CSC 435 Study Log – Kaijun He

Professor Elliot ---Winter 2017-2018

Total Word Count: 7905

**Twenty Concept Section**

1. **Network latency**: It’s a new definition to measure any types of delay which is happened in data communication through an internet, people always use the network latency to consider the quality of the internet speed, in most cases the low network latency means high speed internet. How to measure the network latency? For example, there are two points in network, but based on different conditions, it has different measurement plan, we just use simple condition to measure latency, we indicate the time is synchronous, therefore, in small distributed system, we supposed to have a global time, set up two integer variable m and n, m is the time when point A send out message, n is the time Point B receive the message, and then m – n is the time between sending out message and receiving message, the network latency between these two points are m – n.
2. **Public key encryption**: Public key encryption is also known as asymmetric-key, before we know why we need public, we need to know that there are two different types of keys, public key and private key. The reason why we need these two keys are the security reasons, private key is the key that only your computer know, if you want to do communication between your computers and outside word, so other computers want to access the message you send out, you must have to use your private key which nobody knows but you to generate a public key which anybody can see the public key, they can use the public keys to communicate with it by decode an encrypted message that you send out.
3. **Lamport’s logical Clocks:** As what we know that there is a physical time in real world, but in distributed system, we can’t schedule the order of event based on physical time, because sometimes physical time is not accurate. Therefore, Lamport creates a logical clock to help ordering the events schedule. We will be very familiar with the variable timestamps, so in Lamport’s logical clocks, every time a process sends out a message and send out the timestamps together. For example, there are three processes, A, B, C. process A sends messages to process C with timestamps 5, process B sends out message to process C with timestamps 10, compare two timestamps, therefore process C will receive the Process B first then A.
4. **DHCP**: DHCP is short for dynamic host configuration protocol. It’s a very important protocol to manage internet IP address. DHCP is a client-server protocol to assign IP address dynamically in pool of unique IP address pools. The benefits of DHCP is used to avoid errors caused by manually type in IP address, for example, in the local network, two devices connect to the network and both device are manually type in same IP address, this will trigger error, second benefits is to reducing network administrations. the DHCP protocol will automatically assign a new IP address when the IP address’ lease expired.
5. **Socket:** socket is a word that means two-part plug together and perfectly match, in network and distributed system, the socket is used to make two machines on network to communicate with each other. There are several types of sockets, for example stream socket, in Client-Server application, client is typing in terminal and then what the client types in will send to server by using transmission control protocol. From professor’s programming assignment how does it works, but for the stream socket, you must have to make sure the connection between client and server is still connected, if not the stream socket will not work. Another type is datagram, it’s different from stream socket, this socket type doesn’t need to make sure internet connection is always on, because each message sending out in datagram socket, it carries destination address with it.
6. **Synchronous communication:** it’s also known as real-time communication. There are two nodes, one is sending out message to another node. With the synchronous the sender node is blocked until its request is accepted by the receiver node. Advantages of the synchronous communication is real time collaboration, so the client will get immediate response, but sometimes if the receiver doesn’t accept fast, this will make sender node blocked long time until accepted by node receiver.
7. **Asynchronous communication:** this is used to compared with synchronous communication, asynchronous communication is that receiver is not needed to response to sender directly, what’s this means? It means sender can send message any time even though the receiver is not ready yet. For example, emails are asynchronous communication tools, one user send an email to another person for some rely, but it’s no need for second person to reply immediately. The email receiver can reply anytime that he wants to reply.
8. **Transactions:** transaction is a collection of operations, for example in distributed system, it’s same concepts in database courses, the main characterize feature is atomicity, what that means? For examples, in database a thread is reading and writing which try to modify the value of variable, if the changes are commit correctly, the modification will be saved, if the process is aborted, nothing will be happened, it returns to the states before this thread is running.
9. **structured peer-to-peer systems**: structured means it’s organized logically, there are a lot of types of peer-to-peer systems. It can be structured like circle ring, binary tree, red-black tree, link list, hash table, all the data structure you learn before can be used here. For example, the distributed hash table, first we set up a rule for the hash table, like n = (m mod 5), and then construct a distributed hash table, we store all the keys, and when we needed, then go to hash table look up the key then we will know the node.
10. **Network address transmission:** it’s also known as NAT, we all know that every device, element in network will need an IP address, it must be unique compared to others, as the development of internet, more and more people are using network connected devices, such as computers, so every computer needs a IP address, it’s definitely not enough amount to make every computer in the world has a IP address, therefore the network address transmission comes out, how it works? First there is a public IP address, when people use this IP address, they will find a group of computers, and each computer in that group will use NAT to create some local IP address to each element of the computers’ group.
