A Fault-Tolerant Technique for Site Recovery in Wideband Networks

A dissertation submitted towards the partial fulfillment of the requirement for the Award of the Degree of

Master of Engineering

in

Computer Technology & Applications

Submitted by

Naveen Rana

College Roll No: 17/CTA/03
Delhi University Roll No: 9003

Under the guidance of

Dr. Goldie Gabrani

[Head of Department]

DEPARTMENT OF COMPUTER ENGINEERING

DELHI COLLEGE OF ENGINEERING

UNIVERSITY OF DELHI

JULY 2006

CERTIFICATE

This is to certify that the work that is being presented in this dissertation entitled ***"A Fault-Tolerant Technique for Site Recovery in Wideband Networks"*** submitted by **Naveen Rana** in the partial fulfillment of the requirement for the award of the degree of **Master of Engineering** in Computer Technology and Application, Delhi College of Engineering is an account of his work carried out under my guidance and supervision.

The work embodies in this dissertation has not been submitted for the award of any other degree to the best of my knowledge.

Dr. Goldie Gabrani
Head of Department
Department of Computer Engineering
Delhi College of Engineering
Delhi

ACKNOWLEDGEMENT

It is a great pleasure to have the opportunity to extent my heartiest felt gratitude to everybody who helped me throughout the course of this project.

It is distinct pleasure to express my deep sense of gratitude and indebtedness to my learned supervisor, Dr. G. Gabrani , Head Of Department (CTA), Delhi College of Engineering, for her invaluable guidance, encouragement and patient review. Her continuous inspiration only has made me complete this minor project.

I would also like to take this opportunity to present my sincere regards to my teachers viz. Dr S. K. Saxena, Mr. Rajeev Kumar and Mrs. Rajni Jindal for their support and encouragement.

I would like to express my heartiest felt regards to Faculty of Computer Engineering Department for providing facility and for their co-operation and patience.

I am thankful to my friends and classmates for their unconditional support and motivation during this project.

Naveen Rana
M.E. (Computer Technology & Applications)
College Roll No. 17/CTA/03
Delhi University Roll No : 9003

ABSTRACT

In Today's world, with several business transactions taking place over the Internet, we are interested not only in making sure that things work as intended, but also, when the inevitable failures do occur, that the damage done is minimal. As it turns out, that world conspires against us and is constructed in such a way that, generally, it is simply not possible to devise absolutely foolproof, 100% reliable software. No matter how hard we try, there is always a possibility that something can go wrong. The best we can do is to reduce the probability of failure to an "acceptable" level.

In this dissertation, we will explain basic networking concepts including different categories of networks, transmission media, switching techniques. Our primary focus was on understanding basic fault tolerance methods and how we can take back up of our system and successfully recover from failure. We have discussed some well known backup plans and recovery techniques.

In our proposed method, we have added fault-tolerant capability in our system so that it will continue to run even in case of failure and recover the lost information which was lost at the time of site crash. Our system is based on wideband networks. There are several sites and large number of data items which are initially residing at different sites. In fact, any site might possess more than one data item. Sites will communicate through message passing. Each site keeps information about all other sites in the system. In case of a site failure, the crashed site will stop receiving incoming requests and other sites in the system will continue to do their work. When site recovers from failure, it is the responsibility of other sites in the system to recover the crashed site by sending appropriate information.
Performance analysis of our proposed method is done and system is found to be reliable.

CONTENTS

1. NETWORKING CONCEPTS

1.1 Overview

A computer network is a group of computers and other related devices that are connected by some communication medium such as Ethernet, token ring, wireless, broadband etc. A network can involve permanent connections, such as cables, or temporary connections made through telephone or other communication links [1].

Networks interconnect computer systems at various different sites allowing them to exchange information such as file sharing, Internet connection and data storage. A network can be as small as a LAN (local area network) consisting of a few computers, printers, and other devices, or it can consist of many small and large computers distributed over a vast geographic area (WAN or wide area network).

Traditional telecommunication networks were characterized by specialization. This means that for every individual telecommunication service at least one network exists that transports this service. Each of these networks was specially designed for that specific service and is often not at all applicable for transporting another service. When designing the network of the future, one must take into account all possible existing and future services [3].

The traditional networks were very specialized and suffer from a large number of disadvantages:

- *Service Dependence*
  Each network is only capable of transporting one specific service.
- *Inflexibility*
  Advances in audio, video and speech coding and compression algorithms and progress in VLSI technology influence the bit rate generated by a certain service and thus change the service requirements for the network. In the future new services with unknown requirements will appear. A specialized network has great difficulties in adapting to new services requirements.
- *Inefficiency*
  The internal available resources are used inefficiently. Resources which are available in one network cannot be made available to other networks.

Society is becoming more informational [2] and visually oriented every day. Personal computing facilitates easy access, manipulation, storage, and exchange of information. These processes require

reliable transmission of data information. Communicating documents by images and the use of high resolution graphics terminals provide a more natural and informative mode of human interaction than just voice and data. Video teleconferencing enhances group interaction at a distance. High definition entertainment video improves the quality of picture at the expense of higher transmission bit-rates, which may require new transmission means other than the present overcrowded radio spectrum. A modern Telecommunications network (such as the broadband network) must provide all these different services (multi-services) to the user. It is very important that in the future single network will exist and that this network is service independent. This implies a single network capable of transporting all services, sharing all its available resources between the different services.

For a network to be effective , it should satisfy following important criteria [1]:

- *Reliable :* The network reliability is measured in terms of the time it takes a link to recover from failure , network's robustness in a catastrophe and by frequency of its failures.
- *Performance :* It can be measured by "transit time" (time required for a message to travel from one device to another) and "response time" (elapsed time between an inquiry and a response). The network's performance also depends on number of factors like type of transmission medium, software, hardware, number of users etc.
- *Security :* data on the networks must be protected from unauthorized access and viruses.

1.2 Switching Techniques

We can begin with two rough classifications[4]. If a connection (path) between the origin and the end node is established at the beginning of a session we are talking about circuit or packet (virtual circuit) switching. In case it does not, we refer to message and packet (datagram) switching. On the other hand, when considering how a message is transmitted, if the whole message is divided into pieces we have packet switching (based either on virtual circuit or datagram) but if it does not, we have circuit and message switching.
In the following paragraphs we get into the details of different switching techniques

1.2.1 Circuit switching

In circuit switching, a caller must first establish a connection to a callee before any communication is possible. During the connection establishment, resources are allocated between the caller and the callee. Generally, resources are frequency intervals in a Frequency Division Multiplexing (FDM) scheme or more recently time slots in a Time Division Multiplexing (TDM) scheme. The set of resources allocated for a connection is called a circuit. A path is a sequence of links located between nodes called *switches*.

The path taken by data between its source and destination is determined by the circuit on which it is flowing, and does not change during the lifetime of the connection. The circuit is *terminated* when the connection is closed [1].

Resources remain allocated during the full length of a communication, after a circuit is established and until the circuit is terminated and the allocated resources are freed. Resources remain allocated even if no data is flowing on a circuit, hereby wasting link capacity when a circuit does not carry as much traffic as the allocation permits. This is a major issue since frequencies (in FDM) or time slots (in TDM) are available in finite quantity on each link, and establishing a circuit consumes one of these frequencies or slots on each link of the circuit. As a result, establishing circuits for communications that carry less traffic than allocation permits can lead to resource exhaustion and network saturation, preventing further connections from being established. If no circuit can be established between a sender and a receiver because of a lack of resources, the connection is *blocked*.

A second characteristic : Time cost involved when establishing a connection. In a communication network, circuit-switched or not, nodes need to lookup in a *forwarding table* to determine on which link to send incoming data, and to actually send data from the input link to the output link. Performing a lookup in a forwarding table and sending the data on an incoming link is called *forwarding*. Building the forwarding tables is called *routing* [3]. Routing must be performed for each communication, at circuit establishment time. During circuit establishment, the set of switches and links on the path between the sender and the receiver is determined and messages are exchanged on all the links between the two end hosts of the communication in order to make the resource allocation and build the routing tables. In circuit switching, forwarding tables are hardwired or implemented using fast hardware, making data forwarding at each switch almost instantaneous. Therefore, circuit switching is well suited for long-lasting connections where the initial circuit establishment time cost is balanced by the low forwarding time cost.

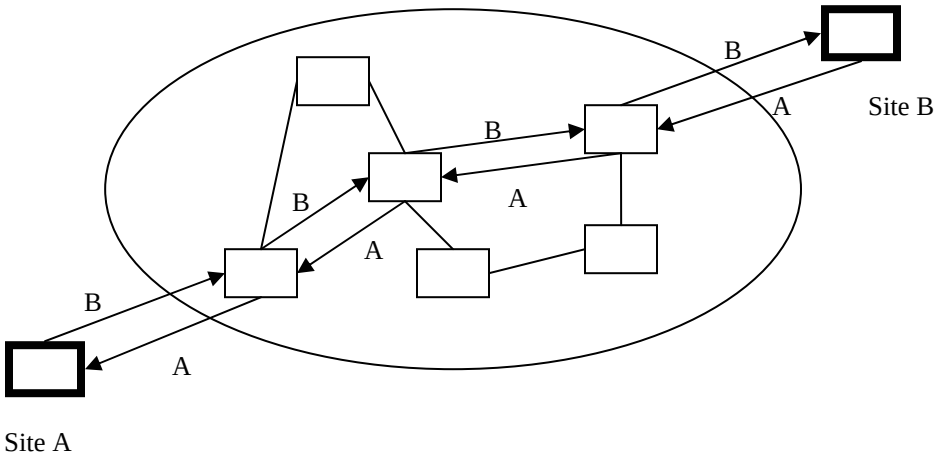


Fig 1.2.1: Circuit switching

1.2.2 Message switching

When a connection is established, the origin-node identifies the first intermediate node in the path to the end-node and sends it the whole message. After receiving and storing this message, the first intermediate node (node A) identifies the second one (node B) and, when the transmission line is not busy, the former sends the whole message (store-and-forward philosophy)[2]. This process is repeated up to the end-node. No communication release or establishment is needed.

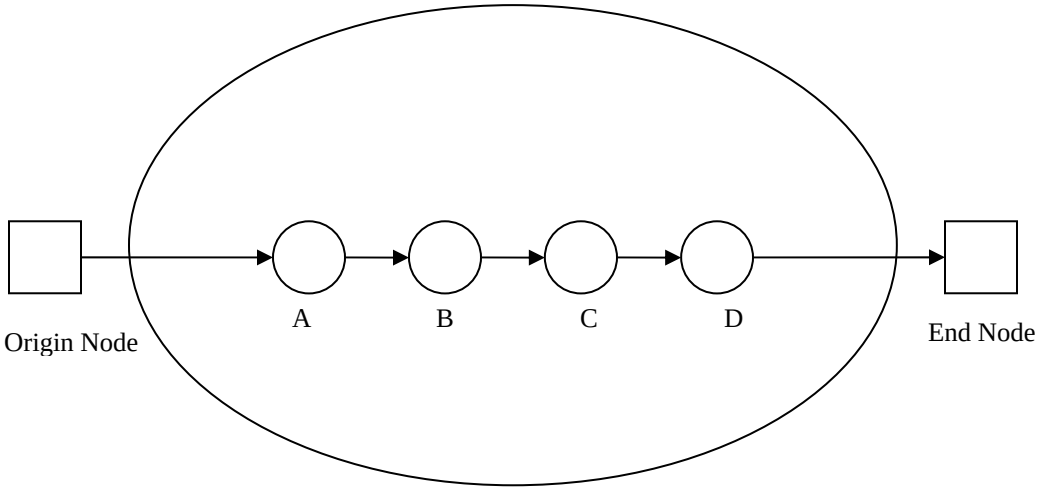


Fig 1.2.2: Message switching

1.2.3 Packet switching based on datagram

Conceived in the 1960's, [5] *packet switching* is a more recent technology than circuit switching which addresses a disadvantage of circuit switching: the need to allocate resources for a circuit, thus incurring

link capacity wastes when no data flows on a circuit. Packet switching introduces the idea of cutting data on a flow into packets which are transmitted over a network without any resource being allocated. If no data is available at the sender at some point during a communication, then no packet is transmitted over the network and no resources are wasted. Packet switching is the generic name for a set of two different techniques: datagram packet switching and virtual circuit packet switching. Here, we give an overview of datagram packet switching.

Different from circuit switching, datagram packet switching does not require establishing circuits prior to transmission of data and terminating circuits after the transmission of data. The switches, called routers, have to make a lookup in the forwarding table, called routing table, for each incoming packet. A routing table contains a mapping between the possible final destinations of packets and the outgoing link on their path to the destination. Routing tables can be very large because they are indexed by possible destinations, making lookups and routing decisions computationally expensive, and the full forwarding process relatively slow compared to circuit switching. In datagram packet switching networks, each packet must carry the address of the destination host and use the destination address to make a forwarding decision. Consequently, routers do not need to modify the destination addresses of packets when forwarding packets [6].

Since each packet is processed individually by a router, all packets sent by a host to another host are not guaranteed to use the same physical links. If the routing algorithm decides to change the routing tables of the network between the instants two packets are sent, then these packets will take different paths and can even arrive out of order. Since routers make routing decisions locally for each packet, independently of the flow to which a packet belongs. Therefore, traffic engineering techniques, which heavily rely on controlling the route of traffic, are more difficult to implement with datagram packet switching than with circuit switching.

1.2.4 Packet switching based on virtual circuit

Virtual circuit packet switching (VC-switching) is a packet switching [6] technique which merges datagram packet switching and circuit switching to extract both of their advantages. VC-switching is a variation of datagram packet switching where packets flow on so-called logical circuits for which no physical resources like frequencies or time slots are allocated. Each packet carries a circuit identifier which is local to a link and updated by each switch on the path of the packet from its source to its destination.

A virtual circuit is defined by the sequence of the mappings between a link taken by packets and the circuit identifier packets carry on this link. This sequence is set up at connection establishment time and identifiers are reclaimed during the circuit termination.

In VC-switching [7], routing is performed at circuit establishment time to keep packet forwarding fast. Other advantages of VC-switching include the traffic engineering capability of circuit switching, and the resources usage efficiency of datagram packet switching. Nevertheless, a main issue of VC-Switched networks is the behavior on a topology change. As opposed to Datagram Packet Switched networks which automatically re compute routing tables on a topology change like a link failure, in VC-switching all virtual circuits that pass through a failed link are interrupted. Hence, rerouting in VC-switching relies on traffic engineering techniques.

1.3 Categorization of Networks

1.3.1 Functional Relationship based networks

On functional relationship basis, networks can be classified into Active Networking, Client-Server and Peer-to-Peer [17].

Active networking allows packets flowing through a network to dynamically modify the operation of the network. The architecture of active network includes execution environments (that can execute active packets), a node operating system capable of supporting one or more execution environments, and active hardware, capable of routing or switching as well as executing code within active packets. In client-server model, client sends request to server and waits for reply. Server who is waiting for request to arrive sends a reply to the client when request arrives.Server normally runs on powerful machines where client runs on common PCs or workstations.

A *Peer-to-Peer* (P2P) does not have the notion of clients or servers, but only equal *peer* nodes that simultaneously function as both "clients" and "servers" to the other nodes on the network. They depends on computing power and bandwidth of the participants in the network. Such networks are useful for many purposes. Sharing content files containing audio, *video,* data and realtime data, such as telephony traffic[17].

1.3.2 Specialized functions based networks

By means of *specialized functions* there is one most common type of network known as Storage Area Network (SAN). This type of network is used to attach omputer storage devices such as disk array controllers and tape libraries to servers. SANs are common in enterprise storage. In a storage network, a

server issues a request for specific blocks, or data segments, from specific disk drives[19]. This method is known as *block storage*.

There are two variations of SANs:

- A network whose primary purpose is the transfer of data between computer systems and storage elements. A SAN consists of a communication infrastructure, which provides physical connections, and a management layer, which organizes the connections, storage elements, and computer systems so that data transfer is secure and robust. The term SAN is usually (but not necessarily) identified with block I/O services rather than file access services
- A storage system consisting of storage elements, storage devices, computer systems, and/or appliances, plus all control software, communicating over a network

1.3.3 Scale based networks

On the basis of scale, networks can be divided into LAN, MAN and WAN. Local Area Network (LAN) covers a small local area, like a home, office, or small group of buildings such as a college[20]. LANs are often characterisized by : a) much higher data rates, b) smaller geographic range - at most a few kilometers and c) they do not involve leased telecommunication lines. "LAN" usually does not refer to data running over local analog telephone lines, as on a private branch exchange (PBX). E.g. ethernet, token ring etc. Metropolitan Area Networks (MAN) usually covers a campus or a city. They typically use wireless infrastructure or optical fiber connections to link their sites. It can have multiple LANs. Wide Area Networks (WAN) used to cover a wide geographical area, involving a large array of computers providing long distance transmission of data, voice, image and video information. They are most often built using leased lines. They can also be built using less costly circuit switching or packet switching methods. Network Protocols including ATM and Frame relay are often used by service providers to deliver the links that are used in WANs. X.25 was an important early WAN protocol.

1.3.4 Protocol based networks

Computer networks may be implemented using a variety of protocol stack architectures
A **protocol stack** is a particular software implementation of a computer networking protocol suite[17]. In other words, the suite is the definition of the protocols, and the stack is the software implementation of them.

There are so many networks based on network protocols but some of them which will be discussed are ATM, IP networks, Wideband, Frame relay.

1.3.4.1 ATM (Asynchronous Transfer Mode)

In traditional networks, the various services were carried via separate networks – voice on the telephone network, data on computer networks or local area networks (LANs), video teleconferencing on private corporate networks, and television on broadcast radio or cable networks[21].

These networks are largely engineered for a specific application and are not suited for other applications. For example, the traditional telephone network is too noisy and inefficient for burst data communication. On the other hand, data networks which store and forward messages using computers have very limited connectivity, usually do not have sufficient bandwidth for digitized voice and video signals, and suffer from unacceptable delays for the real-time signals. Television networks using the radio or the cable medium are largely broadcast networks with minimum switching facilities.

Benefits of a single network for multiple services:

- *Flexible and future safe*
  A network capable of transporting all types of services that will be able to adapt itself to new needs
- *Efficient use of its available resources*
  All available resources can be shared between all services, such that an optimal statistical sharing of the resources can be obtained.
- *Less expensive*
  Since only one network needs to be designed, manufactured and maintained the overall costs of the design, manufacturing, operations and maintenance will be lower.

ATM is an example of such network that can provide all these services. It is considered a packet oriented transfer mode based on asynchronous time division multiplexing and the use of fixed length cells[17].

ATM is a cell relay (cell relay refers to a method of statistically multiplexing fixed-length packets, i.e. cells) network protocol which encodes data traffic into small fixed-sized (53 byte; 48 bytes of data and 5 bytes of header information) cells instead of variable sized *packets* (*frames*) as in packet-switched networks (Internet Protocol or Ethernet). It is a connection-oriented technology, in which a connection is established between the two endpoints before the actual data exchange begins.

To guarantee a fast processing in the network, the ATM header has very limited function. Its main function is the identification of the virtual connection by an identifier which is selected at call set up and guarantees a proper routing of each packet. In addition it allows an easy multiplexing of different virtual connections over a single link. The information field length is relatively small, in order to reduce the internal buffers in the switching node, and to limit the queuing delays in those buffers - small buffers guarantee a small delay and a small delay jitter as required in real time systems. The information field of ATM cells is carried transparently through the network. No processing is performed on it inside the network. All services (Voice, video, Data) can be transported via ATM, including connectionless services.

