one or more threats where an asset is anything that has value to the organization, its business operations and their continuity, including information resources that support the organization's mission.

Data and Computer Security

Dictionary of standards concepts and terms, authors Dennis Longley and Michael Shain, Stockton Press, ISBN 0-935859-17-9, defines vulnerability as:

> 1) In computer security, a weakness in automated systems security procedures, administrative controls, Internet controls, etc., that could be exploited by a threat to gain unauthorized access to information or to disrupt critical processing.
>
> 2) In computer security, a weakness in the physical layout, organization, procedures, personnel, management, administration, hardware or software that may be exploited to cause harm to the ADP system or activity.
>
> 3) In computer security, any weakness or flaw existing in a system. The attack or harmful event, or the opportunity available to a threat agent to mount that attack.

In this paper, the definitions of these classes and transitions are considered axiomatic. A vulnerable state is an authorized state from which an unauthorized state can be reached using authorized state transitions. A compromised state is the state so reached. An attack is a sequence of authorized state

transitions which end in a compromised state. By definition, an attack begins in a vulnerable state. Vulnerability is a characterization of a vulnerable state which distinguishes it from all non-vulnerable states.

National Information Assurance Training and Education Center defines vulnerability

1. A weakness in automated system security procedures, administrative controls, internal controls, and so forth that could be exploited by a threat to gain unauthorized access to information or disrupt critical processing.

2. A weakness in system security procedures, hardware design, internal controls, etc., which could be exploited to gain unauthorized access to classify or sensitive information.

3. A weakness in the physical layout, organization, procedures, personnel, management, administration, hardware, or software that may be exploited to cause harm to the ADP system or activity. The presence of vulnerability does not in itself cause harm; vulnerability is merely a condition or set of conditions that may allow the ADP system or activity to be harmed by an attack.

4. An assertion primarily concerning entities of the internal environment (assets); we say that an asset (or class of assets) is vulnerable (in some way, possibly involving an agent or collection of agents); we write: V (i,e) where: e may be an empty set.

5. Susceptibility to various threats.

6. A set of properties of a specific internal entity that, in union with a set of properties of a specific external entity, implies a risk.

7. The characteristics of a system which cause it to suffer a definite degradation (incapability to perform the designated mission) as a result of having been subjected to a certain level of effects in an unnatural (manmade) hostile environment.

Vulnerability and risk factor models

A resource (either physical or logical) may have one or more vulnerabilities that can be exploited by a threat agent in a threat action. The result can potentially compromise the confidentiality, integrity or availability of resources (not necessarily the vulnerable one) belonging to an organization and/or others parties involved (customers, suppliers).
The so-called CIA triad is the basis of Information Security.

An attack can be active when it attempts to alter system resources or affect their operation, compromising integrity or availability. A "passive attack" attempts to learn or make use of information from the system but does not affect system resources, compromising confidentiality.

OWASP: relationship between threat agent and business impact OWASP depicts the same phenomenon in slightly different terms: a threat agent through an attack vector exploits a weakness (vulnerability) of the system and the related security controls, causing a technical impact on an IT resource (asset) connected to a business impact.

Information security management system

A set of policies concerned with information security management, the information security management system (ISMS), has been developed to manage, according to Risk management principles, the countermeasures in order to ensure the security strategy is set up following the rules and regulations applicable in a country. These countermeasures are also called Security controls, but when applied to the transmission of information they are called security services.[17]

Classification

Vulnerabilities are classified according to the asset class they are related to:

- hardware
  - susceptibility to humidity
  - susceptibility to dust
  - susceptibility to soiling
  - susceptibility to unprotected storage
- software
  - insufficient testing
  - lack of audit trail
- network
  - unprotected communication lines
  - insecure network architecture
- personnel
  - inadequate recruiting process
  - inadequate security awareness
- site
  - area subject to flood
  - unreliable power source
- organizational
  - lack of regular audits
  - lack of continuity plans
  - lack of security

Causes

- Complexity: Large, complex systems increase the probability of flaws and unintended access points

- Familiarity: Using common, well-known code, software, operating systems, and/or hardware increases the probability an attacker has or can find the knowledge and tools to exploit the flaw
- Connectivity: More physical connections, privileges, ports, protocols, and services and time each of those are accessible increase vulnerability
- Password management flaws: The computer user uses weak passwords that could be discovered by brute force. The computer user stores the password on the computer where a program can access it. Users re-use passwords between many programs and websites.

- Fundamental operating system design flaws: The operating system designer chooses to enforce suboptimal policies on user/program management. For example operating systems with policies such as default permit grant every program and every user full access to the entire computer. This operating system flaw allows viruses and malware to execute commands on behalf of the administrator.
- Internet Website Browsing: Some internet websites may contain harmful Spyware or Adware that can be installed automatically on the computer systems. After visiting those websites, the computer systems become infected and personal information will be collected and passed on to third party individuals.
- Software bugs: The programmer leaves an exploitable bug in a software program. The software bug may allow an attacker to misuse an application.
- Unchecked user input: The program assumes that all user input is safe. Programs that do not check user input can allow unintended direct execution of commands or SQL statements (known as Buffer overflows, SQL injection or other non-validated inputs).
- Not learning from past mistakes: for example most vulnerabilities discovered in IPv4 protocol software were discovered in the new IPv6 implementations

The research has shown that the most vulnerable point in most information systems is the human user, operator, designer, or other human: so humans should be considered in their different roles as asset, threat, information resources. Social engineering is an increasing security concern.

Vulnerability consequences

The impact of a security breach can be very high. The fact that IT managers, or upper management, can (easily) know that IT systems and applications have vulnerabilities and do not perform any action to manage the IT risk is seen as misconduct in most legislations. Privacy law forces managers to act to reduce the impact or likelihood of that security risk. Information technology security audit is a way to let other independent people certify that the IT environment is managed properly and lessen the responsibilities, at least having demonstrated the good faith. Penetration test is a form of verification of the weakness and countermeasures adopted by an organization: a White hat hacker tries to attack an organization's information technology assets, to find out how easy or difficult it is to compromise the IT security.

One of the key concepts of information security is the principle of defense in depth: i.e. to set up a multilayer defense system that can:

- prevent the exploit
- detect and intercept the attack
- find out the threat agents and prosecute them

Intrusion detection system is an example of a class of systems used to detect attacks. Physical security is a set of measures to protect physically the information asset: if somebody can get physical access to the information asset, it is quite easy to make resources unavailable to its legitimate users.

Vulnerability disclosure

Responsible disclosure (many now refer to it as 'coordinated disclosure' because the first is a biased word) of vulnerabilities is a topic of great debate. As reported by The Tech Herald in August 2010, "Google, Microsoft, TippingPoint, and Rapid7 have recently issued guidelines and statements addressing how they will deal with disclosure going forward."

A responsible disclosure first alerts the affected vendors confidentially before alerting CERT two weeks later, which grants the vendors another 45 day grace period before publishing a security advisory.

Full disclosure is done when all the details of vulnerability is publicized, perhaps with the intent to put pressure on the software or procedure authors to find a fix urgently.

Well respected authors have published books on vulnerabilities and how to exploit them: Hacking: The Art of Exploitation Second Edition is a good example.

Security researchers catering to the needs of the cyberwarfare or cybercrime industry have stated that this approach does not provide them with adequate income for their efforts. Instead, they offer their exploits privately to enable Zero day attacks. The never ending effort to find new vulnerabilities and to fix them is called Computer insecurity.

Vulnerability inventory

Mitre Corporation maintains a list of disclosed vulnerabilities in a system called Common Vulnerabilities and Exposures, where vulnerability is classified (scored) using Common Vulnerability Scoring System (CVSS).

OWASP collects a list of potential vulnerabilities in order to prevent system designers and programmers from inserting vulnerabilities into the software.

Examples of vulnerabilities

Vulnerabilities are related to:

- physical environment of the system
- the personnel
- management
- administration procedures and security measures within the organization
- business operation and service delivery
- hardware
- software
- communication equipment and facilities

It is evident that a pure technical approach cannot even protect physical assets: one should have administrative procedure to let maintenance personnel to enter the facilities and people with adequate knowledge of the procedures, motivated to follow it with proper care.

Software vulnerabilities

Common types of software flaws that lead to vulnerabilities include:

- Memory safety violations, such as:
  - Buffer overflows and over-reads
  - Dangling pointers
- Input validation errors, such as:
  - Format string attacks
  - SQL injection
  - Code injection
  - E-mail injection
  - Directory traversal
  - Cross-site scripting in web applications
  - HTTP header injection
  - HTTP response splitting
- Race conditions, such as:
  - Time-of-check-to-time-of-use bugs
  - Symlink races
- Privilege-confusion bugs, such as:
  - Cross-site request forgery in web applications
  - Clickjacking
  - FTP bounce attack
- Privilege escalation
- User interface failures, such as:

- Blaming the Victim Prompting a user to make a security decision without giving the user enough information to answer it
- Race Conditions

Disaster Management Information System (DMIS)

Objectives

- To overcome limitation of existing system.
- Effective utilizations of natural resources database in event of disaster.
- Building decision support system for better district administration
- Providing vital information related to pre-disaster and post-disaster at fingertips.
- Facilitating users for easy data integration.
- Editing, updating of spatial and non-spatial data at ease.
- To assist in post disaster damage assessment analysis.
- Provide centralized system that would be time & cost effective and maintenance free.
- Development of user friendly customized DMIS software.

"Disaster management" means a continuous and integrated process of planning, organizing, coordinating and implementing measures which are necessary or expedient for prevention of danger or threat of any disaster, mitigation or reduction of risk of any disaster or its severity or consequences, capacity-building, preparedness to deal with any disaster, prompt response to any threatening disaster situation or disaster, assessing the severity or magnitude of effects of any disaster, evacuation, rescue and relief, rehabilitation and reconstruction. Disaster Management comprises all forms of activities including structural and nonstructural measures to avoid (i.e. prevention) or to limit (i.e. mitigation and preparedness) adverse effects of disasters in the pre-disaster phase and post disaster stage like Response, Relief, Recovery, & Reconstruction.

As per the directives laid under GOI-UNDP program, the Government of Maharashtra (GOM) has a Disaster Management Unit (DMU), which prepares action plan to support and strengthen the efforts of district administration for overall disaster vigilance of the State. In view of preparedness, each district has evolved its own district disaster management action plan (DDMAP). It is anticipated that these multi-hazard response plans would increase the effectiveness of administrative intervention.

The DDMAP addresses the districts‘ response to disaster situations such as earthquakes, floods, cyclones, epidemics, off-site industrial disasters, roads accidents and fires. Some of these disasters such as floods and earthquakes affect widespread area causing extensive damage to life, property and environment while disaster like epidemics only affect populations. Anyhow, the management of these disasters requires far-reaching resources and manpower for containment by remedial action.

GIS is a powerful technology that can assist decision-making in all phases of the disaster management cycle. GIS tools are used for integrating the geographic (i.e. location) and the associated attribute data pertaining to the location and its spatial relationship with numerous other parameters, to

carry out effective spatial planning, minimize the possible damage, ensure immediate action when required and prioritize actions for long-term risk reduction.

Resources database on various themes obtained through Remote Sensing data has been compiled for all the districts of Maharashtra. Similarly attribute data on Demography & Census, government core sectors, and past disaster have been integrated in the DMIS. Spatial and non-spatial database has been generated in GIS environment.
A customized system has been developed for each district for prioritizing hazards for use in developing Mitigation Strategies, Risk Estimation and Hazard and Vulnerability Mapping.

A user-friendly menu driven software has been developed in Arc GIS using Arc Objects with Visual Basic 6.0. It has been designed and customized keeping in mind the skill level of the expected users at the district level. The methodology and database has been customized for easy implementation.

High resolution satellite data for the study area are analyzed for generation of base map as well as DEM. Slopes (or Contours) generated from DEM is used for locating shelter camps in event of disaster.

The themes like Slope, Landuse, and Geology are amalgamated so as to calculate weighted ranks that indicate vulnerability index. For example, Slope (0-1%) + Landuse (River, Reservoir, Double Cropped) + Geology (water body mask, deep alluvial plain) = Rank 1. This index will decide the sensitivity of the flood prone area i.e. very high-risk area, high-risk area, moderate risk area, low risk area and no risk area.

The software has been thoroughly tested before its packaging and deployment. Each district user is authenticated for accessing data in DMIS software.

Standard DMIS software has been developed using prototyping model of Software Development Life Cycle (SDLC). The final software has been installed and implemented in Relief and Rehabilitation Cell at Mantralaya, Mumbai, Maharashtra State.

The DMIS is installed and implemented in all districts of Maharashtra State followed by demonstration to District Collector and training to concerned officials. The User Guide Manuals are provided to users for further guidance and operating software at ease. Detail requirement analysis of the user had been done with respect to the standard procedures followed during disaster and effort is taken to incorporate the same in the software.

Project Deliverables

Following Deliverables were handed over to 15 Collectorate Offices as mentioned in table below and Relief & Rehabilitation Cell at Mantralaya, Mumbai.

- ✓ Natural Resources Database at 1:50,000 scale in GIS format.
- ✓ DMIS customized software

- ✓ Data Requirement Sheet
- ✓ User Manual

Beneficiaries

- Relief and Rehabilitation Cell, Revenue and Forest Dept, Mantralaya, Mumbai.
- Planning Department, Mantralaya, Mumbai.
- Secretaries of various departments
- All District Collectorate (or Disaster Management Units)
- Functional Officers
- NGOs

Computer Crimes

- Definition: the act of using a computer to commit an illegal act
  - Authorized and unauthorized computer access
  - Examples
    - Stealing time on company computers
    - Breaking into government Web sites
    - Stealing credit card information
- Federal and State Laws
  - Stealing or compromising data
  - Gaining unauthorized computer access
  - Violating data belonging to banks
  - Intercepting communications
  - Threatening to damage computer systems
  - Disseminating viruses
- Hacking and Cracking
  - Hacker – one who gains unauthorized computer access, but without doing damage
  - Cracker – one who breaks into computer systems for the purpose of doing damage

Types of computer crime

- Data diddling: modifying data
- Salami slicing: skimming small amounts of money
- Phreaking: making free long distance calls
- Cloning: cellular phone fraud using scanners
- Carding: stealing credit card numbers online
- Piggybacking: stealing credit card numbers by spying
- Social engineering: tricking employees to gain access
- Dumpster diving: finding private info in garbage cans
- Spoofing: stealing passwords through a false login page
- Software piracy

- Computer viruses and destructive code
  - Virus – a destructive program that disrupts the normal functioning of computer systems
  - Types:
    - Worm: usually does not destroy files; copies itself
    - Trojan horses: Activates without being detected; does not copy itself
    - Logic or time bombs: A type of Trojan horse that stays dormant for a period of time before activating

Securing the Web

Web servers are one of the many public faces of an organization and one of the most easily targeted. Web servers represent an interesting paradox namely, how do you share information about your organization without giving away the so-called store? Solving this dilemma can be a tough and thankless job; but it's also one of the most important.

Before I get too far, though, let's take a look at some of the threats that your server faces by virtue of being one of the "troops" on the front line.

Now, there are a tremendous number of threats facing a Web server, and many depend on the applications, operating system, and environment you have configured on the system itself. What I have assembled in this section are some of the more generic attacks that your poor server may face.

Denial of service

The denial of service (DoS) attack is one of the real "old-school" attacks that a server can face. The attack is very simple, and nowadays it's carried out by those individuals commonly known as script kiddies, who basically have a low skill level. In a nutshell, a DoS attack is an attack in which one system attacks another with the intent of consuming all the resources on the system (such as bandwidth or processor cycles), leaving nothing behind for legitimate requests. Generally, these attacks have been relegated to the category of annoyance, but don't let that be a reason to lower your guard, because there are plenty of other things to keep you up at night.

Distributed denial of service

The distributed DoS (DDoS) attack is the big brother of the DoS attack and as such is meaner and nastier. The goal of the DDoS attack is to do the same thing as the DoS, but on a much grander and more complex scale. In a DDoS attack, instead of one system attacking another, an attacker uses multiple systems to target a server, and by multiple systems I mean not hundreds or thousands, but more on the order of hundreds of thousands. Where DoS is just an annoyance, a DDoS attack can be

downright deadly, as it can take a server offline quickly. The good news is that the skill level required to pull a DDoS attack off is fairly high.