11. **Internet Protocols:** it’s also called IP, when we open a network and check the network status, we could find out two IP address which is IPv4, and Ipv6, IPv4 is a kind of 12 digits decimal number split with dot, because the capacity of internet users increasing, so Ipv4 is not enough to supply, therefore new version of internet protocols comes out, it’s IPv6, it’s hex values, by calculation it’s 128 bits hex value. How the internet protocol works? for example, it’s like your house address number, if you want to tell your friend where you live you can’t just say the city or street name, you should also need tell them the house number, so IP address it’s just like house address, it’s used to like users in network to communicate with each other.
12. **Code migration:** in the distribution systems, it’s also known as process migration, this means one process is moved from one component of distributed system into another component of distributed system, first we should understand why does distributed system process migration? For examples there are 10 machines under the distributed system network, one computer’s CPU is running like 99%, rest 9 computer is running like 10% which is just the usage percentage of system itself, so we want make all the computer use similar CPU usage, not just let one computer run so high, so code migration is transferring from heavy load machine to light load duty machine.
13. **TCP:** TCP is also called transmission control protocol, it’s also one of most important protocol in network. It’s used to break application programs’ data into some packages with number in it, because the internet can transfer limited amount of data each time, so TCP helps people to split the data into pieces. And send those packages through IP address to other node in the network, although the packages have same resource and destination IP address, it’s just like the programming assignment in wireshark, we divide a text file into pieces, in wireshark software, we could see that the destination IP is the same but the content in the packages are not same. So, the destination node will wait until all packages arrive, and then combine together to form a file like the file from sender.
14. **DNS:** it’s called as domain name systems, it’s pair with IP address, for example, like google.com, when you open browser and type in google.com, the browser will directly go to 8.8.8.8 which is the IP address of google. It’s just like contact list in your iPhone, if you want to call your friend, best way it’s to find his name and call him, although you are call by his name in contact list, but your phone will link the name with 10 digits phone number.
15. **Distributed denial of service:** it’s also known as DDOS, which is attack website, server to cause denial of service. We know that our design of distributed system through network is try to make clients and server-side like website to communicate well, so for security reason, most of programmer is going keep the process of communication as stable as possible. So DDOS is a very popular method that can cause hacker to attack website by trying to send large number of messages, connection requests or bad packages to the servers or websites, these will cause websites slow or crash. Then it will make website founder to shut down the website or server connection to keep their data safe.
16. **Bully algorithm:** there are three types of messages for bully election algorithm, Election, OK and coordinator, for example there are N processes, and the kth process send a election message to all processes that says it’s time to elect a coordinator and then waiting for a reply OK to determine which one will be the coordinator, start from the k+1th process, if it accept OK message then it will wait for coordinator message to this process after send back Ok message to the kth processer, if it accepts coordinator message , the coordinator will be the k + 1th process, if not just keep going and repeat the steps above until find the real coordinator.
17. **Distributed systems:** for distributed systems, the concept is that a collection of independent elements and then collaborate users into a coherent system. For examples, a Client-server application is a distributed system, when client sending out requests message to servers through the network, waiting for the reply from servers, and a distributed system must be open as well, so for this example, any new clients and servers can join this distributed system, and all the resources in this distributed system are shareable to all clients.
18. **FireWall:** firewall is a software or firmware that set up rules to decide any incoming resources are safe or not, it’s just like filter that block large size items go through the filter, for examples, if some incoming data are conflict with system kernel process rules, the firewall will abort the process of incoming data and block, remove it. Firewalls have several different types, packets-filter, stateful inspection filter, proxy firewalls and next-generation firewalls. We remember DDOS attack, there is an attack based on receiving bad packet, so packet-filter firewalls will examine all incoming packets in isolated environment. And the stateful inspection firewalls will check whether the incoming packet is related or not, we all know that TCP divides large data into small pieces of packet, and all the packets are related to each other, so this kind of firewall will go to check the incoming packets relationships of each other, make sure not some bad packet come in and risk the system security.
19. **Virtual private network:** since the development of internet technology, people are more and more focusing on the security of internet, so knowing of the security level of one website is very important, since unsecured network will steal your personal identity, information and so on. That’s why people sometimes receive junk information by email or phones, because your information has been stolen and sell to some companies. Virtual private network is a technology to a safer connection over a unsecure network, how to ensure it’s safe by using VPN, because all VPN users must use passwords, tokens or other security methods to log in VPN and go to unsecure website through VPN.
20. **Global position** **system:** GPS is a kind of system to calculate the coordination of position on the earth surface, actually we are using GPS a lot in our daily lives, for example the google map when you are typing in address and use is as a navigation. But what’s the GPS in the distributed systems? For a distributed system, it uses several ways to coordinate each node in this system, for example, one method is centralized positioning, it’s used to be given 3 node positions and use these three nodes to calculate one node’s actual location. Another is decentralized positioning, if you learn algorithm course, then you will know how to calculate optimal results. It’s same as this positioning algorithm.

