1. In Parallel Computing, processes use a shared memory and they communicate via the shared memory. While in Distributed Computing, every process uses independent memory, and they communicate by network.

   The Youtube platform is a distributed system.

   Design considerations:
   No global clock: There are servers of Youtube all over the world, and they do not have a common clock. So the systems in the network communicate by messages rather than time-based events.

   Geographical distribution: In order to serve users globally, Youtube uses CDN to deliver web content. CDNs replicate content in multiple places and caches popular videos so users can access videos faster.

   No shared memory: There is no shared memory in the distributed system of Youtube, but it can communicate through the network. Also, distributed systems avoid the problems created by shared memory. The users of Youtube are growing fast, so the memories and resources needed increase very fast. If we use a parallel system, we might run out of resources. But we can add sub-system in distributed systems to fulfill this requirement.

   Independence and heterogeneity: Sub-systems in distributed systems are independent, which are more reliable. The service provided by Youtube needs to be reliable, if one part of the system breaks down, we do not want it to influence other services, and we need other modules to take the responsibility of this service. In a Distributing system, each sub-system is more independent compared to Parallel Computing, so when one module fails, it applies less impact to the whole system, and other sub-system can provide the service instead. Also, since every system has different amounts of resources and processing speed, it is better to calculate and assign different tasks for each system.

   Fail-over mechanism: Youtube uses consistent hashing to distribute data across nodes. When one node breaks down, there are only a small set of nodes that need to be adjusted to take over the dead server. And we can use methods to detect failures, and make the system tolerate failure and recover from failures better.

   Security concerns: Compared to parallel systems, distributed systems are more secure since there are no shared memories. And we can increase the security level by applying encryption techniques.

2. 13.22
   There are four layers in the TCP/IP software stack. The first layer is Application layer, which contains the protocols of HTTP, FTP, TELNET. The second layer is Transport layer, which contains the protocols of TCP, UDP. The third layer is Network layer, which contains the protocol of IP. The fourth layer is Link layer, which uses device drivers to communicate.

TCP is a connection-oriented protocol, it provides reliable link between computers. TCP establishes a connection before transmitting, provides flow control and acknowledgement of data, retransmits the lost data, and closes the connection after transmission. The drawback of TCP is the transmitting speed is slow.

UDP is a datagram-oriented protocol, it transmits data between computers with no guarantee. UDP is faster, simpler and more efficient than TCP, but also it is less reliable than TCP.

13.23
Port is a virtual point to represent a particular process running on a computer, and port is represented by a 16-bit integer.
Well-known ports:
ftp: 21/tcp
ftp is the file transport protocol, which can be used to transmit files between computers through TCP protocol. It can be used in desktops, servers, mobile devices communication.
telnet: 23/tcp
telnet is an application protocol between two computers to communicate in a two-way, collaborative and text-based way. It can be used in servers, routers, and switches communication.
http: 80/tcp, udp
http is Hypertext Transfer Protocol, which is an application layer protocol, and can be used to communicate in distributed, collaborative, hypermedia information systems.

3. There are many functional requirements in twitter services, like posting new tweets, following users, marking tweets as favorites and so on. And it is impossible that these functions all run in a single machine, since the resources offered by one computer is limited. So we need to deploy these services in multiple machines. And providing all the services is the goal of the twitter system.
   There are many non-functional requirements in twitter services as well, like high availability, acceptable latency and consistency. These are the transparency requirements of the twitter system.
   They are important because in order to provide a full-featured, available and reliable service, a distributed system should provide distributed services as much as possible as a single system.

goals	transparencies
Users should be able to post new tweets.	Our service needs to be highly available.
A user should be able to follow other users.	Acceptable latency of the system is 200ms for timeline generation.
Users should be able to mark tweets as	Consistency can take a hit (in the interest

favorites.	of availability); if a user doesn't see a tweet for a while, it should be fine.
The service should be able to create and display a user's timeline consisting of top tweets from all the people the user follows.	
Tweets can contain photos and videos.	
Searching for tweets.	
Replying to a tweet.	
Trending topics – current hot topics/searches.	
Tagging other users.	
Tweet Notification.	
Who to follow? Suggestions?	
Moments.	

4.

1.12

Arguments for allowing concurrency:
When there are idle machines, we can make use of all the resources by running the program concurrently.
Considering the total amount of work is the same, it is much faster to finish the task by executing the instructions simultaneously.

Arguments against allowing concurrency:
It is more complex to write the program.
Need more resources to execute the task.
Might create errors during the execution, for example, multiple threads execute on the same period of data simultaneously.

example of interference:
If there are two threads executing the instruction of adding the number by 2, and the two threads execute on the same number 5 in an object simultaneously, then the number could become 9, which is not our expectation.

how to prevent interference:
we need to avoid different threads using shared resources simultaneously. This can be achieved by using semaphores, which is to lock the resource when there is a thread executing.

1.13

If speed is the first priority, the service might trade more memories for speed. If the memories are limited, the servers might trade other resources for memories.

No, we cannot multicast all requests to achieve mobility transparency. Because there would be much redundant work in this way, some servers would receive the same request. And if there are rules to assign each server its responsible requests, then it would break the mobility transparency.

Reference:
https://www.techtarget.com/searchnetworking/definition/Telnet
https://en.wikipedia.org/wiki/Telnet
https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol