INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION

INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
34100 TRIESTE (ITALY) - P.O.B. 586 - MIRAMARE - STRADA COSTIERA 11 - TELEPHONES: 224281/2/3/4/5/6
CABLE: CENTRATOM - TELEX 460392-I

SMR/90 - 17

COLLEGE ON MICROPROCESSORS:

TECHNOLOGY AND APPLICATIONS IN PHYSICS

7 September - 2 October 1981

AN INTRODUCTION TO DISTRIBUTED COMPUTING - I

R.W. DOBINSON

EP Division
CERN
1211 Geneva 23
Switzerland

EMC/0016K/
R.W. Dobinson
CERN, 1211 Geneva
Switzerland
Summer, 1981

AN INTRODUCTION TO DISTRIBUTED COMPUTING

(Lecture Notes for Students)

AIMS OF THESE LECTURES

* To introduce you to the concept of distributed computing, to explain a little of its historical development and to present some general advantages and disadvantages.

* To present some formal classification schemes for the different types of distributed systems.

* To give practical examples of these different types of distributed computing trying to bring out the different problems to be solved and the techniques used in their implementation.

* To give a simple overview of a vast field, assuming you know very little.

* To provide references for further study.

NON-AIMS

* To cover all topics

* To provide an in-depth study of any topic, especially software

* To make assumptions about your previous knowledge.

My definition of distributed computing:

"Distributed computing is concerned with the interconnection of processors and peripherals to form a co-operative system".

Dictionary definition of co-operative:

"The banding together for mutual benefit and advantage".

My corollary to this is:

"It is the job of the hardware engineer to implement the required interconnections and the much more difficult task of the software specialist to hide their existence from end users of the system".

1. A LITTLE HISTORY AND GEOGRAPHY

How distributed computing came about in the field of large computers

Until comparatively recently "the economy of scale" has applied to computing. Stated quite simply this meant you got more number crunching power per $, per Sfr, etc., if you purchased and ran ran one large computer rather than several small ones. This led to a situation where, in the mid-60's, computer centres were established to serve many users spread over a wide geographical area (Fig. 1). To start with, the users were distributed but the computing was not! The computer, the CPU and all peripherals, resided in the same room and the users came to it, or rather, they brought or sent their programs to be submitted to it. For example, when I was a graduate student in London, a delivery van called at my college once a day. It would collect my deck of punched cards and if I was lucky, bring the results back the next day. This was not very convenient!.

A step forward was to distribute some of the computer peripherals at remote locations; the University of London Computer Centre got around to installing a remote card reader and lineprinter at my college just after I left in 1968 (see Fig. 2). This innovation was certainly much appreciated at the time but still fell short of allowing people to work with the computer from their offices. They still had to go to relatively few job entry points,