**Forum Posting Section**

Post one: After i read the chapter 2 page P111 to P116 about the threads in distributed System.

since i have take some courses about multithreading in operation systems, in my understanding, multithreading in the distributed systems

will be similar to multithreading in one single operation system, so the rescheduling will be the main factor in distributed systems as in operating systems, in the text book it states the characteristics of the multithreading model-- Parallelism, blocking system calls. In operating system, the multithreading rescheduling is using some defined rules like FIFO( first in first out), Round Robin and so on. the rules is in below link :<https://docs.oracle.com/cd/E19455-01/806-5257/mtintro-69291/index.html>.

it's not too hard to understand for single system rescheduling, but what the rescheduling happened at the same time for one single system and muti-computers in distributed systems. For example, there are 5 computers in this distributed system and each computers are running like 4 thread together, so in this condition what kind of rescheduling policies will be used, i mean it's not easy to decide the urgency level of thread in a mass amount of computers, so anyone have any idea about this issue?

Post two:

Hi all,

        it seems like that professor only supply us that windows version capture sample of wireshark, but if we are using mac OS, after type in

traceroute gaia.cs.umass.edu 56, and we suppose to get some output in wireshark, in windows we could see 98 echo ...... with ttl=1 , ttl=2 and so on, i'm confused about what we should look up in mac OS version wire shark, is that UDP protocol with info like 62898->33445 len=28?

Post three:

Hi all:

              since the command given for the lab are mostly about the windows os and a little about linux, for people who are using

mac os, maybe this will be helpful for your lab.

<https://developer.apple.com/legacy/library/documentation/Darwin/Reference/ManPages/man8/ifconfig.8.html>

this is link is the command man page for ifconfig, it's exactly same output when you type in man [command] in your mac teriminal

**Study Log Section**

**Chapter One: Introduction**

* 1. the preface:

1. two technologies which is used to change the computer system are invent since mid 1980s.
2. One is the creation of powerful microprocessors which lead to reduce size of computer and computer can be used by individuals.
3. One is the high-speed internet, as we know it’s called network, generally the network is used to make thousands of computers to communicate with each other.
4. After those two technologies has been invented, more and more network has been built up, so the communication between a large amount of computer has been an issue, therefore after we put those computers together, we call it that form a distributed system.

* 1. what is distributed System?

1. The definition of distributed systems, we could easily understand that a distributed system is a collection of a large amount of computing elements which are not just computer and all computing elements can work well together.
2. Two characteristic features of distributed System
3. One is a collection of a large amount of computing elements which also could work well independently.
4. Second is that those elements could performance well together, just like water, one drop water is water and thousands of water drops also could combined as water.
5. Two group of group membership
6. Open group: any node could able to join the system and send messages, it’s like public Wi-Fi without password, so everyone who could see the Wi-Fi can join.
7. Closed group: Someone mechanism control the whole group, it decides the node is in the group or not, allow them join or leave, so only those nodes in group could send or receive messages.
8. A distributed system is generally formed as Overlay network
9. Two types of overlay network

* Structured overlay: it’s defined as well-organized which means it has some special logic to link each element in the overlay network
* Unstructured overlay: one node is randomly connected to other nodes.

1. We also call well-defined network as P2P (pear to pear)
2. Middleware and Distributed system
3. Middleware is almost same as distributed system but it’s used in network environment.
4. Middleware generally offer some services

* Communication: it’s also called as Remote Procedure Call, it’s likely to call function through network
* Transaction: when we invoke a service then could call it’s a transaction
* Service composition: this is often used in web-based application, one web-based service may use other service.
* Reliability: it’s easily understood as one user send a message to another user, so we should make sure that another user can receive message successfully.
  1. Design Goal

