

PREFACE

In the curricular structure introduced by this University for students of Post-Graduate degree programme, the opportunity to pursue Post-Graduate course in Subjects introduced by this University is equally available to all learners. Instead of being guided by any presumption about ability level, it would perhaps stand to reason if receptivity of a learner is judged in the course of the learning process. That would be entirely in keeping with the objectives of open education which does not believe in artificial differentiation.

Keeping this in view, study materials of the Post-Graduate level in different subjects are being prepared on the basis of a well laid-out syllabus. The course structure combines the best elements in the approved syllabi of Central and State Universities in respective subjects. It has been so designed as to be upgradable with the addition of new information as well as results of fresh thinking and analysis.

The accepted methodology of distance education has been followed in the preparation of these study materials. Co-operation in every form of experienced scholars is indispensable for a work of this kind. We, therefore, owe an enormous debt of gratitude to everyone whose tireless efforts went into the writing, editing and devising of a proper lay-out of the materials. Practically speaking, their role amounts to an involvement in invisible teaching. For, whoever makes use of these study materials would virtually derive the benefit of learning under their collective care without each being seen by the other.

The more a learner would seriously pursue these study materials the easier it will be for him or her to reach out to larger horizons of a subject. Care has also been taken to make the language lucid and presentation attractive so that it may be rated as quality self-learning materials. If anything remains still obscure or difficult to follow, arrangements are there to come to terms with them through the counselling sessions regularly available at the network of study centres set up by the University.

Needless to add, a great part of these efforts is still experimental-in fact, pioneering in certain areas. Naturally, there is every possibility of some lapse or deficiency here and there. However, these do admit of rectification and further improvement in due course. On the whole, therefore, these study materials are expected to evoke wider appreciation the more they receive serious attention of all concerned.

Professor (Dr.) Subha Sankar Sarkar

Vice-Chancellor

Eighth Reprint : November, 2018

Printed in accordance with the regulations of the
Distance Education Bureau of the University Grants Commission.

Post Graduate : Library and Information Science
[MLIS]

Paper - VI
Information Technology : Application
Modules : 1 – 4

: Course Writing :
Sri. Arup Ray Chaudhury

: Editing :
Mr. K. K. Kochukoshy

Notification

All rights reserved. No part of this Book may be reproduced in any form without permission in writing from Netaji Subhas Open University.

Mohan Kumar Chattopadhyay
Registrar



Module : 1

Library Automation

Unit 1	□ Planning and Implementation of Library Automation	7–29
Unit 2	□ Computer Based Acquisition Control	30–42
Unit 3	□ Computer Based Cataloguing and Character Encoding	43–76
Unit 4	□ Computer Based Serials Control	77–87
Unit 5	□ Retrospective Conversion and Bar-coding	88–109

Module : 2

Database Management

Unit 6	□ Data Models	110–123
Unit 7	□ Software System	124–141
Unit 8	□ Online Public Access Catalogue (OPAC)	142–153
Unit 9	□ Database Structure, Organization and Search	154–165

Module : 3

Operating System and Programming

Unit 10 □ Operating System	166–182
Unit 11 □ Multi User Operating Systems	183–203
Unit 12 □ Programming Languages and Algorithm	204–218
Unit 13 □ Flowchart	219–228
Unit 14 □ Search, Sorting Algorithm and Data Structure	229–238

Module : 4

Operating System and Programming

Unit 15 □ Resource Sharing through Networks	239–248
Unit 16 □ Networks and their Classification	249–269
Unit 17 □ Network Architecture and Services	270–290
Unit 18 □ Bibliographic Information Networks	291–309

Unit 1 □ Planning and Implementation of Library Automation

Structure

- 1.0 Objectives**
- 1.1 Introduction**
- 1.2 Library Automation**
- 1.3 Benefits of Library Automation**
- 1.4 Barriers to Library Automation**
- 1.5 Modules and Functions**
- 1.6 Planning and Implementations**
- 1.7 Basic Library Automation Standards related to Technology**
- 1.8 Costs**
- 1.9 General System Requirements**
- 1.10 Library Management System Checklist**
- 1.11 Exercise**

1.0 Objectives

The objectives of the Unit are to :

- Discuss benefits of and barriers to library automation
- Identify basic modules and functions of library automation activities.
- Outline strategy for the planning and implementation of library automation in a library.
- Enumerate library automation standards related to technology.

1.1 Introduction

Planning for an automated system should be part of an overall long-range plan for the library. It is a mistake to let the automation project drive the library's priorities and become an end in itself. Automation should always be used as a means to achieve overall better member service. Libraries must plan to use a local library system as a vehicle for achieving access to resources outside that system. The Internet has created universal connectivity to information resources heretofore unknown and/or inaccessible and Z39.50 interoperability standards and "gateways" has enabled users of individual local systems to access the resources of other systems-anywhere and anytime.

Moreover, the traditional definition of “publishing” has been stretched by the creation and instant availability of informational home pages and Web sites worldwide. Given such increased complexities and heightened levels of expectation, libraries must learn all the more how to plan for the introduction of automation in an organized and systematic fashion.

1.2 Library Automation

Library automation is the use of computer to perform such traditional library activities as acquisition, cataloguing, and circulations. Now-a-days, it encompasses related fields such as information retrieval, inter library loan services, access to online/ internet based resources, and access to external databases.

1.3 Benefits of Library Automation

The benefits of application of computer in housekeeping operations and providing information services in a library are :

- Improved library management. Automation improves library services and increases productivity, efficiency, and accuracy in performing a variety of library operations.
- Linking items of library material in all service points to catalogue records
- Increase the opportunity for inter library loans
- Increase job satisfaction and professional development for library professionals.
- Provide a cohesive package of library services to library members in the library building and online, by coordinating computers, communications, content, and staff competencies.
- Maintain and improve access to worldwide library resources and services via library-automated systems.
- Provide digital tools for library members to access, retrieve, evaluate, and use information resources.
- It allows members to use search strategies that exceed those that can be used with card catalogues. Card catalogues can be search only by author, title, and subject. OPAC can search by using Boolean operators (AND, OR, NOT) and by combining search strategies (e.g. title and author, subject and author etc.)
- OPAC users may limit their search results by such features as publication date, type of material (magazine, book, and video), language, or reading level, and they can sort bibliographics by author, title and publication date.

- The Windows-based OPAC allow for hyperlink searching, a new feature that was not possible in character-based systems (i.e. DOS). Through a hyperlink search a user can find related records in the automated system's database under the keyword or subject.
- It allows members to search the library's collection from locations outside the library's walls. Members who are equipped with a computer and a modem can dial into the OPAC from home, an office, or another remote location.
- Most automation software is compatible with the Z39.50 standard. Having this standard allows users to search OPAC on the Web using common interfaces and/or search features. This means that regardless of the automation software, operating system (Windows 98, Windows XP), or computer platform users have, they can search these OPAC using common search interfaces.
- It provides users with timely access to library materials. Library materials can be placed on shelves as soon as items are processed
- It eliminates routine tasks or performs them more efficiently. The circulation function, which includes check-in, checkout, overdue notices, and inventory, is tedious, repetitive and time-consuming. Automating these functions can save a tremendous amount of time.
- It expedites and simplifies the inventory of library materials. The automated inventory is performed by scanning each item's barcode using a hand-held device (scanner), downloading scanned items into the automated system, and generating a variety of customized reports.
- It encourages cooperative collection development and resource sharing (inter-library loan). Automated media centres and libraries can develop a union catalogue and join bibliographic consortia. A user who does not find an item of interest in the library's local OPAC may identify the libraries in the union catalogue or consortia that have it. Such an item can be borrowed through inter-library loan or by checking to out from a designated library.
- It enables libraries to import MARC (Machine Readable Cataloguing) records. Records can be exported from one system and imported into a new automated system without incurring costs for retrospective conversion. Exploring records is essential for migrating from one automated system to another.
- It reduces (in integrated systems) the amount of time spent on material acquisitions, serials management, budget administration, and record keeping.
- It motivates members, equips them with problem-solving and information retrieval skills and provides them with lifelong learning experiences. In

addition, it reinforces a positive attitude about the media centre or library and improves the image of the librarian or information professional. Members view the media centre or library as an indispensable place for gaining access to global information and consider the librarian or information professional a powerful information provider.

1.4 Barriers to Library Automation

- It is time-consuming. Planning, selecting, and implementing an automated system requires a significant, long-term commitment of staff time. Once selected, implementing an automated system must be maintained on a regular basis.
- It is costly. Start-up costs, software, hardware, network cabling, wiring, furniture; ongoing expenses, such as supplies for printers and barcode labels; annual maintenance and technical support; and conversion of a library's shelf list into a machine-readable format (MARC) may be more than what many libraries may afford.
- The demands of the automated system may not leave staff adequate time to provide new services or to work with students, or other educators. In fact, automation eliminates some tasks but generates new ones. Training scholars/pupils to use the system, ongoing troubleshooting of hardware and software and database maintenance place demands on the library professionals.
- Access to the automated system is unavailable during system downtime. This will hamper user access to the collection, especially if the card catalogue or the shelf list no longer exists in the library.
- Training is the neglected part in library automation activities. The training for handling the software is usually provided by the developers and the cost is included in the price of the software. This vendor provided training is one time and it is certainly not sufficient for successful implementation of the automation programme. Additional training may also be necessary for system software, backup software, and successful management of network. Staffs do need hardware training. The management must appreciate the fact that the training is a continuous process.
- Retrospective conversion of data is a problem for many old and large libraries. As mentioned earlier, manpower saved in other activities could be utilized. Otherwise, outsourcing may be the only option. However, active participation of staffs can only ensure quality of the work.

1.5 Modules and Functions

Planning for library automation covers various facets : automation of library functions, use of electronic resources within the library (e.g. CD-ROMs), accessing remote electronic resources (e.g. the Internet), Office automation (e.g. word-processing, spreadsheets, databases, etc.), and member services (e.g. computer laboratory, multimedia center). Major modules are :

- **Acquisition Module** : It includes tasks such as : periodical material requests, purchase orders, material receipt, budget, vendor performance tracking, and record keeping.
- **Cataloguing Module** : This module performs various cataloguing tasks such as : original cataloguing using the Machine-Readable Cataloguing (MARC) protocol, editing, copying, saving, and retrieving catalogued records. When a record is saved in the cataloguing database, the record automatically appears on the OPAC, and a brief copy of the record is also generated automatically for the circulation module.
- **OPAC Module** : The OPAC function allows searching by author, title, subject, or keyword; searches using Boolean operators (AND, OR, NOT); hyperlink searching (i.e. to find related records under specific words or subjects of interest that appear in a record); Wild card/character searching (i.e. word truncation); and combined search strategy options (e.g. author-title; author-subject). The OPAC module is the only one that is inseparable from cataloguing. The library cannot have the OPAC without the cataloguing module, because the cataloguing module is the database that houses all material records and makes them available to the OPAC. Therefore, the cataloguing module is the heart of the automated system.
- **Serials Module** : The serials function is achieved in the serials module. It covers tasks such as periodicals (Journal) subscriptions, acquisitions, routing, claiming, cancellations, budget, tracking vendor performance, and record keeping.
- **Circulation Module** : Performs the tasks involved in the circulation function, such as : material check-in, check out, inventory, overdue notices, holds and reserves, fines, and statistical reports.
- **Inter library Loan Module** : The interlibrary loan function is accomplished in the interlibrary loan module. This module performs various tasks for the borrowing and lending of materials among libraries.

1.6 Planning and Implementation

Planning is a task or activity that is required on continuous basis for the management of library automation. Planning for library automation has been defined as planning for “integrated systems” that computerize an array of traditional library functions using a common database. While this is still generally true, rapid technological change is forcing a re-examination of what it means to “automate the library.” As physical, spatial and temporal barriers to acquiring information are crumbling, libraries must plan for a broader and more comprehensive approach in providing automated services. Results of planning would be :

- It provides basic framework for actions and services
- It minimizes ad-hoc decisions
- It helps in identifying and differentiating the essential priority actions and not so essential actions
- It helps concentrated and cohesive actions by a group towards achieving the ultimate goals.
- It helps to draft a financially elastic budget, which is capable of readjusting itself to a reducing or enhancing of financial resources.

The term implementation encompasses aspects of evaluation, selection, acquisition, training and installation of software, hardware and training of library professionals and members. Library automation involves several changes. Automation may change

- Roles of different staff members
- Organizational structure
- Workflows, and
- Service patterns etc.

Staff needs to know about the automation plan and how it would impact his/her work/lives. The possibilities of job enrichment of operational staff as a result of automation should be highlighted and planned for well in advance of the actual acquisition and installation of the system.

It is absolutely essential that the library plan in consultation with a committee of representatives of users, the management, representatives from library, in addition, a consultant may be included. However, the planning committee to function efficiently and effectively, there is a need for information, which may either be readily available

or may have to be collected. The different stages of planning for library automations may be classified as follows :

- Pre-Planning Phase
 - ◆ Phase I : Data collection and needs assessment
 - ◆ Phase II : Examination of automation options
- Planning Phase Proper
 - ◆ Phase III : Development of Bibliographic Databases
 - ◆ Phase IV : System Specifications and Requirements
 - ◆ Phase V : Analyze Proposals and Select the Vendor
 - ◆ Phase VI : Start Negotiation with the Top Vendor
 - ◆ Phase VII : Implement the System
 - ◆ Phase VIII : Monitoring and Evaluation

1.6.1 Phase I : Data Collection and needs assessment

One of the most important planning tools involves collecting basic statistical information on the library and its operations. This data will be used for varieties of purposes throughout the planning process, including for preparing preliminary cost estimate, identification of storage requirement and hardware and software requirements. If certain data are not available, reasonably estimated data will be sufficient. Large amounts of time need not be devoted to determine a perfectly exact number. The tasks involved are :

- Study, analyze, and document the current system including review of policy and procedure manuals, training manuals and work flow patterns
- Communicate with users and staffs. Identify core staffs. Conduct surveys about staffs, management and users.
- Collect and compile narrative and statistical information necessary to describe library system and its operations.
- Assess the library's service mandates
- Assess the library's operating environment including physical facilities, budget, user relations, and inter library cooperation/cooperative responsibilities.

It may be necessary to collect data on number of users, user's licenses, number of online users, collection size by types, growth of collection and users, description of any co-operative arrangements involving the library, and needs for peripheral equipment needs by applications (Acquisition, Technical Processing and OPAC etc).

In addition, it is important to take stock of any existing automation in the library by compiling the following data : percentage of collection that has catalogue records in machine-readable form; description of collection without machine-readable records, by category (e.g. monographs, audiovisuals); description of currently-automated library functions (if any); estimates of the location and number of workstations (to show where you intend to have equipment in any future system); and, specifications for any existing equipment to be re-used with any future system (if any).

At the same time that this data is being assembled, it is important to assess user needs and set service priorities. This can be accomplished by undertaking a focused, strategic planning process designed to involve the library's "stakeholders".

1.6.2 Phase II : Examination of Automation Options

A library planning to automate should undertake a process by which representative staff and users can identify service needs and objectives. The purpose of such an effort is to allow participants to articulate their interests and concerns, share perspectives and learn about possibilities in a collaborative setting. Group interaction is an important contributing factor in the success of the goal, which is to develop and sustain library automation in the years ahead. The goal is to determine what services could best benefit from automation. It is necessary to :

- Determine which functions should be automated and in what order of priority. Hence, it is necessary to know how the services are currently organized and being performed, and whether they are being done efficiently.
- Need to study and analyze current procedures. It involves identification of
 - Repetitive processes, which occupy large amount of staff time.
 - Processes, which require retrieval of information from large difficult to maintain files.
 - Most popular services with the users
 - Current workflow patterns
 - Volume of activities
 - Space and equipment are used
 - Cost to perform
 - Current problems/needs in each area.

Library is an integrated system. The various modules of library activities are OPAC, Cataloguing, and Circulation System etc. The tasks involved at this phase are :

- Set priorities for what services to automate in a phased approach
- Begin defining the steps for a phased and coordinated plan to automate

- Analyze hardware, software, and telecommunication needs
- Identify start-up and ongoing costs for automation plan
- Develop a budget for automation

If needs and priorities are clear, functions can be automated in phases, allowing for more effective use of frequently scarce funding. Moreover, it is a way to develop credibility with funding agencies and be able to take advantage of “sudden” funding opportunities. Finally, evaluations of systems and options will be easier and more productive if the highest functional priorities can be matched against the corresponding modules available in the marketplace.

1.6.3 Phase III : Development of Bibliographic Databases (Retrospective Conversion)

The goal is to establish a complete database of library’s holdings that can be ready to load when the new library system is installed. The tasks involved are :

- Identify, describe and document existing shelf list files and begin to standardize the data they contain.
- Verify existing bibliographic information in all shelf list records
- Institute quality control measures to assure consistency of entries within the shelf list file. Locate and add necessary missing bibliographic information to the records
- Inventory the collection
- Implement standardized cataloging practices including book mark etc.
- Prepare specifications for treatment of bibliographic data and item data (including authority control needs)
- Prepare and distribute a Request for Proposals for (RFP) Vendor
- Evaluate proposals and select a vendor
- Organize staff and define procedures for working with vendor to resolve bibliographic record conflicts.

The creation of a high-quality machine-readable database provides the cornerstone upon which all-present and future automation efforts rest. Hardware will become obsolete, software will be replaced, but a well-constructed and well maintained database will be the library’s transportable and viable link from system to system. Moreover, as library users begin to access not only their local system but systems in other libraries as well, the quality of respective databases will influence both the outcome of search strategies and the availability of materials. Database readiness has several important facets :

- Catalogue records must be carefully converted from manual to machine-readable formats;
- Collections must be prepared for conversion through effective and ongoing weeding and inventory programmes;
- Once converted, collections must be properly maintained as titles are added, withdrawn, transferred and re-cataloged; and,
- Standards-for bibliographic, item and member records as well—must be adhered to. In particular, adherence to well-established and accepted standards of description for bibliographic information in a machine-readable database is critical because :
 - Without standards, files cannot easily be transferred from one automated system to another and,
 - It is essential for libraries wishing to participate in resource sharing arrangements with other libraries, which will require such adherence as a condition of participation.

1.6.4 Phase IV : System Specifications and Requirements

It is very difficult to compare systems sensibly and pragmatically solely by randomly looking at systems, talking to sales representatives, reading literature or comparing broad cost quotations. For this reason, libraries use a formal document—often known as a “Request for Proposal,” or RFP—that organizes and standardizes the information provided to and requested from the various system vendors. Utilizing an RFP to solicit written responses from vendors makes it possible to systematically compare functionality, cost, maintenance, support, and all the other issues that are involved in system procurements. The process can save you money and will result in a wiser decision. Prepare an RFP with specifications for an automated system that will meet the service objectives and requirements for the library. Compare relative differences from one system to another.

It is necessary to break the collected data into functional specifications and technical specifications. Functional specifications describe the capabilities that are expected from the system. Technical specifications deal with standards that must be adhered to guarantee minimum system performance. An RFP document should include these essential elements, among others :

- Background information on the library;
- A description of how the proposals should be arranged and submitted;
- Instructions on receiving vendor business and financial information;
- Criteria the library will use to evaluate vendor proposals;

- Questions regarding vendor training and documentation;
- Write functional and technical specifications.

Also, vendors should be asked to describe :

- How they will create bibliographic, item and borrower databases;
- Their system maintenance programmes and services;
- Their site preparation requirements;
- Their delivery and installation methodologies;
- Their system performance guarantees; and
- Their pricing and cost strategies, in detail

1.6.5 Phase V : Analyze Proposals and Select the Vendor

Upon the receipt of vendor proposals, it will be time to begin the process of system evaluation and selection. Goal is to analyze vendor responses to RFP and to select a vendor to implement a system. This process involves a number of key steps :

- Try to weed out proposals that are flawed, e.g., where the vendor fails to reply to any of the functional specifications or the system is missing a module for a high-priority function.
- Begin in-depth reading of the “surviving” proposals, carefully noting both deviations from the requirements as defined by the RFP and any aspect that is handled unusually well. Make a list of any parts of the response that are not clear and require further clarification.
- Schedule system demonstrations. They are an important component of the evaluation process. Allow the vendors to show off the vendor’s system in the most attractive light; however, be prepared with a list of what you want to see along with questions you would like answered. Use the same list with each vendor. This permits more effective cross-comparisons.
- Contact some of each vendor’s current clients-sites of the same library type, and of similar size, where the hardware and software modules that have been proposed to you are currently in use.
- Assign point values to the criteria listed in the RFP and assign scores to the different proposals. The system with the highest score becomes the number one finalist, the system with the second highest score number two and so on. To maintain a negotiating edge, it is better to cut to two vendors rather than one.

1.6.6 Phase VI : Start Negotiation with the Top Vendor

It involves legal counsel, together with library personnel, in drafting or evaluating a contract. Compare list of necessary contract elements to actual contract. Set up a Contract Negotiation Agenda. Bring negotiations to a successful and favourable conclusion. The goal is to purchase a library automation system. The purposes are to :

- Interpret and clarify the differences between a vendor's response and the library's specifications.
- Formalize pricing and payment schedules, warranties, vendor bankruptcy, software infringement, and maintenance conditions.
- Safeguards conformance to any legal requirements necessitated by library's parent organization etc.
- Consult legal expert to analyze terms and conditions.

1.6.7 Phase VII : Implement the System

The goal is to install the selected system. After the system selection process is completed, the library and the vendor will have to negotiate and sign a contract. Library will want to

- Test the system and make sure it suits its needs.
- Make provisions for system maintenance, and
- Train both staff and users as much as possible to prepare them for when the system is up and running. There is often a tendency to focus on the hardware and software aspects of planning, and to ignore the human aspects of automation-training and public relations. Without these, however, library staff or library users may not accept even the most carefully designed system. Public relations allow to :
 - Make users aware of the new system and services;
 - Motivate them to use the system; and,
 - Train them in using the new system and services effectively.

Planning the shift from the existing system to the new system can take one of the approaches depending on complexity of the selected software and preparedness etc.

- Phase implementation : The new system consists of different functional modules. Modules may be installed at different point of time.
- Parallel Implementation : Both manual/old and new systems are operated simultaneously for certain time until the reliability of the new system is established.

- Pilot Implementation : The system may be installed on a smaller scale. This enables better understanding of problems and complexities of the new system.
- Complete Replacement : The old/manual system is replaced on a specific date. This requires highest level of preparedness on the part of the library and good understanding of the new system.

The library as an alternative to, or in conjunction may require a performance bond with a benchmark test. The performance bond needs to be incorporated and fully described in the RFP. Typical performance test involves :

- System Reliability Test
- Functional performance acceptance tests
- Full-load response time test

The tasks involve in this phase are :

- Customize the vendor's system to the library's policies
- Site preparation
- Installation of hardware and software
- Acquire the necessary forms and peripherals equipment
- Load and index the bibliographic databases
- Load and index the user databases
- Train staff, realign workflow and space
- Activate the system
- Evaluate operation and do appropriate testing
- Accept the system

Given the challenge of working on a library automation initiative, there are two parallel opportunities for information professionals. One role is on the data processing and systems implementation side, working hand-in-hand with the IT group (people that control the hardware and networks). The other is on the content management side. Often, due to limited resources one person has to serve both roles.

1.6.8 Phase VIII : Monitoring and Evaluation

Once the system has been installed and in operation for reasonable time period, its performance should be evaluated. Some of the parameters are :

- Type of Routine reports available : Order, Overdue notice, Alert, Claims, Reminders, Accession Registers, Fund Control
- Continuous development of the software
- Robustness of the software

- Errors encounters and response time taken by the Vendor
- Response time, Disk space utilization and quality of Backup

Computer technology and software applications are changing and evolving at an incredibly rapid pace. In general, a life cycle of five years is considered to be acceptable for a computer system before some significant upgrade (installation of additional hardware and/or software providing for increased capability or capacity) or replacement will be necessary.

Because computer and information technology represent a fundamental change in the way libraries do business, libraries must make an ongoing commitment to keeping pace with change. Therefore, like automated systems, plans must also change with time. Plans must be regularly revisited and updated as the environment and needs change. In general, a library should conduct a major re-examination of its plan every five years, and should review its plans on an annual basis.

1.7 Basic Library Automation Standards related to Technology

The following standards and guidelines address a variety of technical issues in the area of library automation. In addition to the core elements of a library automated system (Public access catalogue, cataloguing, circulation, acquisitions, and serials), remote access, imaging, and full-text document management are involved. Because these standards address only a limited number of topics, they are not sufficient by themselves for developing a comprehensive set of specifications.

1.7.1 Z39.50

Z39.50 is an international standard for communication between two computers systems, usually library or information systems. It is a type of software that makes it possible to search multiple disparate library catalogues and other other resources in one search, and bring back one set of results. Z39.50 version 3, introduced about two years ago, displays holdings status of result sets, a necessary component for interlibrary loan. Version 2.0.0. 3 should support ISBN searching.

There is also a client component to Z39.50 for searching library catalogues and downloading records. Bookwhere, a product by Web Clarity, is an example of a Z39.50 client; it can be configured to search multiple Z39.50 resources such as the Colorado Virtual Library, The Library of Congress, and any other database that have a Z39.50 server. The benefits are :

- Broadcast searching— The Z39.50 server allows libraries to be part of a “virtual union catalogue” such as the Colorado Virtual Library where multiple libraries can be searched simultaneously.

- Bibliographic resource sharing— A Z39.50 client allows libraries to share cataloguing resources
- Interlibrary Loan-resource sharing is possible with Z39.50 version 3. Users can view library holdings and shelf status in real time
- Most of the key automation vendors have Z39.50 servers that you can purchase as a separate piece or that comes bundle with other modules. Check with your automation vendor to find out where they are with Z39.50 server and client implementation.

1.7.2 Inter-library Loan Standards

This standard is a protocol for performing inter-library loan transactions between disparate online catalogues. Z39.50 retrieves the bibliographic record, while ISO 10160/61 allows you to place a hold on the record.

1.7.3 Circulation Interchange Protocol

Though not implemented to date, the Circulation Interchange Protocol standard will support circulation transactions between different library systems. The circulation of print and electronic materials, remote member authentication, member borrowing, online payment, and controlled access to electronic documents are some of the functions this protocol will support. This standard will be based on the standard interchange protocol (3M SIP), implemented to support self-checkout machines and developed by the 3M Corporation.

1.8 Cost

Speaking of funding, planners need to be aware that there are certain cost elements involved in the installation and operation of any automated system. These may be summarized as follows :

- Planning and Consulting Costs include direct, out-of pocket costs (e.g., hiring a consultant) and indirect costs (e.g., training staff) associated with getting started.
- Purchase of the System includes the cost of acquiring the initial system hardware and software, as well as the cost of preparing a site for the computer system.
- Telecommunications costs are those fees paid to telephone companies for connecting remote terminals or workstations to a central computer system and Internet facility.
- Conversion costs are those associated with the creation of machine readable bibliographic and, for circulation systems and member records.
- Ongoing Operating Costs include :

- Bar code labels
- Maintenance fees
- Utility costs
- Miscellaneous supply costs
- Telecommunications costs
- Salaries and benefits (if extra staff are hired)

1.9 General System Requirements

The following table presents parameters for evaluation of library management software at functional modules independent step.

Parameters	Descriptions
System Architecture	<ul style="list-style-type: none"> ● The system must employ client/server architecture ● The system must place no limit on record size, other than the limits imposed by the MARC 21 standard. ● The system must be fully self-contained and capable of being operated by library staff with no dependency on vendor services for its routine operation. ● The system server proposed must operate under the Library's choice of operating system. ● The system must permit distribution across multiple servers. ● The system must be an open system, with no dependency on the use of specific models or models of equipment operating systems, RDBMS etc., to ensure the future viability of the system. ● The system must keep a log of each transaction which alters the database. Logs must be data and time stamped so as to allow the system to reconstruct activity for any period.

Parameters	Descriptions
Software Licensing	<ul style="list-style-type: none"> ● Vendor must include an unlimited license for web OPAC users connecting from OPAC workstations within the library, from other libraries, and from PC's in members, homes, offices, classrooms, etc. ● Vendor must license the software for perpetual use for a fixed fee without additional royalties or service fees, except for ongoing software maintenance.
Programming Language	<ul style="list-style-type: none"> ● The system's client applications must be written in C or similar programming language (s). ● The system must offer Application Programming Interfaces (APIs) that enable the Library to develop custom interfaces to all modules. APIs training and documentation must be quoted as an option.
Source Code	<ul style="list-style-type: none"> ● If Vendor ceases to be in the information management software business, or if the Vendor should be declared bankrupt by a court of competent jurisdiction, the Library shall have the right to access, for its own and sole use only, for maintenance use only, one copy of the source code. ● The library waives any claims to ownership or part ownership in the licensed source code or any modifications made to the source code.
Hardware Requirement Sever Client Network	<ul style="list-style-type: none"> ● Vendor must describe server platforms that may be used with the proposed system. ● Vendor must provide minimum requirements for staff PC workstations. ● The system must support a variety of TCP/IP network configurations. (The Library must provide actual network). ● The system must have the ability to support wireless LAN and WAN configurations that support TCP/IP.

Parameters	Descriptions
<p>RDBMS/Applications/ Database Management</p> <p>Industry Standards/Protocols</p>	<ul style="list-style-type: none"> ● System must be available with the Library's choice of a Vendor-developed or vendor independent relational database management system. ● Full-text indexing and a full-text database search feature must be available to provide easy retrieval of records. ● The system must support the following formats for bibliographic data : MARC 21, UNIMARC, and CCF etc.
Record Formats	<ul style="list-style-type: none"> ● Vendor must list other national and international standards with which the proposed system complies.
Z39.50 Bath Profile	<ul style="list-style-type: none"> ● The system, both clients and server, must be fully Z39.50 version 3 compliant and certified for bath Profile Level 1 compliant, across all Functional Areas defined in the Profile
Security/Access Control	<ul style="list-style-type: none"> ● The system must provide security at database, workstation, and individual operator levels. ● The system must provide secure access control based upon unique user login, for types of record (e.g., fund, order, and member) as well as by function performed upon the record (e.g., Display, Add, Edit, Delete.) ● The system must check each user's access privileges at login, and automatically disable or enable client functions (in real time) based upon the user's profile.
<p>Services</p> <p>System Implementation/ Data Migration</p>	<ul style="list-style-type: none"> ● Vendor must facilitate migration of the Library's present databases to the proposed system so that the system must be fully operational on 'Day One.' ● Migration must include bibliographic records (titles), items/copies, authority records, circulation transactions (charges, bills, holds), acquisitions (vendor, orders, funds), serials (control, check in, chronology)

Parameters	Descriptions
	<ul style="list-style-type: none"> ● Vendor must agree that the Library and the Vendor will mutually determine the details of the final implementation plan. ● Vendor must indicate any limitations or qualifications to the format in which Vendor must receive records in order to be migrated. ● Vendor must include a “gap” file or another process by which the databases may be brought up to date during the interval between export of the initial databases and completion of system installation and training.
<p>Training/Consulting/ Documentation</p>	<ul style="list-style-type: none"> ● Vendor must provide a brief description of training courses. ● Vendor must list the number of training days proposed, by type of training course (e.g., staff, administrative). ● The Library must have the option of videotaping training sessions for future library staff training purposes. ● Vendor must include a description of the complete documentation package available with the system. ● The cost of one set of complete documentation on all hardware devices, if purchased from Vendor, and all system and application software modules must be included in the Vendor’s proposal. <p>Documentation updates for all appropriate manuals must be provided on a regular basis as additional capabilities, enhancements, or improvements are made to the system.</p>
<p>Software Maintenance</p>	<ul style="list-style-type: none"> ● The proposed system must carry a minimum one year warranty under which software maintenance must be provided without added cost ● Maintenance of proposed software must be available from the Vendor on an annually renewable contract basis.

Parameters	Descriptions
	<ul style="list-style-type: none"> ● Vendor must provide a software maintenance programme to include all future software updates and system enhancements applicable to system modules ● Major system upgrades must be developed and released annually, so the library will receive the latest enhancements, regardless of the version of Vendor's system that the library originally installed. ● Software enhancements must be made available without further charge to all licensed libraries maintaining an annually renewable software support contract with Vendor.
Equipment Maintenance	<ul style="list-style-type: none"> ● The proposed system must carry a minimum warranty (three years for server; one year for any peripherals quoted) under which equipment maintenance must be provided without added cost. ● Vendor must agree to place any required service call for hardware related problems to any maintenance subcontractor proposed.

1.10 Library Management Checklist

There are certain libraries who have already procured library management software. It is responsibility of such library to evaluate the current system to a modern client/server system. One library may have to change the system if responses to the following questions are mostly negative.

Company

- Many years of dedicated experience in serving the technological needs of libraries.
- Steady revenue growth
- Customer base representing libraries like the present one
- No "Legacy system", i.e., the customers continue to use the original system updated annually with the latest technological advances.

Technology

- Choice of fully developed UNIX or Windows 2003 Server platforms
- Choice of RDBMS
- Full-text bibliographic search facility
- Centralized software and configuration management of all desktop clients in the network
- Easy windows wizards for all system administration functions

Flexibility

- Simple “interview wizards” to enable easy system setup and customization
- Unlimited size of parameter tables (Accession series, document formats, user categories, document types etc)
- Facility for global changes for parameter tables
- Ability to establish additional fields
- Fully documented API

Management Reports

- A variety of report templates
- Additional reporting via API and SQL
- Graphical report interface

OPAC/Web Gateway

- OPAC with Internet access
- Facility for customization of OPAC
- Icon based search facilities

Digital Media Archive

- Simultaneous searching of all formats
- Full-text and natural language searching
- Object should be stored in their native formats.

Client Design

- Server based client profiles enabled individual profiles to be moved automatically from one client PC to other PC
- Client are updated automatically with push technology
- Customization option for each user preferences.

Cataloging/Authority Control

- Z39.50 copy cataloging with automatic holdings posting to central network like OCLC/INFLIBNET etc.
- Automatic generation of authority headings for all necessary access points.

Acquisition

- Pre--Processing in cataloguing
- Divides order amounts and percentages among funds

Serials

- SICI based check-in of issues

Circulation

- Automatic e-mail notices/alerts for fines, reserved copy collect notice etc.
- Phone notification for all notices

Approval Processing

- Library oriented computerized suggestion and approval facilities.

References and Further Reading List

- 1 2005 Basics of library automation standards (<http://www.cde.state.co.us/cdelib/technology/atstan.htm>). Visited last: 20/09/2005
- 2 2005 Library management system checklist (<http://www.libraryhq.com/checklist.html>). Visited last 20/09/2005
- 3 2004 Haravu (LJ). Library automation : design, principles and practice. New Dlehi : Allied Publishers,. 2004.
- 4 2004 Encyclopedia of library and information science. 2nd ed. Ed by Drake Miriam A. NY : Marcel Dekker, 2004., 4v
- 5 1997 Cohn (John M), Kelsey (Ann L) and Fiels (Keith Michael). Planning for automation : a how-to-do-it manual for librarians. 2nd ed. New York : Neal-Schuman Publishers, Inc., 1997.
- 6 1997 Meghabghab (Dania Bilal). Automating media centers and small libraries : a microcomputer-based approach. Englewood : Libraries Unlimited, 1997.
- 7 1994 Harbour (Robin T). Managing library automation. London : ASLIB 1994
- 8 1989 Saffadv (William). Introduction to automation for librarians. 2nd ed. Chicago : ALA, 1989

- 9 1984 Genaway (DC). Integrated online library systems : Principles, planning and implementation. NY : GK. Hall. 1984.
- 10 1983 Rao (I.K Ravichandra). Library automation (DRTC Refresher seminar 14, DRTC, Bangalore, 1983)
- 11 1969 Encyclopedia of library and information science., Ed by Allen kent and Harold Lancour. NY : Marcel Dekker. 1969-. Various volumes

1.11 Exercise

1. Discuss the benefits and barriers of library automation.
2. Describe functions of major modules of library automation activities.
3. Discuss basic library automation standards related to technology.

Unit 2 □ Computer Based Acquisition Control

Structure

2.0 Objectives

2.1 Introduction

2.2 Benefits

2.3 Infrastructure

2.4 Sub-systems and their Functions

2.5 Selection Criteria

2.6 Interfaces with Other Library Automation Sub-system

2.7 File Structure for Acquisition System

2.8 Exercise

2.0 Objectives

The objectives of the Unit are to illustrate :

- Rationale of computer based acquisition system
- Requirements for infrastructure
- Sub-system of a computer based acquisition system
- File structures for acquisitions systems

2.1 Introduction

Libraries have automated acquisitions for a variety of reasons, including lowering unit costs, improving services, speeding the order-cataloguing process, accessing ordering and in-process information, collecting and organizing acquisition data, and linking with other systems both inside and outside the library. Automated acquisition system vary in complexity from order/receipt operations to fully integrated modules that are interactive with library system, including such functions as cataloguing, serials, circulation, and reserves.

The Acquisitions module must be fully integrated with all system modules and must support an unlimited number of material types/formats, funds, vendors, orders, claims and transactions, without added cost. The system must be dynamically updated in real time to maintain the currency of all records and statistics.

2.2 Benefits

The acquisition of documents (books/serials/CD-ROM etc.) is an important activity of any library. The acquisition function includes tasks that require human judgment and routine/clerical works. Computer based acquisition can greatly improve effectiveness and efficiency of library's housekeeping operations. The benefits are :

- Reduce drudgery and error proneness of works, which are of routine nature.
- Enhance information sharing : It reduces data redundancy and ensures consistency of data across the system
- Improve efficiency and timeliness : The various processes involved in the acquisition function can be performed more effectively and efficiently in a computer based acquisition system
- Provide various Management Information Reports : Provide valuable financial and other inputs for management decision making.
- Monitor status of items/activities : computer based system facilitate effective and efficient handling of claims, reminder, alert, receipt, cancellations, and tracking of on-order items etc.
- Optimize staff time and efforts : In an automated system, routine clerical works are left to the machine and thereby enable professional staff to concentrate more on intellectual work and customer care.

2.3 Infrastructure

The infrastructure necessary for a computer based acquisition system include :

- Access to external databases and sources of information
- Internal files and Databases

2.3.1 Access to external databases and sources of information

Verification of bibliographic data of proposed books is an important activity during acquisition. User supplied bibliographic data may not always be sufficient for taking decisions and procurement. The bibliographical verification process facilitates import of bibliographic data into the system. Such access leads to fuller and more accurate records in the acquisition files and enhance the quality of catalogue.

The kind of external databases to be accessed depends on the nature of the library and availability of resources. Another important decision in acquisition is whether to depend on resource sharing arrangements to obtain access to the publication. Access to the union catalogues of library networks/consortium or to OCLC becomes important in such situations.

2.3.2 Access to internal files and databases

In case of an integrated library automation environment, access to the following types of internal files/databases is important for acquisition, notification to individual requester, maintaining uniformity in entering author names, subject terms and publisher name etc., and to avoid unintended duplicate acquisition by verification against on-order and in-process files. The following files are examples of typical requirements :

- Library's Catalogue
- User/Membership file
- Exchange file
- Currency Table
- Budget file
- Order file
 - Current
 - Historical
- In-process file
- Vendor/Publisher Directory
- Authority Files
 - Personal Name
 - Corporate Name
 - Subject
 - Publishers
 - Place Names

2.4 Sub-systems of a Computer based Acquisition System and their Functions

A computer based acquisition system is expected to perform certain managerial and clerical functions. Systems are generally designed to manage regular orders, exchanges, receipts, non-receipts, invoice processing etc. The Acquisitions module must be fully integrated with all system modules and must support an unlimited number of material types/formats, funds, vendors, orders, claims and transactions, without added cost. The system must be dynamically updated in real time to maintain the currency of all records and statistics. The selected functions of a computer based acquisition systems are :

2.4.1 Pre-order searching

During pre-order searching, the Acquisitions module must allow staff to display records by searching any word in any field of a bibliographic record. The system must clearly display item status, including items with on-order status : items charged, on hold, in the library stacks, etc. and detect multiple orders during the order creation process.

2.4.2 Fund accounting

The Acquisitions module must allow :

- Temporary freezing of funds with override capability
- Freezing new order but permitting payment on outstanding orders
- Freezing both new orders and further payment
- Each fund to be subdivided (categorized) by up to five levels
- Each subdivision to be used to group accounts together as an online workstation query
- Each subdivision to be used to group accounts together to report on cumulated spending in the different categories
- Remaining allocations to be carried over from one fiscal year to the next, if desired
- Creation of new accounts at any time
- Input of an initial allocation when an account is first created
- Update of an account at any time
- Closing out an account at any time, so long as there are no outstanding encumbrances against it
- Multiple distribution methods, including :
 - By library-defined holdings code directly to requestor's address
 - indirectly to requestor with shipment to library for processing

For each fund, the Acquisitions module must maintain the following information, which must be available through online display, without the need to generate a report :

- The original budget allocation
- Amount of orders outstanding (encumbered)
- Amount of orders paid, the free balance
- Cash balance, the number of items on order
- Number received

- Number paid for
- Number of orders placed for the fund

2.4.3 Vendor Records

The Acquisitions module must support an unlimited number of vendor records, accessible by vendor name (complete), vendor name (truncated), and vendor code. The vendor record must include at least order and remittance addresses, library-supplied vendor claim period indicator, and performance statistics.

The vendor file must include performance statistics updated automatically and in real time, available online and through reports that includes, but is not limited to :

- Average receipt period in days
- Number of claims sent
- Number of copies cancelled
- Number of copies claimed
- Total amount ordered
- Amount encumbered
- Amount invoiced
- Amount paid
- Total number of orders
- Number of copies not received
- Number of copies paid
- Average order price
- Average price paid
- Supply times

Vendor records must support up to three distinct addresses per vendor, such as ordering, service, and marketing, and also provide notes/comment fields for Library staff.

2.4.4 Selection Records

The Acquisition module must support selection lists with access controlled by user login and password. It facilitates online approval management of resource selection procedures. It should automatically link purchase requests to items contained in a selection file, which can be hidden from public view and made accessible only to authorized users.

2.4.5 Order Records/Invoice Records

Selection lists that, when approved, can be used to create actual orders for Ordering/invoicing. The Acquisitions module must support an unlimited number of

order records. Order records must be searchable by bibliographic information, including item ID. Orders must be searched, browsed, or exactly matched by order ID, author, call number, periodical title, series, subject, title and title control number. Packing list and requisition numbers must also be searchable. Order line searching must also be supported. Acquisitions staff must be able to specify the fiscal cycle and acquisitions to search. The Acquisitions module must :

- Prevent assignment of duplicate order numbers, whether entered manually or assigned automatically
- Support electronic submission of orders
- Support electronic data interchange (EDI)
- Accommodating multi-institutional and multi-fund shared acquisitions
- Automatically determine how to handle a partial receipt of ordered items, based upon Library policies.
- Support an unlimited number of invoice records.
- Facilitate search and retrieve invoices by : invoice ID and optionally, a vendor ID, or a check number. Staff must be able to specify whether to display summary information, extended information, amounts, dates, and/or numbers associated with invoices.
- Link order records to the corresponding bibliographic record
- Report the current status of any and all titles ordered or received
- Use status information to signal for a variety of activities, such as produce purchase order, delete order, produce open order report, etc.
- Enable an authorized operator to access orders through the catalogue, vendor, account, and requestor
- Support MARC record order.

2.4.6 9XX Ordering

Acquisitions module must enable the Library to download MARC records (via the web) from materials vendors and use the imported records to create orders automatically within the Acquisitions module. Vendor must describe how it supports 9XX ordering for vendors and must specify which MARC tags can be used as the source for order data.

2.4.7 Electronic Data Interchange (EDI)

System must enable libraries to use advanced electronic data interchange (EDI) and X12 transaction set technology to manage their interactions with vendors and suppliers. A number of groups and organizations have proposed or implemented EDI messaging for acquisitions. Some of the benefits associated with implementing EDI-based messaging include :

- Reduction in manual labour and paper processing required
- More timely shipment and delivery of orders items
- Improvement in the availability of price information, particularly for serials
- More efficient invoicing and payment procedures, as keying of data is avoided
- More timely information can reduce the number of claims required, particularly for serials

While considerable gains have been made in developing the standardized formats necessary to support EDI-transactions for the purchase of books and serials, the implementation of EDI-based systems has been slower than anticipated. The following outlines some of the obstacles that may be slowing the adoption of EDI by libraries and the book and serial sector.

2.4.7.1 Organizational/Attitudinal barriers

- **Disparate User Community :** In order to purchase materials, libraries must interact with a very diverse group of partners that includes publishers, wholesalers, subscription agents and automated library system vendors. As each of these groups has different interests and objectives, bringing these groups together to work toward a common goal represents a significant organizational challenge. Without the co-operative efforts of this community, it is difficult to attain the support necessary to move toward EDI implementation.
- **Acquisitions is sometimes less a priority than other areas of library operations.** Acquisitions are often considered a ‘housekeeping’ function that may be less a priority than patron-oriented services such as reference and circulation. When decisions are made to fund automation projects, the acquisitions function has traditionally been given a low priority. This situation may improve as many of the other library processes are now supported by well-developed automated systems. This could help to shift the priority towards automating the acquisitions process.
- **Organizational Challenge for Libraries :** Implementing an EDI-based acquisitions function may require libraries to alter the work flows patterns in the acquisitions department, train staff in new procedures and acquire staff with systems training to maintain the new system. This requires considerable planning and coordination on the part of library management.
- **Lack of Education :** Education is required on the part of both librarians and members of the book and serial industry. They require a basic understanding of the existing EDI standards and the benefits which can be accrued through EDI implementation. While progress has been made in this area, these efforts must continue. Education is important in order for librarians to specify their requirements for EDI-based systems to the vendors of library automated systems.

2.4.7.2 Technical Barriers

- Reliance on other parties to develop supporting software : For the most part, libraries do not have the expertise to develop software to support their operations, so they are dependent on library software vendors to develop the supporting software. Software vendors, on the other hand are often reluctant to invest in product development for an uncertain market.
- Automated library systems tend to be based on proprietary architectures : The tight integration of existing library applications with proprietary architectures often makes it difficult to integrate these systems with EDI interfaces such as translation software without the assistance of the system vendor. The library is again dependent on the vendor's priorities and time scales in order to begin operations.
- Adding EDI-capabilities to existing systems : Integrated library systems tend to have a single database that underlies all operations and users are connected to various views of the database. It may be difficult to integrate the EDI data with existing database.

Other library operations involving communication with external systems include invoicing and payment activities. Libraries receive invoices and generally process them for payment by confirming the details, i.e that the items invoiced have actually been received. Often, however, the library does not actually make the payment. Instead a centralized finance function within the library or within a larger organization such as a university or a municipal government is actually responsible for allocating and transferring funds. The library is thus often involved in a complex set of transactions between invoicer, purchaser and payer. These transactions are normally asynchronous and thus lend themselves for consideration as candidates for EDI.

2.4.8 Claiming/Cancellation of Orders

The Acquisitions module must be able to automatically generate a claim/cancellation letter to the appropriate vendor regarding copies/volumes canceled and the reason. The Acquisitions module must be able to transmit claims and cancellations via letter, email, or X12.

If an order passes its library-defined 'date to claim', the system must automatically add a claim segment to the appropriate outstanding line items on the order. Each claim segment must include a claim reason, number of copies claimed, times claimed, part or volume claimed (for multi-part items), date mailed, vendor response, date of vendor response, and claim status (OPEN, RECEIVED, CANCELLED.) The system must update elements in the claim segment automatically.

2.4.9 Currency control

The Acquisitions module must support currency control (order records from vendors who require another nation's currency.) The currency control feature must

automatically calculate foreign currency purchases in local currency and allows the Library to maintain currency exchange tables. Exchanges table should be at Library's discretion. An unlimited number of currencies must be supported.

- Payment
- Claiming
- Cancellation of orders
- Routing
- Statistics and report compilation. Acquisitions statistics must be available in real-time as well as by statistical report(s).

2.5 Criteria for Selection of System

As a minimum, automated acquisition system should perform the same activities required in a manual acquisition system but with added requirement that the system perform these operations more efficiently. When selecting a system, librarian should consider the following points :

- The size of the system, that is, whether or not it will handle the required number of orders, funds etc.
- The cost of installation and maintenance
- Compatibility with other system
- Functions, that is, whether it offers the functions required
- Ease of operations
- Evaluation of its performance in live situation, if possible

2.6 Interface

An integrated system recognizes that each of the subsystems is closely related to other subsystems. Each of the function modules needs to provide interfaces to other subsystems, it means exchange of data and utilization of common resources (Acquisition and Serials Control share same vendor database, Acquisition and Circulation share same member database). The acquisition subsystems should have interfaces to the following areas :

- Cataloguing
- OPAC
- Circulation system

2.7 File Structure

Several factors have to be considered while designing a computer based acquisition system in an integrated environment. Certain decisions, like the following ones, have to be taken in advance :

- Files to be maintained
- Data elements in the records of each file
- Record format and media

The system flow chart below may have to be revised depending on the local procedure and/or mode of procurement etc. However, it is evident that information/data flows both ways, and different types of relations exist. For example in

- Suggestion-Approval Phase : User may suggest a book and the library committee decision may be conveyed to the user. The library may request the user to supply additional information. That is, data is flowing both ways.
- An user may request only one book [one to one relationship], s/he may request more than one book [one to many relationship]
- However, different users may request more than one book [many to one relationship].
- A vendor may supply books and serials [one-to-many relationship].

In an integrated system, same file can be used in more than one module. For example, Acquisition and Serials Control Systems will share the same vendor file. The user file will be shared by the Circulation Control System and Acquisition Control System. Hence, all aspects of possible applications and corresponding data requirements must be taken into account at time defining database structure. Computer based acquisition module will have same module specific files and will access certain common files. In an integrated system acquisition control system, the following files may be maintained and/or accessed :

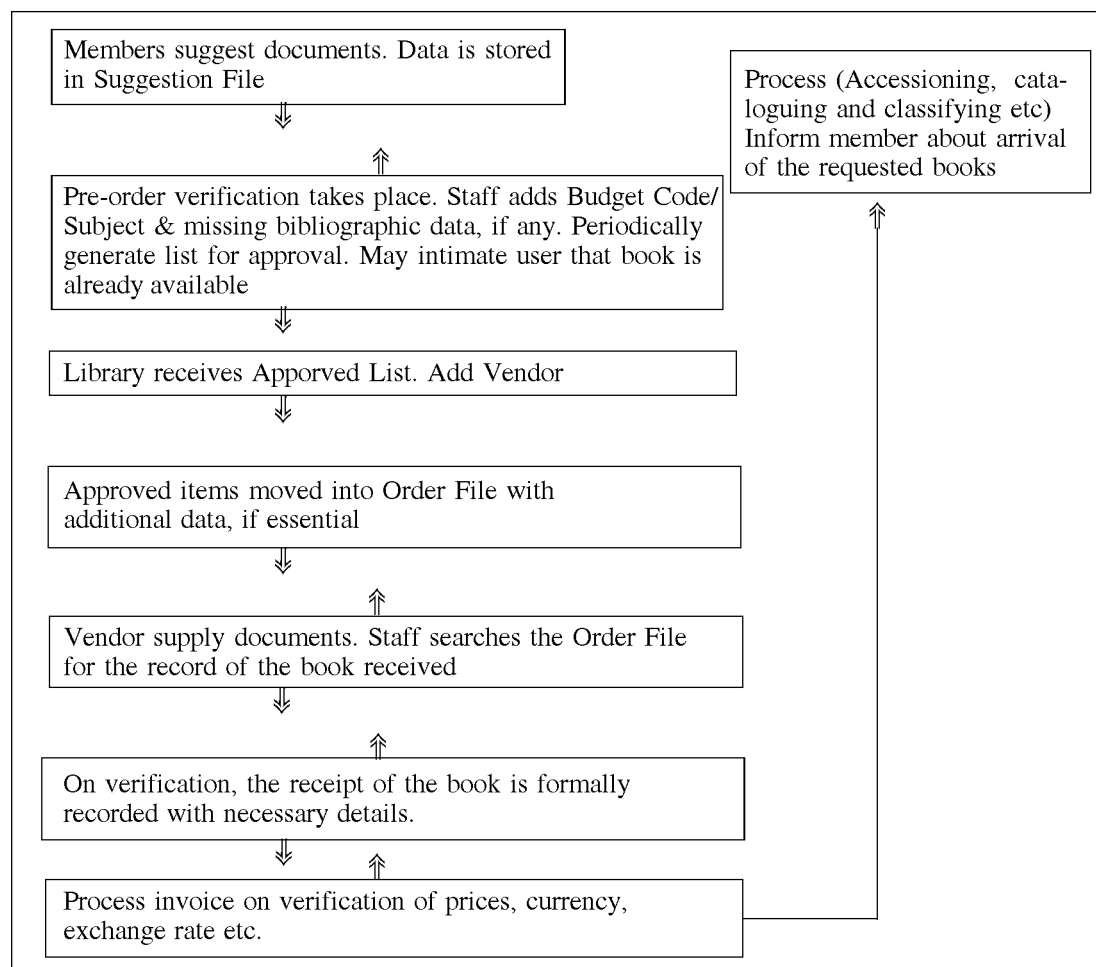
- Member
- Vendor
- Order
- Budget
- Currency
- Exchange Rate
- Bibliographic
- Suggestion
- Approval
- Invoice
- Subject

The following table is an illustrative example of required data files with individual data elements :

<p>Member File</p> <ul style="list-style-type: none"> ● Name ● Position ● Member-ID <ul style="list-style-type: none"> ● Address ● Contact Address 	<p>Exchange File</p> <ul style="list-style-type: none"> ● Currency ● Conversion rate ● Date 	<p>Order File</p> <ul style="list-style-type: none"> ● Order Number ● Order Date ● Bibliographic Item Code ● Vendor-ID ● Copies
<p>Vendor File</p> <ul style="list-style-type: none"> ● Name ● Vendor-ID ● Address ● Contact Details ● Contact Person 	<p>Invoice File</p> <ul style="list-style-type: none"> ● Invoice Number ● Invoice Date ● Invoice Currency ● Invoice Amount ● Vendor-ID ● Payment Details 	
<p>Budget File</p> <ul style="list-style-type: none"> ● Budget Title ● Budget Code ● FinancialYear ● Allocation ● Expenditure ● Balance 	<p>Subject File</p> <ul style="list-style-type: none"> ● Subject Terms ● Subject Code 	
<p>Currency</p> <ul style="list-style-type: none"> ● Name ● Currency-Code 	<p>Bibliographic File</p> <ul style="list-style-type: none"> ● Bibliographic Item Code ● Other elements are based on Standard Cataloguing Code like AACR2R etc] 	
<p>Suggestion File</p> <ul style="list-style-type: none"> ● Bibliographic Item Code ● Member ID ● Budget Code ● Suggestion ID ● Price ● Date ● Copies ● Recommendation 	<p>Approval File</p> <ul style="list-style-type: none"> ● Suggestion ID ● Approval ID ● Bibliographic Item Code ● Budget Code ● Order Type ● Vendor Code ● Date ● Price ● Copies ● Recommendation 	

The data that is collected and stored in the different files is used by routines of the software to perform various operations and processes : maintaining suggestion file, provision for online access to suggestion file for approval, placement orders and monitoring supply of ordered items etc.

The system flow chart of a typical acquisition system is shown in the following figure.



References and further Reading List

- 1 2005 Request for Proposal for a Client/Server Electronic Library System (<http://www.libraryhq.com/rfp.doc>). Visited last : 12/08/2005.
- 2 2005 Library acquisition functions (<http://www.cc.nctu.edu.tw/~claven/course/LibraryAutomation/acquisition.ppt>) Visited last : 25/09/2005

- 3 2004 Haravu (LJ). Library automation : design, principles and practice. New Delhi : Allied Publishers, 2004.
- 4 1997 Cohn (John M), Kelsey (Ann L) and Fiels (Keith Michael). Planning for automation : a how-to-do-it manual for librarians. 2nd ed. New York : Neal-Schuman Publishers, Inc., 1997.
- 5 1997 Meghabghab (Dania Bilal). Automating media centers and small libraries a microcomputer-based approach. Englewood : Libraries Unlimited, 1997.
- 6 1995 Electronic Data Interchange : An Overview of EDI Standards for Libraries (1993). International Federation of Library Associations and Institutions. (<http://www.ifla.org>). Visited last : 25/06/2005.
- 7 1989 Saffadv (William). Introduction to automation for librarians. 2nd ed. Chicago : ALA,1989
- 8 1984 Genaway (DC). Integrated online library systems : principles, planning and implementation. NY : GK. Hall. 1984.
- 9 1983 Rao (I.K. Ravichandra). Library automation (DRTC Refresher seminar 14, DRTC, Bangalore, 1983)

2.8 Exercise

1. Identify the common elements between manual and automated acquisition system.
2. Discuss the importance of different infrastructure for implementing computer based acquisition system.
3. Identify salient functions of an computer based acquisition system.

Unit 3 □ Computer Based Cataloguing and Character Encoding

Structure

- 3.0 Objectives**
- 3.1 Introduction**
- 3.2 Benefits**
- 3.3 Infrastructure**
- 3.4 Subsystems and their Functions**
- 3.5 Standardization/MARC Format**
- 3.6 File and Data structures**
- 3.7 Character Encoding**
 - 3.7.1 ASCII**
 - 3.7.2 ISCII**
 - 3.7.3 UNICODE**
 - 3.7.4 GIST Card**
- 3.8 Exercise**

3.0 Objectives

The objectives of the Unit are to :

- Examine functions of computer based cataloguing
- Review standardization of bibliographic and related standards
- Examine data structures of the catalogue database
- Identify subsystem and corresponding functions of each subsystem in integrated computer based library management system
- Study various character encoding systems

3.1 Introduction

The catalogue is the most valuable surrogate to information resources of a library. It can easily be developed as a byproduct of the computerized acquisition system. The primary purposes of the catalogue are to :

- Provide easy and effective access to the library's collection
- Enable rapid and effective retrieval of information as per request of the user.
- Maintain accuracy, completeness and consistency in description of library resources
- Enable cooperative collection development programmes.

3.2 Benefits

Cataloguing, classification and subject indexing are cost intensive processes in a library. These are done by highly trained professional. The rationale of computer based cataloguing are :

- Computer based cataloguing can reduce the cost of cataloguing. This is primarily possible through sharing of catalogue records in a consortium and importing cataloguing data from library utilities like OCLC and/or other cooperative networks (INFLIBNET/DELNET—strictly speaking such facilities are not yet available from any network in India)
- Enable integrating Internet resources in a library's catalogue.
- Provide remote access to the library catalogue.
- Reduce unnecessary duplication in intellectual cataloguing efforts.
- Generate spine labels and catalogue cards etc.

However, computer based cataloguing can provide the greatest benefits when the concerned library resolves to adhere to international standards-MARC 21,ISO 2709, Z39.50 and ISO ILL protocols etc.

3.3 Infrastructure

For a computer based cataloguing system, the necessary infrastructures are as follows :

- Access to external cataloguing resource databases
- Access to internal databases and files

3.3.1 Access to External Databases

The kind of external databses that are useful to have in computer based cataloguing system are sources of original catalogues and authority data. These databases may be on the Internet, CD-ROM, available in library utility like OCLC. Access to these databases may be free or may need membership.

Even if a library decide to do the original cataloguing, access to standard authority files of personal names, corporate names, conference names, serials, geographic name, language code etc. will be immensely helpful in maintaining consistancy in the library catalogue. The Library of Congress authory files are freely searchable on the Internet.

Access to machine-readable thesaurus or list of subject headings during the cataloguing process would help cataloguers. However, not all integrated library automation software, which are available in India, provide such tools.

3.3.2 Access to Internal Databases/Files

In absence of access to external authority files, a library must be able to create its own authority files for personal names, corporate names, conference names, subject terms etc. Such authority files should be created and updated as an integral part of the cataloguing process.

Cataloguing module must have access to the OPAC so that the cataloguers may verify important data to ensure consistency, accuracy and completeness of the catalogue.

3.4 Subsystems and their functions

A computer based cataloguing system can be broken down into several subsystems based on the functions. A typical computer based cataloguing system will have the following functional subsystems :

Subsystems	Description/Functions
Bibliographic Control/ Cataloguing system	<ul style="list-style-type: none">● It must allow records for any type of material in any format to be created, migrated, searched and displayed, modified, exported, and deleted.● Changes resulting from MARC format integration must be accommodated without reprogramming.● Cataloguing wizards should streamline the process of adding a brief title record, editing existing titles, duplicating an existing title, removing title, call number/volume or copies, creating and editing call number/volume records, adding or editing copies (includes global edits), offering authority control options, linking order line holdings to titles, capturing and editing records from Z39.50 sources using a Z39.50 copy cataloguing client.● When removing a title, call number, and/or item record, the Bibliographic Control/Cataloguing module must alert staff if a bill or hold is associated with the material. Staff must be able to immediately investigate the open transaction, without closing the removal process.

Subsystems	Description/Functions
	<ul style="list-style-type: none"> ● When using the Bound-with process to link bibliographic descriptions for items bound together, the Bibliographic Control/Cataloguing module must use parent and child call number records. A child call number with copies must not be bound with a parent call number. A child call number must be linked to only one parent, but staff must be able to link a parent call number to an unlimited number of child call numbers. ● In the transferring title, call number/volume, and/or copies process, authorized staff must be able to transfer all copies to an existing call number, transfer only selected copies, transfer all volumes, transfer only selected volumes, remove a title automatically after staff elect to transfer the last copy or volume to a new title, search and display bibliographic information without exiting the transfer process. Items on reserve or in transit must not be transferred. Inactive, available title-level holds linked to a call number must also block the transfer. The Bibliographic Control/Cataloguing module must alert staff to such exception conditions when staff attempt to transfer records. At Library staff's discretion, records may be hidden from public user display at the title level (to hide bibliographic data and all associated items), the call number level (to hide selected volumes and all associated copies), item level (to hide selected copies), or by assigning the item to a "shadowed" location. ● It must support MARC format error checking, including error checking (tags, indicators, and subfields) for all formats. ● A utility should verify uniform resource locator(s), or URLs, catalogued within MARC 856 bibliographic fields.