Some of the more common DDoS attacks include:

- FTP bounce attacks. A File Transfer Protocol (FTP) bounce attack is enacted when an attacker uploads a specially constructed file to a vulnerable FTP server, which in turn forwards it to another location, which generally is another server inside the organization. The file that is forwarded typically contains some sort of payload designed to make the final server do something that the attacker wants it to do.
- Port scanning attack. A port scanning attack is performed through the structured and systematic scanning of a host. For example, someone may scan your Web server with the intention of finding exposed services or other vulnerabilities that can be exploited. This attack can be fairly easily performed with any one of a number of port scanners available freely on the Internet. It also is one of the more common types of attacks, as it is so simple to pull off that script kiddies attempt it just by dropping the host name or IP address of your server (however, they typically don't know how to interpret the results). Keep in mind that a more advanced attacker will use port scanning to uncover information for a later effort.
- Ping flooding attack. A ping flooding attack is a simple DDoS attack in which a computer sends a packet (ping) to another system with the intention of uncovering information about services or systems that are up or down. At the low end, a ping flood can be used to uncover information covertly, but throttle up the packets being sent to a target or victim so that now, the system will go offline or suffer slowdowns. This attack is "old school" but still very effective, as a number of modern operating systems are still susceptible to this attack and can be taken down.
- Smurf attack. This attack is similar to the ping flood attack but with a clever modification to the process. In a Smurf attack, a ping command is sent to an intermediate network, where it is amplified and forwarded to the victim. What was once a single "drop" now becomes a virtual tsunami of traffic? Luckily, this type of attack is somewhat rare.
- SYN flooding. This attack requires some knowledge of the TCP/ IP protocol suite—namely, how the whole communication process works. The easiest way to explain this attack is through an analogy. This attack is the networking equivalent of sending a letter to someone that requires a response, but the letter uses a bogus return address. That individual sends your letter back and waits for your response, but the response never comes, because it went into a black hole some place. Enough SYN requests to the system and an attacker can use all the connections on a system so that nothing else can get through.
- P fragmentation/fragmentation attack. In this attack, an attacker uses advanced knowledge of the TCP/IP protocol to break packets up into smaller pieces, or "fragments", that bypass most intrusion-detection systems. In extreme cases, this type of attack can cause hangs, lock-ups, reboots, blue screens, and other mischief. Luckily, this attack is a tough one to pull off.

Web page defacement

Web page defacement is seen from time to time around the Internet. As the name implies, a Web page defacement results when a Web server is improperly configured, and an attacker uses this flawed configuration to modify Web pages for any number of reasons, such as for fun or to push a political cause.

SQL injection

Structured Query Language (SQL) injections are attacks carried out against databases. In this attack, an attacker uses weaknesses in the design of the database or Web page to extract information or even manipulate information within the database.

Poor coding

Anyone who has been a developer or worked in information technology has seen the problems associated with sloppy or lazy coding practices. Poor coding problems can result from any one of a number of factors, including poor training, new developers, or insufficient quality assurance for an application. At its best, poor coding can be an annoyance, where features don't work as advertised; at its worst, applications can have major security holes.

Shrink-wrapped code

This problem is somewhat related to the above issues with poor coding, but with a twist: Basically, this problem stems from the convenience of obtaining precompiled or pre-written components that can be used as building blocks for your own application, shortening your development cycle. The downside is that the components you're using to help build your application may not have gone through the same vetting process as your in-house code, and applications may have potential problem areas. Additionally, it's not unheard of for developers who don't really know how to analyze the code and understand what it's actually doing to put so-called "shrink-wrapped" components in applications. In at least one case I can think of, I'm aware of a developer using a piece of shrink-wrapped code to provide an authentication mechanism for an application that was actually authenticating users, but also covertly e-mailing the same credentials to a third-party.

Protecting Web servers

1. Separate Web servers for internal and external use. This sounds like a no-brainer, but it still bears repeating. Most organizations have Web-based applications or sites used internally, as well as applications and sites used externally. In an ideal situation, these two sets of servers and content should be kept separate, with internal and external sites having their own servers with as little crossover between them as possible. By splitting systems apart like this, you avoid the probability (or at least lessen the risk) of an attacker breaching a server and getting access to data or even internal systems.

2. Separate development and production servers. In my time, I have known several companies that have violated this rule by letting their development team work on production servers to develop their code or tweak existing code. Typically, this is just a case of extreme laziness—one that can lead to catastrophic problems later when an attacker sees your unpolished code and exploits it to his or her own ends. Also, consider that your own developers may compromise security by testing and tweaking code. Do yourself a favor: Implement a development environment.
3. Regular audits. Any Web server or Web application worth its salt will have some method of generating logs of activity on the system. After this information is logged, make it part of your regular routine to scan the logs for problems, such as application failures or suspicious activity. Keep in mind that an audit log is like evidence collected at a crime scene: It's essentially worthless unless you intend to examine it later.
4. Keep your system up to date. Do I really need to go through this one? Yes, I do. Patching a system is an often-overlooked problem when it really shouldn't be. Ideally, you should keep an eye on whether patches, service packs, updates, or other items become available that can help secure your system. Depending on your hosting platform and other factors, you may have the option of having these updates delivered automatically, or you may have to use the old-fashioned manual delivery method. Also keep in mind that many times, updates are the only way to fix problems such as those related to buffer overflows, network client issues, and so on.
5. Vulnerability scanning. In previous articles (see Resources), I covered the topic of vulnerability scanning as a tool for finding problems in your hosting and application infrastructure. Vulnerability scanning can be a very powerful tool in the ongoing struggle to uncover problems relating to software, such as configuration and patching issues. Another advantage is that these scanning tools are regularly updated, so you can use them to find the latest problems that, in a number of cases, include issues that you may not even be aware of, allowing you to address them before they can be exploited. Tools such as the freeware Nessus (see Resources) can be a great asset to administrators regardless of whether you host on Linux®, UNIX®, or some other platform.
6. Developer training. This one may be a bit more difficult to pull off, but it reaps a tremendous reward, if undertaken. Educating developers on secure coding practices can result in the elimination or reduction of problems associated with sloppy or lazy coding.

Intranet

An intranet is a computer network that uses Internet Protocol technology to share information, operational systems, or computing services within an organization. This term is used in contrast to extranet, a network between organizations, and instead refers to a network within an organization. Sometimes, the term refers only to the organization's internal website, but may be a more extensive part of the organization's information technology infrastructure, and may be composed of multiple local area networks. The objective is to organize each individual's desktop with minimal cost, time and effort to be more productive, cost efficient, timely, and competitive.

An intranet can be understood as a private analog of the Internet, or as a private extension of the Internet confined to an organization. The first intranet websites and home pages were published in 1991, and began to appear in non-educational organizations in 1994.

Intranets are sometimes contrasted to extranets. While intranets are generally restricted to employees of the organization, extranets may also be accessed by customers, suppliers, or other approved parties. Extranets extend a private network onto the Internet with special provisions for authentication, authorization and accounting (AAA protocol).

Uses

Increasingly, intranets are being used to deliver tools, e.g. collaboration (to facilitate working in groups and teleconferencing) or sophisticated corporate directories, sales and customer relationship management tools, project management etc., to advance productivity.

Intranets are also being used as corporate culture-change platforms. For example, large numbers of employees discussing key issues in an intranet forum application could lead to new ideas in management, productivity, quality, and other corporate issues.

In large intranets, website traffic is often similar to public website traffic and can be better understood by using web metrics software to track overall activity. User surveys also improve intranet website effectiveness. Larger businesses allow users within their intranet to access public internet through firewall servers. They have the ability to screen messages coming and going keeping security intact.

When part of an intranet is made accessible to customers and others outside the business, that part becomes part of an extranet. Businesses can send private messages through the public network, using special encryption/decryption and other security safeguards to connect one part of their intranet to another.

Intranet user-experience, editorial, and technology team‘s work together to produce in-house sites. Most commonly, intranets are managed by the communications, HR or CIO departments of large organizations, or some combination of these.

Because of the scope and variety of content and the number of system interfaces, intranets of many organizations are much more complex than their respective public websites. Intranets and their use are growing rapidly. According to the Intranet design annual 2007 from Nielsen Norman Group, the number of pages on participants' intranets averaged 200,000 over the years 2001 to 2003 and has grown to an average of 6 million pages over 2005–2007.

Benefits

- Workforce productivity: Intranets can help users to locate and view information faster and use applications relevant to their roles and responsibilities. With the help of a web browser interface, users can access data held in any database the organization wants to make available, anytime and subject to security provisions from anywhere within the company workstations, increasing employees' ability to perform their jobs faster, more accurately, and with confidence that they have the right information. It also helps to improve the services provided to the users.
- Time: Intranets allow organizations to distribute information to employees on an as-needed basis; Employees may link to relevant information at their convenience, rather than being distracted indiscriminately by email.
- Communication: Intranets can serve as powerful tools for communication within an organization, vertically strategic initiatives that have a global reach throughout the organization. The type of information that can easily be conveyed is the purpose of the initiative and what the initiative is aiming to achieve, who is driving the initiative, results achieved to date, and who to speak to for more information. By providing this information on the intranet, staff have the opportunity to keep up-to-date with the strategic focus of the organization. Some examples of communication would be chat, email, and/or blogs. A great real world example of where an intranet helped a company communicate is when Nestle had a number of food processing plants in Scandinavia. Their central support system had to deal with a number of queries every day. When Nestle decided to invest in an intranet, they quickly realized the savings. McGovern says the savings from the reduction in query calls was substantially greater than the investment in the intranet.
- Web publishing allows cumbersome corporate knowledge to be maintained and easily accessed throughout the company using hypermedia and Web technologies.[7] Examples include: employee manuals, benefits documents, company policies, business standards, news feeds, and even training, can be accessed using common Internet standards (Acrobat files, Flash files, CGI applications). Because each business unit can update the online copy of a document, the most recent version is usually available to employees using the intranet.
- Business operations and management: Intranets are also being used as a platform for developing and deploying applications to support business operations and decisions across the internetworked enterprise.
- Cost-effective: Users can view information and data via web-browser rather than maintaining physical documents such as procedure manuals, internal phone list and requisition forms. This can potentially save the business money on printing, duplicating documents, and the environment as well as document maintenance overhead. For example, the HRM company PeopleSoft "derived significant cost savings by shifting HR processes to the intranet". McGovern goes on to say the manual cost of enrolling in benefits was found to be USD109.48 per enrollment. "Shifting this process to the intranet reduced the cost per enrollment to $21.79; a saving of 80 percent". Another company that saved money on expense reports was Cisco.

- Enhance collaboration: Information is easily accessible by all authorised users, which enables teamwork.
- Cross-platform capability: Standards-compliant web browsers are available for Windows, Mac, and UNIX.
- Built for one audience: Many companies dictate computer specifications which, in turn, may allow Intranet developers to write applications that only have to work on one browser (no cross-browser compatibility issues). Promote common corporate culture: Every user has the ability to view the same information within the Intranet.
- Immediate updates: When dealing with the public in any capacity, laws, specifications, and parameters can change. Intranets make it possible to provide your audience with "live" changes so they are kept up-to-date, which can limit a company's liability.
- Supports a distributed computing architecture: The intranet can also be linked to a company‘s management information system, for example a time keeping system.

Planning and creation

Most organizations devote considerable resources into the planning and implementation of their intranet as it is of strategic importance to the organization's success. Some of the planning would include topics such as:

- The purpose and goals of the intranet
- Persons or departments responsible for implementation and management
- Functional plans, information architecture, page layouts, design
- Implementation schedules and phase-out of existing systems
- Defining and implementing security of the intranet
- How to ensure it is within legal boundaries and other constraints
- Level of interactivity (e.g. wikis, on-line forms) desired.
- Is the input of new data and updating of existing data to be centrally controlled or devolved

These are in addition to the hardware and software decisions (like content management systems), participation issues (like good taste, harassment, confidentiality), and features to be supported. Intranets are often static sites. Essentially they are a shared drive, serving up centrally stored documents alongside internal articles or communications (often one-way communication). However organisations are now starting to think of how their intranets can become a 'communication hub' for their team by using companies specialising in 'socialising' intranets. The actual implementation would include steps such as:

- Securing senior management support and funding.
- Business requirements analysis.
- Identify users' information needs.
- Installation of web server and user access network.
- Installing required user applications on computers.

- Ongoing measurement and evaluation, including through benchmarking against other intranets.

Another useful component in an intranet structure might be key personnel committed to maintaining the Intranet and keeping content current. For feedback on the intranet, social networking can be done through a forum for users to indicate what they want and what they do not like.

Intranet software

Microsoft SharePoint is the dominant software used for creating intranets. Estimates indicate that around 50% of all intranets are developed using SharePoint, however there are many alternatives. Other intranet software includes:

- Autonomy Corporation
- Atlassian Confluence
- Bitrix24
- Drupal
- eXo Platform
- Google Sites
- Igloo Software
- IBM Websphere
- Interact Intranet
- Hyperoffice
- Jive Software
- Joomla
- Liferay
- Lotus Notes
- OpenText
- Oracle Fusion Middleware
- Plone (software)
- SAP NetWeaver Portal
- Sitecore
- ThoughtFarmer
- WordPress
- Yammer

wireless network

A wireless network is any type of computer network that uses wireless data connections for connecting network nodes.

Wireless networking is a method by which homes, telecommunications networks and enterprise (business) installations avoid the costly process of introducing cables into a building, or as a connection between various equipment locations. Wireless telecommunications networks are generally implemented and administered using radio communication. This implementation takes place at the physical level (layer) of the OSI model network structure.

Examples of wireless networks include cell phone networks, Wi-Fi local networks and terrestrial microwave networks.Computers are very often connected to networks using wireless links

- Terrestrial microwave –Terrestrial microwave communication uses Earth-based transmitters and receivers resembling satellite dishes. Terrestrial microwaves are in the low-gigahertz range, which limits all communications to line-of-sight. Relay stations are spaced approximately 48 km (30 mi) apart.

- Communications satellites – Satellites communicate via microwave radio waves, which are not deflected by the Earth's atmosphere. The satellites are stationed in space, typically in geosynchronous orbit 35,400 km (22,000 mi) above the equator. These Earth-orbiting systems are capable of receiving and relaying voice, data, and TV signals.

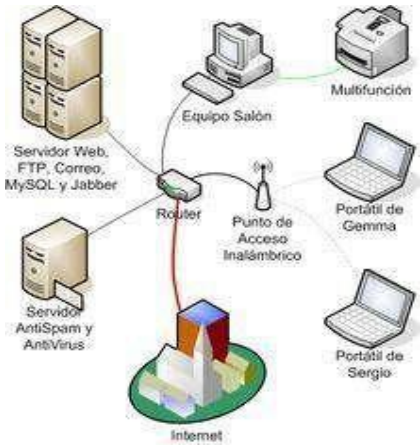
- Cellular and PCS systems use several radio communications technologies. The systems divide the region covered into multiple geographic areas. Each area has a low-power transmitter or radio relay antenna device to relay calls from one area to the next area.

- Radio and spread spectrum technologies – Wireless local area networks use a high-frequency radio technology similar to digital cellular and a low-frequency radio technology. Wireless LANs use spread spectrum technology to enable communication between multiple devices in a limited area. IEEE 802.11 defines a common flavor of open-standards wireless radio-wave technology known as Wifi.

Types of wireless networks

1. Wireless PAN

Wireless personal area networks (WPANs) interconnect devices within a relatively small area, that is generally within a person's reach.[3] For example, both Bluetooth radio and invisible infrared light provides a WPAN for interconnecting a headset to a laptop. ZigBee also supports WPAN applications. Wi-Fi PANs are becoming commonplace (2010) as equipment designers start to integrate Wi-Fi into a variety of consumer electronic devices. Intel "My WiFi" and Windows 7



"virtual Wi-Fi" capabilities have made Wi-Fi PANs simpler and easier to set up and configure.