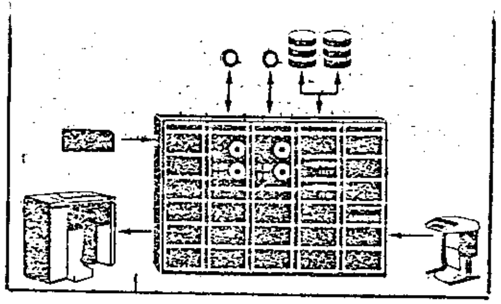


Fig 1 Computer centre with no remote facilities

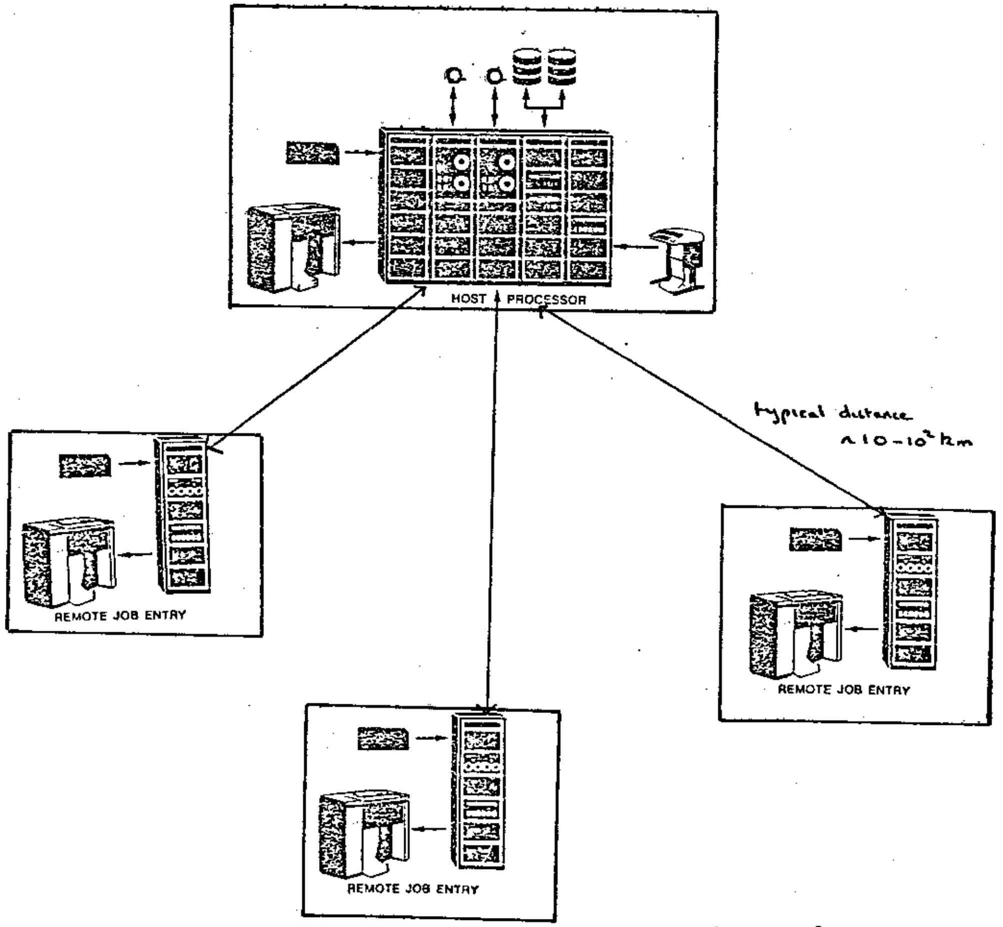


Fig 2 Computer centre with remote job entry

submit their program, and hang around following its progress through the computer system until eventually the results came out. An improved way of distributing computing came through the use of time-shared systems (Fig. 3). Many different users can work in offices and laboratories in very many widely scattered locations. They can communicate by means of teletypes and VDUs with the central computer as if it were in the adjacent room. Programs stored on its discs can be edited and submitted for execution, the results can be examined, all without moving from your office desk. In some cases a user might even talk to his or her program whilst it is running; for example, to give it some input parameters it needs. This then is how distributed computing grew up on large general purpose computers. Now let's go to the other end of the spectrum and investigate how it evolved in the field of mini and microcomputers.

Minicomputers; microcomputers and distributed computing

Large computers, whatever their virtues, when it came to sheer computational ability lacked two attributes: responsiveness and cheapness. For applications such as the automation of scientific experiments where one needed to perform frequent measurements, or control equipment in a feedback loop, large computers were just not suitable. One did not want to use a mulit-million dollar system to read a digital voltmeter and open a valve if the value was outside limits! With the advent of "cheap" minicomputers in the early 1970s a whole new field of computer automation became possible. A trend which was still further accelerated when microprocessors arrived on the scene some five years later. Minicomputers were quickly being used for a whole range of specialized jobs requiring fast and easy input/output but relatively little pure computation. Typically, minis opened and closed relays, changed voltages, measured temperatures and pressures and recorded counting rates from nuclear experiments (Fig. 4). Minicomputers also made possible dramatic advances in office automation; text editing, the production of reports, invoicing, stock control and so on. Very soon it became interesting to link together individual minicomputers. Let's look at some of the reasons for doing this.

1) Minis are cheap to buy and run in their basic form but the cost of adding peripherals can be high. It is therefore worthwhile sharing expensive peripherals, discs, MTs and lineprinters. Linking up minis allows you to do this (Fig. 5).

