# LOCAL AREA NETWORK

Shaikh Hasibul Karim

## INTRODUCTION

During the twentieth centrury, the key technology bloomed is information gathering, processing, and distribution. Among other developments, we have seen the installation of worldwide telephone networks, the invention of radio and television, the birth and unprecedented growth of the computer industry, and the launching of communication satellites.

During the last two decades a whole branch of engineering has been developed to facilitate the transmission of digital information from point to point. This new technology has been called variously 'computer communications', 'data transmission' or 'teleprocessing', and represents a marriage between traditional electronics, information theory, and computer science. As long as only modest quantities of data were being transmitted over long distances, the overall impact of data transmission on computing was essentially minimal. With the advent of low-cost microprocessor and minicomputer systems together with the clustering of a number of such devices, in a relatively small area (an office, factory, polytechnic, university, or laboratory), a need for a specialized form of inexpensive communication network was felt. Thus, a branch of data transmission dealing with the transfer of large quantities of information at high speed between 'geographically distributed computers' arose. This new field was given the name 'Local area network (LAN)'. A LAN serves to link togther the computer facilities on a given site. Since it covers a single site, it should be a private system. That is, it does not form part of the public telephone network and is therefore not subject to the mass constraints normally associated with a public network.

## NETWORK TOPOLOGIES FOR A LAN

The topology of a network describes the way in which the individual users of the network are linked together. Four basic network topologies are suitable for use in a LAN. These are—

1. The unconstrained topology
2. The star network
3. The bus network
4. The ring network

**1. The unconstrained topology:** This sort of network is the most general one (figure-1). The individual nodes / stations are connected together in an arbitrary fashion(Note that in the figures the nodes or stations are represented by microcomputers). Its advantage is that further nodes and links can readily be added without disturbing the hardware of the existing system.

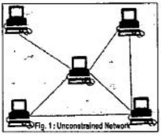

Fig. 1 : Unconstrained Network

The disadvantage of the unconstrained topology is that a decision must be made at each node on the best way to route a message on the way to its destination. Each node must have its own 'road-map' and make a decision on which link the message is to be transmitted on the way to its destination.

**2. The star network:** Figure -2 shows how the star network eliminates the needs for nodes to make routing decisions, by routing all messages from source to destination via one central node (master station).

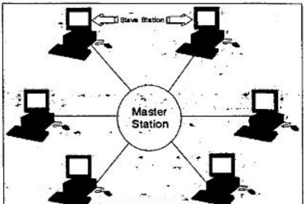


Fig. 2 : The Star Network

The star network has two obvious disadvantages. As all messages pass through the central node, the loss of the central node totally wrecks the network. Further more, because every traffic passes through the central node, it must be capable of working at a sufficient high speed to handle all nodes to which it is connected.

**3. The bus:** The bus topology (Figure-3) is an attempt to minimize the complexity of a network by both removing a special purpose central node and the need for individual nodes to make routing decisions.

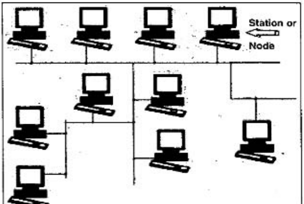


Fig. 3 : The Bus Network

In a bus all nodes are connected to a common data highway. When a message is put on the bus by a node, it flows outwards in all directions and eventually reaches every point in the network. The bus has one topological and one practical restriction. Only one path may exist between any two points, otherwise there would be nothing to stop a message flowing round a loop for ever. The practical limitation is that the bus cannot normally exceed some maximum distance from end to end.

**4. The ring:** Figure-4 illustrates the ring topology, where the nodes are connected together in the form of a ring. Like the bus this leads to a decentralized structure, as no central node is needed to control the ring. Each node simply receives a message from one neighbour and passes it on to its other neighbour. Messages flow in

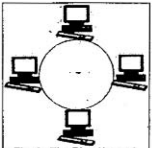

Fig. 4 : The Ring Network

one direction round the ring.

The only routing requirement placed on each node is that it must be able to recognize a message intended for itsefl. One question may arise; what happens when any portion of the ring breaks? There are a number of 'double ring' structures with two links between each of the nodes. If one of the links is broken it is possible for the ring to reconfigure itself and by pass the failure.

We shall now deal with the contention or collision problems. Which are encountered in any kind of network systems.