2. Wireless LAN

Wireless LANs are often used for connecting to local resources and to the Internet

A wireless local area network (WLAN) links two or more devices over a short distance using a wireless distribution method, usually providing a connection through an access point for Internet access. The use of spread-spectrum or OFDM technologies may allow users to move around within a local coverage area, and still remain connected to the network.

Products using the IEEE 802.11 WLAN standards are marketed under the Wi-Fi brand name. Fixed wireless technology implements point-to-point links between computers or networks at two distant locations, often using dedicated microwave or modulated laser light beams over line of sight paths. It is often used in cities to connect networks in two or more buildings without installing a wired link.

3. Wireless mesh network

A wireless mesh network is a wireless network made up of radio nodes organized in a mesh topology. Each node forwards messages on behalf of the other nodes. Mesh networks can "self heal", automatically re-routing around a node that has lost power.

4. Wireless MAN

Wireless metropolitan area networks are a type of wireless network that connects several wireless LANs.

- WiMAX is a type of Wireless MAN and is described by the IEEE 802.16 standard.

Wireless WAN

Wireless wide area networks are wireless networks that typically cover large areas, such as between neighboring towns and cities, or city and suburb. These networks can be used to connect branch offices of business or as a public internet access system. The wireless connections between access points are usually point to point microwave links using parabolic dishes on the 2.4 GHz band, rather than omnidirectional antennas used with smaller networks. A typical system contains base station gateways, access points and wireless bridging relays. Other configurations are mesh systems where each access point acts as a relay also. When combined with renewable energy systems such as photo-voltaic solar panels or wind systems they can be stand alone systems.

Cellular network

A cellular network or mobile network is a radio network distributed over land areas called cells, each served by at least one fixed-location transceiver, known as a cell site or base station. In a cellular network, each cell characteristically uses a different set of radio frequencies from all their immediate neighboring cells to avoid any interference.

Although originally intended for cell phones, with the development of smart phones, cellular telephone networks routinely carry data in addition to telephone conversations:

- Global System for Mobile Communications (GSM): The GSM network is divided into three major systems: the switching system, the base station system, and the operation and support system. The cell phone connects to the base system station which then connects to the operation and support station; it then connects to the switching station where the call is transferred to where it needs to go. GSM is the most common standard and is used for a majority of cell phones.
- Personal Communications Service (PCS): PCS is a radio band that can be used by mobile phones in North America and South Asia. Sprint happened to be the first service to set up a PCS.
- D-AMPS: Digital Advanced Mobile Phone Service, an upgraded version of AMPS, is being phased out due to advancement in technology. The newer GSM networks are replacing the older system.

Global area network

A global area network (GAN) is a network used for supporting mobile across an arbitrary number of wireless LANs, satellite coverage areas, etc. The key challenge in mobile communications is handing off user communications from one local coverage area to the next. In IEEE Project 802, this involves a succession of terrestrial wireless LANs.

Wireless Network Elements

The telecommunications network at the physical layer also consists of many interconnected wireline Network Elements (NEs). These NEs can be stand-alone systems or products that are either supplied by a single manufacturer, or are assembled by the service provider (user) or system integrator with parts from several different manufacturers.

Wireless NEs are products and devices used by a wireless carrier to provide support for the backhaul network as well as a Mobile Switching Center (MSC).

Reliable wireless service depends on the network elements at the physical layer to be protected against all operational environments and applications (see GR-3171, Generic Requirements for Network Elements Used in Wireless Networks - Physical Layer Criteria).

What are especially important are the NEs that are located on the cell tower to the Base Station (BS) cabinet. The attachment hardware and the positioning of the antenna and associated closures/cables are required to have adequate strength, robustness, corrosion resistance, and rain/solar resistance for expected wind, storm, ice, and other weather conditions. Requirements for individual components, such as hardware, cables, connectors, and closures, shall take into consideration the structure to which they are attached.

Difficulties

Interference

Compared to wired systems, wireless networks are frequently subject to electromagnetic interference. This can be caused by other networks or other types of equipment that generate radio waves that are within, or close, to the radio bands used for communication. Interference can degrade the signal or cause the system to fail.

Absorption and reflection

Some materials cause absorption of electromagnetic waves, preventing it from reaching the receiver, in other cases, particularly with metallic or conductive materials reflection occurs. This can cause dead zones where no reception is available. Aluminium foiled thermal isolation in modern homes can easily reduce indoor mobile signals by 10 dB frequently leading to complaints about bad reception of long distance rural cell signals.

Multipath fading

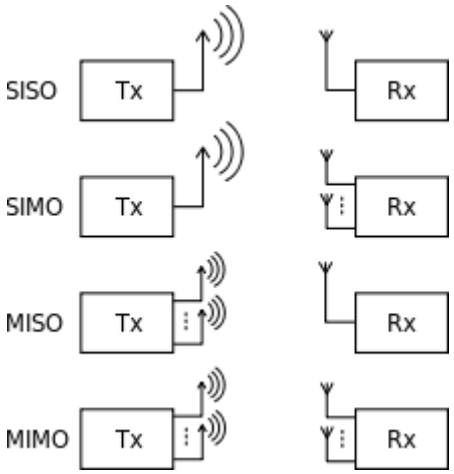
In multipath fading two or more different routes taken by the signal, due to reflections, can cause the signal to cancel out at certain locations, and to be stronger in other places (upfade).

Hidden node problem

The hidden node problem occurs in some types of network when a node is visible from a wireless access point (AP), but not from other nodes communicating with that AP. This leads to difficulties in media access control.

Shared resource problem

The wireless spectrum is a limited resource and shared by all nodes in the range of its transmitters. Bandwidth allocation becomes complex with multiple participating users. Often users are not aware that advertised numbers (e.g., for IEEE 802.11 equipment or LTE networks) are not their capacity, but shared with all other users and thus the individual user rate is far lower. With increasing demand, the capacity crunch is more and more likely to happen. User-in-the-loop (UIL) may be an alternative solution to ever upgrading to newer technologies for over-provisioning.



Capacity Channel

Understanding of SISO, SIMO, MISO and MIMO. Using multiple antennas and transmitting in different frequency channels can reduce fading, and can greatly increase the system capacity.

The maximum data rate of any single wireless link can be described by the Shannon's theorem which is related to the bandwidth in hertz, and the noise on the channel.

Network

The total network bandwidth depends on how dispersive the medium is (more dispersive medium generally has better total bandwidth because it minimises interference), how many frequencies are available, how noisy those frequencies are, how many aerials are used and whether directional antenna are in use, whether nodes employ power control and so on.

Cellular wireless networks generally have good capacity, due to their use of directional aerials, and their ability to reuse radio channels in non-adjacent cells. Additionally, cells can be made very small using low power transmitters, and this fact is used in cities to give network capacity that scales linearly with population density.

Software audit

A software audit review, or software audit, is a type of software review in which one or more auditors who are not members of the software development organization conduct "An independent examination of a software product, software process, or set of software processes to assess compliance with specifications, standards, contractual agreements, or other criteria".

"Software product" mostly, but not exclusively, refers to some kind of technical document. IEEE Std. 1028 offers a list of 32 "examples of software products subject to audit", including documentary products such as various sorts of plan, contracts, specifications, designs, procedures, standards, and reports, but also non-documentary products such as data, test data, and deliverable media.

Objectives and participants

"The purpose of a software audit is to provide an independent evaluation of conformance of software products and processes to applicable regulations, standards, guidelines, plans, and procedures".[2] The following roles are recommended:

- The Initiator (who might be a manager in the audited organization, a customer or user representative of the audited organization, or a third party), decides upon the need for an audit, establishes its purpose and scope, specifies the evaluation criteria, identifies the audit personnel, decides what follow-up actions will be required, and distributes the audit report.
- The Lead Auditor (who must be someone "free from bias and influence that could reduce his

ability to make independent, objective evaluations") is responsible for administrative tasks such as preparing the audit plan and assembling and managing the audit team, and for ensuring that the audit meets its objectives.
- The Recorder documents anomalies, action items, decisions, and recommendations made by the audit team.
- The Auditors (who must be, like the Lead Auditor, free from bias) examine products defined in the audit plan, document their observations, and recommend corrective actions. (There may be only a single auditor.)
- The Audited Organization provides a liaison to the auditors, and provides all information requested by the auditors. When the audit is completed, the audited organization should implement corrective actions and recommendations.

Three Critical Kinds of Software Audit

- There are many ways to ―audit‖ a software application. Indeed the most basic kinds of software audit examine how the software is functionally configured, integrated or utilized within an organization. This kind of review process can be completed either by internal IT, an outside firm or an independent solution provider – typically as a first step in a bigger development project. However the stakes are much higher in three other classes of software audit – with the first type often instilling confidence and the other two, anxiety.

- Software Quality Assurance Audit - The first kind of software audit is part of the software quality assurance (QA) process. The objective of a QA audit is simple – to improve the software. Everything is fair game in a software review – including code, processes, report output, data, test data and media - and anyone close to the software development organization may be asked to conduct the software QA audit. The goal is to assess technical quality, form and function with the aim of improving aspects such as ease-of use, reliability, security and performance.

- Software Licensing Audit – Finally, software can be audited as part of Software Asset Management or Risk Management practices to determine where the software is distributed and how it is used. A license audit may be required to impose greater controls or find cost savings. The audit may seek to enforce software copyright protections. It can be mandated by the courts as part of a legal dispute. It can be ordered by risk managers who seek to determine the organization‘s level of exposure from continued use of the software.

- The Who, What and Why of Software Audits: Tools, Teams and How to Prepare

- Every kind of software audit essentially seeks to understand the same things. What is the true purpose of the software and its value to the organization? How does it perform, weighed against necessary risk? Likewise, most software audits assign similar roles to participants and rely on technological tools to aid examination.

- Software Audit Team – It takes a team to complete a software audit, and it requires the active participation of the organization. The internal Sponsor or Initiator establishes the need for the software audit, the proper participants, their purpose and scope, evaluation criteria and reporting mechanisms. The Lead Auditor is typically an outside examiner free from bias and influence who can make objective evaluations. This person leads the independent auditing team that actually conducts the software review according to audit objectives. Finally, the person responsible for administrative tasks such as documenting action items, decisions, recommendations and reports is called the Recorder. When the software audit is completed, the audited organization implements corrective actions and recommendations.

- Software Audit Tools – Selecting the right tool for the job cannot be understated. Different software audit tools will generate different views of an organization‘s applications and architecture. Make sure that the audit team includes an expert at using the tool of choice, and that it will return sufficient data to determine appropriate actions. For example, software‘s compliance with application security can be audited using a variety of static analysis and dynamic analysis tools that analyze an application and score its conformance with security standards, guidelines and best practices. Lastly, the software auditing tool should report its findings as part of a benchmarking process for future audits by the audit team.

- Prepare for a Software Audit – Chances are most IT organizations will be subject to some type of software audit. The key to surviving the process is organization. For companies that are unprepared, any software audit can become a painful, lengthy exercise requiring countless man-hours. Budgeting for potential audits in advance will avoid surprise expenses that could impact profitability. As examples: annual software compliance audits are a common occurrence in highly regulated industries such as finance and healthcare. Companies undergoing mergers or acquisitions should expect software license audit requests from vendors and suppliers. Software development teams should plan on application security testing as part of their standard QA process. Organizations that are well prepared can not only survive a software audit but improve the quality, compliance and utilization of their software as a result.

Ethics in IT

Policy Statement

John Carroll University provides information technology resources to allow faculty, staff, and students to pursue the University‘s educational mission, which includes teaching, learning, service, research and administration. Thus, Information Technology Resources (―IT Resources‖), as defined in this policy, must be used in a manner that furthers the University‘s mission.

Any access or use of information technology resources that conflicts with this Information Technology Resources Policy (―Policy‖ or ―IT Policy‖) or any other University policy is not acceptable and will be considered a violation of this Policy. Additionally, any activity that interferes, interrupts, compromises, or conflicts with the safe and efficient use of IT Resources is considered a violation of this Policy. This Policy shall apply to all Users including, but not limited to, students, employees (faculty and staff), guests, affiliates, vendors and independent contractors. Use of IT Resources, even when carried out on a privately owned computer that is not managed or maintained by the University, is governed by this Policy. This Policy supersedes any existing policies and procedures that are in conflict with the terms of this Policy.

Purpose

The purpose of this Policy is to ensure an information technology infrastructure that promotes the basic mission and purpose of the University in teaching, learning, service, research and administration, and to ensure compliance with all applicable laws. It also provides notice, to all who use and manage IT Resources, of the University‘s expectations and regulations.

Uses

Users are responsible for the protection of University assets and for the accuracy, integrity and confidentiality of the information to which they have access. Users are expected to uphold the standards and principles of the University while using IT Resources. Accordingly, users are prohibited from using any portion of IT Resources to post or transmit any information, Data, text, file, link, software, chat, communication or other content that is harmful, abusive, discriminatory, hostile, combative, threatening, insulting, embarrassing, harassing, intimidating, defamatory, pornographic, obscene, or which negatively affects the University, another User, or any third party. Users who do not respect this Spirit of Use may be held in violation of this IT Policy.

Definitions

Data. All information that is used by or belongs to the University, or that is processed, stored, posted, maintained, transmitted, copied on, or copied from IT Resources.

Functional Unit(s). The department, office, operating division, program, vendor, entity or defined unit of the University that has been authorized to access or use IT Resources.

IT Resource(s).

User. Any individual who uses, accesses or otherwise employs, locally or remotely, IT Resources, whether individually controlled, shared, stand-alone, or networked, and with or without authorization, is considered a User under this Policy.

Sensitive Data. Data designated as private or confidential by law or by the University. Sensitive Data includes, but is not limited to, employment records, medical records, student records, education records, personal financial records (or other personally identifiable information), research Data, trade secrets, classified government information, proprietary information of the University or any Data that could harm the legitimate financial and reputational interests of the University if unauthorized access is permitted, whether intentionally or unintentionally. Sensitive Data shall not include records that by law must be made available to the general public.

Policy Elaboration

Access to some IT Resources is restricted to specific positions or units as determined by the appropriate functional unit head. Functional unit heads should determine and authorize the appropriate degree of access for each member of their units, and should provide unit members with adequate orientation and training regarding the appropriate use of all IT Resources. Using IT Resources outside of the scope of access granted by the University or attempting to exceed restrictions on access is a serious violation of this Policy and may potentially lead to criminal prosecution.

Technical and Content-Based Restrictions. The University reserves the right to impose technical restrictions on the access to its network in ways that may disrupt the ability to utilize certain devices, programs, and protocols. Additionally, the University expressly reserves the right to impose content-based restrictions on the use of its IT Resources. Such restrictions may be necessary to protect the University and its constituents. The University recognizes that academic freedom and the freedom of inquiry are important values that may be hindered by an overzealous restriction of content. Therefore, any content-based restriction scheme imposed on IT Resources will require appropriate Vice President Authorization.

Access Codes - Users must take precautions to prevent unauthorized use of their access codes (passwords). Users will be held accountable for all actions performed under their access codes, including those performed by other individuals as a result of negligence in protecting the codes.

Privacy - Users are obligated to respect the privacy that other Users have in their own systems, Data, and accounts. Thus, it is a violation of this Policy for any User to engage in electronic ―snooping,‖ or to employ IT Resources for the purpose of ―prying into‖ the affairs of others, i.e., to access or attempt to access electronic files, or to install/utilize image/audio recording devices, without proper authorization to do so for genuine business purposes of the University.

Sensitive Data - IT Resources containing Sensitive Data should be restricted based upon a need to know basis and should be guarded against both internal and external breaches. Thus, IT Resources containing Sensitive Data protected under either state or federal law should be controlled and protected in a manner that meets all pertinent legal requirements.

To the extent there is any uncertainty as to whether any Data constitutes Sensitive Data, it shall be treated as Sensitive Data until a determination is made by the CIO and Functional Unit head, in consultation with the University‘s Office of Legal Affairs.

Violation of Law - Users are responsible for respecting and adhering to University policies and to local, state, and federal laws. Any use of IT Resources in violation of civil or criminal law at the federal, state, or local levels is prohibited. Examples of such use include but are not limited to:

- promoting a pyramid scheme; distributing illegal obscenity; receiving, transmitting, or
- possessing child pornography; infringing copyrights; exceeding authorized access; and
- Making bomb or other threats.