Routing in ATM Network

ATM is connection oriented. Before information is transferred from the terminal to the network, a logical/virtual connection is set. The header values are assigned to each section of a connection for the complete duration of the connection, and translated when switched from one section to another[21]. Signaling and user information are carried on separate virtual channels. Two sorts of connections are possible: Virtual Channel Connections (VCC) and Virtual Path Connections (VPC). When switching or multiplexing on cells is to be performed, it must first be done on VPC, then on the VCC.

Virtual Channels

This function is performed by a header sub field - VCI. Since the ATM network is connection oriented each connection is characterized by a VCI which is assigned at call set up. A VCI has only a local significance on the link between ATM node and will be translated in the ATM nodes. When the connection is released, the VCI values on the involved links will be released and can be reused by other connections.

An advantage of this VCI principle is the use of multiple VCI values for multi component services. For instance video telephony can be composed of 3 components: voice, video and data. Each of which will be transported over a separate VCI. This allows the network to add or remove components during the connection. For instance, the video telephony service can start with voice only and the video can be added later.

Virtual Path

The network has to support semi-permanent connections, which have to transport a large number of simultaneous connections[21].

As ATM is connection oriented, connections are established either semi-permanently, or for the duration of a call, in case of switched services. This establishment includes the allocation of a VCI (Virtual Channel Identifier) and/or VPI (Virtual Path Identifier), and also the allocation of the required resources on the user access and inside the network. These resources are expressed in terms of throughput and Quality of Service. They may be negotiated between user and network for switched connection during the call set up phase.

In ATM there is no physical limitation on the user access rate to the physical transmission medium, apart from the physical cell rate on the medium itself. Multiplexing equipment will do its utmost to avoid cell loss to offer the highest possible throughput whatever the user chooses to send. As virtual connections share physical resources, transmission media and buffer space, unforeseen excessive occupation of resources by one user may impair traffic for other users. In principle, no flow control will be applied to information streams at the ATM layer of the network. In some cases it will be necessary to be able to control the flow of traffic on ATM connections from a terminal to the network. In order to cope with this a GFC (general flow control) mechanism may be used. This function is supported by a specific field in the ATM cell header[17].

Two sets of procedure are used: Uncontrolled Transmission - for the use of point to point configuration. Controlled Transmission - can be used in both point to point and shared medium configuration. Another principle is no error protection on link by link basis. If a link in the connection, either the user to network link or the internal links between the network nodes, introduces an error during the transmission or is temporarily overloaded thereby causing the loss of packets, no special action will be taken on that link to correct this error (= no requesting for retransmission).This error protection can be omitted since the links in the network have a very high quality.

1.3.4.2 IP Networks

Internet Protocol (IP) is a data-oriented protocol [15] used for communicating data across a packet-switched network. IP provides the service of communicable unique global addressing amongst computers. It can be used over a heterogenous network (i.e., a network connecting two computers can be any mix of ethernet, ATM, Token ring, etc.).

IP provides an *unreliable* service i.e. network gives no surity about packet. Following things can happen to data [2]:

- Data can be corrupted
- Duplicate data can arrive
- Data can be lost or discarded
- Packets may reach in different order than they were sent.

The only thing IP does for reliable service is ensure the IP packet's header is error-free through the use of a checksum.

These Networks that allow integrated data, voice and video services over the Internet. The Internet Protocol suite brings together all transmission layer protocols into a single, standardized protocol architecture, which can be utilized by applications for different communication purposes. As a direct result, any application that supports TCP/IP will also be able to communicate over any IP-based network.

This standardized architecture has come as a revolution in network communication. An ever-increasing number of applications that transfer text, sound, live pictures and more utilize IP-based architecture.

1.3.4.3 Wideband Networks

Generally, a wideband transmission is more commonly known as a broadband transmission. When referring to a wireless transmission, a wideband signal refers to a signal capable of being distributed over a wide area [18].

It refers to telecommunications that provide a variety of channels of data over a single communication medium (wire). Today, there are a wide variety of broadband technologies available in most areas, two of the more commonly found and used technologies are cable and DSL broadband (Digital Subscriber Line, DSL is a method for home users and small businesses to have high speed access to the Internet over standard copper lines. Capable of receiving up to 6.1 megabits per second,).

Wideband network is medium-capacity communications circuit/path. It usually implies a speed from 64Kbps to 1.544Mbps. Some people say that wideband and broadband differ in the sense that a wideband network is capable of carrying less information than a broadband network, but more than a narrowband network. It is a transmission medium with greater bandwidth capacity than standard voice lines, but less capacity than broadband channels.

Services given by wideband network includes video teleconferencing, file transferring and video telephony.

1.3.4.4 Frame Relay

It is a high-speed packet switching protocol popular in networks, including WANs, LANs, and LAN-to-LAN connections across vast distances. Network providers commonly implement frame relay for voice and data as an encapsulation technique, used *between* local area networks (LANs) *over* a wide area network (WAN). Each end-user gets a private line (or leased line) to a frame-relay node. The frame-relay network handles the transmission over a frequently-changing path transparent to all end-users.
It can transmit bursts of data over network [16].

The designers of frame relay aimed at a telecommunication service for cost-efficient data transmission for intermittent traffic between local area networks (LANs) and between end-points in a wide area network (WAN). Frame relay puts data in variable-size units called "frames" and leaves any necessary error-correction (such as re-transmission of data) up to the end-points. This speeds up overall data transmission. For most services, the network provides a permanent virtual circuit (PVC), which means that the customer sees a continuous, dedicated connection without having to pay for a full-time leased line, while the service-provider figures out the route each frame travels to its destination and can charge based on usage[1].

When a frame relay network detects an error in a frame, it simply "drops" that frame. The end points have the responsibility for detecting and retransmitting dropped frames.

1.4 Transmission of Data

In computer systems, data transmission means sending a stream of bits or bytes from one location to another using any technologies, such as copper wire, optical fibre, laser, radio, infra-red light or Bluetooth [17].

Transmission of data can be serial and parallel. In serial, bits are sent over a single wire individually. Only one bit is sent at a time. In parallel, Multiple wires are used and transmit bits simultaneously and is much faster than serial transmission as one byte can be sent rather than one bit. We can measure the speed of data transmission by the Baud rate and bit rate. *Bit rate* is the rate of data transmission serially, measured in bits per second. The baud and bit rates are usually the same at lower speeds, however the baud rate can be lower but still giving a high bit rate. The *Baud rate* is the number of bits that can be encoded into a single signal per unit of time. It is the measure of symbol rate i.e. number of different symbolic changes (signalling event) made to the transmission medium per second in a digitally modulated signal.

For example, difference between baud (or signalling rate) and the data rate (or bit rate) is a man using a single semaphore flag. He can move his arm to a new position once each second, so his signalling rate (baud) is 1 symbol per second. However, the flag can be held in one of eight distinct positions: Straight up, 45 degrees left, 90 degrees left, 135 degrees left, straight down (which is the rest state, where he is sending no signal), 135 degrees right, 90 degrees right, and 45 degrees right. This means each signal carries three bits of information, as it takes 3 binary digits to encode 8 distinct states – so the data rate is 3 bits per second.

There are two types of data transmission:

- Wired transmission
- Wireless transmission

1.4.1 Wired transmission

Connection between two or more locations is made by using copper wire, optical fibre etc. Wires are used to bear mechanical loads and to carry electricity and telecommunications signals. Technologies used for this type of transmission are Public switched telephone network, Modems and dialup, Dedicated lines – leased lines, Time-division multiplexing, Packet switching, Frame relay, Ethernet, RS-232 ,Optical fiber transmission [17] .

The **PSTN** is the concentration of the world's public circuit-switched telephone networks, in much the same way that the Internet is the concentration of the world's public IP-based packet-switched networks.

A **dedicated line** is a communications cable dedicated to a specific application, in contrast with a shared resource such as the telephone network or the Internet. Normally, such services may not be provided by a single, discrete, end-to-end cable, but they do provide guarantees of constant bandwidth availability.

Optical fibers transmits signals in the form of light. They are constructed from special kinds of glass and plastic, and they are designed so that a beam of light introduced at one end will remain within the fiber, reflecting off the inner surfaces as it travels down the length of the fiber. Optical fibers are inexpensive, compact, and lightweight and are often packaged many hundred to a single cable

1.4.2 Wireless transmission

Wireless transmission is a characteristic of communications that take place without the use of interconnecting wires or cables, such as by radio, microwave, or infrared [17].

Wireless Application Protocol, WAP, a standard for providing Internet access and other data-based services, such as e-mail, electronic transactions, news, and weather reports, over wireless networks. The WAP is designed to provide such services to digital mobile telephones and other wireless terminals. Just as the TCP/IP standards make it possible for many different kinds of computer equipment to communicate through the Internet, the WAP specification is intended to work across different types of wireless network.

Some examples of wireless technologies are : Bluetooth, IEEE 802.11.

1.5 Dissertation Organization

The organization of this dissertation is as follows.

Chapter 1: Explains the basic concept of networks, types of networks, different types of transmission media and switching techniques in detail.

Chapter 2: Explains the concepts of Fault Tolerance, its issues, its role in software and hardware systems.

Chapter 3: Explains the data storage and recovery techniques in networks. It also highlights about RAID architecture, different types of network storage models and backup procedures.

Chapter 4: It describes the proposed algorithm in detail with performance analysis and simulation results.

Chapter 5: Conclusion and future work discussed.

References

2. FAULT TOLERANCE

With business-to-business transactions taking place over the Internet, we are interested not only in making sure that things work as intended, but also, when the inevitable failures do occur, that the damage is minimal [10].

As it turns out, that world conspires against us and is constructed in such a way that, generally, it is simply not possible to devise absolutely foolproof, 100% reliable software. No matter how hard we try, there is always a possibility that something can go wrong. The best we can do is to reduce the probability of failure to an "acceptable" level. Unfortunately, the more we strive to reduce this probability, the higher the cost.

2.1 Fault-Tolerant Computing Concepts

There is much confusion around the terminology used with fault tolerance. For example, the terms "reliability" and "availability" are often used interchangeably, but do they always mean the same thing? What about "faults" and "errors"?[12]

Fault tolerance is the ability of a system to perform its function correctly even in the presence of internal faults. The purpose of fault tolerance is to increase the dependability of a system. A complementary but separate approach to increasing dependability is *fault prevention*. This consists of techniques, such as inspection, whose intent is to eliminate the circumstances by which faults arise.

Implicit in the definition of fault tolerance is the assumption that there is a specification of what constitutes correct behavior. A *failure* occurs when an actual running system deviates from this specified behavior. The cause of a failure is called an *error*. An error represents an invalid system state, one that is not allowed by the system behavior specification. The error itself is the result of a defect in the system or *fault*. In other words, a fault is the root cause of a failure. That means that an error is merely the symptom of a fault. A fault may not necessarily result in an error, but the same fault may result in multiple errors. Similarly, a single error may lead to multiple failures.

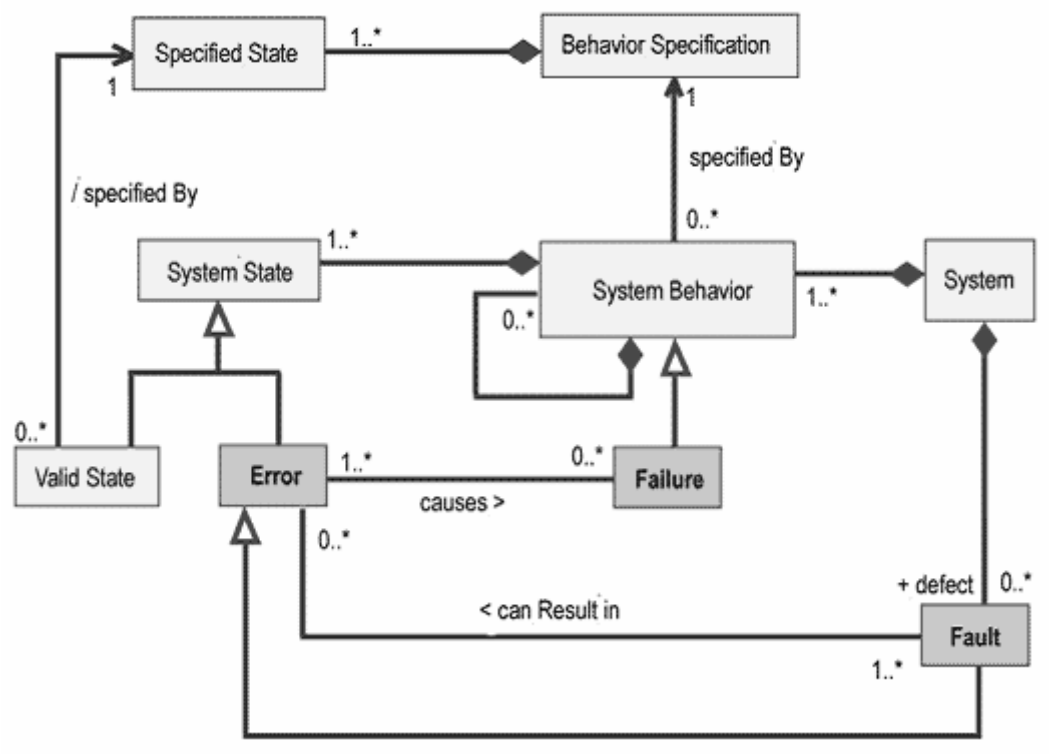


Fig 2.1 (a): Failures, Error and Fault

For example, in a software system, an incorrectly written instruction in a program may decrement an internal variable instead of incrementing it. Clearly, if this statement is executed, it will result in the incorrect value being written [12]. If other program statements then use this value, the whole system will deviate from its desired behavior. In this case, the erroneous statement is the fault, the invalid value is the error, and the failure is the behavior that results from the error. Note that if the variable is never read after being written, no failure will occur. Or, if the invalid statement is never executed, the fault will not lead to an error. Thus, the mere presence of errors or faults does not necessarily imply system failure. As this example illustrates, the designation of what constitutes a fault (the underlying cause of a failure) is relative in the sense that it is simply a point beyond which we do not choose to delve further. After all, the incorrect statement itself is really an error that arose in the process of writing the software, and so on.

At the heart of all fault tolerance techniques is some form of *masking redundancy*. This means that components that are prone to defects are replicated in such a way that if a component fails, one or more of the non-failed replicas will continue to provide service with no appreciable disruption.

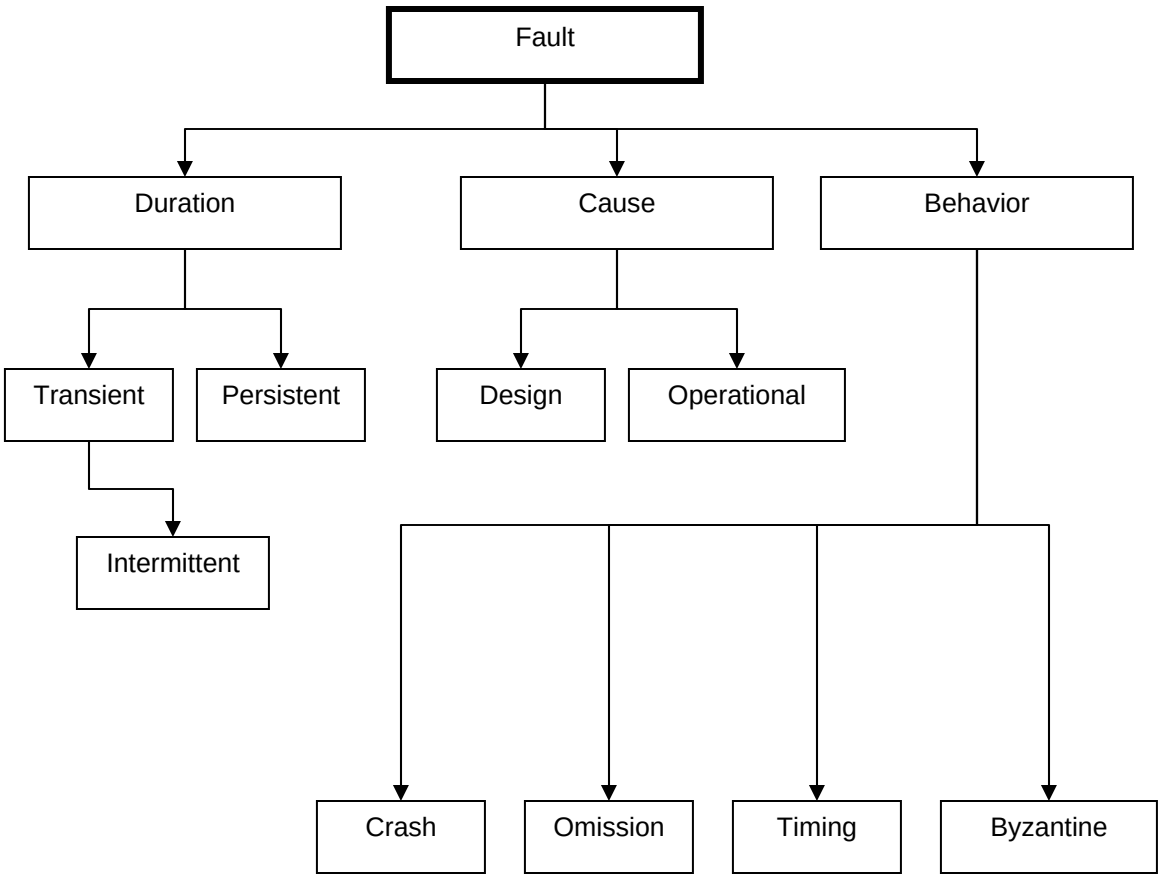


Fig 2.1 (b): classification of Faults

Based on *duration*, faults can be classified as *transient* or *permanent*. A transient fault will eventually disappear without any apparent intervention [9], whereas a permanent one will remain unless it is removed by some external agency. While it may seem that permanent faults are more severe but they are much easier to diagnose and handle. A particularly problematic type of transient fault is the *intermittent* fault that recurs, often unpredictably.

A different way to classify faults is by their underlying *cause*. *Design faults* are the result of design failures. While it may appear that in a carefully designed system all such faults should be eliminated through fault prevention, this is usually not realistic in practice. For this reason, many fault-tolerant systems are built with the assumption that design faults are inevitable, and some mechanisms need to be put in place to protect the system against them. *Operational faults*, on the other hand, are faults that occur during the lifetime of the system and are invariably due to physical causes, such as processor failures or disk crashes.

Finally, based on how a failed component behaves once it has failed, faults can be classified into the following categories[9]:

- **Crash faults** -- the component either completely stops operating or never returns to a valid state.
- **Omission faults** -- the component completely fails to perform its service.
- **Timing faults** -- the component does not complete its service on time.
- **Byzantine faults** -- these are faults of an arbitrary nature.