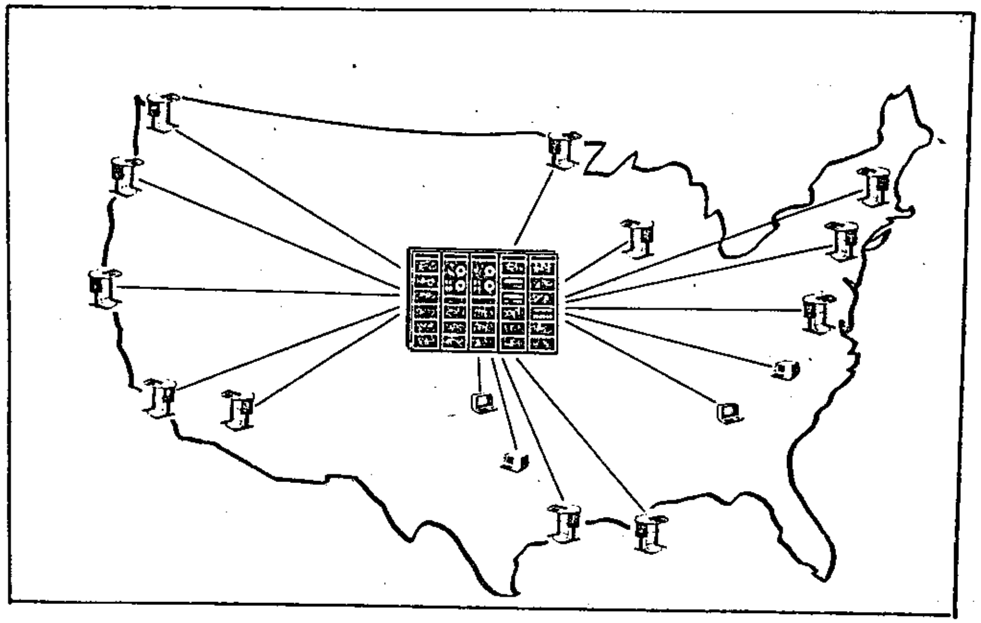


Fig 3 a time shared central computer system with widely distributed terminals

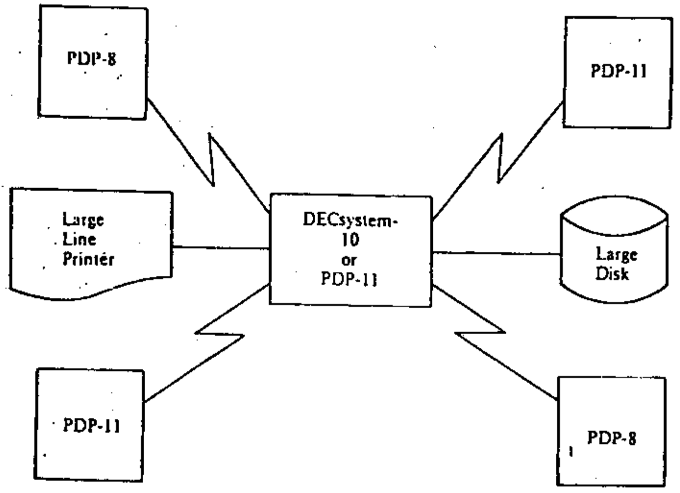


Fig 5 linking up computers to share peripherals

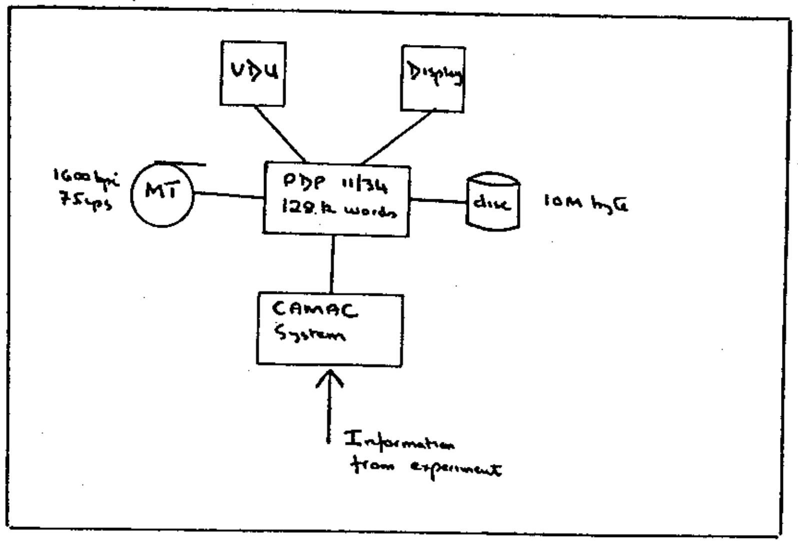


Fig 4 typical mini computer system used at CERN (European Organisation for Nuclear Research) for experimental study of fundamental particles

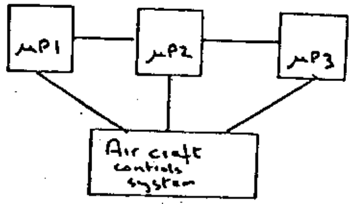


Fig 6 linking up microprocessors for reliability

2) Reliability: if one processor goes down another can take over or the system can continue running in a slightly degraded form (Fig. 6).

3) The need to supervise and correlate individual activities. Several minis could each be running a separate industrial process, from time to time they need to send data to a supervisor computer that produces an overall performance log for the entire plant. Commands are also required to radiate from the supervisor computer, i.e. "there is a drop in water pressure, shut down your process" (Fig. 7).

4) To increase overall computing capacity (Fig. 8).

On top of all this there are often cases where minis can usefully be linked to larger mainframe computers. One example is where from time to time the mini needs back-up in order to do some sizeable computation beyond its own ability (Fig. 9). Another is where a mini is used to off-load work from a large computer; it could act as a front end processor for a cluster of terminals (Fig. 10).

Some other examples of distributed computing

I now want briefly to remind you of some other types of rather specialized distributed computing. Very often what is called a computer is actually made up of several processors. Manufacturers have often designed systems that use two (or more) CPUs in tandem, thereby increasing the overall computing power. Single CPUs may use multiple specialized arithmetic units (e.g., an integer processor and a floating point processor) and dedicated processors are used in some systems for performing I/O.

Advantages and disadvantages of distributed computing

Already at this stage we can draw up a list of pros and cons for distributed computing. Some points we have mentioned previously, some things are obvious if you think about them, and others will become clearer as we go along.

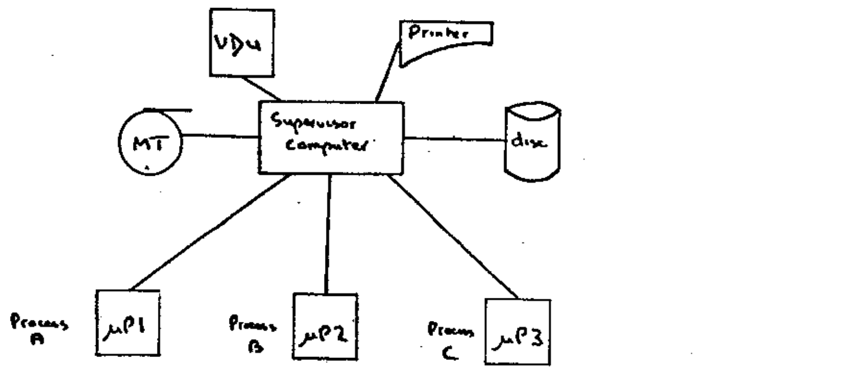


Fig 7 linking up computers to allow supervision and control

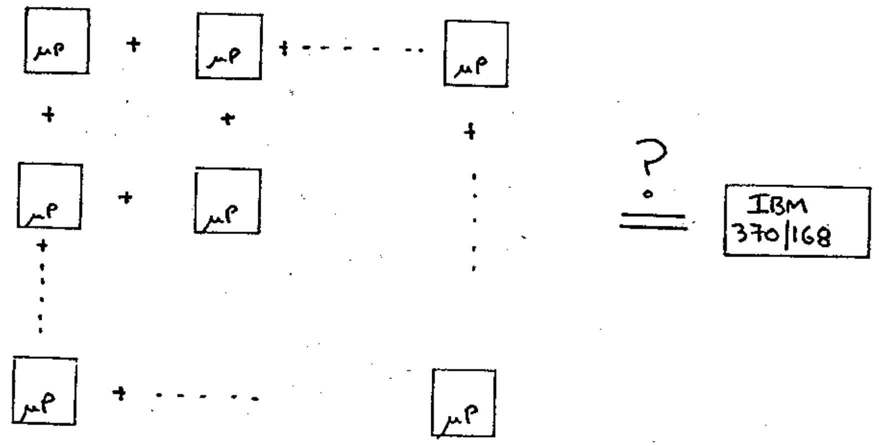


Fig 8 linking up computers to increase overall computing capacity

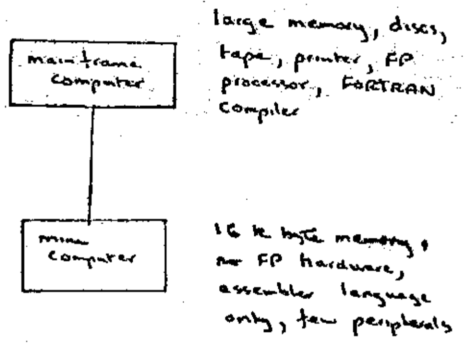

mainframe computer
large memory, discs, tape, printer, FP processor, FORTRAN compiler
mini computer
16 k byte memory, no FP hardware, assembler language only, few peripherals

Fig 9 mini computer will use mainframe for tasks beyond its capability eg) to run a large complicated program, for editing etc.

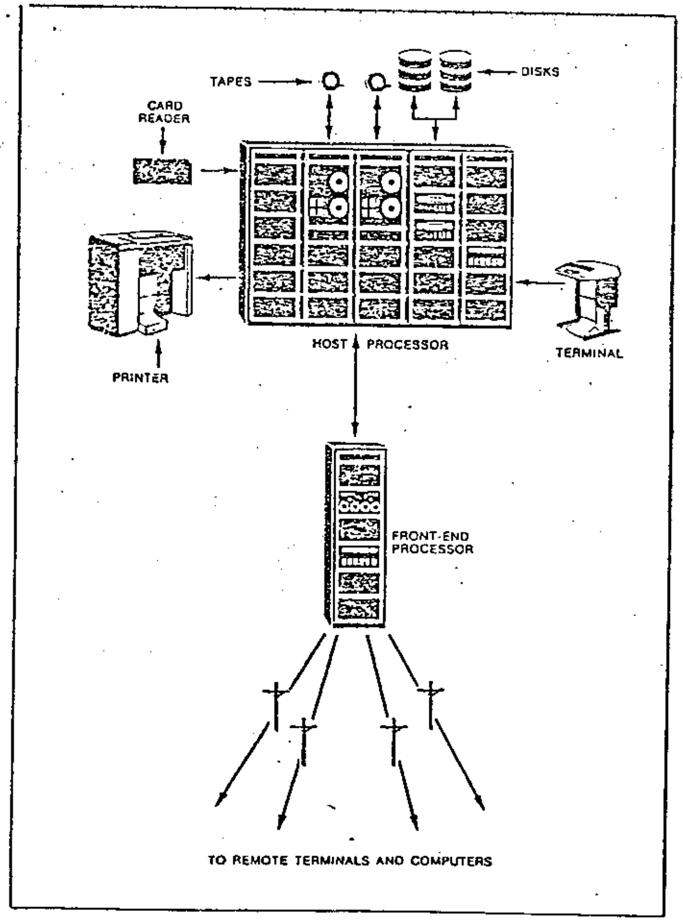


Fig 10 use of a front end proces to off load work from a large computer

Advantages

* Distributes computer power to users so as to permit convenient access and interactiveness.
* Offers modularity, expandibility and flexibility.
* Allows pooling of resources, load sharing and sharing of peripherals.
* Can increase reliability and fault tolerance.
* Increases computational power.
* Increases responsiveness.
* Access to specialized services.
* Information centralization, archiving.
* Possible to closely tailor systems to needs and to more rapidly adapt capacity to needs.

Disadvantages

* Increased software complexity.
* More dependence on communication technology.
* May in some cases decrease reliability.
* More difficult to test and debug.
* Unique expertise needed during design and development phase.

2. CLASSIFICATION OF DISTRIBUTED COMPUTING SYSTEMS

Is it possible to classify distributed computing systems? Yes, one can try. Although beware of taking any classification system too literally. I will present two schemes, of course, they are not completely independent of each other and neither is perfect. There is, however, an advantage in looking at things in more than one way.

One simple way of classifying distributed computing systems

Let us first try to make use of some of the examples already mentioned and classify systems into the following three types (this classification scheme comes from the book "Computer Engineering" mentioned in the references):

- multi-ALUs
- multiprocessors
- computer networks

For each type we will discuss its general characteristics including the degree of responsiveness of different parts of the system to each other (the degree of coupling between components), the physical distance separating them, transfer speeds, and so on.

Figure 11 illustrates a multi-ALU system. The different component processors are typically mounted on adjacent or even the same printed circuit boards. The typical response time, as typified by the time to pass information between the control unit and the processing elements is very short, 10 ns → 1 μs. We can say these systems are very strongly coupled.

Figure 12 shows the basic structure of a multiprocessor system. The component processors are typically micros or minis of the same or similar types. These processors have shared access to primary memory. Interprocessor communication is carried out by directly sharing data in this primary memory. Typical response times are ∿ 1 - 50 μs, the time to access shared data. Multiprocessor systems are described as tightly coupled. The components are normally in the same or adjacent racks.

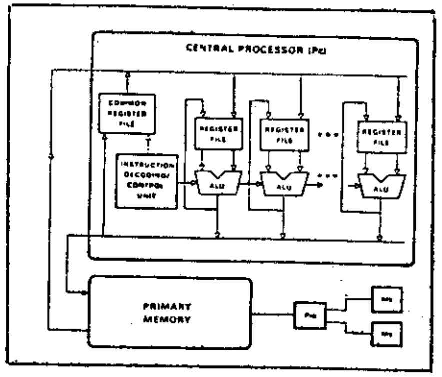


Fig 11 multi ALU system

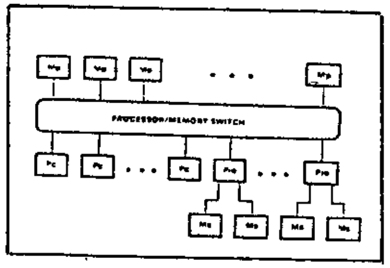


Fig 12 multiprocessor system

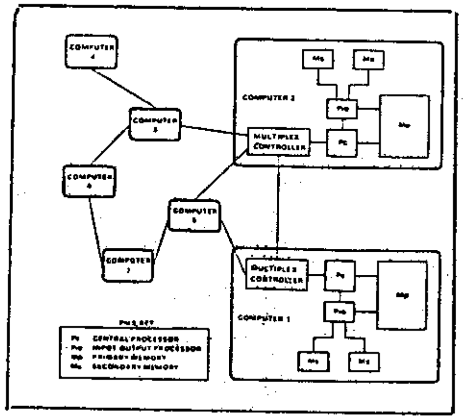


Fig 13 Computer network

Figure 13 shows a computer network. In this type of multiple processor organization each processor has a conventional computer architecture. There is no shared memory. The processors are interconnected by a very wide variety of mechanisms, parallel and serial paths, low speed and high speed, cables, fibre optics and even via satellites. The geographic separation and the data transmission bandwidth vary enormously. Component processors can be in the same room or on different continents. The bandwidth can vary enormously from $\sim 10^2 - 10^8$ bits/s.

The spectrum of computer networks is wide, however; in what follows it will be very convenient to divide them into two types:

- short haul or local area networks (Fig. 14)
- long haul or wide area networks (Fig. 15)

The former type spans distances typically up to a few kms. Interconnections are normally totally under the control of a single organization: a laboratory, an industrial plant, a large business firm. The response time associated with cross-network communications, the communications latency, is typically $\sim$ ms, the transmission speed $\sim$ M bits/s. Wide area networks normally use interconnections which are leased from an authority such as the PTT (post telephone and telegraph). The lines are not the property of the organization(s) running the network which merely buy a service from a third party. Typical response times are 100 - 250 ms, typical transmission speeds $\sim$ K bits/s. Components of the network can be separated by $\sim 1 - 10^3$ km Local networks are clearly more tightly coupled than wide area networks.

Figure 16 illustrates the range of the degree of coupling for the three types of distributed computing systems discussed. It is very useful in accessing the suitability of different schemes for particular applications. An applicaton consisting of a set of parallel processes that need to interact or share data only every 10 to 100 s can clearly run on a loosely coupled computer network. At the other end of the spectrum algorithms that require parallel execution of arithmetic operations within single expressions require interaction almost every instruction cycle. The long response times of networks and even multiprocessors make such organizations impractical. The average time between interprocess interaction is therefore a critical time constant and provides a good incdication of what type of multiple processor architecture will be most suitable.

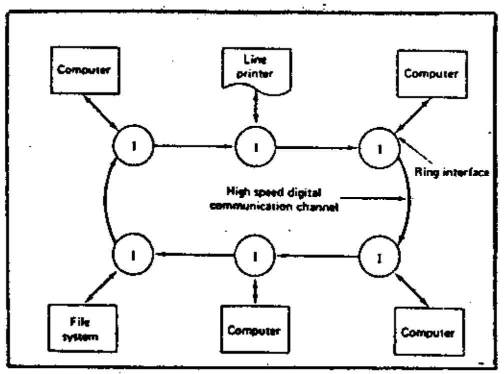


Fig 14 a local area network

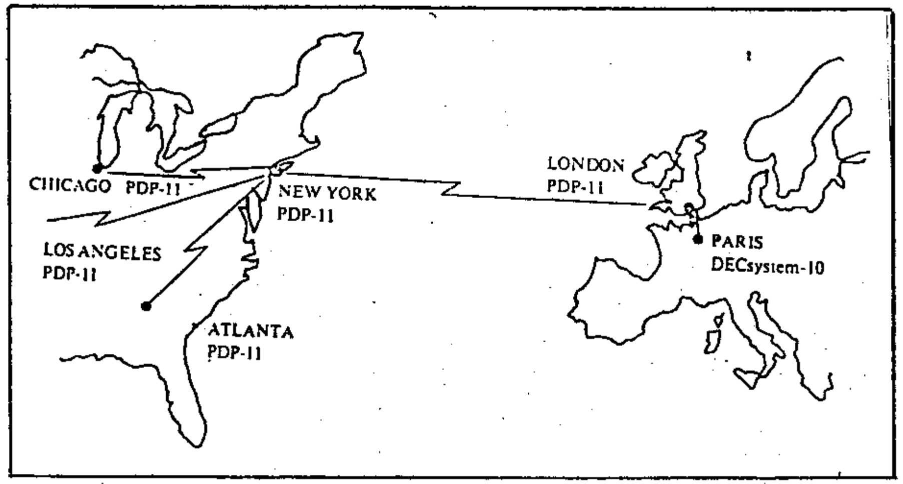


Fig 15 a long haul or wide area network

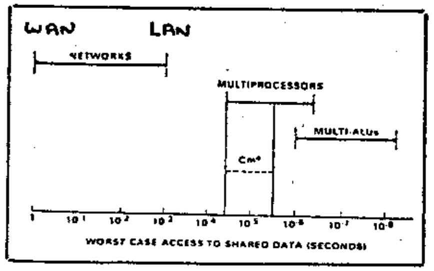


Fig 16 the degree of coupling for various types of distributed computing systems

The taxonomy of Anderson and Jensen

I now want to turn to a classification system or taxonomy introduced by Anderson and Jensen. They consider a collection of processors interconnected to perform some function. Note the term processor is used loosely here as in many cases it is synonomous with peripheral. The processors are connected into various topographical structures and exchange messages by means of paths and switches.

A message is a unit of information transferred between processes executing on processors.
A path is the media over which information is transferred
A switch is an intervening intelligent device that routes a message between sender and receiver

Based on these three concepts, a four-level model for distributed processor topographies can be defined.

LEVEL 1: Connection

The first decision is whether to connect the source and destination directly or indirectly. An indirect connection implies the intervention of one or more switches that select the message route.

LEVEL 2: Routing control

Indirect connections involve routing. This may be centralized or distributed. Either a centralized controller establishes a route for each message or a routing algorithm is executed at each of several switches.

LEVEL 3. Path structure

Paths can be either shared or dedicated. A shared path implies more than two users. It also implies contention for the path must be solved by some form of arbitration.

LEVEL 4: System architecture

This level represents specific system designs. Figures 17 and 18 summarize the model. Figure 19 shows the ten different topographies. Let's now briefly go through them all in turn. Some, but not perhaps all, you will meet again later when I come to present some practical examples of distributed computing systems.

Loops (rings)

Loop (or ring) systems use direct connections, no routing and dedicated paths. A processor transmits a message to its neighbours. The procedure continues until the message arrives at its final destination.

Complete interconnection architecture

These systems use direct connections, no routing and dedicated paths. In a typical system all processors are directly connected to every other processor.

The common or shared memory architecture

The shared memory topography uses direct connections, no routing and a shared path. It is the basis of all multiprocessor systems when it is primary memory that is being shared. However, other memory types such as discs can also be shared. This type of system is not normally classified as a multiprocessor.

Global bus

This interconnection scheme uses direct connections, no switching and a shared path. Access to the shared path must be controlled by a bus arbitrator. Messages are sent from the memory of the source to the memory of the destination via the bus.

```
TRANSFER STRATEGY: DIRECT
    PATH: DEDICATED
         TOPOLOGY: DISTRIBUTED CONTROL LOOP
         TOPOLOGY: COMPLETE INTERCONNECTION
    PATH: SHARED
         TOPOLOGY: COMMON MEMORY
         TOPOLOGY: COMMON DISTRIBUTED
                   CONTROL BUS
TRANSFER STRATEGY: INDIRECT
    ROUTING: CENTRALIZED
         PATH: DEDICATED
              TOPOLOGY: STAR
              TOPOLOGY: CENTRALLY CONTROLLED
                        LOOP
         PATH: SHARED
              TOPOLOGY: COMMON CENTRALLY
                        CONTROLLED BUS
    ROUTING: DECENTRALIZED
         PATH: DEDICATED
              TOPOLOGY: REGULAR NETWORK
              TOPOLOGY: IRREGULAR NETWORK
         PATH: SHARED
              TOPOLOGY: BUS WINDOW
```

