

**Principles of Engineering System Design**  
**Dr. T. Asokan**  
**Department of Engineering Design**  
**Indian Institute of Technology, Madras**

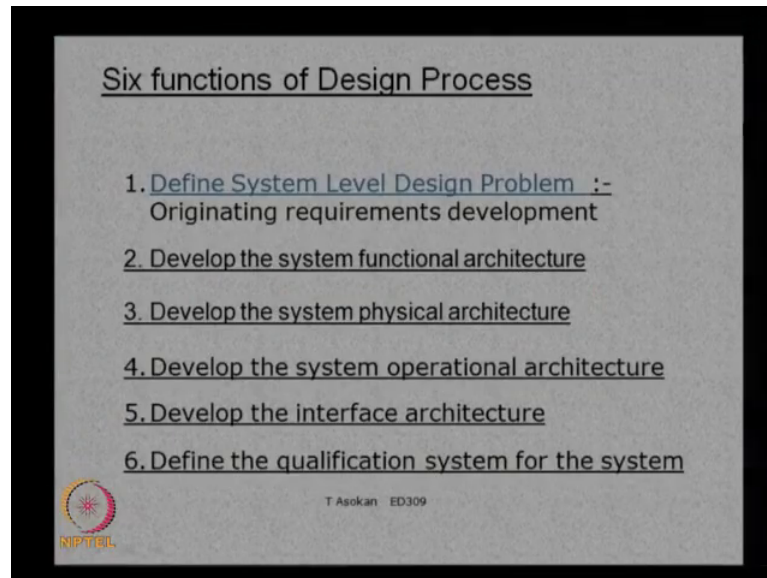
**Lecture – 16**  
**Interface Architecture Development**

Dear friends, welcome back to another session on systems engineering. So, in this lecture we will focus on the interface a design of engineering systems. As you know every system has got many sub systems and sub systems components and then this CIs or the configuration items, so this the system need to interact with the external systems as well as within the system we need to have interaction between subsystems and components and configuration items.

So, these interactions are taking place through the interfaces. So, we can have various interfaces with in the system. You can have a mechanically interface, you can have an electrical interface or you can have a communication interface. So, all these interfaces need to be provided in such a with that we get proper communication we get have an integral communication the integrity of the message is not lost or there is proper passing of messages between the systems and there is proper priority for systems to interact with other systems and communicate messages. So, all these things are basically design using the interfaces. So, interface design becomes one of them most important aspect of engineering system were we ensure that the communication between various systems are proper and they do not lead to failures within the system.

If you look at the history of engineering systems you can see there are many failures of engineering system because of communication failure or the interface failure within the system. We will see one example like that to show the important of interface design and then we discuss about the various kind of the interface to be provider, but are the different properties for this interfaces and what are the different standards to be used for interface design.

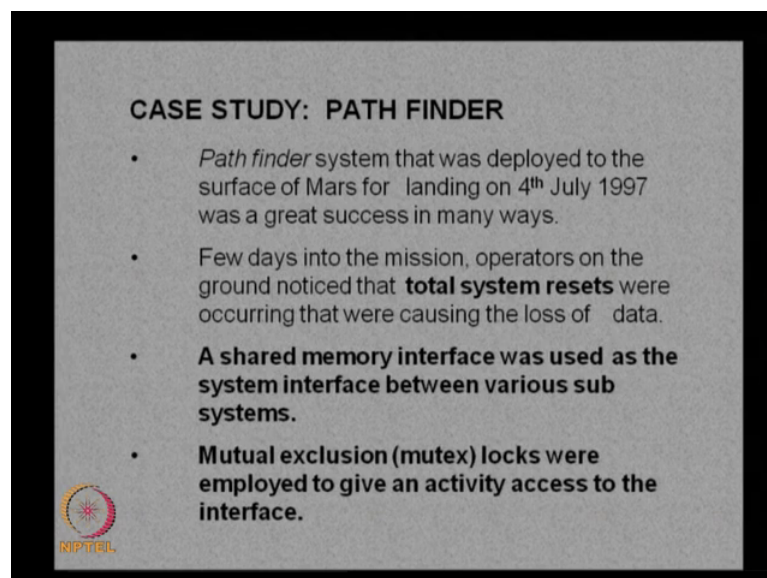
(Refer Slide Time: 02:14)



So, as I mentioned earlier this is the fifth step in the engineering system design where we discussed about the first four stages basically the system level design problem, system function architecture, physical architecture and the operation architecture. So, we completed all these four and the next one is the interface architecture development.

So, as I mentioned earlier will take a case study from the history of engineering system failures and then see how important is the interfaces in the success of an engineering system.

(Refer Slide Time: 02:48)

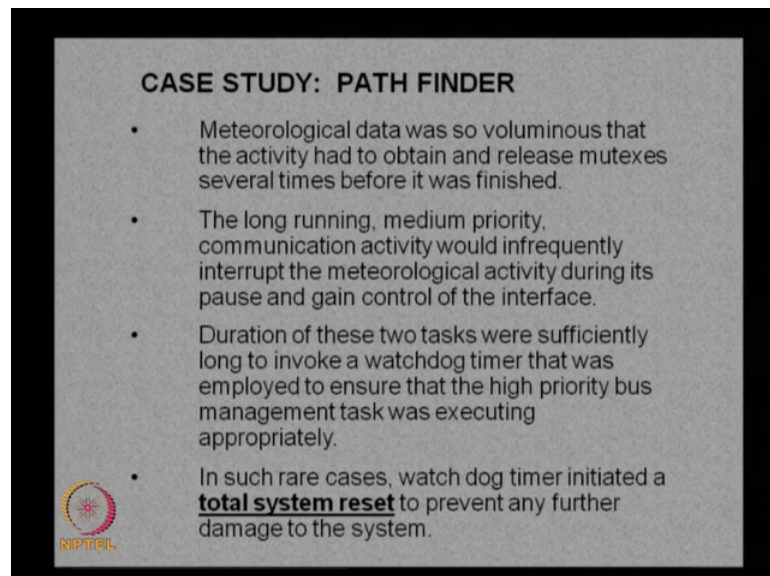


This case studies from the a path finder which was deployed to the surface of a Mars for landing on 4th July 1997 it was a great success in many ways.

So, all there was to the tremendous success from this particular project, but there was an issue when it was initially launched and then it was being used it was found that the total system resets were taking place once in a while. That is few days into the machine operators on the ground realized that there is a total system reset taking place within the system without any particular reason or without them able to identify what actually was reason for this and then an analysis was scared out. And then it was found that there was a shared memory interface which was used as the system interface between various subsystems. So, there are many subsystems with in the path finder and there was in architecture called shared memory interface was used for interfacing this subsystems. And there was a mutex or mutual exclusion lock were employed to give an activity access to the interface.


So, whenever a particular subsystems wants to access these particular interface they need to use a mutex, the mutually exclusion locks and that is to be activated note to get the access to a communication interface.