	<ul style="list-style-type: none"> ● It must support creation, editing and maintenance of Community Information records in the MARC Format for Community Information and in a locally developed format. ● It must include a Z39.50 Copy Cataloguing Client that can capture bibliographic records from any Z39.50 bibliographic resource.
Authority Control	<ul style="list-style-type: none"> ● Link all authority-controlled bibliographic headings with corresponding authority records through an ANSI-standard thesaurus. ● It should include a machine-proposed authority feature based upon the Standard for Machine-Proposed Authority Records (http://www.loc.gov/catdir/pcc/strawn.html) developed for the Programme for Cooperative Cataloguing, which must generate a new authority record with reasonable values in the fixed field and 001, 005, 040, 1XX, and 670 entries (plus the 022 and 642-646 entries for a series heading). These values must be automatically generated based on the information in the bibliographic record being validated. ● It must enable the system administrator to specify whether entry of bibliographic data which does not match an authority record must result in rejection of the input, or in a warning, in which case the system must display a browse list of possible authority headings from which the operator may choose by clicking the desired heading to replace the unauthorized heading in the bibliographic record, automatically flag the unauthorized entry for later display, review, and/or printing. ● It must support multiple authority files, including separate authority indexes for LC name and subject headings, NLM subjects, or other locally-defined indexes. ● It must enable the Library to define an unlimited number of authority formats or types, and to specify the bibliographic fields and subfields addressed by each authority record type through policy configuration.

	<ul style="list-style-type: none"> ● It must automatically generate appropriate <i>See</i> and <i>See Also</i> references from authority records for use in the online catalogue. ● At the Library's discretion, the Authority Control module must be configured either to display cross-references but not to verify headings, or to display cross-references and also to verify headings.
<p>Reports</p>	<ul style="list-style-type: none"> ● It must be fully-integrated with all other system modules, and provide a comprehensive suite of library-customizable report templates. ● It must enable an authorized operator to schedule production of report output at a specified date and time and on a regular periodic basis, such as daily, weekly, monthly, and at pre-specified times. ● It must enable an authorized operator to view completed reports on screen or to e-mail or print the report, at the operator's convenience. ● On any database reports involving materials, such as new accessions lists, shelf lists, high/low circulation lists, and bibliographics, the operator can select items for inclusion based on any combination of bibliographic information (using full Boolean word and phrase searching) and on any combination of control information, for example collection, current status, number of circulations, number of holds, classification, and accessions date. ● It must enable an authorized operator to specify the starting date-and-time and ending date-and-time that the report is to cover on reports involving historical data. ● The fully-integrated Reports module must use the same user interface as other modules. The Reports module must provide lists, counts, and statistical reports for each purchased module, provide row, column, and grand totals in applicable reports, provide reports for all record types within the proposed system, track statistical and management information by counting various staff processes to measure productivity, identifying items which are likely candidates for weeding, or tracking fund information for budgeting, perform housekeeping tasks by changing the status of groups of users, or

	<p>removing users or items in batches when necessary, allow authorized operator(s) to select, customize, name, save and schedule reports, allow staff to display and/or e-mail finished reports, employ an easy, point-and-click interface with dropdown menus for report criteria selection inventory.</p> <ul style="list-style-type: none"> ● The Circulation module must support marking items ‘used’ in-house for statistical reporting. If an item has a special status (charged, on hold, in transit), a glossary must appear next to the item for displaying more item information. ● The Circulation module must support the creation of a shelf list from inventory data in report output. Also, staff must be able to list inventory by item number in a report. ● The Circulation module must support the use of portable inventory scanners, provided the Vendor’s software is loaded to the Library’s portable scanners.
--	---

3.5 Standardization/MARC format

Bibliographic Standards Development is a key element in the Library’s cataloguing strategy. Standardization facilitates the exchange of bibliographic records between cataloguing agencies, thereby producing better, faster, cheaper catalogues. The following Table explains the types of standards involved in a computerized library environment.

Cataloguing Standards	Enabling Library to share catalogue records with other libraries, both in the country and overseas. AACR (Anglo-American Cataloguing Rules, 2nd edition) is a major international standard for the cataloguing of all types of materials collected by general libraries. The family of International Standard Bibliographic Descriptions (ISBD) specifies the requirements for description and identification of information resources.
Exchange Formats	Formats are used to transfer data in a structured form. Exchange formats were developed in parallel with the development of computers and other electronic storage devices to facilitate the transfer of bibliographic data between computer systems.

The MARC 21 formats are standards for the representation and communication of bibliographic and related information in machine-readable form. The Library of Congress in consultation with various user communities maintains the MARC 21 formats.

The primary purpose of UNIMARC is to facilitate the international exchange of data in machine-readable form between national bibliographic agencies. UNIMARC may also be used as a model for the development of new machine-readable bibliographic formats.

Name Authority Control Authority Control is the process of establishing and maintaining consistency in headings in a bibliographic file by means of an authority file.

Subject Access The use of a standardized system of subject headings allows compatible access by subject across different files and catalogues. In libraries document classification enables ordering of material in a helpful way on open shelves, browsing and retrieval of related items in catalogues, meaningful arrangement in subject indexes and bibliographics, analysis of the collections, and increasingly provides structural tools for mapping and organizing Web resources.

3.5.1 Needs for Standardization

In the library context, standards mean a set or code of rules established by national and international organizations for the purpose of bibliographic control, including those providing for the unique identification of bibliographic items, such as the International Standard Book Number and International Standard Serial Number; the uniform description of items, such as the International Standard Bibliographic Description; and the exchange of bibliographic records by means of a bibliographic exchange format, such as MARC (Machine Readable Cataloguing) format.

With the steadily increasing need for information, and the new technical capabilities for handling it, standardization becomes increasingly important-to ensure greater effectiveness of information services in all countries and the use of information and information systems across national, regional and institutional borders. Standardization, therefore, is aimed at achieving uniformity and it serves, in the end, as a quality control tool. Library catalogues which are produced under a certain standard cataloguing code can guarantee a certain level of information retrieval success, because that standard will ensure a certain level of quality so that the record

will be sufficiently complete and will be consistently arranged and hence that the content and the location of the record will be predictable.

A library could devise its own method of organizing the bibliographic information, but it would isolate the library, limit its options, and create much more work for itself. Using the MARC standard prevents duplication of work and allows libraries to better share bibliographic resources. Choosing to use MARC enables libraries to acquire cataloguing data that is predictable and reliable. If a library were to develop a “home-grown” system that did not use MARC records, it would not be taking advantage of an industry-wide standard whose primary purpose is to foster communication of information.

Using the MARC standard also enables libraries to make use of commercially available library automation systems to manage library operations. Many systems are available for libraries of all sizes and are designed to work with the MARC format. Systems are maintained and improved by the vendor so that libraries can benefit from the latest advances in computer technology. The MARC standard also allows libraries to replace one system with another with the assurance that their data will still be compatible.

3.5.2 Development of Exchange Format

MARC is the acronym for Machine-Readable Cataloguing. It provides the mechanism by which computers exchange, use, and interprets bibliographic information, and its data elements make up the foundation of most library catalogues used today. This general description, however, is rather misleading, as MARC is neither a kind of catalogue nor a method of cataloguing. In fact, MARC is a short and convenient term for assigning labels to each part of a catalogue record so that computers can handle it. While the MARC format was primarily designed to serve the needs of libraries, the wider information community as a convenient way of storing and exchanging bibliographic data has since embraced the concept.

3.5.2.1 Importance of MARC

The information from a catalogue card cannot simply be typed into a computer to produce an automated catalogue. The computer needs a means of interpreting the information found on a cataloguing record. The MARC record contains a guide to its data, or little “signposts,” before each piece of bibliographic information.

The place provided for each of these pieces of bibliographic information (author, title, call number, etc.) is called a “field.” The records in simpler computer files sometimes have a fixed number of fields, and each field contains a fixed number of characters. However, to allow proper cataloguing of books and other library items, the best file structure allows for records with an unlimited number of fields and unlimited

field lengths. This flexibility is necessary because not all titles are the same length. Some books are part of a series, requiring a field for that information, while others have no series statement. And audiovisual items have much longer physical descriptions than do most books.

The computer cannot expect a certain type of information to begin and end at the same position in every bibliographic record. The statement of responsibility will not always begin with the 45th character of the record and end at the 107th position, for example. Therefore each MARC record contains a “table of contents” to the record, according to a predefined standard.

If a bibliographic record has been marked correctly and saved in a computer data file, computer programmes can then be written to punctuate and format the information correctly for printing a set of catalogue cards, or for displaying the information on a computer screen. Programmes can be written to search for and retrieve certain types of information within specific fields, and also to display lists of items meeting the search criteria.

3.5.2.2 Development of MARC21

The original MARC format was developed at the Library of Congress in 1965-66 leading to a pilot project, known as MARC-I, which had the aim of investigating the feasibility of producing catalogue data in machine-readable form. Similar work was in progress in the United Kingdom where the Council of the British National Bibliography had set up the BNB MARC Project with the objective of examining the use of machine-readable data in producing the printed *British National Bibliography* (*BNB*). These parallel developments led to Anglo-American cooperation on the MARC-II project, which was initiated in 1968.

Despite cooperation there emerged several versions, e.g. UKMARC, INTERMARC and USMARC, whose paths diverged owing to different national cataloguing practices and requirements. Since the early 1970s an extended family of more than 20 MARC formats has grown up. Differences in data content means that editing is required before records can be exchanged.

One solution to the problem of incompatibility was to create an international MARC format (UNIMARC), which would accept records created in any MARC format. So records in one MARC format could be converted into UNIMARC and then be converted into another MARC format. The intention was that each national agency would need to write only two programmes—one to convert into UNIMARC and one to convert from UNIMARC—instead of one programme for each other MARC format, e.g. INTERMARC to UKMARC, USMARC to UKMARC etc.

MARC-II established certain principles, which have been followed consistently over the years. MARC-II was to prove instrumental in defining the concept of MARC as a communication format. In general terms, the MARC communication format is intended to be :

- ◆ Hospitable to all kinds of library materials
- ◆ Sufficiently flexible for a variety of applications in addition to catalogue production
- ◆ Usable in a range of automated systems

MARC-I dealt mostly with books. However, MARC II was planned to cover all types of materials including books and monographs. During 1970-1975, documentation was issued for other materials i.e. in 1972 films; in 1973 serials, maps and French books; in 1975 records for German, Spanish, and Portuguese material were covered.

MARC became USMARC in the 1980s and MARC21 in the late 1990s. MARC21 is not a new format. After having discussions and making minor changes to both formats that accommodated USMARC and CAN/MARC users' specific needs, the USMARC and CAN/MARC (Canadian MARC) formats were "harmonized" into MARC21 in 1997. The Network Development and MARC Standards Office at the Library of Congress and the Standards and the Support Office at the National Library of Canada maintain the MARC21 formats. Input for development is provided by MARC21 users from around the world, including libraries, library networks and utilities, and library system vendors.

The *MARC21 Format for Bibliographic Data* is designed to be a carrier for bibliographic information, such as titles, names, subjects, notes, publication information, and physical descriptions of items. The *MARC21 Format for Bibliographic Data* contains data elements for the following types of material :

- ◆ **Books-** textual items, monographic in nature, that are in bound book, electronic, or microform.
- ◆ **Continuing resources-** textual items with a recurring pattern of publication, e.g., periodicals, newspapers, and yearbooks.
- ◆ **Computer files-** Computer software, numeric data, computer-oriented multimedia, online systems or services. Other types of electronic resources are coded for their most significant aspect, such as textual ("books" or "serials"), cartographic ("maps"), etc.
- ◆ **Maps-** all types of cartographic materials, including sheet maps and globes in printed, manuscript, electronic, and microform.
- ◆ **Music-** printed and manuscript notated music
- ◆ **Sound recordings-** nonmusical sound recordings and musical sound recordings.

- ♦ **Visual materials-** images and objects, e.g., projected media, motion pictures, two-dimensional graphics, three-dimensional artifacts, naturally occurring objects.
- ♦ **Mixed materials-** primarily archival and manuscript collections of a mixture of forms of material.

MARC21 is an implementation of the American national standard, *Information Interchange Format* (ANSI Z39.2) and its international counterpart, *Format for Information Exchange* (ISO 2709). These standards specify the requirements for a generalized interchange format that will accommodate data describing all forms of materials susceptible to bibliographic description, as well as related information. The five MARC21 communications formats are widely used standards for the representation and exchange of bibliographic, authority, holdings, classification, and community information data in machine readable form. The five MARC21 communication formats are :

- ♦ **Bibliographic Data :** It contains format specifications for encoding data elements needed to describe, retrieve, and control various forms of bibliographic material. The MARC21 Format for Bibliographic Data is an integrated format defined for books, serials, computer files, maps, music, visual materials, and mixed material. With the full integration of the previously discrete bibliographic formats, consistent definition and usage are maintained for different forms of material.
- ♦ **Authority data :** It contains format specifications for encoding data elements that identify or control the content and content designation of those portions of a bibliographic record that may be subject to authority control.
- ♦ **Holdings Data :** It contains format specifications for encoding data elements pertinent to holdings and location data for all forms of material.
- ♦ **Classification Data :** It contains format specifications for encoding data elements related to classification numbers and the captions associated with them. Classification records are used for the maintenance and development of classification schemes.
- ♦ **Community Information :** It provides format specifications for records containing information about events, programmes, services, etc. So that this information can be integrated into the same public access catalogues as data in other record types.

The standards present a generalized structure for records, but do not specify the content of the record and do not, in general, assign meaning to tags, indicators, or data element identifiers. Specification of these elements is provided particular implementations of the standards.

3.5.2.3 Organization of MARC21 Record

A MARC record is composed of the following three elements :

- **Record structure** : is an implementation of ISO 2709, *Information and Documentation-Format for Information Exchange*.
- **Content Designation/Tagging** : is the set of tags and codes that identifies and further characterize the data elements within a record and support the manipulation of the data content.
- **Data content of the record.** The **content** of the data elements that comprise a MARC record is usually defined by standards outside the formats, such as cataloguing rules, classification schemes, subject thesauri, code lists, or other conventions used by the organization that creates a record. The content of certain coded data elements (e.g., the Leader, and field 008) is defined in the MARC formats themselves.

3.5.2.3.1 Record Structure

MARC is a specific implementation of ISO 2709, an international standard that specifies the structure of records containing bibliographic data. It specifies that every bibliographic record prepared for exchange conforming to the standard must consist of a :

- ◆ RECORD LABEL : consisting of 24 characters
- ◆ DIRECTORY consisting of a 3-digit tag of each data field, along with its length and its starting character position relative to the first data field,
- ◆ DATA FIELDS of variable length, each separated by a field separator,

The record layout is as follows :

Record Label	Directory	Data Fields	Record Terminator
--------------	-----------	-------------	-------------------

ISO 2709 further specifies that the data in fields may optionally be preceded by indicators and subdivided into subfields.

Record Label

It is the first field in all MARC records. ISO 2709 prescribes that each record start with a 24-character Record Label. This contains data relating to the structure of the record, which are defined within the standard ISO 2709, and several data elements that are defined for this particular implementation of ISO 2709. These implementation-defined data elements relate to

- The type of record,
- Its bibliographic level and position in a hierarchy of levels,
- The degree of completeness of the record.

The data elements in the Record Label are required primarily to process the record and are intended only indirectly for use in identifying the bibliographic item itself. A key data element in the Record Leader is the type of item being described in the record. It identifies the following item types.

Language (textual) material	Nonmusical sound recording
Manuscript (textual) language	Musical sound recording
Material	
Computer file	Projected medium
Cartographic material	Two-dimensional non projectable graphic
Manuscript cartographic material	Three-dimensional artifact or natural objects
Notated music	Kit
Manuscript music	Mixed material

Directory

A directory entry in MARC21 is made up of a tag, length-of-field, and field starting position. The directory begins in character position 24 of the record and ends with a field terminator. It is of variable length and consists of a series of fixed fields, referred to as “entries.” One entry is associated with each variable field (control or data) present in the record. It contains a series of entries that contain the tag, length, and starting location of each variable field within a record. Each entry is 12 character positions in length. The Directory entry for the Record Control Number (001) appears first. Subsequent entries for variable data fields follow, arranged in ascending order according to the first character of the tag. The stored sequence of the variable data fields in a record is not necessarily the same as to the order of the corresponding Directory entries. The Directory ends with a field terminator character.

Each directory entry is 12 characters in length; and it is represented schematically below. The numbers indicate the character positions occupied by the parts of the entry.

Structure of a directory Entry in MARC 21 Records

TAG	LENGTH_OF_FIELD	STARTING_CHARACTER_POSITION
00-02	03-06	07-11

Tag (character positions 00-02), consists of three ASCII numeric characters or ASCII alphabetic characters (uppercase or lowercase, but not both) used to identify or label an associated variable field. The MARC 21 formats have used only numeric tags. The tag is stored only in the directory entry for the field; it does not appear in the variable field itself.

Length of field (character positions 03-06), contains four ASCII numeric characters which give the length, expressed as a decimal number, or the variable field

to which the entry corresponds. This length includes the indicators, subfield codes, data and field terminator associated with the field. A field length number, of fewer than four digits, is right justified and unused positions contain zeroes (zero fill).

Starting character position (Character positions 07-11), contains five ASCII numeric characters which give the starting character position, expressed as a decimal number, or the variable field to which the entry corresponds relative to the base address of data of the record. A starting character position, of fewer than five digits, is right justified and unused positions contain zeroes (zero fill).

Order of entries, Directory entries for control fields precede entries for data fields. Entries for control fields are sequenced by tag in increasing numerical order. Entries for data fields are arranged in ascending order according to the first character of the tag, with numeric characters preceding alphabetic characters.

Variable-Fields

The variable fields follow the leader and the directory in the record and consist of control fields and data fields. Control fields precede data fields in the record and are arranged in the same sequence as the corresponding entries in the directory. The sequence in which data fields are stored in the record is not necessarily the same as the order of the corresponding directory entries.

Control Fields in MARC 21 formats are assigned tags beginning with two zeroes. They are comprised of data and a field terminator; they do not contain indicators or subfield codes. The control number field is assigned tag 001 and contains the control number of the record. Each record contains only one control number field (with tag 001), which is to be located at the base address of data.

Data fields in MARC21 formats are assigned tags beginning with ASCII numeric characters other than two zeroes. The tag is stored in the directory entry for the field, not in the field itself. The data in a MARC 21 record is organized into fields. Such fields contain indicators and subfield codes, as well as data and a field terminator. There are no restrictions on the number, length, or content of data fields other than those already stated or implied, e.g., those resulting from the limitation of total record length. The variable fields follow the leader and the directory in the record and consist of control fields and data fields. The structure of a data field is shown schematically below.

Structure of a Variable Data Field in MARC 21 Records

INDICATOR_1	INDICATOR_2	DELIMITER	DATA_ELEMENT_IDENTIFIER_1
DATA_ELEMENT_1	...	DELIMITER	DATA_ELEMENT_IDENTIFIER_n
DATA_ELEMENT_n		FT	

Indicators are the first two characters in every variable data field, preceding any subfield code (delimiter plus data element identifier), which may be present. Each indicator is one character and every data field in the record includes two indicators, even if values have not been defined for the indicators in a particular field. Indicators supply additional information about the field, and are defined individually for each field. Indicator values are interpreted independently; meaning is not ascribed to the two indicators taken together. Indicators may be any ASCII lowercase alphabetic, numeric, or blank. A blank is used in an undefined indicator position, and may also have a defined meaning in a defined indicator position. The numeric character 9 is reserved for local definition as an indicator.

Subfield codes identify the individual data elements within the field, and precede the data elements they identify. Each data field contains at least one subfield code. The subfield code consists of a delimiter followed by a data element identifier. Data element identifiers defined in MARC21 may be any ASCII lowercase alphabetic or numeric character. In general, numeric identifiers are defined for data used to process the field, or coded data needed to interpret the field. Alphabetic identifiers are defined for the separate elements which constitute the data content of the field. The character 9 and the following ASCII graphic symbols are reserved for local definition as data element identifiers :

! " # \$ % & ' () * +, - . / : ; < = > ? { } _ ^ ` ~ [] \

A data field may contain more than one data element, depending upon the definition of the field. The last character in a data field is the **field terminator**, which follows the last data element in the field.

Sub-field codes are defined independently for each field; however, parallel meanings are preserved where possible. Sub-field codes are defined for purposes of identification. The order of sub-fields is generally specified by external standards for the data content, such as the cataloguing rules, not by this format.

Field and Sub-field Repeatability

Theoretically, all fields and subfields may be repeated. The nature of the data, however, often precludes repetition. For example, a record may contain only one 1XX field; a bibliographic field 100 may contain only one subfield \$a (Personal name), but may contain more than one subfield \$c (Titles and other words associated with a name). Field and subfield repeatability/non-repeatability is indicated by (R) or (NR) following each field and subfield name.

Bibliographic Format Blocks group variable field tags in blocks according to the first character of the tag, which identifies the function of the data within a record, e.g., main entry, added entry, subject entry. The remainder of the tag identifies the type of information in the field. The type of information in the field, e.g., personal name, corporate name, or title, is identified by the remainder of the tag. The meaning of these blocks depends upon the type of record. The bibliographic format blocks are :

Bibliographic format blocks :	Authority format blocks :
0XX = Control information, numbers, codes	0XX = Control information, numbers, codes
1XX = Main entry	1XX = Heading
2XX = Titles, edition, imprint	2XX = Complex see references
3XX = Physical description, etc.	3XX = Complex see also references
4XX = Series statements	4XX = See from tracings
5XX = Notes	5XX = See also from tracings
6XX = Subject access fields	6XX = Reference notes, treatment, notes, etc.
7XX = Name, etc. added entries or series; linking	7XX = Heading linking entries
8XX = Series added entries; holdings and locations	8XX = Not defined
9XX = Reserved for local implementation	9XX = Reserved for local implementation

Holdings format blocks :	Community information format blocks :
0XX = Control information, numbers, codes	0XX = Control information, numbers, codes
1XX = Not defined	1XX = Primary names
2XX = Not defined	2XX = Titles, addresses
3XX = Not defined	3XX = Physical information, etc.

4XX = Not defined	4XX = Series information
5XX = Notes	5XX = Notes
6XX = Not defined	6XX = Subject access fields
7XX = Not defined	7XX = Added entries other than subject
8XX = Holding and location data notes	8XX = Miscellaneous
9XX = Reserved for local implementation	9XX = Reserved for local implementation

Classification format blocks :	
0XX = Control information, numbers, codes	
1XX = Classification numbers and terms	
2XX = Complex see references	
3XX = Complex see also references	
4XX = Invalid number tracings	
5XX = Valid number tracings	
6XX = Notes	
7XX = Index terms and number building fields	
8XX = Miscellaneous	
9XX = Reserved for local implementation	

Certain blocks in the MARC 21 formats contain data which may be subject to authority control (1XX, 4XX, 6XX, 7XX, 8XX for bibliographic records; 1XX, 4XX, 5XX, 7XX for authority records, etc.). In these blocks, certain parallels of content designation are preserved. The following meanings are generally given to the final two characters of the tag :

X00 = Personal names
X10 = Corporate names
X11 = Meeting names
X30 = Uniform titles
X40 = Bibliographic titles
X50 = Topical terms
X51 = Geographic names

3.5.2.3.2 Content Designation

The goal of content designation is to identify and characterize the data elements, which comprise a MARC record with sufficient precision to support manipulation of the data for a variety of functions. MARC content designation is designed to support functions that include :

- Display-the formatting of data for screen display, for printing on 3×5 cards or in book catalogues, for production of COM catalogues, or for other visual presentation of the data.
- Information retrieval-the identification, categorization, and retrieval of any identifiable data element in a record.
- Some fields serve multiple functions. For example, field 245 (Title Statement) serves both as the bibliographic transcription of the title and the statement of responsibility and as an access point for the title.
- The MARC 21 formats provide for display constants. A display constant is a term, phrase, and/or spacing or punctuation convention that may be system generated under prescribed circumstances to make a visual presentation of data in a record more meaningful to a user. Such display constants are not carried in the data. but may be supplied for display by the processing system. For example, subfield \$x in Series Statement field 490 (and in some other fields) implies the display constant ISSN; also, the combination of tag 780 (Preceding Entry) and second indicator value 2 implies the display constant *Supersedes*.
- The MARC 21 formats support the sorting of data only to a limited extent. In general, sorting must be accomplished through the application of external algorithms to the data.

3.5.2.4 Structure of MARC Record

The block of data below is what the programmer sees when he looks at the contents of a MARC tape or disk. The tags do not appear before the fields, but in a directory

to the data tells which tags (underline added here) should be used and at which position in the character string each field starts (where field is stored).

LEADER

01041 cam 2200265 a 4500

001002000000000300040002000500170002400800410004101000240008202000
02500106200044001310400180017505000240019308200180021710000320
02352450087002672460036003542500012003902600037004023002900439
500004200468520022000510650003300730650001200763^###89048230#/AC/
r91^DLC^19911106082

810.9^891101s1990####maua###j#####000#0#eng##^###\$a###89048230#/AC/r91^###\$a031
6107514 : \$c\$12.95^###\$a0316107506 pbk.) : \$c\$5.95 (\$6.95 Can.)^###\$aDLC\$cDLC\$d
DLC^00\$aGV943.25\$b.B74
1990^00\$a796.334/2\$220^10\$aBrenner,
Richard J., \$d 1941-^10\$aMake the team.\$pSoccer : \$ba heads up guide to super
soccer!/\$cRichard J. Brenner.^30\$a Heads up guide to super soccer.^###\$a1st
ed.^###\$a Boston : \$bLittle, Brown, \$cc1900.^###\$a127 p. : \$bill. ;\$c19cm.^###\$a"A
Sports illustrated for kids book."^###\$alnstructions for improving soccer
skills. Discusses dribbling, heading, playmaking, defense, conditioning,
Mental attitude, how to handle problems with coaches, parents, and other players, and
the history of soccer.^#0\$a Soccer\$vJuvenile literature.^#1\$aSoccer.^

Cracking the code, or, interpreting the directory

Usually, only the computer programmer and the computer come into contact with the record in MARC 21 communications format. The first 24 positions are the leader. In this example the leader fills approximately 1/3 of the first line and ends with "4500." Immediately following the leader, the directory begins. Tags have been underlined in this example. Each individual tag directory is 12 characters long. The first tag is 001. Following each tag, the next four positions show the length of the field. The data in the 001 field (control number) in this record is 20 characters long. The next 5 positions tell the starting point for this field within the data string that follows the directory. The 001 field begins at the 00000 position (the first position is 0). The next tag is 005. It is 17 characters long (the length of the previous position -20-added to its starting spot-00000-equals 20). The next tag is 005. It is 17 characters long and begins at the 24th spot (4 + 20 = 24). This directory tells us :

Tag	Length	Starts at	Tag	Length	Starts at
001	0020	00000	100	0032	00235
003	0004	00020	245	0087	00267
005	0017	00024	246	0036	00354
008	0041	00041	250	0012	00390
010	0024	00082	260	0037	00402
020	0025	00106	300	0029	00439
020	0044	00131	500	0042	00468
040	0018	00175	520	0220	00510
050	0024	00193	650	0033	00730
082	0018	00217	650	0012	00763

Field terminators (displayed as a ^ in this example) mark the end of the directory and the end of each field that follows. Notice that the sum of the 2nd and 3rd column in any row equals the number in the 3rd column in the next row. The starting point of one field plus its length equals the starting position of the next field. This can be verified by counting the character positions within the data, remember that spaces count, as do the field terminators (^). (Two character positions are always reserved for indicators at the beginning of a field.) A record terminator (displayed as a \ in this example) ends each bibliographic record.

3.6 File and Data structures

Several integrated library management software vendors are moving toward using RDBMS and ODBC platforms in their newer versions. It is evident that to present a MARC record using the relational model, several tables and normalizations are required. There is no single standard way in which the MARC record may be defined in RDBMS. Each integrated library automation software decides for itself how it will structure its catalog database. structure of tables depend not only on functional modules of the integrated library management software but also on structure and requirements of the adopted bibliographic standard-MARC21, UNIMARC, and CCF etc.

Before attempting to define a set of normalized tables to represent contents of MARC record in RDBMS, it would be useful to identify desirable functionalities of the cataloguing module :

- It should accommodate MARC tags, indicators and subfield etc.
- It should also facilitate template base cataloguing, if required.

- Catalogue database should be searchable not only by conventional access points (author, title, subject headings etc.) but also by other characteristics- language, location, year of publication, and document type etc.

Since bibliographical data elements is depend on the adopted bibliographic MARC standard, no attempt is made to enumerate the bibliographic data elements. However, the following tables may be necessary to maintain in an integrated library management system :

- Fixed Field Table : It holds data found in the Leader and Control Field, which are important for data entry, import of data and searching
- Field Table : It contains information about MARC tags (fields) and corresponding indicators. Acutal data resides in the different subfields of each field.
- Subfield Table : Contains subfields of each MARC field. It actually contains data.
- Bibliographic Table : It provides the link between the fixed-field records and the Field and Subfield Tables, and Holding Table.
- Keyword Table : It will contain keywords that are automatically extracted from searchable fields-author, title, series, and subject headings etc.
- Keyword-Bibliograph Link Table : Since many-to-many relation exist between keywords and bibliographic records, this link table is necessary.

Proper linking must be established between these tables for a given bibliographic items. However, alternative relational database structure is also possible.

3.7 Character Encoding

In computers data is internally presented as octets, as a rule. An *octet* is a small unit of data with a numerical value between 0 and 255, inclusively. Octets are often called *bytes*. Internally, octets consist of eight bits. Different conventions can be established as regards to how an octet or a sequence of octets presents some data.

In the simplest case, which is still widely used, one octet corresponds to one character according to some mapping table (encoding). Naturally, this allows at most 256 different characters being represented. There are several different encodings, such as the well-known ASCII encoding and the ISO Latin family of encodings.

Character repertoire

- A set of distinct characters. No specific internal presentation in computers or data transfer is assumed. The repertoire per se does not even define an ordering

for the characters; ordering for sorting and other purposes is to be specified separately. A character repertoire is usually defined by specifying names of characters and a sample (or reference) presentation of characters in visible form. Notice that a character repertoire may contain characters which *look* the same in some presentations but are regarded as logically distinct, such as Latin uppercase A, Cyrillic uppercase A, and Greek uppercase alpha. A character *repertoire* specifies a collection of characters, such as “a”, “!”, and “ä”.

Character code

A character code defines numeric codes for characters in a repertoire. For example, in the ISO 10646 character code the numeric codes for “a”, “!”, “ä”, and “‰” (per mile sign) are 97, 33, 228, and 8240.

That is, it assigns a unique numerical code, a *code position*, to each character in the repertoire. As synonyms for “code position”, the following terms are also in use : *code number*, *code value*, *code element*, *code point*, *code set value* and just *code*.

The set of nonnegative integers corresponding to characters need not consist of consecutive numbers; in fact, most character codes have “holes”, such as code positions reserved for control functions or for eventual future use to be defined later.

Character encoding

A method (algorithm) for presenting characters in digital form by mapping sequences of code numbers of characters into sequences of octets. In the simplest case, each character is mapped to an integer in the range 0-255 according to a character code and these are used as such as octets. Naturally, this only works for character repertoires with at most 256 characters. For larger sets, more complicated encodings are needed. A character encoding could, in principle, be viewed purely as a method of mapping a sequence of integers to a sequence of octets.

Character set

The phrase *character set* is used in a variety of meanings. It might denote just a character repertoire but it may also refer to a character code, and quite often a particular character encoding is implied too. Unfortunately the word *charset* is used to refer to an encoding, causing much confusion. It is even the official term to be used in several contexts by Internet protocols, in MIME headers.

Quite often the choice of a character repertoire, code, or encoding is presented as the choice of a *language*. For example, Web browsers typically confuse things quite a lot in this area. A pulldown menu in a programme might be labeled “Languages”, yet consist of character encoding choices (only). A language setting is quite distinct

from character issues, although naturally each language has its own requirements on character repertoire. Even more seriously, programmes and their documentation very often confuse above-mentioned issues with the selection of a font.

3.7.1 ASCII

The name *ASCII* stands for “American Standard Code for Information Interchange”, denotes an old character repertoire, code and encoding. Most character codes currently in use contain ASCII as their subset in some sense. ASCII has been used and is used so widely that often the word *ASCII* refers to “text” or “plain text” in general, even if the character code is something else! The words “ASCII file” quite often mean any text file as opposite to a binary file.

The definition of ASCII also specifies a set of control codes (“control characters”) such as linefeed (LF) and escape (ESC). But the *character repertoire* proper, consisting of the *printable* characters of ASCII, is the following (where the first item is the blank, or space, character) :

```
! " # $ % & ' ( ) * + , - . /
0 1 2 3 4 5 6 7 8 9 : ; | < = > ?
@ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z [ \ ] ^ _
` a b c d e f g h i j k l m n o
p q r s t u v w x y z { } ~
```

The *character encoding specified* by the ASCII standard is very simple, and the most obvious one for any character code where the code numbers do not exceed 255 : each code number is presented as an octet with the same value. Octets 128 -255 are not used in ASCII. (This allows programmes to use the first, most significant bit of an octet as a parity bit, for example.)

8-bits are sufficient for presenting the code numbers and in practice the encoding (at least the normal encoding) is the obvious one where each code position (thereby, each character) is presented as one octet (byte). This means that there are 256 code positions, but several positions are reserved for control codes or left unused (unassigned, undefined).

Limitations of 8-bit Character System

Each character is 8 bits long and can store up to 256 values, of which some are control characters, some are letters, and some are punctuation, and so on, Of those 256 characters, first 128 are standardized by ANSI, meaning they are always exactly the same. The first 128 characters contain control characters (0-31), Indo-Arabic

numbers, common punctuation, English alphabet in lower and upper cases, and some other stuff.

The second 128 characters are not standardized and can mean different things depending on the *codepage*. For Standard English codepage, they have some Greek letters, more punctuation like ellipsis, character-mode graphics symbols (Vertical and horizontal lines, etc), and other things. Non-English codepages store the English symbols in the first 128 characters (as specified by ANSI) and store their native alphabets in the second 128 characters. This leads to two problems :

- The Greek letters and character-mode graphics are thrown away to make room for native characters. As a result, text that uses these thrown out characters will not look correctly if for example a Cyrillic codepage is active.
- Some alphabets, such as Asian ones, have more letters than 128 slots can fit (remember you need separate slots for upper and lower case letters, too). For those layouts, it is physically impossible to store all letters in 256 bytes that a char set provides.

That is, a single 8-bit value is not enough for all languages. One solution is to use something called multi-byte character sets, where a character can take either one or two bytes.

Character Encoding Standard for Indian Scripts

The 18 Scheduled languages in India are Hindi, Bengali, Telugu, Marathi, Tamil, Urdu, Gujarati, Kannada, Malayalam, Oriya, Punjabi, Assamese, Kashmiri, Sindhi, Konkani, Nepali, Manipuri and Sanskrit.

Urdu, Kashmiri and Sindhi are primarily written in Perso-Arabic scripts. As these scripts have a different alphabet, a different character encoding standard is envisaged for them. The Brahmi based scripts can be divided into :

- The **Northern scripts** are Devanagari, Gujarati, Punjabi, Assamese, Bengali and Oriya.
- The **Southern scripts** are Telugu, Tamil, Malayalam and Kannada. The official language of India, Hindi is written in the Devanagari script as are Marathi, Konkani, Nepali and Sanskrit. Manipuri is written extensively in the Bengali script. It is also written in the Meitei script.

Since the 1970s, different committees of the Department of Official Languages and the Department of Electronics (DOE) have been evolving different character encodings and keyboard overlays which would cater to all the Indian scripts. In July 1983, the DOE announced the ISCII-83 code which complied with the ISO 8-bit

recommendations (“Report of the sub-committee on Standardization of Indian Scripts and their Codes for Information Processing”, DOE, July 1983). This also had the recommendation on a common Phonographic based keyboard layout.

A keyboard standard for Indian scripts was brought out by the DOE in 1986 (Report of the committee for “Standardization of Keyboard Layout for Indian Script Based Computers” in Electronics-Information & Planning, Vol. 14, No. 1, October 1986).

There was a revision of the ISCII code by the DOE in 1988 (Report of the sub committee on “Standardization of Indian Script Codes for Information interchange”, DOE, August 1988). In November 1991 this was incorporated in the Indian Standard IS 13194 :1991 and remains the prevalent standard today.

3.7.2 Indian Script Codes for Information Interchange (ISCII)

It was imperative to arrive at a unified character encoding scheme and a common keyboard layout for the Indian scripts in order to facilitate implementations on computers. This is made possible by their common origin from the ancient Brahmi script and by the phonetic nature of the alphabet. The advantages of this philosophy are many. Any software which allows ISCII codes can be used in any Indian script, enhancing its commercial viability. In part, this is also the rationale behind the Unicode standard. Immediate transliteration between different Indian scripts becomes possible. Simultaneous availability of multiple Indian languages in the computer medium would accelerate the process of development and communication.

The ISCII code retains the standard ASCII code while utilizing the upper ASCII codes for Indian scripts. This makes it feasible to use Indian scripts along with English computers and software in an 8-bit environment.

The ISCII code table is a superset of all the characters required in the 10 Brahmbased Indian scripts. The ISCII code contains only the basic alphabet required by the Indian scripts. All the composite characters are formed through combinations of these basic characters. The alphabet in each script may vary but they all share a common phonetic structure. The differences between scripts are primarily in their written forms. ISCII encoding is completely delinked from the physical glyphs used for display making it possible for a script to be displayed in a variety of styles depending on the conjunct (ligature) repertoire available in the glyph set.

Punctuation

All punctuation marks used in Indian scripts are borrowed from English, except for the full-stop, instead of which a Viram (*) is used in the Northern scripts. The Viram is, however, being increasingly substituted by a full-stop. A double Viram (**) is also used in Sanskrit texts for indicating a verse ending.

Numerals

In all the Indian scripts the international numerals are being used increasingly. From the software viewpoint, usage of the same numerals as given in the ASCII set allows proper handling of numerals by existing software. For display rendition purposes however, it may be sometimes desirable to have separate Indian script numerals which are given in the ISCII table.

Phonetic Sequence

The ISCII characters, within a word, are kept in the same order as they would get pronounced. The display order may be different from the phonetic order. Having a spelling according to the phonetic order allows a name to be typed in the same way, regardless of the script it has to be displayed in.

By using only the basic characters in ISCII, there is only one unique way of typing a word. The spelling of a word is now the phonetic order of the constituent basic characters. This provides a unique spelling for each word, which is not affected by the display rendition. Unique spellings are essential for making spelling checkers and dictionaries. They are also essential to facilitate finding of words in a word-processor, or for information retrieval from a data-base.

Direct Sorting

Since there are variations in ordering of a few consonants between different Indian scripts, it is not possible to achieve perfect sorting in all Indian scripts. Special routines would be required for cultural sensitive requirements. For most purposes, however, the direct sorting achieved through the ISCII code should be sufficient.

Display Independence

A word in an Indian script can be displayed in a variety of styles depending on the conjunct repertoire used. ISCII codes however allow a complete delinking of the codes from the displayed fonts.

An ISCII syllable can be displayed using combination of basic shapes. Different implementations can choose variant techniques in combination of these basic shapes. The same text can thus be seen in different font styles by using a different font composition routine.

Transliteration

The ISCII codes are rendered on the display device according to the display composition methodology of the selected script. Transliteration to another script can thus be obtained by merely redisplaying the same text in a different script. Since the display rendering process can be very flexible, it is possible to transliterate the Indian

scripts to the Roman script, using diacritic marks. Similarly it is possible to transliterate them to their scripts such as Perso-Arabic.

Transliteration involves mere change of the script, in a manner that pronunciation is not affected. This is not the same as “translation” here the language itself changes.

The INSCRIPT Keyboard Overlay

The Inscript overlay contains characters required for all the Indian scripts, as defined by the ISCII character set. The Indian script alphabet has a logical structure, derived from the phonetic properties. The Inscript overlay mirrors this logical structure. The overlay has also been optimized from phonetic considerations. It is divided into two parts : the vowel pad on the left hand side, and the consonant pad on the right hand side.

Due to the phonetic/alphabetic nature of the keyboard, a person who knows typing in one Indian script can type in any other Indian script. The logical structure allows ease in learning and speed in touch typing. The keyboard remains optimal both from touch-typing and sight-typing points of view, in all Indian scripts.

UNICODE and ISCII

The Unicode standard for Indian scripts is based on the ISCII-1988 revision. In November 1991, at the time the Unicode standard, Version 1.0 was published, the Bureau of National Standards published the current ISCII in the Indian Standard IS : 13194 : 1991. The Unicode standard remains a superset of the ISCII-1991 character encoding and texts encoded with ISCII-1991 may be automatically converted to Unicode code values and back to their original encoding without loss of information. The following Indian scripts are supported in Unicode. Devanagari, Bengali;, Gurmukhi (Punjabi), Gujarati, Oriya, Tamil, Telugu, Kannada and Malayalam.

CONCLUSIONS

The ISCII character encoding standard in its present form has been under implementation in the field since 1988. It has proven to be robust, with a wide range of applications existing under diverse computing platforms. Its usage has been made mandatory in many projects of state and national significance : the country’s electoral data in Indian scripts is maintained in ISCII.

There has been in recent times, debate on the perceived shortcomings of ISCII and that there is scope for improvement. In the interest of all concerned, it would be prudent to work with the current standard than attempt to change it with the instability and uncertainty inherent in the process.

3.7.3 UNICODE

Unicode is a standard, by the Unicode Consortium, which defines a character repertoire and character code intended to be fully compatible with ISO 10646, and an encoding for it. ISO 10646 is more general (abstract) in nature, whereas Unicode “imposes additional constraints on implementations to ensure that they treat characters uniformly across platforms and applications”,

Unicode is a 16-bit character set designed to address the limitations of 8-bit character sets and inconveniences of multi-byte character sets. In Unicode, each character occupies two bytes, giving a possible total of 65536 values. unicode represents each character by a single number.

Not only letters, digits, punctuation, and ideographs are mapped, but also graphical, line-drawing, musical, mathematical, scientific, and other symbols have been mapped. For the most part, the designers of Unicode tried to accommodate all of the symbols used in other character sets. For instance, since the common PC character set used by IBM included certain line-drawing characters, those line-drawing characters may be found in Unicode.

Even though a character may be used in more than one language, it is defined only once in Unicode. For example, the letter **LATIN capital letter A** is mapped once, even though it is used in English, German, and (as romaji) in Japanese. On the other hand, it was decided that **CYRILLIC capital letter A** is a different character from **LATIN capital letter A**, even though the two letters look a lot like each other. The reasoning behind decisions like these is interesting to linguists, but usually not important to programmers.

To facilitate use of Unicode, and interoperation with other character sets, the Unicode Consortium also publishes conversion tables to convert among Unicode and these other mappings. *Every major character set can be mapped to Unicode, but the converse is not true.* When converting from Unicode to some other character set, there are Unicode characters which cannot be mapped. A programmer must make provisions for dealing with these inconvertible code values.

Unicode reserves certain code values for proprietary characters, so an implementation may define new character mappings. These may be used for alphabets or ideographs not yet defined, or they may be used by firms for corporate symbols. Use of proprietary characters carries the risk that interoperability with other implementations may be reduced.

The mappings defined by Unicode are not static. Occasionally since the first version of Unicode was released, the Unicode Consortium has released updated versions of the standard, which map more and more characters.

The main properties of the characters are :

- Name : a unique name assigned to the character
- Numeric values : the character's numeric values (there are three kinds of numeric value : decimal digit, other digit, and other numeric value)
- Category : letter, digit, symbol, etc.
- Directionality (“directional category”) : shows whether the character has intrinsic left-to-right or right-to-left attributes
- Case : mapping to uppercase, lowercase, and title case equivalents
- Decomposition : the mapping to simpler components
- Combining class : designates how certain characters interact typographically

There are also other properties assigned to the Unicode characters, but these are the most important for most processing. The Unicode standard does not define any specific collating or sorting sequence, although it provides guidance to developers wanting to implement collating sequences.

Operating system support for Unicode

Windows NT, 2000, and XP are all using Unicode internally. At the same time, Windows 95, 98 and ME generally do not support Unicode at all. Unicode versions of Win32 functions are not implemented. This is in part why most programmes use ANSI even though they could use Unicode, especially in the English-speaking countries where localization issues do not exist. However, ANSI functions on NT are implemented quite simply : they convert their arguments to Unicode and pass them to Unicode functions that do the actual work, converting the results back if necessary. Therefore, on Windows NT it is faster to use Unicode functions.

There exists Microsoft Layer for Unicode (MSLU) that is designed to allow Unicode-built programmes to run on Win9x. In your programme you call Unicode versions of the functions implemented by MSLU, and it in turn converts Unicode parameters to ANSI/ and passes them to the Win32 functions to do the work. MSLU is not designed to work on Windows NT, because NT has native support for Unicode and because Unicode to ANSI conversions potentially lose data (remember that you're converting 16-bit characters to 8-bit ones).

Even in circumstances where Unicode is supported in principle, the support usually does not cover *all* Unicode characters, For example, a font available may cover just some part of Unicode which is practically important in some area. On the other hand, for data transfer it is essential to know which Unicode characters the recipient is able to handle. For such reasons, various **subsets** of the Unicode

character repertoire have been and will be defined. For example, the *Minimum European Subset* specified by ENV 1973 : 1995 was intended to provide a first step towards the implementation of large character sets in Europe. It was replaced by three *Multilingual European Subsets* (MES-1, MES-2, MES-3, with MES-2 based on the *Minimum European Subset*), defined in a CEN Workshop Agreement, namely CWA 13873.

Glyph

A Glyph – a visual appearance

It is important to distinguish the character concept from the glyph concept. A *glyph* is a presentation of a particular shape which a character may have when rendered or displayed. For example, the character Z might be presented as a boldface **Z** or as an italic Z, and it would still be a presentation of the same character. On the other hand, lower-case z is defined to be a separate character-which in turn may have different glyph presentations.

Example: a letter and different glyphs for it

LATIN CAPITAL LETTER Z (U+00E9)				
Z	Z	Z	z	Z

Glyph Variation

When a character repertoire is defined (e.g. in a standard), *some* particular glyph is often used to describe the appearance of each character, but this should be taken as an example only. The Unicode standard specifically says (in section 3.2) that great variation is allowed between “representative glyph” appearing in the standard and a glyph used for the corresponding character :

3.7.4 GIST Card

The GIST card makes it possible to use the IBM-PC compatible in a script independent way. What could be done in English, can now be done in different scripts like Tamil, Bengali, Devnagari and so on. The existing software meant for taking input in English can now accept data in any script. It can also display it in the same script or a different one, instantly. This is possible due to the transliteration feature.

The C-DAC GIST Card is a PC add-on card that allows the use of Indian & other scripts, with English, in all existing text-based application packages like dBase, foxPro, Lotus 1-2-3, QBasic, and compilers like C, C++ and Cliper, on MS-DOS.

All major Indian scripts are supported on GIST. It includes-Devnagari (used for Hindi, Marathi and Sanskrit languages), Bengali, Gujarati, Punjabi, Tamil, Telugu,

Malayalam, Kannada, Oriya, and Assamese. Even the right to left scripts like Urdu, Sindhi, Kashmiri, are provided. Other available scripts are English, Tibetan, Druk (Bhutanese), Sinhalese, Russian, Arabic, and Thai.

The C-DAC GIST Card is ideal for multilingual database applications on MS-DOS, wherein the database can be entered, processed, sorted, displayed and printed in any Indian language with English. Some of the major multilingual applications of C-DAC GIST Card are

- Accounting packages
- Payroll-Payslips, employee reports etc.
- Land records - Data entry and printing
- Voter ID Cards
- Printing or Reports, ledgers etc.
- Printing of Invoices, Bills, Receipts, Cash Memos, Vouchers etc.
- Correspondence in Indian languages

There are different type of GIST cards designed to work with GGA/HGA/EGA (V1.3/1.51), and VGA (V1.44) display adapters. Due to differences in the hardware, it is advisable to furnish the following information while ordering the card - type of computer, manufacturer, display adapter, monitor type (colour/monochrome, TTL/composite/Analog).

References and Further Reading List

- 1 2005 Request for Proposal for a Client/Server Electronic Library System (<http://www.libraryhq.com/rfp.doc>). Visited last : 12/08/2005.
- 2 2005 Library acquisition functions (http://www.cc.nctu.edu.tw/~claven/course/Library_Automation/acquisition.ppt) Visited last : 25/09/2005.
- 3 2005 A tutorial on character code issues (<http://www.cs.tut.fi/~jkorpela/chars/index.html>). Date of last update : 2005-08-10. Visited last : 30/09/2005.
- 4 2005 Basics of Unicode (https://www.tatanka.com/doc/man/nigeria/nigeria_guide_html/book1.html). Visited last : 30/09/2005.
- 5 2005 Gist card (<http://www.cdac.in/html/gist/products/gcard.asp>). Visited last : 30/09/2005.
- 6 2005 MARC 21 LITE Bibliographic Format. [<http://www.loc.gov/marc/bibliographic/lite/>]
- 7 2004 Haravu (LJ). Library automation : design, principles and practice. New Dlehi : Allied Publishers. 2004.

- 8 2004 MARC 21 tutorial [http://www.lib.usm.edu/~techserv/pdc/marc21_tutorial_ie/index.htm] Visited last :
- 9 2003 Understanding MARC Bibliographic Machine-Readable Cataloging [<http://www.loc.gov/marc/umb>]
- 10 2003 MARC Development [<http://www.loc.gov/marc/development.html>] Visited last :
- 11 2002 Pudeyey (Oleg). Introduction to Unicode. ([http://www.rpi.edu/~pudeyo/articles/introduction to Unicode.htm](http://www.rpi.edu/~pudeyo/articles/introduction%20to%20Unicode.htm)). Last updated : 15/07/2002
- 12 2000 Bhatt (Shashank). Character Encoding Standard for Indian Scripts - A report. (<http://>). Visited last : 30/09/2005.
- 13 1998 Hall (Patrick) and Clews (John). Customizable Internationalized Software : A win-win strategy for the South Asian software industry : *Proceedings of the SAARC Conference on Extending the use of Multilingual and Multimedia I++nformation and Technology*, Pune, India, September 1998.
- 14 1997 Cohn (John M), Kelsey (Ann L) and Fiels (Keith Michael). Planning dor automation : a how-to-do-it manual for librarians. 2nd ed. New York : Neal-Schuman Pulishers, Inc., 1997.
- 15 1997 Meghabghab (Dania Bilal). Automating media centers and small libraties : a microcomputer-based approach. Englewood : Libraries Unlimited, 1997.
- 16 1996 Unicode Consortium (1996), *The Unicode Standard. Version 2.0*, Addison Wesley 1991-6. ISBN 0-201-48345-9
- 17 1995 Electronic Data Interchange : An Overview of EDI Standards for Libraries (1993). International Federation of Library Associations and Institutions. (<http://www.ifla.org>). Visited last : 25/06/2005.
- 18 1991 Bureau of Indian Standards (1991), Indian Script Code for Information Interchange IS 13194 :1991), New Delhi, India.
- 19 1984 Genaway (DC). Integrated online library systems : principles, planning and implementation NY : GK. hall. 1984.
- 20 1983 Rao (I.k. Ravichandra). Library automation (DRTC Refresher seminar 14, DRTC, Bangalore, 1983)

3.8 Exercise

1. Discuss different subsystems and their functions.
2. Discuss importance of MARC format.
3. Trace the history of development of MARC format.
4. Describe character encoding in the context of computerized cataloging.

Unit 4 □ Computer based Serials Control`

Structure

4.0 Objectives

4.1 Introduction

4.2 Benefits

4.3 Infrastructure

4.4 Subsystem and their functions

4.5 Standards

4.6 Interfaces with other subsystems

4.7 File Structure for computerized serials control system

4.8 Exercise

4.0 Objectives

The objectives of the Unit are to identify :

- Functional areas of computer based serials management
- Major steps involved in various areas of serials management
- Basic data elements of different files/reports etc.

4.1 Introduction

The question of management of serials by computer produces mixed reactions. Some believe that it is most difficult operations to computerize because of idiosyncrasies of serials. Others claim that it is the idiosyncrasies that make management of serials a good candidate for the computerization. Major idiosyncrasies of the serials systems are :

- Different publication schedules (frequencies)
- Change of
 - Title of the serial
 - Publisher of the Serial
- Sometimes, a serial may split or certain serials may merge to form a new serial.
- Publishers may issue special issues, index, separate title page for each volume/ combined title page for all volumes for a particular subscription year.

- It is necessary to make the payment in foreign currencies or to its equivalent local currency in advance
- Publisher may cease publication
- The library may discontinue subscription permanently and/or temporarily.
- Serials issues numbering system. In most of the cases, serials issue number start with 1 for each new volumes. However, serials issue number may be continued irrespective of change of volume number.
- Serials may starts at different points of time in a calendar year.
- Sometimes some issues of serials may be published as combined issue.
- A serial may have more than one volume per year.
- Some serials may be available only as Online from the start or at certain point of time. That is, the library may have only the Online serial or the library may have printed copy up to certain time, and afterwards only the online version is available in the library.

Serials Management adds, edits, deletes, and displays data related to serials, journals, periodicals, and any other materials that are published more than once with some relationship between the issues. Serials Management (hereinafter Serials) functions should be integrated with the other major functions : Cataloguing, Circulation, Online Public Catalogue (OPAC), and Reports. Therefore, changes in Serials should be reflected throughout the system.

A bibliographic record in the catalogue represents all serials received by the library. This may include print serials and electronic serials. Print serial records contain holdings information for all serials as well as links to electronic format if also available.

Circulating copies of print periodicals should be bar-coded and the specific issues entered into the catalog. They are attached to the serial bibliographic record for that title.

4.2 Benefits

The computerization of serials control is important because of the fact that many libraries spend more than 75-80% of their budget on serials. It calls for management of large number of paper files. Considerable manual efforts are needed for the management of serials. The computerization may :

- Reduce the drudgery and error proneness of manual system in relation to ordering, renewals, check-in of loose issues etc.

- Offer greater control over budget management
- Improve claim monitoring
- It may also facilitate cooperative serials acquisition by libraries.
- Improve articles delivery service

4.3 Infrastructure

Infrastructures necessary for a computer based serials control system are access to the :

- External databases
- Internal databases

4.3.1 Access to external databases

Bibliographic verification of new serial is important in subscription management operation. It will also enable data import possibilities. In a computerized system, the serials control module may access the web site of the publisher and/or union catalogue of serials of a given library network. Such access would help in informed decision making and more accurate bibliographic descriptions etc. Similarly, the module may access an authorized site to verify the exchange rate etc.

4.3.2 Access to internal files

A computerized serials control system in an integrated library automation environment requires access to the following types of internal files/databases :

- Library's OPAC
- Master file of Serials
- Master file of Vendors
- Master file of Publisher
- Budget File
- Invoice File
- Subscription/renewals files

4.4 Subsystems and their Functions

A computer based serials control system could be divided into several subsystems based on the discrete functions that are required to perform. The serials control module must control the receipt of journals, series and supplements, using the database common to all other modules, so no information is duplicated. For serial subscriptions, the serials control module must include the following capabilities :

- Subscription Management
- Current Issue Registration [check-in /Receiving]
- Claim monitoring
- Routing (Obsolete)
- Information Retrieval
- Bindery preparation
- Union List of Serials Subsystem
- Reporting

Subscription Management Subsystem

It performs a pre-order search and handle renewals, new subscription and fund control of serials. Typically, this module supports the following activities :

- The Serials Control module must detect and alert operator about duplicated between firm orders and subscription orders for monographs in series.
- Pre order search to avoid duplicate ordering to same serial
- Provide alert for irregular serials
- Provide renewal alerts
- Place orders for new serials
- Place renewal orders for serials that are already on subscription in earlier years.
- Facilitate acquisition of serials on exchanges or as gifts
- Control expenditure.

Current Issue Registration [Check-In/Receiving] Subsystem

This subsystem facilitates registration of loose issues. Each serials control (Check-in) record must :

- be associated with a title in the catalogue
- designate whether or not to enter copy specific information into the catalogue at check-in
- establish the number of latest issues to display in the OPAC, with an authorized operator able to override this designation at check-in
- accept the loading of data for titles held by other libraries and not controlled by the system for output in union list
- record and maintain discard information, for producing automatic discard alerts and instructional slips for disposal of issues
- registration of multiple issues in a single transaction and producing alert for missing issue (already due but not received yet)

- Facilitating recording of special issue, annual volume, title page and index.
- Registration of duplicate copies.

Claim Monitoring

The Serials Control module must flag items as missing, overdue, duplicate, or to be retained for reconsideration. The Serials Control module must enable an authorized operator :

- to automatically generate claim notices at intervals specified, in printed and machine-readable format
- to add a claim to the claim list for a title by filling in a screen workflow
- to send as many claims as desired for a missing issue or copy
- to specify the text of each claim
- to determine claim action dates by expected receipt dates combined with an operator-specified claim interval
- to change the claim interval for each title at any time
- to identify issues requiring second and third claims according to library determined time legs that may be defined for various item types
- to identify items for which three claims have been issued without a response being recorded, and make them available for staff review to determine further action
- The Serials Control module must enable recording specific details of responses to claims.

Using the serials control record pattern, the Serials Check-in and Control module must have the ability to generate predictions, expected issues, for each serial. A prediction record must contain information about a particular issue such as enumeration, chronology, and number of copies expected. At the Library's discretion, predictions may be generated as part of the receipt process once the last predicted issue has been received.

Routing (Obsolete)

The Serials Check-in and Control module must support both formal and informal routing lists. Using formal routing, each routed issue is charged to the first user on the associated routing list. As each user returns the issue to a designated point, the issue is discharged and then automatically charged to the next user on the routing list. With informal routing, staff attaches printed routing slips to the copy being routed. With formal serials routing, the Serials Check-in and Control module must consult each user's priority rank when establishing the order of recipients. Users with higher rank should appear above lowered ranked users within the routing list.

Binding subsystem

This subsystem manages and facilitates binding requirements of a serials binding department. Certain typical functions are :

- Alerting staff about completion of a volume after receiving the last issue of the volume.
- Allow staff to review the situation in a batch mode. Here the staff may provide cut-off date and get a detailed list
- Produces binding orders
- Facilitate receiving and accessioning of bound volumes of serials.

Information Retrieval (Search Facility)

The serials Control module must provide the ability to search for records by :

- Keyword search of every indexed bibliographic field
- Vendor
- Fund number/Fund Code
- Purchase order number
- Location
- System assigned number
- Bibliographic utility assigned number

Union List of Serials Subsystem

This subsystem is important if the concerned library takes part in library network and/or a part of multiple libraries of an organization/institute. The basic functions are :

- Information Retrieval Interface
- Consolidation of holdings information of all serials.

Reports

The Reports module must be fully-integrated with all other system modules, and provide a comprehensive suite of library-customizable report templates. The Reports module must enable an authorized operator to schedule production of report output at a specified date and time and on a regular periodic basis, such as daily, weekly, monthly, and at pre-specified times. The Reports module must enable an authorized operator to :

- View completed reports on screen or to e-mail or print the report, at the operator's convenience.

- Select items for inclusion based on any combination of bibliographic information (using full Boolean word and phrase searching) and on any combination of control information, for example collection, current status, number of circulations, number of holds, classification, and accessions date.
- Specify the starting date-and-time and ending date-and-time that the report is to cover on reports involving historical data.
- Use same user interface as other modules.
- Perform housekeeping tasks
- Select customize, name, save and schedule reports
- Display and/or e-mail finished reports

4.5 Standards

The Library, or each library in a multi library system, may maintain its own serials control record. The serials control module must support the following industry standards :

- SISAC checking with SICI barcode scanning
- MARC21 Format for Holdings Data (MFHD)
- If the library desires, automatically generate multi-library holdings data for the 852, 8533, and 863 holdings tags
- Holdings record follows the punctuation as described in Z39.71
- Supports NISO Z39.45 Claims for Missing Issues of Serials
- Support NISO Z39.55 Computerized Serials Orders, Claims, Cancellations
- Support NISO Z39.56 Serial Item and Contribution Identifier (SICI)
- Offer interface to third-party binding software application (e.g., LINCPlus)

4.6 Interfaces with other library automation subsystems

In an integrated computerized library environment, it should be possible for one module to interact/use data and files of other modules. The computerized serials control system should have access/interfaces to the following subsystems :

- Cataloguing/Bibliographic Description
- OPAC
- Circulation
- System Management

4.7 File Structure for Computerized Serials Control System

There are certain similarities between computerized acquisition system and a computerized serials control system. In an integrated computerized serials control system, the common files are shared by more than one subsystem. Considering the idiosyncrasies of serials, the following files may be maintained and/or accessed :

- Bibliographic File
- Master File for Serials
- Binding File for Serials
- Binding Specification File
- Holdings File
 - New/Loose/Current Issues
 - Bound Volumes
- Budget File
- Approval file
- Order File
- Invoice File
- Vendor/Publisher File
- Currency File
- Exchange Rate File

This is an illustrative example of required data files with individual data elements for unique files of the system :

Bibliographic File

It holds permanent data about serials. Each record of this file contains bibliographic details of the serial as per MARC21 and any other agreed international standard for bibliographic description of serials. The link may BIB-ID field.

Master File-Serials

It holds data relevant for the current year subscription/renewals.

- Serial-ID (Primary key) : Generate automatically
- Key Title/Serial Title
- Publisher
- ISSN
- BIB-ID

- Holding-ID : Link to records of the serial in the Holding File
- Current Subscription Data
 - Starting Date
 - Ending Date
 - Number of volumes per year
 - Stating Issue Number
 - Continuous Issue No : Yes/No
 - Number of Issues per volume
 - Frequency
 - Order-ID
 - Physical Form
- Mode of Subscription : Direct/Vendor
- Separate Index Page
- Separate Title Page
- Binding
 - Yes/No
 - Binding-ID
 - Alert : Yes/No
- URL

Binding Specification File

- Binding-ID
- Spine Title
- Class No
- Book Number
- Volume Number Format
- Issue Number Foramt
- Binding Type
- Binding Colour
- Library Name Format
- Note, if any

Holdings File

The complication of serials cataloguing is that some of the holdings of serials are in bound form while others are held as loose issues until that they are ready for binding.

Library accession only bound volumes. Both these forms are holdings and they must be available in the holdings database/catalogue. The sample fields are :

- Holding-ID : Unique ID for each physical volume (loose/bound). Generated automatically.
- BIB-ID
- Library-ID (In case of Union Catalogue of Serials)
- Location
- Barcode/Accession Number
- Circulation-ID
- Class Number
- Book Number
- Volume Number
- Issue Number
- Copy Number
- Starting Volume Number (in case of bound volume)
- End Volume Number (In case of bound volume)
- Starting Issue Number (In case of bound volume)
- End Issue Number (In case of bound volume)
- Physical Status : Loose/Bound
- Expected Date (In case of 1st issue)
- Received on
- Claim-Count
- Binding-Status

References and Further Reading List

- 1 2005 Request for Proposal for a Client/Server Electronic Library System (<http://www.libraryhq.com/rfp.doc>). Visited last : 12/08/2005.
- 2 1997 Cohn (John M), Kelsey (Ann L) and Fiels (Keith Michael). Planning for automation : a how-to-do it manual for librarians. 2nd ed. New York : Neal-Schuman Publishers, Inc., 1997.
- 3 1997 Meghabghab (Dania Bilal). Automating media centers and small libraries : a microcomputer-based approach. Englewood : Libraries Unlimited, 1997.
- 5 1983 Rao (I.K. Ravichandra). Library automation (DRTC Refresher seminar 14, DRTC, Bangalore, 1983)

- 6 1984 Genaway (DC). Integrated online library systems : principles, planning and implementation. NY : GK. Hall, 1984.
- 7 2004 Haravu (LJ). Library automation : design, principles and practice. New Delhi : Allied Publishers, 2004.