Fig 17

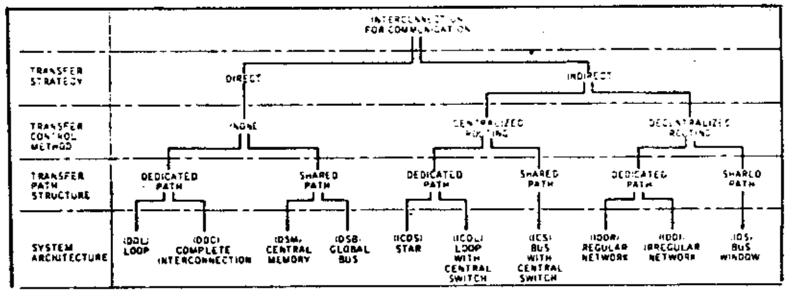
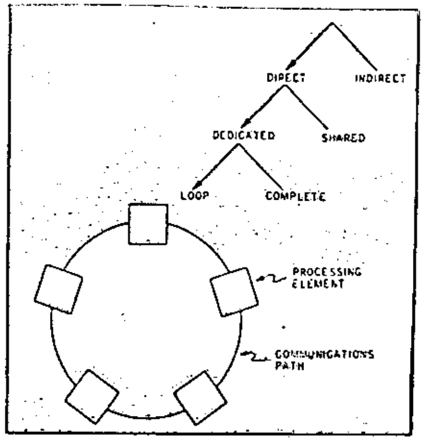