(Refer Slide Time: 04:11)



**CASE STUDY: PATH FINDER**

- Meteorological data was so voluminous that the activity had to obtain and release mutexes several times before it was finished.
- The long running, medium priority, communication activity would infrequently interrupt the meteorological activity during its pause and gain control of the interface.
- Duration of these two tasks were sufficiently long to invoke a watchdog timer that was employed to ensure that the high priority bus management task was executing appropriately.
- In such rare cases, watch dog timer initiated a **total system reset** to prevent any further damage to the system.



So, this was one of the problem was actually lying in this interface. So, there were many subsystem in the path finder and one of the system was meteorological data or it is actually collection of data from various sensors and passing it to the main system. So,

there this data was. So, voluminous that the activity had to obtain and release mutexes several times before it was finished.

So, it was a very voluminous data and there was the time duration for transferring of this data was very a long therefore, this particular subsystem had to release the mutex many times because of the long the voluminous data and then before completing a particular transactions. And at the same time there was another process the long running medium priority communication activity. So, this was a medium priority activity the communication activity would in frequently interrupt the metrological activity during its pause and gain control of the interface.

So, this was what does happening. So, the metrological data was being a transferred at the same time the communication interface, the communication activity will interrupt this particular transmission and then take control of the communication interface and then start transferring data. So, these two are actually creating the problem in the communication. So, duration of these two task were sufficiently long to invoke a watchdog timer that was employed to ensure that the high priority bus management task was executing appropriately.

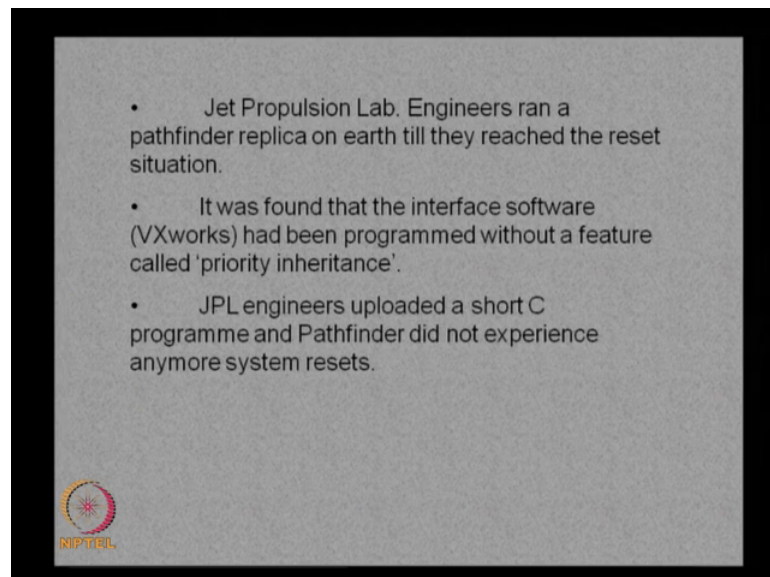
So, there was a bus management task which had got a high priority and particular task of this system is to ensure that everything is functioning properly and whenever there is a problem it will try to eliminate that error or to try to recover from that error. So, the sufficiently long these two tasks basically the meteorological data as well as the communication interface they were actually too long and often the watchdog timer will find that something is wrong with the system because it was not able to access the communication interface it was not able to get the a required data to ensure the a health of the system. So, at this time when the high priority bus management task was executing to check this particular task was a executing or not, it was often found that there was some problem or this particular system was identifying that there is an issue and it was trying to reboot the system.

So, in such rare cases the watch dog timer initiated a total system reset to prevent any further damage to the system. So, when this two particular sub systems were trying to communicate and then trying to get control of the mutex the priority bus management task was not able execute properly and that actually of resultant into a kind of a problem

or kind of a error with identified by the system the health check. And that actually was leading to a total a system reset to ensure that or assuming that there is an error in the system and it was trying to ensure that it is not go leading to major errors. So, it will go for a total system reset. So, this was what was happening in the system.

So, there were many subsystems it was interacting and there was some kind of a health monitoring as well as the error identification of error and eliminating the error. So, these system were actually conflicting and then trying to take control of the system or the communication system at that time total reset were taking place.

(Refer Slide Time: 07:29)

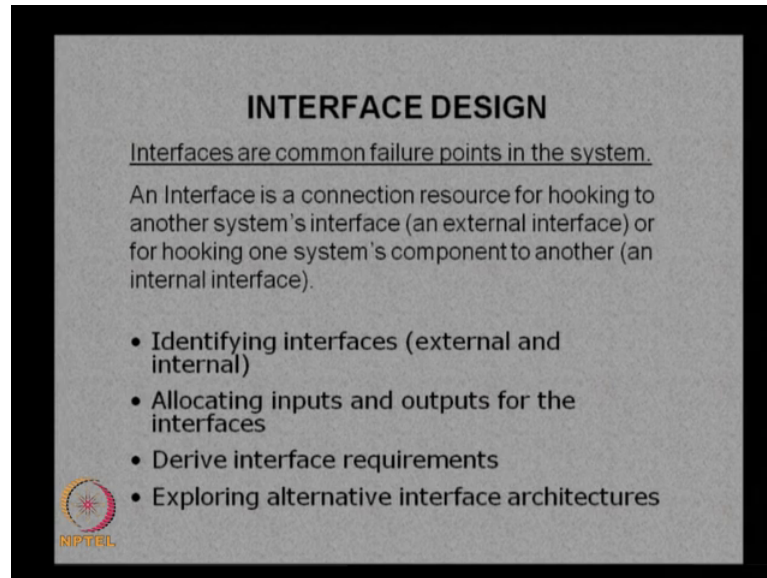


So, the jet propulsion lab engineers ran a pathfinder replica on earth till they reached the reset situation. So, you know to find out what actually is happening in the system the engineers ran a replica on earth till they reach the reset situation then identify the problem that there is a problem with the priorities of each tasks or some task need to be prioritized so that the priority task will always get the mutex and then that will not lead to a error situation.

So, there it was one that the interface software which was used for the interface had been programmed without a feature called priority inheritance. So, this particular feature of priority inheritance which actually a give some particular priority and that inherit that priority for different subsystems that was not there and the engineers uploaded a short C program and pathfinder did not experience any more system pieces. The solution was

very simple, but identification of the problem and then solving it was the difficult task because they had to run the replica on earth and then find out what led to the problem and then they solved the problem.

(Refer Slide Time: 08:34)




**INTERFACE DESIGN**

Interfaces are common failure points in the system.

An Interface is a connection resource for hooking to another system's interface (an external interface) or for hooking one system's component to another (an internal interface).

- Identifying interfaces (external and internal)
- Allocating inputs and outputs for the interfaces
- Derive interface requirements
- Exploring alternative interface architectures

 NIPIT

So, this actually shows that interfaces are very important in engineering systems in even a minor error can actually lead to a total reset or the failure of the complete system and therefore, it is necessary for the system designers to look at the interfaces in detail and have a formal procedure to design the interfaces and ensure that interfaces function properly and it actually provides the necessary communication between different some systems.