Failure Characteristics

Hardware Failures

Hardware failures are typically characterized by a bath tub curve. An example curve is shown below. The chance of a hardware failure is high during the initial life of the module. The failure rate during the rated useful life of the product is fairly low. Once the end of the life is reached, failure rate of modules increases again[12]

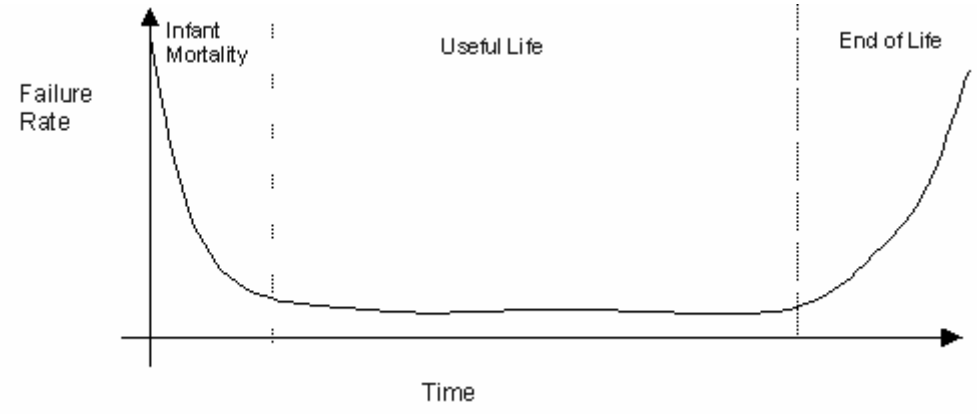


Fig 2.1 (c): Failure characteristics

Hardware failures during a products life can be attributed to the following causes:

- *Design failures :* This class of failures take place due to inherent design flaws in the system. In a well designed system this class of failures should make a very small contribution to the total number of failures.
- *Infant Mortality:* This class of failures cause newly manufactured hardware to fail. This type of failures can be attributed to manufacturing problems like poor soldering, leaking capacitor etc. These failures should not be present in systems leaving the factory as these faults will show up in factory system burn in tests.

- *Random Failures:* Random failures can occur during the entire life of a hardware module. These failures can lead to system failures. Redundancy is provided to recover from this class of failures.
- *Wear Out:* Once a hardware module has reached the end of its useful life, degradation of component characteristics will cause hardware modules to fail. This type of faults can be weeded out by preventive maintenance and routing of hardware.

Software Failures

Software failures can be characterized by keeping track of "*software defect density"* in the system. This number can be obtained by keeping track of historical software defect history. Defect density will depend on the following factors[12]:

- Software process used to develop the design and code (use of peer level design/code reviews, unit testing)
- Complexity of the software
- Size of the software
- Experience of the team developing the software
- Percentage of code reused from a previous stable project
- Rigor and depth of testing before product is shipped.

Defect density is typically measured in number of defects per thousand lines of code (defects/KLOC).

2.2 GENERAL FAULT TOLERANCE PROCEDURE

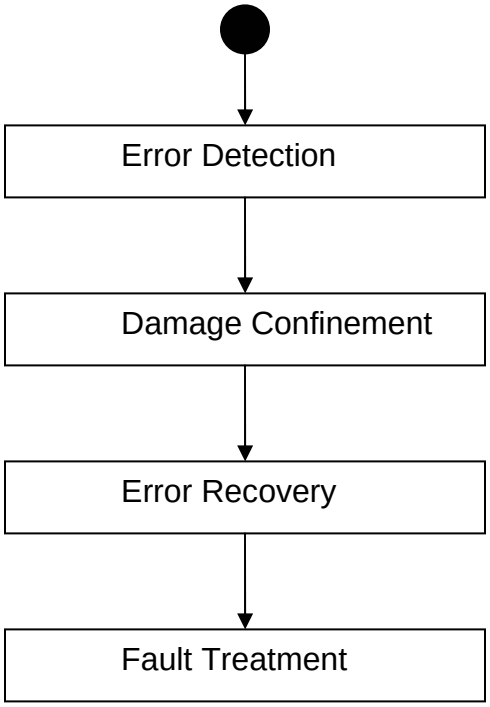


Fig 2.2: *General Fault Tolerance Procedure*

Error detection is the process of identifying that the system is in an invalid state. This means that some component in the system has failed. To ensure that the effects of the error are limited, it is necessary to isolate the failed component so that its effects are not propagated further. This is known as *damage confinement.* In the error recovery phase, the error and more importantly its effects, are removed by restoring the system to a valid state. Finally, in *fault treatment*, we go after the fault that caused the error so that it can be isolated [12].

2.2.1 Error Detection

The most common techniques for error detection are:

2.2.1.1 Replication checks : In this case, multiple replicas of a component perform the same service simultaneously. The outputs of the replicas are compared, and any discrepancy is an indication of an error in one or more components. A particular form of this that is often used in hardware is called *triple-modular redundancy* (TMR), in which the output of three independent components is compared, and the output of the majority of the components is actually passed on. In software, this can be achieved by

providing multiple independently developed realizations of the same component [9]. This is called *N-version programming*.

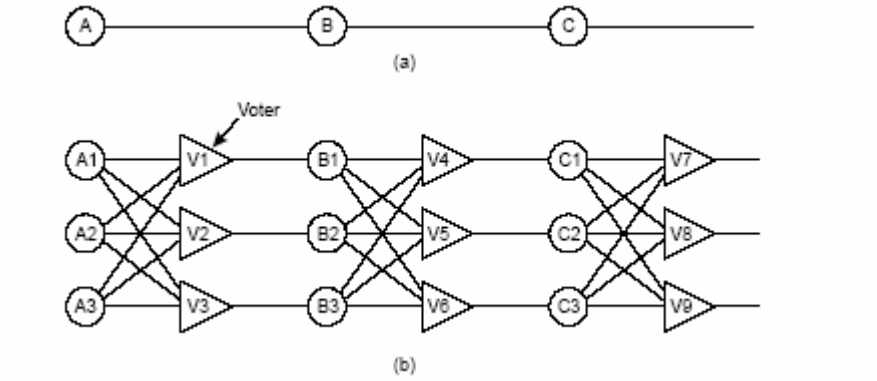


Fig 2.2.1.1 : TMR

(a) signals pass through devices A, B, and C, in sequence. If one of them is faulty, the final result will probably be incorrect

(b) each device is replicated three times. Following each stage in the circuit is a triplicated voter. Each voter is a circuit that has three inputs and one output. If two or three of the inputs are the same, the output is equal to that input. If all three inputs are different, the output is undefined.

Suppose that element A_2 fails. Each of the voters, V_1, V_2, and V_3 gets two good (identical) inputs and one rogue input, and each of them outputs the correct value to the second stage. In essence, the effect of A_2 failing is completely masked, so that the inputs to B_1, B_2, and B_3 are exactly the same as they would have been had no fault occurred[9]. Now consider what happens if B_3 and C_1 are also faulty, in addition to A_2. These effects are also masked, so the three final outputs are still correct

2.2.1.2 Timing checks: This is used for detecting timing faults. Typically a timer is started, set to expire at a point at which a given service is expected to be complete. If the service terminates successfully before the timer expires, the timer is cancelled. However, if the timer times out, then a timing error has occurred. The problem with timers is in cases where there is variation in the execution of a function. In such cases, it is dangerous to set the timer too tightly, since it may indicate false positives. However, setting it too loosely would delay the detection of the error, allowing the effects to be propagated much more widely.

2.2.1.3 Run-time constraints checking: This involves detecting that certain constraints, such as boundary values of variables not being exceeded, are checked at run time. The problem is that such checks introduce both code and performance overhead. A particular form is *robust data structures*, which have built-in redundancy (e.g., a checksum). Every time these data structures are modified, the redundancy checks are performed to detect any inconsistencies.

2.2.1.4 Diagnostic checks: These are typically background audits that determine whether a component is functioning correctly. In many cases, the diagnostic consists of driving a component with a known input for which the correct output is also known.

2.2.2 Damage Confinement

This consists of first determining the extent to which an error has spread (because time may have passed between the time the error occurred and the time it is detected)[12]. It requires understanding the flow of information in a system and following it (starting from a known failed component). Each component along such flows is checked for errors, and the boundary is determined. Once this boundary is known, that part of the system is isolated until it can be fixed.

To assist with damage confinement, special "firewalls" are often constructed between components. This implies a loose coupling between components so that components outside of the firewall can be easily de-coupled from the faulty components inside and re-coupled to an alternative, error-free redundant set. The cost of this is system overhead, both in performance (communication through a firewall is typically slower) and memory.

2.2.3 Error Recovery

To recover from an error, the system needs to be restored to a valid state. There are two general approaches to achieving this [12].

In *backward error recovery*, the system is restored to a previous known valid state. This often requires *check pointing* the system state and, once an error is detected, rolling back the system state to the last check pointed state. Clearly, this can be a very expensive capability. Not only is it necessary to keep copies of previous states, but also it is necessary to stop the operation of the system during check pointing to ensure that the state that is stored is consistent.

Forward error recovery involves driving the system from the erroneous state to a new valid state. It may be difficult to do unless the fault that caused the error is known precisely and is well isolated, so that it does not keep interfering. Because it is tricky, forward error recovery is not used often in practice.

2.2.4 Fault Treatment

In this phase, the fault is first isolated and then repaired. The repair procedure depends on the type of fault. Permanent faults require that the failed component be replaced by a non-failed component. This requires a *standby* component. The standby component has to be integrated into the system, which means that its state has to be synchronized with the state of the rest of the system. There are three general types of standby schemes [12]:

2.2.4.1 Cold standby: This means that the standby component is not operational, so that its state needs to be changed fully when the cutover occurs. This may be a very expensive and lengthy operation.

For instance, a large database may have to be fully reconstructed (e.g., using a log of transactions) on a standby disc. The advantage of cold standby schemes is that they do not introduce overhead during the normal operation of the system. However, the cost is paid in fault recovery time.

2.2.4.2 Warm standby: In this case, the standby component is used to keep the last checkpoint of the operational component that it is backing up. When the principal component fails, the backward error recovery can be relatively short. The cost of warm standby schemes is the cost of backward recovery discussed earlier (mainly high overhead).

2.2.4.3 Hot standby: In this approach, the standby component is fully active and duplicating the function of the primary component. Thus, if an error occurs, recovery can be practically instantaneous. The problem with this scheme is that it is difficult to keep two components in lock step.

In contrast to warm standby schemes, in which synchronization is only performed during checkpoints, in this case it has to be done on a constant basis. Invariably, this requires communications between the primary and the standby, so that the overhead of these schemes is often higher than the overhead for warm standby.

2.3 DEPENDABILITY

Dependability means that our system can be trusted to perform the service for which it has been designed. Dependability can be decomposed into specific aspects[9].

- *Reliability* characterizes the ability of a system to perform its service correctly when asked to do so.
- *Availability* means that the system is available to perform this service when it is asked to do so.
- *Safety* is a characteristic that quantifies the ability to avoid catastrophic failures that might involve risk to human life or excessive costs.
- *Security* is the ability of a system to prevent unauthorized access.

Technically, "*Reliability"* is defined as the probability that a system will perform correctly up to a given point in time. A common measure of reliability, therefore, is the *mean time between failures (MTBF).* Mean Time Between Failures (MTBF) is the average time between failure of hardware modules. It is the average time a manufacturer estimates before a failure occurs in a hardware module. MTBF for hardware modules can be obtained from the vendor for off-the-shelf hardware modules. MTBF for in-house developed hardware modules is calculated by the hardware team developing the board. MTBF for software can be determined by simply multiplying the defect rate with KLOCs executed per second. FITS is a more intuitive way of representing MTBF. FITS is nothing but the total number of failures of the module in a billion hours (i.e. 1000,000,000 hours)[12].

"Availability" is defined as the probability that a system is operational at a given point in time. For a given system, this characteristic is strongly dependent on the time it takes to restore it to service once a failure occurs. A common way of characterizing this is *mean time to repair (MTTR).* Mean Time To Repair, is the time taken to repair a failed hardware module. In an operational system, repair generally means replacing the hardware module. Thus hardware MTTR could be viewed as mean time to replace a failed hardware module. It should be a goal of system designers to allow for a high MTTR value and still achieve the system reliability goals

The two measures for reliability (MTBF) and availability (MTTR) can be used to show the relationship between these two important measures. It is important to distinguish these two technical terms, since they are often used interchangeably in everyday communications. This can lead to confusion. The availability of a system can be calculated from these two measures according to the formula[12]:

Availability = (MTBF) / (MTTR + MTBF)

For systems that never fail, availability is equal to reliability.

2.4 ISSUES IN FAULT TOLERANT SYSTEMS

From a fault-tolerance perspective, distributed systems have a major advantage: They can easily be made redundant, which is at the core of all fault-tolerance techniques. Unfortunately, distribution also means that the imperfect and fault-prone physical world cannot be ignored, so that as much as they help in supporting fault-tolerance, distributed systems may also be the source of many failures. In this section we briefly review these problems[9].

2.4.1 Processing Site Failures

The fact that the processing sites of a distributed system are independent of each other means that they are independent points of failure. While this is an advantage from the viewpoint of the user of the system, it presents a complex problem for developers. In a centralized system, the failure of a processing site implies the failure of all the software as well. In contrast, in a fault-tolerant distributed system, a processing site failure means that the software on the remaining sites needs to detect and handle that failure in some way. This may involve redistributing the functionality from the failed site to other, operational, sites, or it may mean switching to some emergency mode of operation.

2.4.2 Communication Media Failures

Another kind of failure that is inherent in most distributed systems comes from the communication medium. The most obvious, of course, is a permanent hard failure of the entire medium, which makes communication between processing sites impossible. In the most severe cases, this type of failure can lead to partitioning of the system into multiple parts that are completely isolated from each other. The danger here is that the different parts will undertake activities that conflict with each other[22].

A different type of media failure is an *intermittent failure*. These are failures whereby messages traveling through a communication medium are lost, reordered, or duplicated. Note that these are not always due to hardware failures. For example, a message may be lost because the system may have temporarily run out of memory for buffering it. Message reordering may occur due to successive messages taking different paths through the communication medium. If the delays incurred on these paths are different, they may overtake each other. Duplication can occur in a number of ways. For

instance, it may result from a retransmission due to an erroneous conclusion that the original message was lost in transit.

One of the central problems with unreliable communication media is that it is not always possible to positively ascertain that a message that was sent has actually been received by the intended remote destination.

A common technique for dealing with this is to use some type of positive acknowledgement protocol. In such protocols, the receiver notifies the sender when it receives a message. Of course, there is the possibility that the acknowledgement message itself will be lost, so that such protocols are merely an optimization and not a solution [22].

The most common technique for detecting lost messages is based on time-outs. If we do not get a positive acknowledgement within some reasonable time interval that our message was received, we conclude that it was dropped somewhere along the way. The difficulty of this approach is to distinguish situations in which a message (or its acknowledgement) is simply slow from those in which a message has actually been lost. If we make the time-out interval too short, then we risk duplicating messages and also reordering in some cases. If we make the interval too long, then the system becomes unresponsive.

2.4.3 Transmission Delays

While transmission delays are not necessarily failures, they can certainly lead to failures. A delay can be misconstrued as a message loss [22].

There are two different types of problems caused by message delays. One type results from *variable delays (jitter)*. That is, the time it takes for a message to reach its destination may vary significantly. The delays depend on a number of factors, such as the route taken through the communication medium, congestion in the medium, congestion at the processing sites (e.g., a busy receiver), intermittent hardware failures, etc. If the transmission delay is constant, then we can much more easily assess when a message has been lost. For this reason, some communication networks are designed as synchronous networks, so that delay values are fixed and known in advance.

However, even if the transmission delay is constant, there is still the problem of out-of-date information. Since messages are used to convey information about state changes between components of the distributed system, if the delays experienced are greater than the time required to change from one state

to the next, the information in these messages will be out of date. This can have major repercussions that can lead to unstable systems. Just imagine trying to drive a car if visual input to the driver were delayed by several seconds.

Transmission delays also lead to a complex situation that we will refer to as the "R*elativistic effect"*[12]. This is a consequence of the fact that transmission delays between different processing sites in a distributed system may be different. As a result, different sites may see the same set of messages but in a different order.

Distributed sites `NotifierP` and `NotifierQ` each send out a notification about an event to the two clients (`ClientA` and `ClientB`). Due to the different routes taken by the individual messages and the different delays along those routes, `ClientB` sees one sequence (`event1` followed by `event2`), whereas `ClientA` sees a different one (`event2-event1`).

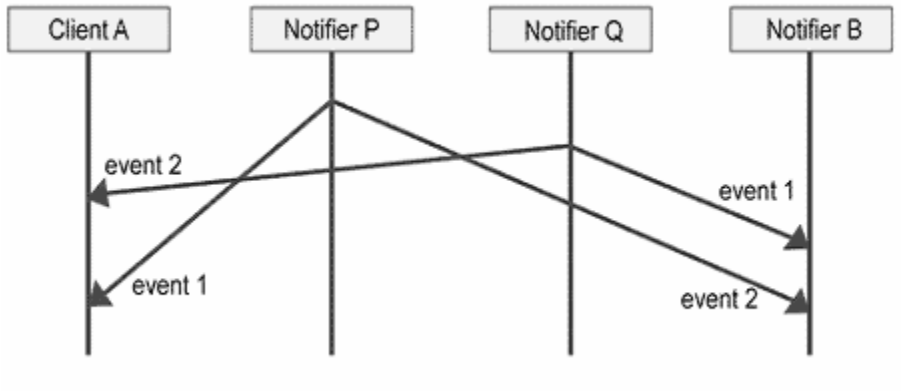


Fig 2.4.3: The Relativistic Effect

As a consequence, the two clients may reach different conclusions about the state of the system.

2.5 SOFTWARE FAULT TOLERANCE

Most Real-time systems focus on hardware fault tolerance. Software fault tolerance is often overlooked. This is really surprising because hardware components have much higher reliability than the software that runs over them. Most system designers go to great lengths to limit the impact of a hardware failure on system performance. However they pay little attention to the systems behavior when a software module fails [23].

There are several techniques that can be used to limit the impact of software faults on system performance. The main idea here is to contain the damage caused by software faults. Software fault tolerance is not a license to ship the system with bugs. The real objective is to improve system performance and availability in cases when the system encounters a software or hardware fault.

2.5.1 Timeouts

Most Real-time systems use timers to keep track of feature execution. A timeout generally signals that some entity involved in the feature has misbehaved and a corrective action is required [23]. The corrective action could be of two forms:

- **Retry:** When the application times out for a response, it can retry the message interaction. You might argue that we do not need to implement application level retries as lower level protocols will automatically recover from message loss. Keep in mind that message loss recovery is not the only objective of implementing retries. Retries help in recovering from software faults too. Consider a scenario where a message sent to a task is not processed because of a task restart or processor reboot. An application level retry will recover from this condition.
- **Abort:** In this case timeout for a response leads to aborting of the feature. This might seem too drastic, but in reality aborting a feature might be the simplest and safest solution in recovering from the errors. The feature might be retried by the user invoking the feature. Consider a case where a call has to be cleared because the task originating the call did not receive a response in time. If this condition can happen only in rare scenarios, the simplest action on timeout might be to clear the call. The user would retry the call.

The choice between retrying or aborting on timeouts is based on several factors. Considering all these factors before making any type of decision are:

- If the feature being executed is fairly important for system stability, it might be better to retry. For example, a system startup feature should not be aborted on one timeout.
- If the lower layer protocol is not robust, retry might be a good option. For example, message interactions using an inherently unreliable protocol like slotted aloha should always be retried.

- Complexity of implementation should also be considered before retrying a message interaction. Aborting a feature is a simpler option. More often than not system designers just default to retrying without even considering the abort option. Keep in mind that retry implementation complicates the code and state machine design.
- If the entity invoking this feature will retry the feature, the simplest action might be abort the feature and wait for an external retry.
- Retrying every message in the system will lower system performance because of frequent timer start and stop operations. In many cases, performance can be improved by just running a single timer for the complete feature execution. On timeout the feature can simply be aborted.
- For most external interactions, the designer might have no choice. As the timeouts and retry actions are generally specified by the external protocols.
- Many times the two techniques are used together. The task retries a message certain number of times. If no response is received after exhausting this limit, the feature might be aborted.

