These notes have been created based on the notes by Dr. Steve Pettifer and Dr Alvaro Fernandes for their COMP18112 course. The Dr Pettifer's original material can be found on the Blackboard page for the course and Dr Fernandes' material can be found at `https://moodle.cs.man.ac.uk/course/view.php?id=256`.

If you would like to contribute, or have found any errors, please send a pull request or create an issue on the GitHub page for the notes: `https://github.com/Todd-Davies/first-year-notes`

Define bandwidth.

1

Define throughput.

2

On a graph diagram of a distributed system, what is represented by the nodes on the graph?

3

Name some properties of edges that connect the nodes in a graph of a distributed system.

4

Why is it that even though connections between nodes can be implemented in different ways (such as wifi or Ethernet), they can be treated as the same?

5

Name the eight axioms of distributed systems.

6

Why is transparency desirable in a distributed system?

7

What is transparency of location? How can we achieve it?

8

Throughput measures the actual rate at which messages are communicated.

2

Bandwidth measures the maximum amount of data that can be communicated within a certain amount of time.

1

The type of connection (wired/wifi/mobile data etc).
The bandwidth.
The latency.

4

The physical nodes of the network (individual systems). Each can host multiple processes and resources.

3

- *Latency is greater than zero.*
- *Bandwidth is less than infinite.*
- *Transport cost is greater than zero.*
- *There is more than one administrator.*
- *The network topology can and will change.*
- *The network is not homogeneous (the nodes and edges differ).*
- *The network is not secure.*
- *The network is not reliable.*

6

The implementation details of each connection is abstracted away by many layers of protocols.

5

An attempt to hide the need to know of where a specific resource is physically located.
Use DNS servers to map host names to IP addresses.

8

It allows us to design systems as though the distributed axioms were false.

7

What is transparency of migration? How can we achieve it?

9

What is transparency of relocation? How is it achieved?

10

What is transparency of replication. How is it achieved?

11

What is transparency of access? How is it achieved?

12

What is transparency of concurrency?

13

What is deadlock?

14

What is livelock?

15

What is transparency of failure? How is it achieved?

16

Transparency of relocation is when parts of the system move while they are being accessed. This is hard to mitigate, and is often a problem with mobile phone communications.

10

When a host moves location in the network, we shouldn't need to know the details of the move.
The DNS architecture implements this, though if a resource keeps moving, then the route through the network and therefore the latency of the connection to the host is hard to predict.

9

Transparency of access is the ability to not care about how a node is implemented. This is often achieved using protocols and API's and middleware.

12

When there is more than one physical resource that does the same job, which one do we use? It is hard to achieve.

11

When two different processes are unable to progress since each is waiting for information from the other.

14

Different users shouldn't need to know that others are using the same resource and may be competing for its time. Atomic operations and enforcing consistency are ways to achieve this, but this can force users to wait on each other (deadlock, livelock etc).

13

Users should not know that a specific node has failed or has recently had downtime. Hard to ensure, since sometimes slow connections are indistinguishable from failed nodes.

16

When two processes change with respect to one another so that neither can make progress.

15

Define systems software.

17

Describe the start of the mainframe era.

18

What advances were made during the mainframe era?

19

What is the danger of multiprogramming?

20

What three things can help multiprogramming be effective?

21

How does the scheduler make decisions about programs?

22

In the beginning of the PC era, what was a typical PC like?

23

What happens to the price of PC's during the PC era?

24

- Hardware was vastly more expensive than people.
- Only single user programs.
- Programs had to be written every time they were to be run.

18

The underlying level of software which allows applications to perform their tasks efficiently. Examples include device drivers and the operating system.

17

One program could potentially have access to data in other programs, which could present security/compatibility concerns. Memory protection is used so that programs can only read and write stuff they own.

20

- *The ability to store programs was developed.*
- *Batch processing, where jobs would be written to a queue to be executed. Lost the ability to debug.*
- *Use interrupt handlers and buffers to allow the CPU and IO to work in parallel.*
- *Multiprogramming so that one program has the CPU and another has the IO to enable full resource utilisation.*

19

Give them priority levels.

22

- *Fairness policies that impose limits on how much resources jobs should use*
- *Schedulers that aim to minimise the time it takes to complete jobs, by reducing their response and turnaround time.*
- *Preemptive scheduling will temporarily stop jobs if they are hogging resources.*

21

Price falls due to commoditization and competition.

24

Everything was cut back with just a single user in mind; no timesharing, multiprogramming or protection in the OS, few resources such as memory and CPU power etc.

23

How does the usage of PC's change during the PC era?

25

What does WIMP stand for and when was it introduced?