1. Four important goal to design a distributed system.
2. Supporting resources share

* Easy for user to access
* Easy for user to share resources remotely

1. Making distribution transparent

* Types of transparency

1. Access: hide how a resource in network is accessed
2. Location: hide the location of resource when it’s invoked, for example, web browser we type in [www.google.com](http://www.google.com), but we don’t know where the google server located at real world and such name is called uniform resource locator.
3. Relocation: hide whether the object’s location is relocated somewhere when it’s invoked. And when google data center move to another place, we don’t know since it only gives us URL.
4. Migration: hide the fact an object is relocated into another location.
5. Replication: hide whether the object is replicated or not. For example, there are several copies of same data, when user receive data, only thing we know it’s that what we receive but who send out, where it comes form, we have no idea.
6. Concurrency: hide that object maybe shared by several users at same time. For example, several users are using same resources but in real life we don’t know how many people are using it.
7. Failure: hide failure and recovered of an object in distributed system. Users don’t know whether there are failures during the process, if we can inspect the message between the communication, we could find out some message say that sending out message fail and resend again.
8. Being open
9. Being open means that it’s easily for other resources to use this distribution.
10. Three characterized features to meat design goal
11. Interoperability: two implementations can work together well and not trigger any problem. Sometimes you will find out that some application launched will cause other application crash.
12. Composability: for example, when you have an implement in system A, and you then store in USB and store in System B, but it’s still work as same in System A
13. Extensibility: a system could be easily to replace any small features, and the system still performance constantly.
14. Be scalable
15. Scalability dimensions
16. Size scalability (three causes for the bottleneck of distributed system

* CPU which indicate as computational ability of computer
* The storage capacity, as the size of data capacity increasing rapidly, the storage capacity has been a big issue for recent computers, not just include the storage capacity, it also relates to the data transfer rate, since the size of data are increasing, therefore the data transfer rate has to increase as well.
* Network between user and core service, it’s also a big problem right now, more and more videos in website provide high quality option, but based on personal users’ network speed, users have to lower quality to make it more smoothly.

1. Geographical scalability: for this issue, it’s caused by synchronous communication process, since the clients are sending message to server at the same time, it has to wait until server send back a response, this will make transaction delay.
2. Administrative scalability: it’s simply called policy conflict, since a distributed system has different administrative policy, but if several root managers run the same domain at same time, how does the distributed system reply to this operation, therefore, it’s a bottleneck for the distribute system development.
3. Scaling technique
4. Hiding communication latencies: this can be simply defined as to do other works when it’s processing requesting for response. It’s used to design an asynchronous communication distributed system
5. Partitioning and distribution: which is split large size data into small pieces and then after all merge it, so for small size data transparent, it will have low responding time. This is also a important technique to do scaling.
   1. Types of distributed systems
6. High performance distributed computing
7. Cluster computing: cluster computing system is a kind of parallel constructor which all workstations are using same system. Those workstations are connected by reliable local network.
8. Grid computing: when the cases are more complex, it means the architectures are different for those workstations, in this case we would like to use grid computing which is combine all similar function resources into one layer and build a grid diagram to connect each layer.
9. Cloud computing: as the development of network technology, more and more distributed systems processes are invoking on the internet not in real pc. Below are three types of services in cloud computing

* Infrastructure-as-a-Service
* Platform-as-a-Service
* Software-as-a-Service

1. Distributed information systems
2. Distributed transaction processing

* Remote procedure call: it’s just a procedure call to remote server.
* It’s also relay on ACID properties

1. Atomic: all transactions are not divisible to outsider.
2. Consistent: all transactions don’t conflict systems violations.
3. Isolated: all transactions are not conflicted by each other
4. Durable: after the transaction commit, it will be not allowed to change in this transaction.
5. Pervasive systems:
6. Ubiquitous computing systems

* Distribution: devices are distributed as what distributed systems’ definition, and they are connected by network. And then easily combine those devices together to make it work as a coherent system.
* Interaction: it’s interaction between different types users, so when user input something, system have to react according to this user.
* Context awareness: one of difference compared to other traditional distributed system
* Autonomy: it’s kind of automatic reaction to application or response, like devices changes and so on. If the user plug in a new device in one small element of system, therefore it will be automatically added to ubiquitous system
* Intelligence: this is simply linked to AI concept, so when system counters some situations, it will solve it by Intelligence ability, actually it includes thousands of algorithms and models.

1. Mobile computing systems: although mobility is a main factor in pervasive system, but in real world there are distributed system using all mobile devices as nodes.
2. Sensor networks: mainly differences to other types of pervasive system is that sensor network is almost a collection of input devices not like other types pervasive system.
   1. Summary
3. In this chapter, we first talk about the definition and characterize features of distributed systems
4. Then we talk about goals for designing those kinds of distributed systems.
5. Last, we talk about different types of distributed systems in real world and talk about the differences.

**Chapter 2** **Architecture**

2.1 Architectural styles

a) Four most important architectures for distributed systems

* + - * 1. Layered architectures: it’s simply a kind of architectural which is combined with thousands of layers.