**THE BUS CONTENTION OR THE COLLISION PROBLEM**

The principal problem faced by the designers of a bus is how to deal with a number of nodes wanting to use the bus at the same time. This is called bus contention. Similar sort of problems are encountered in the other topologies of the network.

In a contention network any node wishing to transmit just goes ahead and puts its message on the bus. As there is no control over when a node may transmit, there is nothing to prevent two or more nodes transmitting simultaneously. If this does happen, there must be a collision of messages and all messages being transmitted are irretrievably scrambled and lost. Different types of procedures are followed in different LAN systems to control the contention and collision problem. In the following section the IEEE (Institute of Electrical & Electronic Engineers) Standard 802 LANs are discussed along with the processes which they adopt in controlling collision problem.

**IEEE STANDARD 802 FOR LANS**

IEEE (Institute of Electrical & Electronic Engineers) has produced several standards for LANs. This standards, collectively known as IEEE 802, include CSMA/CD (Carrier Sense Multiple Access with collision Detection), Token bus and token ring.

* **IEEE STANDARD 802.3 (CSMA/CD)**

CSMA/CD uses the following model to avoid collision.

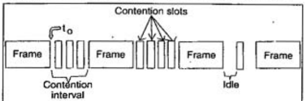


Fig. 5 : CSMA/CD can be in one of threeststes : contention, transmission or idle

At the point marked $t_o$, a station has finished transmitting its frame. Any other stations having a frame to send may now attempt to do so. If two or more stations decide to transmit simultaneously, there will be a collision. Each will detect the collision, abort its transmission, wait a random period of time, and then try again, assuming that no other station has started transmitting in the meantime. Therefore CSMA/CD model will consist of alternating contention and transmission periods, with idle periods occurring when all stations are quiet (e.g., for lack of work).

** **IEEE STANDARD 8-2.4 : TOKEN BUS**

This standard, 802.4, is called a 'Token Bus'. Physically, the token bus is a linear or tree-shaped cable onto which the stations are attached. Logically, the stations or nodes are organized into a ring (see figure-6), with each station knowing the address of the station to its 'left' and 'right'. When the logical ring is initialized, the highest numbered station/node may send the first frame. After it is done, it passes permission to its imediate neighbour by sending the neighbour a special control frame of data bits called a token. The token propagates around the logical ring, with only the token holder being permitted to transmit frames. Since only one station at a time holds the token, collisions do not occur.

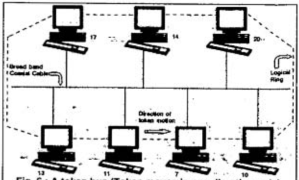


Fig. 6 : A token bus (Token moves in one direction only)

*** **IEEE STANDARD 802.5 : TOKEN RING**

In a token ring a special bit pattern (the token) is passed round the ring from station to station. The station currently holding the token is the station which can transmit data if it so wishes. If it does not wish to take the opportunity to send data itself, it passes the token round the ring. For example, suppose the token has the special pattern 11111111. A station on the ring wishing to transmit, monitors its incoming traffic. When it has detected seven ones it inverts the last bit of the token and passes it on. Thus, a pattern called a 'Connector' (11111110) passes on down the ring, the connector is created to avoid sending the eighth '1' and thereby passing on the token. The station may now transmit its data. After it has transmitted its data, it sends a new token down the ring. As there is only one token, contention cannot arise on the ring unless, of course, a station becomes antisocial and sends out a second token.

**CONCLUSION**

There are many published papers on the relative advantages and disadvantages of the various types of local area network. But one can always find a set of parameters that makes one of the LANs look better than the others. The only general statement that is inarguable is that an overloaded 802.3 LAN will collapse totally, but an overloaded token-based system will have an efficiency approaching 100 percent. For people planning to run their LAN in overloaded mode, 802.3 is definitely not the way to go. For people planning to run with light to moderate load, all three perform well, so that factors other than the performance are probably more important. Since all three LAN types are likely to coexist for years to come, the issue of interconnecting different LANs is an important one. We may discuss that topic some other time in future.

**References :**

[1] The principles of Computer Hardware— by Alan Clements, 1st Edition, Oxford University Press (U.K.).

[2] Computer Networks (2nd Edition) —by Andrew S. Tanenbaum. Prentice, Hall International (U.S.A.).

# PROGRAMMING WITH STYLE

Syed Sabbir Ahmed

To write clear, well-designed programs need practice and some knowledge about structured-programming. This article gives you a general idea of structure and some important tips on programming.