26

What drove an increase in security focus during the PC era?

27

What are the four standard layers of the network protocol stack?

28

When did middleware start to become widely used?

29

In the device era, CPU cycles, storage space and power is all limited. How is this overcome?

30

Since there are millions of requests sent to data centres in the device era, how is the demand handled?

31

What is the aim of the operating system with regard to resource management?

32

Windows, Icons, Menus, Pointer

The WIMP system became common during the PC era.

26

They become used for more simple tasks such as word processing, rather than number crunching and enterprise processing.

25

Name	**Example**
Application Layer	`HTTP`, `POP`, `SSL`
Transport Layer	`TCP`, `UDP`
Internet Layer	`IP`, `ICMP`, `IGMP`
Link Layer	`ARP`, `Ethernet`, `DSL`, `ISDN`

28

The fact that sharing data over networks was becoming increasingly common, which exposed security risks. Authentication and authorisation methods were therefore developed.

27

Make use of connectivity to servers to offload computation somewhere else. This is called the cloud.

30

At the beginning of the web era. API's are used to provide a generic interface between the OS and programming languages so that system calls are no longer needed. The OS must support middleware rather than implementing functions itself.

29

To make the most efficient use of the available resources as possible.

32

Elastically scalable architectures made up of commodity hardware are developed along with middleware such as hadoop and map reduce in order to allow for massively parallel computation.

31

How is memory access controlled by the operating system?

33

Why is it important that a process only have access to isolated areas of memory?

34

Each process executes in a sequential manner, what is a disadvantage of this?

35

What is a **scheduling policy** *with regard to operating systems?*

36

Describe forking.

37

Describe threading.

38

When there are many independent, self-sufficient, autonomous, heterogeneous machines working together with spatial separation, what is required to facilitate effective working?

39

A significantly long interconnect (i.e. a ________) is not ________.

40

So that processes cannot interfere with each others memory, because of a bug or malicious intent.

34

- *The OS assigns a* **unique identity** *to each process.*
- *Each process is then assigned an* **address space**
- *The OS ensures each process P has a single address space A that is exclusive to that process.*

33

A method of granting each process a time during which it is executed, enabling multiple processes to execute in a concurrent fashion, even though there may be only one physical set of resources.

36

If there was some task that took ages to do, say an API request over a slow network connection, then it might be sensible to execute other tasks (maybe preparing data for another API request) while the current task is executing (i.e. make the tasks concurrent). Due to the sequential nature of processes, this is not done.

35

When a new thread is created in a process:

- *A new child process is created*
- *The address space is shared between the child and the parent.*
- *The parent and child processes are less isolated, and can therefore interact together easier, however, this can be dangerous and a source of bugs.*
- *Since no memory is copied, threading is less expensive than forking.*

38

When a process forks (through an OS system call) the following happens:

- *A child process is created*
- *It is given a copy of the parent process's address space, though each address space is distinct.*
- *This causes two copies of the process to be active concurrently.*

37

A significantly long interconnect (i.e. a full blown network) is not cost-neutral.

40

Message exchange over a network, best facilitated with a middleware API to abstract away some complexity.

39

Describe an SISD architecture.

41

Describe an SIMD architecture.

42

Describe a MISD architecture.

43

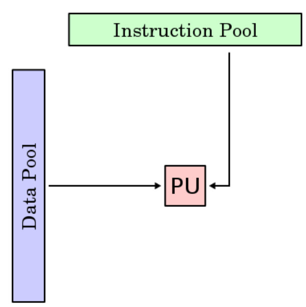
Describe a MIMD architecture.

44

Describe the advantages of a MIMD architecture.

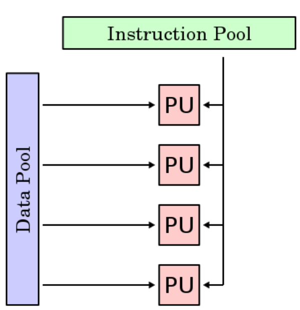
45

What architecture is this:



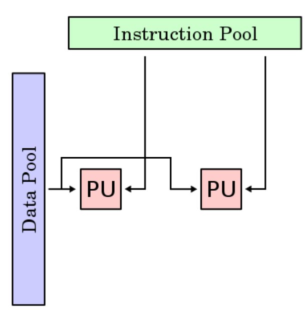
46

What architecture is this:



47

What architecture is this:



48

A single instruction/program, multiple dataset architecture. This could be used to describe a computer with a GPU that is applying **the same** *transformation to many pixels at once. Access to data is* **not** *cost neutral since the data is being shared amongst different instances of a program/instruction.*

42

A single instruction/program, single dataset architecture. An example of this is an older, single core PC, where there is only data space (the main memory) and there is cost neutral access to it.