Intellectual Property Rights - The University takes the issue of intellectual property and similar rights seriously. Accordingly, the University requires every User to adhere to a strict policy of respecting intellectual property rights.

Copyright: With respect to copyright infringement, Users should be aware that copyright law governs (among other activities) the copying, display, and use of software and other works in digital form (text, sound, images, and other multimedia). All copyrighted information, such as text and images, retrieved from IT Resources or stored, transmitted or maintained with IT Resources, must be used in conformance with applicable copyright and other laws. Copied material, used legally, must be properly attributed in conformance with applicable legal and professional standards.

Software: Software may not be copied, installed or used on IT Resources except as permitted by the owner of the software and by law. Software subject to licensing must be properly licensed and all license provisions (including installation, use, copying, number of simultaneous Users, terms of the license, etc.) must be strictly followed. All software licensing is administered under the auspices of ITS.

Fair use: The law permits use of copyrighted material without authorization from the copyright holder for some educational purposes (protecting certain classroom practices and ―fair use,‖ for example), but an educational purpose does not automatically mean that the use is permitted without authorization.

Ownership: All IT Resources developed by University employees, students, and contractors for use by the University, or as part of their normal employment activities, are considered ―works for hire‖. As such, the University is considered the ―author‖ and owner of these resources. This Policy does not alter the University‘s position or policy on intellectual property ownership for faculty and research Data.

Reporting Infringement: It is the responsibility of every User to avoid infringing any intellectual property right and to report the infringement of another User if and when it is discovered. Failure to respect such rights, or report infringements, is a violation of this IT Policy and subject to appropriate sanctions.

User interface

The user interface, in the industrial design field of human–machine interaction, is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operator‘s decision making process. Examples of this broad concept of user interfaces include the interactive aspects of computer operating systems, hand tools, heavy machinery operator controls, and process controls. The design considerations applicable when creating user interfaces are related to or involve such disciplines as ergonomics and psychology.

Generally, the goal of human-machine interaction engineering is to produce a user interface which makes it easy (self explanatory), efficient, and enjoyable (user friendly) to operate a machine in the way which produces the desired result. This generally means that the operator needs to provide minimal input to achieve the desired output, and also that the machine minimizes undesired outputs to the human.

With the increased use of personal computers and the relative decline in societal awareness of heavy machinery, the term user interface is generally assumed to mean the graphical user interface, while industrial control panel and machinery control design discussions more commonly refer to human-machine interfaces.

Quality

All great interfaces share eight qualities or characteristics:

1. Clarity The interface avoids ambiguity by making everything clear through language, flow, hierarchy and metaphors for visual elements.
2. Concision it‘s easy to make the interface clear by over-clarifying and labeling everything, but this leads to interface bloat, where there is just too much stuff on the screen at the same time. If too many things are on the screen, finding what you‘re looking for is difficult, and so the interface becomes tedious to use. The real challenge in making a great interface is to make it concise and clear at the same time.
3. Familiarity even if someone uses an interface for the first time, certain elements can still be familiar. Real-life metaphors can be used to communicate meaning.
4. Responsiveness a good interface should not feel sluggish. This means that the interface should provide good feedback to the user about what‘s happening and whether the user‘s input is being successfully processed.
5. Consistency keeping your interface consistent across your application is important because it allows users to recognize usage patterns.
6. Aesthetics While you don‘t need to make an interface attractive for it to do its job, making something look good will make the time your users spend using your application more enjoyable; and happier users can only be a good thing.

7. Forgiveness A good interface should not punish users for their mistakes but should instead provide the means to remedy them.

Types

- Direct manipulation interface is the name of a general class of user interfaces that allow users to manipulate objects presented to them, using actions that correspond at least loosely to the physical world.
- Graphical user interfaces (GUI) accept input via devices such as a computer keyboard and mouse and provide articulated graphical output on the computer monitor. There are at least two different principles widely used in GUI design: Object-oriented user interfaces (OOUIs) and application oriented interfaces.[13]
- Web-based user interfaces or web user interfaces (WUI) that accept input and provide output by generating web pages which are transmitted via the Internet and viewed by the user using a web browser program. Newer implementations utilize Java, JavaScript, AJAX, Adobe Flex, Microsoft .NET, or similar technologies to provide real-time control in a separate program, eliminating the need to refresh a traditional HTML based web browser. Administrative web interfaces for web-servers, servers and networked computers are often called control panels.
- Touch screens are displays that accept input by touch of fingers or a stylus. Used in a growing amount of mobile devices and many types of point of sale, industrial processes and machines, self-service machines etc.
- Command line interfaces, where the user provides the input by typing a command string with the computer keyboard and the system provides output by printing text on the computer monitor. Used by programmers and system administrators, in engineering and scientific environments, and by technically advanced personal computer users.
- Touch user interface are graphical user interfaces using a touchpad or touch screen display as a combined input and output device. They supplement or replace other forms of output with hepatic feedback methods. Used in computerized simulators etc.
- Hardware interfaces are the physical, spatial interfaces found on products in the real world from toasters, to car dashboards, to airplane cockpits. They are generally a mixture of knobs, buttons, sliders, switches, and touch screens.
- Attentive user interfaces manage the user attention deciding when to interrupt the user, the kind of warnings, and the level of detail of the messages presented to the user.
- Batch interfaces are non-interactive user interfaces, where the user specifies all the details of the batch job in advance to batch processing, and receives the output when all the processing is done. The computer does not prompt for further input after the processing has started.
- Conversational Interface Agents attempt to personify the computer interface in the form of an animated person, robot, or other character (such as Microsoft's Clippy the paperclip), and present interactions in a conversational form.
- Crossing-based interfaces are graphical user interfaces in which the primary task consists in crossing boundaries instead of pointing.

- Intelligent user interfaces are human-machine interfaces that aim to improve the efficiency, effectiveness, and naturalness of human-machine interaction by representing, reasoning, and acting on models of the user, domain, task, discourse, and media (e.g., graphics, natural language, gesture).
- Motion tracking interfaces monitor the user's body motions and translate them into commands, currently being developed by Apple.
- Multi-screen interfaces employ multiple displays to provide a more flexible interaction. This is often employed in computer game interaction in both the commercial arcades and more recently the handheld markets.
- Non-command user interfaces, which observe the user to infer his / her needs and intentions, without requiring that he / she formulate explicit commands.
- Object-oriented user interfaces (OOUI) are based on object-oriented programming metaphors, allowing users to manipulate simulated objects and their properties.
- Reflexive user interfaces where the users control and redefine the entire system via the user interface alone, for instance to change its command verbs. Typically this is only possible with very rich graphic user interfaces.
- Tangible User Interfaces, which place a greater emphasis on touch and physical environment or its element.
- Task-Focused Interfaces are user interfaces which address the information overload problem of the desktop metaphor by making tasks, not files, the primary unit of interaction.
- Text-based user interfaces are user interfaces which output a text. TUIs can either contain a command-line interface or a text-based WIMP environment.
- Voice user interfaces, which accept input and provide output by generating voice prompts. The user input is made by pressing keys or buttons, or responding verbally to the interface.
- Natural-language interfaces – Used for search engines and on web pages. User types in a question and waits for a response.
- Zero-Input interfaces get inputs from a set of sensors instead of querying the user with input dialogs.
- Zooming user interfaces are graphical user interfaces in which information objects are represented at different levels of scale and detail, and where the user can change the scale of the viewed area in order to show more detail.

Reporting
Management reporting systems

Information systems support all levels of management, from those in charge of short-term schedules and budgets for small work groups to those concerned with long-term plans and budgets for the entire organization. Management reporting systems provide routine, detailed, and voluminous information reports specific to each manager‘s areas of responsibility. These systems are typically used by first-level supervisors. Generally, such reports focus on past and present activities, rather than projecting future performance.

Decision support systems and business intelligence

All information systems support decision making, however indirectly, but decision support systems are expressly designed for this purpose. As these systems have been developed to analyze massive collections of data, they have also become known as business intelligence applications. The two principal varieties of decision support systems are model-driven and data-driven.

In a model-driven decision support system, a preprogrammed model is applied to a relatively limited data set, such as a sales database for the present quarter. During a typical session, an analyst or sales manager will conduct a dialog with this decision support system by specifying a number of what-if scenarios. For example, in order to establish a selling price for a new product, the sales manager may use a marketing decision support system. Such a system contains a model relating various factors the price of the product, the cost of goods, and the promotion expense in various media to the projected sales volume over the first five years on the market. By supplying different product prices to the model, the manager can compare predicted results and select the most profitable selling price.

The primary objective of data-driven business intelligence systems is to analyze large pools of data, accumulated over long periods of time in data warehouses, in a process known as data mining. Data mining aims to discover significant patterns, such as sequences (buying a new house, followed by a new dinner table), clusters, and correlations (large families and van sales), with which decisions can be made. Predictive data mining attempts to forecast future outcomes based on the discovered trends. Data-driven decision support systems include a variety of statistical models and may rely on various artificial intelligence techniques, such as expert systems, neural networks, and machine learning. In addition to mining numeric data, text mining is conducted on large aggregates of unstructured data, such as the contents of social media that include social networks, wikis, blogs, and micro blogs. As used in electronic commerce, for example, text mining helps in finding buying trends, targeting advertisements, and detecting fraud.

An important variety of decision support systems enables a group of decision makers to work together without necessarily being in the same place at the same time. These group decision systems include software tools for brainstorming and reaching consensus.

Executive information systems

Executive information systems make a variety of critical information readily available in a highly summarized and convenient form, typically via a graphical digital dashboard. Senior managers characteristically employ many informal sources of information, however, so that formal, computerized information systems are only of partial assistance. Nevertheless, this assistance is important for the chief executive officer, senior and executive vice presidents, and the board of directors to monitor the performance of the company, assess the business environment, and develop

strategic directions for the future. In particular, these executives need to compare their organization‘s performance with that of its competitors and investigate general economic trends in regions or countries. Often individualized and relying on multiple media formats, executive information systems give their users an opportunity to ―drill down‖ from summary information to increasingly focused details.

Acquiring information systems and services

Information systems are a major corporate asset, with respect both to the benefits they provide and to their high costs. Therefore, organizations have to plan for the long term when acquiring information systems and services that will support business initiatives. On the basis of long-term corporate plans and the requirements of various individuals from data workers to top management, essential applications are identified and project priorities are set. For example, certain projects may have to be carried out immediately to satisfy a new government reporting regulation or to interact with a new customer‘s information system. Other projects may be given a higher priority because of their strategic role or greater expected benefits.

Once the need for a specific information system has been established, the system has to be acquired. This is generally done in the context of the already existing information systems architecture of the firm. The acquisition of information systems can either involve external sourcing or rely on internal development or modification. With today‘s highly developed IT industry, companies tend to acquire information systems and services from specialized vendors. The principal tasks of information systems specialists involve modifying the applications for their employer‘s needs and integrating the applications to create coherent systems architecture for the firm. Generally, only smaller applications are developed internally. Certain applications of a more personal nature may be developed where the programming environment supports simple end-user enhancement.

CHAPTER-5

NEW IT INITATIVES

Role of Information Management in ERP

Enterprise resource planning (ERP) is business management software usually a suite of integrated applications that a company can use to collect, store, manage and interpret data from many business activities, including:

- Product planning, cost
- Manufacturing or service delivery
- Marketing and sales
- Inventory management
- Shipping and payment

ERP provides an integrated view of core business processes, often in real-time, using common databases maintained by a database management system. ERP systems track business resources cash, raw materials, production capacity and the status of business commitments: orders, purchase orders, and payroll. The applications that make up the system share data across the various departments (manufacturing, purchasing, sales, accounting, etc.) that provide the data. ERP facilitates information flow between all business functions, and manages connections to outside stakeholders.[

Functional areas

An ERP system covers the following common functional areas. In many ERP systems these are called and grouped together as ERP modules:

- Financial accounting: General ledger, fixed asset, payables including vouchering, matching and payment, receivables cash application and collections, cash management, financial consolidation
- Management accounting: Budgeting, costing, cost management, activity based costing
- Human resources: Recruiting, training, fostering, payroll, benefits, 401K, diversity management, retirement, separation
- Manufacturing: Engineering, bill of materials, work orders, scheduling, capacity, workflow management, quality control, manufacturing process, manufacturing projects, manufacturing flow, product life cycle management
- Order Processing: Order to cash, order entry, credit checking, pricing, available to promise, inventory, shipping, sales analysis and reporting, sales commissioning.
- Supply chain management: Supply chain planning, supplier scheduling, product configuration, order to cash, purchasing, inventory, claim processing, warehousing (receiving, put away, picking and packing).

- Customer relationship management: Sales and marketing, commissions, service, customer contact, call center support - CRM systems are not always considered part of ERP systems but rather Business Support systems (BSS).
- Data services : Various "self–service" interfaces for customers, suppliers and/or employees

Components

- Transactional database
- Management portal/dashboard
- Business intelligence system
- Customizable reporting
- Resource planning and scheduling
- Analyzing the product
- External access via technology such as web services
- Search
- Document management
- Messaging/chat/wiki
- Workflow management

Connectivity to plant floor information

ERP systems connect to real time data and transaction data in a variety of ways. These systems are typically configured by systems integrators, who bring unique knowledge on process, equipment, and vendor solutions.

Direct integration ERP systems have connectivity (communications to plant floor equipment) as part of their product offering. This requires that the vendors offer specific support for the plant floor equipment their customers operate. ERP vendors must be experts in their own products and connectivity to other vendor products, including those of their competitors.

Database integration ERP systems connect to plant floor data sources through staging tables in a database. Plant floor systems deposit the necessary information into the database. The ERP system reads the information in the table. The benefit of staging is that ERP vendors do not need to master the complexities of equipment integration. Connectivity becomes the responsibility of the systems integrator.

Enterprise appliance transaction modules (EATM)-These devices communicate directly with plant floor equipment and with the ERP system via methods supported by the ERP system. EATM can employ a staging table, Web Services, or system–specific program interfaces (APIs). An EATM offers the benefit of being an off–the–shelf solution.

Long term costs can be minimized through careful system testing and thorough documentation. Custom integrated solutions typically run on workstation or server-class computers.

Implementation

ERP's scope usually implies significant changes to staff work processes and practices. Generally, three types of services are available to help implement such changes consulting, customization, and support. Implementation time depends on business size, number of modules, customization, the scope of process changes, and the readiness of the customer to take ownership for the project. Modular ERP systems can be implemented in stages. The typical project for a large enterprise takes about 14 months and requires around 150 consultants. Small projects can require months; multinational and other large implementations can take years. Customization can substantially increase implementation times.

Besides that, information processing influences various business functions e.g. some large corporations like Wal-Mart use a just in time inventory system. This reduces inventory storage and increases delivery efficiency, and requires up-to-date-data. Before 2014, Walmart used a system called Inforem developed by IBM to manage replenishment.

Process preparation

Implementing ERP typically requires changes in existing business processes. Poor understanding of needed process changes prior to starting implementation is a main reason for project failure. The problems could be related to the system, business process, infrastructure, training, or lack of motivation.

It is therefore crucial that organizations thoroughly analyze business processes before they implement ERP software. Analysis can identify opportunities for process modernization. It also enables an assessment of the alignment of current processes with those provided by the ERP system. Research indicates that risk of business process mismatch is decreased by:

- Linking current processes to the organization's strategy
- Analyzing the effectiveness of each process
- Understanding existing automated solutions

ERP implementation is considerably more difficult (and politically charged) in decentralized organizations, because they often have different processes, business rules, data semantics, authorization hierarchies, and decision centers. This may require migrating some business units before others, delaying implementation to work through the necessary changes for each unit, possibly reducing integration (e.g., linking via Master Data management) or customizing the system to meet specific needs.

.

Configuration

Configuring an ERP system is largely a matter of balancing the way the organization wants the system to work with the way it was designed to work. ERP systems typically include many settings that modify system operations. For example, an organization can select the type of inventory accounting FIFO or LIFO to use; whether to recognize revenue by geographical unit, product line, or distribution channel; and whether to pay for shipping costs on customer returns.