2.5.2 Audits

Most Real-time systems comprise of software running across multiple processors. This implies that data is also distributed [24]. The distributed data may get inconsistent in Real-time due to reasons like:

- independent processor reboot
- software bugs
- race conditions
- hardware failures
- protocol failures

The system must behave reliably under all these conditions. A simple strategy to overcome data inconsistency is to implement audits. Audit is a program that checks the consistency of data structures across processors by performing predefined checks [26].

Audit Procedure

a) System may trigger audits due to several reasons:

- periodically
- failure of certain features
- processor reboots
- processor switchovers
- certain cases of resource congestion

b) Audits perform checks on data and look for data inconsistencies between processors.

c) Since audits have to run on live systems, they need to filter out conditions where the data inconsistency is caused by transient data updates. On data inconsistency detection, audits perform multiple checks to confirm inconsistency. A inconsistency is considered valid if and only if it is detected on every iteration of the check.

d) When inconsistency is confirmed, audits may perform data structure cleanups across processors.

e) At times audits may not directly cleanup inconsistencies; they may trigger appropriate feature aborts etc.

2.5.3 Exception Handling

Whenever a task receives a message, it performs a series of defensive checks before processing it. The defensive checks should verify the consistency of the message as well as the internal state of the task. Exception handler should be invoked on defensive check failure.

Depending on the severity, exception handler can take any of the following actions:

- Log a trace for developer post processing.
- Increment a leaky-bucket counter for the error condition.
- Trigger appropriate audit.
- Trigger a task rollback.
- Trigger processor reboot.

Leaky-bucket counters are used to detect a flurry of error conditions. To ignore rare error conditions they are periodically leaked i.e. decremented. If these counters reach a certain threshold, appropriate exception handling is triggered. Note that the threshold will never be crossed by rare happening of the associated error condition. However, if the error condition occurs rapidly, the counter will overflow i.e. cross the threshold.

2.5.4 Task Rollback

In a complex Real-time system, a software bug in one task leading to processor reboot may not be acceptable. A better option in such cases is to isolate the erroneous task and handle the failure at the task level. The task in turn may decide to rollback i.e. start operation from a known or previously saved state. In other cases, it may not be expensive to forget the context by just deleting the offending task and informing other associated tasks[26].

For example, if the Space Slot Manager on the CAS card encounters a exception condition leading to task rollback, it might resume operation by recovering the space slot allocation status from the connection memory. On the other hand, exception in a call task might just be handled by clearing the call task and releasing all the resources assigned to this task.

Task rollback may be triggered by any of the following events:

- Hardware exception conditions like divide by zero, illegal address access (bus error)
- Defensive check leaky-bucket counter overflows.
- Audit detected inconsistency to be resolved by task rollback.

2.5.5 Incremental Reboot

Software processor reboots can be time consuming, leading to unacceptable amount of downtime. To reduce the system reboot time, complex Real-time systems often implement incremental system initialization procedures. For example, a typical Real-time system may implement three levels of system reboot:

Level 1 Reboot: Operating system reboot

Level 2 Reboot: Operating system reboot along with configuration data download

Level 3 Reboot: Code reload followed by operating system reboot along with configuration data download.

Incremental Reboot Procedure

A defensive check leaky-bucket counter overflow will typically lead to rollback of the offending task. In most cases task rollback will fix the problem. However, in some cases, the problem may not be fixed leading to subsequent rollbacks too soon. This will cause the task level rollback counter to overflow, leading to a Level 1 Reboot.

Most of the times, Level 1 Reboot will fix the problem. But in some cases, the processor may continue to hit Level 1 Reboots repeatedly. This will cause the Level 1 Reboot counter to overflow, leading to a Level 2 Reboot.

Majority of the times, Level 2 Reboot is able to fix the problem. If it is unable to fix the problem, the processor will repeatedly hit Level 2 Reboots, causing the Level 2 Reboot counter to overflow leading to Level 3 Reboot.

2.5.6 Voting

This is a technique that is used in mission critical systems where software failure may lead to loss of human life .e.g. aircraft navigation software. Here, the Real-time system software is developed by at least three distinct teams. All the teams develop the software independently. And, in a live system, all the three implementations are run simultaneously. All the inputs are fed to the three versions of software and their outputs are voted to determine the actual system response. In such systems, a bug in one of the three modules will get voted out by the other two versions.

2.6 HARDWARE FAULT TOLERANCE

Most Real time systems must function with very high availability even under hardware fault conditions. There are several techniques that are used to minimize the impact of hardware faults [26].

2.6.1 Redundancy Schemes

Real time systems are equipped with redundant hardware modules. Whenever fault occurs, redundant modules use the functions of failed hardware modules. Hardware redundancy may be provided in one of the following ways:

- One for One Redundancy
- N + X Redundancy
- Load Sharing

2.6.1.1 <u>One for One Redundancy</u>

Here, each hardware module has a redundant hardware module. The hardware module that performs the functions under normal conditions is called *Active* and the redundant unit is called *Standby*. The standby keeps monitoring the active unit at all times. It will takeover and become active if the active unit fails. Since standby has to takeover under fault conditions it has to keep itself synchronized with the active unit operations.

Since the probability of both the units failing at the same time is very low, this technique provides the highest level of availability. The main disadvantage here is that it doubles the hardware cost[26].

2.6.1.2 <u>N + X Redundancy</u>

In this scheme, if N hardware modules are required to perform system functions, the system is configured with N + X hardware modules; typically X is much smaller than N. Whenever any of the N modules fails, one of the X modules takes over its functions. Since health monitoring of N units by X

units at all times is not practical, a higher level module monitors the health of N units. If one of the N units fails, it selects one of the X units (It may be noted that one for one is a special case of N + X).

The advantage lies in reduced hardware cost of the system as only X units are required to backup N units. However, in case of multiple failures, this scheme provides lesser system availability.

2.6.1.3 Load Sharing

In this scheme, under zero fault conditions, all the hardware modules that are equipped to perform system functions, share the load. A higher level module performs the load distribution. It also maintains the health status of the hardware units. If one of the load sharing module fails, the higher level module starts distributing the load among the rest of the units. There is graceful degradation of performance with hardware failure [26].

Here, there is almost no extra hardware cost to provide the redundancy. The main disadvantage is that if a hardware failure happens during the busy hour, system will perform at a sub-optimal level until the failed module is replaced.

Network Load Balancing

Network load balancing is a different flavor of load sharing where there is no higher level processor to perform load distribution. Instead, the load distribution is achieved by hashing on the source address bits. For example, many high traffic websites perform load sharing by broadcasting the HTTP Get request over the Ethernet to all the load sharing machines. The network card on the load sharing machines is appropriately configured to pass a certain portion of the HTTP Get requests to the main computer. The remaining requests are filtered out as they will be handled by other machines. If one of the load sharing machines fails, filter settings on all the active machines are appropriately modified to redistribute the traffic.

2.6.2 Standby Synchronization

For redundancy to work, the standby unit needs to be kept synchronized with the active unit at all times. This is required so that the standby can fit into the active's boots in case the active fails[26]. The standby synchronization can be achieved in the following ways:

- Bus Cycle Level Synchronization
- Memory Mirroring
- Message Level Synchronization
- Checkpoint Level Synchronization

2.6.2.1 Bus Cycle Level Synchronization

In this scheme the active and the standby are locked at processor bus cycle level. To keep itself synchronized with the active unit, the standby unit watches each processor instruction that is performed by active. Then, it performs the same instruction in the next bus cycle and compares the output with that of the active unit. If the output does not match, the standby might takeover and become active. The main disadvantage here is that specialized hardware is needed to implement this scheme. Also, bus cycle level synchronization introduces wait states in bus cycle execution. This will lower the overall performance of the processor.

2.6.2.2 Memory Mirroring

Here, the system is configured with two CPUs and two parity based memory cards. One of the CPU is active and the other is standby. Both the memory cards are driven by the active CPU. No memory is attached to the standby unit. Each memory write by the active is made to both the memory cards. The data bits and the parity bits are updated individually on both the memory cards. On every memory read, the output of both the memory cards is compared. If a mismatch is detected, the processor believes the memory card with correct parity bit. The other memory card is marked suspected and a fault trigger is generated.

The standby unit continuously monitors the health of the active unit by sanity punching or watchdog mechanism. If a fault is detected, the standby takes over both the memory cards. Since the application context is kept in memory, the new active processor gets the application context.

The main disadvantage here is that specialized hardware is needed to implement this scheme. Also, memory mirroring introduces wait states in bus cycle execution. This will lower the overall performance of the processor.

2.6.2.3 Message Level Synchronization

In this scheme, active unit passes all the messages received from external sources to the standby. The standby performs all the actions as though it were active with the difference that no output is sent to the external world. The main advantage here is that no special hardware is required to implement this. The scheme is practical only in conditions where the processor is required to take fairly simple decisions. In

cases of complex decisions, the synchronization can be easily lost if the two processor take different decisions on the same input message.

2.6.2.4 Checkpoint Level Synchronization

To some extent, this one is like message level synchronization as active conveys synchronization information in terms of messages to standby. The difference is that all the external world messages are not conveyed. The information is conveyed only about predefined milestones. For example, in a Call Processing system, checkpoints may be passed only when the call reaches conversation or is cleared. If standby takes over, all the calls in conversation would be retained whereas all calls in transient states will be lost. Resource information for the transient calls may be retrieved by running software audits with other modules. This scheme is not prone to loss of synchronization under normal conditions[26]. Also, the message traffic to the standby is reduced, thus improving the overall performance of the active.

3. DATA STORAGE AND RECOVERY IN NETWORKS

3.1 RAID

RAID stands for "***Redundant Array of Independent Disks".*** It combines two or more disk drives into an array (set). "*RAID controller"* makes the RAID set appear to the host as a single logical disk drive[27]. Properly implemented RAID sets provide:

- Improved I/O performance
- Higher Data availability
- Streamlined management of storage devices

RAID storage has now grown up from academic concept to industry standard.

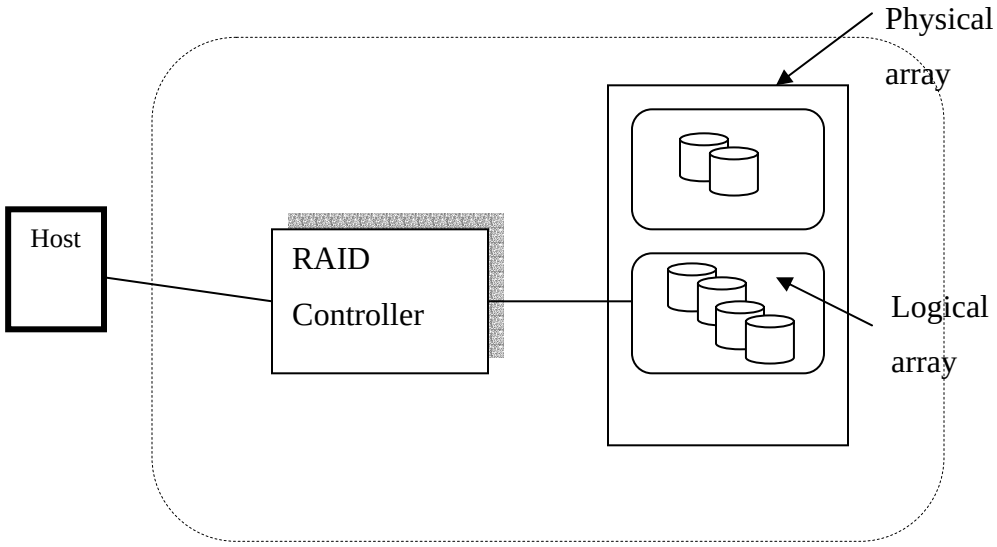


Fig 3.1: RAID components

RAID array is sum of disk array and RAID controller. Physical Disks in RAID array are contained with in smaller sub-enclosures called *"Physical arrays"* that holds a fixed number of physical disks and may also include other supporting hardware such as power supplies.

A subset of disks forming logical groupings called as *"Logical arrays"* or *"RAID set"*.
The operating system see these disk groups as if they were regular disk volumes. The array management software in RAID systems handles management and control of disk aggregations, translation of I/O requests between the logical and physical disks, and error corrections if disk failure occurs.

Strips and Stripes

With in each disk, there are groups of contiguously addressed blocks, called ***strips.*** The set of aligned strips that spans across all the disks within the RAID set is called a ***stripe.***

Stripe width can be calculated by number of data strips in a stripe. *Stripe size* describes the number of data blocks in a stripe.

Striping improves performance by distributing data across the disks in the array. When a large amount of data is written, the first piece is sent to the first drive, the second piece to second and so on. The pieces are put together when data is read. Striping can occur at block level or byte level. Higher stripe width means a higher number of drives and therefore better performance [17].

3.1.1 RAID Redundancy

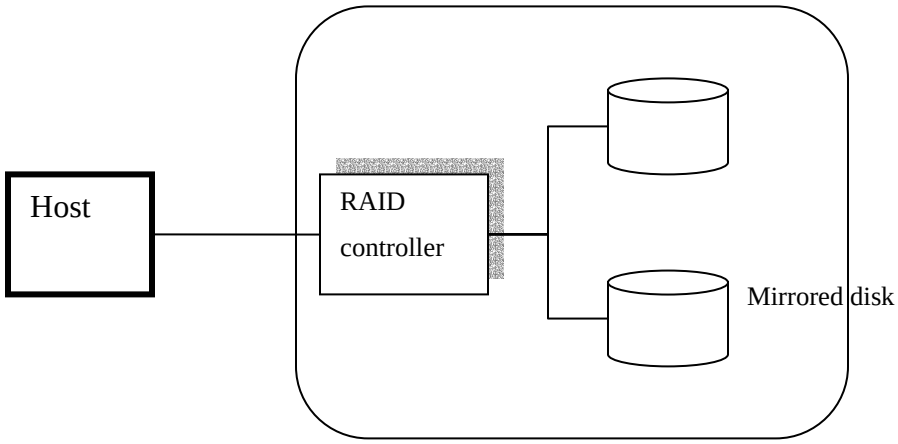


Fig 3.1(b): RAID Redundancy

In RAID architecture, redundancy is introduced to improve fault tolerance. Mirroring uses multiple drives that hold identical copies of data (normally 2 drives). Every write to a disk is also a write to a mirrored disk, meaning that all drives contains the same data. It has some benefits that we can recover from failure at efficient speed. On the other side, it degrades the write performance as each block of host data is written to multiple disks [17].

3.1.2 RAID parity blocks

In certain RAID levels (all but 0, 1, and 2), redundancy is achieved by the use of parity blocks. If a single drive in the array fails, data blocks and a parity block from the working drives can be combined to reconstruct the missing data

Given the diagram below, assume A1 = 00000111, A2 = 00000101, and A3 = 0000000. Ap, generated by XORing A1, A2, and A3, will then equal 00000010. If the second drive fails, A2 will no longer be accessible, but can be reconstructed by XORing A1, A3, and Ap [17].

A1 XOR A3 XOR Ap = 00000101 (A2)

RAID ARRAY

A1	A2	A3	A_P
B1	B2	B_P	B3
C1	C_P	C2	C3
D_P	D1	D2	D3

3.1.3 RAID LEVELS

- RAID 0 : Striped array with no fault tolerance
- RAID 1 : Disk Mirroring
- RAID 3 : Parallel access array with dedicated parity disk
- RAID 4 : Striped array with independent disks
- RAID 5 : Striped array with independent disks and distributed parity

3.1.3.1 RAID 0 : Striped array with no fault tolerance

A RAID 0 (also known as a striped set) splits data evenly across two or more disks with no parity information for redundancy. It is not redundant. RAID 0 is normally used to increase performance, although it can also be used as a way to create a small number of large virtual disks out of a large number of small physical ones. A RAID 0 can be created with disks of differing sizes, but the storage space added to the array by each disk is limited to the size of the smallest disk. RAID 0 implementations with more than two disks are also possible, however the reliability of a given RAID 0 set is equal to the average reliability of each disk divided by the number of disks in the set. That is, reliability (as measured by mean time to failure (MTTF) or mean time between failures (MTBF)) is roughly inversely

proportional to the number of members—so a set of two disks is roughly half as reliable as a single disk. The reason for this is that the file system is distributed across all disks[17].

When a drive fails the file system cannot cope with such a large loss of data and coherency since the data is "striped" across all drives. Data can be recovered using special tools. However, it will be incomplete and most likely corrupt. The transfer speed of the array will be the transfer speed of all the disks added together, limited only by the speed of the RAID controller.

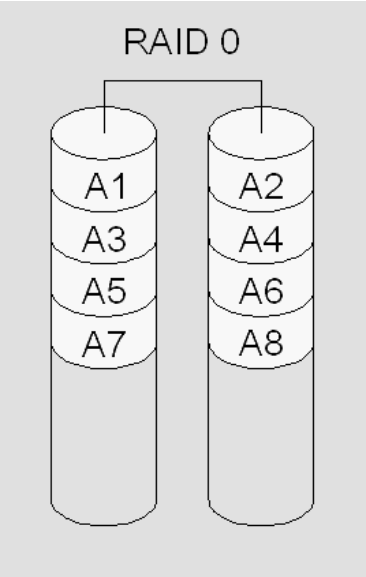


Fig 3.1.3.1 : RAID 0

RAID 0 is useful for setups such as large read-only NFS servers where mounting many disks is time-consuming or impossible and redundancy is irrelevant. Another use is where the number of disks is limited by the operating system. In Microsoft Windows, the number of drive letters for hard disk drives may be limited to 24, so RAID 0 is a popular way to use more disks. It is possible in Windows 2000 professional(& newer) to mount partitions under directories, much like unix, and hence eliminating the need for a partition to be assigned a drive letter. RAID 0 is also a popular choice for gaming systems where performance is desired, data integrity is not very important, but cost is a consideration to most users.

3.1.3.2 RAID 1 : Disk Mirroring

RAID 1 creates an exact copy (or **mirror**) of a set of data on two or more disks. This is useful when read performance is more important than data capacity. Such an array can only be as big as the smallest member disk. Basically, a RAID 1 mirrored pair contains two disks, which increases reliability exponentially over a single disk. Since each member contains a complete copy of the data, and can be addressed independently, reliability is raised by the power of the number of self-contained copies[27].

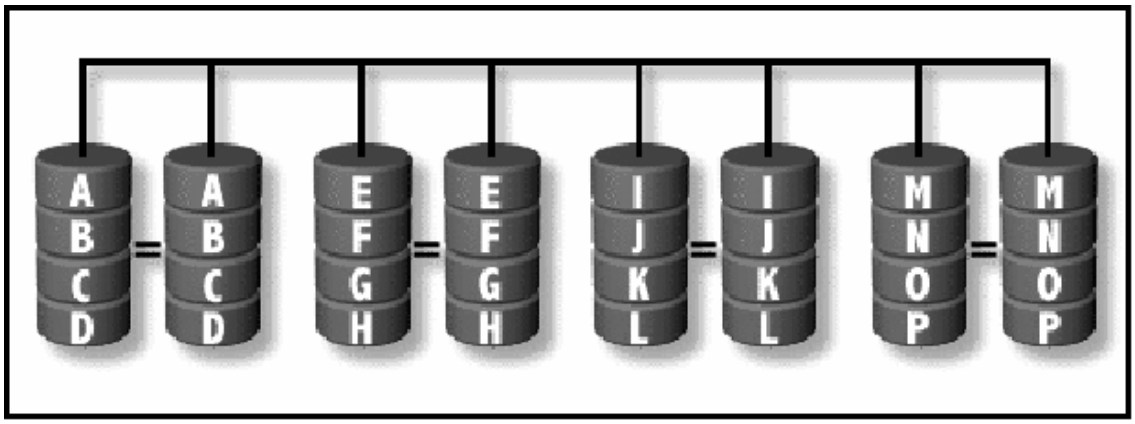


Fig 3.1.3.2: RAID 1

3.1.3.3 RAID 3 : Parallel access array with dedicated parity disk

A RAID 3 uses byte-level striping with a dedicated parity disk. RAID 3 is very rare in practice. One of the side effects of RAID 3 is that it generally cannot service multiple requests simultaneously. This comes about because any single block of data will by definition be spread across all members of the set and will reside in the same location, so any I/O operation requires activity on every disk [17].