4.8 Exercise

1. Discuss various idiosyncrasies of serials.
2. Describe the subsystems and their functions of a computer based serials control system.
3. Briefly discuss standers, which are applicable in serials control system.

Unit 5 □ Retrospective Conversion and Bar-coding

Structure

- 5.0 Objectives**
- 5.1 Introduction**
- 5.2 Benefits**
- 5.3 Phases of Retrospective Conversion**
- 5.4 Opportunity and the need for a National Strategy**
- 5.5 Barcode**
 - 5.5.1 Definition**
 - 5.5.2 Structure**
 - 5.5.3 Benefits**
 - 5.5.4 Symbologies**
 - 5.5.5 Selection of Symbologies**
 - 5.5.6 Barcode Scanners**
 - 5.5.7 Barcode Printing**
 - 5.5.8 Label Printing Software**
- 5.6 Exercise**
- 5.7 References and Further Readings**

5.0 Objectives

The objectives of the Unit are to understand :

- What is retrospective conversion of library catalogues
- Importance of retrospective conversion
- Stages of retrospective conversion
- Processes of retrospective conversion
- What is bar coding
- Importance of bar coding in the library context
- Important points should be considered in selecting bar code symbology

5.1 Introduction

One of the fundamental services that a library offers, in addition to collecting and preserving the fruits of a society's intellectual endeavors, is to describe and organize

its collections in order to guide people to materials on a certain topic or by a certain author. Libraries collect information, make it readily available, and present that information in an organized fashion. For hundreds of years, libraries are providing indexes and lists of what they owned only in paper form. One of the most highly developed (and cherished) systems of this kind of access is the card catalogue.

Retrospective conversion is a process of creating online cataloguing records for existing documents of a library. That is, it is the process of turning a library's existing manual catalogue into a machine-readable form. The success of retrospective conversion is less linked to the chosen technique itself than to the care brought to the initial analysis, i.e., the main issue is less technological than methodological.

Retrospective Conversion is the process by which data on each item in the library is entered into a computer system so that they can be integrated into the database that is the very foundation of many if not most of the services that the Library provides. The majority of the catalogues to be converted are card catalogues; however other forms include printed book and sheaf etc.

5.2 Benefits of a Retrospective Conversion

The benefits of retrospective conversion of library catalogue records are both local, within the holding institutions, and to the wider library, research and scholarly communities. Once the catalogue records are created they are valuable in themselves since other libraries and services can use them without duplicating professional effort. The some of the advantages of conversion of the card catalogue of a library to online records are :

- It provides greatly improved access for users to the library's collection.
- Facilitate remote access to the library catalogue
- Online catalogue record contain more information than the cards
- Integration of records for older material with current cataloguing
- A much better return on the capital value of a library's stock is achieved, maximizing the value of an investment made over a long period of time.
- The inter-library loan system does not become needlessly burdened with requests that could be satisfied from local stock if only the catalogue records for all items in the library were recorded in machine-readable form.
- Scholars and researchers are provided with information, not previously available, for unique or rare items.

- The burden of supplying items too frequently demanded from a few institutions can be shared since the locations of additional copies are known.
- Individual libraries are, for the first time, able to assess their own holdings in the wider national context enabling prudent management decisions to be made in relation to the acquisition, preservation and withdrawal of items.
- The production of a more accurate picture of the nation's total bibliographic resource.
- Facilitate automated circulation using member and item bar codes.
- There is recognition of the importance of organizing efficient and fair access to the total national resource of bibliographic records.
- Computer technology allows the creation of union catalogues in database form at low unit cost, giving immensely greater availability, far easier to maintain and keep up to date.

5.3 Phases of Retrospective Conversion

Retrospective conversion involves several considerations and it can be accomplished in several ways. The optimum method or combination of methods for a given library depends on many factors. The phases of retrospective conversion are :

- Phase I : Preparing for Retrospective Conversion
- Phase-II : Selection of Retrospective Conversion Techniques
- Phase-III : Staffing
- Phase-IV : Quality Control and Problems Resolutions
- Phase-V : Bar coding

5.3.1 Phase-I Preparing for Retrospective Conversion

This phase is concerned with formulation of goals, selection of conversion basis, and weeding out etc. It consists of the following steps :

- **Goal Setting** : It is essential to assess the expectations of the library, needs and requirements. The impact factors are : possible participation in library network, time available, quality of source of information, and need for reclassification etc.
- **Selection of Sources of Information** : It is necessary to establish a conversion basis, which depends on the completeness of information available in different possible sources : Accession Register, Shelf-List, Catalogue Card, and Non-MARC records etc.

- **Identification of Local Fields :** Current practices for recording this type of information need modification. Some systems will allow local data to be entered in variety of fields, while others require information to be entered in a special field.
- **Selection criteria for matching :** It is necessary to evaluate information from several fields to ascertain that a record matches the item on hand. To be considered a match, the database record and the shelf-list record will normally have same title, author, place, publisher, date, pagination, edition etc. However, “match” may occurred even if there are certain discrepancies between the two set of records under certain conditions. Cataloguing rules and procedures have changed over the years, and cataloguing also involved certain degree of personal interpretations. Hence, there will often be minor variations between shelf-list cards and other sources of information (records in databases).
- **Establish priority :** Retrospective conversion is time consuming activity, Hence, prioritize the collections for computerization on the basis of certain well enumerated policy.
- **Verification of shelf-list :** Weeding is an ongoing process but it becomes critical when the decision to automate is made. Determine what exist in the collection to ensure that each record is worth conversion. Titles/copies no longer in the collection or may be discarded should not be converted.
- **Selection of record format :** To ensure completeness of information and possibility of participation in library resource sharing programmes etc, It is essential that standard MARC formats for bibliographic descriptions and authority controls in case of personal and corporate names, geographic name, and subject heading etc. should be followed in aspects of cataloguing.
- **Funding :** Both the one time costs and the continuing expenses of retrospective conversion are substantial. The library must commit funds to support record conversion and to maintain records in an automated system. A funding plan will help to determine internal funding resources and the need for external funding. The plan should estimate costs based on following factors : Collection size and types, media of exposition, language and scripts, completeness of present information, standardization, expertise needed, hardware and software cost, cost of annual maintenance etc.
- **Documentation :** It is crucial to successful retrospective conversion and automation programme. If changes in procedures are necessary, add them to the specification. To ensure efficient work flow, different types of manual should be prepared on various facets of the programme.

5.3.2 Phase-II Selection of Retrospective Conversion Techniques

There are different methods of retrospective conversion available today; ranging from manual transcription of cards to fully automated OCR technology. The following cases must be considered separately :

- **Manual Transcription** : It is time consuming process. Need higher initial facilities for conversion of records.
- **Downloading** : Using machine readable data may immediately seem the best solution, it still involves certain works :
 - Relevant records have to be retrieved and extracted from the source database
 - Certain adaptations and enrichment are necessary-adapting authority file based headings, subject terms and local notes etc.
 - Input of local data : class number, book mark, location mark(s), and accession number etc.

Machine readable records may be created in many ways. The choice of a technique or a combination of techniques depend on the different states in which the records may be available, such as

- Catalogue cards or its equivalent, bibliography
- Catalogue cards in digitized form
- Plain text
- Structured text
- **OCR** : The application of optical scanning technology to retrospective conversion continues to evolve. Information currently on paper, microfilm and other media can be scanned and through OCR, made accessible in electronic form for use online. Applying MARC tagging to the bibliographic information will allow the creation of full MARC records.

Incomplete cataloguing information and illegible or handwritten cards can create problems for this technology. However, these obstacles are small when compared with the possible applications for automating searching, matching, adding local information, and creating new records.

- **Combination of above methods** : An organization may be in a position to employ all these techniques depending on availability of suitable opportunities and/or resources.

5.3.3 Phase-III Staffing

Staffing issues will arise in different steps of the retrospective conversion of library catalogue. There will be additional work for existing staff, and perhaps the need to hire additional staff to assist with such activities as weeding, preparing shelf-list, retrospective conversion (i.e., inputting data/importing data etc.), and implementing the System. The two primary approaches to staffing are :

- Off-site
 - Full vendor contract
 - Batch retrospective conversion
- In-house
 - Libraries can use a number of staffing arrangements to complete a retrospective conversion. In some cases, combinations of arrangements may be preferred. The choice depends on the following points :
 - ◆ Budgets
 - ◆ Collection size and type
 - ◆ Quality expected
 - ◆ Time constraints
 - ◆ Expertise

5.3.3.1 Full Vendor Contact

An off-site conversion is the easiest and most comprehensive way to convert catalogue provided up to date and complete catalogue is available for the collection. The selected vendor(s) completes the project. The fee is based on various factors, such as :

- Volume of records to be converted
- Types of materials
- Languages of the materials
- Special Knowledge requirements
- Volume of new records to be created
- Assignment of local information-subject heading, call number, and location etc.

The library is responsible for complete specifications. The vendor(s) and the library should select a contact person at each end to handle all details. The method does not require additional personnel and equipment. It does not affect current workflow pattern.

Advantages and limitations of full vendor off-site conversion of library catalogues may be summarized as follows :

Advantages	Limitations
<ul style="list-style-type: none"> ● No need to hire, house, train and supervise retrospective conversion staff ● High conversion rate ● Records meet the quality standard ● Less expensive 	<ul style="list-style-type: none"> ● Transfer of source of information (Shelf List Cards) may create problems ● Quality check is a concern ● Not employing high quality workers by the vendor to keep the cost of conversion low

5.3.3.2 Batch Retrospective Conversion

This process is beneficial if the collection consists of mainly books and monographs. If the machine readable records of the collection exist, the data can be formatted for machine upgrading. When pre-existing machine readable records are not available, search arguments are created and entered on floppy for processing. Both alternatives allow the input of local data.

The vendor is responsible for retrieving of matching records. Batch conversion is generally do not allow creation of original records. As a result, no-matches must be resolved by some other method. The cost for batch conversion is generally a unit charge based on the number of records converted. Some vendors may charge a processing fee for each batch processed. Others may require a minimum amount of processing volume.

Advantages and limitations may be summarized as follows :

Advantages	Limitations
<ul style="list-style-type: none"> ● Inexpensive ● Very high rate of conversion ● Library control the quality of data entry ● Vendor(s) takes the responsibility of hardware and software needs. 	<ul style="list-style-type: none"> ● The library may have to hire, train, house, and supervise retrospective conversion staff ● Problems of multiple matches should be resolved. ● Quality of records may not satisfy requirement of a MARC format.

5.3.3 In-house Retrospective Conversion

The library may decide to perform retrospective conversion using a large database which generally have hardware and software support for cataloguing. The library enjoys complete control over creation and editing aspects of the retrospective conversion. The immediate costs for this type of conversion programmes are :

- Subscription of databases
- Search charge, if any
- Transaction fees : addition of holding data, download charges etc.

Advantages and disadvantages are :

Advantages	Limitations
<ul style="list-style-type: none">● Quality of records depends on the performance of the library workers.● New records can be created at the time of detection● Library control the project	<ul style="list-style-type: none">● The library may have to hire, train, house, and supervise retrospective conversion staff● Problems of multiple matches should be resolved.● Quality of records may not satisfy requirement of a MARC format.

5.3.4 Phase-IV Quality Control and Problem Resolution

It often comes toward the end of a retrospective conversion project, and it deals with relatively small portion of the total record processed. The integrity of the database depends, in part, on standardization of the editing process. Retrospective conversion problems solving requires a historical as well as current knowledge of cataloguing policy. Professional who earned their degree since adoption of AACR2 need to take crash course in cataloguing history to understand the format of some of the records they will encounter.

It also requires knowledge of insitution specific cataloguing practice and accession policy etc. Knowing what is not possible on the local system helps to define and limit the options available for solving the problems. Problems in retrospective conversion can be categorized as follows :

- Elements of Bibliographic Descriptions
- Title problems
- Main title and a supplement belong to the same record

- Date problems
- Edition conflicts
- Collation problems
- Photocopies of thesis/monographs
- Publishers
- Independently published items bound together
- Authority control
- Weeding problems : Lost, missing and withdrawn volumes
- Database preparation
- Duplicate record resolution
- Merging local information
- Filling indicator

5.3.5 Phase-V Bar coding

Bar codes were first introduced in the early 80's in the USA to support the grocery industry. Their usage received a forced boost in 1982 when the Department of Defense, USA issued Military Standard 1189, requiring their suppliers to adapt to their far code standards. One thing to remember with bar codes-the application software that accepts the bar code data is in 95% control of the success or failure of an application.

5.4 Opportunity and the need for a National Strategy

There is recognition of the importance of organizing efficient and fair access to the total national resource of bibliographic records. There is concern that all converted records should meet bibliographic standards of a level, which will ensure effective consultation and exchange.

Within the higher education community there is ample empirical evidence that, if retrospective conversion is left to the decisions of individual institutions, its achievement will be haphazard and long-drawn out. Since the achievement of 'universal' retrospective conversion will benefit the whole of the centrally/state funded community, it seems essential to promote a co-coordinated national effort. Further more the centre/state community is both the major provider and the major user of the scholarly library resources in India. It is therefore a responsibility for the centre/state sector to make its own major contribution towards unlocking the nation's total resource of scholarly library material.

There is a need to develop a strategy for implementing a national programme of retrospective conversion. This should incorporate a business plan with tight budgeting and be prepared at an early date. If nothing is done, the consequences for research, especially in the humanities and social sciences, will be seriously affected and will prevent the full use of the unrivalled resources in the nation's often unknown and frequently under-utilized library collections. Significant developments in recent years have provided both the challenge and the opportunity to tackle the retrospective conversion of library catalogues at a national level. A distributed approach, presenting a virtual union catalogue to the user, is the likely way ahead.

The implementation and management of a national programme should be entrusted to an umbrella management group, representative of bodies currently active in the field. This group should have its own secretariat and be responsible for :

- Overall planning,
- Funding,
- Establishing criteria for projects and priorities,
- Tendering,
- Ensuring that catalogues/databases are properly maintained, standards met, and regular monitoring of progress.

There is a need for one body to have responsibility for coordinating a national strategy. At present, agencies and institutions operate independently; there is no single body with overall responsibility for coordinating projects and setting priorities for retrospective conversion in India. The national body must address the following issues.

- **Complexity** : Retrospective conversion of library catalogues is complex. In addition to the number of records requiring conversion, there are problems posed by the sheer range of materials; the languages and scripts involved; and making sure that the large number of libraries from so many sectors—each sector with its own priorities—can work together effectively toward a common goal.
- **Access to items** : A national programme of retrospective conversion requires agreed criteria to ensure satisfactory access to items in the collections of the participating institutions. The legitimate interests of owners must be protected. In return for funding, reasonable facilities for access must be guaranteed for consultation of material, either in its original state, or, where this is not possible, in a surrogate format.

- **Access to catalogue records-standards and distribution :** Converted records will need to be produced to acceptable standards and a decision made as to how these records are to be distributed and accessed. The common bibliographic data and formal rules to which converted catalogues conform should be the minimum required to enable the catalogue records to be consulted effectively and exchanged within and across national boundaries. Records created as a result of a national retrospective conversion programme should ideally remain in the public domain.
- **Staffing and expertise :** There is a need for skilled cataloguers (they are no longer produced by departments of library and information studies in the numbers that they were) and expertise in the management of retrospective conversion projects. Many smaller libraries, and libraries which are not publicly funded, are administered by part-time or voluntary staff and, even when the necessary expertise is possessed by the staff in post there may be little or no spare time to undertake the additional work involved. In the event of a team of cataloguers with the necessary skills being appointed, or brought in from outside the library, the necessary accommodation and equipment must be available.
- **Preservation :** It might be argued that increased handling of items can lead to accelerated deterioration in the physical condition of the items concerned. Collections are far more likely to be in danger through lack of knowledge of what they contain, or inadequate awareness by the owning institutions of the value of the items they possess. This can easily lead to neglect and dispersal of material and prejudice scholarship and the value of collections.
- **Priorities :** Priorities will need to be set to determine which catalogues should be included in the initial phase of a national programme. If the catalogues of larger library collections are converted first then many smaller libraries can benefit from access to the records created; however, conversion of records in particular subject areas, languages, or by dates of publication might be deemed to have greater importance. Priorities will have to be set with full knowledge of local circumstances and other factors, which might assist the shaping of a particular project.
- **Collaboration :** The success of a national programme must depend on the collaboration of libraries across sectors. Cooperation will entail guarantees on the part of participating libraries that they will provide reasonable access; ensure retention of material for which catalogue records have been converted and, as a general rule, contribute to the funding costs; although for many

smaller and privately funded libraries financial inducements to participate in the programme are likely to be necessary.

- **Costs :** Money is vital to solving the problem, as the total cost of retrospective conversion nationally would be very high. As a general rule, matching money would be expected from institutions in receipt of special funding—this could be in ‘kind’, but it could also come from a third party. Allowing for matching money of 50 per cent, the additional money required would be provided from the central pool.
- **Management :** Machinery will have to established for managing a national programme—to coordinate effort, set priorities, target funds and ensure the maintenance of the programme. There needs to be proper management of the awarding of grants; the progress of individual institutions will need to be monitored; applications will have to be vetted; allocation of funds accommodated within budgetary constraints and decisions taken to ensure that the greatest benefits from the programme are derived at the earliest possible date.

5.5 Barcode

Barcodes are a fast, easy, and accurate data entry method. The correct use of barcodes can decrease employee time required and increase an organization’s efficiency. One thing to remember with bar codes : the application software that accepts the bar code data is in 95% control of the success or failure of an application. Remember that bar codes are just another data input method. Barcodes can be read by optical scanners called barcode readers or scanned from an image by special software.

5.5.1 Definition

A **barcode** (also **bar code**) is a machine-readable representation of information in a visual format on a surface. A bar code is a series of varying width vertical lines (called bars) and spaces. Bars and spaces together are named “elements”. There are different combinations of elements, which represent different characters. It is an automatic identification technology, which encodes information into an array of varying width parallel bars and spaces.

The barcode usually does not contain descriptive data. The data in a barcode is just a reference number, which the computer uses to look-up associated record(s) which contain descriptive data and other relevant information. That is, barcodes typically have only identification data in them.

5.5.2 Structure

When a barcode scanner is passed over the bar code, the light source from the scanner is absorbed by the dark bars and not reflected, but it is by the light spaces. A photocell detector in the scanner receives the reflected light and converts the light into an electrical signal.



As the wand is passed over the bar code, the scanner creates a low electrical signal for the spaces (reflected light) and a high electrical signal for the bars (nothing is reflected); the duration of the electrical signal determines wide vs. narrow elements. This signal can be “decoded” by the bar code reader’s decoder into the characters that the bar code represents. The decoded data is then passed to the computer in a traditional data format.

Every barcode begins with a special start character and ends with a special stop character. These codes help the reader detect the barcode and figure out whether it is being scanned forward or backward. Some barcodes may include a checksum character just before the stop character.

5.5.3 Benefits :

- Improve Operational Efficiency : Since barcodes permit faster and more accurate recording of information, work in process can move quickly and be tracked precisely. Quite a bit of time can be spent tracking down the location or status of documents, or anything else that circulates within a library.
- Save Time : Depending on the application, time savings can be significant. Even in routine day-to-day operations the time savings of barcodes add up and improve productivity. Consider a checkout/check-in of 10 documents; it will take approximately 5 minutes or more to write down document details and accession numbers compared to about 10 to 20 seconds to scan the barcodes. In a busy operation this can be a significant saving.
- Reduce Errors : Clerical and data entry errors can be a significant source of costs and related problems : unhappy member, and time spent to track down problems are just a few examples. The typical error rate for human data entry

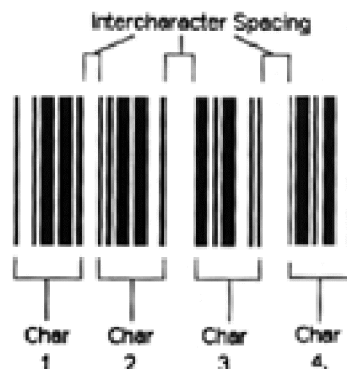
is 1 error per 300 characters. Barcode scanners are much more accurate; the error rate can be as good as 1 error in 36 trillion characters depending on the type of barcode used.

- Cut Costs : Barcodes are effective tools that can be used to address specific, localized problems or integrated into organization-wide information systems. When applied with thought and planning they can save time and reduce errors, resulting in a reduction of costs.

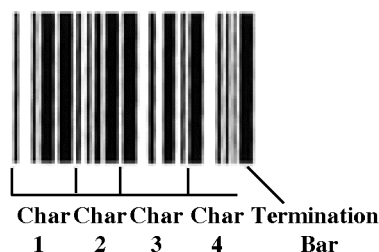
5.5.4 Symbologies

The mapping between messages and barcodes is called a **symbology**. The specification of a symbology includes the encoding of the single digits/characters of the message as well as the start and stop markers into bars and space, the size of the quiet zone required to be before and after the barcode as well as the computation of a checksum. The characteristics are :

- Character Set : A Character Set refers to what data a given barcode symbology can encode. Generally, there are three types of character sets : Numeric, Alphanumeric, and Full ASCII. In theory, a numeric character set will produce the smallest barcode whereas a Full ASCII character set will require more physical space to encode the same data. Of course, a Full ASCII symbology gives more flexibility in encoding more types of information than a numeric symbology.
- Types : There are generally two types of barcode symbologies : discrete and continuous.
 - A discrete symbology is one where each and every character encoded in the symbol may be interpreted individually without respect to the rest of the barcode. Such symbologies have characters that both start and end with a bar. Individual characters are separated by some amount of inter-character spacing. The inter character spacing carries no information-the only duty of the inter character spacing is to separate the characters.



- A continuous symbology is one in which the individual characters of the symbology cannot be interpreted by themselves. This is due to the fact that characters start with a bar and end with a space. The final space is “terminated” by the starting bar of the next character. A character cannot be taken individually since, individually, there is no way to know how wide the last space is without knowing where the next character begins. Continuous symbologies normally implement some kind of special termination bar or termination sequence such that the termination bar terminates the last space of the last data character.



In the above example, each character consists of four bars and four spaces. The first bar of character 2 terminates the last space of character 1. The first bar of character 3 terminates the last space of character 2. The first bar of character 4 terminates the last space of character 3. The termination bar terminates the last space of character 4.

All else being equal, a discrete symbology requires more space to print the same data as a continuous code since the discrete symbology “wastes” space in the inter-character spacing. However, a discrete symbology can generally be printed with less quality—this translates to cheaper printers and more tolerance at scan-time. Other than the amount of space the two types of symbologies require and the types and quality of hardware used to print them, there is no inherent difference in the security afforded by either type. That is to say, it cannot be said that “continuous symbologies are more reliable and secure than discrete symbologies”—nor can the reverse be said.

- Two-width/Multiple-width symbologies : the number of “widths” encoded in its barcodes can also divide Symbologies.

A **Two-Width** symbology has spaces and bars that are either wide or narrow. This has the benefit of simplicity—once it is determined how wide a “narrow” bar or space is, anything over a certain width can be considered “wide”. This allows for a large level of print tolerance in lower-quality printing conditions.

A Multiple-Width symbology is one, which has bars and spaces that may be of 3 or more widths. The narrowest bar or space may be X in width, a medium-width space or bar may be 2X in width, and a wide bar may be 3X in width. Since there are more possible combinations available in a multiple-width symbology, data encoding is often more efficient and results in a tighter barcode.

- Fixed/variable Length Symbologies : Symbologies may be either fixed or variable-length.

A **fixed-length** symbology is one, which must, by definition, encode a certain number of characters or digits. For example, a UPC-A barcode always encodes 12 digits of data. An application may not encode less or more than the pre-defined fixed-length of 12 characters. The symbology itself defines the length of data.

A **variable-length** symbology is one, which can carry a message of any length. For example, Code 128 may encode any number of characters that can reasonably fit physically in the printed barcode. The symbology itself does not define how many characters of data must be encoded.

Note that a variable-length symbology can be implemented by an application such that it is, in effect, fixed-length. For example, if you are encoding an identification number that is always 10 digits in length using Code 128 you are implementing code 128 as if it were fixed-length. However, the fact that you can choose the fixed-length means the symbology itself is variable-length.

5.5.5 Selection of Symbology

For new bar coding projects that don't have industry or customer standards; Code 39 is the typical non-food standard, because almost all bar code equipment reads/prints Code 39. However, Code 39 produces relatively long bar codes; it is not particularly efficient in bar code density, (the maximum density is 9.4 characters per inch including 2 start/stop characters).

Where the label width is an issue and there is numeric data or lower case data, Code 128 is the best alternative; Code 128 also has an extra efficient numeric only packing scheme to produce very dense bar codes, and Code 128 has all 128 ASCII characters. Not all readers read Code 128, so before you settle on it as a standard, be sure that your reader is 128 capable.

The larger the width of the elements, the more space it takes to print the bar code; therefore, the lower the bar code density. The thinner the bar and spaces, the less space

is required and the higher the bar code density. Lower density bar codes are more reliably printed and more consistently read than higher density bar codes, because minor variations (due to printing or damage) are much more serious with high density bar codes-the percentage of distortion is larger. Look at the samples below of different densities :



5.5.6 Barcode Reader/Scanner

There are two basic types of bar code readers : fixed, and portable batch. Fixed readers remain attached to their host computer and terminal and transmit one data item at a time as the data is scanned. Portable batch readers are battery operated and store data into memory for later batch transfer to a host computer. Some advanced portable readers can operate in non-portable mode too, often eliminating the need for a separate fixed reader.

A basic bar code reader consists of a decoder and a scanner, (a cable is also required to interface the decoder to the computer or terminal). The basic operation of a scanner is to scan a bar code symbol and provide an electrical output that corresponds to the bars and spaces of a bar code. A decoder takes the digitized bar space patterns, decodes them to the correct data, and transmits the data to the computer over wires or wireless, immediately or on a batch basis.

5.5.6.1 CCD and LASER Scanners

CCD scanners are a “can’t miss” scanner too. Most have to be placed on the code for reading but some offer “laser-like” distance reading. Some are triggerless and some require the trigger or button to be pushed to initiate reading. CCD scanners scan around 50 times per second; so unsuccessful read attempts go unnoticed.

Traditional CCD scanners have a “depth of field”, (how far you can be away from the bar code and still get a read) of only ½". They have to be placed on the code to

get a read. Just recently, CCD scanners have been developed with a depth of field previously only achieved by laser scanners. These CCD scanners are so unique that they have been termed “Linear Imager” scanners.

With Laser scanners and CCD scanners now sharing the same range, it is important to know the difference between them. Laser scanners use a single spot of light that sweeps across the bar code in a linear fashion. In a sense, lasers act like a wand, transmitting the signal for each bar and space as it “scans” across. This “scanned” pattern is then decoded. A CCD scanner on the other hand, uses an LED array with thousands of CCD light detectors for the reflected light. The entire bar code “image” is captured and then the array elements are transmitted to form a signal pattern identical to the “scanned” pattern from a wand or laser. There are no moving parts in a CCD scanner. The advantages of moving beam laser scanners are :

- Reading bar codes from a distance (typically 3-18 inches, or up to 17 feet with reading low density bar codes).
- Reading moving objects on an assembly line.
- No-hands operation. Some lasers can be mounted to turn on automatically when an object passes under the scanner. Typically used in blood banks, library check-out, etc.
- Reading through glass windows or thick laminates.
- Reading bar codes on curved surfaces, (bags of parts).
- Reading bar codes inside difficult to reach enclosures.
- Laser scanners emit a laser light beam, which sweeps back and forth across the bar code 36 times per second. At this rate, unsuccessful reading attempts go unnoticed; you will only be aware of the one successful decode. Once a read has occurred, the laser turns off, requiring you to release and pull the trigger again to reactivate the laser scanner.
- The lower the density of the bar code, the further the laser scanner can read a bar code. The higher the density of the bar code, the closer to the bar code the laser scanner must be.

5.5.7 Printing Barcodes

There are several methods of getting printed bar codes; these are :

- Buying photocomposed bar codes from a label manufacturer.
- Printing bar codes with inexpensive labeling software on dot matrix, laser, or inkjet printer.
- Printing bar codes on a specialized bar code label printer.

- For manufactures who need bar codes printed in their product's packaging, use purchased film masters or use bar code suitable for PostScript @ film output.

Whatever printing source you decide upon, there are a few common sense tips to pass on :

- Stay away from colored bar codes (use black) and colored backgrounds (use white). Any other colors lower the contrast between bars and spaces and therefore lower readability.
- Do thorough readability testing on any labels before distribution. Be careful. Don't discover a problem after you have distributed 10,000 labels that need to be recalled.

With the proper PC software, today's printers are capable of printing excellent quality bar codes. Ink Jet and Dot Matrix printers cannot print high-density bar codes, but laser printers can. Laser printers actually print the best quality bar codes of any commonly available printing technology.

5.5.7.1 Laser Printing

Laser printer can produce outstanding quality barcode labels. The quality is consistent even when toner gets low. Labels are sectionalized on A-4/any suitable size page in multiple columns and/or rows. Since laser printers feed one sheet at a time, it is impractical to print one label at a time. There is an unprintable area 1/4" inch to the left, right, top, and bottom of any form.

Laser printers are great for producing batches of labels, but if you need only one label (where there are multiple labels per page) at a time, dot matrix or thermal transfer printers are required. Laser printing is the best quality of all types. There are several types of label stock available for laser printers.

5.5.7.2 Ink Jet Printers

These printers are getting better and better. They print pages of labels. Also, use label stock specifically meant for inkjet printers-the stock is usually coated to minimize ink bleed. Always test bar code labels for readability before printing in bulk.

Windows programmes almost exclusively support Inkjet printers. Select a printer that has a separate black cartridge in addition to the color cartridge.

If labels are going to be exposed to water, don't use the inkjet printers-most inkjet ink is water-soluble. Inkjet printers are NOT the best printer to use to print labels that need to withstand the weather or are subjected to constant scanning.

5.5.7.3 Thermal Transfer Printing

Thermal transfer printers are required to print one label at a time or to print a roll of labels so that labels can be applied by applicators directly to boxes. Mostly thermal transfer printers do volume industrial printing in the 90's. They are fast and produce excellent quality bar codes.

Thermal transfer refers to the print head heating up and melting a ribbon onto the label surface. Most thermal transfer printers can also produce "direct thermal" labels, but paper instead of a soft ribbon wears out the print head 10 times faster; another disadvantage of thermal printing is that most thermal labels cannot be read with IR light and deteriorate in sunlight to non-readability over time. The media cost is about the same as laser and direct thermal. Therefore thermal transfer printing is far more popular than thermal printing for serious label production.

The basic paper labels with inexpensive ribbons produce bar codes that can be smeared or smudged with hard rubbing by the fingers. Smudge proof labels can be produced with more expensive synthetic label stock.

The print heads wear out on thermal transfer printers. To maximize the print head life, clean it between every ribbon change to avoid continually replacing print heads. Unlike most dot matrix and laser printers, the thermal transfer printers discussed have scalable text fonts and bar code fonts resident in the printer's firmware. The software necessary to print the bar codes is a series of special command sequences. However, most users want a general purpose design labeling programme which requires no programming.

5.5.7.4 Dot Matrix Printing

Dot matrix printers can produce good quality low volume bar code labels. When printing low-medium (3.7 cpi or lower for Code 39), the labels can be excellent quality. The Epson, IBM, and Okidata printers have adequate graphics capability to yield good quality bar codes.

Both 24-pin and 9-pin printers can produce good quality bar codes. The 24-pin printers produce better bar codes than 9-pin printers, especially as the ribbon is getting low on ink. The 24-pins simply put more ink on the paper. The complete process is quite slow for any practical purpose.

5.5.8 Labeling Software

Because dot matrix, Inkjet and Laser printers are in such widespread use, labeling software to make these printers capable of printing bar codes has become readily available. There are two general types of bar code printing programmes available :

- Menu-driven programmes for operators to design and print labels.
- Bar code font programmes to allow printing of bar codes within other Macintosh or Windows programmes; no programming is necessary by the user.

Now a day, most of the integrated library management programme incorporate sufficiently powerful barcode label printing utility.

5.6 References and Further Readings

- 1 2005 Request for Proposal for a Client/Server Electronic Library System (<http://www.libraryhg.com/rfp.doc>). Visited last : 12/08/2005.
- 2 2005 Barcode introduction. (http://www5.com/in_barcode.php). Visited last : 4/10/2005.
- 3 2004 Haravu (LJ). Library automation : design, principles and practice. New Dlehi : Allied Publishers, 2004.
- 4 2004 Vault Information Services LLC. General symbology : background information. (<http://www.barcodeisland.com/syminfo.phtml>). 2004. Visited last : 21/10/2005
- 5 2004 Encyclopedia of library and information science. 2nd ed. NY : Marcel Dekker, 2004. 4v
- 6 2000 Retrospective conversion : guidelines for libraries (OCLC). (http://www.oclc.org/oclc/promo/6075_retr/6075ret1.htm). Visited last : 22/03/2000
- 7 1997 Cohn (John M), Kelsey (Ann L) and Fiels (Keith Michael). Planning for automation : a how-to-do-it manual for librarians. 2nd ed. New York : Neal-Schuman Publishers, Inc., 1997.
- 8 1997 Meghabghab (Dania Bilal). Automating media centers and small libraries : a microcomputer-based approach. Englewood : Libraries Unlimited, 1997.
- 9 1984 Genaway (DC). Integrated online library systems : principles, planning and implementation. NY : GK. Hall. 1984.
- 10 1983 Rao (I.K. Ravichandra). Library automation (DRTC Refresher seminar 14, DRTC, Bangalore, 1983)
- 11 1990 Encyclopedia of library and information science. Ed by Allen Kent and Harold Lancour. NY : Marcel Dekker. 1969_. Various volumes

5.7 Exercise

1. Define retrospective conversion of library catalogues, and identify its benefits.
2. Briefly describe different phases of retrospective conversion.
3. What is barcode? Describe its applications.

Unit 6 □ Data Models

Structure

6.0 Objectives

6.1 Introduction

6.2 Evolution of Data Management Systems

6.2.1 Early Data Management Systems

6.2.2 Database Management System

6.3 Data Modeling

6.3.1 Hierarchical Model

6.3.2 Network Model

6.3.3 Relational Model

6.3.4 Entity-Attribute-Value (EAV) data model

6.3.5 Object-oriented Model

6.4 Normalization

6.5 Exercise

6.0 Objectives

The objectives of the Unit are to :

- Understand early data management systems
- Understand database approach for management of data system
- Examine the concepts of data models
- Discuss the basics of normalization

6.1 Introduction

Data is an important resource for any organization and applications. A database can loosely be defined as a collection of data, which exists for providing information. The data together with the system, which manage it, termed a database system. It is important to differentiate a database from a database system. There are many types of database management system-IRS (information retrieval system) and bibliographic database system etc.

6.2 Evolution of Data Management Systems

6.2.1 Early Computer based Information System

The objective of any computer-based information handling system is the acquisition and processing of data so that it can easily be placed into meaningful context. The early information management programmes were developed independently without adequate consideration for integrated acquisition, processing and reporting. In early phases of library automation, many institutes developed separate information processing systems for Acquisition, Cataloguing, Serials Management, and Current Awareness Services etc. Each separate application had its own master file, input data, and application programmes.

This lack of integration of different application programmes developed for management of large volume of information gave rise to various problems such as :

- Data required for one application may have already been supplied partly or fully for some other applications, such as input of author name in Acquisition system and Cataloguing System.
- Wastage of storage space by storing same data in multiple files for different application programmes
- Wastage of processing time
- Occurrence of inconsistencies and other errors in data files. While updating may take place in one file (Author name in Catalogue Master File), the same data stored elsewhere may fail to be updated (in Acquisition File/Author-Authority File), thereby causing data inconsistency.
- Independent development of information management programmes for different applications creates difficulties in integrating information.
- The application programmes were developed and data files created along functional boundaries. The unique key data items were not logically related and could not be used for cross-reference purposes during processing and retrieval of information. This seriously limited the information reporting capabilities and generation of meaningful output by the systems.

6.2.2 Database Management System

Database Management System plays an important role to avoid problems associated with early information management system and to readily achieve integration of data. Currently, database is the most important component of any computer based information system. A database may be defined as collections of inter related relevant data stored together to serve multiple applications. A database management system

(DBMS) is collection of software/programmes for processing the database. The data is stored in the database so that it is independent of the programmes using it.

6.2.2.2 Advantages

Some of the advantages that accrue from having an integrated/centralized database are :

- Redundancy can be reduced
- Inconsistency can be avoided/reduced
- Data can be shared
- Standard can be enforced
- Security can be enhanced
- Integrity can be maintained
- Conflicting requirements can be balanced

6.2.2.3 Bibliographic Database System

Bibliographic databases programmes combine many of the characteristics of structured database programmes with some of those associated with textual database programmes. In addition to sorting and selecting features, bibliographic database programmes offer features that are designed specifically to deal with the predominance of textual materials and rules for organizing citations/library catalogue that are associated with the academic and research environments.

- Data Entry forms.
- Field types for specific types of bibliographic information
- List fields or “Authority Lists”
- Note fields
- Word processor compatibility
- “Keys” in documents cite works in the database
- Emphasis on text in basic database functions
- Catalogue records formatting

Bibliographic database programmes have the ability to rearrange the information entered in the forms as citations. Citation formats the citation as you complete the form in a “Preview box” in the publishing style or format you have selected, and also allows you the option of generating formatted citations for all the works cited in a document (using “keys”).

- Data Entry forms : Unlike standard database programmes, bibliographic database programmes provide predefined data entry forms. These “forms” are

specifically designed to hold information for different bibliographic types of source works available in the library. Rather than entering information in formatted form for a book, for instance, one will enter the basic components for a book, along with keywords and a brief summary, in a form :

Author	Majumder, Arun Kumar Bhattacharyya, Pritimoy
Title	Database management system
Edition	1 st
Place	New Delhi
Publisher	Tata McGraw-Hill
Year of Publication	1996
Pages	475
ISBN	0-07-462239-0
Keywords	DBMS, Database Management

Bibliographic database offers a choice of forms that correspond to nearly every type of resource material available in the library. Unlike standard database programmes, bibliographic databases are able to use multiple forms in a single view of the database. And unlike the online database collections, records in a bibliographic database can be edited to include keywords and notes or abstracts that are important.

- Field types for specific types of bibliographic information : In addition to predefined forms, bibliographic database programmes have field “types” that are peculiar to certain kinds of bibliographic data, such as names, titles, keywords and notes or abstracts.
 - Names, for instance : author, editor, translators, and other names of contributors, are entered with first and last names inverted, and the names of individuals separated with semicolons : Majumder, Arun Kumar. This allows bibliographic database programmes to convert names entered into the database into any of the variety of special formats required for names. The ability to reformat names, titles, pages, and journal names for differing publishing styles is an essential characteristic of bibliographic database programmes.

- List fields or “Authority Lists” : In several fields (Keywords, Author, Publisher, Geographic Place Name), index will automatically alphabetize the terms that have been entered in the database, and display the list for reference. One can use the list to enter names and terms consistently, or to search for matching items in the database.
- Note fields : In most bibliographic database programmes, every record has a “Note” or abstract field that expands as you type notes, and lets you use the keystrokes you are used to using in your word processor for entering paragraphs.
- Word processor compatibility : Most bibliographic database programmes assume that the information entered into the database will be processed using word processor, are thus designed with a tight integration to the popular word processing systems.
- “Keys” in documents cite works in the database.
- Emphasis on text in basic database functions : Database programmes allow you to manipulate, search and sort information in a variety of ways.
- Display formatting : Bibliographic database programmes have the ability to rearrange the information entered in the forms at the time of displaying.
- Field Length : It is extremely difficult to predict/fix the field length in bibliographic database. A bibliographic database enables variable field length
- Repeatable Field : Multiple occurrence of data element in a field is very natural in case of bibliographic database. A document may have more than one author; subject index may assign more than one subject headings to a given document. It is also not an efficient solution that one will fix the maximum occurrence of a given field.
- Data retrieval : Searching, sorting, and subset selection enhancements emphasize the fact that bibliographic information is, primarily, textual data. So, for instance, the sorting feature allows you to alphabetize references with a “bibliographic sort”-ignoring leading articles in titles, and the selection feature allows you to retrieve a subset of records containing a particular name or keyword.
- Data output : Ability to output data from the fields and records in the database in different arrangements. The principle difference between bibliographic database programmes and traditional database programmes is in the ability to output or rearrange the information in bibliographic records.

6.3 Data Modeling

The goal of the data model is to make sure that all data objects required by the application functions are completely and accurately represented. Because the data model uses easily understood notations and natural language, it can be reviewed and verified as correct by the end-users. The data model is also detailed enough to be used by database developers as a 'blueprint' for building the physical database. The information contained in the data model will be used to define relational tables, primary and foreign keys, stored procedures, and triggers. A poorly designed database will require more time in the long run.

There are two categories models :

- **Implementation Model** : It is concerned with how the data are represented in the database. Provides concepts that can be understood by
 - End users
 - Computer specialist
 - Hides some details of data storage
 - Can be implemented on a computer system in a direct way
 - Used in current commercial DBMS
 - ◆ Relational
 - ◆ Network
 - ◆ Hierarchical
 - Represents data using record structure (record-based)
- **Conceptual Model** : It is concerned with what is represented in the database. It provides concepts closer to the way many users perceive data. Uses concepts such as entities, attributes, and relationships
 - Entity : a real world object or concepts (e.g. Project)
 - Attribute : properties that describes objects (e.g., Project_Name)
 - Relationships : an interaction or links among entities (e.g., works-on.)
 - ◆ Entity-Attribute-Value (EAV) data model
 - ◆ Object-oriented Data Model

6.3.1 Hierarchical Model

The hierarchical data model organizes data in a tree structure. There is a hierarchy of parent and child data segments. This structure implies that a record can have

repeating information, generally in the child data segments. It collects all the instances of a specific record together as a record type. These record types are the equivalent of tables in the relational model, and with the individual records being the equivalent of rows. To create links between these record types, the hierarchical model uses Parent Child Relationships. These are a 1 : N mapping between record types. For example, an organization might store information about an employee, such as name, employee number, department, salary. The organization might also store information about an employee's children, such as name and date of birth. The employee and children data forms a hierarchy, where the employee data represents the parent segment and the children data represents the child segment. If an employee has three children then there would be three child segments associated with one employee segment. In a hierarchical database the parent-child relationship is one to many. This restricts a child segment to having only one parent segment. Hierarchical DBMSs were popular from the late 1960s, with the introduction of IBM's Information Management System (IMS) DBMS, through the 1970s.

Advantages

- It promotes data security
- It promotes data independence
- It promotes data integrity (parent/child relationship)
- Useful for large databases
- Useful when users require a lot of transactions which are fixed over time
- Suitable for large storage media

Disadvantages

- Requires knowledge of the physical level of data storage.
- Cannot handle the case where a part may belong to two or more components
- New relations or nodes result in complex system management tasks.
- Programmers must be familiar with the appropriate navigation
- Modification to data structure lead to significant modifications to application programmes
- Does not provide the favoured ad-hoc query capability easily.
- No specific or precise standard

6.3.2 Network Model

The popularity of the network data model coincided with the popularity of the hierarchical data model. The network model permitted the modeling of many-to-many relationships in data. In 1971, the Conference on Data Systems Languages (CODASYL) formally defined the network model as the set construct. A set consists of an owner record type, a set name, and a member record type. A member record type can have that role in more than one set; hence the multi-parent concept is supported. An owner record type can also be a member or owner in another set. The data model is a simple network, and link and intersection record types (called junction records by IDMS) may exist, as well as sets between them. Thus, the complete network of relationships is represented by several pair wise sets; in each set some (one) record type is owner (at the tail of the network arrow) and one or more record types are members (at the head of the relationship arrow). Usually, a set defines a 1 : M relationship, although 1 : 1 is permitted. The CODASYL network model is based on mathematical set theory.

Advantages

- Improves on hierarchical model
- An application can access an owner record and all the member records in the set.
- The movement from one owner to another is eased.
- Promotes data integrity because of the required owner-member relationship

Disadvantages

- Difficult to design and use properly
- The user and the programmer must be familiar with the data structure.
- Does not promote structural independence
- Navigational data access problems

6.3.3 Relational Model

RDBMS (relational database management system) is a database based on the relational model developed by E.F. Codd. A relational database allows the definition of data structures, storage and retrieval operations and integrity constraints. In such a database the data and relations between them are organized in tables. A table is a collection of records and each record in a table contains the same fields. Properties of Relational Tables are :

- Values are Atomic
- Each Row is Unique
- Column Values are of the same kind

- The sequence of columns is insignificant
- The sequence of rows is insignificant
- Each column has a unique name

Certain fields may be designated as keys, which mean that searches for specific values of that field will use indexing to speed them up. Where fields in two different tables take values from the same set, a join operation can be performed to select related records in the two tables by matching values in those fields. Often, but not always, the fields will have the same name in both tables. For example, an “orders” table might contain (customer-ID, product-code) pairs and a “products” table might contain (product-code, price) pairs so to calculate a given customer’s bill you would sum the prices of all products ordered by that customer by joining on the product-code fields of the two tables. This can be extended to joining multiple table on multiple fields. Because these relationships are only specified at retrieval time, relational databases are classed as dynamic database management system. The relational database model is based on the Relational Algebra.

Advantages

- Structural independence : i.e. can concentrate on the logical view.
- Data independence
- SQL capability

Disadvantages

- More hardware and operating system overhead : i.e. RDBMS may be slower.
- Ease of use can be a liability : i.e. possible misuse.

Dr. E.F. Codd provided 12 rules that define the basic characteristics of a relational database but implementation of these rules varies from vendor to vendor. In practice, many database products are considered ‘relational’ even if they do not strictly adhere to all 12 rules. A summary of Dr. Codd’s 12 rules is presented below :

- **Codd’s Rule #1. Data is presented in tables :** A set of related tables forms a database and all data is represented as tables. A *table* is a logical grouping of related data in tabular form (rows and columns)
 - Each *row* describes an item (person, place or thing) and each row contains information about a single item in the table
 - Each *column* describes a single characteristic about an item
 - Each *value* (datum) is defined by the intersection of a row and column
 - Data is *atomic*; there is no more than one value associated with the intersection of a row and column

- There is *no hierarchical ranking of tables*
- The relationships among tables are logical; there are *no physical relationships among tables*
- Codd's Rule #2. **Data is logically accessible** : A relational database does not reference data by physical location; there is no such thing as the 'fifth row in the customers table' *Each piece of data must be logically accessible by referencing 1) a table; 2) a primary or unique key value; and 3) a column*
- Codd's Rule #3. **Nulls are treated uniformly as unknown** : *Null* must always be interpreted as an *unknown value*. Null means no value has been entered; the value is not known. 'Unknown' is *not* the same thing as an empty string ("") or zero
- Codd's Rule #4. **Database is self-describing** : In addition to user data, a relational database contains data about itself. There are two types of tables in a RDBMS : *user tables* that contain the 'working' data and *system tables* contain data about the database structure.
- Codd's Rule #5. **A single language is used to communicate with the database management system** : There must be a single language that handles all communication with the database management system.
- Codd's Rule #6. **Provides alternatives for viewing data** : A relational database must not be limited to source tables when presenting data to the user. *Views* are *Virtual tables* or abstractions of the source tables. A view is an alternative way of looking at data from one or more tables. A view definition does not duplicate data; a view is not a copy of the data in the source tables. Once created, a view can be manipulated in the same way as a source table.
- Codd's Rule #7. **Supports set-based or relational operations** : Rows are treated as *sets* for data manipulation operations (SELECT, INSERT, UPDATE, DELETE).

A relational database must support *basic relational algebra operations* (selection, projection; & join) and *set operations* (union, intersection, division, and difference).

Set operations and relational algebra are used to operate on 'relations' (tables) to produce other relations.

A database that supports only row-at-a-time (navigational) operations does not meet this requirement and is not considered 'relational'.

- Codd's Rule **#8. Physical data independence** : Applications that access data in a relational database must be unaffected by changes in the way the data is physically stored (i.e., the physical structure).

An application that accesses data in a relational database contains only a basic definition of the data (data type and length); it does not need to know how the data is physically stored or accessed

- Codd's Rule **#9. Logical data independence** : Logical independence means the *relationships among tables* can change without impairing the function of applications and adhoc queries. The database schema or structure of tables and relationships (logical) can change without having to re-create the database or the applications that use it
- Codd's Rule **#10. Data integrity is a function of the DBMS** : In order to be considered relational, data integrity must be an internal function of the *DBMS*; not the *application programme*
- Codd's Rule **#11. Supports distributed operations** : Data in a relational database can be stored centrally or distributed. Users can join data from tables on different servers (distributed queries) and from other relational databases (heterogeneous queries). Data integrity must be maintained regardless of the number of copies of data and where it resides
- Codd's Rule **#12. Data integrity cannot be subverted** : The DBMS must prevent data from being modified by machine language intervention

6.3.4 Entity-Attribute-Value (EAV) data model

The best way to understand the rationale of EAV design is to understand row modeling (of which EAV is a generalized form). Consider a supermarket database that must manage thousands of products and brands, many of which have a transitory existence. Here, it is intuitively obvious that product names should not be hard-coded as names of columns in tables. Instead, one stores product descriptions in a Products table : purchases/sales of individual items are recorded in other tables as separate rows with a product ID referencing this table. Conceptually an EAV design involves a single table with three columns, an entity an attribute (such as species, which is actually a pointer into the metadata table) and a value for the attribute (e.g., rat). In EAV design, one row stores a single fact. In a conventional table that has one column per attribute, by contrast, one row stores a set of facts. EAV design is appropriate when the number of parameters that potentially apply to an entity is vastly more than those that actually apply to an individual entity.

6.3.5 Object-Oriented Model

Object DBMSs add database functionality to object programming languages. They bring much more than persistent storage of programming language objects. Object DBMSs extend the semantics of the C++, Smalltalk and Java object programming languages to provide full-featured database programming capability, while retaining native language compatibility. A major benefit of this approach is the unification of the application and database development into a seamless data model and language environment. As a result, 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 modest amount of additional effort.

In contrast to a relational DBMS where a complex data structure must be flattened out to fit into tables or joined together from those tables to form the in-memory structure, object DBMSs have no performance overhead to store or retrieve a web or hierarchy of interrelated objects. This one-to-one mapping of object programming language objects to database objects has two benefits over other storage approaches : it provides higher performance management of objects, and it enables better management of the complex interrelationships between objects. This makes object DBMSs better suited to support applications such as financial portfolio, risk analysis systems, telecommunications service applications, World Wide Web document structures, design and manufacturing systems, and hospital patient record systems, which have complex relationships between data.

6.4 Database Normalization Basics

Normalization is often brushed aside as a luxury that only academics have time for. However, knowing the principles of normalization and applying them to your daily database design tasks really isn't all that complicated and it could drastically improve the performance of the DBMS. The concept of normalization and a brief overview of the most common normal forms have been presented.

Basically, normalization is the process of efficiently organizing data in a database. There are two goals of the normalization process : eliminate redundant data (for example, storing the same data in more than one table) and ensure data dependencies (only storing related data in a table). Both of these reduce the amount of space a database consumes and ensure that data is logically stored.

The database community has developed a series of guidelines for ensuring that databases are normalized. These are referred to as normal forms and are numbered from one (the lowest form of normalization, referred to as first normal form or 1NF)

through five (fifth normal form or 5NF). In practical applications, one will often see 1NF, 2NF, and 3NF along with the occasional 4NF. Fifth normal form is very rarely seen.

The normal forms, it's important to point out that they are guidelines and guidelines only. Occasionally, it becomes necessary to stray from them to meet practical business requirements. However, when variations take place, it's extremely important to evaluate any possible ramifications they could have on the system and account for possible inconsistencies.

1. First normal form (1NF) sets the very basic rules for an organized database :
 - Eliminate duplicate columns from the same table.
 - Create separate tables for each group of related data and identify each row with a unique column or set of columns (the primary key).
2. Second normal form (2NF) further addresses the concept of removing duplicate data :
 - Meet all the requirements of the first normal form.
 - Remove subsets of data that apply to multiple rows of a table and place them in separate tables.
 - Create relationships between these new tables and their predecessors through the use of foreign keys.
3. Third normal form (3NF) goes one large step further :
 - Meet all the requirements of the second normal form.
 - Remove columns that are not dependent upon the primary key.
4. Finally, fourth normal form (4NF) has one additional requirement :
 - Meet all the requirements of the third normal form.
 - A relation is in 4NF if it has no multi-valued dependencies.
 - Remember, these normalization guidelines are cumulative. For a database to be in 2NF, it must first fulfill all the criteria of a 1NF database.

References and Further Reading List

- 1 2005 Data structure (http://en.wikipedia.org/wiki/Data_structure). sept 2005. Last visited :
- 2 2005 Chapple (Mike). Database normalization basics. (<http://databases.about.com/od/specificproducts/a/normalization.htm>). Visited last : 22/10/2005
- 3 2001 Codd's 12 rules. (http://www.itworld.com/nl/db_mgr/05072001/). Visited last : 21/10/2005

- 4 2000 Database models. (http://www.frick-cpa.com/ss7/Theory_Models.asp#File). 2000. Visited last : 22/10/2005
- 5 1996 Majumder (Arun K) and Bhattacharyya (Pritimoy). Database management systems.. New Delhi : Tata McGraw-Hill, 1996.
- 6 1994 Elmasri (Ramez) and Navathe (Shamkant B). Fundamentals of database systems. California : Benjamin/Cummings, 1994.
- 7 1985 Data (CJ). An Introduction to database systems. 3rd ed. New Delhi : Narosa, 1985.
- 8 1983 Date (CJ). Database : a primer. Reading : Addison-Wesley, 1983.

6.5 Exercise

1. Describe evolution of data management systems.
2. Briefly describe different data models.
3. Discuss basics of database normalization.
4. Discuss 12 rules that define the basic characteristics of a relational database.

Unit 7 □ Software System

Structure

- 7.0 Objectives**
- 7.1 Introduction**
- 7.2 General Purpose Database Packages**
 - 7.2.1 MS-SQL**
 - 7.2.2 PostgreSql**
 - 7.2.3 CDS/ISIS**
- 7.3 Integrated Library Management Software**
 - 7.3.1 LibSys**
 - 7.3.2 SOUL**
 - 7.3.4 Virtua**
- 7.4 Exercise**

7.0 Objectives

The objectives of the Unit are to discuss :

- General purpose database software
- Bibliographic database software
- Integrated library and information management software

7.1 Introduction

Software is computer programmes; instructions that cause the hardware—the machines—to do work. The two primary software categories are operating systems (system software), which control the workings of the computer, and application software, which addresses the multitude of tasks for which people use computers. System software thus handles such essential, but often invisible, chores as maintaining disk files and managing the screen, whereas application software performs word processing, database management, and the like. Different application software has been discussed in the following sections.

7.2 General Purpose database Packages

7.2.1 MS-SQL : SQL Server 2005

Microsoft SQL Server is a relational database management system produced by Microsoft. It supports a superset of Structured Query Language SQL, the most common database language. It is commonly used by businesses for small to medium sized databases, and in the past 5 years large enterprise databases, and competes with other relational database products for this market segment.

7.2.1.2 History

The code base for Microsoft SQL Server originated in Sybase SQL Server, and was Microsoft's entry to the enterprise-level database market, competing against Oracle, IBM, and Sybase. Microsoft, Sybase and Ashton-Tate teamed up to create and market the first version named SQL Server 4.2 for OS/2 (about 1989) which was essentially the same as Sybase SQL Server 4.0 on Unix, VMS, etc. Microsoft SQL Server v6.5 was the first version of SQL Server that was architected for NT and did not include any direction from Sybase.

About the time Windows NT was coming out, Sybase and Microsoft parted ways and pursued their own design and marketing schemes. Microsoft negotiated exclusive rights to all versions of SQL Server written for Microsoft operating systems. Later, Sybase changed the name of its product to Adaptive Server Enterprise to avoid confusion with Microsoft SQL Server. Until 1994 Microsoft's SQL Server carried three sybase copyright notices as an indication of its origin.

Several revisions have been done independently since with improvements for SQL Server. SQL Server 7.0 was the first true GUI based database server, and a variant of SQL Server 2000 was the first commercial database for the Intel IA64 architecture. During this time there was a rivalry between Microsoft and Oracle's servers for winning the market over enterprise customers.

The current version, Microsoft SQL Server 2000, was released in August of 2000. Microsoft is beta testing its successor, SQL Server 2005. This is scheduled to launch during the week of 7th November 2005, alongside Visual Studio 2005 and BizTalk Server 2006. The June CTP (Community Technology Preview) release is currently available for free download. Different versions for Windows are :

- 1992-SQL Server 4.2
- 1993-SQL Server 4.21 for Windows NT

- 1995-SQL Server 6.0, codenamed SQL95
- 1996-SQL Server 6.5, codenamed Hydra
- 1999-SQL Server 7.0, codenamed Sphinx
- 1999-SQL Server 7.0 OLAP, codenamed Plato
- 2000-SQL Server 2000 32-bit, codenamed Shiloh
- 2003-SQL Server 2000 64-bit, codenamed Liberty
- 2005-SQL Server 2005, codenamed Yukon (not released yet)

7.2.1.3 Description

MS SQL Server uses a variant of SQL called T-SQL, or *Transact-SQL*, a superset of SQL-92 (the ISO standard for SQL, certified in 1992). T-SQL mainly adds additional syntax for use in stored procedures, and affects the syntax of transactions support. (Note that SQL standards require (ACID) Atomic, Consistent, Isolated, Durable transactions.) MS SQL Server and Sybase/ASE both communicate over networks using an application-level protocol called Tabular Data Stream (TDS). The TDS protocol has also been implemented by the Free TDS project in order to allow more kinds of client applications to communicate with MS SQL Server and Sybase databases. MS SQL Server also supports Open Database Connectivity (ODBC).

A stripped-down version of Microsoft SQL Server known as **MSDE** (Microsoft SQL Server Desktop Engine) is distributed with products such as Visual Studio, Visual FoxPro, Microsoft Access, MS Web Matrix, and other products. MSDE has some restrictions : a limit of 2 GB databases, and it comes with no tools to administer it. It also has a workload governor which reduces its speed once you exceed 8 concurrent workloads on the engine.

Microsoft recently announced the successor to MSDE, dubbed **SQL Server Express**. Similar to MSDE, SQL express includes all the core functionality of SQL Server but places restrictions on the scale of databases. It will only utilize a single CPU, 1 GB of RAM, and imposes a maximum size of 4 GB on databases. SQL Express also doesn't include enterprise features such as Analysis Services, Reporting Services, Data Transformation Services and Notification Services. A beta version of SQL Server Express is now available for download. The following table presents important characteristics of the **SQL Server 2005** in a cryptic way.

Scalability and Performance

Feature	Express	Workgroup	Standard	Enterprise	Comments
Number of CPUs	1	2	4	No Limit	Includes support for multi-core processors
RAM	1 GB	3 GB	No Limit	No Limit	
64-bit support	Windows on Windows (WOW)	WOW			
Database Size	4 GB	No Limit	No Limit	No Limit	
Partitioning					Support for large-scale databases
Parallel Index Operation					Parallel processing of indexing operations
Indexed Views					Indexed view creation is supported in all editions. Indexed view matching by the query processor is only supported in Enterprise Edition.

Note : 1 Single REDO thread and the safety setting is always on, 2 Supports only two nodes, 4 Subscriber only

7.2.2 PostgreSQL

PostgreSQL is a sophisticated Object-Relational DBMS, supporting almost all SQL constructs, including sub-selects, transactions, and user-defined types and functions. It is the most advanced open-source database available anywhere. Commercial Support is also available. **PostgreSQL** is a free object-relational database server (database management system), released under the flexible BSD-style license. It offers an alternative to other open-source database systems (such as MySQL and Firebird), as well as to proprietary systems such as Oracle, Sybase, IBM's DB2 and Microsoft SQL Server. PostgreSQL's unusual-looking name gives some readers pause in trying to pronounce it, especially those who pronounce **SQL** as “sequel”. PostgreSQL's developers pronounce it “post-gress-Q-L”. It is also common to hear it abbreviated as simply “postgres.”

7.2.2.2 History

PostgreSQL has had a lengthy evolution, starting with the Ingres project as UC Berkeley. The project leader Michael Stonebraker had left Berkeley to commercialize Ingres in 1982, but eventually returned to academia. After returning to Berkeley in 1985, Stonebraker started a post-Ingres project to address the problems with contemporary database systems that had become increasingly clear during the early 1980s. The code bases of Postgres and Ingres started (and remain) completely separated.

The team released version 1 to a small number of users in June 1989, followed by version 2 with a re-written rules system in June 1990. 1991's version 3 rewrote the rules system again, but also added support for multiple storage managers and for an improved query engine. By 1993 a huge number of users existed and began to overwhelm the project with requests for support and features. After releasing a Version 4—primarily as a cleanup—the project ended.

Although the Postgres project had officially ended, the BSD license (under which Berkeley had released Postgres) enabled Open Source developers to obtain copies and to develop the system further. In 1994 two UC Berkeley graduate students, Andrew Yu and Jolly Chen, added a SQL language interpreter to replace the earlier Ingres-based QUEL system, creating Postgres95. The code was subsequently released to the web to find its own way in the world. 1996 saw a re-naming of the project in order to reflect the database's new SQL query language, Postgres95 became **PostgreSQL**.

The first PostgreSQL release formed version 6.0. Subsequently a group of database developers and volunteers from around the world, coordinating via the Internet, have maintained the software. Since version 6.0, many subsequent releases have appeared, and many improvements have occurred in the system; on January 19, 2005 version 8.0 became the current release.

7.2.2.3 Description

A cursory examination of PostgreSQL might suggest that the system resembles other database systems. PostgreSQL uses the SQL language to run queries on data. That data exists as a series of tables with foreign keys linking related data together. One might characterise the primary advantage of PostgreSQL over some of its competitors as programmability : PostgreSQL makes it much easier to build real-world applications using data taken from the database.

PostgreSQL allows the user to define new types based on the normal SQL types, allowing the database itself to understand complex data. For instance, you can define an address to consist of several strings for things like street number, city and country.

From that point on one can easily create tables containing all the fields needed to hold an address with a single line.