Two totally independent skills are required in computer progra-mming. These are (a) solving problems (of structuring), and (b) writing down problem solutions in a particular programming language.

**STRUCTURE:**

The idea of structure is very simple but it is difficult to find the right structure to solve a particular problem. Problems are clearly understood when they are viewed as a list of things to be done and structuring is a way of ordering the tasks in a coherent way. In most cases when we look at a program listing we see many procedures ordered in a random way. The program may be very logical in small areas but is a complete mess when taken as a whole (a randomly ordered list of clear tasks). Thus the procedures must be logically ordered so that a program has a true logical structure which reflects exectly what it does.

To achieve the above goal we must consider terms like 'hiera-rchies' and 'trees' and two other terms in connection i.e. 'nesting' and 'level'. The idea of hierarchy can be understood by the following analogy:

Consider an artist who is making a sketch of a cockroach which lies under his microscope (Fig 1). The artist can zoom in and out by altering the magnification of the microscope and can thus examine as large or as small an area as is necessary. The artist uses lowest magnification to sketch the general shape of the cockroach. Next by zooming in on a small part of the cockroach the genearal details are drawn. This is repeated until the general shape is filled with general details. Having done this, the artist goes back to add the finest details at the maximum magnification until the picture is complete.

The connection with progra-mming is straightforward. Each part of the drawing process is actually a procedure. The magnification relates to the 'level' at which the artist is working and so the structure diagram also splits into levels. Low magnifications are called high levels because one looks at the object as a whole; high magnifications are called low levels.

The structure diagram shows how the low levels are related to the high levels, hence showing the overall scheme of the program in a much detail way. In this respect structure diagrams are superior to flow charts. This type of problem solving is also called 'Top Down analysis'.

Each procedure in a program (or its structure diagram) branches out into other procedures which lie in the level below. The procedure in a level actually serves the root procedures in the higher level which uses them. In short : a structure diagram consists of one main procedure at 'LEVEL O', which branches out to a new LEVEL called 'LEVEL 1' and contains all those procedures which are used by 'LEVEL O'.

The structure diagram can be squashed sideways so that the levels are maintained and the procedures are listed one after another inside their respective levels. In this way one can easily see the status of a procedure in the hierarchy, also it becomes easy to find a procedure when debugging. The above scheme is not complete yet. There will be some procedures which are needed throughout all the levels, such as procedures for drawing certain shapes. Actually these are routines of very general nature, which behave like basic tools for the main procedures. So these procedures of routines can be place in special levels called toolkits'. Toolkits are needed again and again throughout the program and can be isolated and re-used. Toolkits result in routine librories. Programs constructed in this way are highly portable, since it is only necessary to replace a particular toolkit in order to carry a program over to a new machine.

Lastly, in programs written in complex operating environments such as WIPM environments, there are logically idependent areas of a program called 'zone'. They can be places for interfacing with a operating system, for instance. They could also be used as an alternative to object languages.

**PRESENTATION :**

Another very important factor in computing is presentation. Programming languages are as much for communicating ideas to computers. So a well written program is easy to understand and also easy to debug. Given below are some guidelines which will help you to some extent to write neat and clear programs.

1) Layout & use of space : All procedures and functions must be written in their respective levels. A procedure should never be longer then a page of A4 and should rarely be that long. Both the levels and the procedures themselves should be clearly segregated, with a banner heading each level/toolkit. An area should be reserved on the right-hand side of the page (or screen) for comments and notes. Each new program statement should be on a new line.

2) Names and indentifiers : The names of the procedures should clearly reflect their functions. The names of the variables should clearly reflect their usage. Never choose names without good reason. Never use GOSUB if PROC or procedure will do instead. Take advantage of actual/formal parameters to make the variables consistent and sensible: that is,when variables are passed as arguments to several procedures, make sure that a whole program. If a procedure works out the total of a list of figures call it something like FINDTOTAL or FINDTOT. If a variable stands for the total value of something, call it TOTAL not X.

3) Comments and documentation : Each procedure must have at least one brief comment describing its function. Further comments should be written in the documentation of the program. Documentation must include a speification of exactly what the program does or does not and should contain notes about when the program will fail etc.