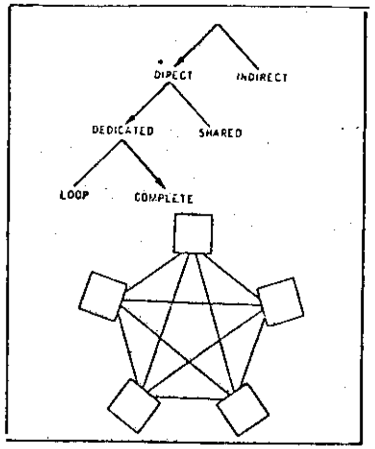
Fig 18

The Taxonomy of Andersen and Jensen

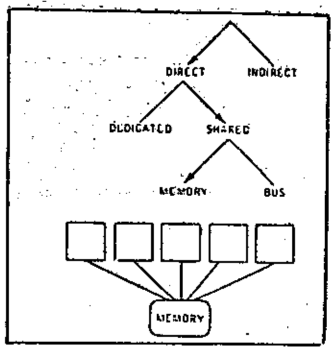
17



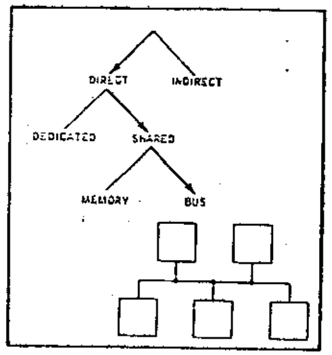
LOOP or RING



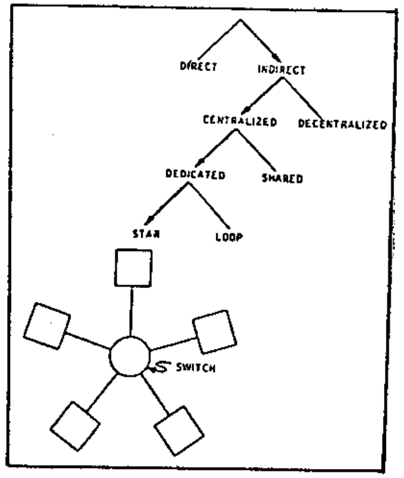
COMPLETE INTERCONNECTION



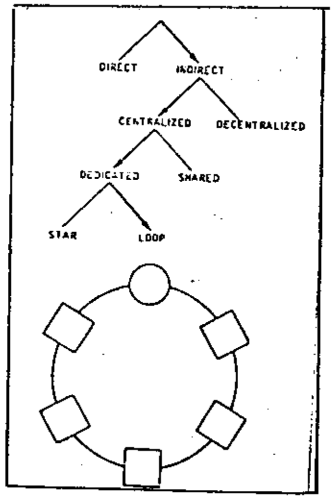
SHARED MEMORY



GLOBAL BUS



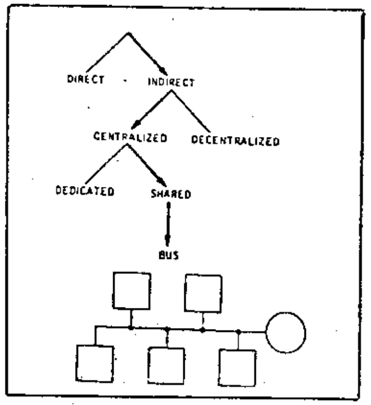
STAR



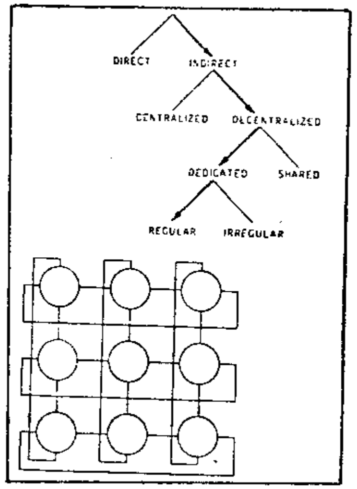
LOOP WITH CENTRAL CONTROL

18

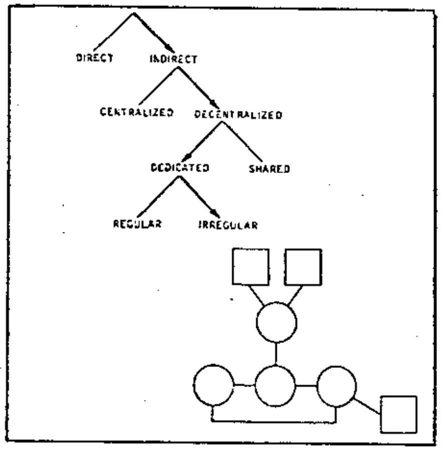
Fig 19



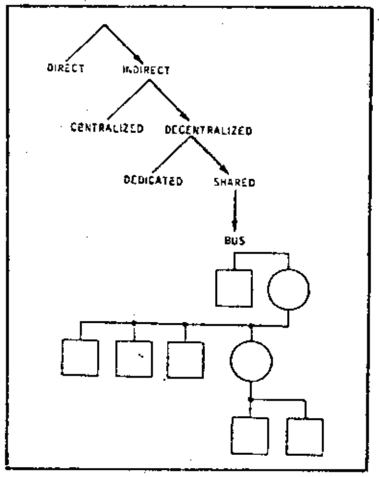
BUS WITH CENTRAL SWITCH



REGULAR NETWORK



IRREGULAR NETWORK



BUS WINDOW

Fig 19 continued

Star

The star architecture uses indirect connections, centralized routing and dedicated paths. Source and destination communicate by the source sending a message to the switch which acts as an intermediate destination. The switch determines the appropriate final destination and routes the message to it. Star topologies are the basis of many time-shared systems (see Fig. 3). However, the centre of the star is, in this case, the focal point of the distributed system (a central multiple accessed computing facility) rather than a switch.

Loop with central control

This topography like the star, uses indirect connections, centralized routing and dedicated paths. Like the star, the switch acts as an intermediate destination for messages from the source, routing the message to its final destination.

Bus with central switch

A bus with a central switch uses indirect connections, centralized routing and a shared path. A processor wishing to transmit a message obtains control of the bus and sends the message to a switch which re-routes it to its final destination.

Regular network

This type of system uses indirect connections, decentralized routing and dedicated paths. Processors are connected together in a regular pattern. A message is routed through the network from processor to processor. Each intermediate destination must determine which of its neighbours will be next to receive the message. Such configurations are not very common, except in array processors, a very specialized form of super computer.

Irregular network

Irregular networks use indirect connections, decentralized routing and dedicated paths. Routing is performed by a series of switches through which messages pass from source to final destination. The dominant application of this type of topology is in long haul networks using interconnections provided by the PTT.

Bus window

The bus window architecture uses indirect interconnection, decentralized routing and shared paths. This architecture evolved out of attempts to design systems composed of interconnected minicomputers. Typically switches are used to interconnect multiple processors on multiple buses. Addresses in the source processor address space are translated (routed) by the switch into the destination address space.

These ten different topographies form a convenient classification scheme or taxonomy. However, the taxonomy is, of course, not ideal. Hybrid schemes combining different topologies together are not considered. In addition, the distinction physically between a switch and a processor may not exist. A computer may act as a source and destination for messages but also as a message switch.

3. A SUMMING UP; WHAT I HAVE PRESENTED SO FAR; WHAT WILL COME NEXT

Let us now take stock of where we are. Historically we have seen how distributed computing came about. I have then tried to classify in a semi-formal way the different types of distributed computing systems. However, please note I have glossed over many things. In discussing the taxonomy of Anderson and Jensen I have said nothing so far about the advantages and disadvantages of the different topologies. Four points can be considered when making up a balance and choosing a system:

1) Performance bottlenecks. Which features limit data throughput and response time?

2) Modularity. How easy is it to expand the system?

3) Fault tolerance and reconfigurability. What is the effect on the overall system of a failure of one of its components? Can the system keep running, albeit in a downgraded form?

4) Interconnection complexity. What is the logical complexity of decisions necessary to route information within the system?

I have also not discussed at all the form that interconnection paths might take: whether these paths are serial or parallel, whether wires, optical fibres or radio waves. All are possible.

My approach during the rest of the course is to present you with a number of different practical examples of distributed computing, in day-to-day situations wherever possible. In doing this I will illustrate the different types of systems we have met so far in a more formal classification scheme. The emphasis will be on micro and mincomputers, not large mainframe machines. I am, also because of time limitations, not going to say anything further about the details of multiple ALU systems. Software is a vast area where again I shall have time to say very little.