Two tier enterprise resource planning

Two-tier ERP software and hardware lets companies run the equivalent of two ERP systems at once: one at the corporate level and one at the division or subsidiary level. For example, a manufacturing company uses an ERP system to manage across the organization. This company uses independent global or regional distribution, production or sales centers, and service providers to support the main company‘s customers. Each independent center or subsidiary may have its own business models, workflows, and business processes.

Given the realities of globalization, enterprises continuously evaluate how to optimize their regional, divisional, and product or manufacturing strategies to support strategic goals and reduce time-to-market while increasing profitability and delivering value. With two-tier ERP, the regional distribution, production, or sales centers and service providers continue operating under their own business model separate from the main company, using their own ERP systems. Since these smaller companies' processes and workflows are not tied to main company's processes and workflows, they can respond to local business requirements in multiple locations.

Factors that affect enterprises' adoption of two-tier ERP systems include:

- Manufacturing globalization, the economics of sourcing in emerging economies
- Potential for quicker, less costly ERP implementations at subsidiaries, based on selecting software more suited to smaller companies
- Extra effort, (often involving the use of Enterprise application integration) is required where data must pass between two ERP systems Two-tier ERP strategies give enterprises agility in responding to market demands and in aligning IT systems at a corporate level while inevitably resulting in more systems as compared to one ERP system used throughout the organization.

Customization

ERP systems are theoretically based on industry best practices, and their makers intend that organizations deploy them as is. ERP vendors do offer customers configuration options that let organizations incorporate their own business rules, but often feature gaps remain even after configuration is complete.

These three options constitute varying degrees of system customization with the first being the most invasive and costly to maintain. Alternatively, there are non-technical options such as changing business practices or organizational policies to better match the delivered ERP feature set. Key differences between customization and configuration include:

- Customization is always optional, whereas the software must always be configured before use (e.g., setting up cost/profit center structures, organisational trees, purchase approval rules, etc.).
- The software is designed to handle various configurations, and behaves predictably in any allowed configuration.
- The effect of configuration changes on system behavior and performance is predictable and is the responsibility of the ERP vendor. The effect of customization is less predictable. It is the customer's responsibility, and increases testing activities.
- Configuration changes survive upgrades to new software versions. Some customizations (e.g., code that uses pre–defined "hooks" that are called before/after displaying data screens) survive upgrades, though they require retesting. Other customizations (e.g., those involving changes to fundamental data structures) are overwritten during upgrades and must be re implemented.

Customization advantages include that it:

- Improves user acceptance
- Offers the potential to obtain competitive advantage vis-à-vis companies using only standard features

Customization disadvantages include that it:

- Increases time and resources required to implement and maintain
- Inhibits seamless communication between suppliers and customers who use the same ERP system uncustomized
- Can create over reliance on customization, undermining the principles of ERP as a standardizing software platform

Extensions

ERP systems can be extended with third–party software. ERP vendors typically provide access to data and features through published interfaces. Extensions offer features such as:

- Archiving, reporting, and republishing
- Capturing transactional data, e.g., using scanners, tills or RFID
- Access to specialized data and capabilities, such as syndicated marketing data and associated trend analytics
- Advanced planning and scheduling (APS)

Data migration

Data migration is the process of moving, copying, and restructuring data from an existing system to the ERP system. Migration is critical to implementation success and requires significant planning. Unfortunately, since migration is one of the final activities before the production phase, it often receives insufficient attention.

The following steps can structure migration planning:

- Identify data to migrate
- Determine migration timing
- Generate data templates
- Freeze the toolset
- Decide on migration-related setups
- Define data archiving policies and procedure

Introduction

In any industry, some of the demands managers face is to be cost effective. In addition to that, they are also faced with challenges such as to analyze costs and profits on a product or consumer basis, to be flexible to face ever altering business requirements, and to be informed of management decision making processes and changes in ways of doing business.

However, some of the challenges holding managers back include the difficulty in attaining accurate information, lack of applications that mimic existing business practices and bad interfaces. When some challengers are holding a manager back, that is where Enterprise Resource Planning (ERP) comes into play.

Over the years business applications have evolved from Management Information Systems with no decision support to Corporate Information Systems, which offer some decision support to Enterprise Resource Planning. Enterprise Resource Planning is a software solution that tackles the needs of an organization, taking into account the process view to meet an organization's goals while incorporating all the functions of an organization.

Its purpose is to make easy the information flow between all business functions within the boundaries of the organization and manage the organization's connections with its outside stakeholders.

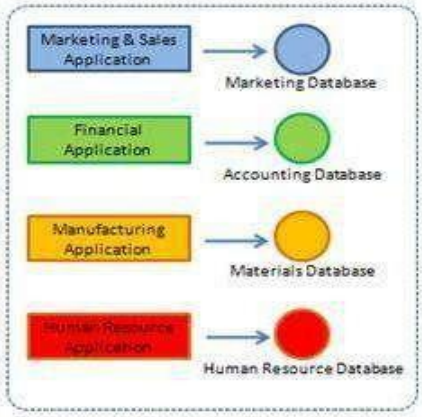
In a nutshell, the Enterprise Resource Planning software tries to integrate all the different departments and functions of an organization into a single computer system to serve the various needs of these departments.

However, if installed correctly this integrated approach can be very cost effective for an organization. With an integrated solution, different departments can easily share information and communicate with one another.

The following diagram illustrates the differences between non-integrated systems versus an integrated system for enterprise resource planning.

TheDrivingForcebehindERP

There are two main driving forces behind Enterprise Resource Planning for a business organization.



Non-integrated System

Integrated System

- In a business sense, Enterprise Resource Planning ensures customer satisfaction, as it leads to business development that is development of new areas, new products and new services.

Also, it allows businesses to face competition for implementing Enterprise Resource Planning, and it ensures efficient processes that push the company into top gear.

- In an IT sense: Most software‘s does not meet business needs wholly and the legacy systems today are hard to maintain. In addition, outdated hardware and software is hard to maintain.

Hence, for the above reasons, Enterprise Resource Planning is necessary for management in today's business world. ERP is single software, which tackles problems such as material shortages, customer service, finances management, quality issues and inventory problems. An ERP system can be the dashboard of the modern era managers.

e-Businesses

usiness (e-Business), derived from such terms as "e-mail" and "e-commerce," is the conduct of business electronically, typically over the Internet, not only buying and selling, but also servicing customers and collaborating with business partners.

The way in which you manage your business relationships has not changed, but the way they are referred to when using e-Business tools has. They are becoming more often known as:

- business to business (B2B)
- business to consumer (B2C) (also known as e-Commerce)
- government to consumer (G2C)
- Government to business (G2B).

Activities using e-Business tools

- trading of goods or services online, such as e-Procurement, primarily through websites
- electronic retailing (e-Tailing)
- use of the Internet, intranets or extranets to conduct research and manage business activities
- website marketing
- online communications, such as email
- Online training for staff (eLearning).

e-Business tools

- mobile phones
- personal digital assistants (PDAs)
- electronic data interchange
- file transfer
- facsimile
- Video conferencing, Internet, intranets and extranets.

e-Business in business

E-Business is more than having a website for your business. Using e-Business tools can make your administrative and operational activities more efficient through:

- accessing the Internet to source information about your industry, suppliers and products and for general research
- the use of electronic transaction, for example online banking, financial management, stock control and compliance reporting to regulatory bodies such as the Australian Taxation Office
- purchasing and selling without a web presence by using email or e-fax
- human resources management, through the development of an intranet for news, policies, staff movements and enabling staff to apply for leave and access their personnel information online
- customer relationship management, which integrates front and back office functions of an organisation through electronic capabilities
- Using appropriate project management software.

Advantages

The benefits of implementing e-Business tools is not so much in the use of technology, as in the streamlining of business processes and the ease in finding new markets. Some of the advantages include:

- quicker and easier communications
- strengthened marketing capabilities and reach
- increased hours of operation (a website provides 24 hour 7 day information to existing and potential customers)
- access to broader information through research
- reducing the cost of doing business by lowering transaction costs and increasing efficient methods for payment, such as using online banking and reducing stationery and postage costs
- The opportunities to adopt new business models and develop tailored customer support.

e-Governance

Several dimensions and factors influence the definition of e-governance or electronic governance. The word ―electronic‖ in the term e-governance implies technology driven governance. E-governance is the application of information and communication technology (ICT) for delivering government services, exchange of information communication transactions, integration of various stand-alone systems and services between government-to-customer (G2C), government-to-business (G2B), government-to-government (G2G) as well as back office processes and interactions within the entire government framework. Through e-governance, government services will be made available to citizens in a convenient, efficient and transparent manner. The three main target groups that can be distinguished in governance concepts are government, citizens and businesses/interest groups.

Generally four basic models are available – government-to-citizen (customer), government-to-employees, government-to-government and government-to-business.

Difference between E-Government and E-Governance

Both the terms are treated to be the same; however, there is some difference between the two. "E-government" is the use of the ICTs in public administration - combined with organizational change and new skills - to improve public services and democratic processes and to strengthen support to public. The problem in this definition to be congruence definition of e-governance is that there is no provision for governance of ICTs. As a matter of fact, the governance of ICTs requires most probably a substantial increase in regulation and policy-making capabilities, with all the expertise and opinion-shaping processes along the various social stakeholders of these concerns. So, the perspective of the e-governance is "the use of the technologies that both help governing and have to be governed". The Public-Private Partnership (PPP) based e-governance projects are hugely successful in India. United Telecoms Limited known as UTL is a major player in India on PPP based e-governance projects. Each project had mammoth state wide area networks in these states.

E-governance is the future; many countries are looking forward to for a corruption-free government. E-government is one-way communication protocol whereas e-governance is two-way communication protocol. The essence of e-governance is to reach the beneficiary and ensure that the services intended to reach the desired individual has been met with. There should be an auto-response to support the essence of e-governance, whereby the Government realizes the efficacy of its governance. E-governance is by the governed, for the governed and of the governed.

Establishing the identity of the end beneficiary is a challenge in all citizen-centric services. Statistical information published by governments and world bodies does not always reveal the facts. The best form of e-governance cuts down on unwanted interference of too many layers while delivering governmental services. It depends on good infrastructural setup with the support of local processes and parameters for governments to reach their citizens or end beneficiaries. Budget for planning, development and growth can be derived from well laid out e-governance systems

Government to customer

The goal of Government to Customer (G2C) e-Governance to be offer a variety of ICT services to citizens in an efficient and economical manner, and to strengthen the relationship between government and citizens using technology.

There are several methods of Government to Customer e-Governance. Two-way communication allows citizens to instant message directly with public administrators, and cast remote electronic votes (electronic voting) and instant opinion voting. Transactions such as payment of services, such as city utilities, can be completed online or over the phone. Mundane services such as name or address changes, applying for services or grants, or transferring existing services are more convenient.

G2C e-Governance is unbalanced across the globe as not everyone has Internet access and computing skills, but the United States, European Union, and Asia are ranked the top three in development.

The Federal Government of the United States has a broad framework of G2C technology to enhance citizen access to Government information and services. Benefits.Gov is an official US government website that informs citizens of benefits they are eligible for and provides information of how to apply assistance. US State Governments also engage in G2C interaction through the Department of Transportation, Department of Public Safety, United States Department of Health and Human Services, United States Department of Education, and others. As with e-Governance on the global level, G2C services vary from state to state. The Digital States Survey ranks states on social measures, digital democracy, e-commerce, taxation, and revenue. The 2012 report shows Michigan and Utah in the lead and Florida and Idaho with the lowest scores. Municipal governments in the United States also use Government to Customer technology to complete transactions and inform the public. Much like states, cities are awarded for innovative technology. Government Technology's "Best of the Web 2012" named Louisville, KY, Arvada, CO, Raleigh, NC, Riverside, CA, and Austin, TX the top five G2C city portals.

European countries were ranked second among all geographic regions. The Single Point of Access for Citizens of Europe supports travel within Europe and e-Europe is a 1999 initiative supporting online government. Main focuses are to provide public information, allow customers to have access to basic public services, simplify online procedures, and promote electronic signatures.

Asia is ranked third in comparison, and there are diverse G2C programs between countries. Singapore‘s e-Citizen Portal is an organized single access point to government information and services. South Korea‘s Home Tax Service (HTS) provides citizens with 24/7 online service such as tax declaration. Taiwan has top ranking G2C technology including an online motor vehicle services system, which provides 21 applications and payment services to citizens.

G2C Concerns

A full switch to Government to Customer e-Governance will cost a large amount of money in development and implementation. In addition, Government agencies do not always engage citizens in the development of their e-Gov services or accept feedback. Customers identified the following barriers to Government to Customer e-Governance: not everyone has Internet access, especially in rural or low income areas; G2C technology can be problematic for citizens who lack computing skills. some G2C sites have technology requirements (such as browser requirements and plug-ins) that won‘t allow access to certain services, language barriers, the necessity for an e-mail address to access certain services, and a lack of privacy.

E-governance makes it possible for employees to become paperless and makes it easy for employees to send important documents back and forth to colleagues all over the world instead of having to print out these records or fax G2E services also include software for maintaining personal information

and records of employees. Some of the benefits of G2E expansion includes,

E-Payroll- maintaining the online sources to view paychecks, pay stubs, pay bills, and keep records for tax information-benefits- be able to look up what benefits an employee is receiving and what benefits they have a right to.

E-training- allows for new and current employees to regularly maintain the training they have through the development of new technology and to allow new employees to train and learn over new materials in one convenient location. E-learning is another way to keep employees informed on the important materials they need to know through the use of visuals, animation, videos, etc. It is usually a computer based learning tool, although not always. It is also a way for employees to learn at their own pace (distance learning).

Maintaining records of personal information- Allows the system to keep all records in one easy location to update with every single bit of information that is relevant to a personal file. Examples being social security numbers, tax information, current address, and other information.

e-governance in India

E-governance is a wonderful tool to bring transparency, accountability and whistle blowing in India. However, it has its own share of challenges that include administrative, legal and technological challenges. There may be instances where e-governance can itself be a source of corruption. Use of e-governance in India would also require an efficient mechanism to deal with e-waste. Absence of privacy and data protection laws has also created many hurdles before successful implementation of e-governance in India. However, the biggest hurdle before Indian e-governance initiatives comes from poor cyber security in India. According to techno legal experts, e-governance without cyber security is useless in India. In fact, that makes the critical infrastructure of India vulnerable to sophisticated cyber attacks. Absence of mandatory e-governance services in India is the main reason for apathy towards this crucial field.

Data Mining

Data mining, the extraction of hidden predictive information from large databases, is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analyses offered by data mining move beyond the analyses of past events provided by retrospective tools typical of decision support systems. Data mining tools can answer business questions that traditionally were too time consuming

to resolve. They scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations.

Most companies already collect and refine massive quantities of data. Data mining techniques can be implemented rapidly on existing software and hardware platforms to enhance the value of existing information resources, and can be integrated with new products and systems as they are brought on-line. This white paper provides an introduction to the basic technologies of data mining. Examples of profitable applications illustrate its relevance to today‘s business environment as well as a basic description of how data warehouse architectures can evolve to deliver the value of data mining to end users.

The Foundations of Data Mining

Data mining techniques are the result of a long process of research and product development. This evolution began when business data was first stored on computers, continued with improvements in data access, and more recently, generated technologies that allow users to navigate through their data in real time.

Data mining takes this evolutionary process beyond retrospective data access and navigation to prospective and proactive information delivery. Data mining is ready for application in the business community because it is supported by three technologies that are now sufficiently mature:

- Massive data collection
- Powerful multiprocessor computers
- Data mining algorithms

Commercial databases are growing at unprecedented rates. A recent META Group survey of data warehouse projects found that 19% of respondents are beyond the 50 gigabyte level, while 59% expect to be there by second quarter of 1996.In some industries, such as retail, these numbers can be much larger.

The accompanying need for improved computational engines can now be met in a cost-effective manner with parallel multiprocessor computer technology. Data mining algorithms embody techniques that have existed for at least 10 years, but have only recently been implemented as mature, reliable, understandable tools that consistently outperform older statistical methods.