So, let us look at how we actually define the interfaces and how do you actually look at the various aspects of the design of interfaces. As you know interfaces are common failure points in the system, so most of the possibility of failure actually comes from the interfaces, which is defined as a connection resource for hooking to another systems interface. So, the interface is basically a resource for hooking to another systems interface or subsystems interface when it is an external system then it is another systems interface or for hooking one systems component to another that is an internal interface.

So, you can have an internal interface or an external interface. An external interface is basically used for a system to hook onto another system; an internal interface is used for hooking to the same systems components for the configuration items.

Important aspects of interface design are basically looking at these interfaces identifying the important interfaces in the system you look at the external interfaces required and internal interfaces required. For example, even you consider the elevator or any other system you can actually identify many interfaces one may be with external system. So, the elevator need to interact with the passengers is an external system and that actually we need to have different kinds of interfaces, the communication interface in terms of data input and some other kind of interface with the maintenance people or interface with the emergency services. So, all these kinds of interfaces need to be identified in advance.

So, that is the first task, identify all the interfaces external. Then internal interface also, in internal interface can be identified using the functional structure or the physical architecture of the system. When we do a and head of zero method of functional decomposition and then identifying all the functions this actually will give us different inputs and outputs from different sub functions and the that will tell us what kind of interfaces are needed between these sub functions.

So, when we convert that into in a physical architecture then we will know that these components which actually provide the functions need the particular kind of interface in terms of digital data going digital data interface or analog data interface or some kind of a mechanical interface. So, this is the first step in identifying or designing the system interface or interface design of systems.

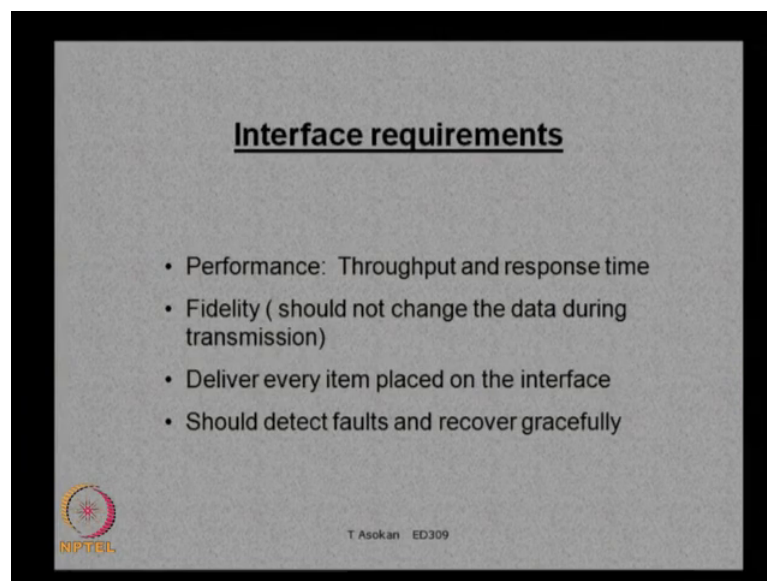
And once we identify the interfaces the next task is basically to allocate the inputs and outputs for the interfaces. So, every interface like every system design task will try to identify the inputs and outputs for the system. So, what kind of input is coming to the system whether it is a digital data input or analog input or what kind of data structure is there for that particular input, what is the size of the input, what kind of transmission speed is needed.

So, all these things need to be identified in the inputs and outputs for the interface and then we derive the interface requirement interface requirements in terms of the system should be able to transmit the digital output from one point to another point or it has to accept the digital requests from passenger or digital data to be requested the system should be able to receive the digitized information from one particular system. So, that kind of interface requirement can be developed after identifying the interfaces.

And then exploring the alternative interface architecture, so there can be many architectures for interface. So, we need to look at what kind of an architecture will be the best for this particular system. So, we will discuss about various architectures for the interfaces, so from these architectures we need to choose a most suitable interface. So, when we discuss about interface design we will actually look at only and other communication interfaces the mechanical interface and other physical interfaces needed within the system will not be discussed here because those are mainly coming from the physical architecture.

So, whenever we identify the particular configuration for the interface or the alternate interface architectures we basically look at the alternatives and then the physical system design of that will be a separate task. So, we do not discuss about the physical interfaces we will be looking more on the communication interfaces to be provided for the system.

(Refer Slide Time: 13:51)



So, the main requirements for an interface is the important performance requirements are basically in terms of performance there are throughput and response time are two important parameters in terms of for interfaces. So, what is the data I can transmit and what is the response time for this transmission, these are the two parameters we need to look at when we design an interface.

And apart from that we need to look at the fidelity of the data that is the integrity of the data whatever we transmit from one point to another point should reach that point



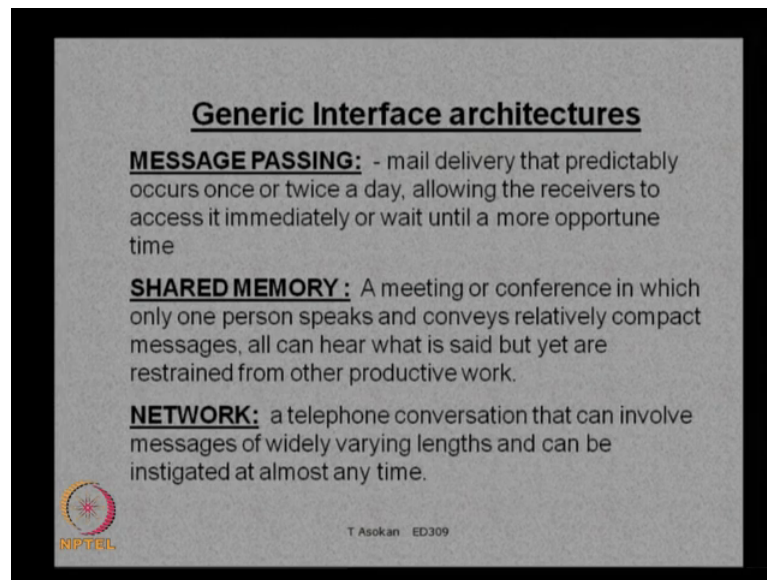
without any damage without any change; that means, no changes should be there in the data during transmission. So, whatever we transmit from one point to another point it should reach the without any variation. So, that is the integrity of the data transmission that is also that is one of another important factor to be considered in the design of interfaces.

And then deliver every item placed on the interface. So, the interface should deliver every item placed at the interface. So, it is not filtered out or it should not lose because of external interference or any other factors there is not be any loss of data in at the interface for every item place that the interface should be transmitted.

And the other one is basically it should detect faults and recover gracefully. So, whenever there is a fault or in a system that is error happening the system should be capable of coming out of that situation gracefully that is without causing any other damage to the other systems. So, whenever there is a problem identified by the interface it should try to minimize the damage and eliminate that particular damage and then start continue to function normally without affecting other system performances. So, these are the important requirements for interface.

So, whenever we design interface we look at these parameters these factors and then ensure that whatever the architecture we choose for the interface basically meets all these requirements. So, that there are no problems in the interface whatever the data we want to transmit is getting transmitted and there now, there is in error the system tries to come out of it gracefully and continue to perform without much problem for the whole system.

(Refer Slide Time: 16:11)



Let us look at the some of the, and generic interface architecture there are different architectures available depending on the situation we need to choose this architecture.

