1. **What properties should the entity identifiers/names in distributed systems have? What are the differences between names and addresses?**

   - Each identifier refers to at most one entity
   - Each entity referred to by at most one identifier
   - An identifier always refers to the same entity (no identifier reuse)
   - The address of an access point should not be used to name the entity (addresses can be re-assigned to other names and vice versa), so a name for an entity should be independent from its addresses

2. **Compare the differences between name servers deployed at the global layer, the administrational layer, and the managerial layer of a large-scale name space.**

Item	Global	Administrational	Managerial
Geographical scale of network	Worldwide	Organization	Department
Total number of nodes	Few	Many	Vast numbers
Responsiveness to lookups	Seconds	Milliseconds	Immediate
Update propagation	Lazy	Immediate	Immediate
Number of replicas	Many	None or few	None
Is client-side caching applied?	Yes	Yes	Sometimes

3. **Discuss the advantages and disadvantages of iterative and recursive name resolution, respectively.**

   Recursive resolution demands more than iterative resolution on each name server.
   However, recursive name resolution also has two advantages over iterative resolution:
   - Caching is more effective than iterative name resolution
     - Intermediate nodes can cache the result
     - With iterative solution, only the client can cache
   - Overall communication cost can be reduced

4. **Explain the working of the network time protocol (NTP) and the Berkeley Algorithm, respectively. Discuss the differences between the two algorithms.**

   The Network Time Protocol (NTP) is a networking protocol for clock synchronization between computer systems by synchronizing a server with a time server which is supposed to have the accurate time.
   The Berkeley algorithm is a method of clock synchronization in distributed computing which assumes no machine has an accurate time source, thus does not require the time agreed on to be the actual time.

5. **What are the limitations of Lamport's logical clocks, and how does the vector clock algorithm solve the problems?**

Lamport's algorithm only gives a partial order of events, from which we cannot derive a total order. Specifically, in this algorithm, if C (a) < C (b), it is not necessarily true that a → b. Vector clock addresses this limitation by letting each process Pi maintain a vector VCi with the following two properties:
- VCi [i] is the number of events that have occurred so far at Pi. In other words, VCi [i] is the local logical clock at process Pi.
- If VCi [j]=k then Pi knows that k events have occurred at P. It is thus P's knowledge of the local time at Pj.
In this way, all causally related events, assigned a time each, are in a certain order.

6. **Explain why mutual exclusion is needed when machines in a distributed system concurrently access shared resources. Compare the advantages and disadvantages of token- and permission- based approaches for mutual exclusion.**

Because concurrent access may corrupt the resource or make it inconsistent, we need an approach to decide which process gets the permission and other processes should be blocked.

Token-based approach:
- Only 1 token is passed around in the system
- Process can only access when it has the token

Advantage:
- Easy to avoid starvation and deadlock

Disadvantage:
- Situation becomes complicated if token is lost

Permission-based approach:
- A process has to get permission before accessing a resource
- Grant permission to only one process at any time

Advantage:
- Doesn't fail because of a lost token

Disadvantage:
- May encounter starvation and deadlock issues

7. **Discuss the tradeoff between large and small consistency units (conit) in keeping data consistent.**

- If the conit is too large, we only update a portion of the content in the conit each time, thus a lot of traffic overhead is introduced.
- If the conit is too small, a lot of updates need to be sent out, which will be carried out in separate communications.

8. **Discuss the differences of sequential, causal consistency and their relation to the eventual consistency.**

The causal consistency is a relaxed model of sequential consistency: sequential consistency requires all writes, either causal or not, to be observed in the same order at each process, while causal

consistency only requires that Writes that are potentially causally related are observed in such order but has no restrictions on concurrent ones.

An eventual consistency is a weak consistency model, compared to sequential and causal consistency models, in the system with the lack of simultaneous updates. It defines that if no update takes a very long time, all replicas eventually become consistent. It is much simpler to implement and could be very performant.

9. Compared the advantages and disadvantages of busy-waiting, blocking synchronizations and monitor-based synchronization.

For busy-waiting synchronization:

Pro: whoever acquires a lock can immediately go to the critical region, so the latency to acquire lock is minimal.

Con: the thread will continuously loop until the lock becomes available — waste a lot of CPU cycles.

For blocking synchronization:

Pro: Waiting processes/threads doesn't consume resources.

Con: OS involvement introduces much overhead.

For monitor-based synchronization:

Pro: Monitors make parallel programming easier and less error prone.