In the evolution from business data to business information, each new step has built upon the previous one. For example, dynamic data access is critical for drill-through in data navigation applications, and the ability to store large databases is critical to data mining. From the user‘s point of view, the four steps listed in Table 1 were revolutionary because they allowed new business questions to be answered accurately and quickly.

Evolutionary Step	Business Question	Enabling Technologies	Product Providers	Characteristics
Data Collection (1960s)	"What was my total revenue in the last five years?"	Computers, tapes, disks	IBM, CDC	Retrospective, static data delivery
Data Access (1980s)	"What were unit sales in New England last March?"	Relational databases (RDBMS), Structured Query Language (SQL), ODBC	Oracle, Sybase, Informix, IBM, Microsoft	Retrospective, dynamic data delivery at record level
Data Warehousing & Decision Support (1990s)	"What were unit sales in New England last March? Drill down to Boston."	On-line analytic processing (OLAP), multidimensional databases, data warehouses	Pilot, Comshare, Arbor, Cognos, Microstrategy	Retrospective, dynamic data delivery at multiple levels
Data Mining (Emerging Today)	"What's likely to happen to Boston unit sales next month? Why?"	Advanced algorithms, multiprocessor computers, massive databases	Pilot, Lockheed, IBM, SGI, numerous startups (nascent industry)	Prospective, proactive information delivery

Table 1. Steps in the Evolution of Data Mining.

The core components of data mining technology have been under development for decades, in research areas such as statistics, artificial intelligence, and machine learning. Today, the maturity of these techniques, coupled with high-performance relational database engines and broad data integration efforts, make these technologies practical for current data warehouse environments.

Given databases of sufficient size and quality, data mining technology can generate new business opportunities by providing these capabilities:

- Automated prediction of trends and behaviors. Data mining automates the process of finding predictive information in large databases. Questions that traditionally required extensive hands-on analysis can now be answered directly from the data quickly. A typical example of a predictive problem is targeted marketing. Data mining uses data on past promotional mailings to identify the targets most likely to maximize return on investment in future mailings. Other predictive problems include forecasting bankruptcy and other forms of default, and identifying segments of a population likely to respond similarly to given events.

- Automated discovery of previously unknown patterns. Data mining tools sweep through databases and identify previously hidden patterns in one step. An example of pattern discovery is the analysis of retail sales data to identify seemingly unrelated products that are often purchased together. Other pattern discovery problems include detecting fraudulent credit card transactions and identifying anomalous data that could represent data entry keying errors.

Data mining techniques can yield the benefits of automation on existing software and hardware platforms, and can be implemented on new systems as existing platforms are upgraded and new products developed. When data mining tools are implemented on high performance parallel processing systems, they can analyze massive databases in minutes. Faster processing means that users can automatically experiment with more models to understand complex data. High speed makes it practical for users to analyze huge quantities of data. Larger databases, in turn, yield improved predictions.

Databases can be larger in both depth and breadth

- More columns. Analysts must often limit the number of variables they examine when doing hands-on analysis due to time constraints. Yet variables that are discarded because they seem unimportant may carry information about unknown patterns. High performance data mining allows users to explore the full depth of a database, without preselecting a subset of variables.

- More rows. Larger samples yield lower estimation errors and variance, and allow users to make inferences about small but important segments of a population.

A recent Gartner Group Advanced Technology Research Note listed data mining and artificial intelligence at the top of the five key technology areas that "will clearly have a major impact across a wide range of industries within the next 3 to 5 years."Gartner also listed parallel architectures and data mining as two of the top 10 new technologies in which companies will invest during the next 5 years. According to a recent Gartner HPC Research Note, "With the rapid advance in data capture, transmission and storage, large-systems users will increasingly need to implement new and innovative ways to mine the after-market value of their vast stores of detail data, employing MPP [massively parallel processing] systems to create new sources of business advantage (0.9 probability)."

Techniques in data mining

- Artificial neural networks: Non-linear predictive models that learn through training and resemble biological neural networks in structure.

- Decision trees: Tree-shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset. Specific decision tree methods include Classification and Regression Trees (CART) and Chi Square Automatic Interaction Detection (CHAID).

- Genetic algorithms: Optimization techniques that use process such as genetic combination, mutation, and natural selection in a design based on the concepts of evolution.

- Nearest neighbor method: A technique that classifies each record in a dataset based on a combination of the classes of the k record(s) most similar to it in a historical dataset. Sometimes called the k-nearest neighbor technique.

- Rule induction: The extraction of useful if-then rules from data based on statistical significance.

Many of these technologies have been in use for more than a decade in specialized analysis tools that work with relatively small volumes of data. These capabilities are now evolving to integrate directly with industry-standard data warehouse and OLAP platforms. The appendix to this white paper provides a glossary of data mining terms.

How Data Mining Works

How exactly is data mining able to tell you important things that you didn't know or what is going to happen next? The technique that is used to perform these feats in data mining is called modeling. Modeling is simply the act of building a model in one situation where you know the answer and then applying it to another situation that you don't. For instance, if you were looking for a sunken Spanish galleon on the high seas the first thing you might do is to research the times when Spanish treasure had been found by others in the past. You might note that these ships often tend to be found off the coast of Bermuda and that there are certain characteristics to the ocean currents, and certain routes that have likely been taken by the ship‘s captains in that era. You note these similarities and build a model that includes the characteristics that are common to the locations of these sunken treasures. With these models in hand you sail off looking for treasure where your model indicates it most likely might be given a similar situation in the past. Hopefully, if you've got a good model, you find your treasure.

Once the model is built it can then be used in similar situations where you don't know the answer. For example, say that you are the director of marketing for a telecommunications company and you'd like to acquire some new long distance phone customers. You could just randomly go out and mail coupons to the general population - just as you could randomly sail the seas looking for sunken treasure. In neither case would you achieve the results you desired and of course you have the opportunity to do much better than random - you could use your business experience stored in your database to build a model.

As the marketing director you have access to a lot of information about all of your customers: their age, sex, credit history and long distance calling usage. The good news is that you also have a lot of information about your prospective customers: their age, sex, credit history etc. Your problem is that you don't know the long distance calling usage of these prospects (since they are most likely now customers of your competition). You'd like to concentrate on those prospects that have large amounts of long distance usage. You can accomplish this by building a model. Table 2 illustrates the data used for building a model for new customer prospecting in a data warehouse.

	Customers	Prospects
General information (e.g. demographic data)	Known	Known
Proprietary information (e.g. customer transactions)	Known	Target

Table 2 - Data Mining for Prospecting

The goal in prospecting is to make some calculated guesses about the information in the lower right hand quadrant based on the model that we build going from Customer General Information to Customer Proprietary Information. For instance, a simple model for a telecommunications company might be:

98% of my customers who make more than $60,000/year spend more than $80/month on long distance. This model could then be applied to the prospect data to try to tell something about the proprietary information that this telecommunications company does not currently have access to. With this model in hand new customers can be selectively targeted.

Test marketing is an excellent source of data for this kind of modeling. Mining the results of a test market representing a broad but relatively small sample of prospects can provide a foundation for identifying good prospects in the overall market. Table 3 shows another common scenario for building models: predict what is going to happen in the future.

	Yesterday	Today	Tomorrow
Static information and current plans (e.g. demographic data, marketing plans)	Known	Known	Known
Dynamic information (e.g. customer transactions)	Known	Known	Target

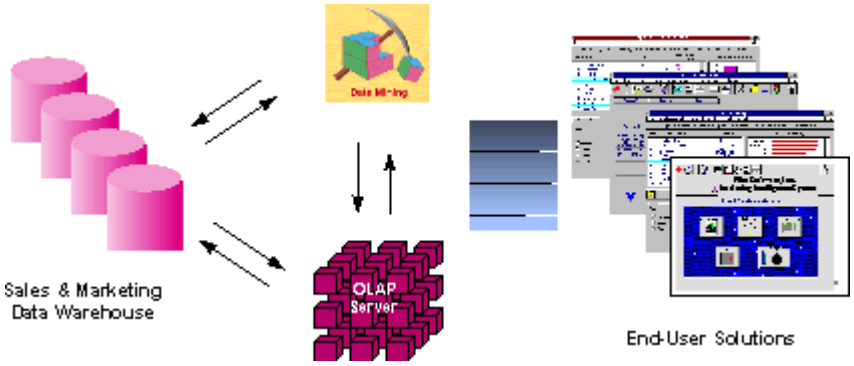
Table 3 - Data Mining for Predictions

If someone told you that he had a model that could predict customer usage how would you know if he really had a good model? The first thing you might try would be to ask him to apply his model to your customer base - where you already knew the answer. With data mining, the best way to accomplish this is by setting aside some of your data in a vault to isolate it from the mining process. Once the mining is complete, the results can be tested against the data held in the vault to confirm the model‘s validity. If the model works, its observations should hold for the vaulted data.

Architecture for Data Mining

To best apply these advanced techniques, they must be fully integrated with a data warehouse as well as flexible interactive business analysis tools. Many data mining tools currently operate outside of the warehouse, requiring extra steps for extracting, importing, and analyzing the data. Furthermore, when new insights require operational implementation, integration with the warehouse simplifies the application of results from data mining. The resulting analytic data warehouse can be applied to improve business processes throughout the organization, in areas such as promotional campaign.

Integrated Data Mining Architecture



The ideal starting point is a data warehouse containing a combination of internal data tracking all customer contact coupled with external market data about competitor activity. Background information on potential customers also provides an excellent basis for prospecting. This warehouse can be implemented in a variety of relational database systems: Sybase, Oracle, Redbrick, and so on, and should be optimized for flexible and fast data access.

An OLAP (On-Line Analytical Processing) server enables a more sophisticated end-user business model to be applied when navigating the data warehouse. The multidimensional structures allow the user to analyze the data as they want to view their business – summarizing by product line, region, and other key perspectives of their business. The Data Mining Server must be integrated with the data warehouse and the OLAP server to embed ROI-focused business analysis directly into this infrastructure. An advanced, process-centric metadata template defines the data mining objectives for specific business issues like campaign management, prospecting, and promotion optimization. Integration with the data warehouse enables operational decisions to be directly implemented and tracked. As the warehouse grows with new decisions and results, the organization can continually mine the best practices and apply them to future decisions.

This design represents a fundamental shift from conventional decision support systems. Rather than simply delivering data to the end user through query and reporting software, the Advanced Analysis Server applies users‘ business models directly to the warehouse and returns a proactive analysis of the most relevant information. These results enhance the metadata in the OLAP Server by providing a dynamic metadata layer that represents a distilled view of the data. Reporting, visualization, and other analysis tools can then be applied to plan future actions and confirm the impact of those plans.

Application

- A pharmaceutical company can analyze its recent sales force activity and their results to improve targeting of high-value physicians and determine which marketing activities will have the greatest impact in the next few months. The data needs to include competitor market activity as well as information about the local health care systems. The results can be distributed to the sales force via a wide-area network that enables the representatives to review the recommendations from the perspective of the key attributes in the decision process. The ongoing, dynamic analysis of the data warehouse allows best practices from throughout the organization to be applied in specific sales situations.
- A credit card company can leverage its vast warehouse of customer transaction data to identify customers most likely to be interested in a new credit product. Using a small test mailing, the attributes of customers with an affinity for the product can be identified. Recent projects have indicated more than a 20-fold decrease in costs for targeted mailing campaigns over conventional approaches.
- A diversified transportation company with a large direct sales force can apply data mining to identify the best prospects for its services. Using data mining to analyze its own customer experience, this company can build a unique segmentation identifying the attributes of high-value prospects.

- A large consumer package goods company can apply data mining to improve its sales process to retailers. Data from consumer panels, shipments, and competitor activity can be applied to understand the reasons for brand and store switching. Through this analysis, the manufacturer can select promotional strategies that best reach their target customer segments.

Comprehensive data warehouses that integrate operational data with customer, supplier, and market information have resulted in an explosion of information. Competition requires timely and sophisticated analysis on an integrated view of the data. However, there is a growing gap between more powerful storage and retrieval systems and the users‘ ability to effectively analyze and act on the information they contain. Both relational and OLAP technologies have tremendous capabilities for navigating massive data warehouses, but brute force navigation of data is not enough. A new technological leap is needed to structure and prioritize information for specific end-user problems. The data mining tools can make this leap. Quantifiable business benefits have been proven through the integration of data mining with current information systems, and new products are on the horizon that will bring this integration to an even wider audience of users.

Business intelligence

Business intelligence (BI) is the set of techniques and tools for the transformation of raw data into meaningful and useful information for business analysis purposes. BI technologies are capable of handling large amounts of unstructured data to help identify, develop and otherwise create new strategic business opportunities. The goal of BI is to allow for the easy interpretation of these large volumes of data. Identifying new opportunities and implementing an effective strategy based on insights can provide businesses with a competitive market advantage and long-term stability.

BI technologies provide historical, current and predictive views of business operations. Common functions of business intelligence technologies are reporting, online analytical processing, analytics, data mining, process mining, complex event processing, business performance management, benchmarking, text mining, predictive analytics and prescriptive analytics.

Components

Business intelligence is made up of an increasing number of components including:

- Multidimensional aggregation and allocation
- De normalization, tagging and standardization
- Real time reporting with analytical alert
- A method of interfacing with unstructured data sources
- Group consolidation, budgeting and rolling forecasts
- Statistical inference and probabilistic simulation
- Key performance indicators optimization
- Version control and process management.

Applications in an enterprise

Business intelligence can be applied to the following business purposes, in order to drive business value.

1. Measurement – program that creates a hierarchy of performance metrics (see also Metrics Reference Model) and benchmarking that informs business leaders about progress towards business goals (business process management).
2. Analytics – program that builds quantitative processes for a business to arrive at optimal decisions and to perform business knowledge discovery. Frequently involves: data mining, process mining, statistical analysis, predictive analytics, predictive modeling, business process modeling, data lineage, complex event processing and prescriptive analytics.
3. Reporting/enterprise reporting – program that builds infrastructure for strategic reporting to serve the strategic management of a business, not operational reporting. Frequently involves data visualization, executive information system and OLAP.
4. Collaboration/collaboration platform – program that gets different areas (both inside and outside the business) to work together through data sharing and electronic data interchange.
5. Knowledge management – program to make the company data driven through strategies and practices to identify, create, represent, distribute, and enable adoption of insights and experiences that are true business knowledge. Knowledge management leads to learning management and regulatory compliance.

In addition to the above, business intelligence can provide a pro-active approach, such as alert functionality that immediately notifies the end-user if certain conditions are met. For example, if some business metric exceeds a pre-defined threshold, the metric will be highlighted in standard reports, and the business analyst may be alerted via email or another monitoring service. This end-to-end process requires data governance, which should be handled by the expert.

Prioritization of projects

It can be difficult to provide a positive business case for business intelligence initiatives, and often the projects must be prioritized through strategic initiatives. BI projects can attain higher prioritization within the organization if managers consider the following:

- As described by Kimball the BI manager must determine the tangible benefits such as eliminated cost of producing legacy reports.
- Data access for the entire organization must be enforced. In this way even a small benefit, such as a few minutes saved, makes a difference when multiplied by the number of employees in the entire organization.
- As described by Ross, Weil & Roberson for Enterprise Architecture, managers should also consider letting the BI project be driven by other business initiatives with excellent business cases.

- Using a structured and quantitative methodology to create defensible prioritization in line with the actual needs of the organization, such as a weighted decision matrix.

Success factors of implementation

According to Kimball et al., there are three critical areas that organizations should assess before getting ready to do a BI project:

1. The level of commitment and sponsorship of the project from senior management
2. The level of business need for creating a BI implementation
3. The amount and quality of business data available.

Business sponsorship

The commitment and sponsorship of senior management is according to Kimball et al., the most important criteria for assessment. This is because having strong management backing helps overcome shortcomings elsewhere in the project. However, as Kimball et al. state: ―even the most elegantly designed DW/BI system cannot overcome a lack of business [management] sponsorship‖.

It is important that personnel who participate in the project have a vision and an idea of the benefits and drawbacks of implementing a BI system. The best business sponsor should have organizational clout and should be well connected within the organization. It is ideal that the business sponsor is demanding but also able to be realistic and supportive if the implementation runs into delays or drawbacks. The management sponsor also needs to be able to assume accountability and to take responsibility for failures and setbacks on the project.