PostgreSQL also allows types to include inheritance, one of the major concepts in object-oriented programming. For instance, one could define a `post_code` type and then create `us_zip_code` and `canadian_postal_code` based on it. Addresses in the database could then take either `us_address` or `canadian_address` form, and specialized rules could validate the data in each case. In early versions of PostgreSQL, implementing new types required writing C extensions and compiling them into the database server; in version 7.4, it became much easier to create and use custom types via `CREATE DOMAIN`.

Most SQL systems allow users to write a *stored procedure*, a block of SQL code that other SQL statements can call. However SQL itself remains unsuitable as a programming language and SQL users can experience great difficulty in constructing complex logic. Worse, SQL itself does not support many of the most basic operations in a programming language, like branching and looping. In PostgreSQL, programmers can write such logic in any one of a considerable set of supported languages.

A built-in language called PL/pgSQL resembles Oracle's procedural language PL/SQL, and offers particular advantages when dealing with query-intensive procedures. Wrappers for popular scripting languages such as Perl, Python, Ruby, and tcl allow harnessing their strengths in string processing and in linking to extensive libraries of outside functions. Procedures requiring the high performance provided by compiling complex logic into machine code can utilise C or C++.

The programmer can insert the code into the server as a *function*, a small wrapper that makes the code resemble a stored procedure. In this way SQL code can call (for instance) C code and vice-versa.

7.2.2.4 Features

Some features of PostgreSQL rarely found in other relational databases include :

- User-defined types
- User-defined operators
- Availability of multiple stored procedure languages, including C, PL/Java, PL/Perl, plpHP, PL/Python, PL/R, PL/Ruby, PL/sh, PL/Tcl, and the native PL/pgSQL
- Support for IP address (including IPv6), CIDR block, and MAC address data types
- Table inheritance (although not fully implemented)

- Rules – a way of implementing server-side logic that allows the application developer to modify the “query tree” of an incoming query
- Concurrency managed via a Multi-Version Concurrency Control (MVCC) design, which largely eliminates the need for read locks, as MVCC allows queries initiated at different points in time to read the version consistent with that point in time
- Expressional indexes – indexes created on the value of an expression, rather than on a single column or set of columns
- Partial indexes – indexes created on a selected portion of the tuples in a table. These can save disk space and improve performance by excluding uninteresting portions of the table from being indexed.

7.2.3 CDS/ISIS

CDS/ISIS is a menu-driven generalized Information Storage and Retrieval system designed specifically for the computerized management of structured non-numerical databases. The name of the software, CDS/ISIS is derived from a Division of UNESCO, The Computer Documentation Service/Integrated Software Information Systems. One of the major advantages offered by the generalized design of the system is that CDS/ISIS is able to manipulate an unlimited number of databases, each of which may consist of completely different data elements. CDS/ISIS offers an integrated programming facility allowing the development of specialized applications and/or the functional extension of the software as originally provided.

7.2.3.1 System functions

The major functions provided by CDS/ISIS allow you to :

- Define databases containing the required data elements
- Enter new records into a given database
- Modify, correct or delete existing records
- Automatically build and maintain fast access files for each database in order to maximize retrieval speed
- Retrieve records by their contents, through a sophisticated search language
- Display the records or portions thereof according to your requirements
- Sort the records in any sequence desired
- Print partial or full catalogues and/or indexes
- Develop specialized applications using the CDS/ISIS integrated programming facility.

7.2.3.2 Hardware Requirements

The minimum and recommended hardware requirements for running CDS/ISIS are the following :

- CPU : 486 processor at 40 MHz (Pentium at 100 MHz or higher recommended)
- RAM : 8MB (16MB or more recommended)
- 1 floppy disk unit
- 1 hard disk (with at least 4Mb free)
- 1 VGA 640×460 colour (super VGA 800×600 or higher recommended)
- 1 printer (optional)
- Although CDS/ISIS is a Windows 3.1 based programme, it runs under Windows 95 and Windows NT without specific known problems.

7.2.3.3 CDS/ISIS-Product family

Some of the products available in the CDS/ISIS software family are :

- CDS/ISIS for DOS
- CDS/ISIS for Windows (all versions)
- CDS/ISIS for UNIX (character mode)
- JavalSIS, Client - Server Internet suite
- UNESCO/BIREME ISIS_DLL, programming tool
- BIREME WWWISIS/Ibiscus GENISIS
- WinIDIS, the interface to IDAMS

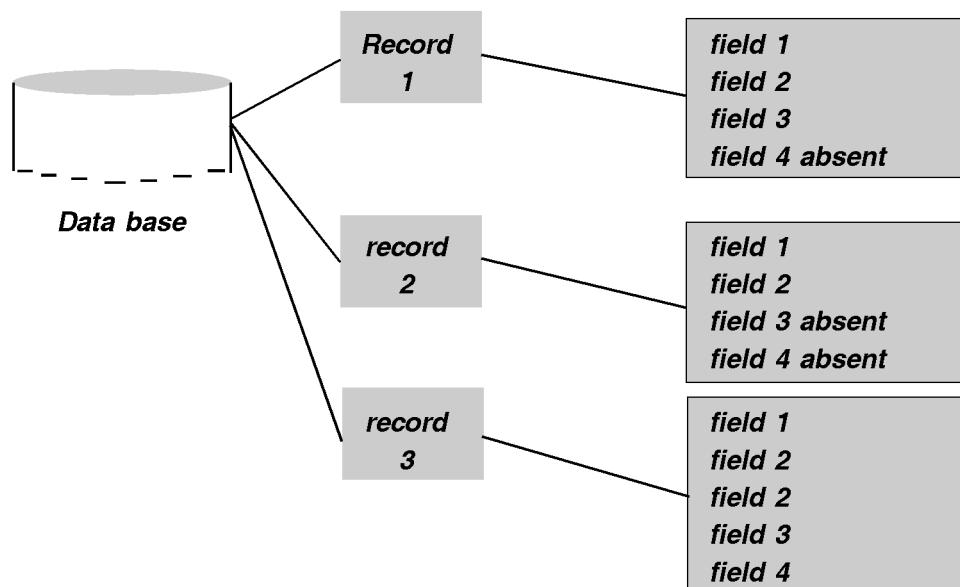
7.2.3.4 Characteristics

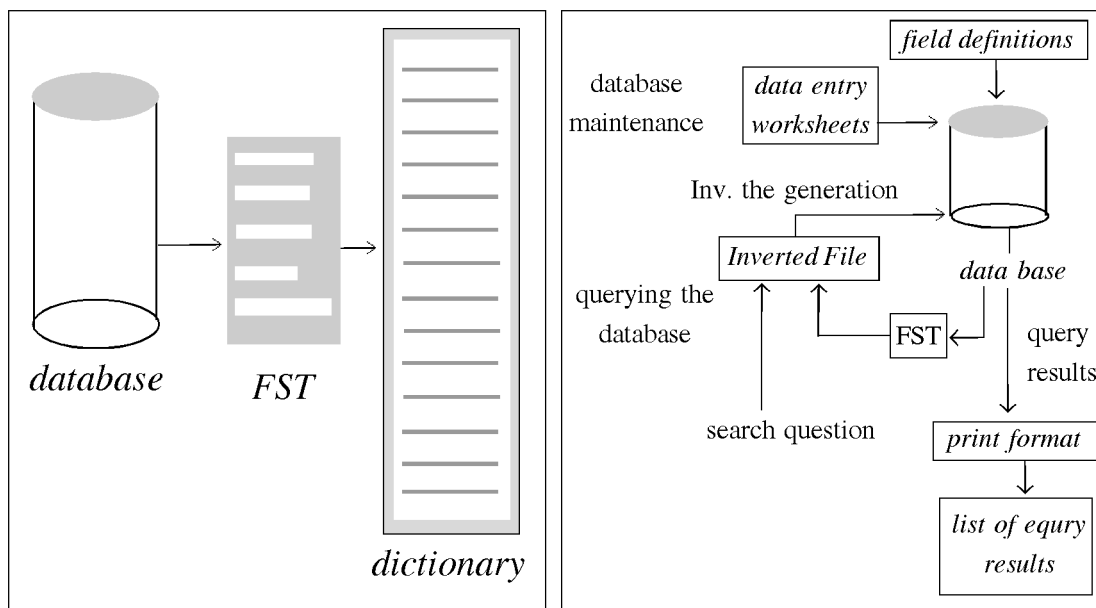
The salient characteristics of CDS/ISIS may be summarized as follows :

- CDS/ISIS is a generalized Information Storage and Retrieval System
- It is intended to be used for structured non-numerical databases containing mainly texts
- It may manage stand-alone as well as local network database systems
- It is specialized in handling variable-length
- Unlike Dbase or MS-Access, CDS/ISIS is not a relational database system, although it provides some relational facilities
- CDS/ISIS deals with questions like : “which research projects deal with basic education in India?”
- It may be expanded by advanced users (programmers) for adding new services and tools

- A wide variety of solutions for Internet publishing of data are already available
- CDS/ISIS allows the user to design the record structure of each database
- Once the database has been created, CDS/ISIS allows to :
 - Create new records, to put new information in the archive
 - Keep the stored information up-to-date by adding new data elements, modifying existing ones and deleting unnecessary information
 - Analyze the content of the database to extract searchable information following your own instruction
- Search and retrieve subsets of the stored information
- Display or print the results the way you want
- Records are stored in a file named “Master File” (MST)
- Each record in the database can be referred to by its unique number, called “Master File Number” (MFN)
- The MFN is automatically assigned by the system when a the record is created
- You may access a record not only by its number but its content. For this purpose, you may maintain a dictionary of searchable terms, also called “Inverted File”

The following figure shows the role of the different components of a CDS/ISIS database





7.3.4 LibSys : An Integrated Library System

LibStys (<http://www.libsys.co.in>) for library automation” is the prime mission of New Delhi based software company - Info-Tek Consultants Pvt. Ltd., engaged in providing software solutions for General Insurance and ERP/CRM since 1984.

7.2.4.1 Modules of LibSys

LybSys is an Integrated Library Management software covering almost all the functions of a library in the following areas :

- Acquisition/Ordering
- Cataloguing
- Circulation
- Serials Control
- Article Indexing
- OPAC
- WebOPAC

Built around its own centralized bibliographic database based on MARC standards, LibSys supports both print and non-print materials. LibSys has various formats to describe each material type, using different fields with variable field lengths. It facilitates import/export of data in standard exchange format such as MARC as well as other formats.

LibSys provides online validation of input data prior to updating the database. Prompts and suitable messages are flashed when the data does not match and 'look-up tables' provides help as and when required.

LibSys supports word based free text searching using Boolean operators. An efficient proprietary indexing feature substantially improves response time while searching for bibliographic information in large databases. The bibliographic database searching is integrated with the Circulation and Acquisition operations.

In circulation operations, either bar code or RFID technology may be employed for identification of both borrowers and materials. LibSys permits self-checkout, book drop etc. using LSmart interface.

LibSys implements security by allowing access through passwords. Security is provided for the user as well as for the library staff. For the staff, access can be controlled even at the function level.

The Java/perl based interface in LibSys Web OPAC provides an effective gateway to Internet and Intranet. The Z39.50 compliance in LibSys ensures keeping pace with the developments in international market place.

LibSys is a powerful software written in 'C/C++' and 'Java' providing user-friendly interface during operations. Although LibSys is based on its own Bibliographic database, it is also available for environment using ORACLE (or SQL Server or MySQL) as back-end RDBMS. Full Graphical User Interface (GUI) front-end is provided for the windows client.

LibSys supports various electronic resources thus making it a modern system with virtual library implementation. Image and multimedia files can be intergrated with LibSys search engine. They may be browsed in multi-windows interface with standard window features. Various formats handled by LibSys include Bitmap (.BMP), TIFF (.TIF), Wave (.WAV), Midi (.MID), Audio-visual Interface (.AVI), etc.

For its LibSys Network (LSNet) activities, LibSys is providing the solution of a Union Catalogue to its users. Software is currently available to link these Union catalogues on-line. This facility of LibSys makes networking of libraries effective and promotes the sharing of resources.

7.2.4.2 Libsys Features

- **Authority files** : Authority files are essential for error free cataloguing. It supports Authority files.
- **Auto Export and Import** : It allows automatic import or export in different bibliographic formats.

- **Automatic Bar Code Generation** : The Bar codes are generated from it itself. Normally one has to buy the software for bar coding separately.
- **Bar code Support for Issue and Return** : Bar codes generated by the system can be used to issue and return the material using the bar code reader. This increases the processing speed of your counters, saves waiting time for the members and reduces manpower as the same counters can handle the increased load.
- **Client Server Technology** : The client Server Technology ensures secure and distributed computing. This technology also prevents viruses from entering the system from the client side into the server. Also only the server needs to be protected. The clients (the users' computers) do not even know the location of the server and thus cannot harm the data.
- **Keyboard Mapping Support** : Keyboard Mapping allows the users to redefine the keyboard according to their wish. This is great in typing with keyboard layouts that one knows and prefers. This benefits the data entry department.
- **WebOPAC over Internet** : A full-fledged WebOPAC is available for the net users.
- **Password Protection for all modules and screens** : The system administrator has the powers to decide who is allowed to do what on the system. This ensures that although all terminals are able to work with LibSys in full-fledged capacity, only certain modules are accessible to certain users.
- **Retrospective Data Conversion Support** : A separate Data Entry option is also provided to do data entries.
- **Multi-lingual support for Indian and International languages/scripts** : Unicode support allows internationalization of the programme for all languages. It also supports ISM Publisher (C-DAC, India) and GIST (C-DAC, India).
- **Support for MARC standards** : It supports MARC21, UNIMARC, CCF etc.
- **Z39.50 Client** : Z39.50 standard of the Library of Congress and the ISO ensures that the library is connected to the libraries across the world and can access their databases.
- **E-mail interface** : Provided email interface for all reports
- **Support for e-Resources** : Support Electronic Resources-URL, Multi-media files (TIFF, GIF, JPEG, Wave, Midi, etc.), PDF file, and other file formats

- **Platform** : LibSys supports open system architecture, from host multi-user system to Client-Server implementation to a total web-based solution. So there are a number of operating platforms to choose from.
- **Server Options** : UNIX, LINUX, Pentium machine with SCO Unix/Unix Ware, SunSparc with SOLARIS, Alpha with OSF/1, RS/6000 with AIX, HP-9000 with HP-UX, SG with IRIX, Windows NT/2000/XP, and Windows 95/98/NT/2000/XP (Stand Alone).
- **Client Options** : Standard Windows (95/98/NT/2000/XP), Web Enabled (Applet-Servlet implementation), Unix Workstations, X-Windows Workstations (xterm), VT220 & compatible terminals, Thin Java clients.
- **RDBMS Options** : LibSys doesn't need an RDBMS as it is built around its own proprietary bibliographic database. However, the option of LibSys with SQL Server, ORACLE, MySQL as back-end RDBMS with odbc interface is also available.

7.2.5 SOUL

SOUL is the state-of the-art library automation software designed and developed by the INFLIBNET. It is user-friendly software developed to work under client server environment. Although looking at the name of the software, one may think that it is meant for university libraries only, but in fact it is flexible enough to be used for automating any type or size of library.

7.2.5.1 General Features

University libraries are complex entities, having large collections and serving a huge clientele. To carry out various operations in a library effectively, there is a need for automation. Computer and communication technologies have brought revolutionary changes in the information acquisition, processing, storage, retrieval and dissemination. Keeping in view the latest trends in Information Technology (IT), INFLIBNET Center has developed Windows based Library Management Software “SOUL”, which provides total solution for Library Automation. SOUL is designed using Client-Server Architecture, which imparts extra strength to storage capacity, multiple access to single database, various levels of security, backup, and storage facilities etc. This software has been designed after a comprehensive study of different library related functions practiced in university libraries. It has MS-SQL Server 7.0 or higher as the backend. This user-friendly software is quite easy to work with. SOUL handles Indian languages/scripts using ISM Publisher of C-DAC. There is an effort is going on to develop a new version of software based on MARC 21 and Unicode standards and RFID protocols for electronic surveillance. The software comprises following modules.

Acquisition

Catalogue

Circulation

OPAC

Serial Control

Administration

The in-built network of the software will allow multiple libraries of the same university to function together as well as access to the distributed databases installed at various university libraries and union catalogue mounted at INFLIBNET using VSAT network.

7.2.5.2 System Requirement

Network Version

The software can work with minimum hardware and software configuration given below, however higher or latest configuration may be purchased to have efficiency of the software.

System for Server :

- Pentium @933 MHz with 128 MB RAM
- 40 GB HDD
- 48xCD-ROM Drive
- 1.44" MB Floppy Drive
- Colour Monitor (SVGA)
- Ethernet card 10/100 Mbps
- Windows-NT/Windows 2000 server (Operating System)
- MS-SQL Server 7.0/Advance server 2000 (RDBMS)
- Personal Web Server or Windows NT IIS with option pack 4.0 installed OR Windows 2000 as server for Web OPAC

System for Client :

- Pentium @ 933 MHz with 64 MB RAM
- 2GB HDD with 10 MB Free space
- 1.44" MB Floppy Drive
- Colour Monitor (SVGA)
- Ethernet card 10/100 Mbps
- Windows-95/98/2000/XP/NT (Operating System)

College Version

- **System for Server/Client :**
- Pentium @ 933 MHz with 128 MB RAM
- 20 GB HDD with 200 MB Free space
- 48×CD-ROM Drive
- 1.44" Floppy Drive
- Colour Monitor (SVGA)
- Ethernet card 10/100 Mbps
- Windows-98/ME/NT/2000/XP/2003 (Operating System)

7.2.5.3 Standards supported by SOUL Software

SOUL is state of the art library management software widely used across India and neighboring countries. SOUL adheres to internationally acceptable standards like **AACR-2, CCF, MARC 21** and **ISO 2709**. Adoption of Standards in SOUL makes user's database globally acceptable and interchangeable.

7.2.6 Virtua

Virtua is a standards based, flexible, open, and Integrated Library System. With visionary features like FRBR (Functional Requirements for Bibliographic Records), User Reviews & Ratings, and a Smart Device interface to the catalogue. Virtua sets a new standard of excellence for the library world. Virtua offers integrated functionality between modules, including the system's cataloguing, acquisitions, serials, circulation and reporting functions. With fully integrated single-client approach, switching from one module to another is seamless as a keystroke or a click of the mouse.

Functional Requirements for Bibliographic Records (FRBR) is the next dimension in meaningful information—a fresh approach to traditional cataloguing that features a more intuitive way of organizing and retrieving information. With FRBR, user only have to search once to retrieve all related materials, even if those materials are catalogued in different languages or editions, or with different subject headings.

7.2.6.1 Salient Features

Certain important features of different versions of the software may be summarized as flows :

- The Functional Requirements for Bibliographic Records (FRBR) Model is an innovative alternative to traditional cataloguing—and Virtua is the first ILS on the market today that supports it. The advanced cataloguing/data-handling

standard of FRBR consolidates related information from disparate resources into a tree structure. The basic FRBR record consists of three entities—work, expression and manifestation. In addition, item records (holding records) can be attached to the manifestation. For example, Beethoven’s Fifth Symphony independently catalogued. The performance of the symphony by, for example, the New York Philharmonic Orchestra represents an “expression” of the work. A CD by Columbia Records containing the particular performance represents a “manifestation” of the work. Two copies of the CD in the library represent two “items” of this manifestation. FRBR Offers a more intuitive way of organizing and retrieving information, which translates to easier cataloguing for professional, and more complete search results for users. With FRBR, users only have to search once to find all related materials, even if those materials are catalogued in different languages or editions, or with different subject headings.

- Extensive policy matrices easily manage single or multiple-branch library requirements. It’s easy to set up an unlimited number of user profiles, item types, locations, and loan periods specific to each library branch. A comprehensive system of alerts and blocks provides flexibility when dealing with patron delinquencies.
- Create a “community” of readers with User Reviews and Ratings : A forum for patrons to write reviews and rank titles. The library has the option of approving a review before it’s posted.
- Virtua takes full advantage of the special strengths of Oracle TM, including its support of Unicode TM, backup and recovery capabilities, rollback features, and indexing capabilities. This makes for a very structured Virtua implementation, with better performance that is easier to install and manage.
- Virtua is the first ILS to fully comply with the Unicode TM standard on both the client and server levels. Users can view and print records in any language.
- Showcase your library’s unique identity. The flexibility of VECTORS allows you to change the entire face of the software. Patrons can view the catalogue through their preferred interface—whether that interface is designed for the advanced user, student, or child. News, weather feeds, blogs and other local information can be seamlessly included, making the library’s portal a central spot for all information—bringing your library users back to the Portal as their primary source for information.
- Give users a “rich” experience through content enrichment. Allow linking to book cover images, full electronic table of contents, published book reviews, video trailers for DVDs, and many other resources.

- Virtua complies with EDI, a timesaving technology standard that allows for electronic transmission of documents such as invoices, purchase orders, and claims. Virtua acquisitions include easy support for blanket orders, memberships and standing orders. For speedy management of serials, Virtua's automatic compression of the 863 serials tag saves you countless hours spent manually editing these tags.
- For speedy and accurate record creation, Virtua offers the following cataloguing shortcuts :
 - Batch processing for record importing
 - Keyboard commands for moving seamlessly between windows
 - Cut-and-paste and point-and-click capabilities decrease editing time and reduce data entry errors
 - Automatic validation of records at the client and server levels—validation that is based on MARC 21, but editable by the library for local practice
 - Online importing of any Z39.50-compliant MARC record
 - Ready-to-use work forms or record templates that you can customize
 - Simple global headings changes that instantly update every affected record
 - Support for multiple thesauri
 - Indexing tailored for the library's specific needs
- Fully-featured : Reserve Book Room handles multiple reserve locations with varying open and closed schedules, loan periods, and loan privileges. Items can be placed on reserve for several instructors and courses. Students have instant online access to course reserves by instructor name, course name, and course ID.
- Self-Check Units-Designed from the ground up to be robust yet attractive units, users self-check frees staff for other duties, taking on in some cases up to 85% of all circulation. Members can checkout multiple items accurately and quickly without assistance from the library staff. The self-check units also feature automatic offline circulation, and a Remote Patron Assistance Service.
- Allows libraries to significantly reduce their notice production costs by delivering overdue and hold pick-up notices by phone. Members are also able to renew their loan periods, automatically, over the phone, with no staff involvement, 24 hours per day.
- Materials Booking-Makes it easy for library staff and members to reserve library equipment, meeting rooms, and materials such as : conference rooms,

computers, DVD players, VCR's, overhead projectors, and course reserve materials. Members simply choose the items of interest, check against the calendar for availability, and book the time required.

References and Further Reading List

- 1 2005 <http://www.vtls.com>
- 2 2005 <http://libsys.co.in>
- 3 2005 What is the history of Microsoft windows (<http://kb.iu.edu/data/abwa.html>). 2005. Visited last : 22/10/2005
- 4 2005 SOUL : Software for university Libraries (<http://web.inflibnet.ac.in/info/soullInfo.jsp>). Visited last : 21/10/2005
- 5 2005 SQL Server 2005 Features Comparison (E:/NSOU/Microsoft SQL Server SQL Server 2005 Features Comparison.htm). Visited last : 21/10/2005
- 6 2005 PostgreSQL (<http://en.wikipedia.org/wiki/PostgreSQL>). Visited last : 21/10/2005
- 7 2005 Microsoft SQL Server (<http://en.wikipedia.org/wiki/MS-SQL>). Visited last : 21/10/2005
- 8 2001 Storti (Davide). CDS/ISIS : information storage and retrieval system. (http://szi-humanistika.ffzg.hr/isis/tutorial/tutorial_files/frame.htm). Visited last : 22/10/2005

7.4 Exercise

1. What is database software? Discuss general-purpose software in details.
2. Discuss the salient features of CDS/ISIS.
3. Describe with examples applications of different indexing techniques of CDS/ISIS.
4. Discuss any one library management software.

Unit 8 □ Online Public Access Catalogue (OPAC)

Structure

- 8.0 Objectives**
- 8.1 Introduction**
- 8.2 Functions**
- 8.3 Subsystems and their functions**
- 8.4 Technology**
- 8.5 Exercise**

1.0 Objectives

The objectives of the Unit are to :

- Identify basic components of an WEB-OPAC
- Enumerate Functions of WEB-OPAC
- Identify features of OPAC

8.1 Introduction

OPAC is a distinct module of an integrated library management system. It allows searching remotely by various access points, some of which are not possible in manual catalogue. The basic components are :

- **User Interface** : Allows users to interact with the system. The navigation between “hits” and across retrieved records using hyperlinks is normal in any web OPAC.
- **Database** : In an integrated library management system, all information about information resources is stored in databases. Authority files may be used for information retrieval.
- **IR Options** : Information Retrieval options include various searches : author, title, class number, subject, keywords, and location. It also facilitates to limit the search by certain parameters like language, year of publication, and type of information resources required etc.
- **Help** : Provides guide to user on how to search the database.

8.2 Functions

Remote access to library catalogue is not a new approach. Book catalogues, the forerunners of card catalogues, were an early form of remote access to library information resources. Like all library catalogues, major functions of OPAC may be summarized as follows :

- Locating information resources by known criteria (Author/Title)
- Identify documents on a given subject
- Facilitate resource sharing
- Help in collection development
- Facilitate copy cataloguing
- Facilitate literature search in 24 × 7 style
- Enable compilation of comprehensive bibliography depending on the need of the user
- Ascertain loan status of an information resource
- Access own account to verify number of information resources borrowed, dates of return and fine due etc.
- Access virtual resources
- Avail enhanced information retrieval facilities (Boolean/Hypertext) and display capabilities.
- Facilitate online reservation of information resources.
- Facilitate online registration for ILL and electronic delivery of digital information resources.
- Act as an interface of online reference service
- Allow user to personalize the interface.
- Enable user to make requests for procurement of information resources online.
- Enable user to check the status of his/her request for procurement(s)
- Facilitate simultaneous search of multiple database

8.3 Subsystems and their functions

The OPAC must provide remote and on-site search facility with an intuitive, easy-to-use Web interface. Desirable features of WEB-OPAC are :

8.3.1 Customization

The Library must be able to create custom web page and/or modify existing OPAC and OPAC Help web pages.

The system must use a single OPAC user login to present all resources and gateways to local databases that an individual is authorized to access. The OPAC must require that a user login to access the library's catalogue, at the Library's discretion. The OPAC module must also support generic or anonymous access to the Library's catalogue.

8.3.2 Searching

1. OPAC must provide keyword, phrase, and Boolean searches, as well as icons linked to Library-assigned search results lists, such as 'Community Organizations', 'Oprah Novels,' 'Holiday Cooking', or 'Science Fair Books'.

2. OPAC keyword searching must be available for every word in every bibliographic record, if desired by the Library. Relevance ranking must be supported. Indexes must be available for searching, including Author, Title, Subject, and Periodical Title indices.

3. The OPAC must enable users to limit (i.e., filter) searches by :
- Publication year (limits retrieval to titles published in, after, or before a specified date, or within a specified date range)
 - Language (limits retrieval to titles whose cataloguing information indicates that they were published in the language specified)
 - Item type (limits retrieval titles belonging to a specific material type out of the list of possible material types established by the Library, e.g., CDROM, DVD, video, reference book, periodical, etc.)
 - Item category (limits retrieval to titles belonging to the user's choice of two specific item categories out of the list of possible categories established by the Library, e.g., Fiction, Nonfiction, Mystery, Children, etc.)
 - Format (limits retrieval to titles in a specified broader material type defined by the Library e.g., one of the seven defined MARC formats).
 - Location (limits retrieval to titles in a specified permanent shelving location within the Library).
 - Library (limits retrieval to items owned by a specified library within a shared catalogue),
4. The OPAC must support broadcast Z39.50 searches of resources and databases.

5. OPAC must display Community Information records in MARC format, if these records are in use by the Library.

6. Community Information records must contain a free text format for data input and support links to scanned images, including map, meeting minutes, etc.

7. The OPAC must enable the user to display maps or graphical shelving plans created by the Library and stored as graphic images.

8. The system must have an icon-based OPAC for children with sets of photo icons, specialized bibliographies such as Suggested Summer Reading, custom pre-set searches designed by the library for standard topics such as science projects or crafts, and preconfigured searches of interest to children developed by library professionals.

8.3.3 User Services/Personalization

1. The OPAC must track an individual user's preferences and interests, organized into a list of "favorites" including, but not limited to, authors, subjects, library activities, reading groups, etc. These "favorites" must be included in a user's personal online account.

2. The system must provide user self-service options, or User Services, through the OPAC, including the ability for users to review the status of their accounts and to view custom displays of :

- Bills
- Items charged, with due dates and accrued fines
- Holds requested, with availability status
- Replies to their requests of library staff, with ability to replay, or cancel request
- Notes from library staff

3. OPAC user self-services must also include, at the Library's discretion :

- Renewing eligible items
- Placing holds on items
- Completing online, library-defined Request forms, e.g., ILL, Purchase Request, Suggestions, Self-registration, Reference Questions, etc.

4. OPAC must support Web-based materials bookings/item reservations. Describe your support for material booking by public users.

5. The OPAC module must automatically analyze the Library's overall circulation and display to lists of the Library's most popular titles, authors, or subjects. The OPAC module must update this information automatically.

6. The system must support MARC 856 fields in bibliographic records, so that OPAC users may click on the hyperlink (either actual URL or Library-substituted public note in subfield z of the MARC 856 field) to launch a linked resource, such as a website, digital image or audio file.

7. The Vendor must describe all digital media archive modules/products available. The digital media archive module must address the media capture, registration, search, retrieval, and administrative requirements of the Library.

8.3.4 Alternate Language Interfaces

Facilitate change of language of the user interface. Vendor must list alternate language interfaces available for its OPAC.

8.3.5 Content Enhancement/Enrichment

In addition to the standard OPAC, Vendor must offer OPAC content enrichment features that will provide users with images and information similar to online book vendors' sites, such as Amazon.com and barnesandnoble.com.

8.3.6 Broadcast Searching

1. Enable users to conduct simultaneous searches across both Z39.50 and non-Z39.50 targets including :

- Commercial abstract and index databases,
- Library catalogues, and
- Search engines (like Google, Teoma, etc.)

2. The broadcast searching option must return a unified search result for all such searches, regardless of the targets or protocols used to search or retrieve results from each source.

3. The broadcast searching option must :

- Speak to each source in its native language to provide superior results.
- Keep access to resources reliable with continuous updates through subscription-based Resource Plugins
- Deliver results merged into a single interface to enable users to limit searches, sort, and filter.

4. Vendor must describe its proposed broadcast searching option including protocols supported.

8.3.7 Context Management Solution

Vendor must offer a context management solution for collecting and organizing content from many different sources into easy-to-understand and easy-to-use context centers, or 'rooms' for its public users.

The context management solution proposed must be designed and built for use by libraries and library users, with library functions and library information in mind.

The context management solution proposed must provide tools for librarians to build and manage collections of resources in an electronic environment regardless of the origins of the resources.

Vendor's context management solution must enable the Library to organize and present resources including but not limited to :

- Traditional library collections (i.e. our OPAC and anyone else's)
- Digital media
- Online databases and publications
- The "free" Web
- Government and non-profit resources of all kinds

Within each 'room', resources must be presented seamlessly to the user. Content must be arranged according to topic, purpose, and/or audience. The context management solution must enable the Library to purchase completely crafted rooms from Vendor or create its own online context centers or 'rooms'. Vendor must offer all of the following options :

- Carefully chosen content organized into pregrouped collections, and covering a minimum of different subject areas specifically appropriate for libraries and built and maintained by a library subject expert.
- Local customization of 'rooms' based on off-the-shelf "blueprints" supplied by the Vendor
- User-friendly tools to enable original local creation of individual rooms

The context management solution proposed must provide plugins to manage identification, syntax, and protocol translation with remote resources. The context management solution proposed must provide authentication of resources that require some type of end user identification prior to use, such as a cookie, specific IP address, login, password, or certificate. Online context centers or "rooms" must be delivered through a context server that is responsible for managing user authentication and the navigation between different rooms.

No programming or other technical expertise must be required by Library staff to operate Vendor's context management solution. The context management solution proposed must incorporate seamlessly the Broadcast Searching and Open URL Resolver components specified above. (These components should also be available separately). When a public user enters a context center or 'room' and selects a resource :

- Resource must open in a new window,
- A frame must appear at the top of the new window with a ‘return to Library Room X’ option,
- The original context room must remain open behind the new window

Vendor must provide sample images of or links to context ‘room’ arrangements.

The context management solution must enable format options within each ‘room’ :

- Inline frames allowing another web site to be hosted within a ‘room’
- Streaming content for displaying syndicated news service feeds.
- XML-style-sheet based content modules to display linked ‘rooms’ content.
- HTML options allowing for inclusion of external HTML.

8.3.8 Open URL Resolution

Vendor must offer a fully functional Open URL Resolver compliant with current 0.1 Open URL specifications. (Indicate plans for support of future versions, especially the NISO 1.0 Open URL specification). The Open URL resolution feature must accept an incoming (source) Open URL and provide genre-specific resolution services.

Vendor’s Open URL Resolver must return and resolve, wherever available, information from :

- Full-text document databases
- Abstract and index databases
- Citation databases
- Content databases with review, tables of contents, first chapters, summaries, author biographies, etc.
- Online library catalogues—both local and remote
- Interlibrary loan and document delivery services
- Web sites
- Electronically accessible resources of all kinds free or licensed, to which the library has access.

Vendor’s Open URL Resolver must include a Web based administrative interface enabling the Library to configure and maintain the holdings and subscription data and determine what targets we want to resolve to for any particular information resource and service.

Vendor’s Open URL Resolver must permit gathering of statistics through standard webserver-style logs to provide management information on the types of link resolution being done by our users.

8.4 Technology

Initially, most OPACs were developed for LAN. Early LAN OPACs had character based menu driven interfaces. However, with the advancement of the Internet, OPACs first become available via telnet protocol. Today, most of the integrated library management systems contain web based OPAC module. The basic technologies are :

8.4.1 DHTML

Dynamic HTML or DHTML is a technique for creating interactive web sites by using a combination of the static markup language HTML, a client-side scripting language (such as JavaScript), the style definition language Cascading Style Sheets and the Document Object Model.

Some disadvantages of DHTML are that it is difficult to develop and debug due to varying degrees of support among web browsers and that the variety of screen sizes means the end look can only be fine-tuned on a limited number of browsers and screen-size combinations.

8.4.2 Common Gateway Interface (CGI) Programming

A CGI programme is any programme designed to accept and return data that conforms to the CGI specification. The programme could be written in any programming language, including C, Perl, Java etc. CGI programmes are the most common way for Web services to interact dynamically with users. The use of CGI programmes on the web server to deliver content dynamically is probably the most used method primarily because this approach is supported on almost all OS.

Major problem with CGI script is that each time a CGI script is executed, a new process is started. For busy websites, this can slow down the server. A more efficient solution is to use the server's API, such as ISAPI and NSAPI. However, Java servlets are becoming increasingly popular.

8.4.3 Dynamic Link Libraries (DLL)

A DLL is a library of executable functions or data that can be used by Windows applications. Typically, a DLL provides one or more particular functions and a programme access the functions by creating either a static or dynamic link to the DLL. A static link remains constant during programme execution while a dynamic link is created by a programme as needed. DLL files usually end with extensions. dll, .exe, .drv, or .fon.

A DLL can be used by several applications simultaneously. Some DLLs are provided with the Windows operating system and are available for any windows applications. Other DLLs are written for a particular application and are loaded with the particular application.

8.4.4 Java

Java is an object-oriented programming language introduced in 1995 by Sun Microsystems, Inc. Java facilitates the distribution of both data and small applications programmes, called applets, over the Internet. Java applications do not interact directly with a computer's central processing unit (CPU) or operating system and are therefore platform independent, meaning that they can run on any type of personal computer, workstation, or mainframe computer. With Java, software developers can write applications that will run on otherwise incompatible operating systems such as windows, the Macintosh operating system, OS/2, or UNIX.

To use a Java applet on the World Wide Web (www), a user must have a Java-compatible browser, such as Navigator from Netscape Communications Corporation, Internet Explorer from Microsoft Corporation, or HotJava from Sun Microsystems. A browser is a software programme that allows the user to view text, photographs, graphics, illustrations, and animations on the www. Java applets achieve platform independence through the use of a virtual machine, a special programme within the browser software that interprets the byte-code—the code that the applet is written in—for the computer's CPU. The virtual machine is able to translate the platform-independent byte-code into the platform-dependent machine code that a specific computer's CPU understands.

Applications written in Java are usually embedded in Web pages, or documents, and can be run by clicking on them with a mouse. When an applet is run from a Web page, a copy of the application programme is sent to the user's computer over the Internet and stored in the computer's main memory. The advantage of this method is that once an applet has been downloaded, it can be interacted with in real time by the user. This is in contrast to other programming languages used to write Web documents and interactive programmes, in which the document or programme is run from the server computer. The problem with running software from a server is that it generally cannot be run in real time due to limitations in network or modem bandwidth—the amount of data that can be transmitted in a certain amount of time.

8.4.5 Scripting Languages

Scripting languages (commonly called scripting programming languages or script languages) are computer programming languages initially designed for “scripting” the

operations of a computer. Early script languages were often called *batch languages* or *job control languages*. A script is more usually interpreted than compiled, but not always. The common properties are : they favor rapid development over efficiency of execution; they are often implemented with interpreters rather than compilers; and they are strong at communication with programme components written in other languages. Scripts are typically stored only in their plain text form (a ASCII) and interpreted, or compiled each time prior to being invoked.

Types of scripting languages

Application-specific languages

Many large application programmes include an idiomatic scripting language tailored to the needs of the application user. Likewise, many computer game systems use a custom scripting language to express the programmed actions of non-player characters and the game environment. Languages of this sort are designed for a single application and while they may superficially resemble a specific general-purpose language (e.g. QuakeC, modeled after C) they have custom features which distinguish them.

ACS

Action Script

AutoLISP

Text processing languages

The processing of text-based records is one of the oldest uses of scripting languages. Many, such as Unix's awk and, later, Perl, were originally designed to aid system administrators in automating tasks that involved Unix text-based configuration and log files. Perl is a special case—originally intended as a report-generation language, it has grown into a full-fledged applications language in its own right. PHP was originally developed as a specialized language for creating dynamic web content, but is now used by some for general system administration tasks as well.

Awk

Perl

Python

Ruby

PHP

Sed

XSLT

Job control languages and shells

Another class of scripting languages has grown out of the automation of job control-starting and controlling the behaviour of system programmes. Many of these languages' interpreters double as command-line interfaces, such as the Unix shell or the MS-DOS COMMAND.COM. Others, such as AppleScript, add scripting capability to computing environments lacking a command-line interface.

bash

csh

sh

General-purpose dynamic languages

Some languages, such as Perl, have begun as scripting languages but developed into programming languages suitable for broader purposes. Other similar languages-frequently interpreted, memory-managed, dynamic-have been described as "scripting languages" for these similarities, even if they are more commonly used for applications programming.

Cold Fusion

Perl

PHP

Python

Ruby

Smalltalk

Extension/embeddable languages

A small number of languages have been designed for the purpose of replacing application-specific scripting languages, by being embeddable in application programmes. The application programmer (working in C or another systems language) includes "hooks" where the scripting language can control the application. These languages serve the same purpose as application-specific extension languages, but with the advantage of allowing some transfer of skills from application to application.

JavaScript, JScript

Tcl (Tool command language)

References and Further Reading List

- 1 2005 Basics of library automation standards (<http://www.cde.state.co.us/cdelib/technology/atstan.htm>). Visited last : 20/09/2005

- 2 2005 Request for Proposal for a Client/Server Electronic Library System (<http://www.libraryhq.com/rfp.doc>). Visited last : 12/08/2005.
- 3 2004 Haravu (LJ). Library automation : design, principles and practice. New Delhi : Allied Publishers, 2004.
- 4 2004 Encyclopedia of library and information science. 2nd ed. NY : Marcel Dekker, 2004. 4v
- 5 1994 Harbour (Robin T). Managing library automation. London : ASLIB 1994
- 6 1994 Genaway (DC). Integrated online library systems : principles, planning and implementation. NY : GK. Hall. 1984.
- 7 1983 Rao (I. K. Ravichandra). Library automation (DRTC Refresher seminar 14, DRTC, Bangalore, 1983)
- 8 1969 Encyclopedia of library and information science. Ed by Allen Kent and Harold Lancour. NY : Marcel Dekker. 1969-. Various volumes

8.5 Exercise

1. What is OPAC.
2. Discuss functions and different subsystems of OPAC.
3. Describe concepts of DLL and CGI.

Unit 9 □ Database structure, Organization and Search

Structure

9.0 Objectives

9.1 Introduction

9.2 Database Architecture

9.2.1 Components of Database Management System

9.2.2 Database Design

9.2.3 Conceptual Design

9.2.4 Physical Database design

9.3 Database Organization

9.3.1 Types of database organization

9.3.2 Strategies for database organization

9.4 Database Design

9.5 Database Search

9.6 Exercise

9.0 Objectives

The objectives of the Unit are to :

- Explore concept of database architecture
- Enumerate different levels of database architecture
- Present an overview of components of a database system
- Explain database organization
- Discuss database search and advanced database search techniques

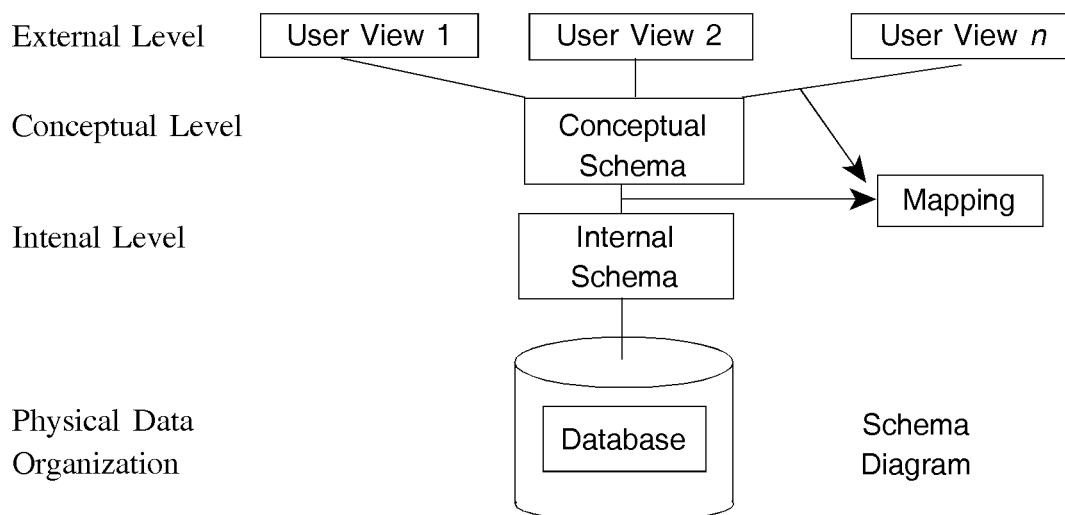
9.1 Introduction

The term database may be defined in several ways. However, the most meaningful way from the standpoint of the design of an information system is that it is a controlled collection of interrelated data. Implicit in this definition is the capability to define interrelationships among data, to minimize redundancy, to storage data in such a ways that can be retrieved to satisfy a variety of needs, and to modify the stored data as per the needs.

9.2 Database Architecture

It is necessary to understand certain key terms to appreciate the database architecture. The selected terms are :

- Database Schema (intension)
 - ◆ refer to a description of database
 - ◆ specify during database design
 - ◆ should not be changed
- schema diagrams
 - ◆ convention to display some aspect of a schema visually
- Schema construct
 - ◆ refers to each object in the schema (e.g. STUDENT)
- Database state (snapshot or extension)
 - ◆ refers to the actual data in the database at a specific time
 - ◆ changes any time we add or delete a record
 - ◆ valid state : the state that satisfies the structure and constraints specified in the schema and is enforced by DBMS
- To define a new database, we specify its database schema to the DBMS (database is empty)
- Database is initialized when we first load it with data

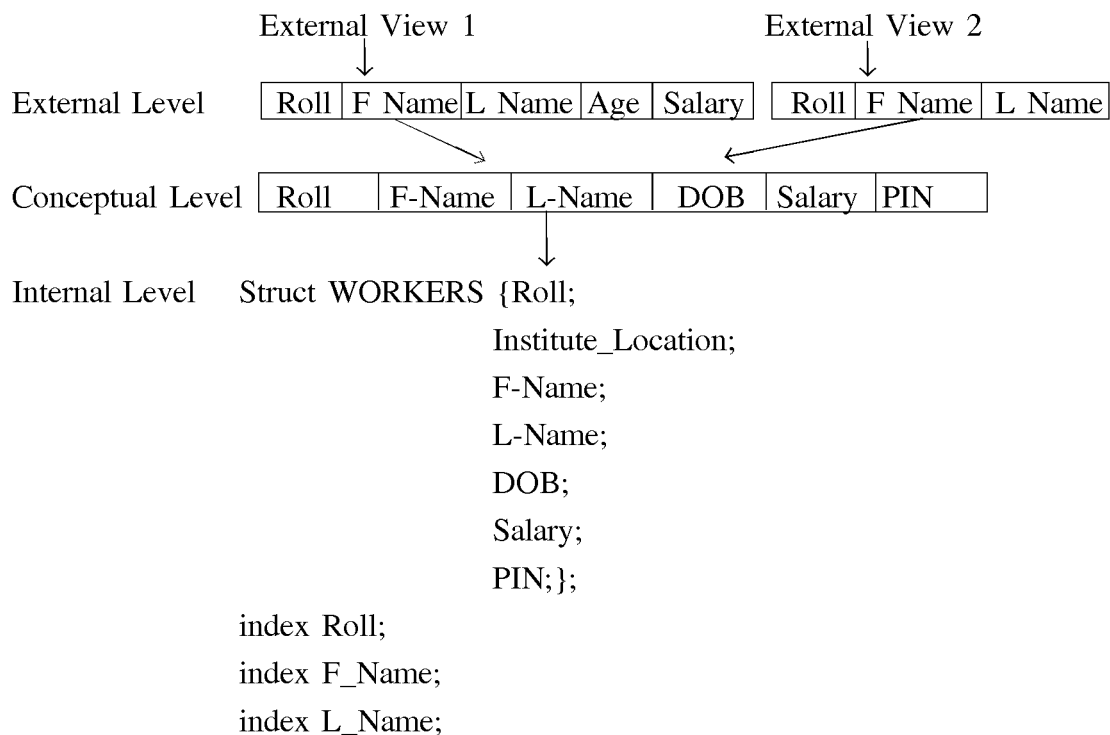


The salient objectives of the database systems are :

- Insulation of data from programme
- Support of multiple user views
- Use of catalogue to store the database schema

To ensure these basic characteristics of the database, the architecture of a database can be grouped into three separate levels of descriptions :

- **Conceptual** : It is the logical description of the database. This overall description is known as a Schema. Describes the structure of the entire database, hides the details of physical storage structures. Concentrates on the describing entities, data types, relationships, operations, and constraints. High-level data models or an implementation data model may be used here. It provides concepts closer to the way many users
 - Perceive data
 - Uses concepts such as entities, attributes, and relationships
 - ◆ Entity : a real world object or concepts
(e.g., Project)
 - ◆ Attribute : properties that describes objects
(e.g., Project_Name)
 - ◆ Relationships : an interaction or links among entities
(e.g., works-on.)
- **External** : Subsets of the schema that contain only the data needed for particular applications provide user views. The sub schemas provide a description at the external level. Includes a number of external schema or user views, each view describes subset of database needed by a particular user. High Level data model or an implementation data model can be used here.
- **Internal** : The description of the physical storage structures on a specific computer system is the internal description. It is concerned with the actual storage representation of data and its relationships. It describes structure of the database (data storage and access path)



Three Level Database architecture

This three level description is a convenient framework. Three-schema is used to implement data independence. This allows changes to be made at one level with minimum impact on the other levels. It ensures different types of data independence, they are :

- Logical data
 - Refers to the immunity of the external schemas to changes in the conceptual schema (e.g., add new record or field)
- Physical data
 - Refers to the immunity of the conceptual schema to changes in the internal schema (e.g., add new access path should not void existing ones)

9.2.1 Components of Database Management System

The basic components of database management systems are :

- Data Description Language (DDL) to translate the schema written in a source language into object schema, thereby creating a logical and physical layout for the database.
- Procedures for adding/deleting/modifying/retrieving etc. Data

- Query Language Translator (QLT) to translate queries written in a source language to a sequence of procedure calls for retrieving data from the database.
- System Software for file and buffer management, report generation, controlling concurrency, recovery from failure and to enforce security.
 - DBMS Interfaces :
 - Menu-based interfaces
 - Graphical Interfaces
 - Forms-based interfaces
 - Natural Languages
 - Interfaces for parametric users
 - Interfaces for DBA
 - Database System Environment/Tools
 - DBMS component Modules : E.g., Data Mgr, DDL Compiler, Query Compiler, etc
 - DBMS utilities : E.g., Backup
 - DBMS Communication Facilities : E.g., LAN
 - Case Tool : Rational Rose
 - Application development Environments : E.g., JBuilder

9.3 Database organization

Databases generally have one of two basic forms-the single-file database or the multi-file relational database. Single-file databases are often called “flat file” systems and relational databases are frequently known as “structured” databases. The type of database system or tools that require for an application depends on a number of factors, such as :

- The complexity of the data involved eg plain text, images, sound files
- The quantity of data to be stored and processed
- Whether the data needs to be accessed and amended by more than one person
- Whether data needs to be imported from or exported to the IT Systems

9.3.1 Types of Database Organization

The four major types of the database organization are :

- Flat
- Hierarchical
- Relational
- Object-oriented

9.3.1.1 Flat Database

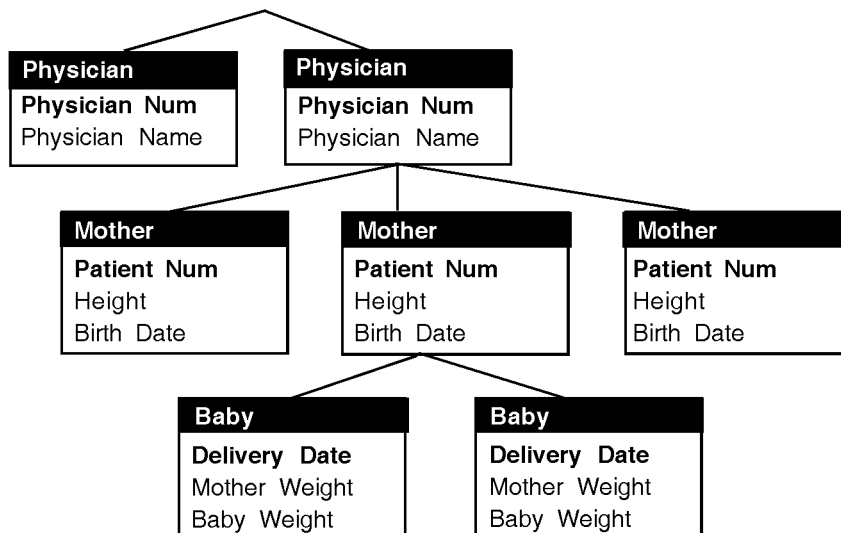
This database contains a single kind of record with a fixed number of fields. Notice the repetition of data, and thus an increased chance of errors.

Record No	Title	Publisher	Country	Phone
1	Serials Management	Elsevier	USA	789-896
2	Library Trends	Elsevier	USA	789-896
3	Granthagar	BLA	India	2442-6896

9.3.1.2 Hierarchical Database

This database depends on hierarchical relationships among different types of data. Points to be noted are :

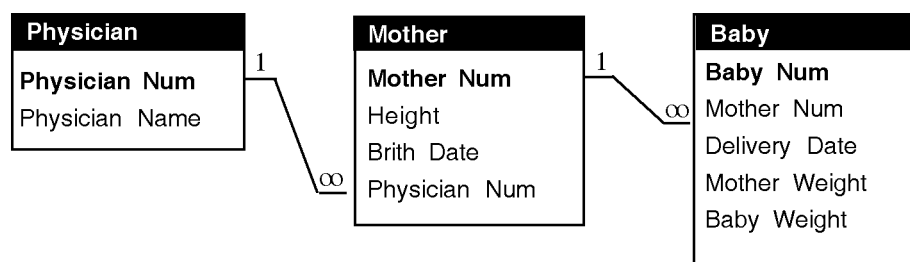
- Can be very easy to answer some questions, but very difficult to answer others
- If one-to-many relationship is violated (e.g., a patient can have more than one physician) then the hierarchy becomes a network



9.3.1.3 Relational Database

Data are organized as logically independent tables. Features of a relational database are :

- Natural
- Not so strongly biased towards specific questions
- Expresses relationships by means of shared data rather than explicit pointers
- Theoretical basis : relational algebra, calculus;
- Operations on tables (Join, Project, Select) to form new tables



9.3.11.4 Object-Oriented Database

Object-oriented analysis is another way to model the world, involving abstraction, encapsulation, modularity and hierarchy (with inheritance). An object consists of data and methods. Classes are used to group objects which have the same types of data and the same methods.

- Abstraction : Consider only features which are necessary for the problem at hand. For example, a person may be defined by the ability to ask for the ID number, age, height and weight
- Encapsulation : The internal structure of an object is hidden. For example, all we know is that we can ask for the age, not whether the age is (1) periodically updated or (2) calculated on demand from birth date and current date.
- Modularity : Grouping classes and objects into ‘cohesive and loosely coupled modules’. For example, the classes person and cat might be implemented within the same module because they share many things (like how to calculate age).
- Hierarchy : Objects are instances of classes, e.g., John is an instance of the class patient. Classes and objects form two different hierarchies.
 - Class hierarchy (‘kind’) : e.g., patient is a kind of person. A class inherits (and specializes) the characteristics of its parent class.
 - Object hierarchy (‘part’) : e.g., the object John. height is a part of the object John.

9.3.2 Strategies for Database Organization

Strategy	Reliability	Expandability	Communication	Manageabilities	Data Integrity
Centralized database resides in one location on host; data values may be distributed to geographically dispersed users for local processing.	Poor	Poor	Very High	Very Good	Excellent
Replicated					
Distributed snapshot databases Copy of portion of the central database created by extraction and replication for use at remote sites.	Good	Very Good	Low to Medium	Very Good	Medium
Replicated, distributed database. Data are replicated and synchronized at multiple sites.	Excellent	Very Good	Medium	Medium	Medium to very Good
Partitioned					
Distributed nonintegrated databases Independent databases that can be accessed by applications on remote computers.	Good				
Distributed, integrated database. Data span multiple computers and software.	Very Good	Very Good	Low to medium	Difficult to very Difficult	Very poor

9.4 Database Design

Database design is the process of developing a database structure from user requirements. Database design is a complex process since database is a shared system and it has to meet the requirements of different users within the accepted data integrity, data security and data privacy. Actual design depends on the particular DBMS being used. However, there are certain steps which are independent of any software.

9.4.1 Design Process

One of the best ways to understand database design is to start with an all-in-one, flat-file table design and add some sample data to see what happens. Identify problems caused by the initial design by analyzing the sample data, Modify the design to eliminate the problems, test some more sample data. check for problems, and re-modify, continuing this process until a consistent and problem free design developed. The major steps involved in developing a computer based application system may be summarized as follows :

9.4.1.1 Analysis Phase

This phase is software independent. The goal is to formulate precise system requirement definitions through the following activities :

- External system activities
 - Define information needs
 - Determine required inputs
 - Define data relationships
- Interface system activities
 - Define computer outputs
 - Define enquiry requirements
 - Identify input sources
 - Define data entry procedures
- Internal system activities
 - Specify data validation requirements
 - Define data protection requirements
 - Specify data validation rules

9.4.1.2 Database Design Phase

The requirements identified in the Analysis Phase have to be oriented according to the potential of the software. This phase involves :

- Defining data items
- Defining data entry form/worksheet
- Defining indexing techniques
- Defining display and output formats

9.4.1.3 Database creation and Maintenance Phase

This phase allows the creation and modification of records in the currently selected database. The various maintenance activities can also be performed.

The phase consists of :

- Data entry/editing
- Creation of index files
- Taking backup of the database

9.4.1.4 Functional Phase

The database application must support searching/information retrieval and production of different outputs on suitable media.

9.5 Database Search

The method used to store, find and retrieve the data from a database is called access method/information retrieval. A query language is used for retrieval of data from a database. The user specifies the requirements in terms of database query language syntax. The main emphasis is on the data is required and not on how it is located. Each DBMS contains procedures that allow a computer programme to access the data from the system. Certain general issues need to be considered at this level, they are :

- Access Control
- Semantics of data
- Variety of responses

9.5.1 Semantics of Data

Familiarity with the structure of records and fields will help to pinpoint specific information quickly. In certain cases, these details may not be available directly from the database schema, and if they are available the data definitions are not sufficient to understand meaning of the data items. To avoid this difficulty a data dictionary is usually required to support a database query language, and provide details of the data to be retrieved.

9.5.2 Variety of Responses

Ensuring search capabilities is not enough. Search system must take into account the variety of possible responses to a query. User expects some data to be retrieved against his/her query. However, under certain conditions, no data may be retrieved. Appropriate messages must be displayed under “no data” situations. This is certainly different from messages for different user and/or system error during execution of a query.

9.5.3 Search Techniques

There are various types of information database : bibliographic, numeric, textual and graphical. To be useful, databases have to be structured in some fairly definite ways. Such structure provides organization for massive quantities of data and allows rapid access to specific pieces of information. Each record, in turn, is made up of fields, which are specific categories of data about the item-the title of a book, the author of an article, the publisher (imprint), the publication date. Familiarity with the structure of records and fields will help to pinpoint specific information quickly. The following table presents an overview of certain advanced search techniques :

Type of Search	purpose	When to Use	Example
Boolean	Specifies multiple words in any field, in any order	You will want to use Boolean searches most of the time you are doing any keyword searches -unless your search term is very new and/or unusual, you will retrieve too much information	Tagore AND Rain (narrows search so that records must have both terms) Tagore OR Rain (Broadens search, records can have either Tagore or Rain)
Truncation	A truncation symbol tells the database to find any words that begin with the letters user typed, regardless of how the words end	Use truncation when you have a keyword term that has many similar forms which may bring up additional information related to your search topic	Truncation symbols vary depending which database you are using. <u>Griffin & Firstsearch</u> : medic* <u>ProQuest</u> : medic? If you are not sure what truncation symbol a database uses, look for the “search tips” or “help” link.
Proximity	Proximity operators allow you to locate one word within a certain distance of another.	use proximity operators when you are searching for keywords, which should appear very close together in a database record to adequately address your topic.	Proximity operators vary depending which database you are using. <u>Griffin</u> : america within 3 econom* <u>FirstSearch</u> : america w3 econom*

			ProQuest: america w/3 econom? If you are not sure what proximity operators a database uses, look for the "search tips" or "help" link.
Limiting / Narrowing	Nearly all databases will allow you to limit your search in some way. This is very helpful in weeding out resources are not useful to you for some reason.	You will generally want to put some limiting criteria on most of your searches just to save yourself the time of looking at less than-helpful resources.	Limiting options vary by database. Look for pull-down menus, check-boxes, etc. while you are searching

References and Further Reading List

- 1 2005 Data structure ([http://en.wikipedia.org/wiki/Data structure](http://en.wikipedia.org/wiki/Data_structure)). Sept 2005. Last visited : 23/10/2005
- 2 2005 Chapple (Mike). Database normalization basics. ([http://databases.about.com/od/sepecificproducts/a/normalization. htm](http://databases.about.com/od/sepecificproducts/a/normalization.htm)). visited last : 23/10/2005
- 3 2000 Database models. (http://www.frickepa.com/ss7/Theory_Models.asp#File). 2000. Visited last : 23/10/2005
- 4 1996 Majumder (Arun K) and Bhattacharyya (Pritimoy). Database management systems.. New Delhi : Tata McGraw-Hill, 1996.
- 5 1994 Elmasri (Ramez) and Navathe (Shamkant B). Fundamentals of database systems. California : Benjamin/Cummings, 1994.
- 6 1985 Date (CJ). An Introduction to database systems. 3rd ed. New Delhi : Narosa, 1985.
- 7 1983 Date (CJ). Database : a primer. Reading : Addison-Wesley, 1983.
- 8 1985 Caryln (Shamlin). A user's guide for defining software requirements. Wellesley : QED, 1985.

9.6 Exercise

1. What is database architecture? Define different important terms associated with the concept of database architecture.
2. Describe components or database management systems.
3. Discuss different types of database organization.

Unit 10 □ Operating System

Structure

10.0 Objectives

10.1 Operating System

10.1.0 Introduction

10.1.1 Definition

10.1.2 Evolution of Operating System

10.1.3 Classification of Operating System

10.1.4 Types of Operating Systems

10.1.5 Loading of an Operating Systems (Booting)

10.1.6 System Architecture

10.2 Single User Operating System

10.2.1 MS-DOS

10.2.2 MS-WINDOWS XP

10.3 Exercise

10.0 Objectives

The objectives of the Unit are to :

- Explain the concept of OS
- Trace the evolution of OS
- Discuss process of booting a computer
- Outline general aspects of MS-WINDOWS OS

10.1 Operating Systems

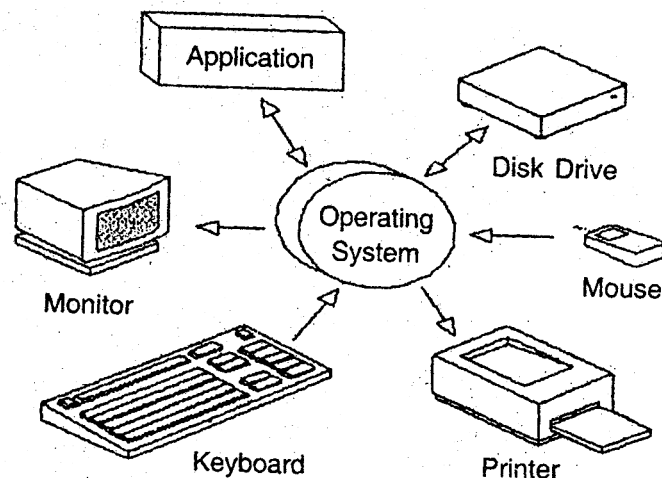
10.1.0 Introduction

Operating System (OS) is the basic software that controls a computer. There are many types of operating systems, the complexity of which varies depending upon what type of functions are provided, and what the system is being used for. Some systems are responsible for managing many users on a network. Other operating systems do

not manage user programmes at all. These are typically found in hardware devices like petrol pumps, airplanes, video recorders, washing machines and car engines.

10.1.1 Definition

It is the master control programme that runs the computer. The first programme loaded when the computer is turned on, its main part, the “kernel,” resides in memory at all times. Application programmes “talk to” the operating system for all user interfaces and file management.