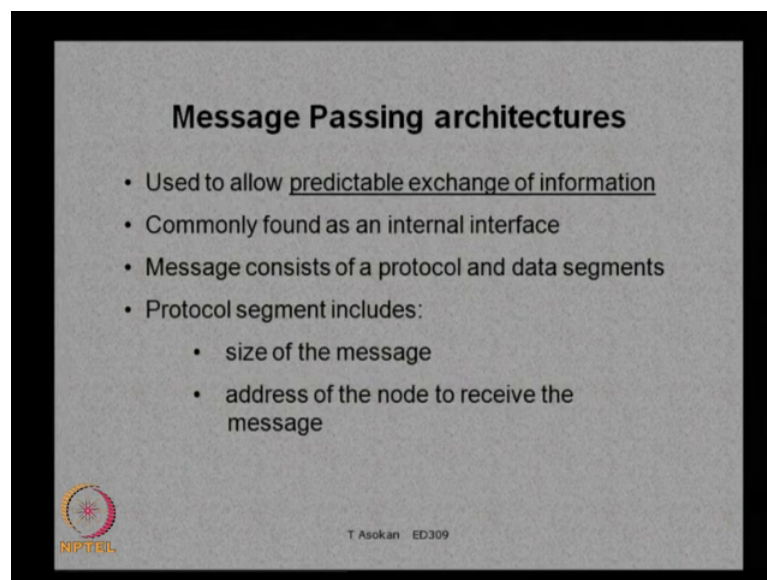
So, there are basically three generic interface architecture the first one is known as message passing. So, this is a something like a mail delivery that predictably occurs once or twice a day allowing the receivers to access it immediately or it until a more opportune time. So, it is a very predictable data exchange. So, the system will be knowing that what will be the frequency at which that data will be transmitted. So, every hour or every second depending on the subsystems the system can actually identify what is the frequency of this data transmission and it will continue to transmit this data and the receiver can actually access it immediately or actually can access it at a later stage. So, this kind of architecture is the message passing. We will see the details of message passing bit later.

The other one is shared memory architecture this is something like a meeting or conference in which only one person speaks and conveys relatively compact messages where all can hear what he said, but yet are restrained from other productive work. Here it is unlike the message passing it is like where everyone can speak, but not at the same time. So, one person can actually speak and others can listen and other whenever one person is speaking all other only others can what can do is only to listen to that one. So, they cannot really interact. So, it is only one way communication from one person. So,

he passes the message to a shared memory and others can actually get it from that memory shared memory. So, that is the shared memory architecture.

And the last one is the network architecture which is very common. So, it is like a telephone conversation that can involve message. So, widely varying lengths and can be instigated at almost any time. So, this is network which is very common in most of the systems. So, here we can actually pass messages and different people can hear and then others can also pass messages we can have different architectures and network itself depending on the requirement can have one to one or you can have one too many or you can have redundant communication and network or loops that is the network architecture.

(Refer Slide Time: 18:32)



Let us just see the architecture of the message passing. So, that was the first one we discussed the message passing architecture. This is basically predictable exchange of information.

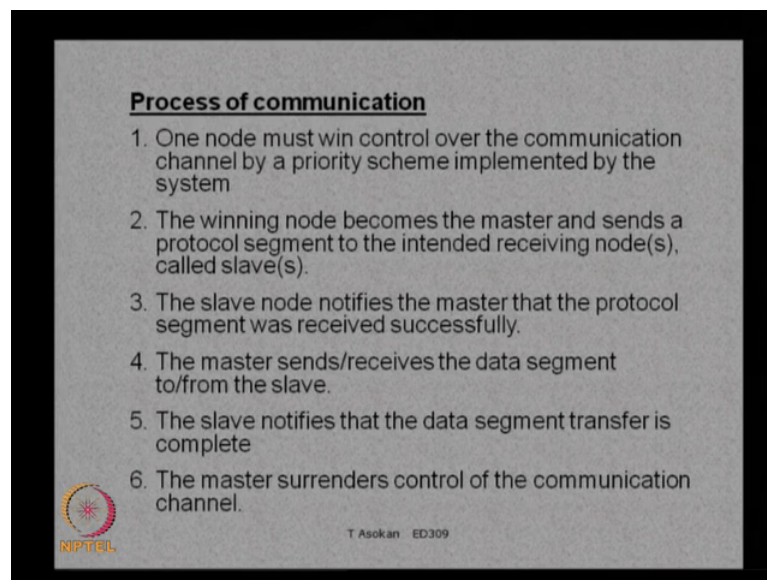
So, as I mentioned the predictable exchange of information whenever. Now, there is a predictable exchange of information we can use the message passing architecture. This is commonly found as an internal interface in system. So, this cannot be normally used has between the system and the external system because the external system the interaction between system and external system may not be always predictable, but within the system the system designers will be knowing what kind of data to be transmitted

between the subsystems. So, in that case we can go for a message passing architecture because of the predictable nature. And every message here will be consisting of a protocol and data segment. So, the message to be passed between the elements of the system of the subsystems will be having a message protocol and a data segment.

The protocol segment will include us the size of the message and address of the node to receive the message. So, when we have multiple nodes in the system, so the message will be having a protocol which will actually tell the size of the message and address of the node to receive the message and then other segments will be having the data segments also. So, this actually the first segment will tell to whom the particular message is addressed and then what is the size of the message and the second, but will give actual message.


And the message passing protocol or the communication the process of communication in message passing architecture is basically one nodes must will control over the communication channel by a priority scheme implemented by the system.

(Refer Slide Time: 20:02)



**Process of communication**

1. One node must win control over the communication channel by a priority scheme implemented by the system
2. The winning node becomes the master and sends a protocol segment to the intended receiving node(s), called slave(s).
3. The slave node notifies the master that the protocol segment was received successfully.
4. The master sends/receives the data segment to/from the slave.
5. The slave notifies that the data segment transfer is complete
6. The master surrenders control of the communication channel.

 T Asokan ED309

So, there will be many people who using the same interface. So, one node must win control over the communication channel by a priority scheme. So, this priority scheme is design by the designers. So, the system designers will be knowing which one is more important and how do we actually prioritize between different nodes in the system and

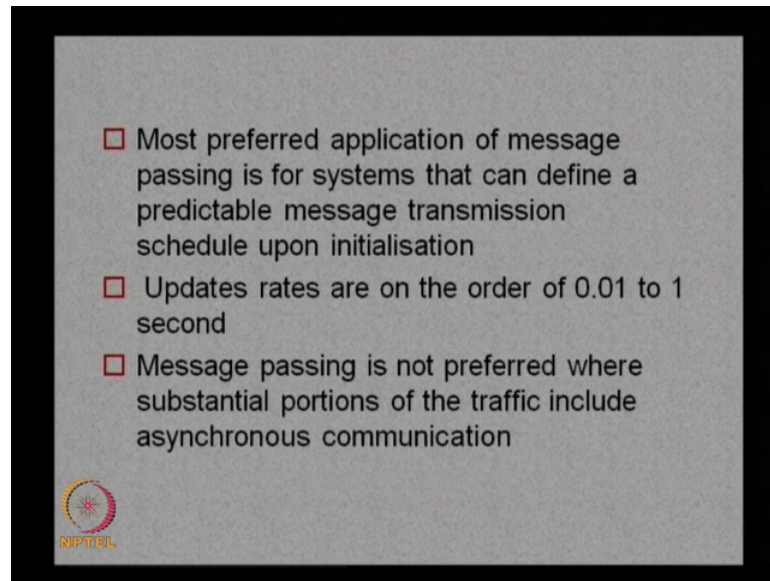
based on that priority one node will take over the control of the interface. So, this node will be able to transfer the data once it has got the control over the nodes.