Support from multiple members of the management ensures the project does not fail if one person leaves the steering group. However, having many managers work together on the project can also mean that there are several different interests that attempt to pull the project in different directions, such as if different departments want to put more emphasis on their usage. This issue can be countered by an early and specific analysis of the business areas that benefit the most from the implementation. All stakeholders in project should participate in this analysis in order for them to feel ownership of the project and to find common ground.

Another management problem that should be encountered before start of implementation is if the business sponsor is overly aggressive. If the management individual gets carried away by the possibilities of using BI and starts wanting the DW or BI implementation to include several different sets of data that were not included in the original planning phase. However, since extra implementations of extra data may add many months to the original plan, it's wise to make sure the person from management is aware of their actions.

Business needs

Because of the close relationship with senior management, another critical thing that must be assessed before the project begins is whether or not there is a business need and whether there is a clear

business benefit by doing the implementation. The needs and benefits of the implementation are sometimes driven by competition and the need to gain an advantage in the market. Another reason for a business-driven approach to implementation of BI is the acquisition of other organizations that enlarge the original organization it can sometimes be beneficial to implement DW or BI in order to create more oversight.

Companies that implement BI are often large, multinational organizations with diverse subsidiaries. A well-designed BI solution provides a consolidated view of key business data not available anywhere else in the organization, giving management visibility and control over measures that otherwise would not exist.

Amount and quality of available data

Without proper data, or with too little quality data, any BI implementation fails; it does not matter how good the management sponsorship or business-driven motivation is. Before implementation it is a good idea to do data profiling. This analysis identifies the ―content, consistency and structure ―of the data. This should be done as early as possible in the process and if the analysis shows that data is lacking, put the project on hold temporarily while the IT department figures out how to properly collect data.

When planning for business data and business intelligence requirements, it is always advisable to consider specific scenarios that apply to a particular organization, and then select the business intelligence features best suited for the scenario.

Often, scenarios revolve around distinct business processes, each built on one or more data sources. These sources are used by features that present that data as information to knowledge workers, who subsequently act on that information. The business needs of the organization for each business process adopted correspond to the essential steps of business intelligence. These essential steps of business intelligence include but are not limited to:

1. Go through business data sources in order to collect needed data
2. Convert business data to information and present appropriately
3. Query and analyze data
4. Act on the collected data

The quality aspect in business intelligence should cover all the process from the source data to the final reporting. At each step, the quality gates are different:

1. Source Data:
    - Data Standardization: make data comparable
    - Master Data Management: unique referential

2. Operational Data Store (ODS):
    - Data Cleansing: detect & correct inaccurate data
    - Data Profiling: check inappropriate value, null/empty
3. Data warehouse:
    - Completeness: check that all expected data are loaded
    - Referential integrity: unique and existing referential over all sources
    - Consistency between sources: check consolidated data vs. sources
4. Reporting:
    - Uniqueness of indicators: only one share dictionary of indicators
    - Formula accuracy: local reporting formula should be avoided or checked

BI Portals

A Business Intelligence portal (BI portal) is the primary access interface for Data Warehouse (DW) and Business Intelligence (BI) applications. The BI portal is the user's first impression of the DW/BI system. It is typically a browser application, from which the user has access to all the individual services of the DW/BI system, reports and other analytical functionality. The BI portal must be implemented in such a way that it is easy for the users of the DW/BI application to call on the functionality of the application.

The BI portal's main functionality is to provide a navigation system of the DW/BI application. This means that the portal has to be implemented in a way that the user has access to all the functions of the DW/BI application.

The most common way to design the portal is to custom fit it to the business processes of the organization for which the DW/BI application is designed, in that way the portal can best fit the needs and requirements of its users.

The BI portal needs to be easy to use and understand, and if possible have a look and feel similar to other applications or web content of the organization the DW/BI application is designed for (consistency).

The following is a list of desirable features for web portals in general and BI portals in particular:

- Usable
  User should easily find what they need in the BI tool.
- Content Rich
  The portal is not just a report printing tool; it should contain more functionality such as advice, help, support information and documentation.

- Clean
  The portal should be designed so it is easily understandable and not over complex as to confuse the users

- Interactive
  The portal should be implemented in a way that makes it easy for the user to use its functionality and encourage them to use the portal. Scalability and customization give the user the means to fit the portal to each user.

- Value Oriented
  It is important that the user has the feeling that the DW/BI application is a valuable resource that is worth working on.

Pervasive computing

Ubiquitous computing (pervasive) is a concept in software engineering and computer science where computing is made to appear everywhere and anywhere. In contrast to desktop computing, ubiquitous computing can occur using any device, in any location, and in any format. A user interacts with the computer, which can exist in many different forms, including laptop computers, tablets and terminals in everyday objects such as a fridge or a pair of glasses. The underlying technologies to support ubiquitous computing include Internet, advanced middleware, operating system, mobile code, sensors, microprocessors, new I/O and user interfaces, networks, mobile protocols, location and positioning and new materials.

This new paradigm is also described as pervasive computing, ambient intelligence, or 'every ware'. Each term emphasizes slightly different aspects. When primarily concerning the objects involved, it is also known as physical computing, the Internet of Things, hap tic computing, and 'things that think'. Rather than propose a single definition for ubiquitous computing and for these related terms, taxonomy of properties for ubiquitous computing has been proposed, from which different kinds or flavors of ubiquitous systems and applications can be described.

Ubiquitous computing touches on a wide range of research topics, including distributed computing, mobile computing, location computing, mobile networking, context-aware computing, sensor networks, human-computer interaction, and artificial intelligence.

Core concepts

At their core, all models of ubiquitous computing share a vision of small, inexpensive, robust networked processing devices, distributed at all scales throughout everyday life and generally turned to distinctly common-place ends. For example, a domestic ubiquitous computing environment might interconnect lighting and environmental controls with personal biometric monitors woven into clothing so that illumination and heating conditions in a room might be modulated, continuously and imperceptibly. Another common scenario posits refrigerators "aware" of their suitably tagged contents, able to both plan a variety of menus from the food actually on hand, and warn users of stale or spoiled food.

Ubiquitous computing presents challenges across computer science: in systems design and engineering, in systems modeling, and in user interface design. Contemporary human-computer interaction models, whether command-line, menu-driven, or GUI-based, are inappropriate and inadequate to the ubiquitous case. This suggests that the "natural" interaction paradigm appropriate to a fully robust ubiquitous computing has yet to emerge - although there is also recognition in the field that in many ways we are already living in an ubicomp world (see also the main article on Natural User Interface). Contemporary devices that lend some support to this latter idea include mobile phones, digital audio players, radio-frequency identification tags, GPS, and interactive whiteboards.

Mark Weiser proposed three basic forms for ubiquitous system devices: tabs, pads and boards.

- Tabs: wearable centimetre sized devices
- Pads: hand-held decimetre-sized devices
- Boards: metre sized interactive display devices.

These three forms proposed by Weiser are characterized by being macro-sized, having a planar form and on incorporating visual output displays. If we relax each of these three characteristics we can expand this range into a much more diverse and potentially more useful range of Ubiquitous Computing devices. Hence, three additional forms for ubiquitous systems have been proposed:

- Dust: miniaturized devices can be without visual output displays, e.g. Micro Electro-Mechanical Systems (MEMS), ranging from nanometres through micrometers to millimetres. See also Smart dust.
- Skin: fabrics based upon light emitting and conductive polymers, organic computer devices, can be formed into more flexible non-planar display surfaces and products such as clothes and curtains, see OLED display. MEMS device can also be painted onto various surfaces so that a variety of physical world structures can act as networked surfaces of MEMS.
- Clay: ensembles of MEMS can be formed into arbitrary three dimensional shapes as artifacts resembling many different kinds of physical object (see also Tangible interface).

In his book The Rise of the Network Society, Manuel Castells suggests that there is an ongoing shift from already-decentralized, stand-alone microcomputers and mainframes towards entirely pervasive computing. In his model of a pervasive computing system, Castells uses the example of the Internet as the start of a pervasive computing system. The logical progression from that paradigm is a system where that networking logic becomes applicable in every realm of daily activity, in every location and every context. Castells envisages a system where billions of miniature, ubiquitous inter-communication devices will be spread worldwide, "like pigment in the wall paint".
Ubiquitous computing may be seen to consist of many layers, each with their own roles, which together form a single system:
Layer 1: task management layer

- Monitors user task, context and index
- Map user's task to need for the services in the environment.

Layer 2: environment management layer

- To monitor a resource and its capabilities
- To map service need, user level states of specific capabilities

Layer 3: environment layer

- To monitor a relevant resource
- To manage reliability of the resources

Cloud computing

Cloud computing is computing in which large groups of remote servers are networked to allow the centralized data storage, and online access to computer services or resources. Clouds can be classified as public, private or hybrid.

Cloud computing is the result of evolution and adoption of existing technologies and paradigms. The goal of cloud computing is to allow users to take benefit from all of these technologies, without the need for deep knowledge about or expertise with each one of them. The cloud aims to cut costs, and help the users focus on their core business instead of being impeded by IT obstacles.

The main enabling technology for cloud computing is virtualization. Virtualization software separates a physical computing device into one or more "virtual" devices, each of which can be easily used and managed to perform computing tasks. With operating system–level virtualization essentially creating a scalable system of multiple independent computing devices, idle computing resources can be allocated and used more efficiently. Virtualization provides the agility required to speed up IT operations, and reduces cost by increasing infrastructure utilization. Autonomic computing automates the process through which the user can provision resources on-demand. By minimizing user involvement, automation speeds up the process, reduces labor costs and reduces the possibility of human errors.

Users routinely face difficult business problems. Cloud computing adopts concepts from Service-oriented Architecture (SOA) that can help the user break these problems into services that can be integrated to provide a solution. Cloud computing provides all of its resources as services, and makes use of the well-established standards and best practices gained in the domain of SOA to allow global and easy access to cloud services in a standardized way.

Cloud computing also leverages concepts from utility computing to provide metrics for the services used. Such metrics are at the core of the public cloud pay-per-use models. In addition, measured services are an essential part of the feedback loops in autonomic computing, allowing services to scale on-demand and to perform automatic failure recovery.

data/compute intensive parallel applications with much more affordable prices compared to traditional parallel computing techniques.

- Grid computing -"A form of distributed and parallel computing, whereby a 'super and virtual computer' is composed of a cluster of networked, loosely coupled computers acting in concert to perform very large tasks."
- Mainframe computer - Powerful computers used mainly by large organizations for critical applications, typically bulk data processing such as: census; industry and consumer statistics; police and secret intelligence services; enterprise resource planning; and financial transaction processing.
- Utility computing - The "packaging of computing resources, such as computation and storage, as a metered service similar to a traditional public utility, such as electricity."
- Peer-to-peer - A distributed architecture without the need for central coordination. Participants are both suppliers and consumers of resources (in contrast to the traditional client–server model).

Characteristics

Cloud computing exhibits the following key characteristics:

- Agility improves with users' ability to re-provision technological infrastructure resources.
- Application programming interface (API) accessibility to software that enables machines to interact with cloud software in the same way that a traditional user interface (e.g., a computer desktop) facilitates interaction between humans and computers. Cloud computing systems typically use Representational State Transfer (REST)-based APIs.
- Cost reductions claimed by cloud providers. A public-cloud delivery model converts capital expenditure to operational expenditure. This purportedly lowers barriers to entry, as infrastructure is typically provided by a third party and does not need to be purchased for one-time or infrequent intensive computing tasks. Pricing on a utility computing basis is fine-grained, with usage-based options and fewer IT skills are required for implementation (in-house).The e-FISCAL project's state-of-the-art repository contains several articles looking into cost aspects in more detail, most of them concluding that costs savings depend on the type of activities supported and the type of infrastructure available in-house.
- Device and location independence enable users to access systems using a web browser regardless of their location or what device they use (e.g., PC, mobile phone). As infrastructure is off-site (typically provided by a third-party) and accessed via the Internet, users can connect from anywhere.
- Maintenance of cloud computing applications is easier, because they do not need to be installed on each user's computer and can be accessed from different places.

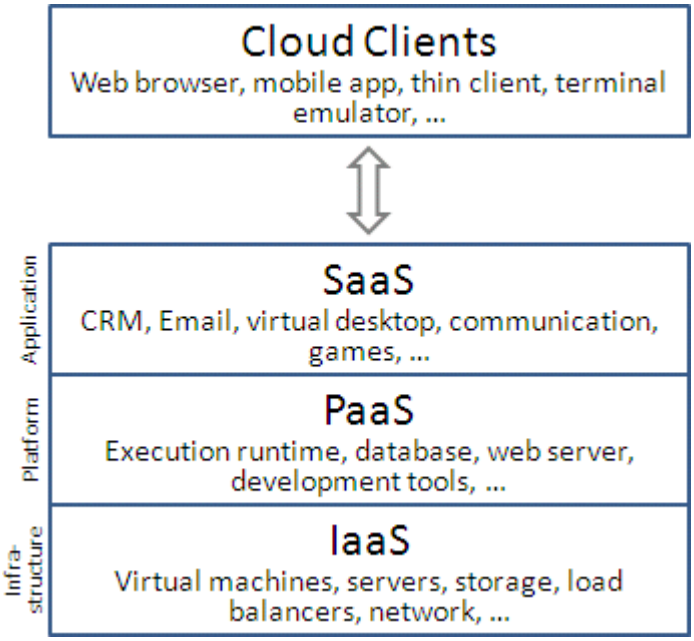
- Performance is monitored and consistent and loosely coupled architectures are constructed using web services as the system interface.

- Productivity may be increased when multiple users can work on the same data simultaneously, rather than waiting for it to be saved and emailed. Time may be saved as information does not need to be re-entered when fields are matched, nor do users need to install application software upgrades to their computer.
- Reliability improves with the use of multiple redundant sites, which makes well-designed cloud computing suitable for business continuity and disaster recovery.
- Scalability and elasticity via dynamic ("on-demand") provisioning of resources on a fine-grained, self-service basis in near real-time (Note, the VM startup time varies by VM type, location, OS and cloud providers), without users having to engineer for peak loads.
- Security can improve due to centralization of data, increased security-focused resources, etc., but concerns can persist about loss of control over certain sensitive data, and the lack of security for stored kernels. Security is often as good as or better than other traditional systems, in part because providers are able to devote resources to solving security issues that many customers cannot afford to tackle. However, the complexity of security is greatly increased when data is distributed over a wider area or over a greater number of devices, as well as in multi-tenant systems shared by unrelated users. In addition, user access to security audit logs may be difficult or impossible. Private cloud installations are in part motivated by users' desire to retain control over the infrastructure and avoid losing control of information security.

The National Institute of Standards and Technology's definition of cloud computing identifies "five essential characteristics":

- On-demand self-service. A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.
- Broad network access. Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).
- Resource pooling. The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.
- Rapid elasticity. Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear unlimited and can be appropriated in any quantity at any time.
- Measured service. Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts).

and consumer of the utilized service



Cloud computing providers offer their services according to several fundamental models:

Infrastructure as a service (IaaS)

In the most basic cloud-service model & according to the IETF (Internet Engineering Task Force), providers of IaaS offer computers – physical or (more often) virtual machines – and other resources. (A hypervisor, such as Xen, Oracle VirtualBox, KVM, VMware ESX/ESXi, or Hyper-V runs the virtual machines as guests. Pools of hypervisors within the cloud operational support-system can support large numbers of virtual machines and the ability to scale services up and down according to customers' varying requirements.)

IaaS clouds often offer additional resources such as a virtual-machine disk image library, raw block storage, and file or object storage, firewalls, load balancers, IP addresses, virtual local area networks (VLANs), and software bundles. IaaS-cloud providers supply these resources on-demand from their large pools installed in data centers. For wide-area connectivity, customers can use either the Internet or carrier clouds (dedicated virtual private networks).

To deploy their applications, cloud users install operating-system images and their application software on the cloud infrastructure. In this model, the cloud user patches and maintains the operating systems and the application software. Cloud providers typically bill IaaS services on a utility computing basis: cost reflects the amount of resources allocated and consumed.

