1) application architecture - designed by the application developer and dictates how the application is structured over the various end systems.
2) client-server - an always-on host, called the server, which services requests from many other hosts, called clients.
3) data center - housing many hosts, is often used to create a powerful virtual server.
4) P2P - direct communication between pairs of intermittently connected hosts, called peers.
5) self-scalability - property of a system to handle a growing amount of work by adding resources.
6) ISP friendly – file distribution that ISP's agree with.
7) P2P incentives – convincing users to volunteer bandwidth, storage, and computation resources.
8) Messages – how two different end systems communicate with each other across the computer network.
9) API - interface between the application layer and the transport layer within a host.
10) IP address – how a host is identified on the internet.
11) port number - a way to identify a specific process to which an Internet or other network message is to be forwarded when it arrives at a server.
12) reliable data transfer - guaranteed data delivery.
13) loss-tolerant applications - multimedia applications that can tolerate some amount of data loss.
14) bandwidth-sensitive apps - Applications that have throughput requirements.
15) elastic applications - can make use of as much, or as little, throughput as happens to be available.
16) TCP connection - After the handshaking phase, a TCP connection is said to exist between the sockets of the two processes.
17) UDP - lightweight transport protocol, providing minimal services. UDP is connectionless, so there is no handshaking before the two processes start to communicate. UDP provides an unreliable data transfer service.
18) application-layer protocol - defines how an application's processes, running on different end systems, pass messages to each other.
19) HTTP - the Web's application-layer protocol. implemented in two programs: a client program and a server program.
20) web page – documents that consists of objects.
21) HTML file – a webpage coded in html that can be displayed in a web browser.
22) web browser - implement the client side of HTTP.

23) web server - implement the server side of HTTP.

24) stateless protocol - maintains no information about the clients.

25) non-persistent connection - each TCP connection is closed after the server sends the object—the connection does not persist for other objects.

26) persistent connection - the server leaves the TCP connection open after sending a response.

27) RTT - the time it takes for a small packet to travel from client to server and then back to the client.

28) request message – a message sent by an http client to an http server.

29) request line - The first line of an HTTP request message. The request line has three fields: the method field, the URL field, and the HTTP version field.

30) header lines – lines after the request line that let the client and server pass additional info.

31) response message – a response returned by the http server to a http client.

32) status line - has three fields: the protocol version field, a status code, and a corresponding status message.

33) entity body - the meat of the message—it contains the requested object itself.

34) Cookies - allow sites to keep track of users.

35) set-cookie – a header which contains the identification number.

36) web cache - also called a proxy server—is a network entity that satisfies HTTP requests on the behalf of an origin Web server.

37) proxy server – web cache.

38) conditional GET - a mechanism that allows a cache to verify that its objects are up to date.

39) if-modified-since - a HTTP header that is sent to a server as a conditional request. If the content has not changed the server responds with a 304 status code. If the content has changed the server responds with a 200 status code.

40) control connection - the client sends commands regarding session state changes.

41) data connection – used to send a file.

42) out-of-band - data transferred through a stream that is independent from the main in-band data stream.

43) in-band - involves managing devices through the common protocols such as telnet or SSH, using the network itself as a media.

44) State – an instance of a network.

45) USER – username, Used to send the user identification to the server

46) PASS – password, Used to send the user password to the server

47) LIST – Used to ask the server to send back a list of files

48) RETR - filename: Used to retrieve (that is, get) a file

49) STOR - filename: Used to store a file

50) user agents - software that is acting on behalf of a user.

51) mail servers – house user mailboxes

52) SMTP - e-mail's principal application-layer protocol. provides for the transfer of e-mail messages).

53) Mailbox – a location in one of the mail servers.

54) message queue -where messages are held.

55) SMTP handshake - SMTP clients and servers introduce themselves before transferring information. the SMTP client indicates the email address of the sender (the person who generated the message) and the e-mail address of the recipient.

56) HELO - SMTP command sent by an e-mail client when connecting to an e-mail server.

57) MAIL FROM – specify sender

58) RCPT TO – specify recipients.

59) DATA – specify the body

60) QUIT – terminates conversation with server

61) mail message formats

62) From - The email address of the sender.

63) To - The email address of the recipient.

64) Subject - A short summary of what the message is about

65) POP3 - A mail access protocol used to transfer mail from the recipient's mail server to the recipient's user agent.

66) IMAP - A server that will associate each message with a folder

67) socket - One endpoint of a two-way communication link between two programs running on the network.

68) UDP - connectionless and sends independent packets of data from one end system to the
69) other, without any guarantees about delivery

70) socket() - The first parameter indicates the address family. The second parameter indicates the type of socket

71) sendto() - attaches the destination address (serverName, serverPort) to the message and sends the resulting packet into the process's socket
72) recvfrom() - extract the clientside (source) port number from the segment it receives from the client

73) close() - closes the socket and terminates the process

74) bind() - assigns) the port number 12000 to the server's socket.

75) connect() - initiates the TCP connection between the client and server.

76) send() - sends the string sentence through the client's socket and into the
77) TCP connection - a full-duplex connection in that two processes can send messages to each other over the connection at the same time.

78) recv() - receives characters from the server

79) listen() - server listens for TCP connection requests from the client.

80) accept() - accepts a TCP/UDP connection.