And the winning node becomes the master and sends a protocol segment to the intended receiving node. So, once the node takes. So, the control over the network or the interface it actually provides in becomes a master and sends a protocol segment to the intended receiving node called slave. So, now, it becomes the master and all of that become slaves, then it will send a protocol segment to the receiving node which is the slave here. And once the slave node notify the master that the protocol segment was received successfully. So, once the master sends a protocol the receiver receives it and then gives an acknowledgement that the protocol has been received successfully and once that happens the master sends the data segment to the slave.

So, basically the master will be sending the data to the slave or the other way if the slave can actually give a segment saying that the data has been received. So, the master can actually send the data to the slave and then slave will give an acknowledgment. So, this is the normal process of communication in message passing architecture. So, there will be multiple nodes, one node will get the control of interface and then that node becomes the master and then it sends a protocol segment to the slave and the slave accepts that protocol and confirm the received of the protocol and once that is confirmed master sends that data and then the play acknowledges a received of data.

So, that particular cyclists over and then the node will actually give you the control the master surrenders control of the communication channel. So, once that the predictable exchange is over and master will just surrender the control of the communication channel. So, that any other node can actually take over that particular become a master again based on a priority scheme. So, this is the way how the communication takes place in message passing architecture.

(Refer Slide Time: 22:49)



And this is the most preferred application of message passing for system that can define a predictable message transmission scheduled upon initialization. So, message passing as I mentioned can be used for predictable data exchange. So, whenever system initiates and there are predictable data exchanges for such systems only this is most appropriate. So, the system designers will be knowing how the system works and what kind of interface data transfers takes place, so based on that we can go for this particular message passing architecture.


And here the update rates are on the order of 0.01 to 1 second. So, that is the update rate in transmission and message passing is not preferred where substantial portions of the traffic include asynchronous communication. So, when there is asynchronous communication where it is not predictable or it not happening in the periodic interval and that is a very substantial part of the communication then it is very difficult to implement message passing architecture because it is barely used for predictable data exchange only.

So, whenever there is an asynchronous data transmission we go for the next one which is called the shared memory architecture.

(Refer Slide Time: 23:55)

**Shared Memory Architectures**

- Asynchronous communication requests are handled
- A fast access storage device, typically a memory device


 T Asokan ED309

Here in shared memory architecture asynchronous communication requests are handled. So, unlike the message passing where it is predictable data exchange in shared memory we go for a synchronous communication. So, whenever there is an asynchronous communication request with go for the shared memory architecture. Here a fast access storage device typically a memory device is used. So, this is the shared memory basically shared memory is shared by different nodes, so a fast access storage device that is used as the memory device.

(Refer Slide Time: 24:36)

**Communication process**

- A processor generates a read or write request for another address in shared memory
- The current owner of this variable is notified of the request
- The cache memory of the current owner is dumped to shared memory
- The read or write request of the processor is completed with a data transfer



In the communication process in this is basically a processor generates a read or write request for another address in shared memory. So, the shared memory will be having all the address of those nodes which are using the particular shared memory. So, processor generates a read or write request for another address in shared memory whichever system wants to transmit a data to a particular another node. So, it actually generates a read or write request for that particular address.

The current owner of this variable is notified of the request. So, one variable may be used by some other owner or some others are using the particular variable then this will be notified and to that user and the memory of that current owner is dumped to the shared memory. So, whoever wants to write a particular variable suppose there are some global variables to be updated like temperature or pressure or some other data to be updated in the system in that case whoever wants to write or update that data will send a message to the shared memory requesting for access to that particular variable. So, some other system may be using that variable. So, that particular subsystem will be intimated about this particular request and then if it is possible to dump that particular variable then the present user will dump that variable to the shared memory and then the read or write request of the processor is completed with a data transfer.

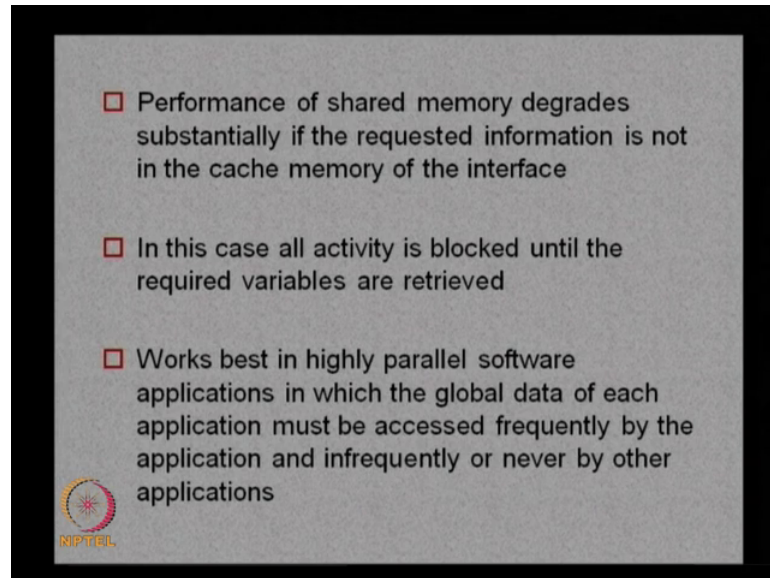
So, once that is a made available to the requester then the data transfer will take place and again the variable will be available with the press the owner till the next free case this owner variable will be kept there and whenever there is another request for this data this particular variable that will be a dumped again to the memory and can be used by other users. So, this is the normal communication process in shared memory architecture.

So, as I mentioned there is a memory device a fast access memory device you stay here and whenever a particular user wants to access a variable in the shared memory it gives a notification. And then whoever is using that variable will dump that variable to the shared memory system is requested for that variable can read or write the data and then again complete the data transfer and then release that variable or you can keep is the variable till the next request comes. So, this is the normal way of communication in shared memory architecture.



So, here actually can be asynchronous because I mean there are there is no particular time interval on which the particular data comes. So, whenever any system wants to access the variable and then transfer the data.

(Refer Slide Time: 27:23)



So, the problems with this shared memory architecture is that the performance of this degrades substantially if the requested information is not in the cache memory of the interface. So, whenever that particular information or the variable is not available in the cache memory then it becomes a problem and the performance degrades because it will wait for that data to come or this activity is blocked until the required variables are retrieved.