41

A multiple instruction/program, multiple dataset architecture. Here, a form of interconnect will bind together several self sufficient autonomous components. The interconnect isn't cost neutral, but the gain in functionality is great. This architecture is widely used in applications such as clouds, clusters, and even multi-core PC's.

44

A multiple program/instruction, single dataset architecture. Though not as useful as other architectures, a sample MISD use case is in safety critical systems where two independent processors could compute the same function on the same data and check their results against each other.

43

SISD

46

A MIMD architecture allows you to scale out instead of up. An architecture that lets you scale out is one where you can add more hardware (such as servers/hard drives) as opposed to upgrading existing hardware which is scaling up (i.e. increasing the RAM in a server).

45

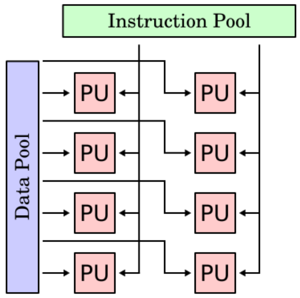
MISD

48

SIMD

47

What architecture is this:



49

MIMD architectures (aka ____ architectures) often communicate over a ____ such as ____ that exists ____ to the architecture itself.

50

What does vBNS stand for?

51

What does OC stand for? What does OC-n mean?

52

What is the network protocol stack?

53

What are the five layers of the protocol stack?

54

Describe the job of the application layer.

55

Describe the job of the transport layer.

56

MIMD architectures (aka shared-nothing architectures) often communicate over a network such as the Internet that exists separately to the architecture itself.

50

MIMD

49

OC stands for Optical Carrier.

$$OC\text{-}n = n \times 54.84 Mbit/s$$

52

Very high performance Backbone Network Service.

51

$$\begin{aligned} Application &\leftrightarrow Transport \\ &\leftrightarrow Internet \\ &\leftrightarrow Link \\ &\leftrightarrow Physical \end{aligned}$$

54

Layers of protocols that talk to each other to facilitate network connectivity.

53

To make sure that there is a logical channel for messages to flow through. E.g. TCP

56

To take care of specific application needs such as sending emails or HTTP requests.

55

Describe the job of the Internet layer.

57

What is the job of the link layer?

58

What is the job of the physical layer?

59

What is the hourglass model?

60

What is a packet switched network?

61

What is the job of a router in a packet switched network?

62

What are some differences between TCP and UDP?

63

What is packet encapsulation?

64

Making sure that there is a physical channel to pass data through. E.g. MAC

58

To make sure packets sent over the network reach their intended destination. E.g. IP

57

Very varied application layers at the top of the network stack (emails, streaming video etc), as simple protocols as possible in the middle of the stack, and very varied physical devices at the bottom (bluetooth, wifi, optical fibre etc).

60

To physically pass messages between networked components. Deals with all the physics etc. E.g. Wifi

59

To examine the header data in each packet and send the packet on to another router in the network that is closer to the packet's destination, while accounting for how saturated datapaths are in order to find an efficient route.

62

- *A network of networks formed by interconnecting routers.*
- *The message is divided into packets before transmission.*
- *The packets are reconstituted into the original message at reception.*
- *Packets may take varied routes while in transit, and be grouped with unrelated packages from other messages.*

61

Each layer of the networking stack wraps messages in a packet that contains information about the packet. By the time the packet has descended to the physical layer, then it may have many layers of headers and metadata.

64

- *TCP is connection oriented, UDP isn't.*
- *TCP is reliable and ensures that messages arrive at their destination, intact and in the right order. If the messages do not arrive, then it will be known. UDP only provides a corruption guarantee.*
- *UDP is more lightweight than TCP and is better for quick communication where the occasional loss or out of order packet isn't a problem.*

63

Why does packet encapsulation enable flexibility?

65

What does IPC stand for?

66

A successful communication between processes must consist of one process being ________ coinciding with another process being ________. This is non trivial with ________.

67

Describe Direct Message Exchange as a Distributed Architecture.

68

What is asymmetric DME?

69

What is a symmetric DME?

70

What is a MME architecture?

71

If a message in a MME architecture goes to more than one destination it is described as being a ________, if messages only go to one destination, then the system is described as being ________.

72

Inter-Process Communication

66

It doesn't matter about what information the packet is carrying, at each layer of the network stack, the callee layer has multiple choices of what protocol to use for the next level on the stack, and simply chooses the most convenient/suitable one.

65

In order to communicate, the sending process must be in a position to send and the receiving process must be in a position to receive messages. Upon receiving a message, a process may interpret it and then respond.