Example :

```
find_total (array : a);
begin
...... { Comments should be separated from program code };
end;
```

4) Loops & nested loops :Loops and their surrounding brackets should be indented and should be clearly visible. Always put a block bracket (that is {,}, begin, end etc.) on a line of its own. Example:

```
for n := 1 to 10 do
begin
writeln ('....');
end
```

When loops are nested, indent for every loop and make the level of nesting clear. For example write,

```
for (n =1; n<= 10 ;n++)
{
while (a< b)
{
printf ("......")
printf (".......")
}
printf ("......);
}
```

in preference to

```
for (n =1; n<=10 ; n++)
{while  { a <b) {
printf (".........");
printf ("..........");}}
```

5) **Goto :** It is never necessary to use GOTO. If the need arise, it is a sign that the program is poorly constructed or inappropriate for the given language.

# Hewlett-Packard and EDS Join World Cup France '98

FRANCE 98

Information Technology (IT) being a factor for the success of organization of complex project like World Cup, received top priority from the France Organizing Committee of 1998 World Cup Football.

After a close competition between the best French and foreign IT giants, the Committee finally chose Electronic Data Systems (EDS) and Hewlett-Packard (HP) as first two official suppliers IT systems of 1998 World Cup.

To fulfill the ambition of implementing most advanced IT system in the history of World Cup, the Committee has choosen two partners in the forefront of technology and progress, more than three years before the event. By providing the committee with their best serivices and hardware support, EDS and HP will gradually become key players in the organization of the 1998 World Cup.

The signing of these first two sponsorship agreements with two companies of international repute to team up with France 1998 is an evidence of the confidence they have in the France Organizing Committee. In coming months six other multinational companies will join EDS and HP as official suppliers of the 1998 World Cup Football.

**Hewlett-Packard**

The World's second largest manufacturer in the field of computers, HP an American company based in Palo Alto, California, is the leader for Unix systems and for printers. Founded in 1939, HP has more than 98,000 employees in some 120 countries. In 1994 HP had a turnover of US$ 25 billion of which 54% was achieved in markets outside the USA.

**IT Challanges of World Cup '98**

One of the specific challanges of the 1998 World Cup is the geographic distribution of venue cities within France. The matches will be held in the 10 cities of Bordeaux, Lens, Lyon, Marseille, Montpellier, Nantes, Paris, St-Denis, St-Etienne and Toulouse.

A network linking together the headquarters of the Organizing Committee, the International Broadcast Centre, the main Press Centre (all located in Paris), as well as each of the host cities will thus have to be installed.

Each host city will have France Organizing Committee offices, a stadium, a hospitality village, a press centre, an accreditation centre, the FIFA hotel, press hotels and a warehouse.

The network set up which will be based on a client/server achitecture, will thus include approximately one hundred local networks, linked togather by a wide area network.

HP will have its goal to provide the France Organizing Committee with all hardwares and networking expertise required for the successful organization of the 1998 World Cup, using the latest technological innovations.

This hardware will include:

- Central servers
- Local servers in the different host cities
- PC Workstations for the press and the international needs of the committee (about 1800 units).
- Printers
- Network equipment (Concentrators, routers, cards etc.)
- CAD workstations and plotters
- Scanners

HP will also provide advices to determine the best technical architecture and maintain the hardware.

**EDS**

Founded in 1962, EDS became an independent subsidiary of General Motors in 1984. In 36 countries it employs more than 80,000 people to supply IT services to more than 8,000 customers with a global turnover of US $ 10.5 billion in 1994.

**IT Goal of World Cup 98**

The first goal of the Information Technology Department of the French Organizing Committee is to design, develop, install and manage the IT systems needed for the preparation and running of the 16th Football World Cup.

In one way or another, IT is integrated into almost every aspect of the organization like-accreditations, ticketing, transportation, accommodations, communication etc. Considering the great number of persons to coordinate and to serve, and the need to have these people communicating using reliable, relevant and timely data, the IT solution becomes a critical factor to ensuring the success of the World Cup.

The IT applications can be divided into four domains:

- Results processing system (especially statistics)
- Information and communication system of the World Cup. In this field EDS shall use latest technological innovations.
- Operational management of World Cup (accreditations, volunteers, uniforms, accommodations, protocol, transport and ticketing management).
- In-house management of the French committee itself.

As the integrator, EDS will have the overall responsibility of the IT project and will supply the following services:

- IT consulting
- Application design and development
- Systems integration
- Determination of technical architectures
- Technical support
- Hardware and software installation
- Server and network facilities management
- On-site operations supprot

**Azam Mahmood**