Pure layer organization

Mixed layer organization

Layered organization with up-calls

Layered communication protocols: this is a kind of protocols to show how to communicate between those layers. It generally allows to send message from one layer to one or more layers.

And this kind of service or protocol in the internet will be called TCP which is short for transmission control protocol.

Application layering will be divided into three logic levels

The application-interface level: it’s used to handle interface internal or external.

The processing level: it likes a connection between database and interface level.

The data level: simply we could say it’s database in the application.

* + - * 1. Object-based architectures and service-oriented architectures.

objected-based architectures: it provides a way to encapsulating data and methods to other object, for examples, it’s like client and server relation architectures.

Service-oriented architectures: it’s also called SOA, it’s same as objected-based architectures, but this time the component will be services. It also has state and methods.

* + - * 1. Resource-centered architectures

This architecture just treats the distributed system as a collection of resources, therefore for this distributed system, resources are removing or adding by components in remote way.

This architecture is normally used in web and called as Representational State Transfer which is RESTful architectures. It has four key characteristics.

All the resources are through a single naming scheme

All services have at most four operations which is PUT, GET, DELETE and POST

Messages which are sending to or receiving are self-described

After finish the operation, all the component under this service will be erase the data about the caller.

* + - * 1. Event-based architectures: there is event bus at center and then all components will be used to communicate with event bus.

2.2 Middleware organization

* 1. Wrappers
     1. A wrapper is a special component to be used to fit all interface to application since some interfaces are not working well in different applications. So, we need to make a cover, plug to fix the unsuitable problem.
  2. Interceptors: it’s like a software tool which it can be used to block the control progress of application and make others to execute.
     1. Request-level interceptor
     2. Message-level interceptor
  3. Modifiable middleware
     1. In some cases, we want to change the middleware, we maybe need to turn off that part first and make changes
     2. But we want to modify the middleware without shutting it down.

2.3 System architecture

* 1. centralized organizations
     1. simple client-server architecture: it’s a client-server model, every time a client send request to server and waiting for a reply from server, this architecture is client-server architecture.
     2. Multitiered architectures
        1. Two-tiered architecture: it’s two tier which is client machine and server machine, client machine normally includes user interface and some part of application, server machine includes database and some part of application.
        2. Three-tiered architecture: it’s not simple client-server, it has three tier: Client, Application server and Database server, so the client request from application server and then application server request data what client need from Database server, Database send back data to application server, at last return response to client.
     3. decentralized organizations: peer-to-peer systems (a system architecture which support horizontal distribution which is called peer-to-peer system)
        1. structured peer-to-peer systems: structured means it’s organized logically, for example, tree type, grid and so on. If we can tell what kind of structure it is then it’s structured peer-to-peer system. In our lecture we implement the distributed has table, it’s using hash table to store data.
        2. unstructured peer-to-peer systems: unstructured means random, for example, like graphs, we only know the vertex and edges, but we don’t know how it connect to each other.
        3. hierarchically organized peer-to-peer networks: it can be understood simply that peers have levels, super peers, weak peers. Main part structure is super peer, so all super peers connect to each other and then weak peer connect to super peer, it only can connect to one super peer.
  2. hybrid architectures; it’s not simple as client-server architectures or peer-to-peer network structure, a lot of distributed systems have mixed types which means they may have combine several types together.
     1. edge-server systems: content provider connect to internet service provider and then internet service provider connect to each client
     2. collaborative distributed systems; for example, like bit torrent. All clients will become one node of the system and cooperate with each other together to increase performance of distributed systems
  3. Example architectures
  4. The network file system
     1. Network File System: this network file system is generally used in Unix systems. If you used Unix system before, you will understand that every user log in Unix system will see a view of local file system and then all users could operate on those file system image, each time user modify stuff is a communication between the user’s system and server.
     2. In the NFS, the client-server communication is invoked by remote procedure calls. It’s how NFS client-server work from one machine to another machine. In Unix, we should remember some command in terminal, like mkdir, rm and so on, it’s all RPCs.
  5. The web
     1. Simple Web-based systems: we know simple client-server model before, then for web-based system, it’s a browser web-Server model,
        1. HTTP: hypertext transfer protocol, which it’s used to fetch content through browser.
        2. Uniform resource Locator (URL): it’s defined in chapter one before.
     2. Multitiered architectures: it’s an improvement of basic web-based architectures.
        1. Common Gateway Interface:(CGI) this interface is used to take user data as input.
  6. Summary

1. First, we talk about architectural styles
2. Then we talk about middleware organization
3. At last we talk about different types of system architectures and give some example architectures in real life problem.