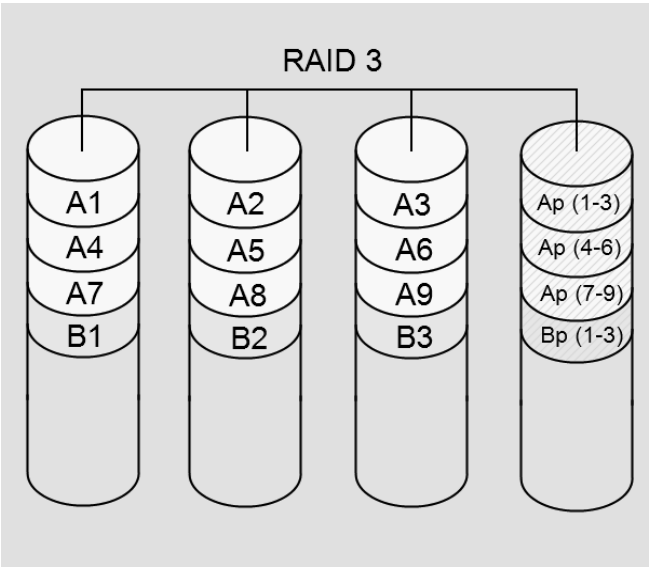


Fig 3.1.3.3: RAID 3

A request for block "A" consisting of bytes A1-A9, would require all three data disks to seek to the beginning (A1) and reply with their contents. A simultaneous request for block B would have to wait. This level requires minimum three drives to implement and have good fault tolerance.

3.1.3.4 RAID 4: Striped array with independent disks

A RAID 4 uses block-level striping with a dedicated parity disk. RAID 4 looks similar to RAID 3 except that it stripes at the block, rather than the byte level. This allows each member of the set to act independently when only a single block is requested. If the disk controller allows it, a RAID 4 set can service multiple read requests simultaneously.

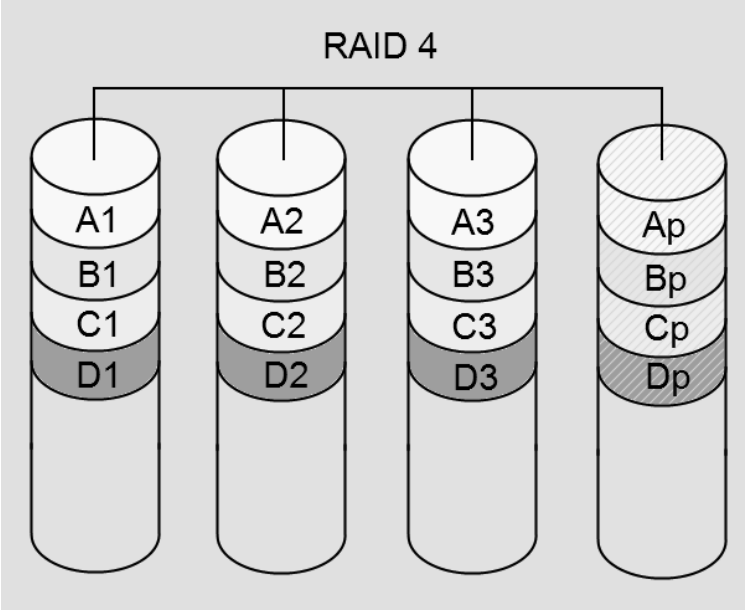


Fig 3.1.3.4: RAID 4

In our example above, a request for block "A1" would be serviced by disk 1. A simultaneous request for block B1 would have to wait, but a request for B2 could be serviced concurrently [17].
The random read performance for this level is good and has better sequential read and write performance. On the other side, random write performance is poor and has a poor controller design.

3.1.3.5 RAID 5 : Striped array with independent disks and distributed parity

A RAID 5 uses block-level striping with parity data distributed across all member disks. RAID 5 has achieved popularity due to its low cost of redundancy. Generally, RAID 5 is implemented with hardware support for parity calculations.

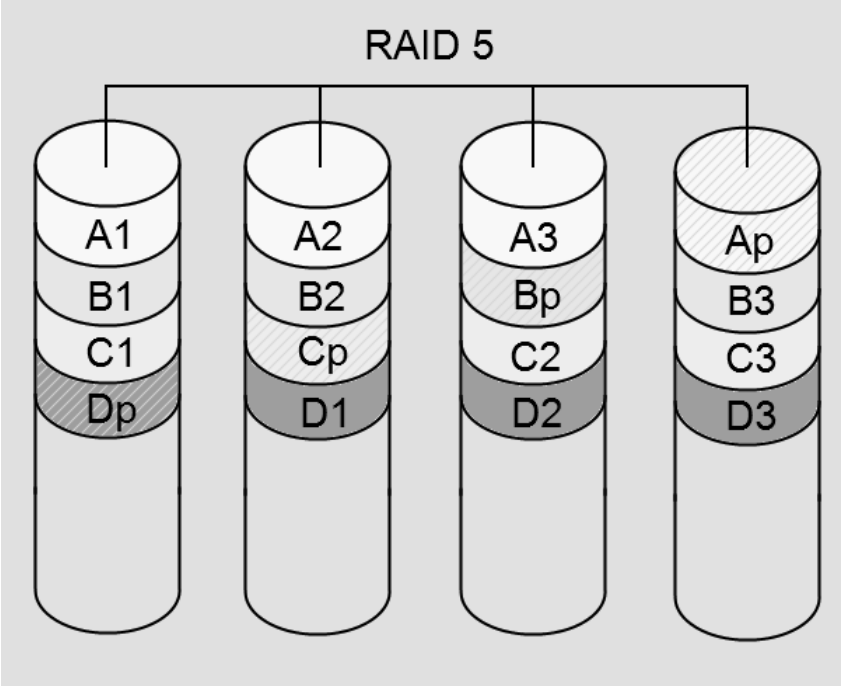


Fig 3.1.4.5: RAID 5

In the example above, a read request for block "A1" would be serviced by disk 1. A simultaneous read request for block B1 would have to wait, but a read request for B2 could be serviced concurrently [17].

Every time a block is written to a disk in a RAID 5, a parity block is generated within the same stripe. A block is often composed of many consecutive sectors on a disk. A series of blocks (a block from each of the disks in an array) is collectively called a "stripe". If another block, or some portion of a block, is written on that same stripe the parity block (or some portion of the parity block) is recalculated and rewritten. For small writes, this requires reading the old data, writing the new parity, and writing the new data. The disk used for the parity block is staggered from one stripe to the next, hence the term "distributed parity blocks". RAID 5 writes are expensive in terms of disk operations and traffic between the disks and the controller

3.2 Network Storage Systems

The storage system consists of a node (computer), a link and Device to store data. The main purpose of storage systems is to provide storage to one or more computers. These systems are responsible for actual transmission of data and to read or write data [28].

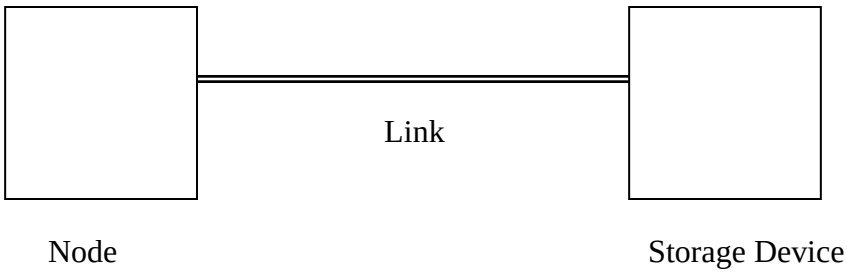


Fig 3.2 : Simple Storage system

The node always communicates with the applications which require data. Link joining node and storage device is responsible for carrying read or write commands and data between node and device to store. These links follows some protocols (which are a predefined format for communication) so that both sender and receiver will agree on what is being communicated.
There are two most popular interfaces used for storage devices: SCSI and FC.

SCSI (Small Computer System Interface) is commonly used as means of internal storage in computers. It supports multiple simultaneous data access. They are not useful for home users or business desktop users. SCSI can connect many devices to a computer and are highly backward compatible. They have faster transfer speeds up to 320 megabytes per second. They can be used as external storage interfaces but they can be used for limited distance and have limited hop count [29].

FC (Fibre Channel) connect servers to shared storage devices and used as network storage. The term fibre channel refers to both hardware components and the storage protocol that communicates across the channel elements. They can be used for larger distance.

The Network storage enables:

1. More effective utilization of available storage
2. Increased flexibility
3. Better availability of data

There are different network storage models which we will discuss below:

- Direct Attached Storage (DAS)
- Networked Attached Storage (NAS)
- Fibre Channel Storage Area Networks (FC SAN)
- IP Storage Area Networks (IP SAN)

3.2.1 Direct Attached Storage (DAS)

DAS is the very primary level of storage. All the devices of DAS either reside as an integrated part of host computer (e.g. hard drives, removable storage devices etc.) pr directly connected to a single server externally (RAID). All the storage resources are dedicated to the sites that are using them [29].

Two types:

1. Internal DAS
2. External DAS

Internal DAS: used in very small environments as it is easier to deploy and inexpensive. They are generally managed through operating system.

External DAS: used in environments with a single host and few hosts. Examples are small businesses, workgroups etc. Large organizations may use DAS for mission critical applications too. They are not dependent on a particular host's operating system.

In DAS, all hosts must be directly connected. There is no fault tolerance in the existing system. The scalability of DAS is limited.

3.2.2 Networked Attached Storage (NAS)

NAS is a shared storage on network. A NAS server is a storage device that consists of a high performance file server and attached to a LAN. It is single purpose machine whose job is to serve as high speed communication gateway to file data.

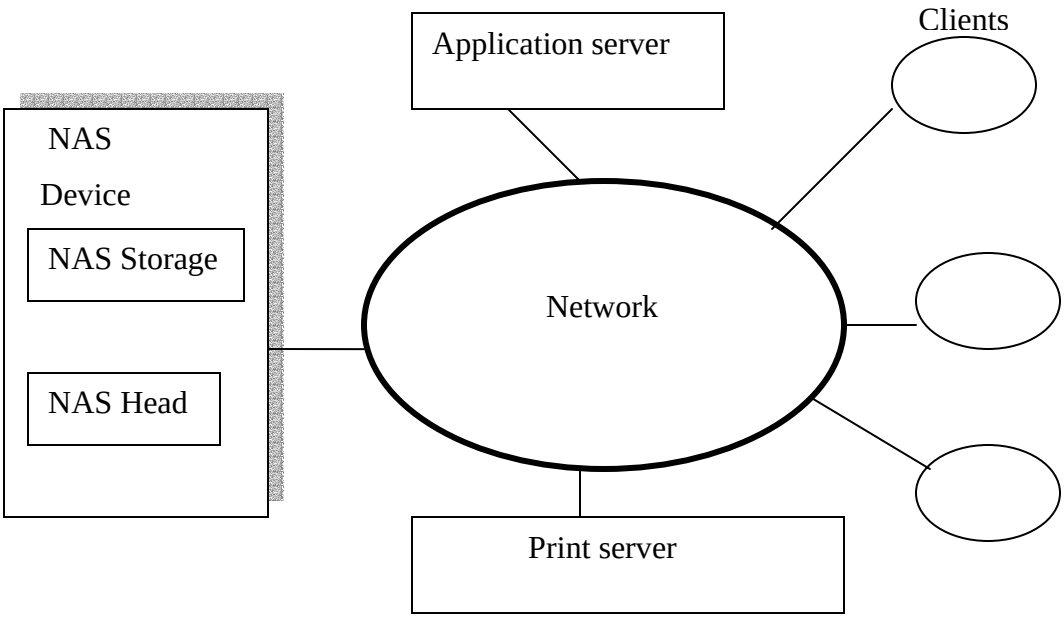


Fig 3.2.2: NAS Architecture

NAS head can be located at remote location but NAS storage is dedicated to NAS applications i.e. they are integrated. NAS supports multiple protocols.

Some benefits of NAS:

- *Provides Flexibility:* using many industry standards, it can work with many type of clients on both UNIX or Microsoft window platforms.
- *Central storage:* easy to manage.
- *Scalable :* It has high performance that enables it to address many different type of business applications
- *High availability:* It has many replication and recovery options. Also, it uses clustering technology for fail over in the event of file failure.
- *High Security:* in conjunction with industry standards, it handles security and user authentication.
- *Global information access:* enable greater file sharing even at a larger distance.
- *Improve Efficiency:* eliminates problems faced during file access and improve efficiency through specialized operating system.

3.2.3 Fibre Channel Storage Area Networks (FC SAN)

A Storage Area Network (SAN) is a dedicated network that carries data between computer systems and storage devices (e.g. tapes and disks). Data transfer in SAN is secure and robust [32].

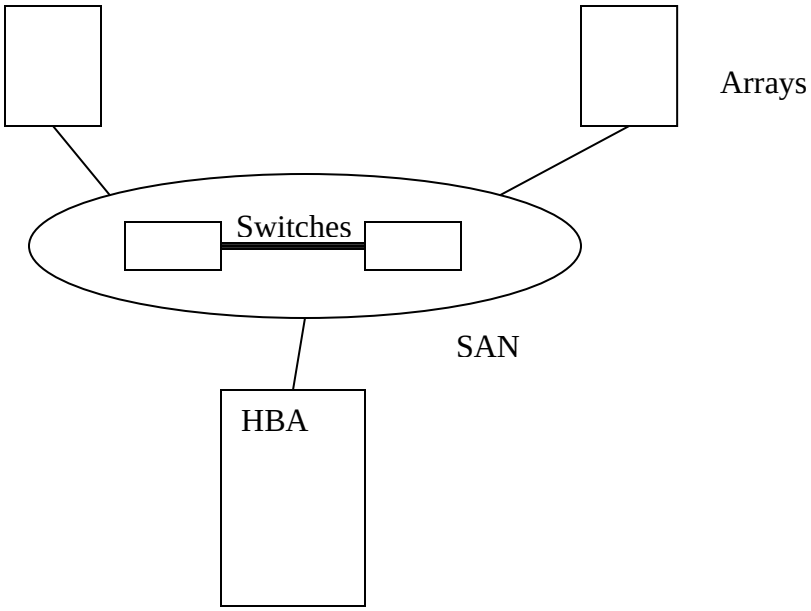


Fig 3.2.3: SAN components

HBA (Host Bus Adapter) does low level interface functions automatically to improve host processor performance. All hosts connect to SAN via HBA. The cabling done is optical. Fibre channel switches are used to connect the nodes. It also uses storage arrays. Fibre channel define protocols for performing high speed serial data transfer upto 400 Megabytes per second. It provides standard data transport medium over which computer systems communicate with devices such as disk storage arrays.

Connectivity is provided by fibre channel hubs, switches and directors. These devices act as the communication link between nodes with in the SAN. A fibre channel switch is capable enough to route data from one physical port to another directly.

Some benefits of SAN are:

- *Higher bandwidth :* since they are using Fibre channel
- *SCSI extension:* SCSI over fibre channel implementations allow these devices to be connected in dynamic fibre channel topologies which can travel longer distances and provides higher level of flexibility and manageability.
- *Centralized storage and easy management*
- *Scalable:* up to 16 million devices.
- *Secure Access*

3.2.4 IP Storage Area Networks (IP SAN)

Earlier, The traffic in IP networks was at file system level. Now, emerging technologies provide transfer of block level data over an existing IP network infrastructure.
IP is chosen as a storage transport because its management is easier. Many administrators in the organizations are familiar with it. It also supports multi- vendor interoperability. It has many robust and mature security options available. Cost can be reduced by using these networks[30].

We can send Blocks over IP using three protocols which are gaining more popularity today. These are:

- FCIP
- iFCP
- iSCSI

3.2.4.1 FCIP (Fiber Channel to IP)

It is a TCP/IP based encapsulating protocol for connecting fibre channel SANs. The FC frames are first encapsulated into IP packets and transmitted through a “dumb” tunnel, essentially creating an extending routing system of fibre channel switches [28].

FCIP describes mechanisms that allow the interconnection of Fibre Channel storage area networks over IP-based networks to form a unified storage area network in a single Fibre Channel fabric. FCIP relies on IP-based network services to provide the connectivity between the storage area network over local area networks, metropolitan area networks, or wide area networks.

The primary function of an FCIP Entity is forwarding FC Frames, employing FC Frame Encapsulation. Viewed from the IP Network perspective, FCIP Entities are peers and communicate using TCP/IP . Each FCIP Entity contains one or more TCP endpoints in the IP-based network. Viewed from the FC Fabric perspective, pairs of FCIP Entities, in combination with their associated FC Entities, forward FC Frames between FC Fabric elements. The FC End Nodes are unaware of the existence of the FCIP Link.

An FCIP Entity MAY contain multiple FCIP Link Endpoints, but each FCIP Link Endpoint (FCIP_LEP) communicates with exactly one other FCIP_LEP. FCIP Entities do not actively participate in FC Frame routing. The FCIP Control & Services module MAY use TCP/IP quality of service features. It is necessary to statically or dynamically configure each FCIP entity with the IP addresses and TCP port numbers corresponding to FCIP Entities with which it is expected to initiate communication [28].

To support IP Network security, FCIP Entities MUST: 1) implement cryptographically protected authentication and cryptographic data integrity keyed to the authentication process, and 2) implement data confidentiality security features.

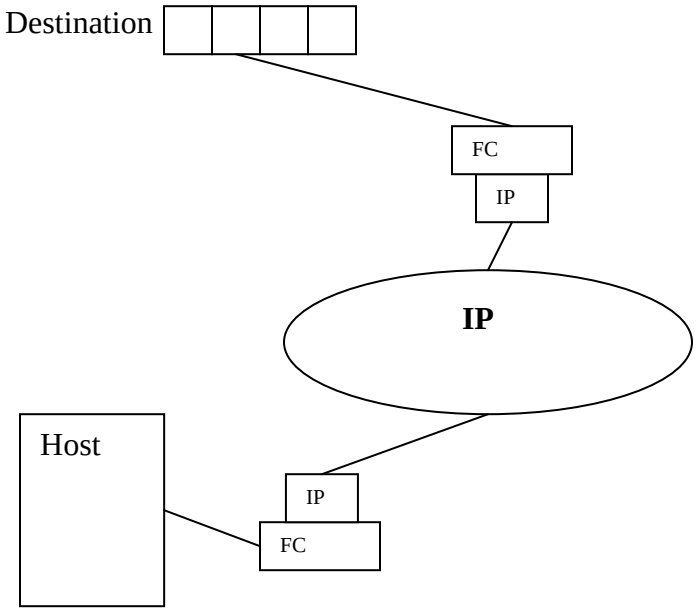


Fig 3.2.4.1: FCIP

It is used to extend fibre channel networks over greater distances over IP based infrastructures. FCIP provides support for existing applications and is very cost effective. It also allows multi point networking. They are widely available and follows mature standards.