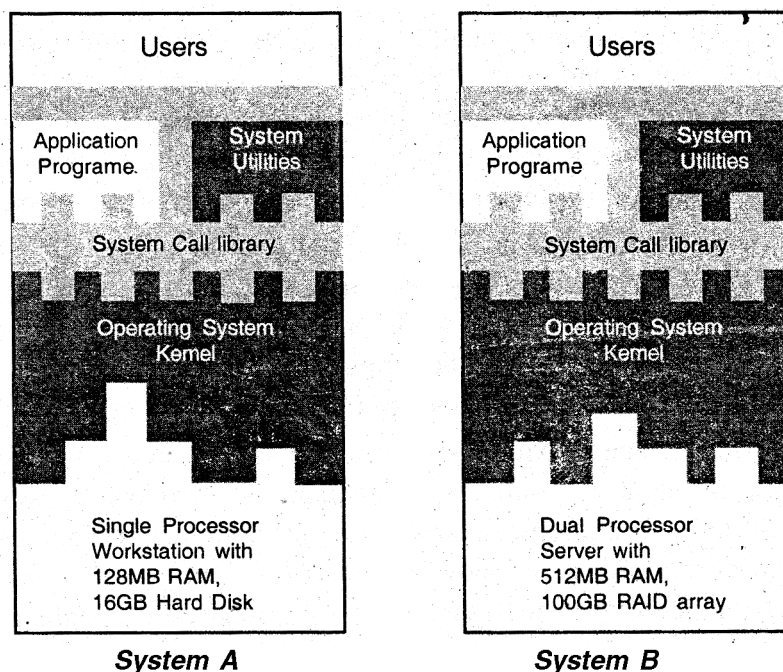


The operating system has six major functions :

1. User Interface : All arez graphics based today, the user interface includes the windows, menus and method of interaction between user and the computer. Prior to the Mac, Windows and Motif (Unix) interfaces, all interaction was based on commands entered by the user. Operating systems may support optional interfaces (different shells) for different functionality and appearance.
2. Job Management : Job management controls the order and time in which programmes are run and is more sophisticated in the mainframe environment where scheduling the daily work has always been routine. In a desktop environment, batch files can be written to perform a sequence of operations which can be scheduled to start at a given time.
3. Task Management : Multitasking, which is the ability to simultaneously execute multiple programmes, is available in all operating systems today. Critical in the mainframe and large server environment, applications can be prioritized to run faster or slower depending on their purpose. In the desktop world, multitasking is necessary just for keeping several applications open at the same time so user can bounce back and forth among them.

4. **Data Management** : Data management keeps track of the data on disk, tape and optical storage devices. The application programme deals with data by file name and a particular location within the file. The operating system's file system knows where that data are physically stored (which sectors on disk) and interaction between the application and operating system is through the programming interface. Whenever an application needs to read or write data, it makes a call to the operating system.
5. **Device Management** : Device management controls peripheral devices by sending those commands in their own proprietary language. The software routine that knows how to deal with each device is called a "driver." The operating system contains all the drivers for the peripherals attached to the computer. when a new peripheral is added, that device's driver is installed into the operating system.
6. **Security** : Multi-user operating systems provide password protection to keep unauthorized users out of the system. Large operating systems also maintain activity logs and accounting of the user's time for billing purposes. They also provide backup and recovery routines for starting over in the event of a system failure.

The following figure presents the architecture of a typical operating system and shows how an OS succeeds in presenting users and application programmes with a uniform interface without regard to the details of the underlying hardware.



10.1.2 Evolution of Operating System

Historically operating systems have been tightly related to the computer architecture, it is good idea to study the history of operating systems from the architecture of the computers on which they run. Operating systems have evolved through a number of distinct phases or generations which corresponds roughly to the decades.

First Generations (1940s)

The earliest electronic digital computers had no operating systems. Machines of the time were so primitive that programmes were often entered one bit at a time on rows of mechanical switches (plug boards). Programming languages were unknown (not even assembly languages). Operating systems were unheard of.

Second Generation (1950s)

By the early 1950's the routine had improved somewhat with the introduction of punch cards. The General Motors Research Laboratories implemented the first operating systems in early 1950's for their IBM 701. The system of the 50's generally ran one job at a time. These were called single-stream batch processing systems because programmes and data were submitted in groups or batches.

Third Generation (1960s)

The systems of the 1960's were also batch processing systems, but they were able to take better advantage of the computer's resources by running several jobs at once. So operating systems designers developed the concept of multiprogramming in which several jobs are in main memory at once; a processor is switched from job to job as needed to keep several jobs advancing while keeping the peripheral devices in use.

For example, on the system with no multiprogramming, when the current job paused to wait for other I/O operation to complete, the CPU simply sat idle until the I/O finished. The solution for this problem that evolved was to partition memory into several pieces. With a different job in each partition. While one job was waiting for I/O to complete, another job could be using the CPU.

Another major feature in third-generation operating system was the technique called spooling (simultaneous peripheral operations online). In spooling, a high speed device like a disk interposed between a running programme and a low-speed device involved with the programme in input/output. Instead of writing directly to a printer, for example outputs are written to the disk. Programmes can run to completion faster, and other programmes can be initiated sooner when the printer becomes available, the outputs may be printed.

Another feature present in this generation was time-sharing technique, a variant of multiprogramming technique, in which each user has an on-line (i.e., directly connected) terminal. Because the user is present and interacting with the computer, the computer system must respond quickly to user requests, otherwise user productivity could suffer. Timesharing systems were developed to multiprogramme large number of simultaneous interactive users.

Fourth Generation (1970s)

With the development of LSI (Large Scale Integration) circuits, chips, operating system entered in the personal computer and the workstation age. Microprocessor technology evolved to the point that it becomes possible to build desktop computers as powerful as the mainframes of the 1970s. Two operating systems have dominated the personal computer scene : MS-DOS, written by Microsoft, Inc. for the IBM PC and other machines using the Intel 8088 CPU and its successors, and UNIX, which is dominant on the large personal computers using the Motorola 6899 CPU family.

10.1.3 Classification of Operating Systems

Operating systems may be classified by both how many tasks they can perform ‘simultaneously’ and by how many users can be using the system ‘simultaneously’. That is : single-user or multi-user and single-task or multitasking. A multi-user system must clearly be multi-tasking. The table below shows some examples.

OS	Users	Tasks	Processors
MS/PC DOS	S	S	1
Windows 3x	S	QM	1
Macintosh System 7.*	S	QM	1
Windows9x	S	M*	1
UNIX/LINUX	M	M	n
NT	S/M	M	n
Windows 2000	M	M	n

The first of these (MS/PC DOS/Windows 3x) are single user, single-task systems which build on a ROM based library of basic functions called the BIOS. Only a single user application could be open at any time. Windows 95 replaced the old co-routine approach of quasi-multitasking with a true context switching approach, but only a single user system, without memory protection. Windows NT added a proper kernel with memory protection, based on the VMS system, originally written for the DEC/Vax.

Later versions of Windows NT and Windows 2000 (a security and kernel enhanced version of NT) allow multiple logins also through a terminal server. Windows 2000 thus has comparable functionality to Unix in this respect.

Windows 9x is purported to be preemptive multitasking but most programme crashes and also crash the entire system. This might be due to the lack of proper memory protection. The claim is somewhat confusing.

Unix is arguably the most important operating system today. It comes in many forms, developed by different manufacturers. Originally designed at AT & T, UNIX split into two camps early on : BSD (Berkeley software distribution) and system 5 (AT&T license). A standardization committee for Unix called POSIX, formed by the major vendors, attempts to bring compatibility to the Unix world.

10.1.4 Types of Operating Systems

Operating systems are divided into categories that define their characteristics. Modern systems may use combinations of those described below.

Batch	The earliest type, allowed only one programme to run at a time. The programme was entered into the computer, then run till completed. The data used by the programme could not be modified whilst the programme was running. Any errors in the programme or data mean starting all over again.
Inter-Active	These allow the modification and entry of data whilst the programme is running. Typical systems are airline reservations and languages such as BASIC.
Time-Sharing/Multi-user	These share the computer system amongst more than one user, and employ pre-emptive scheduling techniques.
Multi-Tasking	More than one process may be executed at once. The processor is switched rapidly between the processes. A user may run more than one process at a time.
Real-Time	Primarily used in process control, telecommunications, etc. The OS monitors various inputs which affect the execution of processes, changing the computers model of the environment, thus affecting the outputs, within a guaranteed time period (usually < 1 second).
Multi-Processor	A computer that has more than one processor dedicated to running processes.
EMBEDDED	An embedded operating system means the operating system is self-contained in the device and resident in ROM. Typical systems that use embedded operating systems are household appliances, car management systems, traffic control systems and energy management systems.

10.1.5 Loading of an Operating System (Booting)

The operating system may be loaded into the computers memory in two ways.

1. It is already present in ROM (so is permanent, immediately accessible and difficult to update)
2. It is loaded from disk when the computer is turned on.

If the operating system is already present in ROM (for systems like industrial controllers, petrol pumps etc), it will gain control immediately the processor is powered on. This method is best suited for small appliances and hand held devices where the operating system is relatively simple and small.

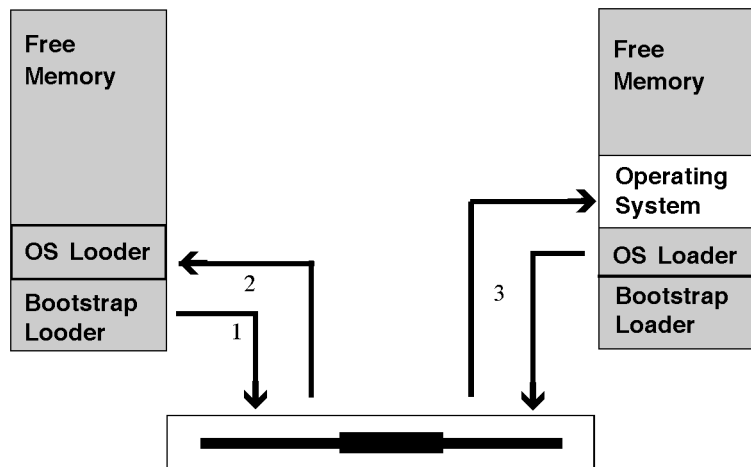
For more complex systems, the operating system is usually stored on secondary media (such as disk), and is loaded into the computer memory (RAM) when the computer is powered on. Advantages of this type of system are that changes to the operating system are easier to make and implement.

When user turn on the power to a computer, the first programme that runs is usually a set of instructions kept in the computer's read-only memory (ROM). This code examines the system hardware to make sure everything is functioning properly. This power-on self test (POST) checks the CPU, memory, and basic input-output systems (BIOS) for errors and stores the result in a special memory location. Once the POST has successfully completed, the software loaded in ROM (sometimes called the BIOS or firmware) will begin to activate the computer's disk drives. In most modern computers, when the computer activates the hard disk drive, it finds the first piece of the operating system : the bootstrap loader.

10.1.5.1 Bootstrap Process

The bootstrap loader is a small programme that has a single function. It loads the operating system into memory and allows it to begin operation. In the most basic form, the bootstrap loader sets up the small driver programmes that interface with and control the various hardware subsystems of the computer. It sets up the divisions of memory that hold the operating system, user information and applications. It establishes the data structures. Then it turns control of the computer over to the operating system.

The bootstrap process describes the task of initially loading the operating system from disk into RAM. A small routine stored in ROM, called the BOOTSTRAP LOADER or IPL (Initial Programme Loader), reads a special load routine from the diskette.



- 1: Request OS Loader routine from disk unit
- 2: Transfer OS loader into memory
- 3: OS loader loads rest of OS into memory

In floppy based system, this routine is normally located on Track 00, sector 00 (or 01), and is called the boot sector. The code contained in the sector is transferred into RAM, and then executed. It has the sole responsibility for loading the rest of the operating system into memory.

10.2 Single User Operating System : MS-WINDOWS

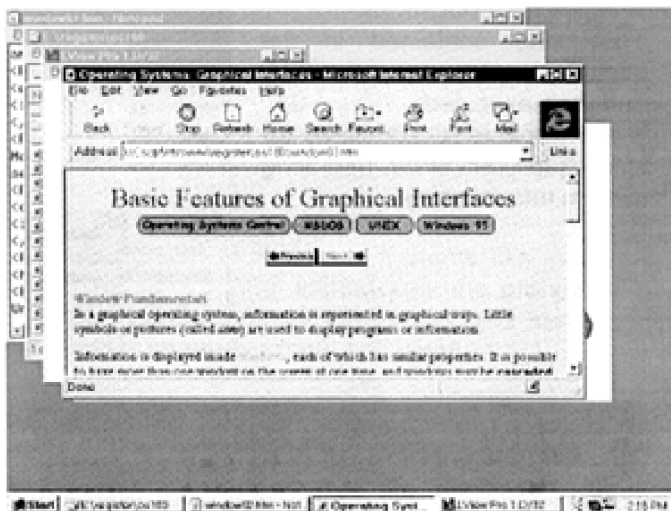
Operating systems such as Windows 95, Windows NT Workstation and Windows 2000 professional are essentially single user operating systems. They provide the capability to perform tasks on the computer system such as writing programmes and documents, printing and accessing files.

In essence, a single-user operating system provides access to the computer system by a single user at a time. If another user needs access to the computer system, they must wait till the current user finishes what they are doing and leaves. The Windows operating system supports the following file systems.

FAT	The MS-DOS operating system introduced the File Allocation Table system of keeping track of file entries and free clusters. Filenames were restricted to eight characters with an addition three characters signifying the file type. The FAT tables were stored at the beginning of the storage space.
FAT32	An updated version of the FAT system designed for Windows 98. It supports file compression and long filenames.
NTFS	Windows NT introduced the NT File System, designed to be more efficient at handling files than the FAT system. It spreads file tables throughout the disk, beginning at the center of the storage space.

10.2.1 Window Fundamentals

In a graphical operating system, information is represented in graphical ways. Little symbols or pictures (called icons) are used to display programmes or information. Information is displayed inside windows, each of which has similar properties. It is possible to have more than one window on the screen at one time, and windows may



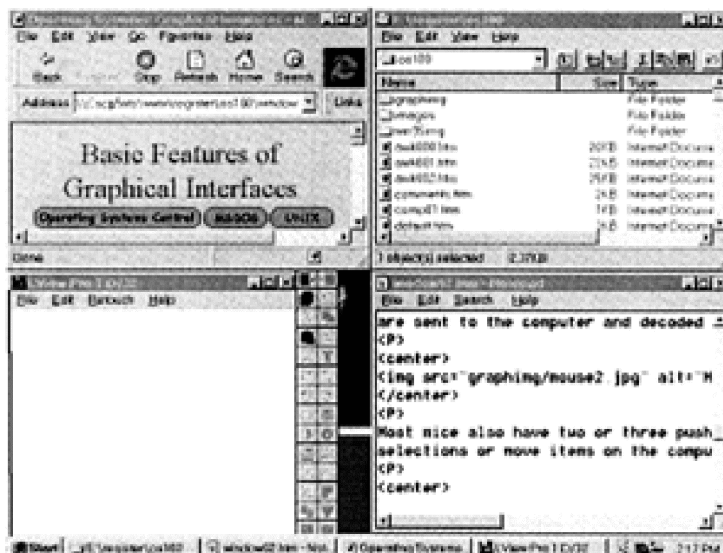
be cascaded (on top of one another) or tiled (all displayed at once and all visible).

In this picture, the windows have been cascaded. This makes each window appear on top of each other, one after the other.

The front most window is considered to be the active window, i.e., the window to which the users commands will be sent.

In Windows95 or WindowsNT, the title bar of the window is shown in the default color Blue.

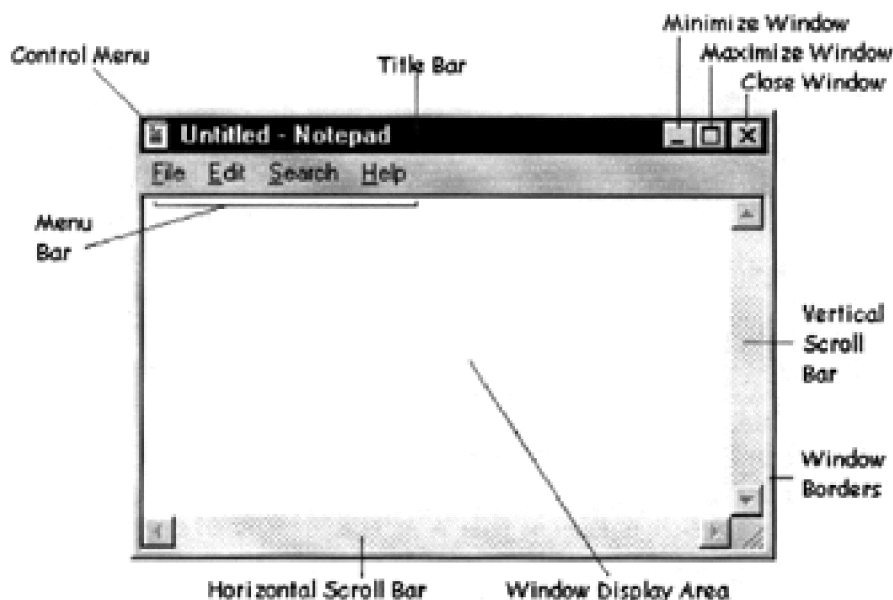
In this picture, the images have been tiled. This makes all windows visible at the same time, but resizes the dimensions of each window so that they all fit on the available screen space at once.



To cascade or tile all windows on the desktop area, right mouse click on an empty portion of the taskbar and select cascade Windows or Tile Windows from the menu.

10.2.2 Window Properties

Each window has the same properties and behaves the same way. This provides a consistent interface to the user, as all commands are the same for each window and the operations that the user performs on each window are identical. In the diagram below, we see the basic window as presented by Windows 95 or Windows NT. Each property is listed on the diagram, and below is an explanation for each of the window components.



10.2.2.1 The Title Bar

This normally displays the name of the programme associated with the window. If the background color of the title bar is blue, the window is active and any user commands will be processed by that window. You can also toggle between a maximized window size and the windows normal size by double clicking in the title bar area.

10.2.2.2 The Control Menu

Clicking on the Control Menu pops up a small Window of selectable options, which include the operations of Restore, Move, Size, Maximize, Minimize and Close the Window.




10.2.2.3 The Horizontal and Vertical Scroll Bars

When the amount of information displayed in the window exceeds the viewing space of the window, scroll bars are automatically to the side and bottom of the window. This allows the user to scroll the contents of the window in order to view the remaining information. Arrows are used to indicate the direction of scrolling on the scroll bar, and an indicator bar represents the relative position of the viewing area compared to the total size of the information.

Clicking on the arrows associated with the scroll bar move the viewing window up or down one line, or across or back one character position. You can also click on the small indicator bar within the scroll bar and drag it with the mouse to quickly scroll the windows contents.

10.2.2.4 The Minimize Maximize Close Window Buttons

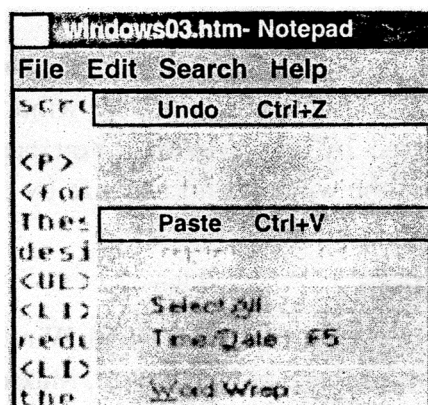
These buttons are located on the top right corner of each window. Clicking on them once performs the desired action associated with the button.

- The minimize button  reduces the window and places it on the taskbar at the bottom of the window.
- The maximize button  expands the window to fill the entire desktop screen area.
- The close button  closes the window.

To minimize all windows on the desktop area, right mouse click on an empty portion of the taskbar and select Minimize all Windows from the menu.

10.2.2.5 Menu Bar

The menu bar presents a number of options that the programme associated with the Window supports. On clicking an option on the menu bar will popup a submenu of choices that user can select from.



10.2.2.6 The Windows Borders

The windows borders show the dimensions of the window. Any window can be resized, either made smaller or larger, by dragging the window border appropriately.

- To make the window taller or shorter : Move the mouse pointer to either the top or bottom window border, and when it changes to a resize arrow \leftrightarrow , then hold the left mouse button down and drag the window border to its new position, then let release the left mouse button.
- To make the window narrower or wider : Move the mouse pointer to either the left or right window border, and when it changes to a resize arrow \updownarrow , then hold the left mouse botton down and drag the window border to its new position, then let release the left mouse botton.

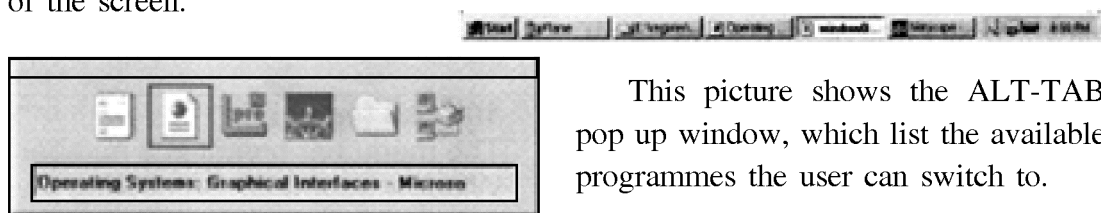
10.2.2.7 Moving a Window

A window can be repositioned on the desktop screen display area by moving the mouse cursor into the title bar area, then holding the left mouse button down and dragging the window to the new position, then releasing the left mouse button.

10.2.2.8 Switching between Windows

Switching between windows is possible by clicking on the programmes icon on the taskbar or pressing ALT-TAB keys on the keyboard. When you press ALT-TAB, it will pop up a window of the available programmes. Hold the ALT key down, and pressing the tab key will move the selection to the next window in the list. When the desired window is highlighted, release the ALT key and that window will become active.

Clicking on the applications icon on the taskbar can also do switching to another application. The following picture shows the Windows taskbar, located at the bottom of the screen.



This picture shows the ALT-TAB pop up window, which list the available programmes the user can switch to.

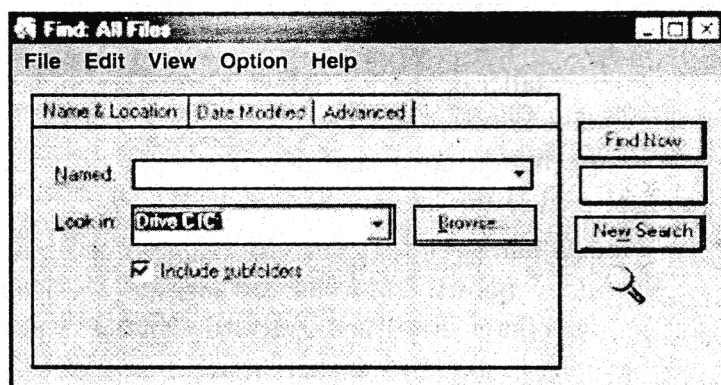
10.2.3 Window objects and components

This section discusses window options such as buttons and dialog boxes.

10.2.3.1 Text Boxes

Text boxes allow you to enter text information. To enter text, first click inside the text area using the mouse, and the cursor will change to a vertical flashing bar |

showing you that text box allows the user to specify a file to find on the computer. The name of the text box entry field is called Named :



10.2.3.2 Radio Buttons

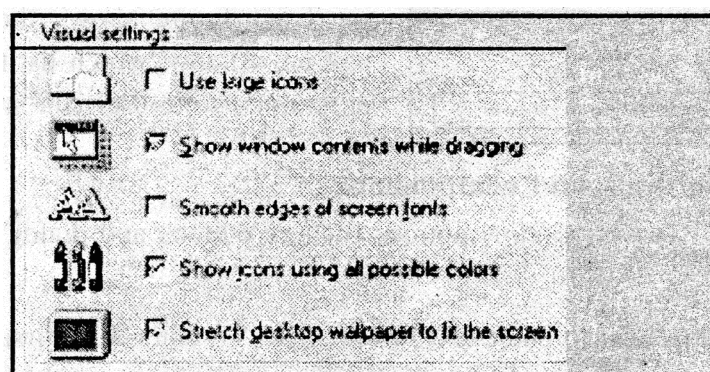
Radio buttons allow users to select one of a number of options from a selection. In the following image, a choice between Tiled and Centered is offered. A radio button is enabled when there is a black dot in its center. A radio button is disabled when it is empty. To enable a radio button, simply click once on it. To disable a radio button that is enabled, simply click once on it. It works like a toggle switch.



10.2.3.3 Check Boxes

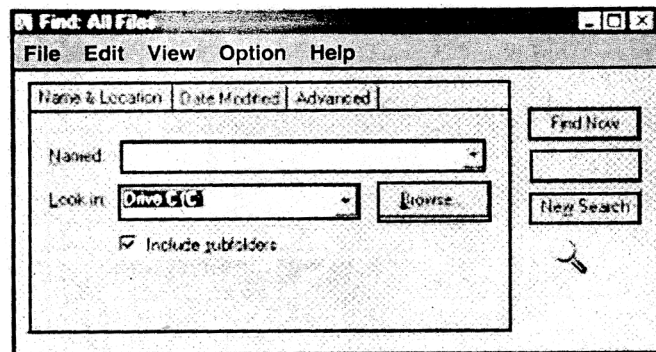
Check boxes allow users to select one or more options from a selection. In the following image, the options Show window contents while dragging, Show icons using all possible colors and Stretch desktop wallpaper to fit the screen are all enabled.

A check box is enabled when it has a tick in it, when a check box is empty, that option is not selected. To enable a check box, simply click once on it. To disable a check box that is enabled, simply click once on it. It works like a toggle switch.



10.2.3.4 Dialog Boxes

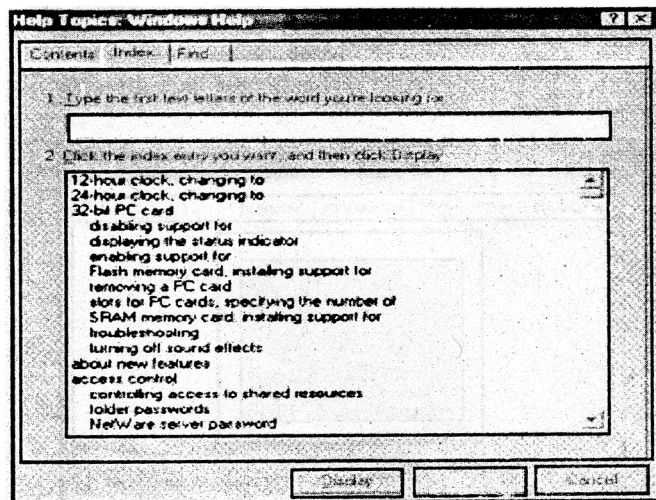
Dialog boxes allow you to make choices and enter data. They combine text boxes with radio buttons and check boxes.



To close a dialog box, press the ESC key.

10.2.3.5 List Boxes

List boxes present a number of choices. You select one by double-clicking on the item you want. Often the list of choices is in a scrollable window box.

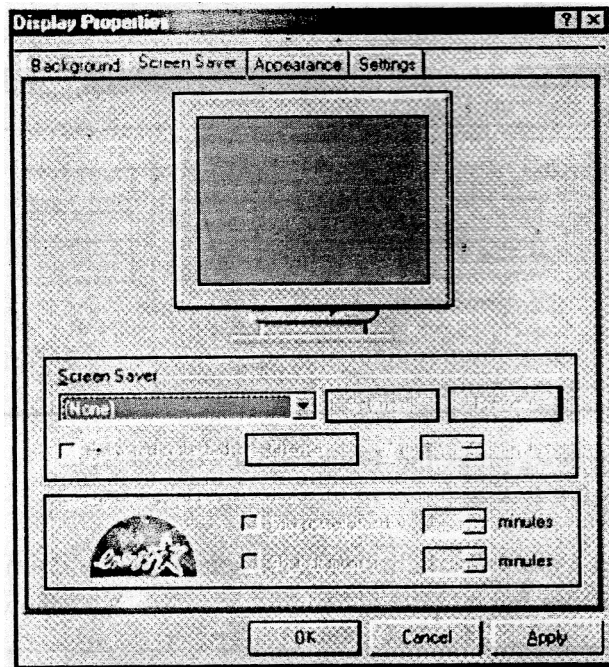



In this example, the Help dialog box of Windows lists a number of help topics that the user can double-click on to reveal the help associated with that item.

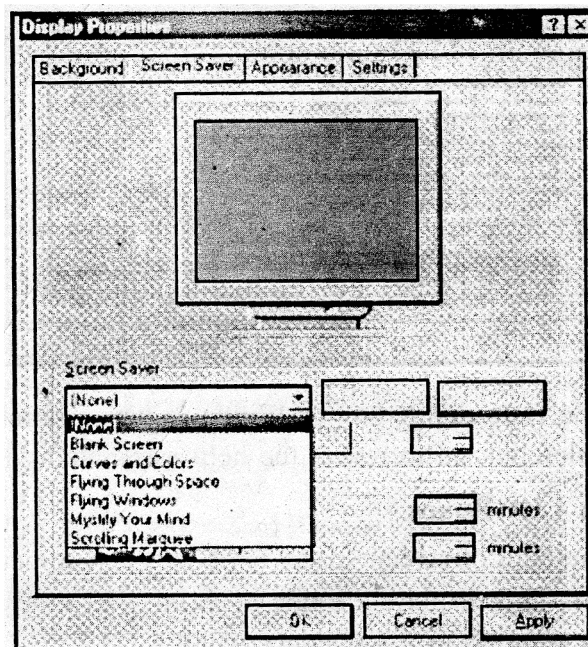
10.2.3.6 Drop down List Boxes

To minimize the amount of screen space, list boxes can sometimes be arranged as a drop down list box. This displays a single item, but when the list box is clicked on, the range of items pops up in a secondary window.

A drop down list box is shown below. In this example, it is part of the Dialog box associated with the Display Properties.



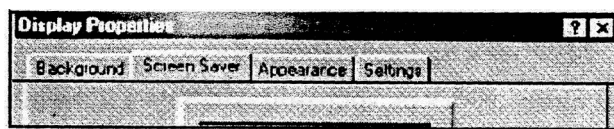
Notice the symbol  at the end of the box. Clicking on this symbol reveals the list of options.



10.2.3.7 Tab Controls

Tab controls allow a number of different dialog boxes associated with a device to be presented as a single combined control. For instance, if we looked at the screen display in Windows, there are so many things that can be changed, like screen saver, wall-paper, size and resolution, video display driver and so on.

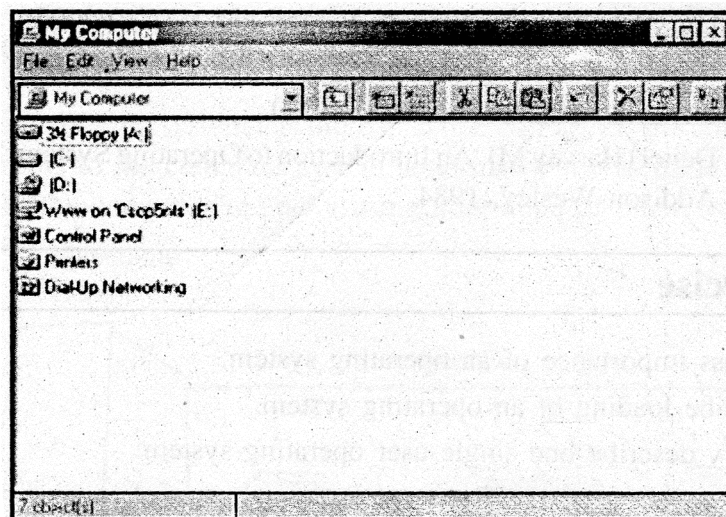
Putting all of these on a single dialog box is cumbersome and there is just not enough screen real estate, so, a number of dialog boxes are used, but they are combined using the tab control. It looks like multiple sections, and each tab has a heading. Clicking on the tab item reveals the dialog box associated with that tab.



In this example, the tab control for the Windows desktop properties is displayed. Note there are FOUR distinct dialog boxes : the current choice is Screen Saver.

10.2.3.8 Toolbars

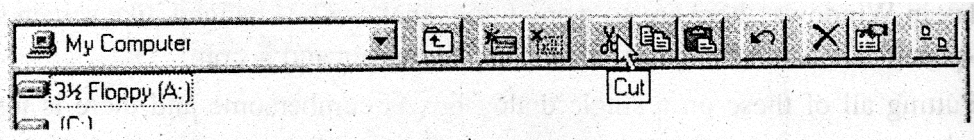
Toolbars appear on a number of windows and application programmes. An example is the My computer window.



The toolbar is displayed underneath the Menu Bar Options of the window. An expanded view looks like



The toolbar consists of a number of icons (little pictures), each representing a command. As the mouse cursor is moved along each icon, a text description will pop up revealing the available control that is underneath the mouse cursor.



Toolbars provide shortcuts to regularly used operations like cut paste, close, and help.

References and Further Readings

- 1 2005 What is the history of Microsoft windows (<http://kb.iu/data/abwa.html>). 2005. Visited last :
- 2 2005 Operating system (http://en.wikipedia.org/wiki/Operating_system). Visited last : 21/10/2005
- 3 2005 Coustan (Dabid) and Franklin (Curt). How operating system works (http://computer.howstuffworks.com/operating-system_system.htm). Visited last : 21/10/2005
- 4 2001 Introduction to UNIX : course outline (http://www.doc.ic.ac.uk/~wjk/UnixIntro/Lecture_1.html). Visited last : 21/10/2005
- 5 2000 Brown (Brain). Operating system introduction, v4.0. (<http://goforit.unk.edu/opsys/default.htm>). 2000.
- 6 1984 Deitel (Harvey M). An Introduction to Operating System. Massachusetts : Addison-Wesley, 1984.

10.3 Exercise

1. Discuss importance of an operating system.
2. Describe loading of an operating system.
3. Briefly describe one single user operating system.

Unit 11 □ Multi User Operating Systems

Structure

11.0 Objectives

11.1 Introduction

11.2 UNIX/LINUX

11.3 WINDOWS NT

11.4 Exercise

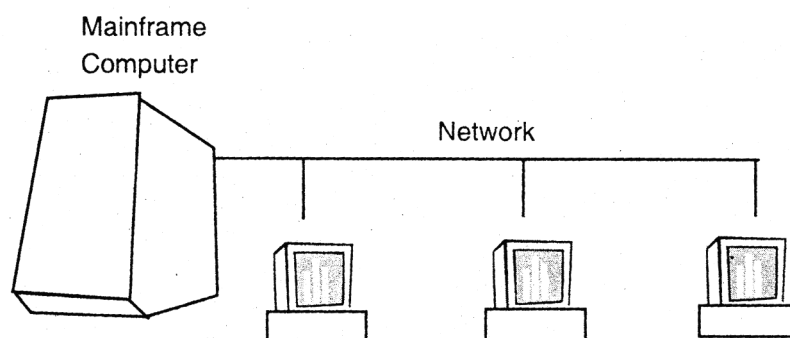
11.0 Objectives

The objectives of the Unit are to :

- Introduce the concept of multi-user and multi-tasking operating system
- Provide basics of UNIX/LINUX
- List selected commands of UNIX/LINUX
- Present brief description of Windows NT

11.1 Introduction

A multi-user operating system lets more than one user access the computer system at one time. Access to the computer system is normally provided via a network, so that users access the computer remotely using a terminal or other computer.



The operating system for a large multi-user computer system with many terminals is much more complex than a single-user operating system. It must manage and run

all user requests, ensuring they do not interfere with each other. Devices that are serial in nature (devices which can only be used by one user at a time, like printers and disks) must be shared amongst all those requesting them (so that all the output documents are not jumbled up). If each user tried to send their document to the printer at the same time, the end result would be garbage. Instead, documents are sent to a queue, and each document is printed in its entirety before the next document to be printed is retrieved from the queue.

In addition, the operating system provides each user with an interface that accepts, interprets and executes user commands or programmes. This interface is commonly called a SHELL or command line interpreter (CLI). In some systems this might be a simple text mode line-by-line entry using keywords (such as MSDOS or UNIX), in other systems it might be highly graphical using windows and a pointing device such as a mouse (such as X-Windows).

The advantage of having a multi-user operating system is that normally the hardware is very expensive, and it lets a number of users share this expensive resource. This means the cost is divided amongst the users. It also makes better use of the resources. Since the resources are shared, they are more likely to be in use than sitting idle being unproductive.

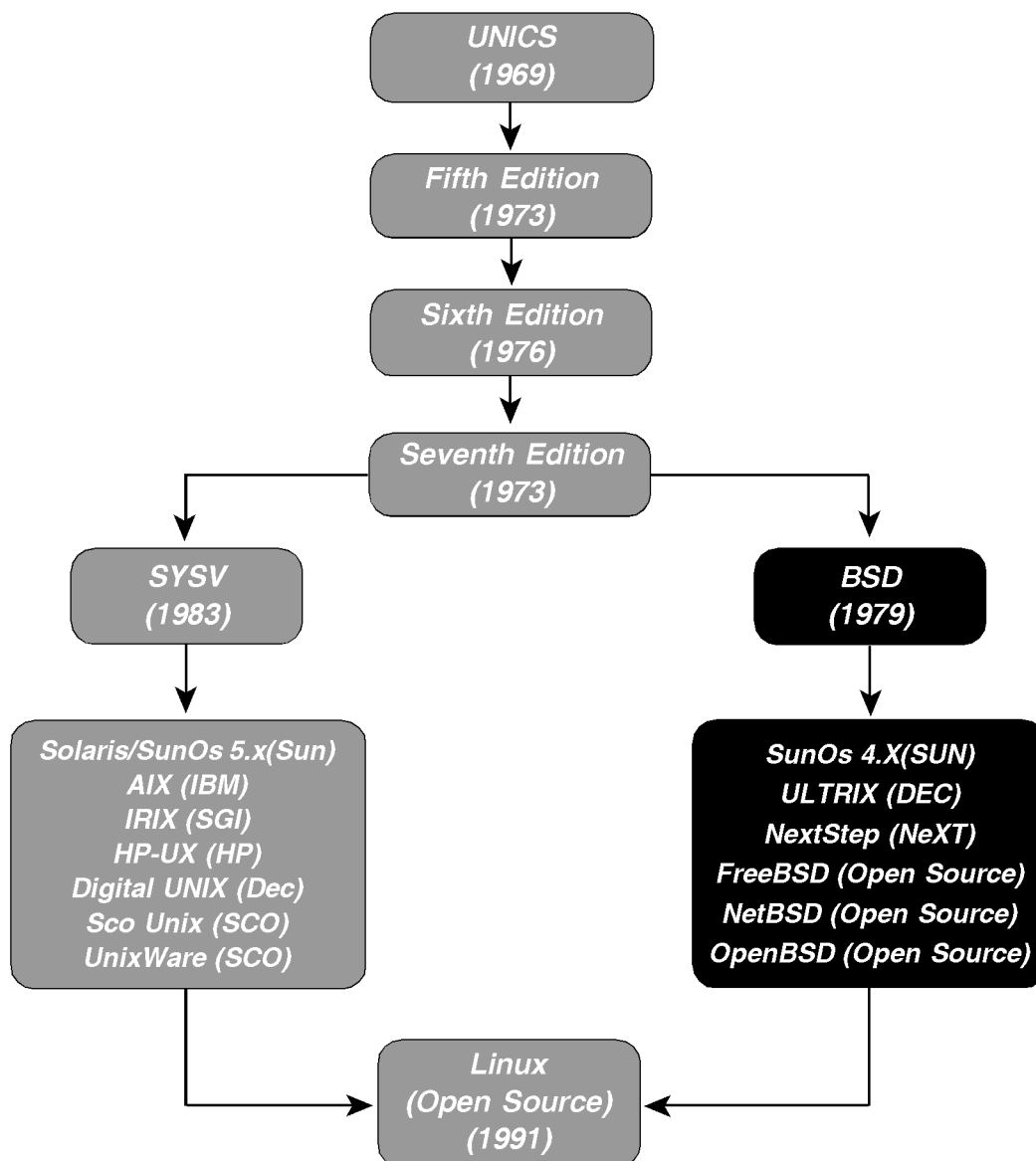
One problem with multi-user computer systems is that as more users access it, the performance becomes slower and slower. Another disadvantage is the cost of hardware, as a multi-user operating system requires a lot of disk space and memory. In addition, the actual software for multi-user operating systems tend to cost more than single-user operating systems.

11.2 UNIX and LINUX

UNIX is a powerful computer operating system originally developed at AT&T Bell Laboratories. It is very popular among the scientific, engineering, and academic communities due to its multi-user and multi-tasking environment, flexibility and portability, electronic mail and networking capabilities, and the numerous programming, text processing and scientific utilities are available. It has also gained widespread acceptance in government and business. Over the years, two major forms (with several vendors' variants of each) of UNIX have evolved : AT & T UNIX System V and the University of California at Berkeley's Berkeley Software Distribution (BSD).

11.2.1 Brief History

In the late 1960s, researchers from General Electric, MIT and Bell Labs launched a joint project to develop an ambitious multi-user, multi-tasking OS for mainframe computers known as MULTICS (Multiplexed Information and Computing System). MULTICS did inspire Ken Thompson, who was a researcher at Bell Labs, to write a simpler operating system himself. He wrote a simpler version of MULTICS on a PDP7 in assembler and called his attempt UNICS (Uniplexed Information and Computing System, eventually shortened to UNIX).



Ken Thompson then teamed up with Dennis Ritchie, the author of the first C compiler in 1973. They rewrote the UNIX kernel in C-this was a big step forwards in terms of the system's portability-and released the Fifth Edition of UNIX to universities in 1974. The Seventh Edition, released in 1978, marked a split in UNIX development into two main branches : SYSV (System 5) and BSD (Berkeley Software Distribution). BSD arose from the University of California at Berkeley where Ken Thompson spent a sabbatical year. Students at Berkeley and other research institutions continued its development. AT&T and other commercial companies developed SYSV. UNIX flavors based on SYSV have traditionally been more conservative, but better supported than BSD-based flavors.

Linux is a free open source UNIX OS for PCs that was originally developed in 1991 by Linus Torvalds, a Finnish undergraduate student. Linux is neither pure SYSV nor pure BSD. Instead, incorporates some features from each (e.g. SYSV- style startup files but BSD-style file system layout) and aims to conform to a set of IEEE standards called POSIX (Portable Operating System Interface).

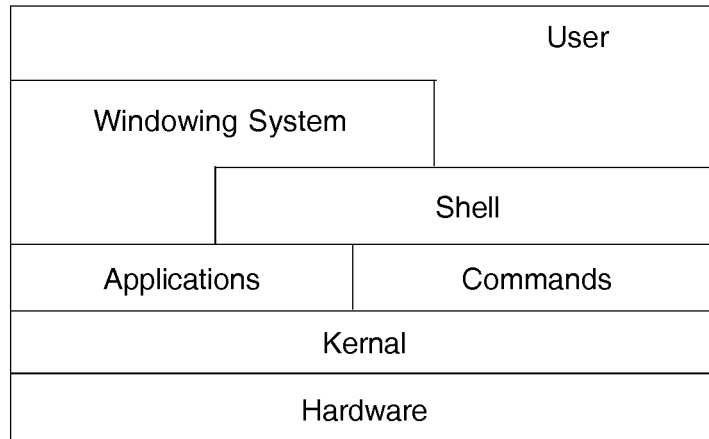
The open source nature of Linux means that the source code for the linux kernel is freely available so that anyone can add features and correct deficiencies. The open source approach has not just successfully been applied to kernel code, but also to application programmes for Linux.

As Linux has become more popular, several different development streams or distributions have emerged, e.g. Redhat, Slackware, Mandrake, Debian, and Caldera, etc. A distribution comprises a prepackaged kernel, system utilities, GUI interfaces and application programmes.

11.2.2 Architecture of the UNIX LINUX Operating System

Several layers of interaction are occurring between the computer hardware and user. The first layer is the *Kernel*, which runs on the actual machine hardware and manages all interaction with the hardware. All *applications* and *commands* in UNIX interact with the kernel, rather than the hardware directly, and they make up the second layer. On top of the applications and commands is the command-interpretor programme, the *shell*, which manages the interaction between user, applications, and the available UNIX commands. Most UNIX commands are separate programmes, distinct from the kernel. A final layer, which may or may not be present is a *windowing system* such as X. The windowing system usually interacts with the shell, but it can also interact directly with applications. The final "Layer" is user. User will interact with the entire operating system through just the shell, or through a combination of the shell

and the window system. The figure below gives a visual representation of the layers of UNIX.



Linux has all of the components of a typical OS. They are :

- **Kernel :** The Linux kernel includes device drive support for a large number of PC hardware devices, advanced processor and memory management features, and support for many different types of file systems.
- **Shells and GUIs :** Users interacts with the system through a command interpreter programme called the *Shell*. Linux supports two forms of command input : through textual command line shells similar to those found on most UNIX systems (e.g.sh-the Bourne shell, bash-the Bourne again shell and csh-the C shell) and through graphical interfaces (GUIs) such as the KDE and GNOME window managers. Remote connection to a server will typically be through a command line shell. In addition to processing user command requests, UNIX shells have their own syntax and control constructs. User can use these shell commands to make processing more efficient, or to automate repetitive tasks. User can even store a sequence of shell commands in a file, called a *shell script*, and run it just like an ordinary programme.
- **System Utilities :** Virtually every system utility has been ported to Linux. These system utilities are designed to be powerful tools that do a single task extremely well. Users can often solve problems by interconnecting these tools instead of writing a large monolithic application programme. Like other UNIX flavors, Linux's system utilities also include server programmes called daemons, which provide remote network and administration services. A daemon is usally spawned automatically at system startup and spends most of its time lying dormant waiting for some event to occur.

- **Application Programmes :** Linux distributions typically come with several useful application programmes as standard. Examples include the emacs editor, xv (an image viewer), gcc (a C compiler), g++ (a C++ compiler), and soffice (StarOffice, which is an MS-Office style clone that can read and write Word, Excel and PowerPoint files). Redhat Linux also comes with rpm, the Redhat Package Manager, which makes it easy to install and uninstall application programmes.

11.2.3 Basic UNIX/LINUX Elements

Six basic elements of UNIX are :

1. **Commands :** are the instructions user give the system to tell it what to do.
2. **Files :** are collections of data that have been given *filenames*. A file is analogous to a container in which user can store documents, raw data, or programmes. A single file might contain the text of a research project, statistical data, or an equation processing formula. Files are stored in *directories*.
3. **Directories :** is similar to a file cabinet drawer that contains many files. A directory can also contain other directories. Every directory has a name, like files.
4. **User environment :** is a collection of items that describe or modify how user-computing session will be carried out. It contains things such as where the commands are located and which printer to send user output to.
5. **Processes :** A command or application running on the computer is called a *process*.
6. **Jobs :** The sequence of instructions given to the computer from the time user initiate a particular task until it ends it is called a *job*. A job may have one or more processes in it.

11.2.4 Syntax of UNIX Commands

A UNIX command line consists of the name of the UNIX command followed by its arguments. The option flags act like adverbs by modifying the action of the command, and filenames and expressions act like objects of the verb. The general syntax for a UNIX command is :

Command -flag options *file/expression*

Flags need not always be specified separately, each with their own preceding dash. Many times, the flags can be listed one after the other after a single dash. User should follow the following rules with UNIX commands :

1. UNIX commands are case-sensitive, but most are lowercase.
2. UNIX commands can only be entered at the shell prompt.
3. UNIX command lines must end with a RETURN.
4. UNIX options often begin with a “-” (minus sign).
5. More than one option can be included with many commands.

11.2.5 UNIX/LINUX File System

The UNIX operating system is built around the concept of a file system which is used to store all of the information that constitutes the long-term state of the system. This state includes the operating system kernel itself, the executable files for the commands supported by the operating system. Configuration information, temporary work-files, user data, and various special files that are used to give controlled access to system hardware and operating system functions. Every item stored in a UNIX file system belongs to one of four types :

1. **Ordinary files** : Ordinary files can contain text, data, or programme information. Files cannot contain other files or directories. Unlike other operating systems, UNIX filenames are not broken into a name part and an extension part (although extensions are still frequently used as a means to classify files). Instead they can contain any keyboard character except for '/' and be up to 256 characters long (note however that characters such as *, ?, # and & have special meaning in most shells and should not therefore be used in filenames). Putting spaces in filenames also makes them difficult to manipulate-rather use the underscore '_'.
2. **Directories** : Directories are containers or folders that hold files, and other directories.
3. **Devices** : To provide applications with easy access to hardware devices, UNIX allows them to be used in much the same way as ordinary files. There are two types of devices in UNIX -block-oriented devices which transfer data in blocks (e.g. hard disks) and character-oriented devices that transfer data on a byte-by-byte basis (e.g. modems and dumb terminals).
4. **Links** : A link is a pointer to another file. There are two types of links - a hard link to a file is indistinguishable from the file itself. A soft link (or symbolic link) provides an indirect pointer or shortcut to a file. A soft link is implemented as a directory file entry containing a pathname.

The UNIX file system is laid out as a hierarchical tree structure, which is anchored at a special top-level directory known as the root (designated by a slash '/'). Because of the tree structure, a directory can have many child directories, but only one parent directory.

To specify a location in the directory hierarchy, we must specify a path through the tree. The path to a location can be defined by an absolute path from the root /, or as a relative path from the current working directory. To specify a path, each directory along the route from the source to the destination must be included in the path, with each directory in the sequence being separated by a slash. To help with the specification of relative paths, UNIX provides the shorthand "." for the current directory and ".." for the parent directory. For example, the absolute path to the directory "play" is /home/will/play, while the relative path to this directory from "zeb" is ../will/play.

The following table shows some typical directories that may be found on UNIX systems and briefly describes their contents. Note that although certain subdirectories appear as part of a seamless logical file system, they do not need be present on the same hard disk device; some may even be located on a remote machine and accessed across a network.

Directory	Typical Contents
/	The "root" directory
/bin	Essential low-level system utilities
/user/bin	Higher-level system utilities and application programmes
/sbin	Superuser system utilities (for performing system administration tasks)
/lib	Programme libraries (collections of system calls that can be included in programmes by a compiler) for low-level system utilities
/usr/lib	Programme libraries for higher.level user programmes
/tmp	Temporary file storage space (can be used by any user)
/home or	User home directories containing personal file space for each user.
/homes	Each directory is named after the login of the user.
/etc	UNIX system configuration and information files
/dev	Hardware devices
/proc	A pseudo-filesystem which is used as an interface to the kernel. Includes a sub-directory for each active programme (or process).

Some UNIX files begin with a period, for example, **.cshrc** or **.login**. Files that begin with a period will not appear in a normal directory listing and are usually UNIX environment and application setup files.

11.2.13 File and Directory permissions

It is important to protect user. The UNIX operating system maintains information, known as permissions, for every file and directory on the system. Every file or directory in a UNIX file system has three types of permissions (or protections) that define whether certain actions can be carried out. The permissions are :

1. read (**r**) A user who has read permission for a file may look at its contents or make a copy of it. For a directory, read permission enables a user to find out what files are in that directory.
2. write (**w**) A user who has *write* permission for a file can alter or remove the contents of that file. For a directory, the user can create and delete files in that directory.
3. execute (**x**) A user who has *execute* permission for a file can cause the content of that file to be executed (provided that it is executable). For a directory, execute permission allows a user to change to that directory.

For each file and directory, the read, write, and execute permissions may be set separately for each of the following classes of users :

- User (u) : The user who owns the file or directory.
- Group (g) : Several users purposely grouped together so that they can share access to each other's files.
- Others (o) : The remainder of the authorized users of the system.

The primary command that displays information about files and directories is **ls**. The **-l** option will display the information in a long format. You can get information about a single UNIX file by using **ls-i filename**.

Each file or subdirectory entry in a directory listing obtained with the **-l** option consists of seven fields :

1. permission mode,
2. link count,
3. owner name,
4. group name,
5. file size in bytes,
6. time of last modification, and
7. the filename (the group name appears only if the "g" flag is also specified, as in **ls-ig**).

The first 10 characters make up the mode field. If the first character is a “d” then the item listed is a directory; if it is a “-” then the item is a file; if it is an “l” then it is a link to another file.

Characters 2 through 4 refer to the owner’s permissions, characters 5 through 7 to the group’s permissions (groups are defined by the system administrator), and the last three to the general public’s permissions. If a particular permission is set, the appropriate letter appears in the corresponding position; otherwise, a dash indicates that the permission is not given.

The second field in the output from **ls-l** is the number of links to the file. In most cases it is one, but other users may make links to your files. thus increasing the link count. A special warning to people using links to other people’s files : your “copies” of their files can be counted against them by the file quota system available on certain UNIX variants. That is why making links other than symbolic links to other people’s files is strongly discouraged. The third field gives the userid of the owner of the file. The group name follows in the fourth field (if the **-g** option is used in conjunction with **-l**). The next two fields give the size of the file (in bytes) and the date and time at which the file was last modified. The last field gives the name of the file.

```
ls -l myfile
-rw-r--r-- 1 owner 588 Jul 15 14 : 39 myfile
```

A file’s owner can change any or all of the permissions with the **chmod** (change mode) command. The **chmod** command allows user to dictate the type of access permission that user want each file to have. In the previous example the current permissions for **myfile** are read for everybody, write for the owner, and execute by no one.

The arguments supplied to **chmod** and a symbolic specification of the changes required, followed by one or more filenames. The specification consists of whose permissions are to be changed : **u** for user (owner), **g** for group, **o** for others, or some combination thereof (**a** (**all**) has the same effect as **ugo**), how they are to be changed (+ adds a permission, -removes a permission, and = sets the specified permissions, removing the other ones) and which permission to add or remove (**r** for read, **w** for write, and **x** for execute). For example, to remove all the permissions from **myfile** :

```
chmod a-rwx myfile
ls-l myfile
----- 1 owner 588 Jul 15 14 : 41 myfile
```

(Note : **chmod a = myfile** achieves the same effect.)

To allow read and write permissions for all users :

```
chmod ugo+rw myfile
ls-i myfile
-rw-rw-rw 1 owner 588 Jul 15 14 : 42 myfile
```

To remove write permission for your groups and other users :

```
chmod go-w myfile
ls-i myfile
-rw-r-r-1 owner 588 Jul 15 14 : 42 myfile
```

Finally, to allow only read permission to all users :

```
chmod a=r myfile
ls-i myfile
-r-r-r- 1 owner 58 Jul 15 14 : 43 myfile
```

Now the file is protected by allowing only read access; it cannot be written to or executed by anyone, *including user*. Protecting a file against writing by its owner is a safeguard against accidental overwriting, although *not* against accidental deletion.

11.2.14 Wildcard Characters

Using wildcard characters that allow you to copy, list, move, remove, etc. items with similar names is a great help in manipulating files and directories.

1. The symbol `?` will match any single character in that position in the file name.
2. The symbol `*` will match zero or more characters in the name.
3. Characters enclosed in brackets `[and]` will match any one of the given characters in the given position in the name. A consecutive sequence of characters can be designated by `[char char]`.

11.2.15 Processes

Every command or programme running under UNIX is called a *process*. A sequence of related processes is called a *job*. Your applications and even your shell itself are processes. The windowing system is also a process, or a collection of processes. The UNIX kernel manages the processes on the system, usually without distinguishing among them. UNIX is a multi-tasking system—it allows you to continue to work in the foreground while running one or more jobs in the background. It also runs the processes of many users simultaneously. You could even log off and come back later if the background jobs do not require interaction with you.

11.2.16 Running Background Jobs

Putting a programme into an unattended state where it continues to execute is referred to as putting it (the process or job) into the *background*. (Running a programme on one machine and displaying its output on another via a windowing system like X is not considered backgrounding the job.). Adding an & (ampersand) at the end of the command line instructs UNIX to run the job in the background.
jobname&

11.3 Windows NT

Windows NT is a family of operating systems produced by Microsoft, and was succeeded by Windows 2000 (still based on Windows NT). Although the company decided to drop “NT” from the Windows naming scheme for marketing, it internally still uses Windows NT as a means of reference. The latest available operating systems based on NT are Windows XP and Windows Server 2003.

11.3.1 Development

When development started in November 1988, Windows NT was to be known as OS/2 3.0, the third version of the operating system developed jointly by Microsoft and IBM. Microsoft hired a group of developers from Digital Equipment Corporation led by Dave Cutler to build Windows NT and many elements reflect earlier DEC experience with VMS and RSX-11. The OS is designed to run on multiple instruction set architectures, with the kernel separated from the hardware by a hardware abstraction layer. APIs are implemented as subsystems atop the publicly undocumented native API; it was this that allowed the late adoption of the Windows API. Originally a microkernel design, subsequent releases have integrated more functions into the kernel for better performance. Windows NT was the first operating system to use Unicode internally.

11.3.2 Releases

Windows NT Releases

NT Ver.	Marketing Name	Editions	Release date	Build
NT 3.1	Windows NT 3.1	Workstation, Advanced Server	July <u>1993</u>	528
NT 3.5	Windows NT 3.5	Workstation, Server	September 1994	807
NT 3.51	Windows NT 3.51	Workstation, Server	May 1995	1057

NT Ver.	Marketing Name	Editions	Release date	Build
NT 4.0	Windows NT 4.0	Workstation, Server, Enterprise Edition, Terminal Server, Embedded	July 1996	1381
NT 5.0	Windows 2000	Professional, Server, advanced Server, Datacenter Server	February 2000	2195
NT 5.1	Windows XP	Home, Professional Media Center (2004 & 2005), Tablet PC, Starter, Embedded, N	October 2001	2600
NT 5.2	Windows Server 2003	Standard, Enterprise, Datacenter, Web, XP Pro x64	March 2003	3790
NT 6.0	Windows Vista	Starter, Home Basic, Home Premium, Professional, Small Business, Enterprise, Ultimate (x64 editions will be available too)	2006 (expected)	Unknown
NT Longhorn 6.0 + (codename)		Server Unknown	2007 (expected)	Unkown

The first release was given version number 3.1. The NT version is no longer marketed, but is said to reflect the degree of changes to the core of the operating system. The build number is an internal figure used by Microsoft's developers.

11.3.3 Supported platforms

Windows NT 3.1 ran on Intel IA-32 x86, DEC Alpha, and MIPS R4000 processors. Windows NT 3.51 added support for PowerPC processors. Intergraph Corporation

ported Windows NT to its Clipper architecture and later SPARC, but neither version was sold to the public. Windows NT 4.0 was the last major release to support Alpha, MIPS, or PowerPC, though development of Windows 2000 for Alpha continued until 1999, when Compaq stopped support for Windows NT on that architecture. Only 2 of the Windows NT 4.0 variants (IA-32 and Alpha) have a full set of service packs available. All of the other ports done by 3rd parties (Motorola, Intergraph, etc.) have few, if any, publicly available updates.

Windows XP 64-Bit, windows Server 2003 Enterprise, and Windows Server 2003 Datacenter support Intel's IA-64 processors. Microsoft had released four editions for the AMD 64 : Windows XP Professional x64 Edition, Windows Server 2003 Standard x64 Edition, Windows Server 2003 Enterprise x64 Edition and Windows Server 2003 Datacenter x64 Edition.

The Xbox uses a heavily modified and stripped down Windows 2000 kernel. This kernel was heavily modified again for the Xbox 360 which runs on Power PC. This version is not for sale, and little is generally known about it.

11.3.4 Windows NT File systems

It supports different file systems. The following table presents an overview of supported file system :

File System	Description
FAT File system	Used with DOS, it can only support partitions up to 4 G. No spaces are allowed in the file name.
FAT 32 or VFAT File System	<p>VFAT-Virtual File Allocation Table introduced by Windows 95. Some documentation says NTWS cannot use FAT32.</p> <ul style="list-style-type: none"> ● Filenames up to 255 characters long. ● Names begin with a letter and exclude " / \ [] : ; = , ^ * ? ● The last part is the extension but spaces can be used ● It supports file attributes used by DOS such as read-only, archive, system, and hidden. ● Won't support running POSIX applications. RISC computers can only boot from FAT files systems. FAT file systems support dual booting of operating systems. FAT partitions provide no local security, only share level security. ● Filenames up to 255 characters long ● File names preserve case but are not case sensitive. ● Exclude " / \ < > : * ?

File System	Description
NTFS File System	<ul style="list-style-type: none"> ● Supports built in file compression as a file attribute. Compression is applied to files in a folder if that folder has its compression attribute set. Also optionally sub folders and their contents may be compressed. Compression is not supported if the cluster size is above 4K in size. Moved files retain their compression attribute, but if they are copied they will assume the compression attribute of the target folder. ● Provides automatic transaction tracking of disk activity for correcting corrupt or failed operations. ● Supports auditing. ● Provides sector sparing. ● There is a recycle bin for each user ● Windows 16 bit and DOS environments can't use this filesystem. ● A master file table is used to save individual file, boot sector, disk structure, and file recovery information. ● Automatically makes 11 character DOS file names. When the first 8 characters of long filenames match, the first four DOS file names use the first four characters of the long name, the ~and 1, then ~2, etc. After the fourth duplicate name, the first two characters are used, then the next four characters are hashed, then the ~ character then a number. The first two duplicate file names may be: DOCU~1.DOC and DOCU~2.DOC. The long extension is used as part of the extension for the 8.3 filename alias. Directory entries used by long filenames include 1 for the 8.3 alias and 1 for each 13 characters in the long filename. ● Provides file logging ability and file recovery. ● Supports POSIX. ● Maximum file or partition size of 16 exabytes. ● Supports file sharing with Macintosh clients.

11.3.5 Features

The section presents various features of Windows NT

Microsoft Operating System	Features
Windows NT Server	<ul style="list-style-type: none"> ● Supports 256 clients on the RAS or DUN server. ● Supports unlimited clients for resource access ● Supports unlimited outbound connections. ● Can support up to 4 processors and 32 processors if OEM version of the system. ● Peer Web Services (PWS) provided for small workgroup web publishing. ● Can import or export directory replication. ● Includes System Management Server, SNA Server, and SQL Server, IIS (Internet Information Server), DNS Name Server for fully qualified internet domain names. ● Apple Share server that allows access to Macintosh and can act as a gateway to Netware servers ● Supports DHCP, BOOTP, WINS and multi-protocol routing. ● Can remotely reboot Windows 95 clients. ● Supports disk fault tolerance with mirroring, striping and hot fixing ● Supports directory replication which replicates directories and files to other computers on the network. ● Support RAID
Windows NT Workstation	<ul style="list-style-type: none"> ● Can support one remote dial in session as a RAS or DUN server. ● Supports up to 10 clients for resource access. ● Any number of peer to peer outbound connections. ● Can support up to 2 processors and 32 processors if OEM version of the system. ● Peer web servers. ● Can import directory replication and not export. ● No disk fault tolerance support.
Windows NT Server & Workstation	<ul style="list-style-type: none"> ● A 32 bit operating system ● Supports 16 and 32 bit applications. ● There is no direct hardware access. ● Provides memory protection. ● Logon is mandatory. The CTRL-ALT-DEL key combination is used to logon since it disables user mode and TSR programs in NT.