**Chapter Three Processes**

3.1 Threads

1. Introduction to threads:
   1. Definition of process: it's defend as a program in execution
   2. Definition of thread: it just like process, but generally smaller than process, a program may include multiple threads and each thread can run independently.
2. Thread usage in non-distributed systems: first, we need to know how multiple process works, when process A done then block It and let process B start running
3. Thread implementation
   1. Drawback of user-level threading model: multiple threads go to one single schedule entity.
   2. In kernel mode, it will use one-to-one threading model
4. Threads in distributed systems: in distributed system benefits of threads is used to avoid block entire process when block system calls.
   1. Multithreaded clients
   2. Multithreaded server: three ways
      1. Multithreading: it’s parallel and blocking system calls
      2. Single-threaded process: non-parallel, it also blocks system calls.
      3. Finite-state machine: parallel and non-blocking system calls.

3.2 Virtualization

1. Principle of virtualization
   1. Virtualization and distributed systems
   2. Types of virtualization
      1. ISA: instruction set architecture, which is the interface between hardware and software, it divides into two parts, privileged instructions and general instruction. Privilege instructions are only allowed to be executed by operating system and general instructions can be executed by any program.
      2. Interface works on system calls
      3. Interface works on API, it’s working on library calls, when you work on java programming, you will find out a lot examples about API.
   3. Application of virtual machines to distributed systems
      1. Infrastructure-as-a-service (Iaas)
      2. Platform-as-a-service (Paas)
      3. Software-as-a-service (Saas)
      4. Network address translation: which is used to transfer public address into private one.

3.3 Client

1. Networked user interfaces: in this case, everything in communication between client and server is processed and stored in server side not client side
2. Thin-client network computing.
3. Client-side software for distribution transparency.

3.4 Servers

* + 1. General design issues: as what we know the client-server process, the services are always doing the same activity which is waiting for a request from client side. Therefore, how to make servers more efficiency in waiting requests will be a big design issue.
       1. Concurrent versus iterative servers:
          1. Iterative servers: each server is handling requests from client by itself.
          2. Concurrent servers: compared to iterative servers, the concurrent server is not directly handling requests by itself, instead it will pass the request to a thread, so the concurrent server is not just processing one request but using one thread handle one request and another thread handle another client’s request.
       2. Contacting a server: end points, as what we know that each server will listen to a specific end point which is also known as port, for example http for www will listen to port 80.
       3. Interrupting a server: this means that when client decide not to stop doing request, the problem is how to stop it for remote server. There are several ways to stop it, like physically disconnect the connection of internet
       4. Stateless versus stateful servers:
          1. Stateless servers: for example, like web server, it will forget where this request comes from when server finish its job.
          2. Stateful servers: for stateful servers, servers will be knowing the client information all the time and servers decide to delete the information of clients or not.
    2. Object servers: it’s compared to traditional server, object servers are not supply specific service, so the service support by object servers will depend on the data in that servers.
    3. Example: The apache web server: 50% more website are using the apache web server and, in this part, textbook point out why it’s so popular, because it’s configurable, extensible and independent in specific platforms
    4. Server clusters: at most time, server clusters are organized in local network, of course some server clusters will be organized through the world wide web.
       1. Local-area clusters: one type of server clusters that all servers are organized in local network.
          1. General organization: three tiers

Logical switch

Application or compute servers.

Distributed file or database system.

* + - * 1. Request dispatching: this is known as front end, for example that’s what we see in a browser web page, when user click on link in the web page it will send out a request, this is request dispatching. Absolutely request will pass through logical switch to decide to send request or not.
      1. Wide-area clusters: this can be think as lots of local server clusters are combined together into a big server clusters through internet.
         1. Request dispatching: for wide-area servers clusters, the request dispatching will be main issue, such as decide which server will be used to send request. It will be a complete progress for the request dispatching, after server is selected, then it will look up domain name system, then return IP address through DNS.
      2. Case study: PlanetLab
         1. General organization: it’s organized by vserver, operation system and hardware.
      3. Vservers: main difference to traditional servers are that Vserver is rely on shared operating system kernel mode not like normal virtual system machine.

3.5 Code migration

1. Reasons for migrating code: because designer will not want to make one machine is busy load and rest of machines are not using, this is kind of wastes of resources, therefore data should be transferred from one to another to increase the efficiency and performance of distributed system.
2. Migration in heterogeneous systems: when in complex systems, the code migration will be more complex. There are two big problem on migration.
   1. One is migrating the entire memory image.
   2. Another is migrating binding to local resources.
   3. Summary: This chapter is mainly talking about processes in distributed system. Then we talk about threads and what they do in processes and tell the differences between process and thread, after that the textbook discuss the components of the distributed system and how it works such as client, servers, and also gives out some examples in real world. After that we know the importance of code migrating and how it works.