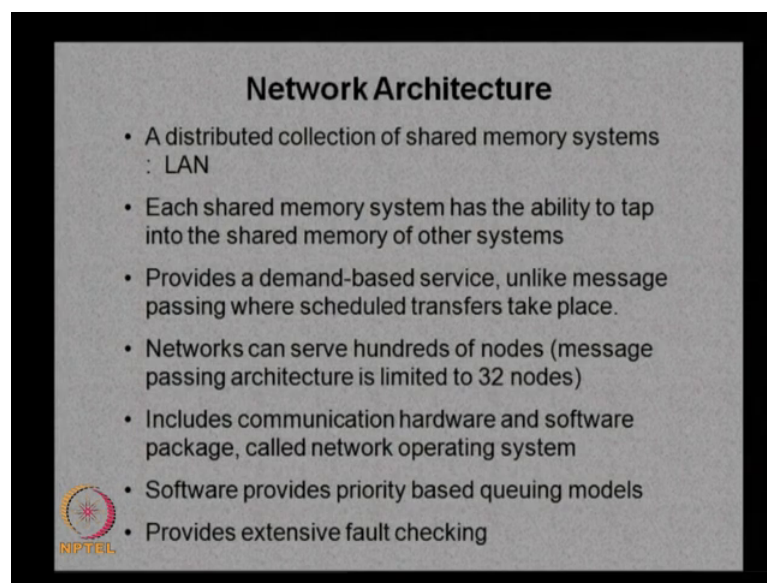
So, until that the particular request is completed do not be able to do any other task. So, whenever a particular variable or a task is not this not available within the cache memory it has to get it from some other users then it will wait till that variable is made available and then only it will go for the next task. So, that the actually this may reduce the performance sometimes. So, that is one of the problem with the shared memory architecture.

And it works invest in highly parallel software applications in which the global data of each application must be accessed frequently by the application and infrequently or never by other applications. So, this is best architecture for parallel computation applications parallel software applications where the software will be having some global

variables it will be used by other subsystems, but it has to update the data frequently. So, in this case this particular application can actually access it frequently and then update it, but others may or may not be using it. So, that kind of situation this is well suited.


But in the cases where others are also using then that first problem will be there like the edges to wait till that is made available to the first application, so that may degrade the performance. But otherwise if there is only it is a global variable which need to be complete related frequently by the application then this is a very good architecture.

(Refer Slide Time: 29:09)



**Network Architecture**

- A distributed collection of shared memory systems  
: LAN
- Each shared memory system has the ability to tap into the shared memory of other systems
- Provides a demand-based service, unlike message passing where scheduled transfers take place.
- Networks can serve hundreds of nodes (message passing architecture is limited to 32 nodes)
- Includes communication hardware and software package, called network operating system
- Software provides priority based queuing models
- Provides extensive fault checking

 NIPTEIL

Last one is the network architecture. Network architecture as the name says it is a distributed collection of shared memory system. So, you have multiple shared memory systems in a network that is the network architecture normally the local area network. So, here you will be having multiple shared memory systems. So, we can actually interconnect many shared memory systems to get network architecture.

So, here each shared memory system has the ability to tap into the shared memory of other systems. So, every shared memory can actually tap into the memory of other shared memory systems. So, that is how we are actually networking it. So, you have a multiple set of shared memory system each one can access the shared memory of other systems also. And provides a demand based service unlike message passing where scheduled transfers takes place again here also its a demand based service not like the message passing where it is fixed or the scheduled transfer its more of an demand based

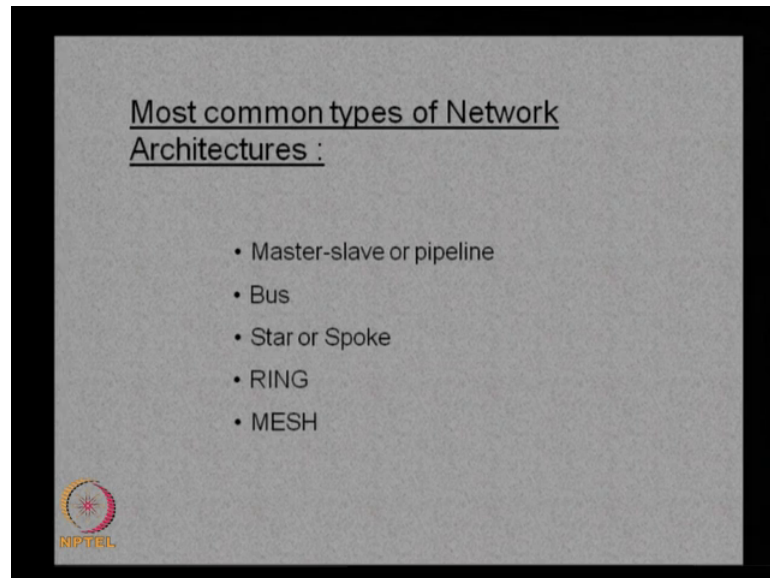
service. So, whenever a system wants to access the shared memory of other system. So, it can actually demand for that particular variable or particular access and it will be provided. So, based on the demand it can be provided.

And then networks can serve hundreds of nodes. So, the advantage here is that it can have hundreds of node message passing can its limited to only thirty two nodes. So, here that is the advantage you can have a number of nodes in the network and it includes communication hardware and software package. So, you will be having a hardware for communication and along with that there may be a software also it will be the network operating system.

So, apart from having hardware software will be there to regulate the flow of traffic and provide regulated access to different shared memories. So, and we need to use a network operating system to provide that and the software provides priority based queuing model. So, this software will be having some priority based models queuing models so that access to the two different resources will be based on this particular priority that is decided by the system designer, but that is implemented through the network operating system it will priority based queuing.

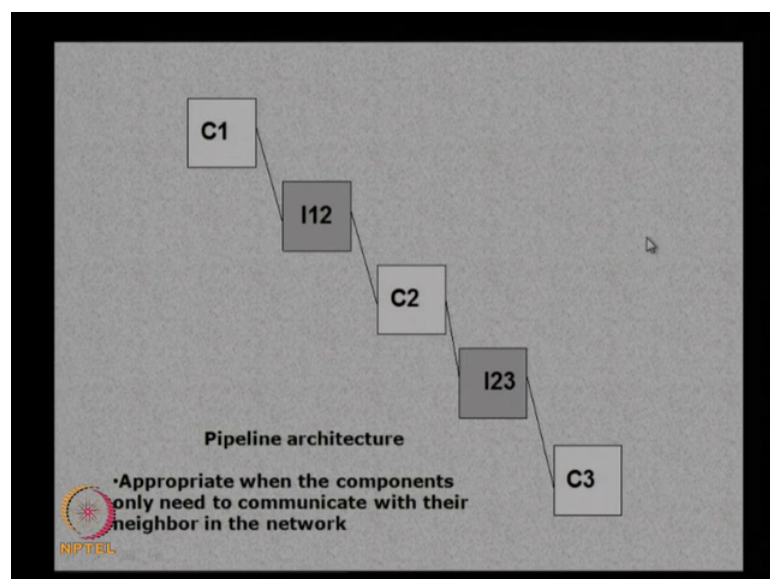
And it provides extensive fault checking also. So, apart from providing the interfaces it actually end a queuing models the system will provide extensive fault checking also. These are the important points about network architecture. So, you have many shared memory systems connected together and you can have hundreds of nodes because it is not limited to the message passing architecture or other architecture. So, you can have multiple nodes here and there will be hardware and software attached with this and the software provides a priority queuing models as well as for the fault and fault checking systems also.