	<ul style="list-style-type: none"> ● Expanded memory support-Supports up to 4 GB of RAM. ● Preemptive Multitasking - Threads may be assigned relative priorities. ● Expanded File system support-Supports the New Technology File system (NTFS) for increased security and can track disk transactions which will aid in the recovery of data should loss of power occur. It supports up to 16 exabytes of disk space. Supports FAT, VFAT, and the CDFS file systems. ● Local and shared (extensive) security. ● Multiple platform support-The Hardware Abstraction Layer (HAL) isolates the operating system from the platform. Supports both CISC and RISC platforms. ● Can operate multiple microprocessors on one computer using symmetric multiprocessing. ● Can run OS2 1.3 but not 2.0
--	--

11.3.6 Windows NT Structure

NT runs in two modes :

1. Kernel mode (Ring 0)-Executive which runs in protected memory mode with full privileges.
2. User mode (Ring 2) - Runs with privileges to access its own memory area.

User applications and environmental subsystems execute in this mode. Applications are allocated a virtual 4GB of memory with 2 for the user and 2 for executive services. NT is modular in nature allowing it to have cross platform portability due primarily to the HAL module described below. The NT Architecture has 5 layers.

3. Application - Runs in user mode.
4. Subsystems-Runs in user mode.
5. Executive Services-Runs in Kernel mode.
6. Kernel-Runs in kernel mode.
7. HAL-Runs in kernel mode.

The NT architecture model in more detail, from the top down :

1. User Level-The environmental subsystem and user applications execute at this level which runs in Ring 3, a non-privileged processor mode. User mode code can be preempted, is pageable, and can be context switched. User applications must use executive services to access devices or memory.
2. Kernel Level also called executive services run in the protected mode of the processor ring. Cannot be paged or context switched.

11.3.7 A Comparison of LINUX and WINDOWS

Parameters	WINDOWS	LINUX
Flavors	All the flavors of Windows come from Microsoft, Windows has two main lines: "Win9x", which consists of Windows 95, 98, 98SE and Me, and "NT class" which consists of Windows NT, 2000 and XP.	The various distributions of Linux come from different companies (i.e. Linspire, Red Hat, SuSE, Mandrake, Knoppix, Slackware). There are many special purpose
Customization	Linux is customizable in a way that Windows is not. Microsoft line of division is Professional and Home edition.	versions of Linux above and beyond the full distributions described above. For example, <u>NASLite</u> is a version of Linux that runs off a single floppy disk and converts an old computer into a file server. This ultra small edition of Linux is capable of networking; file sharing and being a web server.
GUI	The Windows GUI has changed from Windows 3.1 to Windows 95 (drastically) to Windows 2000 (slightly) to Windows XP (fairly large). Windows XP has a themes feature that offers some customiza-tion of the look and feel of the GUI. The Windows GUI is an integral component of the OS.	Typically provides two GUIs, KDE and Gnome. The GUI is optional. Speed, efficiency and reliability are all increased by running a server instance of Linux without a GUI, something that server versions of Windows cannot do. The detached nature of the Linux GUI makes remote control and remote administration of a Linux computer simpler and more natural than a Windows computer.
Text Mode Interface	Windows users sometimes call it a DOS prompt. Each version of Windows has a single command interpreter, but the different flavors of Windows have different interpreters. In general, the command interpreters in the Windows 9x series are very similar to each other and	Linux users refer to it as a shell. Linux like all versions of Unix, supports multiple command interpreters, but it usually uses one called BASH (Bourne Again Shell). Others are the Korn shell.

Parameters	WINDOWS	LINUX
	the NT class versions of Windows (NT, 2000, XP) also have similar command interpreters. There are however differences between a Windows 9x command interpreter and one in an NT class flavor of Windows.	
Cost	For desktop or home use Windows is expensive. Microsoft allows a single copy of Windows to be used on only one computer. Starting with Windows XP, they use software to enforce this rule (activation).	For desktop or home use. Linux is very cheap or free. For server use, Linux is very cheap compared to Windows. Once you have purchased Linux, you can run it on any number of computers for no additional charge.
Installation	Installing Windows from scratch is much easier than installing Linux from scratch	The different distributions of Linux have their own installation programs. However, installation of multiple OS is feasible.
Running from CD	Windows can not run from a CD.	Normally Linux also runs from a hard disk. but there are quite a few versions of Linux that run <i>completely</i> from a CD without having to be installed to a hard disk (the term for this is a "Live" CD). Among the Linux distros that have a CD-only version are Knoppix, and SuSE (called Live-Eval) etc.
Application Software	There is more application software available for Windows.	
Viruses & Spyware	The vast majority of all malicious software (of all types) runs on Windows.	
Users & Passwords	Both Linux and Windows 2000/XP require a userid and password and boot time.	Both Linux and Windows can group users into groups and assign privileges to the group rather than to each individual user.
Software Restrictions		

Parameters	WINDOWS	LINUX
Supported Hardware	More hardware works with Windows than works with Linux.	
Hardware Support for OS		Linux runs on many different hardware platforms, not so with Windows.
Clustering		Linux has an edge here. It has been used to make enormous clusters of computers.
Multiple Users	<p>Windows is not a multi-user system. That is, Windows is designed to be used by one person at a time. Databases running under Windows allow concurrent access by multiple users, but the Operating System itself is designed to deal with a single human being at a time.</p> <p>Windows, of course, can run many programs concurrently, as can Linux. There is a multi-user version of Windows called Terminal Server but this is not the Windows pre-installed on personal computers.</p>	Linux is a multi-user system. Linux, like all Unix variants, is designed to handle multiple concurrent users.
Networking	They both do TCP/IP	Linux can also do Windows networking, which means that a Linux computer can appear on a network of Windows computers and share its files and printers. Linux machines can participate on a Windows based network and vice versa.
Hard Disk Partition	Windows must boot from a primary partition. Windows must boot from the first hard disk.	Linux can boot from either a primary partition or a logical partition inside an extended partition. Linux can boot from any hard disk in the computer.
File Systems	Windows uses FAT12, FAT16, FAT32 and/or NTFS with NTFS almost always being the best choice. Linux also has a number of its own native file systems.	The default file system for Linux used to be ext2, now it is typically ext3.
File Hierarchy	Windows uses a volume-based file hierarchy. Windows uses letters of the alphabet to represent different devices and different hard disk partitions.	Linux uses a unified scheme. In Linux all directories are attached to the root directory, which is identified by a forward-slash, "/".
Case Sensitivity	Not case sensitive with commands and file/folder names	Case sensitive with commands and file/directory names

References and Further Readings

- 1 2001 Introduction to UNIX : course outline (<http://www.doc.ic.ac.uk/~wjk/UnixIntro/Lecture1.html>). Visited last : 21/10/2005
- 2 2005 LINUX vs. Windows. (<http://www.michaelhorowitz.com/Linux.vs.Windows.html>). Visited last : 25/10/Windows.
- 3 2005 Windows NT. ([http://en.wikipedia.org/wiki/Windows NT](http://en.wikipedia.org/wiki/Windows_NT)). Visited last : 25/10/2005

11.4 Exercise

1. Discuss architecture of Linux.
2. What is a multi tasking operating system?
3. Discuss six basic elements of Unix.

Unit 12 □ Programming Languages and Algorithm

Structure

12.0 Objectives

12.1 Programming Languages

12.1.1 Introduction

12.1.2 Definition

12.1.3 Features

12.1.4 History

12.1.5 Classification

12.1.6 Programme Translation Sequence

12.2 Algorithm

12.3 Exercise

12.0 Objectives

The objectives of the Unit are to :

- Explain the concept of programming language
- Examine the features of programming language
- Discuss classification of programming languages
- Present an overview of history of programming language
- Trace stages of programme translation
- Introduce the concept of Algorithm

12.1 Programming Languages

12.1.1 Introduction

Programming Language is an artificial language used to write a sequence of instructions that can be run by a computer. Similar to natural languages, such as Bengali, programming languages have a vocabulary, grammar, and syntax. However, natural languages are not suited for programming computers because their vocabulary and grammatical structure may be interpreted in multiple ways. The languages used to programme computers must have simple logical structures, and the rules for their grammar, spelling, and punctuation must be precise.

Programming languages allow people to communicate with computers. Once a job has been identified, the programmer must translate, or code it into a list of instructions that the computer will understand. A computer programme for a given task may be written in several different languages. Depending on the task, a programmer will generally pick the language that will involve the least complicated programme.

12.1.2 Definition

A programming language or computer language is a standardized communication technique for expressing instructions to a computer. It is a set of syntactic and semantic rules used to define computer programmes. A language enables a programmer to precisely specify what data a computer will act upon, how these data will be stored/transmitted, and precisely what actions to take under various circumstances.

12.1.3 Features of programming language

Programming languages use specific types of statements, or instructions, to provide functional structure to the programme. A statement in a programme is a basic sentence that expresses a simple idea—its purpose is to give the computer a basic instruction. Statements define the types of data allowed, how data are to be manipulated, and the ways that procedures and functions work. Programmers use statements to manipulate common components of programming languages, such as variables and macros (mini-programmes within a programme).

Statements known as data declarations give names and properties to elements of a programme called variables. Variables can be assigned different values within the programme. The properties variables can have are called types, and they include such things as what possible values might be saved in the variables, how much numerical accuracy is to be used in the values, and how one variable may represent a collection of simpler values in an organized fashion, such as a table or array. In many programming languages, a key data type is a pointer. Variables that are pointers do not themselves have values; instead, they have information that the computer can use to locate some other variable—that is, they point to another variable.

Each programming language can be thought of as a set of formal specifications concerning syntax, vocabulary, and meaning. These specifications usually include :

- Data and Data Structures
- Instruction and Control Flow
- Reference Mechanisms and Re-use
- Design Philosophy

12.1.3.1 Data types

Internally, all data in modern digital computers are stored in binary format. The data typically represent information in the real world such as names, bank accounts and measurements and so the low-level binary data are organized by programming languages into these high-level concepts. The particular system by which data are organized in a programme is the *type system* of the programming language. Languages can be classified as *statically typed* systems, and *dynamically typed* languages.

With **statically typed languages**, there usually are pre-defined types for individual pieces of data (such as numbers within a certain range, strings of letters, etc.), and programmatically named values (variables) can have only one fixed type, and allow only certain operations : numbers cannot change into names and vice versa. Most mainstream statically typed languages, such as C, C++, C#, Java and Delphi, require all types to be specified explicitly; advocates argue that this makes the programme easier to understand.

Dynamically typed languages treat all data locations interchangeably, so inappropriate operations (like adding names, or sorting numbers alphabetically) will not cause errors until run-time-although some implementations provide some form of static checking for obvious errors. Examples of these languages are Objective-C, Lisp, Smalltalk, JavaScript, Tcl, Prolog, Python, and Ruby.

12.1.3.2 Data structures

Most languages also provide ways to assemble complex data structures from built-in types and to associate names with these new combined types (using arrays, lists, stacks, files).

Object oriented languages allow the programmer to define data-types called “Objects” which have their own intrinsic functions and variables (called methods and attributes respectively). A programme containing objects allows the objects to operate as independent but interacting sub-programmes : this interaction can be designed at coding time to model or simulate real-life interacting objects. This is a very useful, and intuitive, functionality. Languages such as Python and Ruby have developed as OO (Object oriented) languages.

12.1.3.3 Instruction and control flow

Once data has been specified, the machine must be instructed how to perform operations on the data. Elementary statements may be specified using keywords or may be indicated using some well-defined grammatical structure.

Each language takes units of these well-behaved statements and combines them using some ordering system. Depending on the language, different methods of grouping these elementary statements exist. This allows one to write programmes that are able to cover a variety of input, instead of being limited to a small number of cases. Furthermore, beyond the data manipulation instructions, other typical instructions in a language are those used for control flow (branches, definitions by cases, loops, backtracking, functional composition).

12.1.3.4 Design philosophies

Since programming languages are artificial languages, they require a high degree of discipline to accurately specify which operations are desired. Programming languages are not error tolerant; however, the burden of recognizing and using the special vocabulary is reduced by help messages generated by the programming language implementation. There are a few languages which offer a high degree of freedom in allowing self-modification in which a programme re-writes parts of itself to handle new cases. Typically, only machine language, Prolog, PostScript, and the members of the Lisp family (Common Lisp, Scheme) provide this capability.

12.1.3.5 Compiled and interpreted languages

Computer programmes written in any language other than machine language must be either interpreted or compiled. An interpreter is software that examines a computer programme one instruction at a time and calls on code to execute the operations required by that instruction. This is a rather slow process. A compiler is software that translates a computer programme as a whole into machine code that is saved for subsequent execution whenever desired. Much work has been done on making both the compilation process and the compiled code as efficient as possible. When a new language is developed, it is usually at first interpreted. If the language becomes popular, it becomes important to write compilers for it, although this may be a task of considerable difficulty. There is an intermediate approach, which is to compile code not into machine language but into an intermediate language that is close enough to machine language that it is efficient to interpret-though not so close that it is tied to the machine language of a particular computer. It is use of this approach that provides the Java language with its computer-platform independence.

Programming languages generally fall into one of two categories : compiled or interpreted. With a compiled language, code is reduced to a set of machine-specific instructions before being saved as an executable file. With interpreted languages, the code is saved in the same format that you entered. Compiled programmes generally run faster than interpreted ones because interpreted programmes must be reduced to

machine instructions at runtime. However, with an interpreted language you can do things that cannot be done in a compiled language. For example, interpreted programmes can modify themselves by adding or changing functions at runtime. It is also usually easier to develop applications in an interpreted environment because you don't have to recompile your application each time you want to test a small section.

12.1.4 History of programming languages

Charles Babbage is often credited with designing the first computer-like machines, which had several programmes written for them (in the equivalent of assembly language) by Ada Lovelace. However, in the 1940s the first recognizably modern, electrically powered computers were created.

Subsequent breakthroughs in electronic technology (transistors, integrated circuits, and chips) drove the development of increasingly reliable and more usable computers. The first widely used high level programming language was FORTRAN, developed during 1954-57 by an IBM team led by John W. Backus. It is still widely used for numerical work, with the latest international standard released in 2004.

In the late 1960s, the first object-oriented languages, such as SIMULA, emerged. Logic languages became well known in the mid 1970s with the introduction of PROLOG, a language used to programme artificial intelligence software. During the 1970s, procedural languages continued to develop with ALGOL, BASIC, PASCAL, C, and Ada. SMALLTALK was a highly influential object-oriented language that led to the merging of object-oriented and procedural languages in C++ and more recently in JAVA. Although pure logic languages have declined in popularity, variations have become vitally important in the form of relational languages for modern databases, such as SQL (Structured Query Language).

Dennis Ritchie and Brian Kernighan developed the C programming language, initially for DEC PDP-11 in 1970. Later with lead of Bjarne Stroustrup the programming language C++ appeared starting from 1985 as the Object oriented language vertically compatible with C. Sun Microsystems developed Java in 1995 which became very popular as the initial programming language taught at universities. Microsoft presented the C# programming language, bringing garbage collection, generic types, and introspection to a language that C++ programmers could learn easily, in 2001. There are other languages such as Python, Visual Basic, etc..

12.1.5 Classification

Programming languages can be classified as either low-level languages or highlevel languages. Low-level programming languages, or machine languages, are

the most basic type of programming languages and can be understood directly by a computer. Machine languages differ depending on the manufacturer and model of computer. High-level languages are programming languages that must first be translated into a machine language before they can be understood and processed by a computer. Examples of high-level languages are C, C++, PASCAL, and FORTRAN. Assembly languages are intermediate languages that are very close to machine language and do not have the level of linguistic sophistication exhibited by other high-level languages, but must still be translated into machine language.

12.1.5.1 Machine Languages

In machine languages, instructions are written as sequences of 1s and 0s, called bits, that a computer can understand directly. An instruction in machine language generally tells the computer four things :

1. Where to find one or two numbers or simple pieces of data in the main computer memory (Random Access Memory, or RAM)
2. A simple operation to perform, such as adding the two numbers together,
3. Where in the main memory to put the result of this simple operation, and
4. Where to find the next instruction to perform.

While the computer in machine language eventually reads all executable programmes, they are not all programmed in machine language. It is extremely difficult to programme directly in machine language because the instructions are sequences of 1s and 0s. A typical instruction in a machine language might read 10010 1100 1011 and mean add the contents of storage register A to the contents of storage register B.

12.1.5.2 Assembly Language

Computer programmers use assembly languages to make machine-language programmes easier to write. In an assembly language, each statement corresponds roughly to one machine language instruction. An assembly language statement is composed with the aid of easy to remember commands. The command to add the contents of the storage register A to the contents of storage register B might be written ADD B, A in a typical assembly language statement. Assembly languages share certain features with machine languages. For instance, it is possible to manipulate specific bits in both assembly and machine languages. Programmers use assembly languages when it is important to minimize the time it takes to run a programme, because the translation from assembly language to machine language is relatively simple. Assembly languages are also used when some part of the computer has to be

controlled directly, such as individual dots on a monitor or the flow of individual characters to a printer.

12.1.5.3 High-Level Languages

High-level languages are relatively sophisticated sets of statements utilizing words and syntax from human language. They are more similar to normal human languages than assembly or machine languages and are therefore easier to use for writing complicated programmes. These programming languages allow larger and more complicated programmes to be developed faster. However, high-level languages must be translated into machine language by another programme called a compiler before a computer can understand them. For this reason, programmes written in a high-level language may take longer to execute and use up more memory than programmes written in an assembly language.

12.1.5.3.1 Classification of High Level Languages

High-level languages are commonly classified as procedure-oriented, functional, object-oriented, or logic languages. The most common high-level languages today are procedure-oriented languages. In these languages, one or more related blocks of statements that perform some complete function are grouped together into a programme module, or procedure, and given a name such as “procedure A.” If the same sequence of operations is needed elsewhere in the programme, a simple statement can be used to refer back to the procedure. In essence, a procedure is just a mini-programme. A large programme can be constructed by grouping together procedures that perform different tasks. Procedural languages allow programmes to be shorter and easier for the computer to read, but they require the programmer to design each procedure to be general enough to be used in different situations.

Functional languages treat procedures like mathematical functions and allow them to be processed like any other data in a programme. This allows a much higher and more rigorous level of programme construction. Functional languages also allow variables—symbols for data that can be specified and changed by the user as the programme is running—to be given values only once. This simplifies programming by reducing the need to be concerned with the exact order of statement execution, since a variable does not have to be redeclared, or restated, each time it is used in a programme statement. Many of the ideas from functional languages have become key parts of many modern procedural languages.

Object-oriented languages are outgrowth of functional languages. In object-oriented languages, the code used to write the programme and the data processed by

the programme are grouped together into units called objects. Objects are further grouped into classes, which define the attributes objects must have. A simple example of a class is the class Book. Objects within this class might be Novel and Short Story. Objects also have certain functions associated with them, called methods. The computer accesses an object through the use of one of the object's methods. The method performs some action to the data in the object and returns this value to the computer. Classes of objects can also be further grouped into hierarchies, in which objects of one class can inherit methods from another class. The structure provided in object-oriented languages makes them very useful for complicated programming tasks.

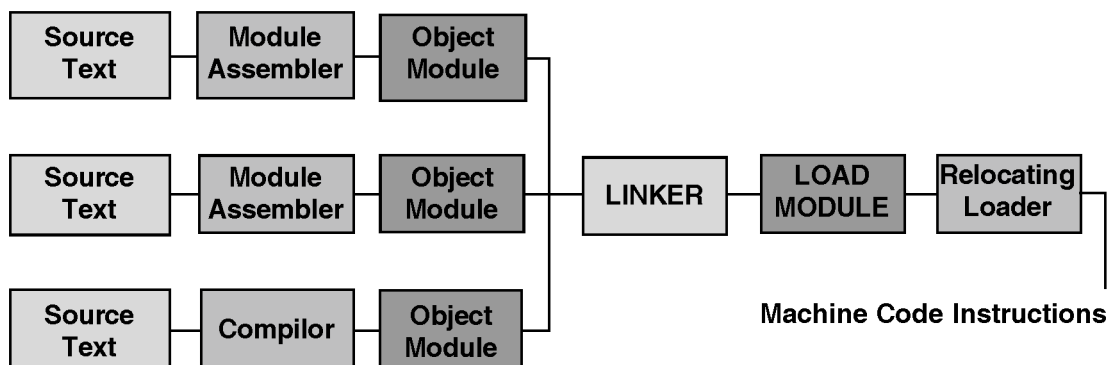
Logic languages use logic as their mathematical base. A logic programme consists of sets of facts and if-then rules, which specify how one set of facts may be deduced from others, for example :

If the statement X is true, then the statement Y is false. In the execution of such a programme, an input statement can be logically deduced from other statements in the programme. Many artificial intelligence programmes are written in such languages.

12.1.6 Programme Translation Sequence

In developing a software programme to accomplish a particular task, the programme developer chooses an appropriate language, develops the algorithm (a sequence of steps, which when carried out in the order prescribed, achieve the desired result), implements this algorithm in the chosen language (coding), then tests and debugs the final result. There is also a probable maintenance phase.

When you write a programme in a source language such as Pascal or C, the programme statements (in the source text file) need to be converted into the binary bit-patterns which make sense to the target processor (the processor on which the software will be run). This process of conversion is called translation.



12.1.6.1 Assemblers

Assemblers are programmes that generate machine code instructions from a source code programme written in assembly language. The features provided by an assembler are :

- Allows the programmer to use mnemonics when writing source code programmes.
- Variables are represented by symbolic names, not as memory locations
- Symbolic code is easier to read and follow
- Error checking is provided
- Changes can be quickly and easily incorporated with a re-assembly
- Programming aids are included for relocation and expression evaluation.

A mnemonic is an abbreviation for a machine code statement. During the translation phase, each mnemonic is translated to an equivalent machine code instruction.

```
MOV AX, offh
is translated to the binary bit patterns
10111000 (this means MOV AX)
11111111 (this is ff hexadecimal)
00000000 (this is 0)
```

Assemblers also provide keywords called pseudo-ops. These keywords provide directions (hence they are also called assembler directives) to the assembler. Pseudo-ops do not generate machine instructions. The following pseudo-op

```
DB 'ab'
```

allocates and initializes a byte of storage for each character of the string, thus two bytes will be allocated, one initialized to the character 'a' whilst the other byte would be initialized to the character 'b'.

The assembler does not normally generate executable code. An assembler produces an object code file that must be further processed (linked) in order to generate a file that can be executed directly.

12.1.6.2 Interpreter

The source code programme is run through a programme called an interpreter. Each line of the programme is sent to the interpreter that converts it into equivalent machine code instructions. These machine code instructions are then executed. The next source line is then fetched from memory, converted and executed. This process is repeated till the entire programme has been executed.

Examples of interpreted languages are BASIC (Beginners All Purpose Symbolic Instruction Code) and Java.

12.1.6.3 Compiler

Compilers accept source programmes written in a high level language and produce object code programmes that are then linked with standard libraries to produce an executable file. Compilers generate code that is reasonably fast, but is target specific (it only runs on a particular computer system)

The source programme is

```
Written using an editor.      # include <stdio. h>
Most compiled                main {
languages do not use         printf ("Hello world/n');
line numbers. The           return 1;
example on the right is     }
```

a C programme.

Once the programme has been written using the appropriate source statements, it is then passed to a compiler that converts the entire programme into object code. The object code cannot be run on the computer system, so the object code file is then sent to a linker that combines it with libraries (other object code) to create an executable programme. Because the entire programme is converted to machine code, it runs very quickly.

12.1.6.4 Linker

The BASIC interpreter already has its own libraries for Input and Output (I/O), so BASIC programmes don't need linking. The source programme is converted directly into executable code.

Compiled languages (as well as assembled) need both linking and loading. The output of compilers and assemblers are stored in an intermediate format called object code. This is stored as a file on disk. The object code must be combined with other object code files or libraries (special object files) before execution.

The linker combines the programmes object code with the runtime object code files (for handling files, screen output, the keyboard etc) into an executable format. The types of files that exist at each phase of the programme translation sequence are,

myprog.c	source code programme
myprog.obj	object code produced by compiler
myprog.exe	executable file produced by linker

12.1.6.5 Loaders

It is normally the responsibility of the Operating System to load and execute files.

The part of the operating system that performs this function is called a loader. There are two types of loaders, relocating and absolute.

The absolute loader is the simplest and quickest of the two. The loader loads the file into memory at the location specified by the beginning portion (header) of the file, and then passes control to the programme. If the memory space specified by the header is currently in use, execution cannot proceed, and the user must wait until the requested memory becomes free.

The relocating loader will load the programme anywhere in memory, altering the various addresses as required to ensure correct referencing. The decision as to where in memory the programme is placed is done by the Operating System, not the programmes header file. This is obviously more efficient, but introduces a slight overhead in terms of a small delay whilst all the relative offsets are calculated. The relocating loader can only relocate code that has been produced by a linker capable of producing relative code. A loader is unnecessary for interpreted languages, as the executable code is built up into the memory of the computer.

12.1.6.6 Locator

Programme locators convert the output of the linker (the executable file) into an absolute load format file. This type of file will eventually reside in specific memory locations, and is used to embed software into EPROM chips.

12.6.6.7 Cross Reference Utility (CRef)

These allow the programmer to generate a table that lists all symbols, labels, names, modules etc. Each occurrence is listed, and generally the source programme is given line numbers to facilitate this process.

The Cref utility should detect data variables and assign symbols to them, presenting a variety of formats (by name, module etc). The Cref table is useful in debugging, as the programmer can ascertain in which modules a particular variable is referenced.

12.6.6.8 Disassembler

Disassemblers convert machine code instructions into mnemonic opcodes and operands, facilitating debugging at the machine code level. The more sophisticated disassemblers provide for

- Generation of symbols and labels
- Cross reference tools

- Disassembly of memory or disk files
- Output of disassembly to disk file
- Relocation information

12.6.6.9 Debuggers and Monitors

A monitor is a small programme that allows machine code access. A monitor provides, Debuggers provide much the same facilities as monitors, but generally provide a wider range of features,

- provision for HLL source debugging
- split screens, windowing
- reference by symbols, module names and labels
- radix changing
- Dynamic tracing of hardware interrupts
- Operating system calls and stack tracing

12.6.6.10 Cross Assembler

Cross assemblers allow a programmer to develop machine code programmes on one computer system for another system (target). In this way, a programmer can develop a machine code programme for a Macintosh computer system using an IBM-PC. The cross-assembler running on the PC generates the machine code instructions necessary for the Macintosh.

12.2 Algorithm

12.2.1 Introduction

The word algorithm comes from the name of the 9th century Persian mathematician Abu Abdullah Muhammad bin Musa al-Khwarizmi. The word algorism originally referred only to the rules of performing arithmetic using Arabic numerals but evolved into algorithm by the 18th century. The word has now evolved to include all definite procedures for solving problems or performing tasks.

The first case of an algorithm written for a computer was Ada Byron's notes on the analytical engine written in 1842, for which she is considered by many to be the world's first programmer. However, since Charles Babbage never completed his analytical engine the algorithm was never implemented on it.

Algorithms are essential to the way computers process information, because a computer programme is essentially an algorithm that tells the computer what specific steps to perform (in what specific order) in order to carry out a specified task, such as calculating employees' paychecks or printing students' report cards. Thus, an

algorithm can be considered to be any sequence of operations which can be performed by a computer system.

Typically, when an algorithm is associated with processing information, data is read from an input source or device, written to an output device, and/or stored for further use. Stored data is regarded as part of the internal state of the entity performing the algorithm.

For any such computational process, the algorithm must be rigorously defined, specified in the way it applies in all possible circumstances that could arise. That is, any conditional steps must be systematically dealt with, case-by-case; the criteria for each case must be clear (and computable).

Because an algorithm is a precise list of precise steps, the order of computation will almost always be critical to the functioning of the algorithm. Instructions are usually assumed to be listed explicitly, and are described as starting ‘from the top’ and going ‘down to the bottom’, an idea that is described more formally by flow of control.

12.2.2 Definition

In computer science an algorithm is a finite set of well-defined instructions for accomplishing some task which, given an initial state, will terminate in a corresponding recognizable end-state. Algorithms often have steps that repeat (iterate) or require decisions (such as logic or comparison) until the task is completed. Correctly performing an algorithm will not solve a problem if the algorithm is flawed or not appropriate to the problem. Different algorithms may complete the same task with a different set of instructions in more or less time, space, or effort than others.

Nowadays, a formal criterion for an algorithm is that it is a procedure that can be implemented on a completely-specified Turing machine or one of the equivalent formalisms. Turing’s initial interest was in the halting problem : deciding when an algorithm describes a terminating procedure. In practical terms computational complexity theory matters more : it includes the problems called NP-complete, which are generally presumed to take more than polynomial time for any (deterministic) algorithm. NP denotes the class of decision problems that can be solved by a non-deterministic Turing machine in polynomial time.

Some writers restrict the definition of algorithm to procedures that eventually finish. Others include procedures that could run forever without stopping, arguing that some entity may be required to carry out such permanent tasks. In the latter case, success can no longer be defined in terms of halting with a meaningful output. Instead,

terms of success that allow for unbounded output sequences must be defined. For example, an algorithm that verifies if there are more zeros than ones in an infinite random binary sequence must run forever to be effective. If it is implemented correctly, however, the algorithm's output will be useful : for as long as it examines the sequence, the algorithm will give a positive response while the number of examined zeros outnumber the ones, and a negative response otherwise. Success for this algorithm could then be defined as eventually outputting only positive responses if there are actually more zeros than ones in the sequence, and in any other case outputting any mixture of positive and negative responses.

Certain countries, such as the USA, allow some algorithms to be patented, provided a physical embodiment is possible.

12.2.3 Classes

There are many ways to classify algorithms. One way of classifying algorithms is by their design methodology or paradigm. There is a certain number of paradigms, each different from the other. Furthermore, each of these categories will include many different types of algorithms. Some commonly found paradigms include :

- Divide and conquer
- Dynamic programming
- The greedy method
- Linear programming
- Others

Another way to classify algorithms is by implementation. A recursive algorithm is one that invokes (makes reference to) itself repeatedly until a certain condition matches, which is a method common to functional programming. Algorithms are usually discussed with the assumption that computers execute one instruction of an algorithm at a time. Those computers are sometimes called serial computers. An algorithm designed for such an environment is called a serial algorithm, as opposed to parallel algorithms, which take advantage of computer architectures where several processors can work on a problem at the same time. The various heuristic algorithms would probably also fall into this category, as their name (e.g. a genetic algorithm) describes its implementation.

References and Further Readings

- 1 2005 Algorithm (<http://en.wikipedia.org/wiki/Algorithm>). Visited last : 25/10/2005.

- 2 2005 Programming language (http://en.wikipedia.org/wiki/Programming_language#Features_of_programming_language). Visited last : 21/10/2005.
- 3 Lamont (Michael). Algorithm and data structure (<http://linux.wku.edu/~lamonml/kb.html>). Visited last : 25/10/2005.

12.3 Exercise

1. Discuss general features of computer programming languages.
2. Describe different generations of programming languages.
3. Discuss programme translation sequence.

Unit 13 □ Flowchart

Structure

13.0 Objectives

13.1 Introduction

13.2 Definition

13.3 Types of flowchart

13.4 Guidelines for drawing a flowchart

13.5 Advantages

13.6 Limitations

13.7 Creating flowchart on a computer

13.8 Examples

13.9 Exercise

13.0 Objectives

The objectives of the Unit are to understand the :

- Meaning of flowchart
- Basic parts of the flowchart such as flowchart symbols and the flow lines connecting these symbols.
- Advantages and limitations of flowchart

13.1 Introduction

The use of data processing equipment has focused attention upon the necessity for an orderly representation of information flow. The sequence in which operations are to be executed should be precisely stated. The data and sequence of operations to be performed upon the data together constitute the information flow.

A flow chart can be customized to fit any need or purpose. For this reason, flow charts can be recognized as a very unique quality improvement method. Flowcharts were used historically in electronic data processing to represent the conditional logic or computer programmes. The characteristics of flowcharts may be summarized as follows :

- A language to specify the logic of algorithms
- Flowcharts are not a programming language, rather it is an algorithm specification language.
- In an algorithm we specify the computation operations (arithmetic, logical, data input, data output) to be done, and their sequence.
- Each type of computation operation is denoted by a special symbol in the flowchart language
- Symbols are connected by arrows to represent the order of the operations (sequence)
- Flowchart are useful during algorithm design
- A flowchart can be relatively easily translated to a C++ programme

13.2 Definition

A flowchart (also spelled flow-chart and flow chart) is a schematic representation of a process. They are commonly used in business/economic presentations to help the audience to visualize the content better, or to find flaws in the process.

Information system flowcharts show how data flows from source documents through the computer to final distribution to users. Programme flowcharts show the sequence of instructions in a single programme or subroutine. Different symbols are used to draw each type of flowchart.

Four particular types of flow charts have proven useful when dealing with a process analysis : top-down flow chart, detailed flow chart, work flow diagrams, and a deployment chart. Each of the different types of flow charts tends to provide a different aspect to a process or a task. Flow charts provide an excellent form of documentation for a process, and quite often are useful when examining how various steps in a process work together.

13.3 Types of Flowchart

The pictorial representation of the programmes or the algorithm is known as flowcharts. It is nothing but a diagrammatic representation of the various steps involved in designing a system. Flowcharts are of three types :

- System flowcharts
- Run flowcharts
- Programme flowcharts

13.3.1 System Flowcharts

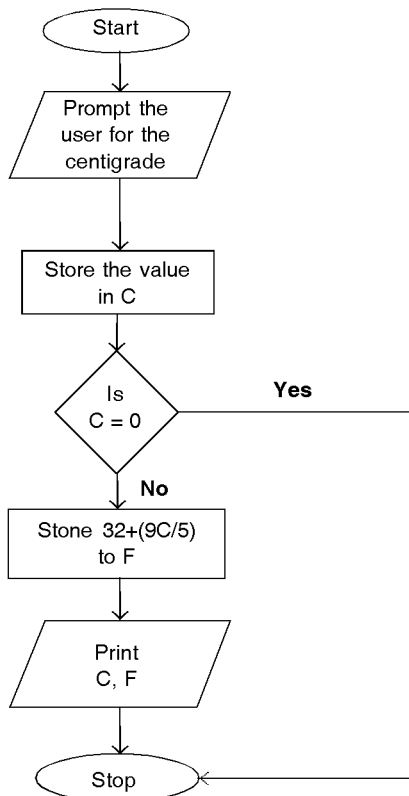
System flowchart describes the data flow for a data processing system. It provides a logical diagram of how the system operates. It represents the flow of documents, the operations performed in data processing system. It also reflects the relationship between inputs, processing and outputs. Following are the features of system flowcharts :

- The sources from which data is generated and device used for this purpose
- Various processing steps involved
- The intermediate and final output prepared and the devices used for their storage

The following is a sample of system flowchart for the following algorithm :

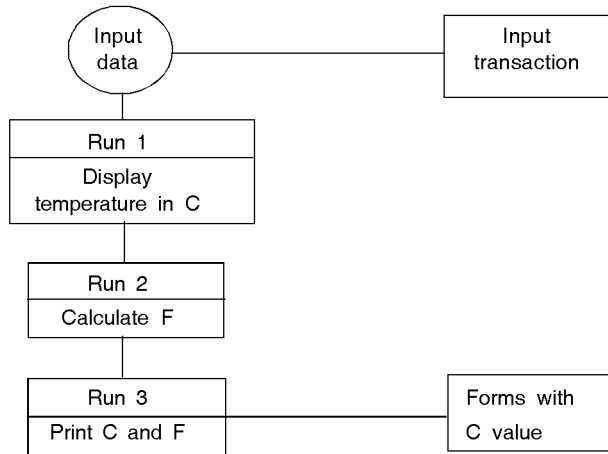
Prompt the user for the centigrade temperature.

1. Store the value in C
2. Set F to $32+(9 \cdot C/5)$
3. Print the value of C, F
4. Stop



13.3.2 Run flowcharts

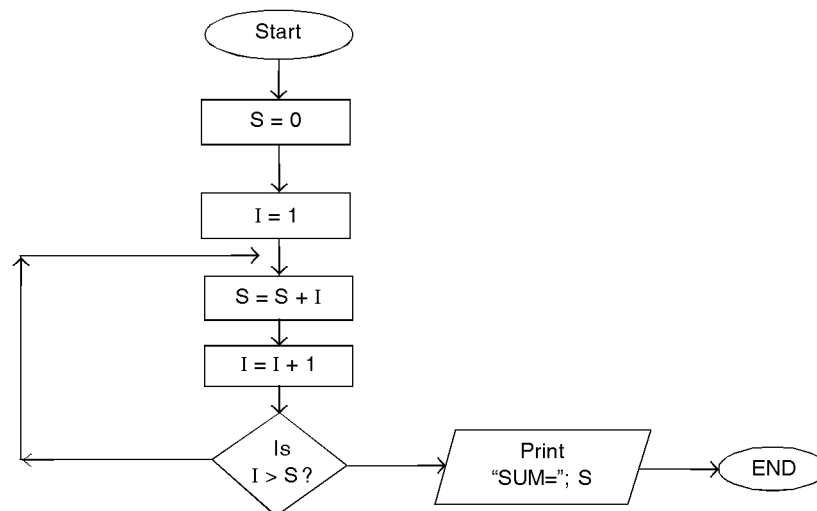
Run flowcharts are used to represent the logical relationship of computer routines along with inputs, master files, transaction files and outputs. The following illustrates a run flowchart.



13.3.3 Programme flowcharts

A programme flowchart represents, in detail, the various steps to be performed within the system for transforming the input into output. The various steps are logical/ arithmetic operations, algorithms etc. It serves as the basis for discussions and communication between the system analysts and the programmers. Programme flowcharts are quite helpful to programmers in organizing their programming efforts. These flowcharts constitute an important components of documentation for an application.




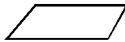


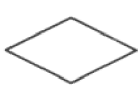








The following figure represents a programme flowcharts for finding the sum of first five natural numbers (1-5).



13.4 Guidelines for drawing a flowchart

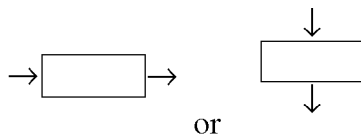
Flowcharting is a tool for analyzing processes. It allows breaking any process down into individual events or activities and to display these in shorthand form showing the logical relationships between them. Constructing flowcharts promotes better understanding of processes and better understanding of processes is a prerequisite for improvement.

Flowcharts are usually drawn using some standard symbols; however, some special symbols can also be developed when required. Some standard symbols, which are frequently, required for flowcharting many computer programmes are shown in the following table :

Symbols	Explanation
	Start or end of the programme
	Use it to represent an event, which occurs automatically. Such an event will trigger a subsequent action, for example 'receive telephone call', or describe a new state of affairs.
	Data
	Input or output operation
	Stored data
	Off-page connector
	Decision making and branching. Use it to represent a decision point in the process. Typically, the statement in the symbol will require a 'yes' or 'no' response and branch to different parts of the flowchart accordingly.
	Manual input
	Predefined process
	Connector or joining of two parts of programme. Use it to represent a point at which the flowchart connects with another process. The name or reference for the other process should appear within the symbol.
	Sequential Access Storage (Magnetic Tape)
	Magnetic Disk
	Direct Access storage
	Flow lines
	Display

There are no hard and fast rules for constructing flowcharts, but there are guidelines, which are useful to bear in mind. Here are six steps, which can be used as a guide for completing flowcharts.

- In drawing a proper flowchart, all necessary requirements should be listed out in logical order.
- The flowchart should be clear, neat and easy to follow. There should not be any room for ambiguity in understanding the flowchart.
- The usual direction of the flow of a procedure or system is from left to right or top to bottom.
- Only one flow line should come out from a process symbol.



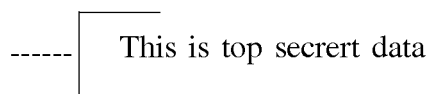
- Only one flow line should enter a decision symbol, but two or three flow lines, one for each possible answer, should leave the decision symbol.



- Only one flow line is used in conjunction with terminal symbol.



- Write within standard symbols briefly. As necessary, you can use the annotation symbol to describe data or computational steps more clearly.



- If the flowchart becomes complex, it is better to use connector symbols to reduce the number of flow lines. Avoid the intersection of flow lines if you want to make it more effective and better way of communication.
- Ensure that the flowchart has a logical start and finish.
- It is useful to test the validity of the flowchart by passing through it with a simple test data.

A flowchart consists of a set of ‘flowchart symbols’ connected by arrows. Each symbol contains information about what must be done at that point & the arrow shows the ‘flow of execution’ of the algorithm i.e. they show the order in which the instructions must be executed. The purpose of using flowcharts is to graphically present the logical flow of data in the system and defining major phases of processing along with the various media to be used.

13.5 Advantages

The benefits of flowcharts are as follows :

- Communication : Flowcharts are better way of communicating the logic of a system to all concerned.
- Effective analysis : With the help of flowchart, problem can be analyzed in more effective way.
- Proper documentation : Programme flowcharts serve as a good programme documentation, which is needed for various purposes.
- Efficient Coding : The flowcharts act as a guide or blueprint during the systems analysis and programme development phase.
- Proper Debugging : The flowchart helps in debugging process.
- Efficient Programme Maintenance : The maintenance of operating programme becomes easy with the help of flowchart. It helps the programmer to put efforts more efficiently on that part.

13.6 Limitations

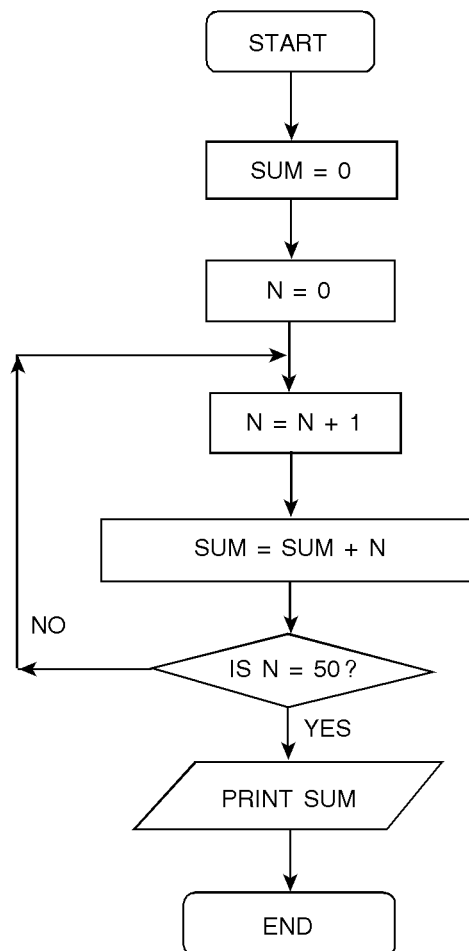
- Complex logic : Sometimes, the programme logic is quite complicated. In that case, flowchart becomes complex and clumsy.
- Alterations and Modifications : If alterations are required the flowchart may require re-drawing completely.
- Reproduction : As the flowchart symbols cannot be typed, reproduction of flowchart becomes a problem.
- The essentials of what is done can easily be lost in the technical details of how it is done.

13.7 Creating flowcharts on a computer

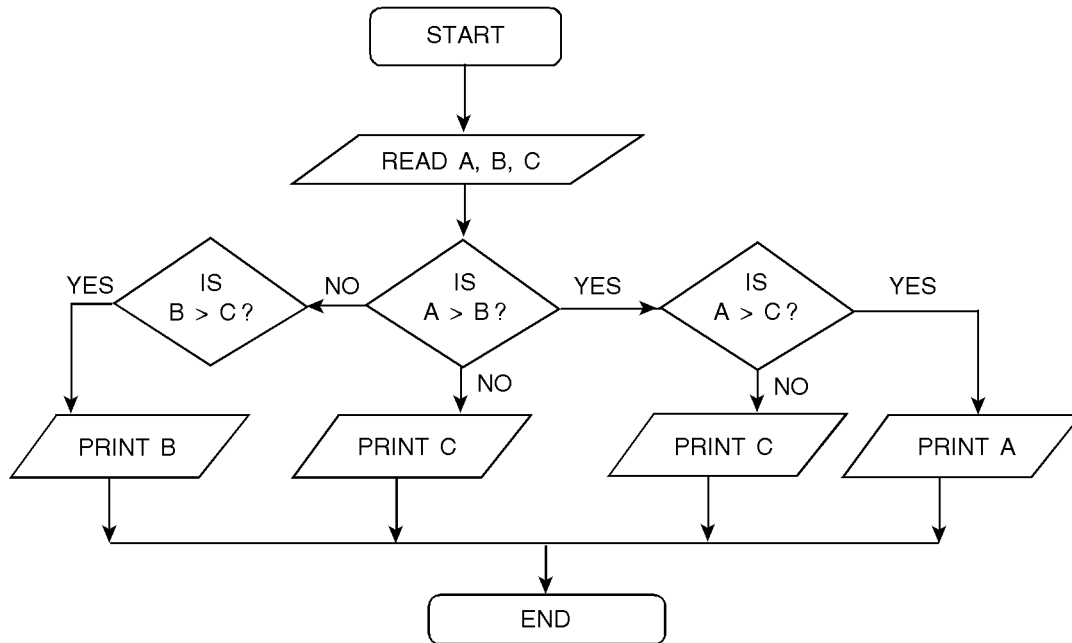
There are various packages for creating flowcharts, according to different standards. The most common is UML, for which there are abundant packages for various platforms. See UML article for list. The creation of simple flowcharts on a computer is fairly easy with any vector-based drawing programme, but Microsoft Word (versions 97 through 2003) and Openoffice. Org (Draw app) both have specialized tools for making consistent charts. For Mac OS X OmniGraffle is an excellent application.

13.8 Examples

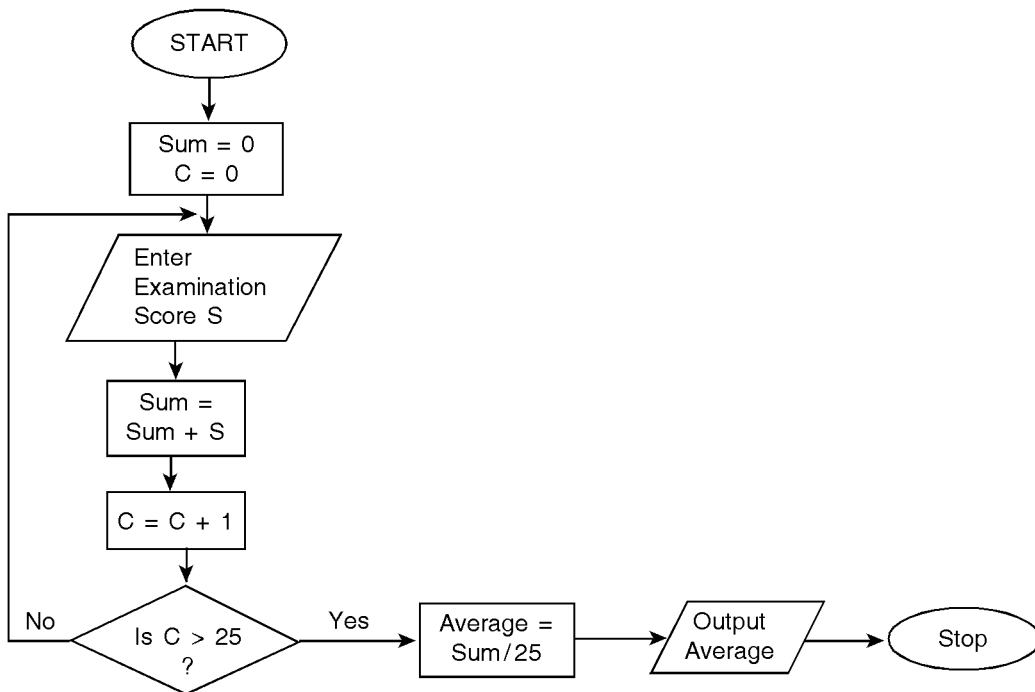
13.8.1 Draw a flowchart to find the sum of first 50 natural numbers.



13.8.2 Draw a flowchart to find the largest of three numbers A, B, and C.



13.8.3 Calculate the average from 25 examination scores



References and Further Readings

- 1 2005 NOS : Certificate in Computer Application. Lesson 25 : Flowcharting. (<http://nos.org/html/basic2.htm>). Visited last : 4/10/2005
- 2 2005 Flowcharts (<http://deming.eng.clemson.edu/pub/tutorials/qctools/flowm.htm>) Visited last : 4/10/2005
- 3 1970 Flowcharting techniques (NY : IBM, 1970). Document Code : C20-8152-1).

13.9 Exercise

1. What is flowchart? Describe various types of flowcharts.
2. Describe how to design flowchart of an activity.
3. Describe important flowchart symbols.

Unit 14 □ Search, Sorting Algorithm and Data Structure

Structure

14.0 Objectives

14.1 Introduction

14.2 Data Structure

14.3 Search and Sorting Algorithm

14.4 Classification

14.5 Summaries of the seven most popular sorting algorithms

14.5.1 Bubble sort

14.5.2 Heap sort

14.5.3 Insertion sort

14.5.4 Merge sort

14.5.5 Quick sort

14.5.6 Selection sort

14.5.7 Shell sort

14.6 Memory Usage Patterns and Index sorting

14.7 Exercise

14.0 Objectives

The objectives of the Unit are to :

- Present an overview of data structures
- Explain the concept behind sorting algorithms
- Identify the basic classes of sorting algorithms
- Present an overview of certain sorting algorithms
- Discuss the impacts of memory on the sorting algorithms

14.1 Introduction

Algorithms and data structures are the building blocks of computer programmes. Common algorithms include searching a collection of data, sorting data, and numerical

operations such as matrix multiplication. Data structures are patterns for organizing information, and often represent relationships between data values. Some common data structures are called lists, arrays, records, stacks, queues, and trees.

One of the fundamental problems is ordering a list of items. There's a plethora of solutions to this problem, known as sorting algorithms. Some sorting algorithms are simple and intuitive, such as the bubble sort. Others, such as the quick sort are extremely complicated, but produce lightening-fast results.

14.2 Data Structures

A major area of study in computer science has been the storage of data for efficient search and retrieval. The main memory of a computer is linear, consisting of a sequence of memory cells that are numbered 0, 1, 2, in order.

Similarly, the simplest data structure is the one-dimensional, or linear, array, in which array elements are numbered with consecutive integers and the element numbers may access array contents. Data items (a list of names, for example) are often stored in arrays, and efficient methods are sought to handle the array data.

A data structure is a way of storing data in a computer so that it can be used efficiently. A well-designed data structure allows a variety of critical operations to be performed on using as little resources, both execution time and memory space, as possible. Different kinds of data structures are suited to different kinds of applications, and some are highly specialized to certain tasks. Some common data structures are called lists, arrays, records, stacks, queues, and trees. For example, B-trees are particularly well-suited for implementation of databases, while routing tables rely on networks of machines to function.

In the design of many types of programmes, the choice of data structures is a primary design consideration, as experience in building large systems has shown that the difficulty of implementation and the quality and performance of the final result depends heavily on choosing the best data structure. After the data structures are chosen, the algorithms to be used often become relatively obvious. Sometimes things work in the opposite direction—data structures are chosen because certain key tasks have algorithms that work best with particular data structures. In either case, the choice of appropriate data structures is crucial.

This insight has given rise to many formalized design methods and programming languages in which data structures, rather than algorithms, are the key organizing factor. Most languages feature some sort of module system, allowing data structures to be safely reused in different applications by hiding their verified implementation

details behind controlled interfaces. Object-oriented programming languages such as C++ and Java in particular use objects for this purpose.

The fundamental building blocks of most data structures are arrays, records, discriminated unions, and references. There is some debate about whether data structures represent implementations or interfaces. How they are seen may be a matter of perspective. A data structure can be viewed as an interface between two functions or as an implementation of methods to access storage that is organized according to the associated data type.

14.3 Search and sorting algorithm

Search techniques must address, for example, how a particular name is to be found. One possibility is to examine the contents of each element in turn. If the list is long, it is important to sort the data first—in the case of names, to alphabetize them. Just as the alphabetizing of names in a telephone book greatly facilitates their retrieval by a user, the sorting of list elements significantly reduces the search time required by a computer algorithm as compared to a search on an unsorted list. Many algorithms have been developed for sorting data efficiently. These algorithms have application not only to data structures residing in main memory but even more importantly to the files that constitute information systems and databases.

In computer science a sorting algorithm is an algorithm that puts elements of a list in a certain order. The most used orders are numerical order and lexicographical order (alphabetic order). Efficient sorting is important to optimizing the use of other algorithms (such as search and merge algorithms) that require sorted lists to work correctly; it is also often useful for canonicalizing data and for producing human-readable output.

14.4 Classification

The common sorting algorithms can be divided into two classes by the complexity of their algorithms. There's a direct correlation between the complexity of an algorithm and its relative efficiency. Algorithmic complexity is generally written in a form known as Big-O notation, where the O represents the complexity of the algorithm and a value n represents the size of the set the algorithm is run against.

For example, $O(n)$ means that an algorithm has a linear complexity. In other words, it takes ten times longer to operate on a set of 100 items than it does on a set of 10 items ($10 * 10 = 100$). If the complexity was $O(n^2)$ (quadratic complexity), then it

would take 100 times longer to operate on a set of 100 items than it does on a set of 10 items.

The two classes of sorting algorithms are $O(n^2)$, which includes the bubble, insertion, selection, and shell sorts; and $O(n \log n)$ which includes the heap, merge, and quick sorts.

In addition to algorithmic complexity, the speed of the various sorts can be compared with empirical data. Since the speed of a sort can vary greatly depending on what data set it sorts, accurate empirical results require several runs of the sort be made and the results averaged together. The run times also depend on system configurations.

14.5 Summaries of the seven most popular sorting algorithms

One of the fundamental problems of computer science is ordering a list of items. There's a plethora of solutions to this problem, known as sorting algorithms. Some sorting algorithms are simple and intuitive, such as the bubble sort. Others, such as the quick sort are extremely complicated, but produce lightening-fast results. Below are descriptions to algorithms for seven of the most common sorting algorithms.

- Bubble sort
- Heap sort
- Insertion sort
- Merge sort
- Quick sort
- Selection sort
- Shell sort

14.5.1 Bubble Sort

The bubble sort is the oldest and simplest sort in use. Bubble sort is the most straightforward and simplistic method of sorting data that could actually be considered for real world use. The algorithm starts at the beginning of the data set. It compares the first two elements, and if the first is greater than the second, it swaps them. It continues doing this for each pair of adjacent elements to the end of the data set. It then starts again with the first two elements, repeating until no swaps have occurred on the last pass. This algorithm, however, is vastly inefficient, and is rarely used except in education (i.e., beginning programming classes). A slightly better variant is generally called cocktail sort, and works by inverting the ordering criteria and the pass

direction on alternating passes. The bubble sort is generally considered to be the most inefficient sorting algorithm in common usage.

A fair number of algorithm purists claim that the bubble sort should never be used for any reason. Realistically, there isn't a noticeable performance difference between the various sorts for 100 items or less, and the simplicity of the bubble sort makes it attractive. The bubble sort shouldn't be used for repetitive sorts or sorts of more than a couple hundred items.

14.5.2 Heap Sort

Heap sort is a member of the family of selection sorts. This family of algorithms works by determining the largest (or smallest) element of the list, placing that at the end (or beginning) of the list, then continuing with the rest of the list. Straight selection sort runs in $O(n^2)$ time, but Heap sort accomplishes its task efficiently by using a data structure called a heap, which is a binary tree where each parent is larger than either of its children. Once the data list has been made into a heap, the root node is guaranteed to be the largest element. It is removed and placed at the end of the list, and then the remaining list is "heapified" again. This is repeated until there are no items left in the heap and the sorted array is full. Elementary implementations require two arrays—one to hold the heap and the other to hold the sorted elements.

To do an in-place sort and save the space the second array would require, the algorithm below "cheats" by using the same array to store both the heap and the sorted array. Whenever an item is removed from the heap, it frees up a space at the end of the array that the removed item can be placed in.

The heap sort is the slowest of the $O(n \log n)$ sorting algorithms, but unlike the merge and quick sorts it doesn't require massive recursion or multiple arrays to work. This makes it the most attractive option for very large data sets of millions of items.

As mentioned above, the heap sort is slower than the merge and quick sorts but doesn't use multiple arrays or massive recursion like they do. This makes it a good choice for really large sets, but most modern computers have enough memory and processing power to handle the faster sorts unless over a million items are being sorted.

The "million item rule" is just a rule of thumb for common applications—high-end servers and workstations can probably safely handle sorting tens of millions of items with the quick or merge sorts.

14.5.3 Insertion Sort

Insertion sort is similar to bubble sort, but is more efficient as it reduces element comparisons somewhat with each pass. An element is compared to all the prior

elements until a lesser element is found. In other words, if an element contains a value less than all the previous elements, it compares the element to all the previous elements before going on to the next comparison. Although this algorithm is more efficient than the Bubble sort, it is still inefficient compared to many other sort algorithms since it, and bubble sort, move elements only one position at a time. However, insertion sort is a good choice for small lists (about 30 elements or fewer), and for nearly-sorted lists. These observations can be combined to create a variant of insertion sort which works efficiently for larger lists. This variant is called shell sort (see below).

The insertion sort works just like its name suggests—it inserts each item into its proper place in the final list. The simplest implementation of this requires two list structures—the source list and the list into which sorted items are inserted. To save memory, most implementations use an in-place sort that works by moving the current item past the already sorted items and repeatedly swapping it with the preceding item until it is in place.

Like the bubble sort, the insertion sort has a complexity of $O(n^2)$. Although it has the same complexity, the insertion sort is a little over twice as efficient as the bubble sort.

The insertion sort is a good middle-of-the-road choice for sorting lists of a few thousand items or less. The algorithm is significantly simpler than the shell sort, with only a small trade-off in efficiency. At the same time, the insertion sort is over twice as fast as the bubble sort and almost 40% faster than the selection sort. The insertion sort shouldn't be used for sorting lists larger than a couple thousand items or repetitive sorting of lists larger than a couple hundred items.

14.5.4 Merge Sort

Merge sort takes advantage of the ease of merging already sorted lists into a new sorted list. It starts by comparing every two elements (i.e. 1 with 2, then 3 with 4...) and swapping them if the first should come after the second. It then merges each of the resulting lists of two into lists of four, then merges those lists of four, and so on; until at last two lists are merged into the final sorted list.

The merge sort splits the list to be sorted into two equal halves, and places them in separate arrays. Each array is recursively sorted, and then merged back together to form the final sorted list. Like most recursive sorts, the merge sort has an algorithmic complexity of $O(n \log n)$.

Elementary implementations of the merge sort make use of three arrays—one for each half of the data set and one to store the sorted list in. The below algorithm merges

the arrays in-place, so only two arrays are required. There are non-recursive versions of the merge sort, but they don't yield any significant performance enhancement over the recursive algorithm on most machines.

The merge sort is slightly faster than the heap sort for larger sets, but it requires twice the memory of the heap sort because of the second array. This additional memory requirement makes it unattractive for most purposes-the quick sort is a better choice most of the time and the heap sort is a better choice for very large sets. Like the quick sort, the merge sort is recursive which can make it a bad choice for applications that run on machines with limited memory.

14.5.5 Quick Sort

Quicksort takes an element from an array and places it in its final position whilst at the same time partitioning the array so that all elements above the partition element are larger and all elements below are smaller. It then sorts each sub-array (partition) recursively. The sort is usually implemented by scanning p from the bottom of an array until an element larger than the partition element is found, then scanning down from the top for an element smaller than the partition element; these two elements are then swapped. When the scans from bottom and top meet the partition element is swapped into its final position. Quicksort is an in place sort so it has modest memory requirements and does not involve copying, if implemented carefully bad / worst case performance is extremely unlikely, and quicksort is probably the fastest sorting method. Quicksort runs in $O(n \log n)$ time.

As mentioned earlier, it's massively recursive (which means that for very large sorts) you can run the system out of stack space pretty easily. It's also a complex algorithm-a little too complex to make it practical for a one-time sort of 25 items, for example. Ironically, the quick sort has horrible efficiency when operating on lists that are mostly sorted in either forward or reverse order-avoid it in those situations.

14.5.6 Selection Sort

The selection sort works by selecting the smallest unsorted item remaining in the list, and then swapping it with the item in the next position to be filled. The selection sort has a complexity of $O(n^2)$.

The selection sort is the unwanted stepchild of the n^2 sorts. It yields a 60% performance improvement over the bubble sort, but the insertion sort is over twice as fast as the bubble sort and is just as easy to implement as the selection sort. In

short, there really isn't any reason to use the selection sort-use the insertion sort instead.

If you really want to use the selection sort for some reason, try to avoid sorting lists of more than a 1000 items with it or repetitively sorting lists of more than a couple hundred items.

14.5.7 Shell Sort

Donald Shell invented shell sort in 1959. It improves upon bubble sort and insertion sort by moving out of order elements more than one position at a time. One implementation can be described as arranging the data sequence in a two dimensional array (in reality, the array is an appropriately indexed one dimensional array) and then sorting the columns of the array using the Insertion sort method. Although this method is inefficient for large data sets, it is one of the fastest algorithms for sorting small numbers of elements (sets with less than 1000 or so elements). Another advantage of this algorithm is that it requires relatively small amounts of memory. The shell sort is the most efficient of the $O(n^2)$ class of sorting algorithms. Of course, the shell sort is also the most complex of the $O(n^2)$ algorithms.

The algorithm makes multiple passes through the list, and each time sorts a number of equally sized sets using the insertion sort. The size of the set to be sorted gets larger with each pass through the list, until the set consists of the entire list. (Note that as the size of the set increases, the number of sets to be sorted decreases). This sets the insertion sort up for an almost-best case run each iteration with a complexity that approaches $O(n)$.

The items contained in each set are not contiguous-rather, if there are i sets then a set is composed of every i -th element. For example, if there are 3 sets then the first set would contain the elements located at positions 1, 4, 7 and so on. The second set would contain the elements located at positions 2, 5, 8, and so on; while the third set would contain the items located at positions 3, 6, 9, and so on.

The shell sort is by far the fastest of the N^2 class of sorting algorithms. It's more than 5 times faster than the bubble sort and a little over twice as fast as the insertion sort, its closest competitor.