3.2.4.2 iFCP (Internet Fibre Channel Protocol)

It is a gateway-to-gateway protocol for the implementation of fibre channel fabric in which TCP/IP switching and routing elements replace fibre channel components. Fibre channel switches use IP as

inter-switch fabric protocol [28]. It makes use of existing IP infrastructure (cabling, switches) to support server-to-storage communications.

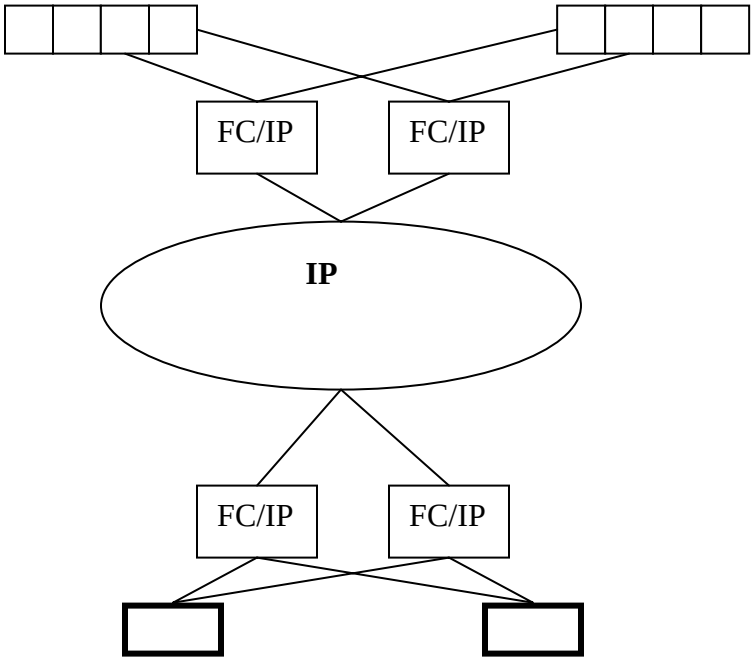


Fig 3.2.4.2: iFCP

The main function of the iFCP protocol layer is to transport fibre channel frame images between locally and remotely attached N_PORTs. When transporting frames to a remote N_PORT, the iFCP layer encapsulates and routes the fibre channel frames comprising each fibre channel Information Unit via a predetermined TCP connection for transport across the IP network.

When receiving fibre channel frame images from the IP network, the iFCP layer de-encapsulates and delivers each frame to the appropriate N_PORT. The iFCP layer processes the following types of traffic:

- FC-4 frame images associated with a fibre channel application protocol.
- FC-2 frames comprising fibre channel link service requests and responses
- Fibre channel broadcast frames
- iFCP control messages required to setup, manage or terminate an iFCP session.

iFCP is very flexible as it can function as

- IP tunnel for FC-to-FC FCP data transport
- Bridge for FC-to-IP FCP data transport
- Replacement for FC for IP-to-IP FCP data transport

It has a TCP session for every virtual port-to-port connection. iFCP is a TCP/IP protocol that transports encapsulated frame images between gateways. iFCP uses TCP to provide congestion control, error detection and recovery. The primary objective of iFCP is to allow interconnection and networking of existing fibre channel devices at wire speeds over an IP network. The protocol and method of frame address translation defined permit the attachment of fibre channel storage devices to an IP-based fabric by means of transparent gateways[28].

The fundamental entity in fibre channel is the fibre channel network. Unlike a layered network architecture, a fibre channel network is largely specified by functional elements and the interfaces between them. These consist, in part, of the following:

1. N_PORTs -- The end points for fibre channel traffic.
2. FC Devices -- The fibre channel devices to which the N_PORTs provide access.
3. Fabric Ports -- The interfaces within a fibre channel network that provide attachment for an N_PORT.
4. The network infrastructure for carrying frame traffic between N_PORTs.
5. Within a switched or mixed fabric, a set of auxiliary servers, including a name server for device discovery and network address resolution.

The iFCP protocol enables the implementation of fibre channel fabric functionality on an IP network in which IP components and technology replace the fibre channel switching and routing infrastructure.

3.2.4.3 iSCSI (SCSI over IP)

It transfers blocks of data using TCP/IP network. TCP is a reliable transport since it retransmits dropped packets where as IP is an unreliable as packet dropping is allowed.
SCSI frames are encapsulated into IP packets and transmit it over network to either a box that extracts SCSI from IP packets and sends these on to FC-based external storage.
IP encapsulation is done on host (Host Bus Adapter). It is highly reliable[28].

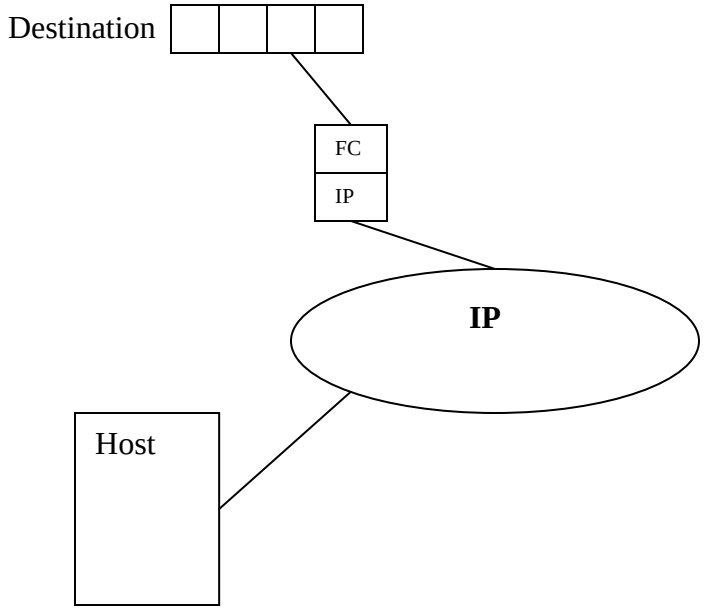


Fig 3.2.4.3: iSCSI

The primary market driver for the development of the iSCSI protocol is to enable broader access of the large installed base of DAS over IP network infrastructures. By allowing greater access to DAS devices over IP networks, these storage resources can be maximized by any number of users or utilized by a variety of applications such as remote backup, disaster recovery, and storage virtualization. A secondary driver of iSCSI is to allow other SAN architectures such as Fibre Channel to be accessed from a wide variety of hosts across IP networks. iSCSI enables block-level storage to be accessed from Fibre Channel SANs using IP storage routers or switches, furthering its applicability as an IP-based storage transport protocol [31].

Between the standards efforts coming to completion, iSCSI-compliant products will enable users to rapidly deploy IP SAN environments and immediately take advantage of the "plug-and-play" benefits of iSCSI. Many iSCSI products are already available, based on early versions of the specification.

iSCSI defines the rules and processes to transmit and receive block storage applications over TCP/IP networks. At the physical layer, iSCSI supports a Gigabit Ethernet interface so that systems supporting iSCSI interfaces can be directly connected to standard Gigabit Ethernet switches and/or IP routers. The iSCSI protocol sits above the physical and data-link layers and interfaces to the operating system's standard SCSI Access Method command set. iSCSI enables SCSI-3 commands to be encapsulated in TCP/IP packets and delivered reliably over IP networks.

iSCSI can be supported over any physical media that supports TCP/IP as a transport, but today's iSCSI implementations are on Gigabit Ethernet. The iSCSI protocol runs on the host initiator and the receiving target device. iSCSI can run in software over a standard Gigabit Ethernet network interface card (NIC) or can be optimized in hardware for better performance on an iSCSI host bus adapter (HBA).

iSCSI also enables the access of block-level storage that resides on Fibre Channel SANs over an IP network via iSCSI-to-Fibre Channel gateways such as storage routers and switches

3.3 Back Up / Recovery Techniques

By the term “Back up”, means copy of the online data that resides on primary storage. If the data gets corrupted or deleted on primary storage then we can use this backup copy. Normally, The backup is retained over a period of time, depending on the type of data and on the type of back up[33].

The backup copy can be created as *simple copy* since there can be multiple copies of that particular data. Also, it can be created as *Mirrored copy* i.e. the copy is always updated with whatever is written on primary (master) copy. The data that is backed up may use media such as tape or disk. Backup copy is needed mostly when master copy is lost or corrupted.

Remote backup can be one of the choice for backup. The master copy is directly saved on the backup media that is available on the other site. The backup media can be a remote file system[34].

Backup can be one of the three different types :

1. *Operational backup :* needed only when data is lost or corrupted
2. *Archival backup:* used to save transaction records, business work products for regulatory compliance.
3. *Recovery from Disaster:* Here we have to recover all data. For e.g., a company has lost all its data in major disaster then all data needs to be recovered from backup copies.

Why we need to back up our data?

There are some reasons mentioned below for the need of backups [34]:

- Event and natural disasters such as earthquakes, floods, tornados, lightning strikes, hurricanes, accidents, power grid failures etc. forces us to think about some back up plans as they can cause huge damage to a particular industry.

- Security breaches can lead to considerable data loss.
- Different corporations may establish their own policies for their property backup.
- Software failures can destroy or lose data or it can be because of virus attack.
- Any physical damage to a storage media can eventually lead to data loss.
- Human error or unhappy employees or hackers may destroy data.

3.3.1 General backup architecture

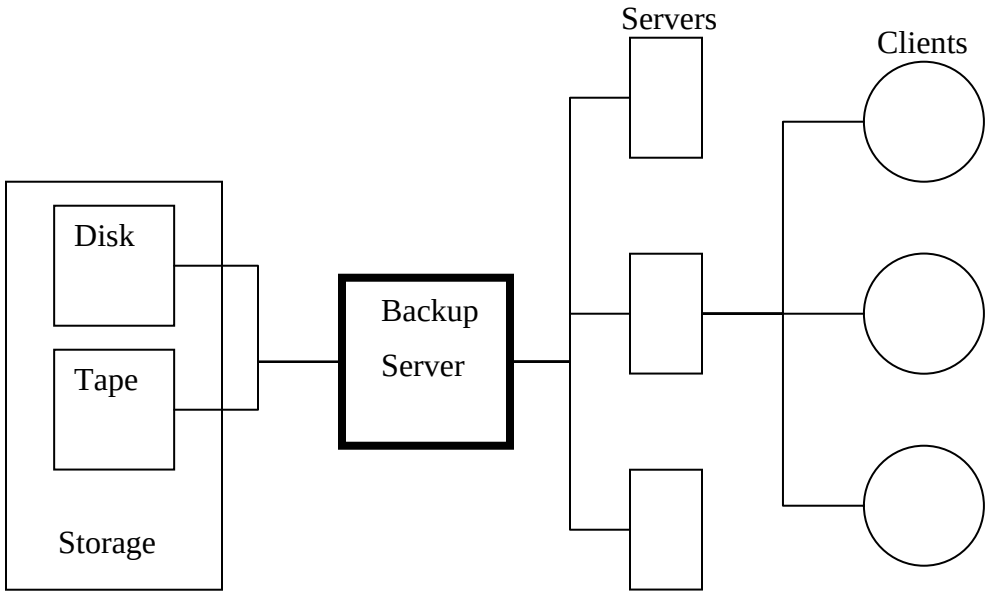


Fig 3.3.1: General backup architecture

The Back up server starts the backup process and sends request to servers to send their data[35]. The servers sends their data to backup server and that data is stored on device storage systems. When backup of all data is taken then backup server writes the catalog to a disk file and closes the connection to storage device.

Some businesses needs to consider about re-storage requirements, where that particular restore occurs, what are the most frequent restore requests, which data need backup, how often we need data backup (weekly, monthly), what is the time required to take the backup, how many copies of data to create. These requirements help one or the other way in bringing stability to backup plans.

Hot backup vs Cold Backup

Hot backup means applications are still running and users can access them without any interruption while backup process is going in background.
In C*old backup*, everything will be shut down during back up process.

There are three main topologies used for backups [36]:

1. Direct Attached based backup: backup data flows directly from host to be backed up to the tape, without using LAN. There is no central management and they do not help in growing the environment.

2. LAN based backup: backup data flows through LAN. There is central management but there might be some issues in LAN.

3. SAN based backup: also known as LAN free backup. There is no data movement over LAN instead data travels through SAN (storage area networks) to backup device.

3.3.2 Remote Replication

The process of reproducing data is called replication. Remote Replication means making a copy of the data (exact copy) called as “replica” that is available on other site. Replication can be used as a alternate source of backup and acts as a source of fast recovery. A true replica should be able to recover all data from the production volumes.

3.3.2.1 Remote Replication Types

Remote Replication has two types [37] :

- Synchronous Replication
- Asynchronous Replication

Synchronous Replication

It always ensures that data at both the sites is identical at all times. Initially server sends some data to write on source (primary) site. Data is committed at both source site and destination site before the data write is acknowledged. When source (primary) commits write , it will be transmitted to destination

(replica) site and when acknowledged by remote , source send ‘write complete’ signal to the host. One disadvantage of this replication is the longer response time.

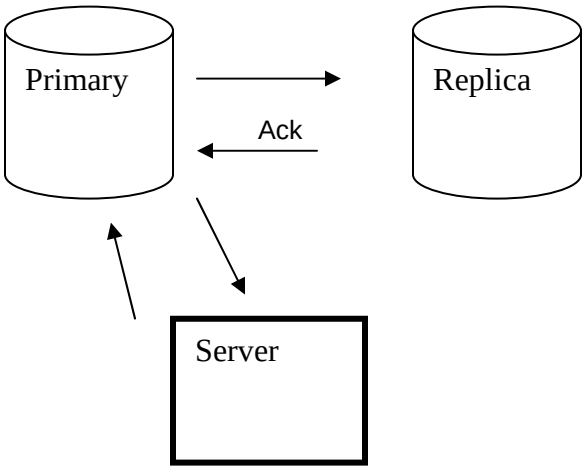


Fig 3.3.2.1(a) : Synchronous Replication

Asynchronous Replication

When data is committed by source site, it immediately send acknowledgement to host. Then Data is buffered and forwarded to remote site. The data at remote site will be behind the source site. Response time will be faster. They are normally deployed over extended distances. They also need sufficient buffers.

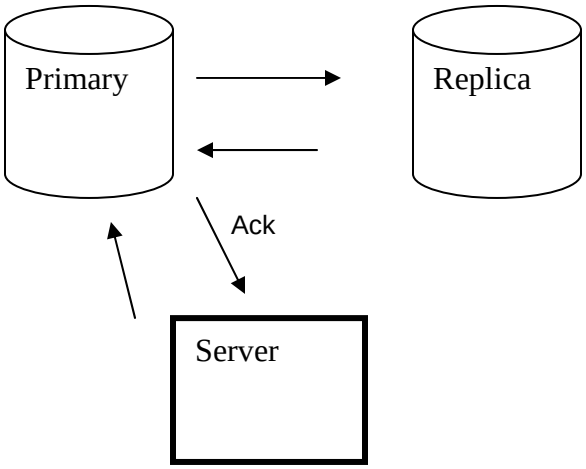


Fig 3.3.2.1(b): Asynchronous Replication

3.3.2.2 Remote Replication Technologies

There are two types of Remote Replication technologies used :

- Host based
- Storage Array based

Host based remote replication

There are duplicate volume groups are both source and remote sites. Each volume group consists of multiple disks [37].

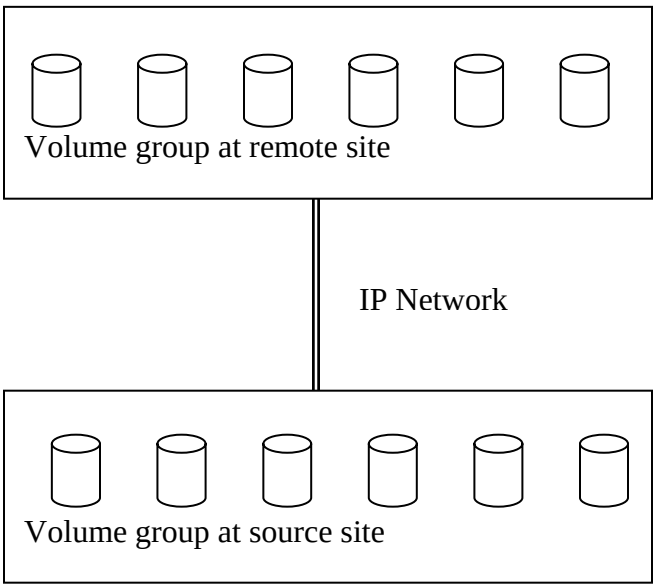


Fig 3.3.2.2 (a) : Host based remote replication

Normally, the writes are queued in a log file and sent to the remote site in the order received over the IP network.

They have significant advantage: storage arrays from different vendors can be used at two different sites. They use IP networks thus eliminating the need of dedicated networks. Also, this replication can be operated in synchronous and asynchronous modes too. They have low bandwidth requirement and offers less overhead on the CPU.

Array base Remote Replication :

In this type, replication is done by the array operating environment. When the source site sends a request for write to source array, it forwards the request to remote array via dedicated channels (Fibre channel, Gigabit Ethernet) over a shared network.

Only writes can be forwarded to the remote array. To protect the application from response time elongation, it can use asynchronous mode. Any write is acknowledged by the source as soon as it is received by the source array.

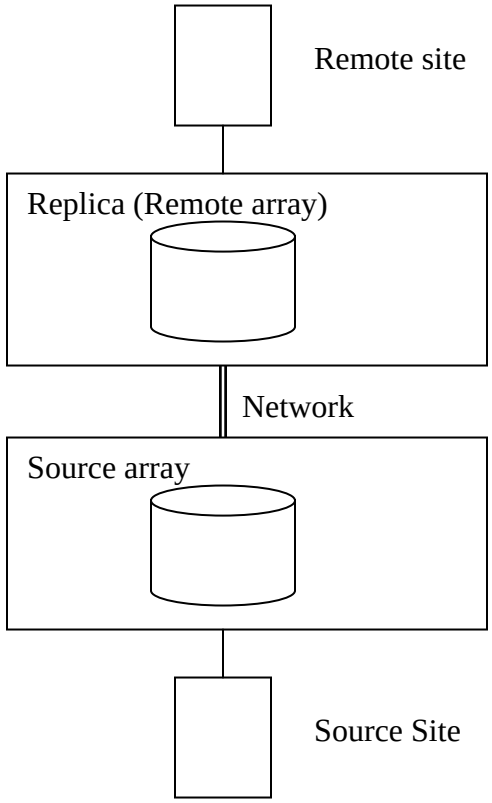


Fig 3.3.2.2 (b) : Array based remote replication

4. PROPOSED MODEL WITH PERFORMANCE ANALYSIS

4.1 Proposed Model

Our Aim is: To add fault-tolerant capability in our system so that it will continue to run even in case of failure and recover the lost information which was lost at the time of site crash.

To understand our proposed algorithm in detail, lets create some scenario. Suppose there are total X number of sites (computers) and Y data items in the system.
We are assuming that:

- Our system is based on wideband network.
- Any site in the system can have more than one data items.
- Data can move from one site to another.
- Every site is responsible for maintaining its own information and can only communicate through passing messages. There is no shared memory.