Chapter four Communication

4.1 Foundations

1. Layered protocols: simple concept of communication in distributed systems, it’s just sending and receiving messages.
   1. The OSI reference model
      1. Physical layer: it’s the lowest level to deal with binary and the physically connection between two computers.
      2. Data link layer: to detect transmission problems.
      3. Network layer: handling network traffic
      4. Transport layer: control and handle real time data interaction.
      5. Session layer: handling sessions between applications
      6. Presentation layer:
      7. Application layer: top layers which is what we think as application we use.
2. Middleware protocols: as we know what’s the middleware in distributed systems, it’s an application in application layer, for example DNS, remote procedure calls are applications as middleware protocols in OSI reference model.
3. Types of communication
   1. Asynchronous communication: it’s simply defined as the communication process is using a global clock, it’s not in time phase, it might be delay to deal with information.
   2. Synchronous communication: this means the information are dealing at same time, for example like chat room, when two user speaks, they are send and receiving messages at the same time.

4.2 Remote procedure call: just call procedure in remote machines

1. Basic RPC operation: RPC is used to make remote call like it works like in local machine.
   1. Client stub: code transfers requests from network calls into local calls in client side.
   2. Server stub: code transfers requests from network calls into local calls in server side.
2. Parameter passing: it’s kind of packing code process from client stub by taking its parameters and then send them to the server stub. It’s also called as parameter marshaling.
3. RPC-based application support
   1. Stub generation: it’s a normal parameter structure in storage, given a function ID for given example in textbook, and parameter store afterwards.
4. Language-based support: compared to application support, language-based support will be easier in particular language.
5. Variations on RPC:
   1. Asynchronous RPC: it’s not same normally RPC, in asynchronous RPC, the call local procedure return from call time will be same as call remote procedure request. it can be considered as a delayed synchronous RPC.
   2. Multicast RPC: it’s one client to sending an RPC requests to a group of servers.
6. Example: DCE RPC: distributed computing environment
   1. Introduction to DCE: middleware system
   2. Goals of DCE RPC: same as RPC just make a remote call like call in local machine
   3. Writing a client and a server: simple steps to write a client and a server
   4. Binding a client to a server: how to combine client and server together to make works as a distributed system. The main problem will be the located the server machine.
   5. Performing an RPC: in actual code, each time message trans passing will include port which bind the client and server together.

4.3 Message-oriented communication

1. Simple transient messaging with sockets: for example, in java, it’s given a socket interface so you can use socket to communicate with client and server by given port number and name. check man page when doing programming, normal procedure is create and bind a socket, accept connections and send data to server or client, at last close the connection.
2. Advanced transient messaging
   1. Using messaging patterns: ZeroMQ; avoid bothering the conditions that each time client and server has to make a new socket, in that case it’s a kind of one-on-one pattern, using ZeroMQ pattern is like one-to-many or many-to-one, so it will improve the performance of internet connection a lot.
   2. The message-passing interface (MPI): there are two issues for high-performance message-oriented operations.
      1. For simple send and receive operations, sometimes it will be located at wrong level, since it’s high speed and in heavy load duty.
      2. TCP/IP is not suitable for high speed internet networks, because fast communication will cause big problem when it’s sending thousands of requests and server is handling thousands of replying. It can’t handle it.
3. Message-oriented persistent communication
   1. Message-queuing model: insert required sent message in a queue to send to servers
   2. General architecture of a message-queuing system: first, there is a queue manager, and then it will store message in a local queue, based on the information received by queue manager, the queue manager will decide the schedule of queue.
   3. Message brokers: it’s used to convert incoming message into some format that destination server could understand easily.
   4. Example: IBM’s WebSphere message-queuing system
      1. Overview
      2. Channels: each channel has one queue
      3. Message transfer: rule for message sending is FIFO, first in first out rule and given some limitation to message length, retry count and so on.
      4. Managing overlay networks: it’s an issue in large network systems.
   5. Example: advanced message queuing protocol (AMQP)
      1. Basics:
      2. AMOP communication
      3. AMOP messaging

4.4 Multicast communication

1. Application-level tree-based multicasting: put all nodes into an overlay network and then spread information to each component.
   1. Performance issues in overlays
   2. Relative Delay Penalty: delay between two nodes in the overlay.
   3. Link cost: create a link between each node, it will make a large cost if there are a lot of links for multicast communication.
2. Flooding-based multicasting: it can be simply known as a spreading information process, for example, a node A sending out information to node D, there are 2 nodes B, C in between, for flooding multicasting, it will past information from node A to node B first, and then passing to nearby node C, at last passing to node D. it will not directly send to node D.
3. Gossip-based data dissemination: spread like disease.
   1. Information dissemination models: like disease, but this time it’s not spread virus, it will spread information. Benefits for gossip-based data dissemination is easy to spread updates.
   2. Removing data: by the spread pattern of gossip-based data dissemination, the removing data is a big problem, because even though we delete data in one node, it maybe will get copy from other nodes as well.