68

A successful communication between processes must consist of one process being ready to send coinciding with another process being ready to receive. This is non trivial with independent timelines.

67

Direct Message Exchange where no special roles are assigned to processes, also known as a peer-to-peer architecture. Event synchronisation is less simple with a symmetric DME than with an asymmetric one.

70

Direct Message Exchange where processes are assigned special roles such as a client or server in order to simplify the process. All the server has to do is wait for messages and all the client has to do is send messages.

69

If a message in a MME architecture goes to more than one destination it is described as being a public subscribe system, if messages only go to one destination, then the system is described as being point to point.

72

A Mediated Message Exchange architecture is one where inter process communication goes through a central middleware component (often referred to as message orientated middleware). Allows for event synchronisation to be decoupled in time since the middleware is always avalible.

71

What is RPC?

73

Why is middleware such as MME, RPC and RMI beneficial?

74

What is a protocol?

75

The operating system provides an ______ for IPC.

76

Name two important OS services relied on by an API for IPC in the OS.

77

Sockets are ______ the network stack provided by the operating system to abstract away the details of ______. In this manner, communicating processes ______ to each others sockets in an ______ style manner.

78

What the lifecycle of a client side socket?

79

How is a server side socket **set up***?*

80

They raise the level of abstraction at which we can build and engineer distributed systems.

74

A remote procedure call is similar to a message, though the data is passed as parameters to a method. It is often facilitated by middleware such as `CORBA`*. When OOP principles are involved, RPC is called* **Remote Method Invocation (RMI)**.

73

The operating system provides an API (application programming interface) for IPC.

76

A set of rules for inter process communication that stipulates the precise sequence of events that must be enacted by the communicating processes for them to successfully communicate.

75

Sockets are an additional layer on the network stack provided by the operating system to abstract away the details of low level IPC. In this manner, communicating processes read and write to each others sockets in an I/O style manner.

78

- *Keeping network buffers.*
- *Providing synchronisation mechanisms.*

77

1. *The socket is created for a given transport protocol (E.g. UDP) and Internet protocol (E.g. IPv4)*
2. *Set the options for the socket, such as whether it is reusable.*
3. *Bind the socket to the address and port to listen on.*
4. *Start listening for connections, set the maximum number of connections to leave in a queue if there is a significant load.*

80

1. *The socket is created for a given transport protocol (E.g. TCP) and Internet protocol (E.g. IPv6)*
2. *A connection is then opened with the desired IP/port combination.*
3. *The client then sends a request through the socket and waits for a response (through the same socket).*
4. *The server then replies with a response, which may come in chunks.*
5. *Once the response has been received, the socket is closed and discarded.*

79

How are connections handled in a server side socket?

81

When is the benefit most useful from massively parallelising a task?

82

When a task is divided into parts for execution on many machines, what challenges are presented in doing this?

83

Operations that seem may actually further down the software stack. This can cause problems if seemingly atomic operations are running concurrently, since they could further down the stack and interfere with each other.

84

What is a synchronisation primitive?

85

What is a barrier?

86

When can an expression be split into multiple parts?

87

We can split the following expression into `x + y` *and* `a + b` *to be parallelized. What is the blocking point?*

```
c = (a+b) * (x+y)
```

88

1. *The socket waits to accept a connection.*
2. *When a connection is accepted, a new socket is created with the IP address and port of the connecting client in it.*
3. *The server process then handles the new connection as appropriate, e.g. by spawning a thread and passing the socket to that thread.*
4. *The server process can then continue in its loop (back to stage 1).*

Note that spawned sockets and their handlers need to be closed properly after the client has disconnected.

81

When the data can be partitioned and multiple computers can work on the data independently then the performance benefits are most significant.

82

- *What if we have more workers than tasks, or more tasks than workers?*
- *How do we know if all the workers have finished?*
- *How do we aggregate the results of the workers?*

83

Operations that seem atomic may actually compile down into several different steps further down the software stack. This can cause problems if seemingly atomic operations are running concurrently, since they could interleave further down the stack and interfere with each other.

84

A language construct that provides access to a shared variable with a OS/hardware guaranteed atomicity. Used to enforce execution sequences.

85

A synchronisation primitive that forces multiple processes to wait at the barrier until they have all reached the barrier. For this to be efficient, all the processes must reach the barrier more or less at the same time.

86

When independent branches of the expression do not have any shared state, they can be executed in any order or parallelized.

87

The assignment to `c`.

88

When designing tasks for parallelized computation, the goal is to identify and eliminate as many as possible and eliminate them, so that the can be carried out on different machines as as possible.

89

How does map reduce make it easier to make tasks independent?

90

Why are functional programs easy to parallelize?