The shell sort is still significantly slower than the merge, heap, and quick sorts, but its relatively simple algorithm makes it a good choice for sorting lists of less than 5000 items unless speed is hypercritical. It's also an excellent choice for repetitive sorting of smaller lists.

14.5.8 Comparative Chart

The following table presents an overview of certain parameters of the above-discussed sort algorithms.

Sorting Algorithms	Class	Advantages	Limitations
Bubble Sort	$O(n^2)$	Simplicity and ease of implementation.	Horribly inefficient.
Heap Sort	$O(n \log n)$	In-place and non-recursive, making it a good choice for extremely large data sets.	Slower than the merge and quick sorts.
Insertion Sort	$O(n^2)$	Relatively simple and easy to implement.	Inefficient for large lists.
Merge Sort	$O(n \log n)$	Marginally faster than the heap sort for larger sets.	At least twice the memory requirements of the other sorts; recursive.
Quick Sort	$O(n \log n)$	Extremely fast.	Very complex algorithm, massively recursive.
Selection Sort	$O(n^2)$	Simple and easy to implement.	Inefficient for large lists, so similar to the more efficient insertion sort that the insertion sort should be used in its place.
Shell Sort	$O(n^2)$	Efficient for medium size lists.	Somewhat complex algorithm, not nearly as efficient as the merge, heap, and quick sorts.

14.6 Memory Usage Patterns and Index Sorting

When the size of the array to be sorted approaches or exceeds the available primary memory, so that disk or swap space must be employed, the memory usage pattern of a sorting algorithm becomes important, and an algorithm that might have been fairly efficient when the array fit easily in RAM may become impractical. In this scenario,

the total number of comparisons becomes (relatively) less important, and the number of times sections of memory must be copied or swapped to and from the disk can dominate the performance characteristics of an algorithm.

For example, the popular recursive quicksort algorithm provides quite reasonable performance with adequate RAM, but due to the recursive way that it copies portions of the array it becomes much less practical when the array does not fit in RAM.

References and Further Readings

- 1 2005 Algorithm and data structure. (<http://linux.wku.edu/~lamonml/algor>). Visited last : 26/10/2005.
- 2 1996 Majumder (Arun K) and Bhattacharyya (Pritimoy). database management systems. New Delhi : Tata McGraw-Hill, 1996.
- 3 1994 Elmasri (Ramez) and Navathe (Shamkant B). Fundamentals of database systems. California : Benjamin/Cummings, 1994.
- 4 1985 Data (CJ). An Introduction to database systems. 3rd ed. New Delhi : Narosa, 1985.
- 5 1983 Date (CJ). Database : a primer. Reading : Addison-Wesley, 1983.

14.7 Exercise

1. Discuss different search algorithms.
2. What is an algorithm ?

Unit 15 □ Resource Sharing through Networks

Structure

15.0 Objectives

15.1 Introduction

15.2 Concept

15.3 Inpetus for Resource Sharing

15.4 Benefits of Resource Sharing

15.5 Barriers to Resource Sharing

15.6 Classification of Resource Sharing Networks

15.7 Infrastructure of Resource Sharing

15.8 Resource Description in the Digital Age

15.9 Resource sharing in a virtual library environment

15.10 Copyright in Networked Electronic Publishing

15.11 Open Access Initiative

15.12 Exercise

15.0 Objectives

The objectives of the Unit are to :

- Enumerate the concept of resource sharing
- Discuss the benefits and barriers of resource sharing
- Identify potential areas/functions for resource sharing
- Present an overview of resource sharing tools

15.1 Introduction

Consortium, network, and cooperative are the terms used to address the organizational arrangements for achieving a variety of resource sharing objectives. It is apparent that no library can provide all of the materials needed by its users. It is absolutely necessary to share resources among institutions. Internet provides convenient access to most library catalogues and national bibliographic databases with location information. This has facilitated interlibrary loan activities. New breakthroughs in networking, improvements in electronic transmission of data make resource sharing

a viable alternative. Automation has greatly accelerated the growth of resource sharing. As access to resources improved and the ability to request materials was streamlined, resource-sharing activity skyrocketed. Resource sharing is a well-developed concept in libraries of advanced countries for several decades. However, libraries in India are reassessing and reorganizing their collection development goals and service strategies, as they make the painful transition from the affluence to the austerity.

15.2 Concept

Resource sharing is a concept, which has developed to include many cooperative activities between libraries and other stakeholders. Interlibrary loan continues to be the mainstay of resource sharing. However, union catalogue development, cooperative cataloguing, cooperative reference, cooperative collection development and joint storage of material are all components of Resource Sharing Strategy.

Resource sharing is comprised of transactions by which a library makes its materials or copies of its materials available to the clientele of another library upon request. "Materials" are specific, identified items in any format in library collections, including returnable items (e.g., books and videocassettes) and nonreturnable items (e.g., photocopies and faxes). Resource sharing should at least include :

- Inter Library Loan/Document Delivery
- Reciprocal Borrowing
- On-site/Remote access to reference/information services and collections, and
- Reference and research support and collaboration among library staff.

15.3 Impetus for Resource Sharing

Resource sharing and interlibrary loan are adjunct to, not substitutes for, collection development in individual libraries. The exchange of materials between libraries is an important element in the provision of library service and it is believed to be in the public interest to encourage such an exchange. Selected impetuses are :

- Pressure for enhanced collaboration :
 - Information explosion : individual library cannot provide everything
 - Rising costs of library materials without increasing budgets
 - Potential of information technology
- Library provision not a key issue for most institutions

15.4 Benefits of Resource Sharing

The economic benefits of resource sharing may be summarized as follows :

- **Cost** : No library can afford to acquire all needed materials due to high procurement costs, and cost of processing, servicing, and storage (space and furniture do cost a lot)
- **Speed** : Application of technology ensures effective and efficient delivery of resources at the point of use. Thus, it will reduce the time needed for development of technology etc.
- **Rationalization of investment** : Cooperative acquisition ensures higher availability of resources among the participating libraries with same investment for procurement etc.
- **Reduced Overhead Cost** : Shared cataloguing using network resources (copy cataloguing etc.) reduces expenditure for information processing.

15.5 Barriers to Resource Sharing

The resources that could be shared between libraries are space, collection (development and use), staff (development and sharing), and technology. The barriers to resource sharing may be summarized as follows :

- Reluctance to give up ownership (size matters!)
- Reluctance to take risks (e.g. to divert spending)
- Reluctance to compromise (e.g. on specification for system)
- Fear (on the whole unfounded) of swamping
- Reluctance to provide professional leadership
- Discovery still a barrier
 - Resources remain uncatalogued
 - Catalogues remain unconverted
 - No single/virtual single national catalogue
- Need for more flexible approaches to e-licences for consortia working
- Widespread support for further digitisation of resources, to be shared
- Need for further work on digital archiving
- Almost all libraries have internal focus.
- Lack of fund and expertise for library automation
- Long distance between participating libraries and poor communication facility.
- Lack of exposure in comparative and international librarianship

15.6 Classification of Resource Sharing Networks

Although there is no operational example of resource sharing network, which includes all possible library functions, there are now in existence so many cooperative resource sharing activities and/or networks that it has become possible and instructive to classify them among a number of different dimensions, such as :

- Housekeeping Operations
 - Acquisition
 - Processing
 - Storage
- User Services
 - Reference
 - Document Delivery
- Type of libraries
- Subject matter
- Type of materials
- Forms of materials
- Nature of cooperative arrangement

The resource sharing networks may also be classified on the basis of distribution of resources. They are :

- Equally Distributed Networks. Only participants libraries of equal status can use the materials
- Star Networks : One participant library holds substantial materials. Resources to be utilized by other members of the network
- Hierarchical Network : Unsatisfied needs are forwarded to next bigger/higher resource centre. Higher resource centre should not be approached at first.
- Mixed Networks : Any combinations of above-mentioned networks

15.7 Resource Sharing Agreements

There are several basic agreements among libraries that must be developed if resource-sharing system is to be functional. It should be based on the premise that resource sharing among member libraries is in the public interest and should be

encouraged; interlibrary borrowing is an integral part of collection development, not an ancillary option; interlibrary lending is vital to the success of every library's ability to borrow needed materials. This agreement should include :

- Definition
- Purpose : Identify the objectives including service types and standards
- Scope : Explains the types of materials and areas of activity should covered by the resource sharing arrangement.
- Responsibilities of the Requesting library
- Responsibilities of the Supplying Library
- Retention Policy : Weedout policy of the each participating library must be satisfy needs and scope of the network.
- Classification of Users : Each member library has an obligation to serve its primary customers as a first priority. Primary customers are defined as those individuals affiliated with a member library as a student, faculty, employee and/or resident of a geographic community etc. The library mandated to serve each group of primary customers would be referred to as the "home library". A client using a library other than his/her home library shall be designated a secondary user.
- Protocols : Protocols for access to member collections will be developed to encourage balance in the demands on particular institutions and to avoid overburdening any one institution. Member institutions will encourage their primary customers to fully use the resources of their home library prior to accessing other member libraries.
- Expenses : Some charges for resource sharing activities may be applied. However, resource sharing among members shall not be considered a revenue generating activity.
- Confidentiality : User confidentiality shall be protected and the exchange of information about customers limited to that which is required to provide service.
- Violation of the Agreement : The financial agreement should permit individual libraries to withdraw, but sufficiently constraining to avoid disturbances of the system.

However, there should be agreement on acquisition policies, both to ensure consistent development of holdings and to avoid redundancy. There should also be agreement on bibliographic control. Best is standardization, so that users of each

participating library have a common means of accessing the catalogue/bibliographic control tools of others. Funding should be based on an obligation for long-term support to permit the benefits to develop. The financial agreement should permit individual libraries to withdraw, but sufficiently constraining to avoid disturbances of the system. This will also ensure commitment of parent institution.

15.8 Infrastructure of Resource Sharing

Effective resource sharing presupposes an infrastructure that permits users to discover materials in both print and electronic formats. In past, most of surrogates to information resources were manually developed. Naturally, the availability of such tools for document discovery were limited and localized. Hence, resource sharing was limited to selected resourceful libraries. Many libraries were unable to participate and/or take benefits of networks.

Uses of computers, increasing conversion of library catalogues, web access to document discovery tools, low cost communication facilities (email/Internet), and electronic document delivery system has revolutionized resource sharing worldwide. This ability to link resources at the user's desktop using facilities of full-text database will make possible a very different way for the individual researcher or practitioner to access information. There are various tools available, which help in document discovery and/or document delivery. Two major approaches are union catalogue and Z39.50 based distributed search systems. These two approaches should be considered complementary rather than competitive. Technologies to create linkage between the bibliographic apparatus of catalogues and abstracting & indexing databases and primary content in electronic form, such as the new Serial item and Contribution Identifier (SICI) standard are also key elements in the infrastructure to support resource sharing.

Open URL is a type of URL that contains resource metadata for use primarily in libraries. National Information Standards Organization (NISO), has developed OpenURL and its data container (the ContextObject) as international ANSI standard Z39.88.

The OpenURL standard is designed to support mediated linking from information resources (sources) to library services (targets). A "link resolver", or "link-server", parses the elements of an OpenURL and provides links to appropriate services as identified by a library. A source is generally a bibliographic citation or bibliographic record used to generate an OpenURL. A target is a resource or service that helps

satisfy user's information needs. Examples include full-text repositories; abstracting, indexing, and citation databases; online library catalogues; and other Web resources and services. Pubmed and MathSciNet and others provides similar facilities, which not only facilitates resource discovery but also provides new means for resource delivery.

A&I databases and other secondary information resources are now common services offered to library users alongside access to catalogues. They are available from a wide range of sources scattered across the network. Increasingly, technologies like Z39.50 are enabling consistent user interfaces to wide ranges of A&I databases accessible through the network. Many of these A&I have coverage that overlaps with other competing/complementary A&I databases in complex ways. While most library users today search databases sequentially, there is growing need for interfaces that will consolidated records retrieved from multiple A&I databases into a logical union A&I databases. The characteristics of such a consolidation process are highly dynamic and are likely to be based on distributed search approaches than on traditional union catalogue style consolidation. It is interesting to note that some of the commercial search services, such as DIALOG have offered such capabilities for consolidation of records from multiple A&I databases. While key standards to support linkages from A&I databases to primary content are now coming into place, actual implementation of such linkages is relatively new.

E-content suppliers have added value to resource delivery mechanism. OCLC offers a number of outbound links : OpenURL, netLibrary, infotrieve, JSTOR, online booksellers. It's important to show users all the options, e.g. it's available online, it's available at home library, it's owned by home library but it's not available. Reduce the number of options the user has to process.

15.8 Resource Description in the Digital Age

Resource description, known more familiarly within the library community as cataloguing or indexing, is undergoing intense scrutiny with the rapid proliferation of and access to, digital resources. There are many initiatives addressing a range of issues : the need for and definition of, a basic set of metadata elements; the examination of library cataloguing objectives and records structures; persistent addresses for resources; and the proposal for a data registry to facilitate interoperability among metadata schemes. The importance of a framework for resource discovery created through formal resource description is reiterated.

15.9 Resource sharing in a virtual library environment

Resource sharing in the sense of sharing printed documents is largely based on scarcity of financial resources, which resulted in reductions in the range and depth of information resources individual libraries can make available. Traditional co-operative projects do not offer a real solution to the problem of deteriorating collections. Information technology has a profound effect on resource sharing activities. A collection is no longer bound by the structure of four walls. The library's primary task has always been to select stabilize, protect, and provide access to relevant and representative information resources. The *collection* function, however, is expanding to include a *connection* function. The virtual library concept can be seen as a model for resource sharing. The three major components of resource sharing are : bibliographic access, interlibrary lending, and co-operative collection development. A new facet of resource sharing is the development of joint licensing agreements that permit consortia of libraries to share responsibilities and costs of providing access to electronic resources.

In the electronic environment most electronic information available commercially relies on licensing for access by libraries. The combined buying power of a consortium has a better chance than do individual libraries of persuading database providers to alter unacceptable terms in addition to lowering their prices.

Traditional co-operative collection development seeks to rationalize and distribute responsibility for acquiring little-used specialized publications. Shared approaches to licensing tend to focus on high-use high-demand databases which all or most members of a consortium wish to make available. The ability to provide immediate access from anywhere makes it far more shareable than the peripheral material that was the traditional object of co-operative collection development.

15.10 Copyright in Networked Electronic Publishing

The publication of scholarly works in a networked electronic environment presents many opportunities for solving some of the problems that currently exist in the print world. At the same time, copyright law, a form of legal protection developed primarily for printed works, has been used to create stumbling blocks both for faculty authors and their institutions. This has occurred because publishers have required a transfer of copyright to the publisher as a quid proquo for getting the work published. New models of copyright ownership and management can be developed for electronic

publishing of scholarly works and research results that will provide greater control to the faculty author, ease the distribution and permissions process for the use of copyrighted works in teaching and research, and ultimately will reduce costs to universities which currently must repurchase faculty-produced works from commercial publishers.

15.11 Open Access Initiative

The Open Access movement seeks to make scholarly, peer-reviewed journal articles freely available to anyone, anywhere over the World Wide Web. There were some very significant developments in the area of Open Access (OA) in 2004, including statements by major fund providers in support of Open Access. There are now so many Open Access scholarly journal articles freely available that it is now essential for libraries to be aware of and use the resources and related tools. Libraries can provide more resources faster for users by supplementing paid resources with ones that are Open Access.

Library resources, such as link resolvers, are beginning to incorporate Open Access materials and web searches for Open Access materials. For example, the researcher software suite includes Open Access collections along with subscription-based resources in the CUFTS journals knowledgebase, and a web search for an Open Access copy of an article in the GODOT link resolver. SFX also incorporates Open Access journals. After exhausting more traditional resources, interlibrary loans staff are beginning to include Google searching in their workflow.

References and Further Readings

- 1 2004 Jebaraj (Franklin David) and Devadoss (Fredrick Robert). Library and information networks in India (Library Philosophy and Practice 6(2), 2004). (<http://www.webpages.uidaho.edu/~mbolin/lppv6n2.htm>)
- 2 1999 Bakker (Trix). Resource sharing in a virtual library environment : user oriented collection management. (<http://www2.fmg.uva.nl/sociosite/bakker/resovirt.html#intro>). 1999.
- 3 1997 Glitz (Beryl). Electronic Publishing and Resource Sharing : How will our Document Delivery Models Change? (<http://nlnm.gov/psr/lat/v6n2/index.html>). 1997.
- 4 1997 Resource sharing in changing environment-Library Trends 45(3), Winter 1997. p 367-573.

- 5 1993 Godden (Irene P), ed. *Advances in librarianship*. V17. San Diego : Academic Press, 1993.
- 6 1969 *Encyclopedia of library and information science*. Ed by Allen Kent and Harold Lancour. NY : Marcel Dekker. 1969-. Various volumes.

15.12 Exercise

1. What is resource sharing?
2. Examine various areas of activities that can benefit from resource sharing.
3. Discuss barriers and impetus of resource sharing.
4. Describe resource sharing in virtual library environment.

Unit 16 □ Networks and their Classification

Structure

16.0 Objectives

16.1 Introduction

16.2 Classification of Networks

16.3 Telecommunication Networks

16.4 Computer Networks

16.4.1 Development of LAN and WAN

16.4.2 Advantages

16.4.3 Disadvantages

16.4.4 Networking Hardware

16.4.4.1 File Server

16.4.4.2 Workstation

16.4.4.3 Network Interface Card

16.4.4.4 Hub

16.4.4.5 Switch

16.4.4.6 Repeater

16.4.4.7 Bridge

16.4.4.8 Router

16.4.4.9 Gateway

16.4.5 Network Cabling

16.5 Exercise

16.0 Objectives

The objectives of the Unit are to :

- Classify networks by various characteristics within the knowledge domain of library and information services.
- Understand role of telecommunication
- Describe components of a computer network
- Enumerate the concept of bibliographic information network

16.1 Introduction

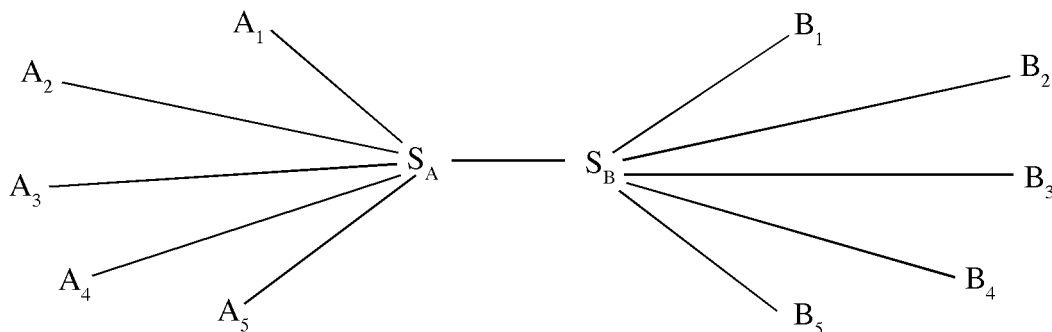
Network plays a crucial role in modern life. The modern economy would be very much diminished without the transportation, communication, information, and railroad networks. Networks have been developed to link banks, government offices, educational institutes and libraries etc. The objective of a communication system is to allow the exchange of information. The information may reside within the same system and/or may be physically located in different environments. Any network consists of three components :

1. Transmission Media
2. Control Mechanism
3. Interface

16.2 Classification of Networks

Networks are composed of links that connect nodes. It is inherent in the structure of a network that many components of a network are required for the provision of a typical service.

The classification in network type (one-way or two-way) is not a function of the topological structure of the network. Rather, it depends on the interpretation of the structure to represent a specific service. For example, the network of the following figure can be interpreted as a two-way telephone network where SA represents a local switch in city A , A_i represents a customer in city A , and similarly for SB and B_j . In this network, there are two types of local phone calls $A_iS_A A_k$ and $B_jS_B B_l$, as well as long distance phone call $A_iS_A S_B B_j$. We can also interpret the network of the following figure as an Automatic Teller Machine network. Then a transaction (say a withdrawal) from bank B_j from ATM A_i is $A_i S_A S_B B_j$. Connections $A_iS_A A_k$ and $B_jS_B B_l$ may be feasible but there is no demand for them.



Traditionally, networks were analyzed under the assumption that a single firm owned each network. Once, one of the most important networks, the AT&T telecommunications network in the US, was broken to pieces, economic research focused in the 80s and '90s on issues of interconnection and compatibility. Significant reductions in costs also contributed and will contribute to the transformation toward fragmented ownership in the telecommunications sector in both the United States and abroad. Costs of transmission have fallen dramatically with the introduction of fiber optic lines. Switching costs have followed the fast cost decreases of microchips and integrated circuits. These cost reductions have transformed the telecommunications industry from a natural monopoly to an oligopoly. The same cost reductions have made many new services, such as interactive video and interactive games, feasible at low cost. Technological change now allows for joint transmission of digital signals of various communications services. Thus, the monopoly of the last link closest to home is in the process of being eliminated,⁸ since both telephone lines and cable lines (and in some cases PCS and terrestrial satellites) will provide similar services.

16.3 Telecommunication

Communication is the electronic transfer of information from one location to another. “Data communications” or “datacom” refers to digital transmission and “telecommunications” or “telecom” refers to a mix of voice and data, both analog and digital. “Networking” refers specifically to LANs and WANs. The term “communications” generally pertains to telecom-related subjects such as PBXs, modems, call centers and the like.

The world’s largest communication system is the telephone network, which is a mix of analog and digital communications. The system, which used to be entirely analog and transmitted only voice frequencies, is now almost entirely digital. The only analog part is the line between telephone node and a digital conversion point (digital loop carrier) within. Analog systems are error prone, because the electronic frequencies get mixed together with unwanted signals (noise) that are nearby.

In analog telephone networks, amplifiers were placed in the line every few miles to boost the signal, but they could not distinguish between signal and noise. Thus, the noise was amplified along with the signal. By the time the receiving person or machine got the signal, it may have been impossible to decipher.

In a “digital” network, only two (binary) distinct frequencies or voltages are transmitted. Instead of amplifiers, repeaters are used, which analyze the incoming signal and regenerate a new outgoing signal. Any noise on the line is filtered out at

the next repeater. When data are made up of only two signals (0 and 1), they can be more easily distinguished from the garble.

Telecommunications may be defined as devices and systems that transmit electronic signals across long distances. Telecommunication covers all forms of distance and/or conversion of the original communications, including radio, telegraphy, television, telephony, data communication and computer networking. One of the roles of the telecommunications engineer is to analyse the physical properties of the line or transmission medium, and the statistical properties of the message in order to design the most effective encoding and decoding mechanisms.

The elements of a telecommunication system are a :

- Transmitter : The transmitter is a device that transforms or encodes the *message* into a physical phenomenon; the *signal*.
- Medium (line) : The transmission medium, by its physical nature, is likely to modify or degrade the signal on its path from the transmitter to the receiver.
- Channel imposed upon the medium : A path for electrical transmission between two or more points without common carrier-provided terminal equipment; also called a link, line, circuit or facility. All channels have *noise*. Another important aspect of the channel is called the bandwidth. A low bandwidth channel, such as a telephone, cannot carry all of the audio information that is transmitted in normal conversation, causing distortion and irregularities in the speaker's voice, as compared to normal, in-person speech.
- Receiver : The receiver has a decoding mechanism capable of recovering the message within certain limits of signal degradation. Sometimes, the final "receiver" is the human eye and/or ear and the recovery of the message is done by the brain.

Telecommunications messages can be sent in a variety of ways and by a wide range of devices. It may use different channel and/or signals. The telecommunication networks may be classified according to different characteristics.

16.3.1 Classification of Telecommunication networks by Channels

A path for electrical transmission between two or more points without common carrier-provided terminal equipment; also called a link, line, circuit or facility. Types of channels are :

- A **simplex Communication** system is one where all signals flow in one direction. These systems are often employed in broadcast networks, where the receivers do not need to send any data back to the transmitter/broadcaster. Example : Television, Commercial Radio Broadcast.

- A **duplex Communication** system is one where signals can flow in both directions between connected parties. These systems are employed in nearly all communications networks, either to allow for a “two-way street” between connected parties or to provide a “reverse path” for the monitoring and remote adjustment of equipment in the field.
 - Half-duplex : A *half-duplex* system allows communications in both directions, but only one direction at a time (not simultaneously). Any radio system where you must use “Over” to indicate the end of transmission, or any other procedure to ensure that only one party broadcasts at a time would be a *half-duplex* system. A good analogy for a *half-duplex* system would be a one lane road with traffic controllers at each end. Traffic can flow in both directions, but only one direction at a time with this being regulated by the controllers.
 - Full-duplex : A *full-duplex* system allows communication in both directions, and unlike half-duplex allows this to happen simultaneously. Most telephone networks are *full duplex* as they allow both callers to speak at the same time. A good analogy for a *full-duplex* system would be a two lane road with one lane for each direction. Example : Telephone.

16.3.2 Classification of Telecommunication Networks by Communication

Telecommunication can be point-to point, point-to-multipoint or broadcasting, which is a particular form of point-to-multipoint that goes only from the transmitter to the receivers.

- ◆ Point-to-point : The messages can be sent from one sender to a single receiver (point-to-point) or from one sender to many receivers (Point-to-multipoint). Personal communications, such as a telephone conversation between two people or a facsimile (fax) usually involve point-to-point transmission.
- ◆ Point-to-multipoint : It is most typically used in wireless Internet and IP Telephony via gigahertz radio frequencies. PT2MP systems have been designed both as single and bi-directional systems. A central antenna or antenna array broadcasts to several receiving antennae and the system uses a form of Time-domain Multiplexing to allow for the back-channel traffic.
- ◆ Broadcasting : It is the distribution of audio and video signals (programmes) to a number of recipients (“listeners” or “viewers”) that belong to a large group. Television and radio programmes are distributed through radio broadcasting or cable, often both simultaneously. By coding signals and having decoding equipment in homes, the latter also enables subscription-based channels and pay-per-view services.

16.3.3 Classification Telecommunication networks by Signal Characteristics

Based on the signal characteristics, telecommunication networks may be classified as follows :

1. Telegraph Networks
2. Telephone Networks
3. Fax/Telex Networks
4. Computer Networks

16.3.3.1 Telegraph Networks

Telegraph services use both wire line and wireless media for transmissions. Soon after the introduction of the telegraph in 1844, telegraph wires spanned the country. A message sent by telegraph was called a telegram. Telegrams were printed on paper and delivered to the receiving party by the telegraph company. With the invention of the radio in the early 1900s, telegraph signals could also be sent by radio waves. Wireless telegraphy made it practical for ocean going ships as well as aircraft to stay in constant contact with land-based stations.

16.3.3.2 Telephone Networks

The telephone network also uses both wire line and wireless methods to deliver voice communications between people, and data communications between computers and people or other computers. The part of the telephone network that currently serves individuals residences and many businesses operates in an analog mode and relays electronic signals that are continuous, like the human voice. Digital transmission is now used in some sections of the telephone network that send large amounts of calls over long distances. However, since the rest of the telephone system is still analog, these digital signals must be converted back to analog before they reach users.

16.3.3.3 Telex/Facsimile Transmission Networks

Teletype, telex, and facsimile transmission are all methods for transmitting text rather than sounds. Facsimile transmission now provides a cheaper and easier way to transmit text and graphics over distances. Fax machines contain an optical scanner that converts text and graphics into digital, or machine-readable, codes. This coded information is sent over ordinary analog telephone lines through the use of a modem included in the fax machine. The receiving fax machine's modem demodulates the signal and sends it to a printer also contained in the fax machine.

16.3.3.4 Data Networks

Data networks with the ability to send and receive audio, video, text, software, and multimedia, is one of the fastest-growing segments of the telecommunications.

Computer-telecommunications symbiosis takes advantage of existing telephone connections to transmit digital data. This type of transmission is frequently done over the Internet. Some computers connect directly to the digital portion of the telephone network using the Integrated Services Digital Network (ISDN), but this requires the installation of special devices and telephone line conditioning. An improved modem system for regular phone lines, called Digital Subscriber Line (DSL), is being developed to increase modem speed tremendously. The difference between voice and data communication :

Voice Communication	Data Communication
Continuous	Discrete
Low bandwidth for long duration	High bandwidth for short duration
Half duplex	Half/Full duplex
Real time	Near real time
Loss acceptable	No distortion acceptable
Error tolerable	Error unacceptable
Sharing not possible	Sharing possible

16.4 Communication Networks based on Switching Techniques

Most communication systems have started with point-to-point links, which directly connect together the users wishing to communicate, using a dedicated communications circuit. As the distance between users increases beyond the manageable length of the cable, the connection between the users was formed by a number of sections.

As the number of connected users increased, it has become infeasible to provide a circuit, which connects every user to every other user, and some sharing of the transmission circuits (know as “switching”) has become necessary. To accomplish this goal, the data communications network has evolved. A network is a set of nodes that are interconnected to permit the exchange of information. Three switching techniques have been proposed for building networks :

- Circuit switching
- Message switching
- Packet switching

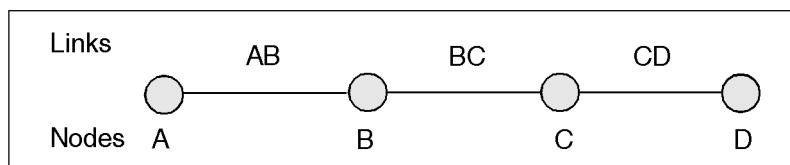
Each allows sharing communication facilities among multiple users, and each uses equipment located at the nodes to replace the patch-panels used in a point-to-point connection. Packet switching is most often used for data communication. Most

networks consists of many links (see the figure below), which allow more than one path through the network between nodes. A data communications network must be able to select an appropriate path for each required connection.

The relative performance of circuit switching and packet switching depends strongly on the speed and “cost” of establishing a connection.

16.4.1 Circuit switching

It allows communication equipment and circuits, to be shared among users. Each user has sole access to a circuit (functionally equivalent to a pair of copper wires) during network use. Consider communication between two points A and D in a network. The connection between A and D is provided using (shared) links between two other pieces of equipment, B and C.



A connection between two systems A & D formed from 3 links

Network use is initiated by a connection phase, during which a circuit is set up between source and destination, and terminated by a disconnect phase. After a user requests a circuit, the desired destination address must be communicated to the local switching node (B). In a telephony network, this is achieved by dialing the number.

Node B receives the connection request and identifies a path to the destination (D) via an intermediate node (C). This is followed by a circuit connection phase handled by the switching nodes and initiated by allocating a free circuit to C (link BC), followed by transmission of a call request signal from node B to node C. In turn, node C allocates a link (CD) and the request is then passed to node D after a similar delay.

The circuit is then established and may be used. While it is available for use, resources (i.e. in the intermediate equipment at B and C) and capacity on the links between the equipment are dedicated to the use of the circuit.

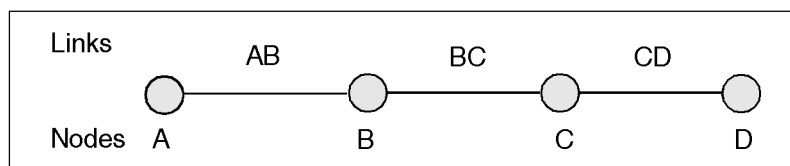
After completion of the connection, a signal confirming circuit establishment (a connect signal in the diagram) is returned; this flows directly back to node A with no search delays since the circuit has been established. Transfer of the data in the message then begins. After data transfer, the circuit is disconnected; a simple disconnect phase is included after the end of the data transmission.

Delays for setting up a circuit connection can be high, especially if ordinary telephone equipment is used. Call setup time with conventional equipment is typically

on the order of 5 to 25 seconds after completion of dialing. New fast circuit switching techniques can reduce delays. Trade-offs between circuit switching and other types of switching depend strongly on switching times.

16.4.2 Message Switching

Sometimes there is no need for a circuit to be established all the way from the source to the destination. Consider a connection between the users (A and D) in the figure below (i.e. A and D) is represented by a series of links (AB, BC, and CD).



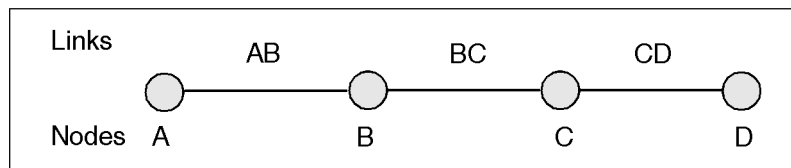
A connection between two systems A & D formed from 3 links

For instance, when a telex (or email) message is sent from A to D, it first passes over a local connection (AB). It is then passed at some later time to C (via link BC), and from there to the destination (via link CD). At each message switch, the received message is stored, and a connection is subsequently made to deliver the message to the neighboring message switch. Message switching is also known as store-and-forward switching since the messages are stored at intermediate nodes en route to their destinations. Most message switched networks do not use dedicated point-to-point links and therefore a call must be set-up using a circuit switched network.

16.4.3 Packet Switching

Packet switching is similar to message switching using short messages. Any message exceeding a network-defined maximum length is broken up into shorter units, known as packets, for transmission; the packets, each with an associated header, are then transmitted individually through the network. The fundamental difference in packet communication is that the data is formed into packets with a pre-defined header format, and well known "idle" patterns, which are used to occupy the link when there is no data to be communicated.

Packet network equipment discards the "idle" patterns between packets and processes the entire packet as one piece of data. The equipment examines the packet header information and then either removes the header (in an end system) or forwards the packet to another system. If the out-going link is not available, then the packet is placed in a queue until the link becomes free. A packet network is formed by links, which connect packet network equipment.



Communication between A and D using circuits, which are shared using packet switching.

There are two important benefits from packet switching.

- The first and most important benefit is that since packets are short, the communication links between the nodes are only allocated to transferring a single message for a short period of time while transmitting each packet. Longer messages require a series of packets to be sent, but do not require the link to be dedicated between the transmission of each packet. The implication is that packets belonging to other messages may be sent between the packets of the message being sent from A to D. This provides a much fairer sharing of the resources of each of the links.
- Another benefit of packet switching is known as “pipelining”. Pipelining is visible in the figure above. At the time packet 1 is sent from B to C, packet 2 is sent from A to B; packet 1 is sent from C to D while packet 2 is sent from B to C, and packet 3 is sent from A to B, and so forth. This simultaneous use of communications links represents a gain in efficiency, the total delay for transmission across a packet network may be considerably less than for message switching, despite the inclusion of a header in each packet rather than in each message.

16.4 Computer Network

Networks are connections between groups of computers and associated devices that allow users to transfer information electronically. It implies techniques, physical connections, and computer programmes used to link two or more computers. Network users are able to share files, printers, and other resources; send electronic messages; and run programmes on other computers. The computers on a network may be linked through cables, telephone lines, radio waves, satellites, or infrared light beams. The basic reasons for the growth of computer networks are :

- Resource sharing
- Data Sharing
- Communication and data exchange.

In effect, computer network may be considered as a communication channel.

16.4.1 Development of LAN & WAN

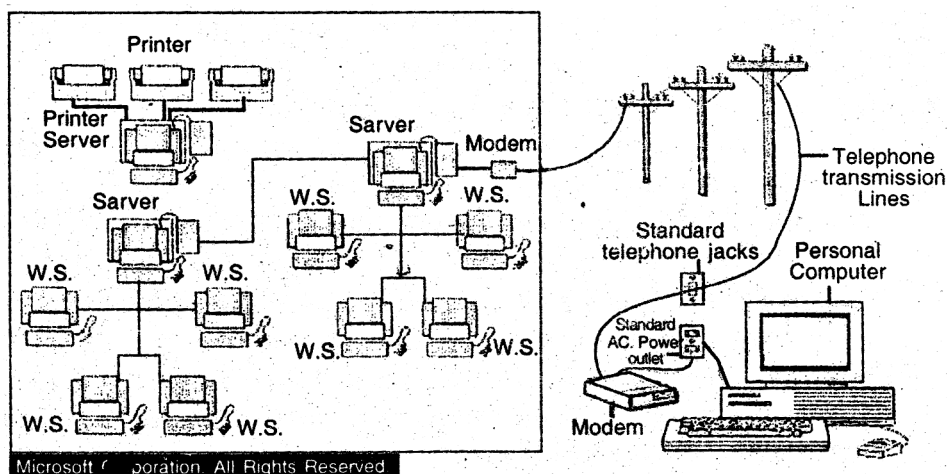
For historical reasons, the industry refers to nearly every type of computer network as an “area network.” The most commonly discussed categories of computer networks include the following :

- Local Area Network (LAN)
- Wide Area Network (WAN)
- Metropolitan Area Network (MAN)
- Storage Area Network (SAN)
- System Area Network (SAN)
- Server Area Network (SAN)
- Small Area Network (SAN)
- Personal Area Network (PAN)
- Desk Area Network (DAN)
- Controller Area Network (CAN)
- Cluster Area Network (CAN)

LANs and WANs were the original flavours of network design. The concept of “area” made good sense at this time, because a key distinction between a LAN and a WAN involves the physical distance that the network spans. A third category of the Computer Network, the MAN, also fit into this scheme as it too is centered on a distance-based concept.

As technology improved, new types of networks appeared on the scene. These, too, became known as various types of “area networks” for consistency’s sake, although distance no longer proved a useful differentiator.

The following figure represents a local area network. Individual computers are called workstations (W.S.), and communicate to each other via cable or telephone line linking to servers. The red line represents the larger network connection between servers, called the backbone; the blue line shows local connections. A modem (modulator/demodulator) allows computers to transfer information across standard telephone lines. Modems convert digital signals into analog signals and back again, making it possible for computers to communicate, or network, across thousands of miles.



16.4.2 Advantages

- **Speed.** Networks provide a very rapid method for sharing and transferring files. Without a network, copying them to floppy disks, then carrying or sending the disks from one computer to other to share files. This method of transferring files is very time-consuming.
- **Cost.** Network versions of many popular software programmes are available at considerable savings when compared to buying individually licensed copies. Besides monetary savings, sharing a programme on a network allows for easier upgrading of the programme. The changes have to be done only once, on the file server, instead of on all the individual workstations.
- **Security.** Files and programmes on a network can be designated as “copy inhibit,” to prevent illegal copying of programmes. Also, passwords can be established for specific directories to restrict access to authorized users.
- **Centralized Software Management.** One of the greatest benefits of installing a computer network is the fact that all of the software can be loaded on one computer (the file server). This eliminates the need to spend time and energy installing updates and tracking files on independent computers throughout the building.
- **Resource Sharing.** Sharing resources is another area in which a network exceeds stand-alone computers. Most organizations cannot afford enough laser printers; fax machines, modems, scanners, and CD-ROM players for each computer. However, if these or similar peripherals are added to a network, they can be shared by many users.
- **Electronic Mail.** The presence of a network provides the hardware necessary to install an e-mail system. E-mail aids in personal and professional communication for all personnel, and it facilitates the dissemination of general information to the entire staff. Electronic mail on a LAN can enable workers to communicate with peers. If the LAN is connected to the Internet, workers can communicate with others throughout the world.
- **Flexible Access.** Networks allow workers to access their files from computers throughout the office. Students can begin an assignment in their classroom, save part of it on a public access area of the network, and then go to the media center after school to finish their work. Students can also work cooperatively through the network.
- **Workgroup Computing.** Workgroup software (such as Microsoft Back Office) allows many users to work on a document or project concurrently. For example, educators located at various schools within a country could simultaneously contribute their ideas about new curriculum standards to the same document and spreadsheets.

16.4.3 Disadvantages

- **Expensive to Install.** Although a network will generally save money over time, the initial costs of installation can be prohibitive. Cables, network cards, and software are expensive, and the installation may require the services of a technician.
- **Requires Administrative Time.** Proper maintenance of a network requires considerable time and expertise. Many organizations have installed a network, only to find that they did not have budget for the necessary administrative support.
- **File Server May Fail.** Although a file server is no more susceptible to failure than any other computer, when the files server “goes down”, the entire network may come to a halt. When this happens, the entire organization may lose access to necessary programmes and files.
- **Cables May Break.** Some of the configurations are designed to minimize the inconvenience of a broken cable; with other configurations, one broken cable can stop the entire network.

16.4.4 Networking Hardware

Networking hardware includes all computers, peripherals, interface cards and other equipment needed to perform data processing and communications within the network. The major components of a computer network are :

- File Servers
- Workstations
- Network Interface Cards
- Switches
- Repeaters
- Bridges
- Routers

16.4.4.1 File Servers

A file server stands at the heart of most networks. It is a very fast computer with a large amount of RAM and storage space, along with a fast network interface card. The network operating system software resides on this computer, along with any software applications and data files that need to be shared. The file server controls the communication of information between the nodes on a network.

16.4.4.2 Workstations

All of the user computers connected to a network are called workstations. A typical workstation is a computer that is configured with a network interface card, networking software, and the appropriate cables. Workstations do not necessarily need floppy disk drives because files can be saved on the file server. Almost any computer can serve as a network workstation.

16.4.4.3 Network Interface Cards

The network interface card (NIC) provides the physical connection between the network and the computer workstation. Most NICs are internal, with the card fitting into an expansion slot inside the computer. Some computers, such as Mac Classics, use external boxes that are attached to a serial port or a SCSI port. Network interface cards are a major factor in determining the speed and performance of a network. It is a good idea to use the fastest network card available for the type of workstation you are using. The three most common network interface connections are Ethernet cards, LocalTalk connectors, and Token Ring cards.

16.4.4.4 Hub

It is a small rectangular box, often made of plastic that receives its power from an ordinary wall outlet. This network device joins multiple computers or other computer devices together to form a single network segment. On this network segment, all computers can communicate directly with each other. Ethernet hubs are by far the most common type.

16.4.4.5 Switch

A **network switch** is a small device that joins multiple computers together at a low-level network protocol layer. Technically, network switches operate at layer two (Data Link Layer) of the OSI model. Most switches are active that is they electrically amplify the signal as it moves from one device to another. Switches no longer broadcast network packets as hubs did in the past; they memorize addressing of computers and send the information to the correct location directly.

16.4.4.6 Repeaters

Since a signal loses strength as it passes along a cable, it is often necessary to boost the signal with a device called a repeater. The repeater electrically amplifies the signal it receives and rebroadcasts it. Repeaters can be separate devices or they can be incorporated into a concentrator. They are used when the total length of your network cable exceeds the standards set for the type of cable being used. A repeater connects two segments of the network cable.

16.4.4.7 Bridges

A bridge is a device that allows segmenting a large network into two smaller, more efficient networks. A bridge monitors the information traffic on both sides of the network so that it can pass packets of information to the correct location. Most bridges can “listen” to the network and automatically figure out the address of each computer on both sides of the bridge. The bridge can inspect each message and if necessary, broadcast it on the other side of the network.

A bridge reads the outermost section of data on the data packet, to tell where the message is going. It reduces the traffic on other network segments, since it does not send all packets. Bridges can be programmed to reject packets from particular networks. Bridging occurs at the data link layer of the OSI model, which means the bridge cannot read IP addresses, but only the outermost hardware address of the packet. The bridge can read the Ethernet data which gives hardware address of the destination address, not the IP address. Bridges forward all broadcast messages. Only a special bridge called a translation bridge will allow two networks of different architectures to be connected. Bridges do not normally allow connection of networks with different architectures.

16.4.4.8 Routers

A router translates information from one network to another; it is similar to a super intelligent bridge. Routers select the best path to route a message, based on the destination address and origin. The router can direct traffic to prevent head-on collisions, and is smart enough to know when to direct traffic along back roads and shortcuts.

While bridges know the addresses of all computers on each side of the network, routers know the addresses of computers, bridges and other routers on the network. Routers can even “listen” to the entire network to determine which sections are busiest—they can then redirect data around those sections until they clear up.

A router is necessary to connect LAN to the Internet. The router serves as the translator between the information on the LAN and the Internet. It also determines the best route to send the data over the Internet. Routers can route :

- Signal traffic efficiently
- Messages between any two protocols
- Messages between linear bus, star, and star-wired ring topologies
- Messages across fiber optic, coaxial and twisted-pair cabling

16.4.4.9 Gateway

A gateway can translate information between different network data formats or network architectures. It can translate TCP/IP to AppleTalk so computers supporting TCP/IP can communicate with apple brand computers. Most gateways operate at the application layer, but can operate at the network or session layer of the OSI model. Gateways will start at the lower level and strip information until it gets to the required level and repackage the information and work its way back toward the hardware layer of the OSI model. To confuse issues, when talking about a router that is used to interface to another network, the word gateway is often used. This does not mean the routing machine is a gateway as defined here, although it could be.

16.4.5 Network Cabling

Cable is the medium through which information usually moves from one network device to another. There are several types of cable, which are commonly used with LANs. In some cases, a network will utilize only one type of cable; other networks will use a variety of cable types. The type of cable chosen for a network is related to the network's topology, protocol, and size. Understanding the characteristics of different types of cable and how they relate to other aspects of a network is necessary for the development of a successful network.

The following sections discuss the types of cables used in networks and other related topics.

- Unshielded Twisted Pair (UTP) Cable
- Shielded Twisted Pair (STP) Cable
- Coaxial Cable
- Fiber Optic Cable
- Wireless LANs
- Cable Installation Guides

16.4.5.1 Unshielded Twisted Pair (UTP) Cable

Twisted pair cabling comes in two varieties : shielded and unshielded. Unshielded twisted pair (UTP) is the most popular and in generally the best option for small networks (See fig. 1).

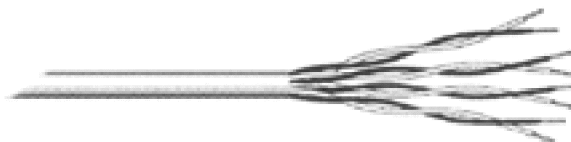


Fig. 1. : Unshielded twisted pair

The quality of UTP may vary from telephone-grade wire to extremely high-speed cable. The cable has four pairs of wires inside the jacket. Each pair is twisted with a different number of twists per inch to help eliminate interference from adjacent pairs and other electrical devices. For cable, the tighter the twisting, the higher is the supported transmission rate and the greater the cost per foot.

Categories of Unshielded Twisted Pair

Type	Use
Category 1	Voice Only (Telephone Wire)
Category 2	Data to 4 Mbps (LocalTalk)
Category 3	Data to 10 Mbps (Ethernet)
Category 4	Data to 20 Mbps (16 Mbps Token Ring)
Category 5	Data to 100 Mbps (Fast Ethernet)

16.4.5.2 Shielded Twisted Pair (STP) Cable

A disadvantage of UTP is that it may be susceptible to radio and electrical frequency interference. Shielded twisted pair (STP) is suitable for environments with electrical interference; however, the extra shielding can make the cables quite bulky. Shielded twisted pair is often used on networks using Token Ring topology.

16.4.5.3 Coaxial Cable

Coaxial cabling has a single copper conductor at its center. A plastic layer provides insulation between the center conductor and a braided metal shield (See fig. 3). The metal shield helps to block any outside interference from fluorescent lights, motors, and other computers.



Fig. 3. : Coaxial cable

Although coaxial cabling is difficult to install, it is highly resistant to signal interference. In addition, it can support greater cable lengths between network devices than twisted pair cable. The two types of coaxial cabling are thick coaxial and thin coaxial.

Thin coaxial is also referred to as thinnet. 10Base2 refers to the specifications for thin coaxial cable carrying Ethernet signals. The 2 refers to the approximate maximum segment length being 200 meters. In actual fact the maximum segment length is 185 meters. Thin coaxial cable is popular in small networks, especially linear bus networks.

Thick coaxial cable is also referred to as thicknet. 10Base5 refers to the specifications for thick coaxial cable carrying Ethernet signals. The 5 refers to the maximum segment length being 500 meters. Thick coaxial cable has an extra protective plastic cover that helps keep moisture away from the center conductor. This makes thick coaxial a great choice when running longer lengths in a linear bus network. One disadvantage of thick coaxial is that it does not bend easily and is difficult to install.

16.4.5.4 Fiber Optic Cable

Fiber optic cabling consists of a center glass core surrounded by several layers of protective materials (See fig. 5). It transmits light rather than electronic signals eliminating the problem of electrical interference. It has also made it the standard for connecting networks between buildings, due to its immunity to the effects of moisture and lighting. Fiber optic cable has the ability to transmit signals over much longer distances than coaxial and twisted pair. It also has the capability to carry information at vastly greater speeds. This capacity broadens communication possibilities to include services such as video conferencing and interactive services. The cost of fiber optic cabling is comparable to copper cabling; however, it is more difficult to install and modify.

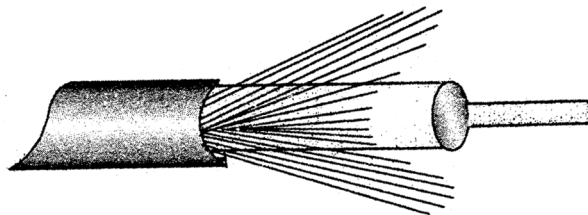


Fig. 5. : Fiber optic cable

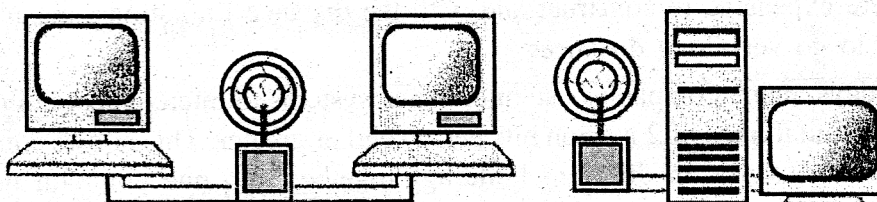
Facts about fiber optic cables :

- Outer insulating jacket is made of Teflon or PVC.
- Kevlar fiber helps to strengthen the cable and prevent breakage.
- A plastic coating is used to cushion the fiber center.
- Center (core) is made of glass or plastic fibers.

Ethernet cable summary

Specification	Cable Type	Maximum length
10 BaseT	Unshielded Twisted Pair	100 meters
10 Base2	Thin Coaxial	185 meters
10 Base5	Thick Coaxial	500 meters
10 BaseF	Fiber Optic	2000 meters
100 BaseTX	Unshielded Twisted Pair	220 meters

16.4.5.5 Wireless LANs



Not all networks are connected with cabling; some networks are wireless. Wireless LANs use high frequency radio signals, infrared light beams, or lasers to communicate between the workstations and the file server or hubs. Each workstation and file server on a wireless network has some sort of transceiver/antenna to send and receive the data. Information is relayed between transceivers as if they were physically connected. For longer distance, wireless communications can also take place through cellular telephone technology, microwave transmission, or by satellite.

Wireless networks are great for allowing laptop computers or remote computers to connect to the LAN. Wireless networks are also beneficial in older buildings where it may be difficult or impossible to install cables.

The two most common types of infrared communications used in small networks are line-of-sight and scattered broadcast. Line-of-sight communication means that there must be an unblocked direct line between the workstation and the transceiver. If a person walks within the line-of-sight while there is a transmission, the information would need to be sent again. This kind of obstruction can slow down the wireless network.

Scattered infrared communication is a broadcast of infrared transmissions sent out in multiple directions that bounces off walls and ceilings until it eventually hits the receiver. Networking communications with laser are virtually the same as line-of-sight infrared networks.

Wireless LANs have several disadvantages. They provide poor security, and are susceptible to interference from lights and electronic devices. They are also slower than LANs using cabling.

16.4.5.6 Micro-wave systems

Microwave system does not use cable as a transmission medium rather it uses the air. Using very high frequency signals, microwave support thousands of telephone channels and several television channels on the one circuit. Microwave is a radio system, which uses very high frequencies to send and receive data. Because of the high frequencies involved, stations are located about 30 kilometers apart and in line of sight (visible to each other). Microwave systems have sufficient bandwidth capacity to support a large number of voice channels and one or two TV channels. Dishes and towers were expensive to construct and with the distance limitations, meant it was expensive to go very long distances.

Nowadays, many companies use microwave systems to interconnect buildings at high-speed digital links of 2 million bits per second or greater. This allows companies to link their networks in different buildings together into one common network, allowing the sharing and accessing of information. Today, microwave systems are used in a number of areas, such as linking local area networks together between campus buildings, sending radio signal from a radio station to its transmitter site and the sending of video or audio signals from an outside sports event back to a TV broadcasting studio. Microwave systems have the advantage of medium capacity, medium cost, and can go long distances. Its disadvantages are noise interference, geographical problems due to line of sight requirements and are becoming outdated.

16.4.5.7 Satellite systems

Ground stations with large dishes communicate with a communication satellite in geo-stationary orbit around the earth. Each channel is managed by a transponder, which can support thousands of speech channels and about 4 TV channels simultaneously. The cost of satellite links is still very expensive (about \$4M per transponder). It is primarily used for intercontinental links.

Satellite systems are comprised of ground based transmitter and receiver dishes, with an orbital satellite circuit (called a transponder). Signals are transmitted to the orbiting satellite, which relays it back to another ground station. The footprint coverage of a single satellite system is very large, covering thousands of square kilometers. Satellite is used to carry television channels and telephone conversions between countries. Satellite systems have the advantage of low cost per user (for PAY TV), high capacity, and very large coverage. Its disadvantages are high install cost in

launching a satellite and receive dishes and decoders required, and delays involved in the reception of the signal.

References and Further Readings

- 1 2003 Local area network (http://www.webopedia.com/TERM/L/local_area_network_LAN.html).
- 2 2001 Fairhurst (Gorry). Datagram networks. (<http://www.erg.abdn.ac.uk/users/gorry/course/intro-pages/cs.html>). Visited last : 25/10/2005
- 3 1995 Hunter (Philip). Network operating systems : making the right choices. NY : Addison-Wesley, 1995
- 4 1994 McNamara (John E). Local area networks : an introduction to the technology. New Delhi : PIH, 1994.

16.5 Exercise

1. Identify the major components of networks.
2. How do you classify networks?
3. Discuss classification of telecommunication networks.
4. Discuss communication networks based on switching techniques.

Unit 17 □ Network Architecture and Services

Structure

17.0 Objectives

17.1 Introduction

17.2 Network Connections

17.3 Network Architecture

17.3.1 Topology

17.3.1.1 Linear Bus

17.3.1.2 Star

17.3.1.3 Star-Wired Ring

17.3.1.4 Tree

17.3.2 Protocol

17.3.2.1 Ethernet

17.3.2.2 LocalTalk

17.3.2.3 Token Ring

17.3.2.4 FDDI

17.3.2.5 ATM

17.3.2.6 TCP/IP

17.4 Open System Interconnection (Standard Layered Framework for Network Design)

17.5 Exercise

17.0 Objectives

The objectives of the Unit are to :

- Explain the concept of network architecture
- Study OSI models

17.1 Introduction

Networks are connections between groups of computers and associated devices that allow users to transfer information electronically. It implies techniques, physical

connections, and computer programmes used to link two or more computers. Network users are able to share files, printers and other resources; send electronic messages; and run programmes on other computers. A network has three layers of components : application software, network software and network hardware. The computers on a network may be linked through cables, telephone lines, radio waves, satellites, or infrared light beams. In effect, computer network may be considered as a communication channel. The basic reasons for the growth of computer networks are :

- Resource sharing
- Data sharing
- Communication and data exchange.

17.2 Network Connections

A network has two types of connections : Physical connections and Logical or Virtual Connections.

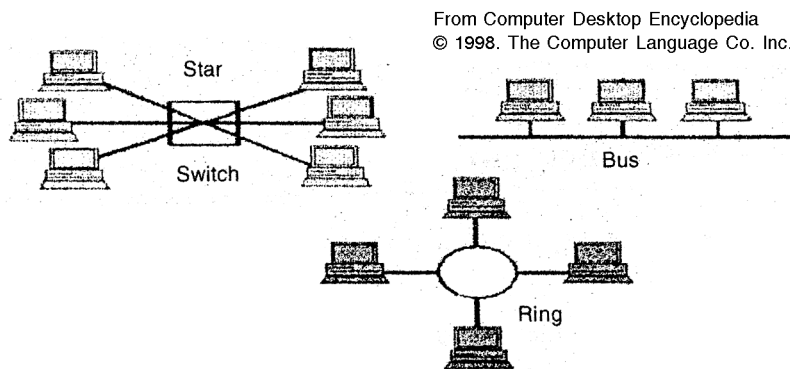
- Physical connections that let computers directly transmit and receive signals. Physical connections are defined by the medium used to carry the signal, the geometric arrangement of the computers (topology) and the method used to share information.
- Logical, or virtual, connections that allow computer applications, such as word processors, to exchange information. Logical connections are created by network protocols and allow data sharing between applications of different types of computers, such as an Apple Macintosh and an IBM personal computer (PC), in a network. Some logical connections use client-server application software and are primarily for file and printer sharing. The Transmission Control Protocol/Internet Protocol (TCP/IP) suite is the set of logical connections used by the Internet. TCP/IP, based on peer-to-peer application software creates a connection between any two computers.

17.3 Network Architecture (Topology)

Computers communicate with other computers via networks. The simplest network is a direct connection between two computers. However, computers can also be connected over large networks, allowing users to exchange data, communicate via electronic mail, and share resources such as printers. Computers can be connected in

several ways. The physical topology of a network refers to the configuration of cables, computers, and other peripherals. Physical topology should not be confused with logical topology (Protocol) which is the method used to pass information between workstations. Main types of physical topologies are :

- Linear Bus
- Star
- Star-Wired Ring
- Tree



17.3.1.1 Linear Bus

A linear bus topology consists of a main run of cable with a terminator at each end. All nodes (file server, workstations, and peripherals) are connected to the linear cable. Ethernet and Local Talk networks use a linear bus topology.

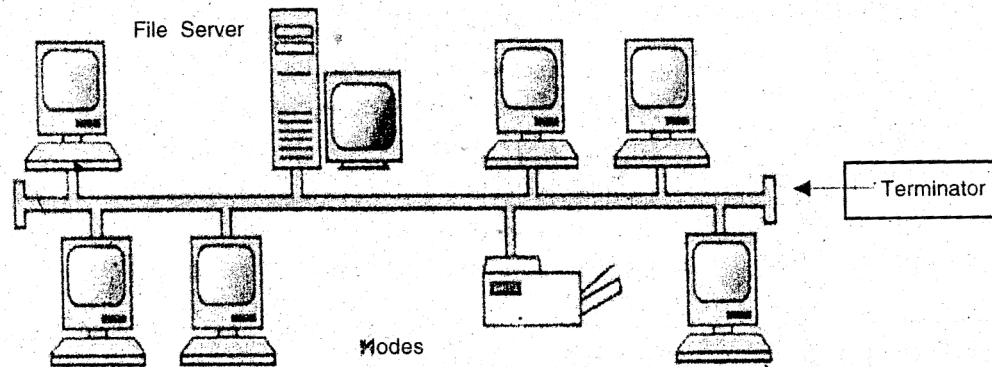


Figure : Linear Bus topology

In a bus configuration, computers are connected through a single set of wires, called a bus. A bus requires a full duplex medium (one which signals can flow in either direction). One computer sends data to another by broadcasting the address of the receiver and the data over the bus. All the computers in the

network look at the address simultaneously and the intended recipient accepts the data. A bus network, unlike a ring network, allows data to be sent directly from one computer to another. However, only one computer at a time can transmit data. The others must wait to send their messages. Messages are detected by all nodes but are accepted only by the node(s) to which they are addressed. Because a bus network relies on a common data “highway,” a malfunctioning node simply ceases to communicate; it doesn’t disrupt operation.

Advantages of a Linear Bus Topology

- Easy to connect a computer or peripheral to a linear bus.
- Requires less cable length than a star topology.
- Lack of routing and the lack of centralized control provide substantial reliability.

Disadvantages of a Linear Bus Topology

- Entire network shutdown if there is a break in the main cable.
- Terminators are required at both ends of the backbone cable.
- Difficult to identify the problem if the entire network shutdown.
- Not meant to be used as a stand-alone solution in a large building.
- Segmenting the network for maintenance is difficult.
- The impedance irregularities caused by the installation of taps cause signal reflections that can interfere data transmission if nodes/taps are placed too closed to each other.

17.3.1.2 Star

In a star configuration, computers are linked to a central computer called a hub. A computer sends the address of the receiver and the data to the hub, which then links the sending and receiving computers directly. A star network allows multiple messages to be sent simultaneously, but it is more costly because it uses an additional computer, the hub, to direct the data.

A star topology is designed with each node(file server, workstations and peripherals) connected directly to a central network hub or concentrator. Data on a star network passes through the hub or concentrator before continuing to its destination. The hub or concentrator manages and controls all functions of the network. It also acts as a repeater for the data flow. This configuration is common with twisted pair cable; however, it can also be used with coaxial cable or fiber optic cable.

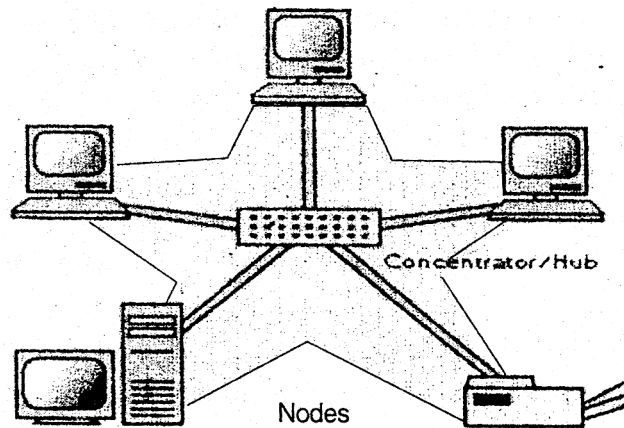


Figure : Star topology

A star network is reliable in the sense that a node can fail without affecting any other node on the network. Its weakness, however, is that failure of the central computer results in a shutdown of the entire network. And because each node is individually wired to the hub, cabling costs can be high.

Advantages of a Star Topology

- Easy to install and wire.
- No disruptions to the network when connecting or removing devices.
- Easy to detect faults and to remove parts.

Disadvantages of a Star Topology

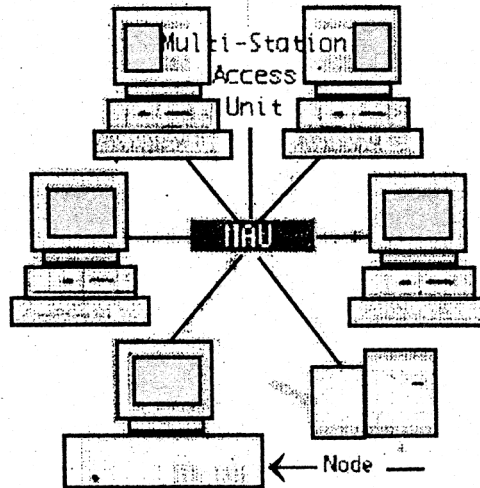
- Requires more cable length than a linear topology.
- If the hub or concentrator fails, nodes attached are disabled.
- More expensive than linear bus topologies because of the cost of the concentrators.

The protocols used with star configurations are usually Ethernet or LocalTalk. Token Ring uses a similar topology, called the star-wired ring.

17.3.1.3 Star-Wired Ring

In a ring configuration, data are transmitted along the ring and each computer in the ring examines this data to determine if it is the intended recipient. If the data are not intended for particular computer, the computer passes the data to the next computer in the ring. This process is repeated until the data arrive at their intended destination. A ring network allows multiple messages to be carried simultaneously, but since each computer checks each message; data transmission is slow. A star-wired ring topology may appear (externally) to be the same as a star topology. Internally, the MAU (multi

station access unit) of a star-wired ring contains wiring that allows information to pass from one device to another in a circle or ring. The Token Ring protocol uses a star-wired ring topology.



17.3.1.4 Tree

A tree topology combines characteristics of linear bus and star topologies. It consists of groups of star-configured workstations connected to a linear bus backbone cable. Tree topologies allow for the expansion of an existing network and enable organizations to configure a network to meet their needs.

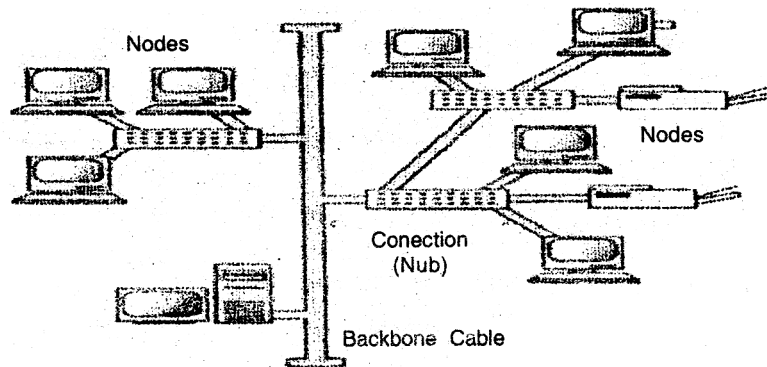


Figure : Tree topology

Advantages of a Tree Topology

- Point-to-point wiring for individual segments.
- Supported by several hardware and software vendors.

Disadvantages of a Tree Topology

- Overall the type of cabling used limits length of each segment.
- If the backbone line breaks, the entire segment goes down.
- More difficult to configure and wire than other topologies.

17.3.1.5 5-4-3 Rules

A consideration in setting up a tree topology using Ethernet protocol is the 5-4-3 rule. One aspect of the Ethernet protocol requires that a signal sent out on the network cable reach every part of the network within a specified length of time. Each concentrator or repeater that a signal goes through adds a small amount of time. This leads to the rule that between any two nodes on the network there can only be a maximum of 5 segments, connected through 4 repeaters/concentrators. In addition, only 3 of the segments may be populated (trunk) segments if they are made of coaxial cable. A populated segment is one, which has one, or more nodes attached to it. In Figure (Tree Topology), the 5-4-3 rules is adhered to. The furthest two nodes on the network have 4 segments and 3 repeaters/concentrators between them. This rule does not apply to other network protocols or Ethernet networks where all fiber optic cabling or a combination of a fiber backbone with UTP cabling is used. If there is a combination of fiber optic backbone and UTP cabling, the rule simply translated to 7-6-5 rule.

Summary Chart :

Physical Topology	Common Cable	Common Protocol
Linear Bus	Twisted Pair Coaxial Fiber	Ethernet LocalTalk
Star	Twisted Pair Fiber	Ethernet LocalTalk
Star-Wired Ring	Twisted Pair	Token Ring
Tree	Twisted Pair Coaxial Fiber	Ethernet

17.3.2 Protocol

Networks use protocols, or rules, to exchange information through a single shared connection. These rules include guidelines that regulate the following characteristics of a network :

- Access method,
- Allowed physical topologies,
- Types of cabling, and
- Speed of data transfer.

These protocols prevent collisions of data caused by simultaneous transmission between two or more computers. Computers on most LANs use protocols known as Ethernet or Token Ring. An Ethernet-linked computer checks if a shared connection is in use. If not, the computer transmits data. Since computers can sense an idle connection and send data at the same time, transmitting computers continue to monitor their shared connection and stop transmitting if a collision occurs. Token Ring protocols pass a special message called a token through the network. A computer that receives the token is given permission to send a packet of information or if the computer has no packet to send, it passes the token to the next computer. The most common protocols are :

- Ethernet
- LocalTalk
- Token Ring
- FDDI
- ATM

17.3.2.1 Ethernet

The Ethernet protocol is by far the most widely used. Ethernet uses an access method called CSMA/CD (Carrier Sense Multiple Access/Collision Detection). This is a system where each computer listens to the cable before sending anything through the network. If the network is clear, the computer will transmit. If some other node is already transmitting on the cable, the computer will wait and try again when the line is clear. Sometimes, two computers attempt to transmit at the same instant. When this happens a collision occurs. Each computer then backs off and waits a random amount of time before attempting to retransmit. The Ethernet protocol allows for linear bus, star, or tree topologies. Data can be transmitted over wireless access points, twisted pair, coaxial, or fiber optic cable at a speed of 10 Mbps up to 1000 Mbps.

17.3.2.2 LocalTalk

LocalTalk is a network protocol that was developed by Apple Computer, Inc. for Macintosh computers. The method used by LocalTalk is called CSMA/CA (Carrier sense Multiple Access with Collision Avoidance). It is similar to CSMA/CD except

that a computer signals its intent to transmit before it actually does so. LocalTalk adapters and special twisted pair cable can be used to connect a series of computers through the serial port. The Macintosh operating system allows the establishment of a peer-to-peer network without the need for additional software. With the addition of the server version of AppleShare software, a client/server network can be established. The LocalTalk protocol allows for linear bus, star or tree topologies using twisted pair cable. A primary disadvantage of LocalTalk is speed. Its speed of transmission is only 230 Kbps.

17.3.2.3 Token Ring

IBM developed the Token Ring protocol in the mid-1980s. The access method used involves token-passing. In Token Ring, the computers are connected so that the signal travels around the network from one computer to another in a logical ring. A single electronic token moves around the ring from one computer to the next. If a computer does not have information to transmit, it simply passes the token on to the next workstation. If a computer wishes to transmit and receives an empty token, it attaches data to the token. The token then proceeds around the ring until it comes to the computer for which the data is meant. At this point, the receiving computer captures the data. The Token Ring protocol requires a star-wired ring using twisted pair or fiber optic cable. It can operate at transmission speeds of 4 Mbps or 16 Mbps. Due to the increasing popularity of Ethernet, the use of Token Ring has decreased.

17.3.2.4 FDDI

Fiber Distributed Data Interface (FDDI) is a network protocol that is used primarily to interconnect two or more local area networks, often over large distances. The access method used by FDDI involves token-passing. FDDI uses a dual ring physical topology. Transmission normally occurs on one of the rings; however, if a break occurs, the system keeps information moving by automatically using portions of the second ring to create a new complete ring. A major advantage of FDDI is speed. It operates over fiber optic cable at 100 Mbps.

17.3.2.5 ATM

Asynchronous Transfer Mode (ATM) is a network protocol that transmits data at a speed of 155 Mbps and higher. ATM works by transmitting all data in small packets of a fixed size; whereas, other protocols transfer variable length packets. ATM supports a variety of media such as video, CD-quality audio, and imaging. ATM employs a star topology, which can work with fiber optic as well as twisted pair cable.

Protocol Summary

Protocol	Cable	Speed	Topology
Ethernet	Twisted Pair,	10 Mbps	Linear Bus, Star,
Fast Ethernet	Twisted Pair, Fiber	100 Mbps	Star
LocalTalk	Twisted Pair	.23 Mbps	Linear Bus or Star
Token Ring	Twisted Pair	4Mbps-16 Mbps	Star-Wired Ring
FDDI	Fiber	100 Mbps	Dual ring
ATM	Twisted Pair, Fiber	155-2488 Mbps	Linear Bus, Star, Tree

17.3.2.6 TCP/IP

TCP/IP is a layered protocol, which means that after an application initiates the communication, the message (data) to be transmitted is passed through a number of stages or layers, until it actually moves out onto the wire. The data are packaged with a different header at each layer. At the receiving end, the corresponding programmes at each protocol layer unpackage the data, moving it “back up the stack” to the receiving application.

TCP/IP is composed of two parts : TCP (Transmission Control Protocol) and IP (Internet Protocol). TCP is a connection-oriented protocol that passes its data to IP, which is connectionless. TCP sets up a connection at both ends and guarantees reliable delivery of the full message sent. TCP tests for errors and requests retransmission if necessary, because IP does not. Although the OSI model is widely used and often cited as the standard, most Unix workstation vendor have used TCP/IP protocol. TCP/IP is designed around a simple four layer scheme. It does omit some features found under the OSI model. Also it combines the features of some adjacent OSI layers and splits other layers apart. The four network layers defined by TCP/IP model are as follows.

1. Layer 1 - Link : This layer defines the network hardware and device drivers.
2. Layer 2 - Network : This layer is used for basic communication, addressing and routing. TCP/IP uses IP and ICMP protocols at the network layer.
3. Layer 3 - Transport : Handles communication among programmes on a network. TCP and UDP falls within this layer.
4. Layer 4 - Application : End-user applications reside at this layer. Commonly used applications include NFS, DNS, arp, rlogin, talk, ftp, ntp and traceroute.

17.4 Standard Sayered Framework for Network Design (Open System Interconnection)

The Open Systems Interconnection (OSI) reference model has been an essential component of computer network design model since its inception in 1984. OSI is an abstract model, meaning that actual network implementations need not to adhere to it strictly. The ISO (International Standards Organization) has created a layered model, called the OSI (Open Systems Interconnect) model, to describe defined layers in a network operating system. The purpose of the layers is to provide clearly defined functions that can improve inter-network connectivity between “computer” manufacturing companies. Each layer has a standard defined input and a standard defined output.

This is a top-down explanation of the OSI Model. It starts with the user’s PC and it follows what happens to the user’s file as it passes through the different OSI Model layers. The top-down approach was selected specifically (vs. starting at the Physical Layer and working up to the Application Layer) for ease of understanding. It is used here to show how the user’s files are transformed (through the layers) into a bit stream for transmission on the network. These are the 7 Layers of the OSI model :

7. Application Layer (Top Layer)
6. Presentation Layer
5. Session Layer
4. Transport Layer
3. Network Layer
2. Data Link Layer
1. Physical Layer (Bottom Layer)

The OSI model divides the complex task of host-to-host networking, traditionally called internetworking, into layers. Layers in the OSI model are ordered from lowest level to highest in a stack. The OSI contains seven layers in two groups :

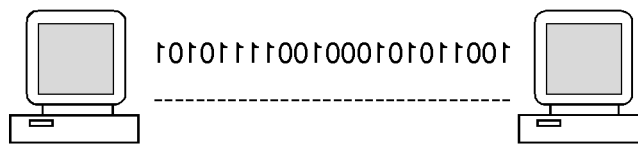
Upper Layers

OSI designates the application, presentation and session layers as “upper” layers. Generally speaking, software in these layers performs application specific functions like data formatting, encryption, and connection management. Higher layers cover network requests and responses, representation of data and network protocol as seen from a user’s point of view.

The Application layer supplies network services to end-user applications. Network services are typically protocols that work with user's data. For example, in a Web browser application, the Application layer protocol HTTP packages the data needed to send and receive Web page content. The Application layer provides data to (and obtained data from) the Presentation layer.

Lower Layers

The remaining lower layers provide more primitive network-specific functions like routing, addressing, and flow control. The lower layers deal with electrical signals, chunks of binary data, and routing of these data across networks.



The physical layer is responsible for the ultimate transmission of data over network communications media. It operates with data in the form of bits that are sent from the physical layer of the sending (source) device and received at the physical layer of the destination device. Ethernet cabling, Token Ring network technology and SCSI all function at the Physical Layer. Hubs and Repeaters are standard network devices that function at the Physical layer. At the physical layer, data are transmitted using the type of signaling supported by the physical medium :

- Electric Voltage
- Radio Frequencies
- Pulses of infrared or ordinary light

The OSI Model of computer Networks

Lower Layers	Physical	Hub, Repeater Ethernet, Token Ring, Ethernet, ATM, IP, IPX
	Data Link	
	Network	
	Transport	TCP, UDP, SPX
	Session	AppleTalk, Winsock
	Presentation	GIF, MPEG
	Application	HTTP, FTP, SMTP
Upper Layers		

Layer 7-Application Layer

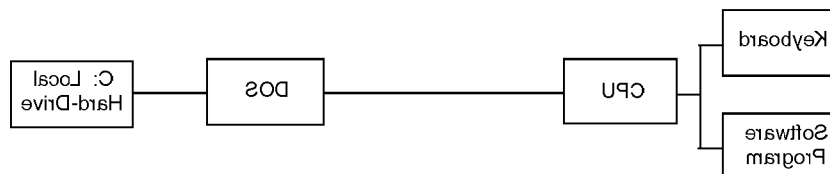


Fig. 1.1 : Basic PC Logical Flowchart

A basic PC logic flowchart is shown in Fig. 1. The keyboard & application are shown as inputs to the CPU (requesting access to the hard disk). The keyboard requests accesses through user inquiries and the application seeks access through “File Openings” and “Saves”. The CPU, through the Disk Operating System, sends and receives data from the local hard disk (“C:” in this example).

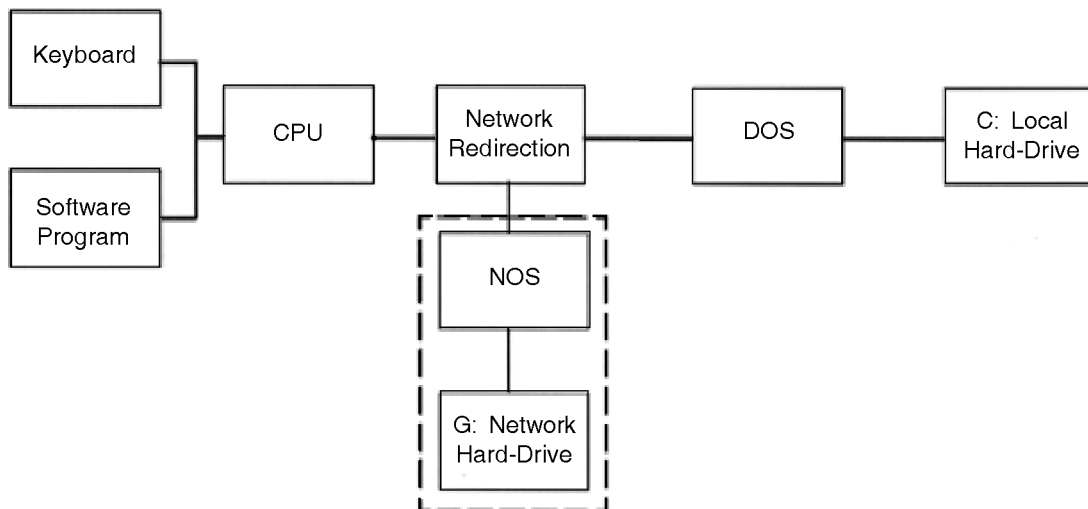


Fig. 2. : Simple Network Redirection

A PC setup as a network workstation has a software “Network Redirector” (the actual name depends on the network—we will use a generic term here) placed between the CPU and DOS (as shown in Fig 2.). The Network Redirector is a TSR (Terminate and Stay Resident) Programme : it presents the network hard disk as another local hard disk (“G:” in this example) to the CPU. The “Network Redirector” intercepts all CPU requests. The Network Redirector Checks to see if either a local or a network drive is requested. If a local drive is requested, the request is passed on to DOS. However, if a network drive is requested, the request is then passed onto the network operating system (NOS).

Electronic mail (E-Mail), client-server databases, games played over the network, print and file servers, remote logons, and network management programmes (or any “network aware” applications) are all aware of the network redirector. They have the ability to communicate directly with other “network applications” on the network. The “Network Aware Applications” and the “Network Redirector” make up Layer 7 (the Application layer of the OSI Model, as shown in Fig. 3).

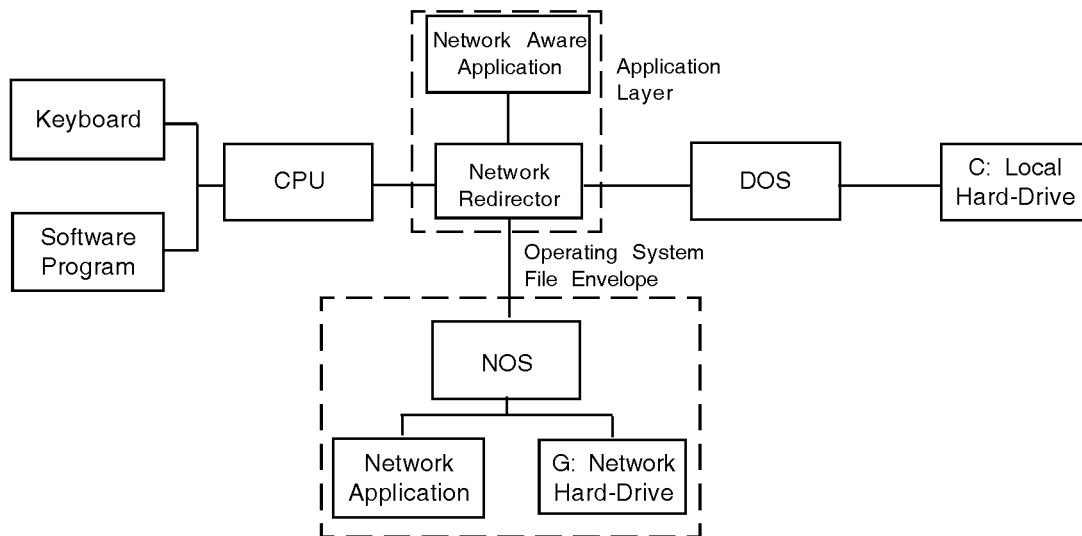
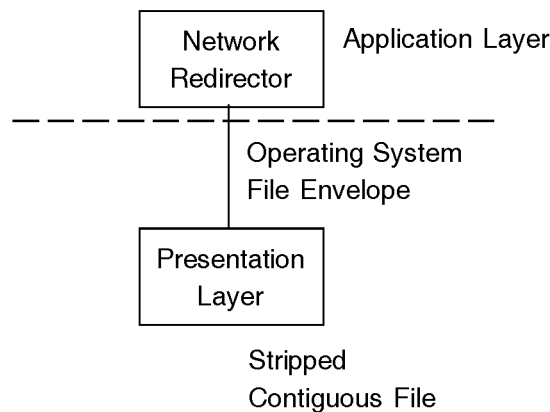


Fig. 3. : PC Workstation with Network Aware Software

Layer 6 - Presentation Layer

The Network Redirector sends CPU operating system native code to the network operating system : the coding and format of the data is not recognizable by the network operating system. The data consists of file transfers and network calls by network aware programmes.

For example, when a dumb terminal is used as a workstation (in a mainframe or minicomputer network), the network data is translated into (and from) the format that the terminal can use. The Presentation layer presents data to and from the terminal using special control characters to control the screen display (LF-line feed, CR-carriage return, cursor movement, etc..). The presentation of data on the screen would depend on the type of terminal that's used : VT100, VT52, VT420, etc.



Similarly, the Presentation layer strips the pertinent file from the workstation operating system's file envelope. The control characters, screen formatting, and workstation operating system envelope are all stripped or added to the file (if the workstation is receiving or transmitting data to the network). This could also include translating ASCII file characters from a PC world to EBCDIC in an IBM Mainframe world.

The Presentation Layer also controls security at the file level : this provides both file locking and user security. The DOS Share programme is often used for file locking. When a file is in use, it is locked from other users to prevent 2 copies of the same file from being generated. If 2 users both modified the same file, and User A saved it, then User B saved it, then User A's changes would be erased! At this point, the data is contiguous and complete (i.e. one large data file).

Layer 5-Session Layer

The Session layer manages the communication between the workstation and the network. The Session layer directs the information to the correct destination, and identifies the source to the destination. The Session layer identifies the type of information as data or control. The Session layer manages the initial start-up of a session, and the orderly closing of a session. The Session layer also manages Logon procedures and Password recognition (See Fig. 5).

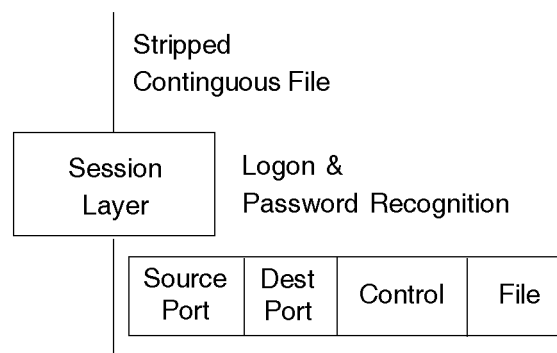
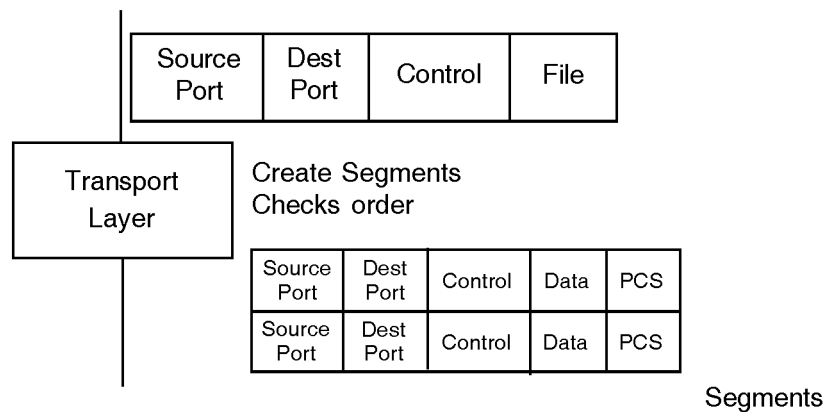


Fig. 5. : Session Layer

Layer 4-Transport Layer

In order for the data to be sent across the network, the file must be broken up into usable small data segments (typically 512-18K bytes). The Transport layer breaks up the file into segments for transport to the network, and combines incoming segments into a contiguous file. The Transport layer does this logically, not physically, and it is done in software as opposed to hardware.



The Transport layer provides error checking at the segment level (frame control sequence). This makes sure that the datagrams are in the correct order : the Transport layer will correct out of order datagrams. The Transport layer guarantees an error-free host-to-host connection. It is not concerned with the path between machines.

Layer 3-Network Layer

The Network layer is concerned with the path through the network. It is responsible for routing, switching, and controlling the flow of information between hosts. The Network layer converts the segments into smaller datagrams than the network can handle : network hardware source and destination addresses are also added. The Network layer does not guarantee that the datagram will reach its destination.

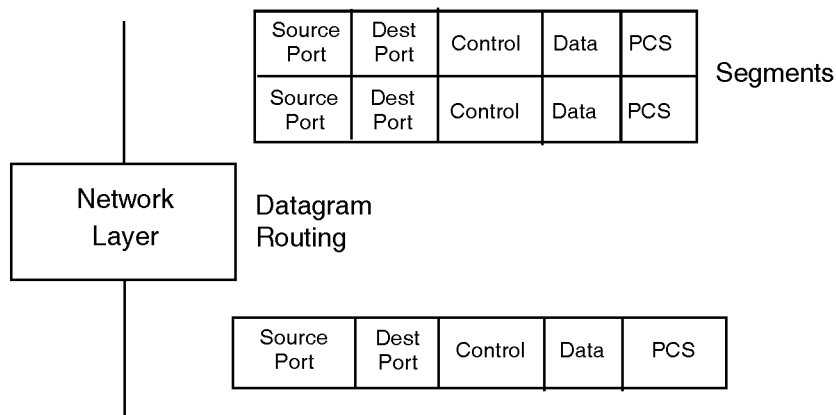


Fig. 7. : Network Layer

Layer 2 - Data Link Layer

The Data Link layer is a firmware layer of the network interface card. The data link layer puts the datagrams into packets (frames of bits : 1s & 0s) for transmission,

and assembles received packets into datagrams. The Data Link layer works at the bit level, and adds start/stop flags and bit error checking (CRC or parity) to the packet frame. Error checking is at the bit level only : packets with errors are discarded and a request for re-transmission is sent out. The Data Link layer is primarily concerned with bit sequence.

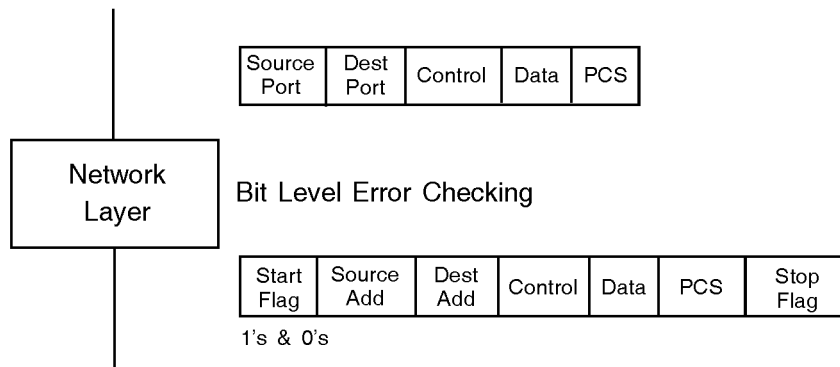


Fig. 8. : Data link Layer

Layer 1 - Physical Layer

The Physical layer concerns itself with the transmission of bits. It also manages the network card's hardware interface to the network. The hardware interface involves the type of cabling (*coax*, twisted pair, etc.), frequency of operation (1 Mbps, 10 Mbps, etc.), voltage levels, cable terminations, topography (star, bus, ring, etc.), etc. Examples of Physical layer protocols are as follows : 10 Base5- Thickenet, 10Base2- Thinnet, 10BaseT - twisted pair, ArcNet, FDDI, etc. (see Fig. 9).

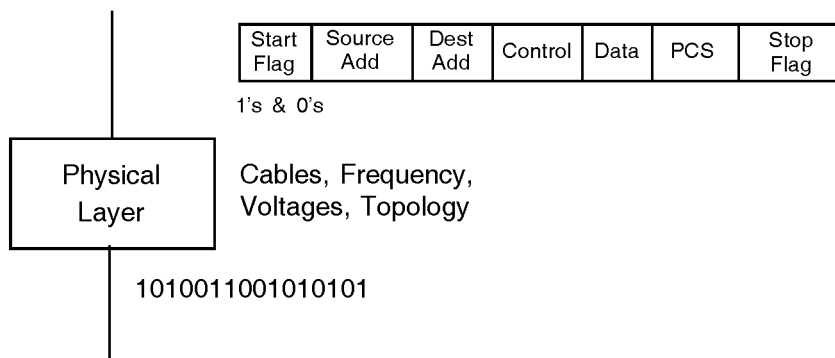


Fig. 9. : Physical Layer

Layer-Specific Communication

Each layer may add a Header and a Trailer to its Data (which consists of the next higher layer's Header, Trailer and Data as it moves through the layers). The Headers

contain information that specifically addresses layer-to-layer communication. For example, the Transport Header (TH) contains information that only the Transport layer sees. All other layers below the Transport layer pass the Transport Header as part of their Data.

Application Layer PDU



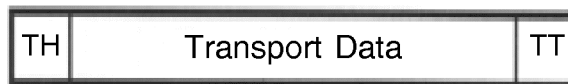
Presentation Layer PDU



Session Layer PDU



Transport Segment



Network Datagram



Data Link Packet



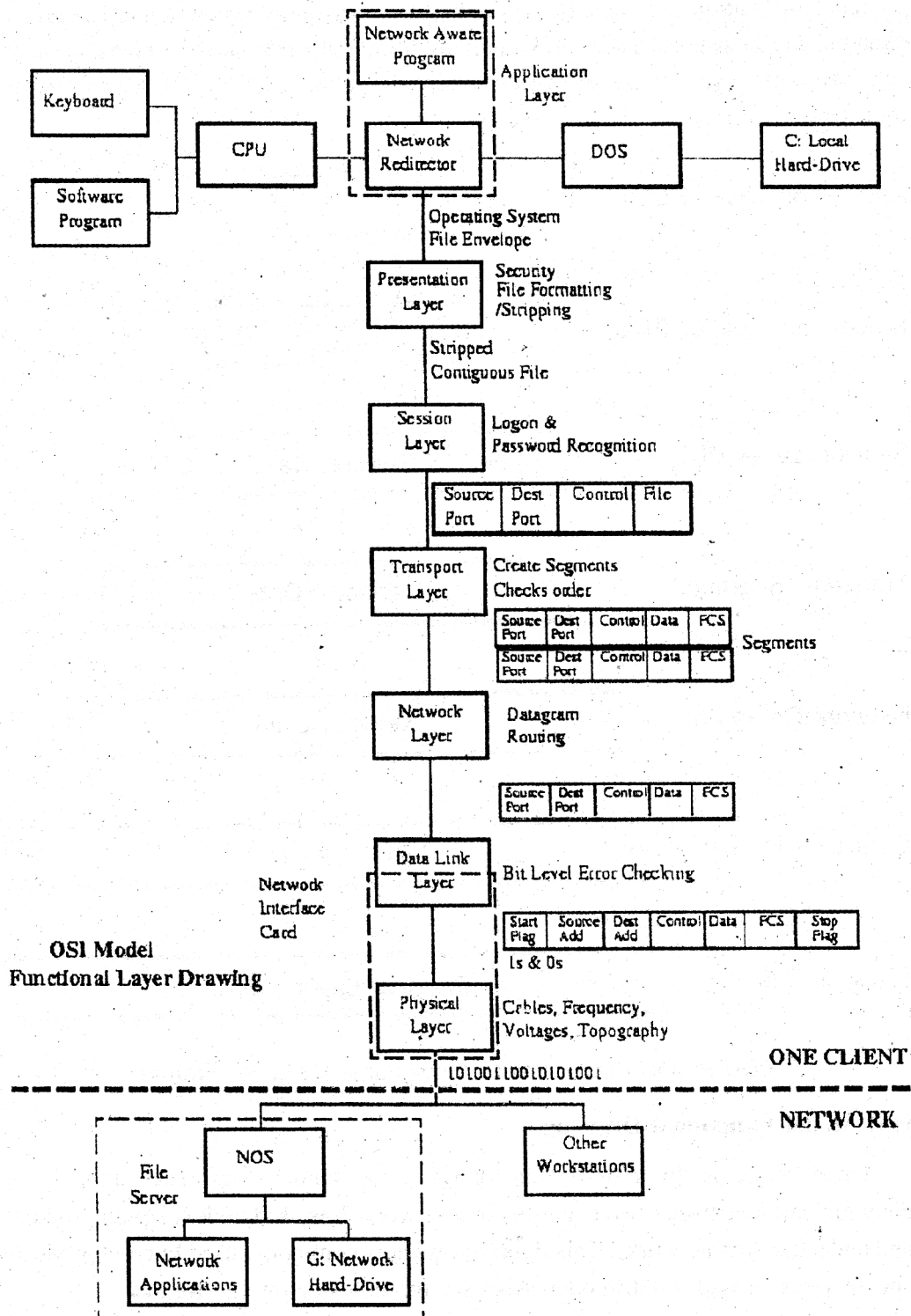
Physical Bits

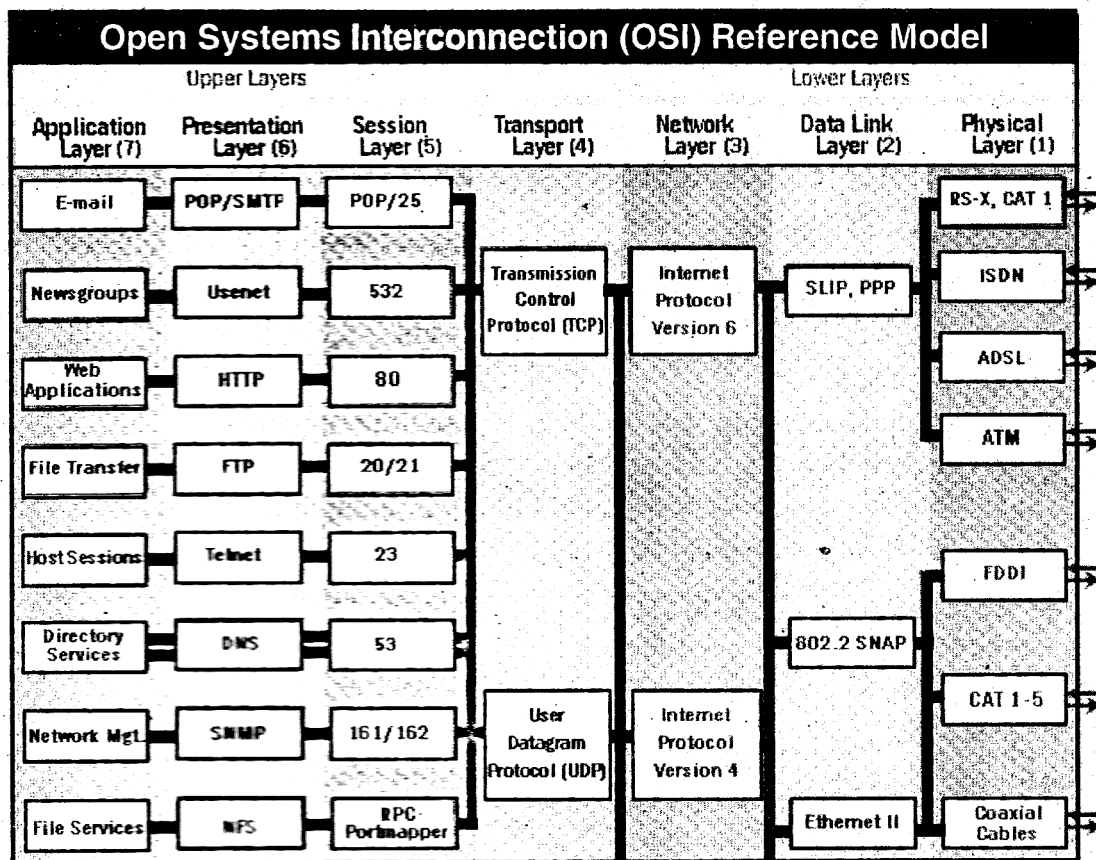


PDU-Protocol Data Unit (a fancy name for Layer Frame)

OSI Model Functional Drawing

Open Systems Interconnection (OSI) is a standard reference model for communication between two end users in a network. It is used in developing products and understanding networks. This figure shows how commonly used Internet products and services fit within the model. Also see the notes below the figure.





Benefits

The layered approach in the OSI model offers several advantages to system implementations. By design into logical smaller pieces, vendors can more easily solve network design problems. A product from one vendor that implements OSI Layer 2 functionality, for example, will likely to interoperate with another vendor's OSI Layer 3 product because both vendors are following the OSI model.

References and Further Readings

- 1 2001 Blanchard (Eugene). Introduction to the ISO-OSI model ([http://www.linuxports.com/howto/intro to networking/c4412.htm](http://www.linuxports.com/howto/intro%20to%20networking/c4412.htm)). Visited last : 12/10/2005
- 2 2005 OSI Reference Model illustrated (<http://searchnetworking.techtarget.com/sDefinition/0,,sid7gci523729,00.html>)
- 3 2001 Computer Desktop Encyclopedia. 9th ed. Osborne-McGraw-Hill, 2001

17.5 Exercise

1. Discuss different network topologies.
2. Discuss different major network protocols.
3. Discuss OSI model.

Unit 18 □ Bibliographic Information Networks

Structure

18.0 Objectives

18.1 Introduction

18.2 Bibliographic Standards

18.3 Bibliographic Information Network

18.3.1 Objectives

18.3.2 Control of Bibliographic Information Networks

18.3.3 Characteristics

18.3.4 Benefits

18.4 Information Programme and Systems Coordinated in India

18.4.1 International

18.4.2 Regional

18.4.3 Countrywide Area Network

18.4.4 Metropolitan Networks

18.5 Exercise

18.0 Objectives

The objectives of the Unit are to :

- Examine the concept of bibliographic information network (BIN)
- Explain the concept of standardization in library & information activities
- Discuss various aspects of bibliographic information networks-objectives, requirements, topologies, services etc.
- Present an overview of international, regional, national, and city based library and information networks
- Examine the treatments of bibliographic information networks in various policy statements of Government of India.
- Discuss the concept behind OAI

18.1 Introduction

Bibliographic networked started with the formal networking of libraries that had converted their catalogues into MARC records. Probably first successful bibliographic network was OCLC (Ohio Library College Centre, now known as Online Computer Library Centre), which was established in 1963. However, with the change of technology and its easy availability, many more library networks came into existence around the world.

18.2 Bibliographic Standards

A library/Network could devise its own method of organizing the bibliographic information, but it would isolate the library, limit its options, and create much more work for itself. Every national library system has two fundamental functions :

- ◆ To collect, preserve, and disseminate national print and non-print publishing and research output; and,
- ◆ To exchange their information with other national systems.

In other words, library materials should be organized in such a way that provides for effective retrieval and exchange. There is an urgent need for the national library system to standardize the way it organizes its collection for retrieval and exchange. What it needs are both a national cataloguing code and a national MARC format. This is not to say that cataloguing code not being currently used in libraries; rather, it implies that that cataloguing code should be standardized and implemented nationwide. The following Table explains the types of standards involved in a computerized library environment.

Cataloguing Standards	Enabling library to share catalogue records with other libraries, both in the country and overseas. AACR (Anglo-American Cataloguing Rules, 2nd edition) is a major international standard for the cataloguing of all types of materials collected by general libraries. The rules for description in Part I of AACR are based on the general framework for the description of library materials, the ISBD(G) - General International Standard Bibliographic Description. Close correspondence also exists between the chapters in AACR, which relate to different types of material and the corresponding ISBD.
-----------------------	--

	<p>The family of International Standard Bibliographic Descriptions (ISBD) specifies the requirements for description and identification of information resources. Responsibility for the development and maintenance of ISBD resides with the International Federation of Library Associations and Institutions. - Section of Cataloguing, in which the British Library plays an active role.</p>
Exchange Formats	<p>Formats are used to transfer data in a structured form. Exchange formats were developed in parallel with the development of computers and other electronic storage devices to facilitate the transfer of bibliographic data between computer systems.</p> <p>The MARC 21 formats are standards for the representation and communication of bibliographic and related information in machine-readable form. The Library of Congress in consultation with various user communities maintains the MARC 21 formats.</p> <p>The primary purpose of UNIMARC is to facilitate the international exchange of data in machine-readable form between national bibliographic agencies. UNIMARC may also be used as a model for the development of new machine-readable bibliographic formats.</p>
Name Authority Control	<p>Authority Control is the process of establishing and maintaining consistency in headings in a bibliographic file by means of an authority file.</p>
Subject Access	<p>The use of a standardized system of subject headings allows compatible access by subject across different files and catalogues. In libraries document classification enables ordering of material in a helpful way on open shelves, browsing and retrieval of related items in catalogues, meaningful arrangement in subject indexes and bibliographies, analysis of the collectios, and increasingly provides structural tools for mapping and organizing Web resources.</p>
ISSN	<p>The International Standard Serial Number (ISSN) is an internationally accepted code, which identifies the title of serial publications. The ISSN is not connected with ownership of the journal, nor does it confer copyright or protect the title of the serial from use by other publishers. The ISSN is not related to legal deposit. The users of ISSN are :</p>

Publishers who wish to identify their serial publications and incorporate a barcode on the magazine, journal or newspaper for sale via the major retailers, *Libraries*, which use the ISSN as a fundamental identifier for distinguishing between identical serial titles and facilitating checking and ordering procedures, collection management, legal deposit, interlibrary loans etc., *Catalogue databases*, which use the ISSN as a record control number and can make use of the records on the ISSN register, *Documentation centres and databases*, which handle bibliographic references and use the ISSN for more accurate serials citation, abstracting and indexing services etc. *Subscription agencies* that act as intermediaries between publishers and their customers use the ISSN to ensure the correct serial publication is ordered, *Academics* who wish to cite in full details of publications for research purposes.

18.3 Bibliographic Information Network

The pressure for resource sharing due to the devaluation of the currencies in different countries, rising costs of published materials, information explosion, and limited infrastructure resulted design and development of networks. It is set of inter-related information systems associated with communication facilities, which are cooperating through more or less formal agreements in order to implement information handling operations to offer better services to the users. Bibliographic information networks are responsible for collection of bibliographic information on various subjects and making them searchable to the users through computer terminals within a campus and/or over the Internet. These networks are responsible for

- Collection
- Processing and Dissemination
- Development of standards and procedures
- Modernization of hardware and software

A bibliographic information network makes information available on information materials issued within the territory it serves and other information materials included in the libraries and collections of the territory it serves. Bibliographic Information Networks contain more bibliographic records, but database consisting of textual and/or numerical information are also very common.

Networks are set up to improve and develop the bibliographic services already provided by the individuals libraries and other sources of information within their territory, and to plan and coordinate those which individual organizations cannot themselves provide, and to serve as recognized centres for all requests not met elsewhere for bibliographic information and as a clearing houses for international inquiries. Where necessary they will also put inquiries in touch with other sources of bibliographic information.

18.3.1 Objectives of Bibliographic Information Networks

Bibliographic Information Networks take many forms. There is steadily increasing number of national Networks. In addition, regional networks are a features of a number of countries, and there are number of networks serving smaller areas. Among the many activities undertaken by networks in various parts of the world, the following are undertaking, though no existing networks includes all of them :

- Copyright registration of some or all forms of information materials issued within the territory served.
- Formation of union catalogue of such materials
- Provision of catalogue entries for those materials
- Provision for various forms of centralized cataloguing
- Formation of union catalogues of foreign materials in the stock of libraries and collections within the territory.
- International exchange and ILL
- International coordination of bibliographic work and standardization
- Reproduction of information materials as a substitute for national or international loan
- Publication of guides to libraries and special collection and their resources within the territory.
- Coordination of bibliographic information and records
- Provision of access to unpublished theses and dissertations
- Identification of materials
- Studies and supports library automation
- Compilation of retrospective national/regional/local bibliographies
- Forwarding requests to the appropriate individual libraries and collections
- Dissemination of bibliographic information of materials published in or about the territory.

- Provision of advice, direction, and supervision of local bibliographic activity; its simulation, coordination, and recording. Finally, reporting such activity to international organizations
- Motivate smaller libraries to participate and benefit from the network to maximize the national development.
- To strengthen bibliographic control of the country's own technological output and to establish computer based bibliographic databases in subject areas of interest to the region.
- Stimulate and promote creation of non-bibliographic databases in different areas.
- Improve national information infrastructure.
- Develop and promote the technical and organizational structure and capabilities for exchange of data.
- Coordinate with other local, regional, national and international networks.
- Formulate policies, programmes and standards for different areas of activities.

18.3.2 Control of Bibliographic Information Networks

Control of bibliographic networks depends in most cases on the main sources of a center's income. Where cooperating libraries mostly provide support voluntarily, control usually takes the form of an elected council of representatives of member libraries. In countries where the centre draws the greater part of its income from government funds, it is usually made responsible to the appropriate government department, or, to the national or national university library (ies). In the latter case, it may therefore be housed at or near the national library. In other case it is customary to choose a strategic site for the centre, and the choice may be influenced by possibilities of cutting overheads. Many centres draw a small income from various activities.

Many national and international bibliographic databases are now available in computerized forms. There are various types of bibliographic databases :

- Catalogues
 1. Individual library
 2. Union Catalogue
- Bibliographic Databases
 1. National
 2. Books-in-Prints
- Abstracting & Indexing Services

- Full-Text Databases
 1. Journals/Serials
 2. News papers
 3. E-Books
- Numerical/Factual databases
- Digital Libraries.

18.3.3 Characteristics of Bibliographic Information Networks

Library cooperation is not a new phenomenon. However, computerization of libraries provides an opportunity for formal cooperation among the automated libraries at a scale, which was not possible earlier. The characteristics of bibliographic information networks are :

- Data : Bibliographic data in MARC format, Numeric data, full text databases
- Retrieval : Author/title/series/subject terms/keywords/document codes
- Access : LAN and/or Internet
- Users : From any where
- Equal opportunity of access
- Interdependence
- Standards and quality
- Shared decision making
- Intergration and Coordination
- Centralization
- Cost and productivity

18.3.4 Benefits of Bibliographic Information Network

There are various benefits of development of bibliographic information networks. Salient benefits are as follows :

- Better bibliographic control
- Easy to ascertain availability
- Wider access to Information
- Rationalization of resource development policies and programme
- Promotion and protection of copyright and IPR
- Maximization of uses of resources
- Effective and efficient document delivery service

18.4 Information Programme and Systems Coordinated in India

It is relevant to mention that during the past decades several important international information programmes and systems have been established. Many libraries from India are also contributing to there programme and system. A brief descriptions of efforts to establish International, Regional and National Bibliographic/Information Networks is given below :

18.4.1 International Networks

18.4.1.1 INIS [<http://www.iaea.org/inis/>]

The **International Nuclear Information System** (INIS) is information system on the peaceful uses of nuclear science and technology. INIS is operated by the International Atomic Energy Agency (IAEA) in collaboration with its Member States and co-operating international organizations.

INIS provides a comprehensive information reference service for literature on the peaceful applications of nuclear science and technology. To do this, INIS processes most of the world's scientific and technical literature that falls within its subject scope and maintains a database which contains millions of bibliographic references, making it the world's most comprehensive information source on the peaceful applications of nuclear science and technology. Additionally, INIS also maintains a unique collection of full text non-conventional (grey) literature that would be difficult to obtain elsewhere.

18.4.1.2 AGRIS [<http://www.fao.org/agris/>]

AGRIS is the international information system for the agricultural sciences and technology. The Food and Agriculture Organization of the United Nations (FAO) created it in 1974, to facilitate information exchange and to bring together world literature dealing with all aspects of agriculture. AGRIS is a cooperative system in which participating countries input references to the literature produced within their boundaries and in return, draw on the information provided by the other participants. To date, 240 national, international and intergovernmental centres participate.

The AGORA site provides access to over 400 journals from major scientific publishers in the fields of food, agriculture, environmental science and related social sciences. AGORA is available to students and researchers in qualifying not-for-profit institutions in eligible developing countries.

18.4.1.3 POPIN [<http://www.un.org/popin/home/about.html>]

The population Information Network (POPIN) was founded on 9 May 1979, by resolution 1979/33 of the United Nations Economic and Social Council. POPIN strives to make international, regional and national population information, particularly information available from United Nations sources, easily available to the international community. The development of POPIN was partially supported by the United Nations Population Fund (UNFPA).

The Population Information Network is a decentralized community of population institutions organized into global, regional and national networks. Global POPIN activities, including this web site, are established by and within the United Nations Population Division. Regional population networks have been established by the

- Economic Commission for Africa (POPIN Africa),
- Economic and Social Commission for Asia and the Pacific (Asia-Pacific POPIN),
- Economic Commission for Latin America and the Caribbean (IPALCA),
- Economic and Social Commission for Western Asia (POPIN Western Asia) and the
- Association for Population/Family Planning Libraries and Information Centers-International for Northern America (Northern American POPIN).

Regularly issued publications which provide updates on POPIN activities include :

- Population Newsletter
- Asia-Pacific POPIN Bulletin
- PADIS Newsletter
- Boletín del PROLAP
- APLIC-International Communicator.

Bibliographic Databases maintained by the global and regional POPIN networks include :

- DOCPAL : Bibliographic database of citations to Latin American population literature. Produced by the Latin American Demographic Centre (CELADE) of the United Nations Economic Commission for Latin America and the Caribbean (ECLAC/CEPAL). Also available in CD-ROM format from CELADE.

- EBIS POPFILE : Bibliographic database of citations to Asia-Pacific population literature. Produced by the Economic and Social Commission for Asia and the Pacific (ESCAP). Also, available on diskette from ESCAP. For national databases, please contact Asia-Pacific POPIN.
- PADIS bibliographic database of citations to African population literature. Produced by the Economic Commission for Africa (ECA). Also, available on diskette from the ECA.

For further information about the Population Information Network, please contact : The Director, Population Division, Department of Economic and Social Affairs, United Nations Secretariat, 2 United Nations Plaza (Rm.DC2-1950), New York, NY 10017, USA. Telephone : (212) 963-3179; Fax : (212) 963-2147.

18.4.2 Regional

Significant regional information networks are : ASTINFO and TECHNINET ASIA.

18.4.2.1 ASTINFO [<http://www.unesco.org/webworld/regoff/astinfo.htm>]

ASTINFO is a co-operative programme, which aims to promote the exchange of information and experience in science and technology among countries in the Asia/Pacific region. It was established in 1983 as a result of the Second Conference of the Science Ministers and Economic Planning Bodies in the Asia/Pacific region (CASTASIA II). held in March 1982 in Manilla (Philippines). ASTINFO comprises co-ordinating units in 18 Member states; and some 82 national/regional institutions now hold the status of ASTINFO Associated Centres and Networks. A quarterly Newsletter is published. The aims and benefits of ASTINFO include :

- Computerization of library and Information Management Services;
- Pilot projects on Specialized information systems and services;
- Development of National Information Policies and Standards;
- Innovative approaches to education and training of library and information personnel;
- Establishment of document delivery systems and services;
- Serving as a forum for communication and information exchange.

18.4.2.2 TECHNINET ASIA [<http://www.technonet.org.sg/>]

Technonet Asia is a non-stock, non-profit and non-political international development organization comprising 9 Small Micro and Medium Enterprises (SMME) promotion and development organizations (PO) from 8 Asian countries, with its Secretariat based in Singapore.

Technonet Asia has gained international recognition as a dynamic organization in the promotion of technical cooperation among developing countries and has been highly acclaimed as an effective model in North-South and South-South collaboration achieved through sharing and exchange of institutional expertise.

As a cooperative network, its mission is to improve the efficiency and competitiveness of small micro and medium enterprises (SMMEs) in member countries through the employment of innovative approaches for enhancing its contributions to their respective economies.

Technonet achieves its objectives through action-oriented programmes incorporating a suitable blend of consultancy, training, institution building and appropriate management methods. These are effected through the following initiatives :

- Formulation and implementation of new and innovative support programme for SMMEs promotion and development in the Asia-Pacific region.
- Promotion of capacity building among SMME support institutions in the member countries.
- Conduct of regional scale techno-economic research studies on policy initiatives, best practices and successful programmes
- Implementation of multi-country technical assistance projects supported by international donor organizations and serves as an effective conduit to channel assistance to SMME institutions to carry out priority based development programmes.
- Maintain information database on SMME practitioners and professionals to serve on call in the member countries in the specific areas of SMME needs.
- Actively promoting and facilitating the networking among member organizations in the region.

Technonet Asia was established in 1973 as a project of the International Development Research Center of Canada. The organization was registered as an autonomous legal entity in January 1980 in the Republic of Singapore. It is now recognized as a regional non-government organization (NGO).

Technonet Asia is governed by a Council composed of heads of the POs and the Executive Director and is headed by a Chairman elected annually. Coordination and administrative functions are performed by Technonet Center (Secretariat) in Singapore. Through a special arrangement with the Participating Organizations, Technonet Center plans and implements the national and regional programmes.

Technonet Center, based in Singapore, acts as the secretariat of the network and is headed by a full-time Executive Director. It maintains a core of programme

and administrative staff and is responsible for project formulation, planning, implementation, management, monitoring and evaluation.

18.4.3 National

18.4.3.1 Bibliographic Information Networks in India

The growth of library networks in India may be traced back to the policies and programme that were initiated by the Government of India/State Government since 1950s, however, it was actually in the 1980s when the organized efforts were made for collection and dissemination of information. Various policy papers included recommendation for development of library network. Since 1988 many networks have been established in India-INFLIBNET, CALIBNET, DELNET etc. however, none of these networks really achieved any great success. Winding up of NISSAT may clearly indicate the ground condition. Brief description of selected library networks is provided here :

- INFLIBNET
- DELNET
- CALIBNET
- ADINET

The explosion in the amount of literature that is available, increases among the number of users and their different needs, and the application of electronic media are forcing libraries to construct and participate in networks. Retrieval through telecommunications networks and access to international databases are available for searching for information on various subjects. With the advent of networks, remote transmission of texts and graphics, video clips and animated clips are also possible. Some factors that are responsible for the development of library and information networks in India are :

- The report of the working group of the planning commission on modernization of library services and informatics for the seventh five year plan, 1985-90.
- The National Policy on Library & Information systems document (1986) accepted by the ministry of HRD, Government of India.
- The report on national policy on university libraries prepared by the Association of Indian Universities (1987).
- The UGC report on information systems for science and technology under the Department of Science & Industrial Research (DSIR) Government of India has been vigorously promoting an integrated approach to library automation and networking.

A network may fail in the early stages if there is not proper planning or if adequate funds are not available. Moreover, a common memorandum of agreement signed by the participating libraries at the institutional level is essential for the success of a network venture. On a more practical level, catalogue data must be in a standard, machine-readable form for it to be shared and exchanged. And finally, a continuous flow of external assistance is crucial for the network's survival. These points may explain the status of different networks in India.

18.4.3 Countrywide Area Network

18.4.3.1 VIDYANET

Title : VIDYANET (Dedicated Communication Computer Net)

Sponsor : TATA Institute of Fundamental Research, Bombay

Objectives : To provide rapid means of communications by linking computers at various institutions in India to similar networks outside the country; to stimulate corporate research, the day-to-day exchange of research information and the execution of joint projects and publications

Services : File transfer facility; sharing of computer resources and access to remote applications, databases, libraries, etc.

18.4.3.2 INFLIBNET

INFLIBNET is an autonomous Inter-University Centre of the University Grants Commission (UGC) of India. It is a major National Programme initiated by the UGC in 1991 with its Head Quarter at Gujarat University Campus, Ahmedabad. Initially started as a project under the IUCAA, it became an independent Inter University Centre in 1996. INFLIBNET is involved in modernizing university libraries in India and connecting them as well as information centres in the country through a nation-wide high-speed data network using the state-of-art technologies for the optimum utilization of information. INFLIBNET is set out to be a major player in promoting scholarly communication among academicians and researchers in India.

The objectives of INFLIBNET as envisaged in Memorandum of Association are :

- To promote and establish communication facilities to improve capability in information transfer and access, that provide support to scholarship, learning, research and academic pursuit through cooperation and involvement of agencies concerned.
- To establish **INFLIBNET : Information and Library Network** a computer communication network for linking libraries and information centres in

universities, deemed to be universities, colleges, UGC information centres, insitutions of national importance and R&D institutions, etc. avoiding duplication of efforts.

In order to fulfill the broad objectives, INFLIBNET will :

- Promote and implement computerization of operations and services in the libraries and information centres of the country, following a uniform standard.
- Evolve standards and uniform guidelines in techniques, methods, procedures, computer hardware and software, services and promote their adoption in actual practice by all libraries, in order to facilitate pooling, sharing and exchange of information towards optimal use of resources and facilities.
- Evolve a national network interconnecting various libraries and information centres in the country and to improve capability in information handling and service.
- Provide reliable access to document collection of libraries by creating online union catalogue of serials, theses/dissertations, books, monographs and non-book materials (manuscripts, audio-visuals, computer data, multimedia, etc.) in various libraries in India.
- Provide access to bibliographic information sources with citations, abstracts etc. through indigenously created databases of the Sectoral Information Centres of NISSAT, UGC Information Centres, City Networks and such others and by establishing gateways for online accessing of national and international databases held by national and international information networks and centres respectively.
- Develop new methods and techniques for archival of valuable information available as manuscripts and information documents in different Indian languages, in the form of digital images using high-density storage media.
- Optimise information resource utilization through shared cataloguing, inter-library loan service, catalogue production, collection development and thus avoiding duplication in acquisition to the extent possible.
- Enable the users dispersed all over the country, irrespective of location and distance, to have access to information regarding serials, theses/dissertations, books, monographs and non-book materials by locating the sources wherefrom available and to obtain it through the facilities of INFLIBNET and union catalogue of documents.
- Create databases of projects, institutions, specialists etc. for providing online information service.

- Encourage co-operation among libraries, documentation centres and information centres in the country, so that the resources can be pooled for the benefit of the weaker resource centres by stronger ones.
- Train and develop human resources in the field of computerized library operations and networking to establish, manage and sustain INFLIBNET.
- Facilitate academic communication amongst scientists, engineers, social scientists, academics, faculties, researchers and students through electronic mail, file transfer, computer/audio/video conferencing, etc.
- Undertake system design and studies in the field of communications, computer networking, and information handling and data management.
- Establish appropriate control and monitoring system for the communication network and organize maintenance.
- Collaborate with institutions, libraries, information centres and other organizations in India and abroad in the field relevant to the objectives of the Centre.
- Create and promote R & D and other facilities and technical positions for realizing the objectives of the Centre.
- Generate revenue by providing consultancies and information services.
- Do all other such things as may be necessary, incidental or conducive to the attainment of all or any of the above objectives.

During the recent period quite a large number of libraries and information centers are forming networks. The advent of computer networking as an accepted part of the library and information infrastructure has had a very significant impact on the way in which library and information systems are perceived. India is thus on the threshold of a new era of computer communication networks both for general purposes and for library and information purposes.

18.4.3.3 DELNET [<http://www.delnet.nic.in/about-ourselves.htm>]

DELNET has been in operation since January 1988 and was registered as a society in 1992. It was initially sponsored by the National Information System for Science and Technology (NISSAT), Department of Scientific and Industrial Research, Government of India and is currently being promoted by the National Informatics Centre, Department of Information Technology, Ministry of Communications and Information Technology, Government of India and India International Centre, New Delhi.

DELNET has been established with the prime objective of promoting resource sharing among the libraries through the development of a network of libraries. It aims to collect, store and disseminate information besides offering computerized services to users, to coordinate efforts for suitable collection development and also to reduce unnecessary duplication wherever possible.

DELNET has been actively engaged with the compilation of various Union Catalogues of the resources available in member-libraries. It has already created the Union Catalogue of Books, Union List of Current Periodicals, Union Catalogue of Periodicals, CD-ROM Database, Database of Indian Specialists, Database of Periodical Articles, Union List of Video Recordings, Urdu Manuscripts' Database, Database of Theses and Dissertations, DEVINSA Database, sample databases of language publications using GIST technology and several other databases, The data is being updated in each of these databases and is growing rapidly. All the DELNET databases have been resident on DELSIS, and in-house software developed on BASISPlus, an RDBMS, the product of Information Dimensions Inc. of USA, which has been provided to DELNET courtesy National Informatics Centre, New Delhi.

DELNET provides an array of facilities including E-mail to its member-libraries including both institutional and associate institutional members. DELNET'S relentless efforts in resource sharing have proved extremely effective. It has indeed been a big leap towards the modernization of libraries in India.

18.4.4 Metropolitan Networks

18.4.4.1 CALIBNET [<http://www.calibnet.org/aboutus.htm>]

CALIBNET, a Government of India Project, has been launched by the National Information Systems for Science and Technology (NISSAT), Department of Scientific & Industrial Research (DSIR); and managed by the CALIBNET *Society* established under the West Bengal Government's Societies Registration Act 1961.

CALIBNET aims to provide the individual libraries and their reading members with cost-effective solutions to their information problems. The primary objective of the Project is building access to library & information resources available in the eastern region. This has been pursued through implementation of a series of Databases, Bibliographic, Factual and Intellectual Assets of West Bengal. This apart, CALIBNET provides its Participating Members and the user community at large with the following :

- Active links in this Web Page to access
 - Indian Library & Network Resources
 - Overseas Library Resources on India
 - Worldwide Library Catalogues
 - National Libraries of the World
 - Newspapers & Journals
 - Electronic Reference Tools
 - Factual Information Sources
 - Document Supply Services, and
- Varied Programmes for
 - On-Demand Information Services
 - Consultative Service for Library Automation
 - Manpower Development Opportunities, and
 - Research & Development in IT Applications

18.4.4.2 ADINET [[http://www. alibnet.org/Objectives.html](http://www.alibnet.org/Objectives.html)]

ADINET is an Information Network of Libraries in and around Ahmedabad. ADINET was registered as a Society in October 1994. National Information System sponsored it for Science and Technology (NISSAT), Department of Scientific and Industrial Research, Government of India. The objectives are to :

- Bring about cooperative mode of working amongst libraries and information centers in and around Ahmedabad.
- Integrate the economic, scientific and technical information systems into an effective network in and around Ahmedabad.
- Facilitate and promote sharing of resources amongst the libraries and information centres in and around Ahmedabad by developing and maintaining a central online Union Catalogue containing bibliographic information on books, serials and non-book materials of all the participating libraries.
- Coordinate with other regional, national and international networks, libraries, information and documentation centres for exchange of information and documents.
- Offer technical guidance to member libraries on selecting, storing, sharing and disseminating information.
- Coordinate efforts for suitable collection development and reduce unnecessary duplication whenever possible.

- Develop databases of projects, specialists and institution in and around Ahmedabad.
- Create awareness amongst all users' groups and to educate them in the utilization of information.
- Develop resources and to propagate information in ways appropriate to the needs of users in and around Ahmedabad.
- Help library and information centre users and also individuals who practice different professions in getting specialized information of their interest.

18.4.4.3 MYLIBNET [[http://www. mylibnet.org/](http://www.mylibnet.org/)]

There are 116 colleges/institutions affiliated to University of Mysore, out of which 34 colleges are located within Mysore city. All the libraries within Mysore city will be networked in the first phase with the following objectives :

- To **share resources** available with all the libraries.
- To provide a **faster communication** to all the libraries through Electronic Mail facility.
- To develop **software tools for better library management**.
- To create awareness in the field of latest Information Technology by conducting seminars/workshops/training programmes.
- To setup a information base in collaboration with industries.
- To conduct surveys.
- To **flash arrival of new books/journals**, announcement of events like seminar/workshop/training programmes.

References and Further Readings

- 1 2004 Jebaraj (Franklin David) and Devadoss (Fredric Robert). Library and information networks in India (Library Philosophy and Practice 6(2), 2004). (<http://www.webpages.uidaho.edu/~mbolin/lppv6nhtm>)
2. 1999 Bakker (Trix). Resource sharing in a virtual library environment : user oriented collection management. (<http://www2.fmg.uva.nl/sociosite/bakker/resovirt.html#intro>). 1999.
3. 1997 Resource sharing in changing environment-Library trends 45(3), Winter 1997. p 367-573
4. 1993 Godden (Irene P), ed. Advances in librarianship. V17. San Diego : Academic Press, 1993
5. 1969 Encyclopedia of library and information science. Ed by Allen Kent and Harold Lancour. NY : Marcel Dekker. 1969-. Various volumes

18.5 Exercise

1. What is bibliographic information network ?
2. Discuss the characteristics of library networks.
3. Discuss Information Programme and Systems Coordinated in Inida.

NOTES

NOTES

NOTES