(Refer Slide Time: 32:04)



Let us look at some of the network architecture is commonly used in engineering systems. Again as I told you we are looking only at the communication system here as the interface. So, in the network architecture there are most commonly applied or implemented architectures are master slave or pipeline architecture our m bus architecture, another one is the star or spoke architecture, then the ring architecture and mesh architecture. So, these are the commonly used in engineering systems.

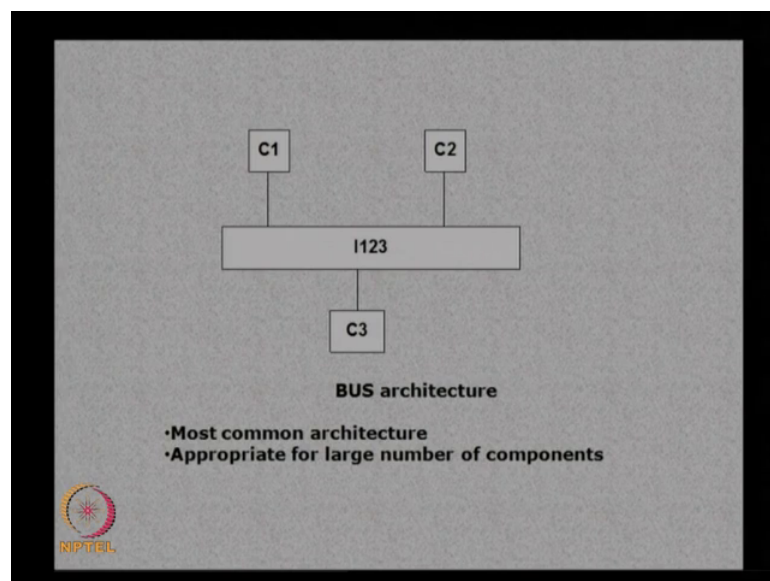
(Refer Slide Time: 32:36)



The master slave architecture as you can see here this is a master slave or a pipeline architecture where you have components C1, C2, C3 and there is interface I12, I23 etcetera. So, this kind of architecture allows you to have communication between adjacent blocks or adjacent components. So, C1 can actually communicate to C2, and C2 you can communicate to C3, but there is no other communication possible here.

So, this is appropriate and the components only need to communicate with their neighbor in the network. So, whenever procurement is only to communicate with the neighbor then this is the best way of architecture that this is the pipeline architecture. Though I12 I23 represents the interfaces and C1 C2 axes there are represent the components for communication. So, limitation is that it can communicate only with the neighbor. So, C1 can communicate only with the C2, similarly C2 can communicate only with the C3 or C1 because these are two neighbors. So, whenever these the requirement is only communication with neighbor you can implement the pipeline architecture.

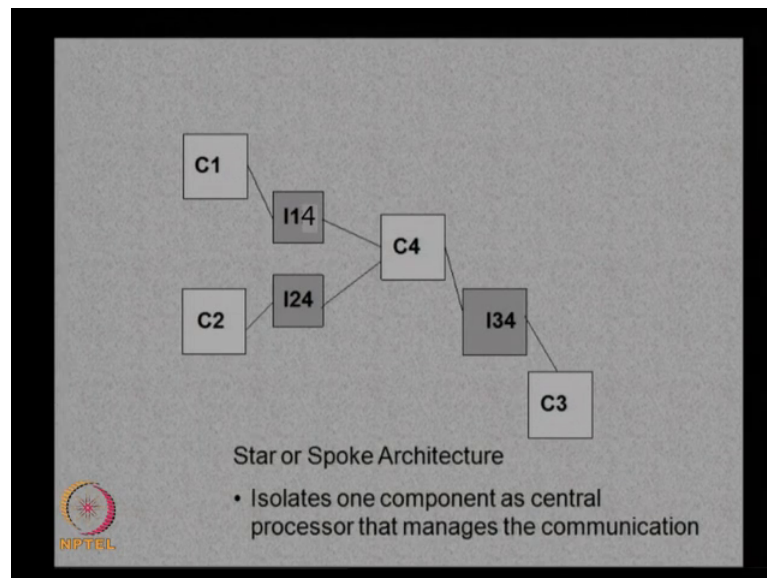
(Refer Slide Time: 33:42)



And the other one is the bus architecture which is the most common architecture we can see it in many places this kind of architecture where there be a common interface I123 where I can have one two three is actually here C1 C2 C3 are the components. So, these components can have common interface element. So, I123 represents the common interface between these components.

So, he can actually have any number of components over here. So, this bus will act as an interface between all these components. So, you can actually have communication between C3 to C1 or C2 to C3 through this interface or you can have any number of components and all these components can have an interface can this connect between this components through the interface. And this is appropriate for large number of components when you have a large number of components to be connected then this is the most appropriate one versus the bus architecture where you will see in many of the computers and other systems bus architecture is employed because you can have many components connected to it and all the communication can be through a common bus. This is a very commonly employed architecture in networks.

(Refer Slide Time: 34:51)

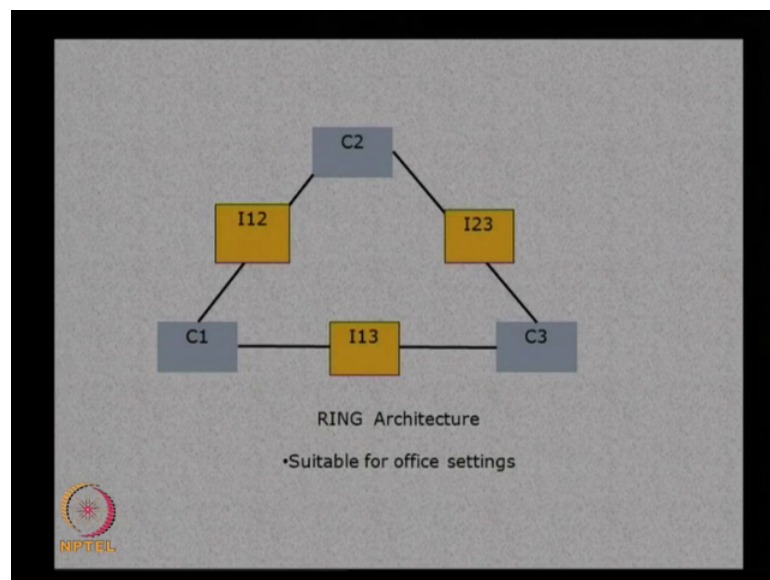


The third one is known as the star or spoke architecture. Here you can see that it actually takes one component as a master or a central processor which actually manages the communication between different components. So, you can see here take C4 is taken as a central processor and then this actually allows communication between various components I14, I24 or I34 these are the interfaces provided here between 3 and 4, and 2 and 4, and this is 1 and 4, this is I14. So, these are the interfaces provided between the central processor and the components and if the C3 wants to communicate with C2 and actually C4 can manage this through this I34 and I24.