Each site in this system keeps information about other site by maintaining a matrix, "*position matrix (PM_)*". PM_ has the information about all other sites in the system such as which data other sites are having? It's the responsibility of each and every site to keep its PM_ updated and PM_ of other sites taking part in the communication.

Every data has a *"access counter (AC_)"*, attached with it which tell us that how many times that particular data is accessed and by whom. So before receiving any data item , every site checks AC_ and if the receiving site has access counter smaller than received access counter than it(receiving site) will update its access counter with a received one. Thus, every site try to keep latest information about data item. The higher the AC_, more latest the data item is.

Every data item has *"data relocated counter (DM_)"* associated with it so whenever data item moves from one site to other, it updates its DM_ by means of message passing with its target site. Following this way, we can always tell how many times that particular data has been moved to other site. There are various private data structures maintained by each site. So after handling the data items, it is the responsibility of each site participating in the communication to update other sites with the latest information structure so that all the sites will be having the reliable information about the network.

Every site maintains a queue of sites requesting for particular data item , *"global site queue (GSQ_)"*. So if any site (who is holding data and busy) receives multiple requests from different sites for the same data item then the receiving site puts the requesting site in the GSQ_. Thus, at any given time only one site can access the data item, insuring mutual exclusion.

When site is finished with its work then it allows the site present at the front of GSQ_ to access the data item. Now, consider if site at the front of GSQ_ wants to relocate the data then what about other remaining sites in the queue waiting for that data item? According to our proposed model, site at front of global message queue sends update message to every member of GSQ_ about data relocation of the data item they are waiting for.

We will explain ***Fault tolerance*** in our proposed algorithm with the help of an example.

Suppose there are five sites in the system: A, B, C, D, E

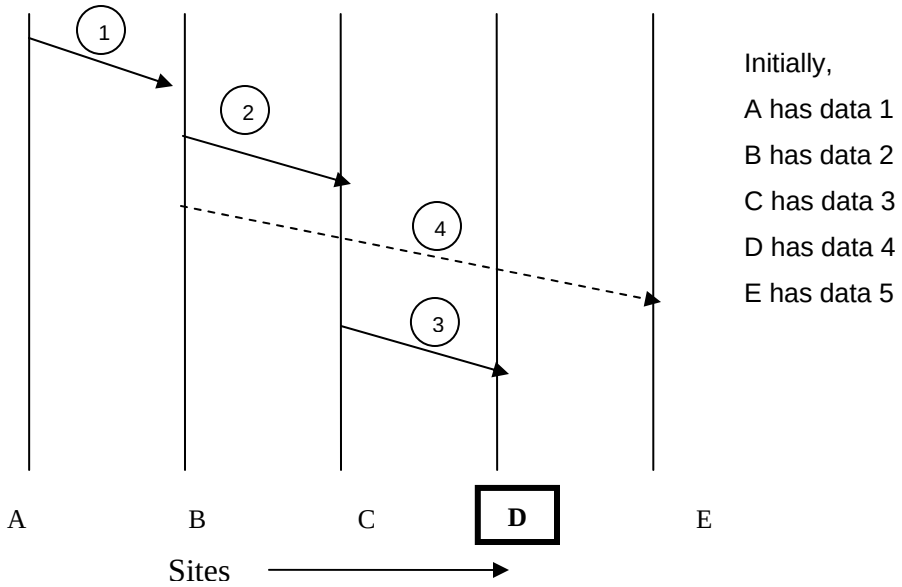
Initially site A has data 1, B has data 2, C has data 3, D has data 4, E has data 5.
Site A sends request message to site B for data 4 since A has information in its position matrix (PM_) that B has data 4 but actually data 4 resides on site D. B forwards the request to site C according to the information present in its PM_. C forwards it to D. Finally, D has data 4.

Also, in the mean while, B sends request for data 5 to site E.

At this moment, **site D crashes???**

If system is without fault tolerance feature then it may go down and site B will never receive data 5. But according to our proposed method, system will continue to execute other requests.

The crashed site (site D) will not take part in the communication and let others continue to do their work. The idea is, when D will restart after failure, its information before crashing should be recovered so D sends "RecoverMe" message to all. Having said that each site maintains information about other site, all sites try to recover site D by sending messages with whatever information they are having about D.
In fig 4.1, we have shown the above scenario:



(1) : Site A sends request for data 4 to site B .

(2) : Site B forward request to site C

(3) : Site C forward request to site D

(4) : Site B sends request for data 5 to site E

Fig 4.1 : Scenario where Site D crashes

4.2 Simulation and Results with Performance Analysis

In our simulation experiments, we have considered following parameters for obtaining results:

- Number of sites (X) = 32
  Number of data items (Y) =20
  [Values of X and Y can be changed.]
- Simulation Cycle Counter (**SCC**) varied from 1 to 15.
- Wait Cycle (**WC**) varied from 10 seconds to 3 minutes (180 seconds).
- Total Requests generated per Wait Cycle = 5000.

In this performance the comparison the following two performance measures are considered:

Requests Generated per wait cycle: the average number of requests generated per DC. Message complexity is measured by counting the number of communication hops in the network.

Total Time Delay per wait cycle: this is the total time taken by all sites in the system per wait cycle.

The interface of our proposed model consists of all the parameters mentioned above.
In our proposed method, we can crash more than one site per wait cycle.

Now, we will do the performance analysis by using graphical user interface designed for our proposed model. We will explain step by step procedure to obtain the simulation results.

Step 1: Launch the application by clicking the "prjFT.exe" file. Following screen will appear.
In this screen, we can enter the number of sites we want to crash and their respective site numbers. This system is INITIAL state (before running simulation).

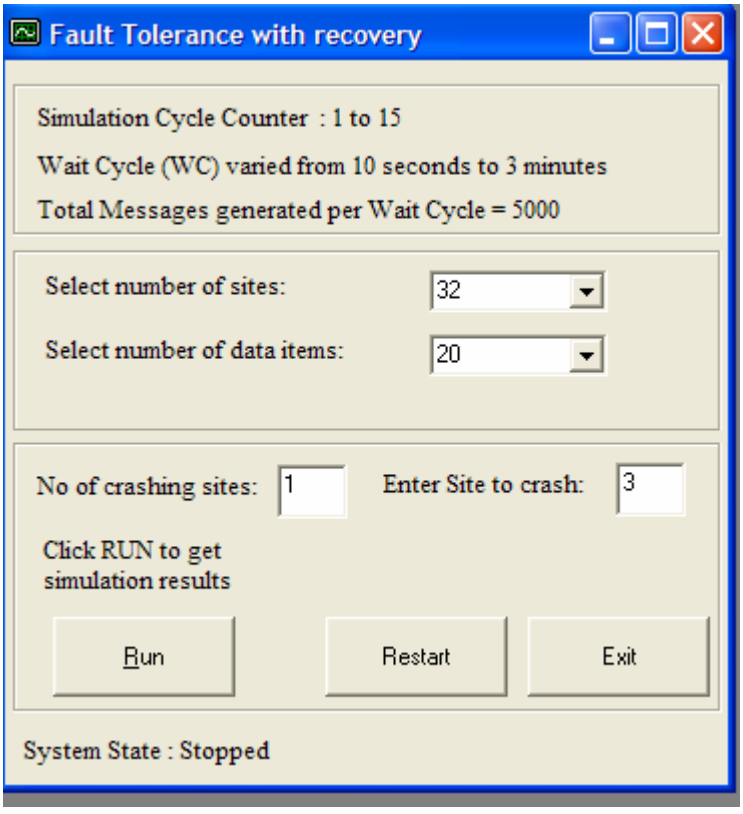


Fig 4.2 : Initial state with stopped system state

Step 2: press RUN to start the simulation. The system state at the bottom will be changed to RUNNING and also display the status of log file saving our simulation results.

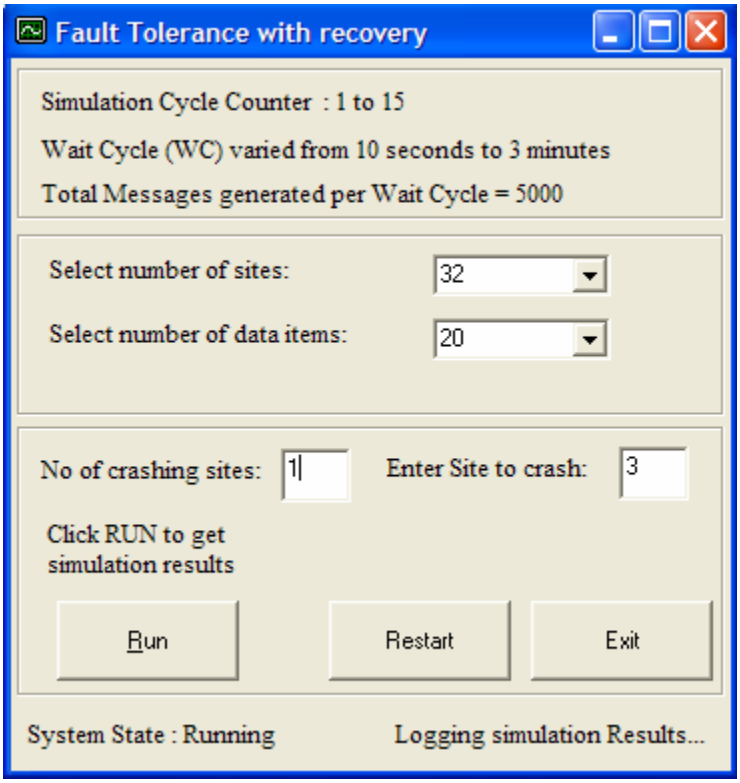


Fig 4.3 Running state

Step 3: When entered site will crash it will display system critical warning message. But our system state will not change and it will continue to run in spite of failure. In our performance model, we have repeated the site failure for every wait cycle and then taken the results. Our system runs successfully for large number of sites (X= 64, X=128) and large number of data items (Y=35, Y=50) too .

The crashed site will receive any incoming requests. Other sites keep on sending message to crashed till timeout.

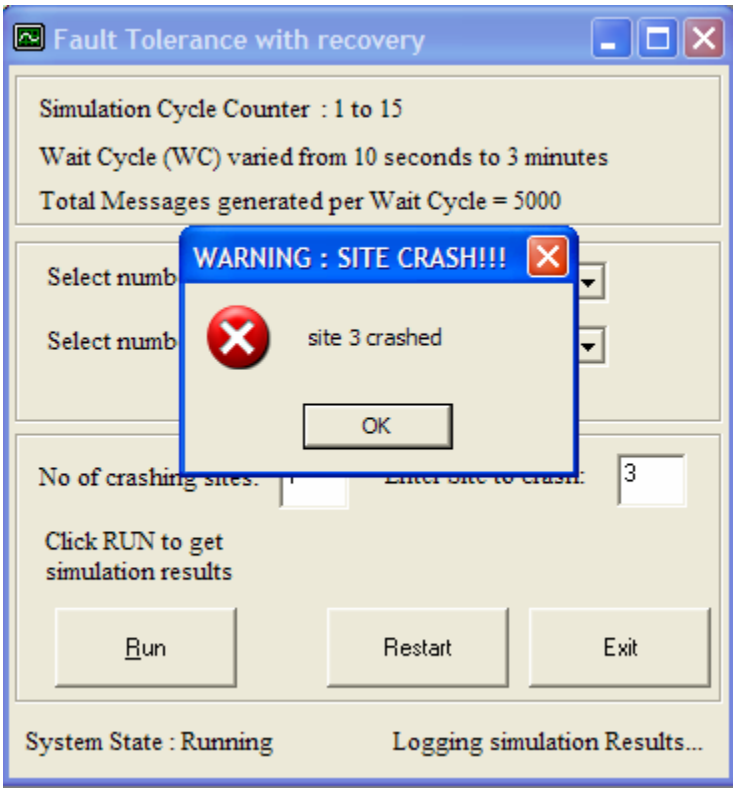


Fig 4.4: site 3 crashed

Case 4: When a site recovers from failure, it sends "RecoverMe" message to all to update its position matrix (PM_). This is time when other sites came to know that this particular site had been crashed and they will recover it by sending update messages. *We are not recovering the data item a crashed site was holding before crash still we can locate the missing data.*

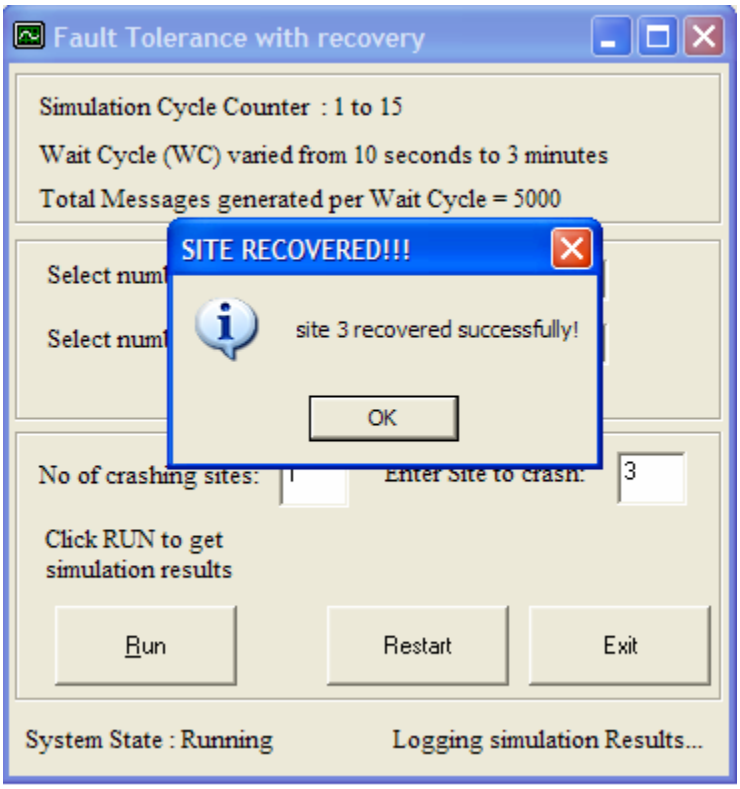


Fig 4.5 Site Recovered

Simulation results:

SCC = 1 , WC= 10.000000 , Total Time Delay per WC = 291 , MSG per WC = 3.898200
SCC = 1 , WC= 20.000000 , Total Time Delay per WC = 302 , MSG per WC = 3.909800
SCC = 1 , WC= 30.000000 , Total Time Delay per WC = 311 , MSG per WC = 3.894000
SCC = 1 , WC= 40.000000 , Total Time Delay per WC = 312 , MSG per WC = 3.905400
SCC = 1 , WC= 50.000000 , Total Time Delay per WC = 312 , MSG per WC = 3.900600
SCC = 1 , WC= 60.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.906400
SCC = 1 , WC= 70.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.913600
SCC = 1 , WC= 80.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.909200
SCC = 1 , WC= 90.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.900600
SCC = 1 , WC= 100.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.896400
SCC = 1 , WC= 110.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.890800
SCC = 1 , WC= 120.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.898800
SCC = 1 , WC= 130.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.905200
SCC = 1 , WC= 140.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.902600
SCC = 1 , WC= 150.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.900800
SCC = 1 , WC= 160.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.895000
SCC = 1 , WC= 170.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.902600
SCC = 1 , WC= 180.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.894800
SCC = 2 , WC= 10.000000 , Total Time Delay per WC = 296 , MSG per WC = 3.907200
SCC = 2 , WC= 20.000000 , Total Time Delay per WC = 306 , MSG per WC = 3.905400
SCC = 2 , WC= 30.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.909000
SCC = 2 , WC= 40.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.907400
SCC = 2 , WC= 50.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.915600
SCC = 2 , WC= 60.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.897200
SCC = 2 , WC= 70.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.901400
SCC = 2 , WC= 80.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.889200
SCC = 2 , WC= 90.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.892600
SCC = 2 , WC= 100.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.900400
SCC = 2 , WC= 110.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.896000
SCC = 2 , WC= 120.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.908400
SCC = 2 , WC= 130.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.901000
SCC = 2 , WC= 140.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.904200
SCC = 2 , WC= 150.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.906600

SCC = 2 , WC= 160.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.900600
SCC = 2 , WC= 170.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.904600
SCC = 2 , WC= 180.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.899000
SCC = 3 , WC= 10.000000 , Total Time Delay per WC = 288 , MSG per WC = 3.886800
SCC = 3 , WC= 20.000000 , Total Time Delay per WC = 303 , MSG per WC = 3.900200
SCC = 3 , WC= 30.000000 , Total Time Delay per WC = 311 , MSG per WC = 3.903400
SCC = 3 , WC= 40.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.906800
SCC = 3 , WC= 50.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.892200
SCC = 3 , WC= 60.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.913400
SCC = 3 , WC= 70.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.921800
SCC = 3 , WC= 80.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.893400
SCC = 3 , WC= 90.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.910000
SCC = 3 , WC= 100.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.895000
SCC = 3 , WC= 110.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.901200
SCC = 3 , WC= 120.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.904400
SCC = 3 , WC= 130.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.910200
SCC = 3 , WC= 140.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.912200
SCC = 3 , WC= 150.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.915000
SCC = 3 , WC= 160.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.905000
SCC = 3 , WC= 170.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.916000
SCC = 3 , WC= 180.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.898400
SCC = 4 , WC= 10.000000 , Total Time Delay per WC = 294 , MSG per WC = 3.896600
SCC = 4 , WC= 20.000000 , Total Time Delay per WC = 304 , MSG per WC = 3.887600
SCC = 4 , WC= 30.000000 , Total Time Delay per WC = 306 , MSG per WC = 3.884200
SCC = 4 , WC= 40.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.904400
SCC = 4 , WC= 50.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.904600
SCC = 4 , WC= 60.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.905600
SCC = 4 , WC= 70.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.897600
SCC = 4 , WC= 80.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.897400
SCC = 4 , WC= 90.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.891200
SCC = 4 , WC= 100.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.898800
SCC = 4 , WC= 110.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.893400
SCC = 4 , WC= 120.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.904400
SCC = 4 , WC= 130.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.903400
SCC = 4 , WC= 140.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.905800
SCC = 4 , WC= 150.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.906600
SCC = 4 , WC= 160.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.905800
SCC = 4 , WC= 170.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.894200
SCC = 4 , WC= 180.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.899000
SCC = 5 , WC= 10.000000 , Total Time Delay per WC = 291 , MSG per WC = 3.911400
SCC = 5 , WC= 20.000000 , Total Time Delay per WC = 307 , MSG per WC = 3.896600
SCC = 5 , WC= 30.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.904000
SCC = 5 , WC= 40.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.904200
SCC = 5 , WC= 50.000000 , Total Time Delay per WC = 312 , MSG per WC = 3.892400
SCC = 5 , WC= 60.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.904400
SCC = 5 , WC= 70.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.911800
SCC = 5 , WC= 80.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.900800
SCC = 5 , WC= 90.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.918000
SCC = 5 , WC= 100.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.906200
SCC = 5 , WC= 110.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.904400
SCC = 5 , WC= 120.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.897000
SCC = 5 , WC= 130.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.915200
SCC = 5 , WC= 140.000000 , Total Time Delay per WC = 321 , MSG per WC = 3.911600
SCC = 5 , WC= 150.000000 , Total Time Delay per WC = 321 , MSG per WC = 3.900400
SCC = 5 , WC= 160.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.900200
SCC = 5 , WC= 170.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.903200
SCC = 5 , WC= 180.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.907200
SCC = 6 , WC= 10.000000 , Total Time Delay per WC = 293 , MSG per WC = 3.898200
SCC = 6 , WC= 20.000000 , Total Time Delay per WC = 304 , MSG per WC = 3.903800
SCC = 6 , WC= 30.000000 , Total Time Delay per WC = 307 , MSG per WC = 3.895800
SCC = 6 , WC= 40.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.903200
SCC = 6 , WC= 50.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.902000
SCC = 6 , WC= 60.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.917000
SCC = 6 , WC= 70.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.908800
SCC = 6 , WC= 80.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.914800
SCC = 6 , WC= 90.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.898600
SCC = 6 , WC= 100.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.876800