91

The `map` *function takes and and returns*

92

Why is `map` *parallelizable?*

93

`Reduce` *takes a and , and returns*

94

How can `reduce` *be parallelized?*

95

How is the map and the reduce stage synchronised with each other in Hadoop and other map-reduce engines?

96

By limiting the expressiveness of the computation of the tasks so they can be more easily parallelized.

90

When designing tasks for parallelized computation, the goal is to identify and eliminate as many synchronisation points as possible and eliminate them, so that the independent tasks can be carried out on different machines as efficiently as possible.

89

The `map` *function takes a collection* `[i_1, ..., i_n]` *and a function* `f` *and returns the collection* `[f(i_1), ..., f(i_n)]`

92

Since pure functions are side effect free, meaning that two functions used in the same expression are guaranteed not to share state and so can be parallelized safely.

91

`Reduce` *takes a binary associative function* `f(x,y)` *and a collection* `[i_1, ..., i_n]`*, and returns* $f(i_n,f(\dots,f(i_0, i_1))\dots)$

94

Since the function applying to the collection can be applied to each item in the collection independently, and is side effect free.

93

With a barrier.

96

Have different parts of the list run in different tasks and then apply the function to the result of tasks to combine them together again at the end.

95

What useful features do map-reduce engines such as Hadoop implement?

97

Just look over this one. Make sure you understand how the map-reduce architecture works.

98

If you have a network built of identical worker nodes, each working on a partition of some dataset, then the computation is ________*, and could be referred to as a form of* ________ *strategy. However, if you have a network built of heterogeneous workers that are operating on data that are not partitions of a single dataset, then you have* ________ *computation, which is a case of* ________.

99

A distributed system working within one company is referred to as an ________ *(EAI) system.*

100

A distributed system spanning multiple companies is referred to as an ________ *(B2B) system.*

101

What is an RIA?

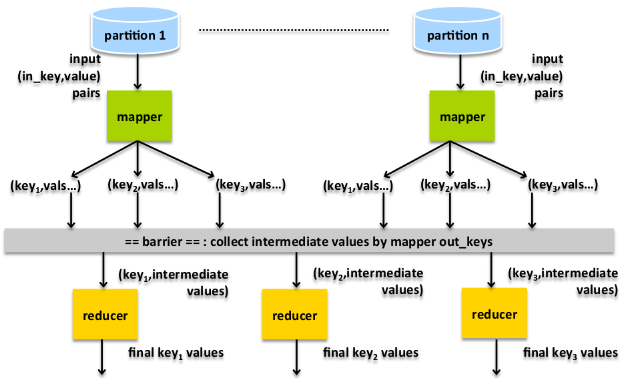
102

What is a web mashup?

103

The `WS-*` *view is a family of standards that all start with '*`WS-`*'. It is an example of a* ________ *(SOA), and interactions are defined by a* ________ *protocol called* ________. `WS-*` *interactions sit on top of other protocols such as* `HTTP` *or* `SMTP`.

104



98

- *Load balancing*
- *Fault tolerance*
- *Splitting, spawning and merging of tasks*

97

A distributed system working within one company is referred to as an enterprise application integration (EAI) system.

100

If you have a network built of identical worker nodes, each working on a partition of some dataset, then the computation is not inherently distributed, and could be referred to as a form of divide-and- conquer strategy. However, if you have a network built of heterogeneous workers that are operating on data that are not partitions of a single dataset, then you have an inherently distributed computation, which is a case of unite-and-conquer.

99

A Rich Internet Application (e.g. Gmail)

102

A distributed system spanning multiple companies is referred to as an business to business integration (B2B) system.

101

The `WS-*` *view is a family of standards that all start with '*`WS-`*'. It is an example of a Service Oriented Architecture (SOA), and interactions are defined by a higher up protocol called SOAP.* `WS-*` *interactions sit on top of other protocols such as* `HTTP` *or* `SMTP`*.*

104

A lightweight web application that combines processing and data resources from different sources into a single application.

103

What does REST stand for?

105

How does REST operate?

106

In a RESTful service, interactions between components are ______, and the service is seen as a ______ (______).

107

What (according to the lecture slides) features do SOAP API's offer to justify them being more heavyweight?

108

This is the real difference between REST and SOAP API's.

109

In the Web of Data view, the web is ______.

110

Why is the data exposed by API's often hard to use with processing algorithms?

111

The Web of Data (WoD) is a ______, similar to a relational database, but with no predefined ______. The ______ between ______ define relationships in the data.

112

The sender places a payload into a request such as `HTTP Post`*, which is then sent to a receiver. The receiver will then respond via the same method with the response (if any).*

106

REpresentational State Transfer.