4.5 Summary: this chapter is mainly covering the communication of distribution system, first it indicates the structure of basic cases, and it give some examples to inform the details how they interact with each other and spread information form start point to destination with several different types of ways.

**Chapter Six Coordination**

6.1 Clock synchronization

1. Physical clocks: timer or counter, for small system, it’s generally given a global counter which is can be used in the whole system.
2. Clock synchronization algorithms: when through internet, it will have a problem, if the server and client locations are not in same time zone, people give out a new concept time drift rate to synchronize the time.
3. Network time protocol: a protocol used to pair servers, and record down the offset of two servers, for example, if we know the server A, then we will know the server B time as well by given offset.
4. The Berkeley algorithm: it’s not like other algorithms, Berkeley algorithm the time is active, some other algorithm, the time is passive. So Berkeley algorithm will notify other servers to modify times.
5. Clock synchronization in wireless networks
   1. Reference broadcast synchronization: it’s a pretty special protocol, because most of algorithms will make sender and receiver’s times both synchronized. In this protocol, it only makes receiver synchronized.

6.2 Logical clocks

1. Lamport’s logical clocks: it’s working in two situations:
   1. If in same process, one must occur before another
   2. If in different process, a message can’t be received before it is sent.
   3. It’s not like normal logical times, so for Lamport’s logical clocks must rely on the rules of Lamport’s logical clocks. When it’s not suitable, then the algorithm will make changes to fix the conflicts.
2. Example: Total-ordered multicasting: this is avoiding the order problem that causes the transaction received are not the same, so it requires a same order to do the multicasting to make sure the results are same.
3. Vector clocks

6.3 Mutual exclusion

1. Overview: two different categories for distributed mutual exclusion algorithms.
   1. Token-based solutions mutual exclusion: it’s used to avoid starvation and but the problem is that when token lost, it will need to create another token.
   2. Permission-based approach: in this approach, if system want to run a process, it will need a permission to start it.
2. A centralized algorithm: it’s to elect process as the coordinator, then before access shared resources, contact coordinator first to check whether the process can get permission.
3. A distributed algorithm: like lamport’s algorithm, it will need a total-order schedule for all the events.
4. A token-ring algorithm: for this algorithm, the process will be formed as a circle ring and absolutely it will have a token passing through each process.
5. A decentralized algorithm: it’s opposite way compare to centralized algorithm, instead of only have one coordinator, this time each process has its own coordinator.

6.4 Election algorithms: election algorithm to select coordinator

1. Bully algorithm: for N processes, and kth process send a message to all processes that says it’s time to elect a coordinator, therefore, start from k+1, if it accepts coordinator will be k + 1 process, if not keep going.
2. A ring algorithm: to select a coordinator in this algorithm, that when processes start, it will send message to its successor.
3. Elections in wireless environments: start with one node and passing message to nearby and then it will be treated as its parent, keep doing this until acknowledges the receipt and find out the coordinator.
4. Election in large-scale systems: for example, in peer-to-peer system, so there are super peers, in our textbook, it will give a special list to store super peers.

6.5 Location systems

1. GPS: global position system: GPS system like google map, use GPS to locate users’ location.
2. When GPS is not an option: drawback of GPS, it’s available indoor.
3. Logical positioning of nodes: same as GPS, network has its own coordinates systems which is NCS.
4. Centralized positioning: given 3 node positions and use these three nodes to calculate one node’s actual location.
5. Decentralized positioning: if you learn algorithm course, then you will know how to calculate optimal results. It’s same as this positioning algorithm.

6.6 Distributed event matching

1. Centralized implementations: one core server handle all subscriptions and notifications.

6.7 Gossip-based coordination

1. Aggregation:
2. A peer-sampling services: a node selects another node randomly from rest available nodes
3. Gossip-based overlay construction: we know gossip-based election, it’s almost same way, like disease spread process.

6.8 Summary: in this chapter, it’s mainly talking about three things: clock, position system and event matching.