SCC = 6 , WC= 110.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.915400
SCC = 6 , WC= 120.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.902800
SCC = 6 , WC= 130.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.895600
SCC = 6 , WC= 140.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.903200
SCC = 6 , WC= 150.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.889200
SCC = 6 , WC= 160.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.905200
SCC = 6 , WC= 170.000000 , Total Time Delay per WC = 321 , MSG per WC = 3.908600
SCC = 6 , WC= 180.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.897000
SCC = 7 , WC= 10.000000 , Total Time Delay per WC = 289 , MSG per WC = 3.909800
SCC = 7 , WC= 20.000000 , Total Time Delay per WC = 305 , MSG per WC = 3.912200
SCC = 7 , WC= 30.000000 , Total Time Delay per WC = 309 , MSG per WC = 3.901200
SCC = 7 , WC= 40.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.917400
SCC = 7 , WC= 50.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.910800
SCC = 7 , WC= 60.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.891800
SCC = 7 , WC= 70.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.902600
SCC = 7 , WC= 80.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.887600
SCC = 7 , WC= 90.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.905200
SCC = 7 , WC= 100.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.906200
SCC = 7 , WC= 110.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.914000
SCC = 7 , WC= 120.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.909600
SCC = 7 , WC= 130.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.900800
SCC = 7 , WC= 140.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.907400
SCC = 7 , WC= 150.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.901800
SCC = 7 , WC= 160.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.922000
SCC = 7 , WC= 170.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.915600
SCC = 7 , WC= 180.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.897600
SCC = 8 , WC= 10.000000 , Total Time Delay per WC = 301 , MSG per WC = 3.917600
SCC = 8 , WC= 20.000000 , Total Time Delay per WC = 303 , MSG per WC = 3.901800
SCC = 8 , WC= 30.000000 , Total Time Delay per WC = 309 , MSG per WC = 3.890000
SCC = 8 , WC= 40.000000 , Total Time Delay per WC = 311 , MSG per WC = 3.895400
SCC = 8 , WC= 50.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.909600
SCC = 8 , WC= 60.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.880400
SCC = 8 , WC= 70.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.889800
SCC = 8 , WC= 80.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.890400
SCC = 8 , WC= 90.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.896200
SCC = 8 , WC= 100.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.908800
SCC = 8 , WC= 110.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.892800
SCC = 8 , WC= 120.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.907200
SCC = 8 , WC= 130.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.899000
SCC = 8 , WC= 140.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.900800
SCC = 8 , WC= 150.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.881600
SCC = 8 , WC= 160.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.915000
SCC = 8 , WC= 170.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.899600
SCC = 8 , WC= 180.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.901800
SCC = 9 , WC= 10.000000 , Total Time Delay per WC = 292 , MSG per WC = 3.905800
SCC = 9 , WC= 20.000000 , Total Time Delay per WC = 307 , MSG per WC = 3.895200
SCC = 9 , WC= 30.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.891800
SCC = 9 , WC= 40.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.902200
SCC = 9 , WC= 50.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.900600
SCC = 9 , WC= 60.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.897600
SCC = 9 , WC= 70.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.897800
SCC = 9 , WC= 80.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.897600
SCC = 9 , WC= 90.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.904400
SCC = 9 , WC= 100.000000 , Total Time Delay per WC = 299 , MSG per WC = 3.896800
SCC = 9 , WC= 110.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.898000
SCC = 9 , WC= 120.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.896000
SCC = 9 , WC= 130.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.901000
SCC = 9 , WC= 140.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.909600
SCC = 9 , WC= 150.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.917800
SCC = 9 , WC= 160.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.901200
SCC = 9 , WC= 170.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.909400
SCC = 9 , WC= 180.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.903200
SCC = 10 , WC= 10.000000 , Total Time Delay per WC = 295 , MSG per WC = 3.912600
SCC = 10 , WC= 20.000000 , Total Time Delay per WC = 303 , MSG per WC = 3.895600
SCC = 10 , WC= 30.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.906200
SCC = 10 , WC= 40.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.923600
SCC = 10 , WC= 50.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.902400

SCC = 10 , WC= 60.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.891200
SCC = 10 , WC= 70.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.912400
SCC = 10 , WC= 80.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.902600
SCC = 10 , WC= 90.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.913600
SCC = 10 , WC= 100.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.904200
SCC = 10 , WC= 110.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.895400
SCC = 10 , WC= 120.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.908000
SCC = 10 , WC= 130.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.900400
SCC = 10 , WC= 140.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.912200
SCC = 10 , WC= 150.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.912600
SCC = 10 , WC= 160.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.900800
SCC = 10 , WC= 170.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.905600
SCC = 10 , WC= 180.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.897000
SCC = 11 , WC= 10.000000 , Total Time Delay per WC = 293 , MSG per WC = 3.904200
SCC = 11 , WC= 20.000000 , Total Time Delay per WC = 308 , MSG per WC = 3.914600
SCC = 11 , WC= 30.000000 , Total Time Delay per WC = 308 , MSG per WC = 3.897200
SCC = 11 , WC= 40.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.909600
SCC = 11 , WC= 50.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.916200
SCC = 11 , WC= 60.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.907200
SCC = 11 , WC= 70.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.890600
SCC = 11 , WC= 80.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.907200
SCC = 11 , WC= 90.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.911000
SCC = 11 , WC= 100.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.897800
SCC = 11 , WC= 110.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.899800
SCC = 11 , WC= 120.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.906000
SCC = 11 , WC= 130.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.905600
SCC = 11 , WC= 140.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.895800
SCC = 11 , WC= 150.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.900600
SCC = 11 , WC= 160.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.899800
SCC = 11 , WC= 170.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.894800
SCC = 11 , WC= 180.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.907200
SCC = 12 , WC= 10.000000 , Total Time Delay per WC = 290 , MSG per WC = 3.899400
SCC = 12 , WC= 20.000000 , Total Time Delay per WC = 309 , MSG per WC = 3.911200
SCC = 12 , WC= 30.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.906600
SCC = 12 , WC= 40.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.905400
SCC = 12 , WC= 50.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.894800
SCC = 12 , WC= 60.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.902000
SCC = 12 , WC= 70.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.907000
SCC = 12 , WC= 80.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.890600
SCC = 12 , WC= 90.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.902800
SCC = 12 , WC= 100.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.913000
SCC = 12 , WC= 110.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.917600
SCC = 12 , WC= 120.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.894800
SCC = 12 , WC= 130.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.908600
SCC = 12 , WC= 140.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.900800
SCC = 12 , WC= 150.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.910400
SCC = 12 , WC= 160.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.899600
SCC = 12 , WC= 170.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.900000
SCC = 12 , WC= 180.000000 , Total Time Delay per WC = 321 , MSG per WC = 3.908000
SCC = 13 , WC= 10.000000 , Total Time Delay per WC = 290 , MSG per WC = 3.899600
SCC = 13 , WC= 20.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.923200
SCC = 13 , WC= 30.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.904600
SCC = 13 , WC= 40.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.906600
SCC = 13 , WC= 50.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.889400
SCC = 13 , WC= 60.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.921200
SCC = 13 , WC= 70.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.908800
SCC = 13 , WC= 80.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.906600
SCC = 13 , WC= 90.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.899000
SCC = 13 , WC= 100.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.907400
SCC = 13 , WC= 110.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.903400
SCC = 13 , WC= 120.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.907400
SCC = 13 , WC= 130.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.893600
SCC = 13 , WC= 140.000000 , Total Time Delay per WC = 319 , MSG per WC = 3.904000
SCC = 13 , WC= 150.000000 , Total Time Delay per WC = 321 , MSG per WC = 3.915000
SCC = 13 , WC= 160.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.907400
SCC = 13 , WC= 170.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.905800
SCC = 13 , WC= 180.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.911000

SCC = 14 , WC= 10.000000 , Total Time Delay per WC = 292 , MSG per WC = 3.912600
SCC = 14 , WC= 20.000000 , Total Time Delay per WC = 305 , MSG per WC = 3.910000
SCC = 14 , WC= 30.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.912800
SCC = 14 , WC= 40.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.901600
SCC = 14 , WC= 50.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.904800
SCC = 14 , WC= 60.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.900800
SCC = 14 , WC= 70.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.908600
SCC = 14 , WC= 80.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.905400
SCC = 14 , WC= 90.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.909000
SCC = 14 , WC= 100.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.908800
SCC = 14 , WC= 110.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.905000
SCC = 14 , WC= 120.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.888600
SCC = 14 , WC= 130.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.904600
SCC = 14 , WC= 140.000000 , Total Time Delay per WC = 317 , MSG per WC = 3.900600
SCC = 14 , WC= 150.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.909000
SCC = 14 , WC= 160.000000 , Total Time Delay per WC = 320 , MSG per WC = 3.903600
SCC = 14 , WC= 170.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.909400
SCC = 14 , WC= 180.000000 , Total Time Delay per WC = 321 , MSG per WC = 3.911600
SCC = 15 , WC= 10.000000 , Total Time Delay per WC = 292 , MSG per WC = 3.912000
SCC = 15 , WC= 20.000000 , Total Time Delay per WC = 313 , MSG per WC = 3.913200
SCC = 15 , WC= 30.000000 , Total Time Delay per WC = 310 , MSG per WC = 3.908400
SCC = 15 , WC= 40.000000 , Total Time Delay per WC = 312 , MSG per WC = 3.906000
SCC = 15 , WC= 50.000000 , Total Time Delay per WC = 314 , MSG per WC = 3.898800
SCC = 15 , WC= 60.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.905000
SCC = 15 , WC= 70.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.901800
SCC = 15 , WC= 80.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.902400
SCC = 15 , WC= 90.000000 , Total Time Delay per WC = 315 , MSG per WC = 3.905800
SCC = 15 , WC= 100.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.892000
SCC = 15 , WC= 110.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.907400
SCC = 15 , WC= 120.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.896400
SCC = 15 , WC= 130.000000 , Total Time Delay per WC = 316 , MSG per WC = 3.896400
SCC = 15 , WC= 140.000000 , Total Time Delay per WC = 322 , MSG per WC = 3.921800
SCC = 15 , WC= 150.000000 , Total Time Delay per WC = 321 , MSG per WC = 3.907200
SCC = 15 , WC= 160.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.901400
SCC = 15 , WC= 170.000000 , Total Time Delay per WC = 318 , MSG per WC = 3.896600
SCC = 15 , WC= 180.000000 , Total Time Delay per WC = 321 , MSG per WC = 3.914600

5. CONCLUSION AND FUTURE WORK

Conclusion:

In this dissertation, we have proposed a fault tolerant method for site recovery in wideband networks.
In our proposed model,there are several sites and large number of data items which keeps on moving from one location to another. Communication in the system is done by means of message passing.

As we are aware that *Fault tolerance* is the ability of a system to perform its function correctly even in the presence of internal faults. Strictly speaking ,In general, fault tolerance follows some basic procedures like *Error detection, damage confinement,* error recovery, fault treatment. *Error detection* is the process of identifying that the system is in an invalid state. This means that some component in the system has failed. To ensure that the effects of the error are limited, it is necessary to isolate the failed component so that its effects are not propagated further. This is known as *damage confinement.* In the error recovery phase, the error and more importantly its effects, are removed by restoring the system to a valid state. Finally, in *fault treatment,* we go after the fault that caused the error so that it can be isolated.

Lets discuss some important points covered in this disssertation about our proposed algorithm. We are not randomly crashing the site in the system. We are assuming that reason for site crash is hardware failure. We explicitly enter the site to crash. Suppose the site A was holding some data d before crash and site A was executing that data. At the time of crashing, site state was executing. So we will recover state of site A. Also, since there are multiple data items in the system and one site can have more than one data. Therefore, we are maintaining a queue for the sites who wants to access the same data item. The sites at the front of queue will be serviced first and other requesting site will be added from read end.In this way, only one site can access the data at any one time, thus maintaining the data consistency.

We are also keeping track of howmany times a particular data item has been requested. We are maintaing a counter for every data item that will be incremented each time when request for that data item is generated. Every site updates its formation about this counter so that it will have the information how many time this data item has been requested.

Since data can be moved from one site to another so we are also keeping track of how many times that data item is moved from its parent location. Having said that every site keeps itself updated about all the

data items moving in the system. Every site knows where is that particular data item. Suppose any site sends request for a particular data itme to target site. If target site does not have data then it forwards the request. Any crashed site will be recovered with all this information discussed above.
We are not recovering the data item instead the crashed site who has recovered from failure can find out the data item it was holding at the time of crashing.

Our focus is to increase the dependability of a system. Dependability means that our system can be trusted to perform the service for which it has been designed. Dependability can be decomposed into specific aspects. *Reliability* characterizes the ability of a system to perform its service correctly when asked to do so. *Availability* means that the system is available to perform this service when it is asked to do so.

The fact that the processing sites of a system are independent of each other means that they are independent points of failure. Another kind of failure that is inherent in most networks comes from the communication medium. In the most severe cases, this type of failure can lead to partitioning of the system into multiple parts that are completely isolated from each other. While transmission delays are not necessarily failures, they can certainly lead to failures. A delay can be misconstrued as a message loss.

Data should be recovered on networks. For this purpose, we have explained backup and recovery methods on networks. The Network storage enables more effective utilization of available storage, increased flexibility, and better availability of data. The main purpose of storage systems is to provide storage to one or more computers. These systems are responsible for actual transmission of data and to read or write data. There are some models discussed on network storage like Direct Attached Storage (DAS), Networked Attached Storage (NAS), Fibre Channel Storage Area Networks (FC SAN), IP Storage Area Networks (IP SAN).

To make the system fault tolerant one way is to prevent the fault. So we need to take the back up of our data .Back up means taking copy of the online data that resides on primary storage. Operational backup is needed only when data is lost or corrupted. Archival backup is used to save transaction records, business work products for regulatory compliance. In case of recovery from Disaster we have to recover all data. For e.g., a company has lost all its data in major disaster then all data needs to be recovered from backup copies. The process of reproducing data is called replication. Remote Replication means

making a copy of the data (exact copy) called as “replica” that is available on other site. Replication can be used as a alternate source of backup and acts as a source of fast recovery

The proposed method allows us to recover multiple sites in a fault tolerant manner i.e. if any site crashes in the system then it will not let the system down instead it will stop accepting incoming requests. When crashed site recovers from failure, then it sends recover message to all other sites in the system and other sites also cooperate in recovering the crashed site to its state it had before crash.

We have carried out performance tests where we will calculate messages and total time delay per wait cycle. The system is running on large number of sites (64 to 128) and large number of data iems (20 to 50). There are 5000 requesting genrated in every cycle.

Future Work:

We are not recovering the data item which crashed site was holding before crash but we can locate the missing data. So in future, with a system having several sites and large data items, we can direct our work in recovering the data item. We have considered wideband networks for our system. The future work can be targeted toward using different networks like gigabit Ethernet. We can design a system for wireless networks too as in our proposed method we are using connection oriented networks.

6. REFERENCES

[1] Behrouz A. Forouzan : Data communications and Networking, second edition, Tata Mcgraw Hill edition,2003.

[2] Halsall,Fred. : Data communications, Computer Networks and Open systems, Fourth edition,MA:Addison-Wesley,1995.

[3] Andrew S. Tanenbaum: Computer Networks, Fourth edition, Printice Hall.

[4] http://telecom.tbi.net/switching.htm

[5] William Stallings : Isdn and Broadband Isdn , MacMillan Publishing Company,1991.

[6] G.B. Bleazard : Why Packet Switching?, Blackwell Publishers, sep 1979

[7] Switching Networks - by Ding-Zhu Du, Hung Q Ngo ; Kluwer Academic publications

[8] Andrew S. Tanenbaum, Albert S. Woodhull : Operating Systems Design and Implementation, Second Edition, Pearson Education, 2004.

[9] Mukash Singhal, Niranjan Shivaratri : Advanced concepts in operating systems. Distributed, database and multiprocessor operating systems. TATA Mc. Hill, 2001

[10] Silberchatz and Galvin : Operating System Concepts, Fifth Edition.

[11] Lamport L.: How to make a Multiprocessor Computer that Correctly Executes Multiprocessor Programs. IEEE Transactions on Computers, C-28(9):690-691, 1997.

[12] http://www-128.ibm.com/developerworks/rational/library/114.html : Fault Tolerance

[13] pcl.cs.ucla.edu/projects/parsec

[14] Intelligent Broadband Multimedia Networks - by Syed V Ahamed, Syed V Aham : Springer publications.

[15] http://www.axis.com/documentation/whitepaper/ip_networks_basics.htm : IP networks

[16] www.comptia.org/sections/ssg/glossary.aspx : frame relay

[17] http://en.wikipedia.org/wiki/

[18]www.fortfrancesbroadband.ca/terms.htm http://www.computerhope.com/jargon/w/wideband.htm: wideband networks :

[19] http://www.dothill.com/tutorial/tutorial.swf : SAN introduction

[20] Charp, S. (Ed.). (1994). Networking & telecommunications. "T.H.E." ("Technical Horizons in Education"), 21(10). (EJ 483 802-807) : LAN

[21] http://www.byte.com/art/9709/sec4/art3.htm : ATM networks

[22] Pankaj Jalote : Fault Tolerance in Distributed Systems

[23] Software Fault Tolerance: Achievement and Assessment Strategies (Research Reports Espirit. Project 300. Request, Vol 1) by M.Kersken(Editor), F.Saglietti (Editor).

[24] Software Fault Tolerance Techniques and Implementation by Laura L. Pullum

[25] Fault Tolerance and Reliability Techniques for High-Density Random-Access Memories by Kanad Charkaborty, Prentice Hall PTR, June 2002.

[26] http://www.eventhelix.com/RealtimeMantra/FaultHandling/

[27] www.csce.uark.edu/~aapon/courses/ ioparallel/presentations/raid88.ppt : RAID

[28] http://www.iscsistorage.com/ipstorage.htm

[29] http://searchstorage.techtarget.com/tip/1,289483,sid5_gci790308,00.html
http://searchstorage.techtarget.com/tip/1,289483,sid5_gci937145,00.html:

[30] http://www.javvin.com/protocolFCIP.html : FCIP

[31] http://www.networkworld.com/details/712.html

[32] http://www.expresscomputeronline.com/20051212/technology05.shtml : FC SAN

[33] Recovery Techniques for Database Systems; Joost S.M. Verhofstad BNSR, 522 University Avenue, Toronto, Ontario M5G 1W7 Canada.

[34] http://www.vantagetech.com/ ; http://www.drivelabs.us/: Data Recovery

[35] The Disaster Recovery Handbook: A Step-By-Step Plan to Ensure Business Continuity and Protect Vital Operations, Facilities, and Assets by Michael Wallace, Lawrence Webber.

[36]http://network.jacklewis.net/articles/Disk-basedbackupisLAN-basedorSAN-basedthefairestmirrorofthemall-Connectivity.html

[37]http://www.wwpi.com/index.php?option=com_content&task=view&id=446&Itemid=44