105

- *Security, authentication and authorisation*
- *Robust addressing and reliable messaging*
- *Transactional semantics (fault tolerance features)*

The above are very useful in EAI and B2B contexts.

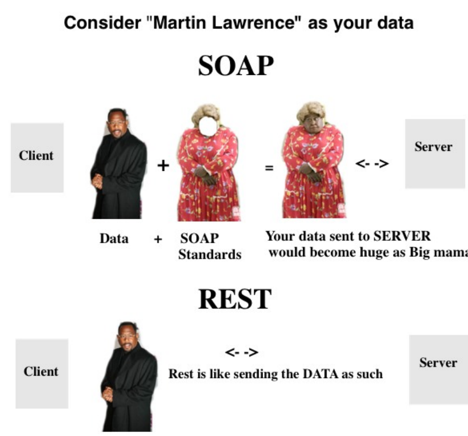
108

In a RESTful service, interactions between components are operations on resources, and the service is seen as a Resource Oriented Architecture (ROA).

107

In the Web of Data view, the web is a massive distributed database.

110



109

The Web of Data (WoD) is a graph, similar to a relational database, but with no predefined schema. The edges between nodes define relationships in the data.

112

Even though the data might be able to be parsed (i.e. if it's in JSON), the semantics of the data are still unclear since different API's follow different specifications and therefore it is hard to extract useful, unambiguous data.

111

What does RDF stand for? What is it?

113

In the WoD, we have ____ data where resources are URI-assigned nodes, linked by ____.

114

What does OWL mean?

115

Describe the **follow your nose** *approach to querying the web of data.*

116

What is data source federation?

117

MAC addresses are made up of a ____ number purchased by manufacturers from the ____, and another ____ number issued by ____. In this way, the ____ needs to ensure only that the second number needs to be unique.

118

MAC address stands for? Why is it silly to call a MAC address an 'address'?

119

An IP (____) address serves two purposes; as a ____ and as an indication of ____. The ____ (____) assigns blocks of addresses to regional authorities, meaning that IP addresses do have some ____ relationship.

120

In the WoD, we have linked data where resources are URI-assigned nodes, linked by explicitly declared relationships.

Resource Description Language. There are various implementations of RDF, such as XML, turtle format, N-triples format etc.

- *Select and connect to an initial seed URI.*
- *Send it your declarative query.*
- *Iterate through the triples in the result and decide whether to hyperlink the triple.*
- *Now you can do more queries if needed to expand your results set.*

Web Ontology Language

MAC addresses are made up of a 24 bit OUI number purchased by manufacturers from the IEEE, and another 24 bit NIC number issued by the manufacturer. In this way, the manufacturer needs to ensure only that the second number needs to be unique.

Middleware that emulates one virtual database from many physical (local) data sources. It does this by constructing many smaller queries on it's local data sets based on a global query that it has been given and then compiling the results together as a response. This removes the need for decision making such as in the follow your nose approach to traversing the WoD.

An IP (Internet Protocol) address serves two purposes; as a unique identifier and as an indication of where a device is on a network. The IANA (Internet Assigned Numbers Authority) assigns blocks of addresses to regional authorities, meaning that IP addresses do have some geographical relationship.

Media Access Control

An address contains information about the whereabouts of an entity. A MAC address is really just a Unique Identifier.

What does DNS stand for?

121

When a DNS server is given a ______ to translate into an ______ address, it looks to see if it already has the answer in its cache and returns it if so. Otherwise, it will ______ the hierarchy until it does find an answer.

122

How does the DNS system control how caching is used?

123

What does `HTTP` stand for?

124

The web is a ______ protocol in order to save the resources that would be required if a ______ was to be left open for the duration of a session.

125

If `HTTP` is a stateless protocol, then how is state retained on the web?

126

What is the function of the `HEAD` HTTP verb?

127

What is the function of the `GET` HTTP verb?

128

When a DNS server is given a domain name to translate into an IP address, it looks to see if it already has the answer in its cache and returns it if so. Otherwise, it will propagate the request up the hierarchy until it does find an answer.

122

Domain Name Server.

121

HyperText Transfer Protocol

124

Each DNS record has a TTL (Time To Live), which dictates how long the server should retain the record in its cache before replacing it from higher up the hierarchy. TTL values can be from seconds up to 68 years, or never.

123

The web server retains the state, not the `HTTP` *protocol.*

126

The web is a stateless protocol in order to save the resources that would be required if a connection was to be left open for the duration of a session.

125

Gets a resource from the server.