So, the central processor this actually can manage communication between other components also. Here it is not only one to one communication we can actually have communication between various other components also only thing is that this is managed through a central processor. So, that is the difference here in a star or spoke architecture.

This again useful because if want to have not regular communication with the component, but still want to have some kind of connectivity between different components then it is possible to use this kind of an architecture because the components the central component can actually manage the communication between various components.

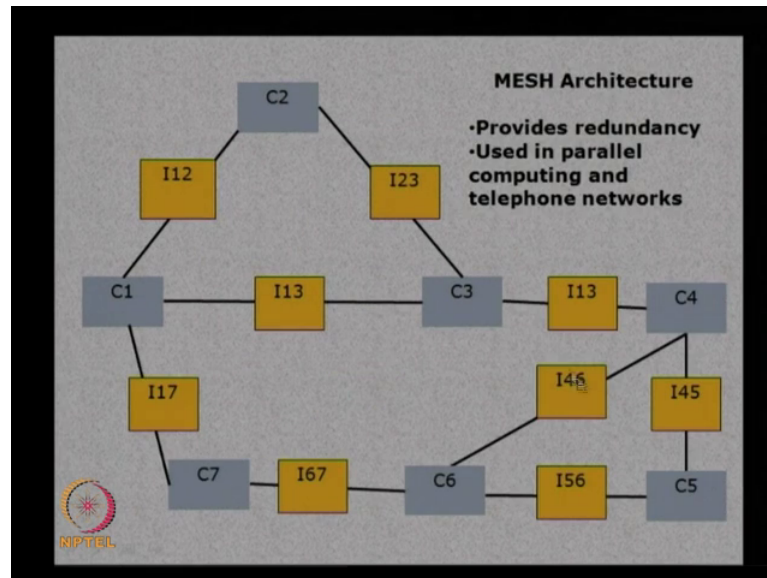
(Refer Slide Time: 36:13)



And this is known as the ring architecture which is again it is more like a closed loop control you can have communication between C1 C2 C2 C3 C 1.

So, these are the interface elements which actually makes it possible to have communication between elements and again it is a closed loop, but still not always one to one communication may be possible because in this case means only three components you can have C3 to C2 C3 to C1 like that, but when you have more components then this may not be really possible, but still you can have a closed loop architecture using a ring architecture. So, that is normally used in for office settings.

(Refer Slide Time: 36:53)



And this is an extension of the ring architecture. So, ring architecture as I mentioned when you have more components then it may not be possible to have ring architecture. So, in this case you can go for mesh architecture.

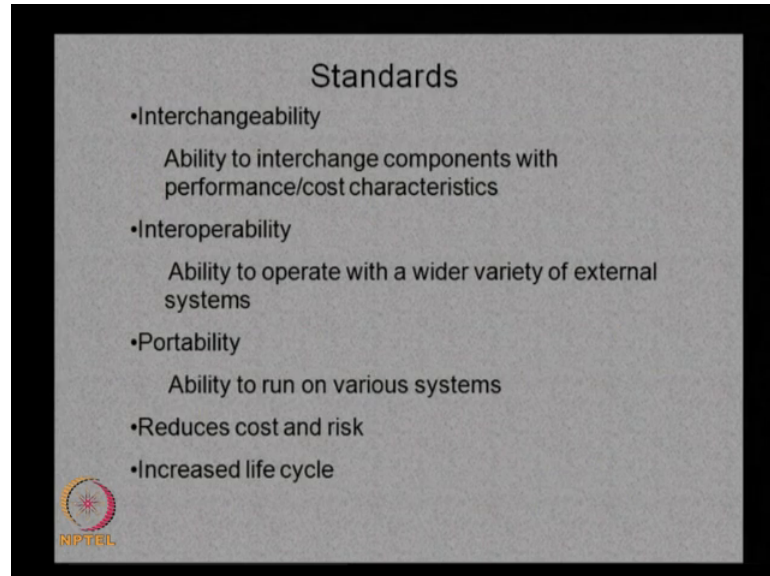
This mesh architecture basically provides redundancy because you can have multiple connection you can have many ways of connecting to a particular component. So, you can have a redundancy between the components and then it is used in parallel computing and telephone networks. So, commonly used in parallel computing and telephone network where you will have multiple activities provided over here. So, you can have multiple connections or redundant connection between different elements as you can see C1 is connected to C2, C2 is connected to C3, C3 to C1, and C1 to C 7, C 6, C3, C6 to C4, C4 to C5 like this. So, you can have multiple redundancies also in the network.

So, if one of these links are not working then probably it can actually connect through C Cs can be connected to C4 through these links through I56 and I45. So, this way you can have a redundancy in the connection bring similarly you can provide a connection between C6 and C3 also or you can have a connection between C6 and C1 by providing another interface through this. So, like this you can have multiple connections and redundancies can be incorporated which is useful whenever you need a very reliable network connection especially like a telephone networks you found line fails you can



actually go for another way of interacting or linking the components. So, in that kind of situations a mesh architecture is use.

(Refer Slide Time: 38:29)



So, this other about the interface is commonly used. We can have different kinds of interfaces as I mentioned we can have a message passing architecture or we can have shared memory architecture or you can have network architecture. And we saw that network itself we can have different ways of thinking the a nodes you can have a mesh type or star type or you can have an in line or I pipeline type of architecture, but whenever we have this kind of architectures it is necessary to have some kind of a standard where we ensure that whatever the form you use it is possible to have to maintain some kind of a flexibility within the system. So, we need to have various aspects for the system like we would like to have the system to be interchangeable that is if you are using a particular network or architecture we should be able to use different components that basically a interchangeabilities, basically the ability to interchange components with the performance or cost characteristics.

So, if you want to change one component in terms of performance or cost characteristics if you want to change with one component then it should be possible. And similarly you should be able to have interoperability also basically they ability to operate with a wider variety of external systems. So, whenever you have many external systems and you want to operate with different systems then you need to have inter operability also. Similarly

portability is the ability to run on the area system, if you have particular software and or hardware you should be able to run it on a different system. So, you not be restricted to a particular operating system or a particular hardware, so we need to have portability also.

Similarly when you have a standard followed then we can actually reduce the cost and the risk of using that particular one and there you can have an increased lifecycle because we always follow a particular standard in interfaces. So, this actually shows that we need to have a proper standards in developing the interfaces basically to ensure that whatever we develop can be interchanged with other systems or we can have an interoperability basically like personal computers normally designed with a standard interfaces power supply or hard disk or the ram that we always having some kind of a standard.

So, that you can change the component or we can operate it with a different component or you can actually put it in a different system. So, like this, this standards basically allows you to have freedom in the design as well as ensure that the cost and performance around optimized. So, there are different standards to be followed and there are commercial and existing standards in interface design.

So, we will stop here for the time being and we will look at the this standard what kind of standards are being used in the industry in the design of interfaces and how important they are and how do we actually implement these standards in a interface design these points. We will discuss in the next class, till then good bye.