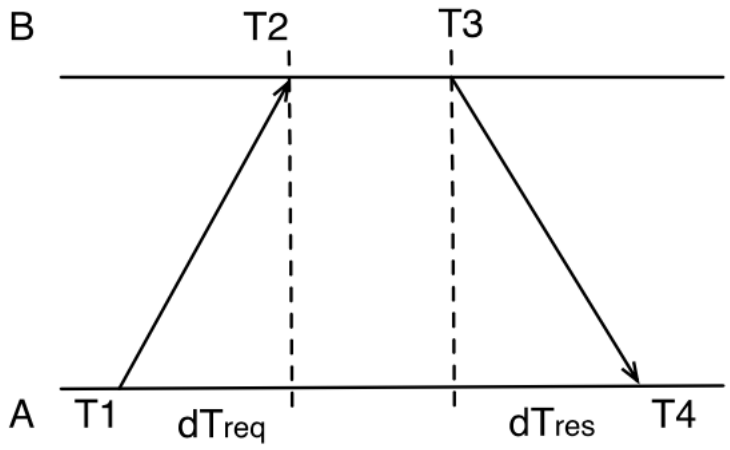
Con: Not all programming languages support monitors. With monitors, although programming becomes much easier, the user has no control over how programming is done — the performance of using monitor is not as good as using basic sync primitives.

10. What are the requirements for building a dependable system?

Dependability implies the following:

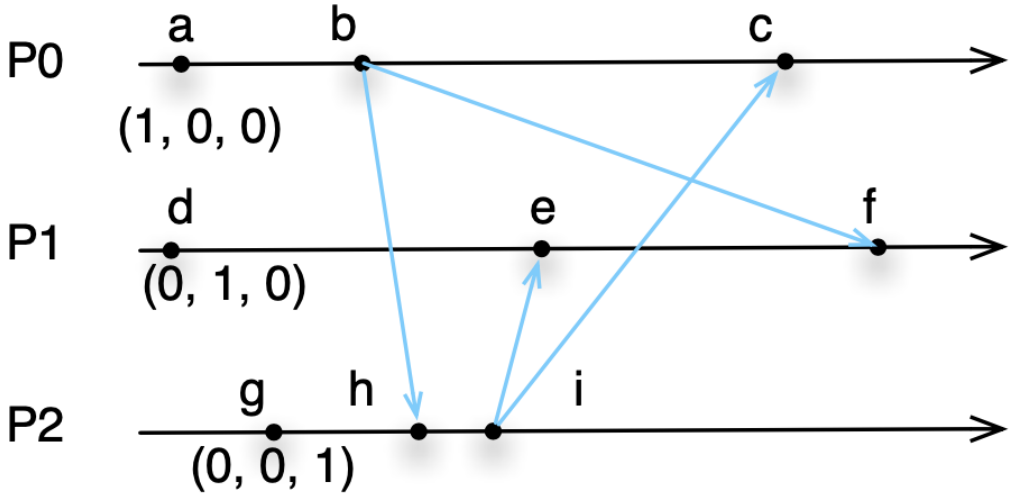
- Availability: A system is ready to be used immediately
- Reliability: A system can run continuously without failure
- Safety: When a system temporarily fails, nothing catastrophic happens
- Maintainability: A failed system can be easily repaired

11. Consider the following situation, in which machine A sends a clock synchronization request to time server B. The request has a timestamp of T1 = 1:52:19.200 (reads as one o'clock, fifty- two minutes, nineteen seconds and 200 milliseconds). Time server B receives the request at T2 and sends a response with its local timestamp at T3 = 1:52:19.500. The response is received at A at time T4 = 1:52:19.600. The network time protocol (NTP) is used for synchronization. Assume that the latency of the time server to respond to a clock synchronization request is 100 milliseconds, what is the offset of A's clock relative to that of the time server and what timestamp should A set its clock to? Show your work and explain the practical meaning of the formula you use.



T3 – T2 = 0.1
T2 = 0.5 – 0.1 = 0.4
dTreq = T2 – T1 = 0.2
dTres = T4 – T3 = 0.6 – 0.5 = 0.1
Θ = T3 + (dTreq + dTres)/2 - T4 = 0.5 + (0.2+0.1)/2 – 0.6 = 0.05
A is lagged by 0.05 s.
A should be preponed by 0.05 s, so T4 should become 1:52:19.550

12. Consider the following communications between processes P0, P1, P2. Answer the questions below. What are the vector timestamps of the six remaining events?



b	c	e	f	h	i
(2, 0, 0)	(2, 0, 2)	(0, 1, 0)	(2, 1, 0)	(2, 0, 1)	(2, 0, 2)

13. Give examples (using diagrams) of the following client-centric consistency models: monotonic reads, monotonic writes, read-your-writes, and writes-follow-reads.

Refer to the slides for examples.