128

Provides metadata about a resource. Just like a `GET` *request, except the body of the request is not returned. Used to let clients know if cached data has changed.*

127

What is the function of the `POST` *HTTP verb?*

129

What is the function of the `OPTIONS` *HTTP verb?*

130

What does the 'store and forward' model mean?

131

What is the SMTP protocol?

132

What is the POP protocol?

133

What is IMAP?

134

What are the four Coffman conditions?

135

What is deadlock?

136

Returns the list of commands supported by the server.

130

Submits data to the server. The state of the server may change as a result of a `POST` *request.*

129

The Simple Mail Transfer Protocol is a method for sending email. It is a text-based, stateful, connection based protocol, so a connection to the server must be opened before the message can be transferred, and then closed again after.

132

When a message needs to be transmitted along a network, but not all the nodes along the path may be active at the same time. Consequently, the message is stored by each node until it has been confirmed to have been received by the next node.

131

The Internet Message Access Protocol is an email protocol that lets you see the head of a message rather than just having to download the whole thing. This means you can more easily cache emails on a client and then download any changes that have occurred. It also lets users have multiple inboxes, server side searching and it deals with attachments better than POP.

134

The Post Office Protocol. Lets you read and delete emails from a server. If a message is not deleted, then it will stay unread for the next time (though a client may choose just not to download it next time). Like SMTP, POP is text based, stateful and connection based.

133

A deadlock is a situation in which two or more competing actions are each waiting for the other to finish, and thus neither ever does.

136

Mutual Exclusion	*There must be at least one resource that cannot be shared.*
Hold and Wait	*A process is currently holding one (or more) resource(s), and is waiting for another resource currently held by another process.*
No preemption	*Nothing can force a process to give up a resource unless it does so willingly.*
Circular wait	*There must be a cycle of dependencies so that processes are waiting for each other to relinquish resource control before they themselves do.*

135

What is livelock?

137

What is resource starvation?

138

What is the difference between thin and thick clients?

139

Describe 'reduction to absurdity'.

140

Give an example of 'reduction to absurdity'.

141

With a Lamport clock, how is the clock of A affected if B sends A a message?

142

Describe the purpose of a mutex.

143

What is a critical section?

144

Starvation is where a process is perpetually denied necessary resources. Without those resources, the program can never finish its task.

138

Livelock is similar to deadlock except that the states of the system constantly change with regard to each other, but they never actually progress.

137

This is a type of argument where an idea is shown to be false by showing that an absurd result can be obtained by the acceptance of the idea.

140

A thin (or dumb) client will do little processing, and merely just echo or display whatever the server tells it to. A thick client on the other hand will have functionality built into it that is able to work independently of the server.

139

```
if LCy < (LCx + 1):
    LCy = LCx + 1
```

142

- *Assume that a successful coordinated action occurs as a result of messages being passed with communicating parties.*
- *Assume the last message in the sequence is lost.*
- *If the action still succeeds, then the last message was unimportant.*
- *Repeat until no messages are exchanged.*
- *Our initial assumption was clearly wrong!*

141

A critical section is a sequence of actions that should execute in an atomic matter. That is to say that if the critical section does not succeed, then the state of the system should be exactly the same as when the critical section started.

144

A mutex ensures that two processes do not enter into a critical section of their execution at the same time and therefore prevents race conditions and enables the safe sharing of resources.

143

Describe the two-phase commit.

145

Describe the bully algorithm.

146

Describe Cristian's algorithm?

147

Describe the Berkly algorithm.

148

What is an overlay network?

149

How was Napster's distributed architecture organised?

150

How was Gnutella's network organised?

151

How was KaZaa's network organised?

152

1 A node wants to be a coordinator, so it sends a message to all the other nodes with it's number.
2 All the other nodes will only reply if they have a higher number. If they do, they will start another election.
3 The process continues until a node receives no replies, and is therefore the coordinator, at which point, it will inform all the other nodes of this.

146

1 A coordinator node wishes all of the nodes to perform an action.
2 It sends a request to all of the other nodes.
3 The other nodes attempt to perform the action, and reply with `commit` *or* `abort` *depending on whether they to performed the action.*
4a If any one of the nodes replied `abort`*, then the coordinator node will reply* `global_abort` *and all the nodes will roll back their state to that of the start of the transaction.*
4b If all the nodes replied `commit`*, then the coordinator node will send* `global_commit` *and the nodes will retain their current state.*

145

1 A master node is chosen.
2 The master polls the other nodes for the time.
3 The master observes the round trip time for each state, and estimates the time of each client.
4 The master then averages out the clock times and sends each node a message containing the difference that it is from the average.

148

In order to synchronise the clocks of two systems, one system S will send it's time to another system C. After receiving the request from S, C will then reply with it's own clock time. When S receives this, then it will set its clock to be the time that C sent added to the round trip time (discernible from C's reply) divided by two.

147

Napster had a central server which acted as a file directory/search engine that enabled peers to connect to each other and exchange files.

150

A virtual network that is superimposed onto an existing network. Nodes in an overlay network can be thought of as connected by virtual links that may be composed of many actual physical links in the real network.

149

- *There were two types of nodes; normal and supernodes.*
- *Nodes would be promoted to supernodes if they had sufficient uptime and bandwidth (though they could demote themselves if they didn't want to be a supernode).*
- *Supernodes would take on higher level tasks for the child nodes, such as indexing files and sharing data with other supernodes.*
- *Supernodes improved querying substantially, since failed queries needed only to go around the supernodes, not all of the nodes.*

152

- *To join, you have to have the IP address of a node in the network.*
- *Queries for files are 'flooded' around the network, with peers asking their neighbours if they don't have the file.*
- *This was inefficient, since if the peers didn't have the file, then the request would go around the whole network before failing.*

151

Why has BitTorrent been immune to successful prosecution?

153

How does BitTorrent get people to seed content?

154

How does BitTorrent work?

155

Describe BitTorrent's choke/unchoke policy.

156

A hash function maps a ____ to a ____, which then allows you to quickly index into a table to find the result you want.

157

In a ring based hash table, if there are more buckets than nodes, then how can we assign all the buckets to at least one node?

158

In a distributed hash table, how many keys is each node responsible for if there are N nodes and K keys?

159

In Chord, a ____ is created so that queries don't have to go all the way around the ring. The i^{th} entry of a nodes finger table contains the address of the successor $(n + 2^{i-1}) mod 2^m$.

160

If a client limits its upload speed, then the protocol will throttle the download speed to the client accordingly with a 'more you give, more you get' philosophy.

154

The BitTorrent protocol doesn't know about the contents of files, only their hash code. This means you can't search for torrents based only on the files themselves (though you can use a third party indexing service such as pirate bay). Also, the protocol is completely distributed, so BitTorrent can be argued to be just an efficient file sharing mechanism rather than a copyrighting infringing mechanism.

153

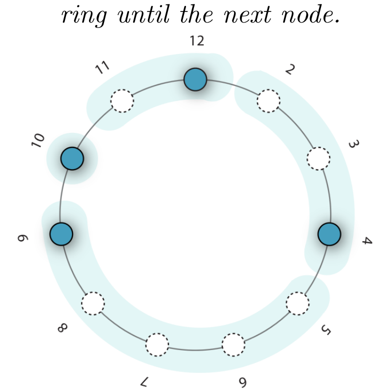
- *A peer serves the 4 best connected peers in its peer set at any one time.*
- *Every ten seconds, the worst of the four is choked if there is a better peer*
- *Every thirty seconds, a random peer from the set is unchoked in the hope that it offers good speeds.*

156

- *Files are split into blocks, by default* `256Kb`.
- *A torrent file is created which contains metadata about the file, and a link to a tracker server which will tracks active clients.*
- *A client will contact the tracker for a swarm of other clients downloading the file, these other clients are called the peer set.*
- *If the number of peers in the set drops below 20, then more will be requested from the server.*

155

Make nodes responsible for all the buckets below them in the ring until the next node.



158

A hash function maps a key (the thing youre searching for) to a number, which then allows you to quickly index into a table to find the result you want.

157

In Chord, a finger table is created so that queries don't have to go all the way around the ring. The i^{th} entry of a nodes finger table contains the address of the successor $(n + 2^{i-1}) mod 2^m$.

160

$\frac{K}{N}$

159

What are the four components of a game engine?

161

What were some of the limitations of Doom's gameplay?

162

How did Quake implement online gameplay?

163

What is client side prediction?

164

Why is the Unreal game engine so good?

165

When answering questions, Alvaro wants us to...

166

The speed of the gameplay was limited to the speed of the slowest connected computer over the network that was being played on. The P2P architecture also meant there was no saved state if all the players disconnected.

162

- *Rendering engine*
- *Physics engine*
- *Networking layer*
- *Behaviour module*

161

Sending information to change the state of a client rather than sending the whole new state in order to save bandwidth. E.g. Sending a trajectory instead of sending a position ten times per second.

164

There was a server (dedicated or participant) that had the full game state, which allowed clients to perform at their own rates, and games to persist if clients left. Client side prediction was utilised to reduce network load.

163

Describe, explain, justify, contrast

166

It allows developers to 'tear off' aspects of the game to be processed entirely on the client side, and thus saving bandwidth and server CPU cycles. Such aspects may be animations such as explosions or characters dying.

165