Platform as a service (PaaS)

In the PaaS models, cloud providers deliver a computing platform, typically including operating system, programming language execution environment, database, and web server. Application developers can develop and run their software solutions on a cloud platform without the cost and complexity of buying and managing the underlying hardware and software layers. With some PaaS offers like Microsoft Azure and Google App Engine, the underlying computer and storage resources scale automatically to match application demand so that the cloud user does not have to allocate resources manually. The latter has also been proposed by an architecture aiming to facilitate real-time in cloud environments.

Platform as a service (PaaS) provides a computing platform and a key chimney. It joins with software as a service (SaaS) and infrastructure as a service (IaaS), model of cloud computing.

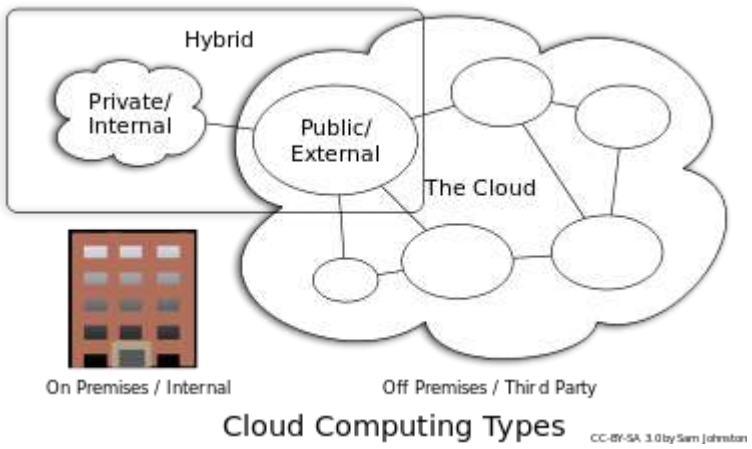
Software as a service (SaaS)

In the business model using software as a service (SaaS), users are provided access to application software and databases. Cloud providers manage the infrastructure and platforms that run the applications. SaaS is sometimes referred to as "on-demand software" and is usually priced on a pay-per-use basis. SaaS providers generally price applications using a subscription fee.

In the SaaS model, cloud providers install and operate application software in the cloud and cloud users access the software from cloud clients. Cloud users do not manage the cloud infrastructure and platform where the application runs. This eliminates the need to install and run the application on the cloud user's own computers, which simplifies maintenance and support. Cloud applications are different from other applications in their scalability which can be achieved by cloning tasks onto multiple virtual machines at run-time to meet changing work demand. Load balancers distribute the work over the set of virtual machines. This process is transparent to the cloud user, who sees only a single access point. To accommodate a large number of cloud users, cloud applications can be multitenant, that is, any machine serves more than one cloud user organization.

The pricing model for SaaS applications is typically a monthly or yearly flat fee per user, so price is scalable and adjustable if users are added or removed at any point.

Proponents claim SaaS allows a business the potential to reduce IT operational costs by outsourcing hardware and software maintenance and support to the cloud provider. This enables the business to reallocate IT operations costs away from hardware/software spending and personnel expenses, towards meeting other goals. In addition, with applications hosted centrally, updates can be released without the need for users to install new software. One drawback of SaaS is that the users' data are stored on the cloud provider's server. As a result, there could be unauthorized access to the data. For this reason, users are increasingly adopting intelligent third-party key management systems to help secure their data.



Cloud computing types

Private cloud

Private cloud is cloud infrastructure operated solely for a single organization, whether managed internally or by a third-party, and hosted either internally or externally. Undertaking a private cloud project requires a significant level and degree of engagement to virtualize the business environment, and requires the organization to reevaluate decisions about existing resources. When done right, it can improve business, but every step in the project raises security issues that must be addressed to prevent serious vulnerabilities. Self-run data centers are generally capital intensive. They have a significant physical footprint, requiring allocations of space, hardware, and environmental controls. These assets have to be refreshed periodically, resulting in additional capital expenditures. They have attracted criticism because users "still have to buy, build, and manage them" and thus do not benefit from less hands-on management, essentially "[lacking] the economic model that makes cloud computing such an intriguing concept".

Public cloud

A cloud is called a "public cloud" when the services are rendered over a network that is open for public use. Public cloud services may be free or offered on a pay-per-usage model. Technically there may be little or no difference between public and private cloud architecture, however, security consideration may be substantially different for services (applications, storage, and other resources) that are made available by a service provider for a public audience and when communication is effected over a non-trusted network. Generally, public cloud service providers like Amazon AWS, Microsoft and Google own and operate the infrastructure at their data center and access is generally via the Internet. AWS and Microsoft also offer direct connect services called "AWS Direct Connect" and "Azure Express Route" respectively, such connections require customers to purchase or lease a private connection to a peering point offered by the cloud provider.

Hybrid cloud

Hybrid cloud is a composition of two or more clouds (private, community or public) that remain distinct entities but are bound together, offering the benefits of multiple deployment models. Hybrid cloud can also mean the ability to connect collocation, managed and/or dedicated services with cloud resources.

Gartner, Inc. defines a hybrid cloud service as a cloud computing service that is composed of some combination of private, public and community cloud services, from different service providers.[64] A hybrid cloud service crosses isolation and provider boundaries so that it can't be simply put in one category of private, public, or community cloud service. It allows one to extend either the capacity or the capability of a cloud service, by aggregation, integration or customization with another cloud service.

Varied use cases for hybrid cloud composition exist. For example, an organization may store sensitive client data in house on a private cloud application, but interconnect that application to a business intelligence application provided on a public cloud as a software service. This example of hybrid cloud extends the capabilities of the enterprise to deliver a specific business service through the addition of externally available public cloud services.

Another example of hybrid cloud is one where IT organizations use public cloud computing resources to meet temporary capacity needs that cannot be met by the private cloud. This capability enables hybrid clouds to employ cloud bursting for scaling across clouds. Cloud bursting is an application deployment model in which an application runs in a private cloud or data center and "bursts" to a public cloud when the demand for computing capacity increases. A primary advantage of cloud bursting and a hybrid cloud model is that an organization only pays for extra compute resources when they are needed. Cloud bursting enables data centers to create an in-house IT infrastructure that supports average workloads, and use cloud resources from public or private clouds, during spikes in processing demands.

Other clouds

Community cloud

Community cloud shares infrastructure between several organizations from a specific community with common concerns (security, compliance, jurisdiction, etc.), whether managed internally or by a third-party, and either hosted internally or externally. The costs are spread over fewer users than a public cloud (but more than a private cloud), so only some of the cost savings potential of cloud computing are realized.

Distributed cloud

Cloud computing can also be provided by a distributed set of machines that are running at different

locations, while still connected to a single network or hub service. Examples of this include distributed computing platforms such as BOINC and Folding@Home. An interesting attempt in such direction is Cloud@Home, aiming at implementing cloud computing provisioning model on top of voluntarily shared resources.

Inter cloud

The Inter cloud is an interconnected global "cloud of clouds" and an extension of the Internet "network of networks" on which it is based. The focus is on direct interoperability between public cloud service providers, more so than between providers and consumers (as is the case for hybrid- and multi-cloud).

Multicolor cloud

Multicolor is the use of multiple cloud computing services in a single heterogeneous architecture to reduce reliance on single vendors, increase flexibility through choice, militate against disasters, etc. It differs from hybrid cloud in that it refers to multiple cloud services, rather than multiple deployment modes (public, private, and legacy).

CMM- Capability Maturity Model

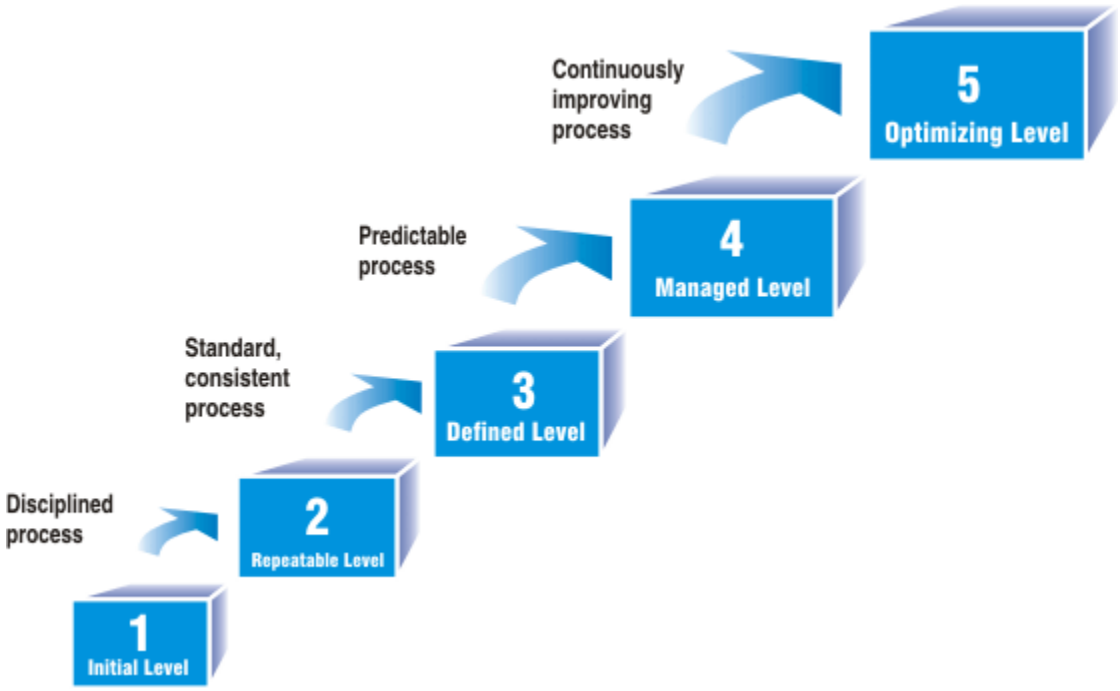
CMM (Capability Maturity Model) is a model of process maturity for software development - an evolutionary model of the progress of a company‘s abilities to develop software. In November 1986, the American Software Engineering Institute (SEI) in cooperation with Mitre Corporation created the Capability Maturity Model for Software.

Development of this model was necessary so that the U.S. federal government could objectively evaluate software providers and their abilities to manage large projects.

Many companies had been completing their projects with significant overruns in schedule and budget. The development and application of CMM helps to solve this problem. The key concept of the standard is organizational maturity.

A mature organization has clearly defined procedures for software development and project management. These procedures are adjusted and perfected as required.

In any software development company there are standards for processes of development, testing, and software application; and rules for appearance of final program code, components, interfaces, etc.



The CMM model defines five levels of organizational maturity

1. Initial level is a basis for comparison with the next levels. In an organization at the initial level, conditions are not stable for the development of quality software. The results of any project depend totally on the manager‘s personal approach and the programmers‘ experience, meaning the success of a particular project can be repeated only if the same managers and programmers are assigned to the next project. In addition, if managers or programmers leave the company, the quality of produced software will sharply decrease. In many cases, the development process comes down to writing code with minimal testing.

2. Repeatable level. At this level, project management technologies have been introduced in a company. That project planning and management is based on accumulated experience and there are standards for produced software (these standards are documented) and there is a special quality management group. At critical times, the process tends to roll back to the initial level.

3. Defined level. Here, standards for the processes of software development and maintenance are introduced and documented (including project management). During the introduction of standards, a transition to more effective technologies occurs. There is a special quality management department for building and maintaining these standards. A program of constant, advanced training of staff is required for achievement of this level. Starting with this level, the degree of organizational dependence on the qualities of particular developer‘s decreases and the process does not tend to roll back to the previous level in critical situations.

4. Managed level. There are quantitative indices (for both software and process as a whole) established in the organization. Better project management is achieved due to the decrease of digression in different project indices. However, sensible variations in process efficiency may be different from random variations (noise), especially in mastered areas.

5. Optimizing level. Improvement procedures are carried out not only for existing processes, but also for evaluation of the efficiency of newly introduced innovative technologies. The main goal of an organization on this level is permanent improvement of existing processes. This should anticipate possible errors and defects and decrease the costs of software development, by creating reusable components for example.

The Software Engineering Institute (SEI) constantly analyzes the results of CMM usage by different companies and perfects the model taking into account accumulated experience.

INFORMATION MANAGEMENT

Question Bank

Part A

1. What is meant by an Information Management?
2. List down the resources of Information management.
3. Compare information and data.
4. What is System Software and Application software?
5. What are the types of Information system?
6. What is System Development Methodology?
7. List down some of the common System Development Methodology
8. What is meant by Transaction Process system?
9. Write about Decision Support System.
10. What is Management Information Systems?
11. List some examples for MIS?
12. What is an EIS?
13. What is software prototyping?
14. What is the Different functional information system?
15. What is meant by executive information System?
16. What is Geographic information system?
17. List down the characteristics of Information
18. State the different types of Computers?
19. List down the components of DSS.
20. What is Information Technology?
21. Define SDLC?
22. List out the various SDLC Models?
23. Define Software Engineering
24. What is Data Dictionary?
25. What is the need for DFD?
26. What is system flow chart?
27. What is system analysis and design?
28. What is feasibility Study? Discuss its types.
29. What is structured analysis?
30. What is end user computing? List its advantages.
31. What are the possible risks associated with end user computing?
32. Define system design. List its objectives.
33. What are the major advantages of structured programs?
34. What is the purpose of structured walkthrough?
35. What is a Decision table?
36. What is Data Flow Diagram?
37. What is an Entity?
38. What is an Attribute?

39. What is Object Oriented Analysis and Design?
40. Define DBMS
41. Define data and information.
42. What are the four major components of a database system?
43. What is database?
44. What are the advantages of database system?
45. What are the Limitations of File Systems?
46. What is Data Dictionary?
47. What are the objectives of DBMS?
48. Define data model.
49. How data models are classified?
50. What is ER diagram?
51. What are the various types of attributes?
52. What is relational data model?
53. What is a Primary Key?
54. What is a Foreign Key?
55. What is an object?
56. What is encapsulation?
57. Identify the framework of a Information System.
58. Define MIS. Give an example.
59. What are structured Programs?
60. What are the features of Modern information systems?
61. Differentiate between DSS and MIS
62. Where are Expert Systems used?
63. Write any two activities carried out by HRIS.
64. Why Cost Benefit Analysis is carried out?
65. What is the difference between Intranet, Extranet and Internet?
66. Why Quality Control is necessary is Information System Design?

PART – B

1. Develop a MIS for a manufacturing organization indicating the different types of information subsystems depending on functional areas. High light the flow of information and the corresponding levels of information. What are the types of reports the system would generate.
2. i) Evolution of various computer based information systems.
   ii) Types of knowledge representation in an Executive information system and expert system.
3. Explain the stages involved in traditional system life cycle development.
4. What are the tools used in structured methodologies of system development and indicate their significance over traditional tools.
5. What is encryption and how can it be used for security purposes?
6. Name any four subsystems of personal system in an organization and design the information subsystem for them.
7. Classify the errors encountered during data entry and how can they be resolved.
8. Explain the methods of testing the security of the information systems.
9. Explain the method of conducting cost benefit analysis of Information system.
10. Describe the procedure involved in building a MIS for marketing Airline Industry.
11. Describe the types of decision and related information requirements in an organization.
12. Explain the functions of an information system.
13. Draw the DFD for course registration in any university. Indicate all the important parts.
14. How is object-relational DBMS used in information management? Explain.
15. What are the benefits of DSS? Explain in detail.
16. How can you design Marketing Information Systems? Discuss.
17. Discuss the common threats to computerized information systems.
18. Describe some of the general and applications controls.
19. What are the characteristics of e-business? Describe in detail.
20. Explain the scope of business intelligence.
21. Explain the components of IT.
22. Explain knowledge management system? And components of KMS?
23. Explain the steps of SDLC.
24. What is UML diagram? Explain the activity and use case diagram with example.
25. What is database? Explain the network data model with example.
26. Explain the structure and components of data warehouse.
27. Explain the computer crime? Explain hacking and cracking?
28. What is software audit? Explain the process of software audit?
29. What is E-business? Explain the applications of E-business.
30. What is cloud computing? Explain the components and models of cloud computing.