What has to be done:

A **group report** that reasonably collects

(a) what properties of those defined for a distributed file system meets the service of the SAR/SZA subject,

(b) what properties it lacks in relation to a cloud system as defined in Activity 1.2.

Properties of distributed systems

We will first explain the basics of each property, so later on, we can discuss if the SAR app has certain properties or not.

TRANSPARENCY

The main objective of a distributed system (from now on DS) is to provide the user and the apps with a view of the system resources as if it was managed by a single virtual machine.
They are different aspects of transparency, such as:

- Identification: The namespaces of the resources are independent of the distribution of the resources and the topology of the network.
- Location of the resources: Neither the users nor the apps know in which node the accessed resource is. This implies that resources can migrate between nodes without the apps being affected.
- Replication: As in the location of the resources, neither the users nor the apps know how many copies there are of each resource.
- Parallelism: An app can be executed in parallel, without the app having to specify it. This affects the performance, and can cause performance issues.
- Sharing: The fact that a resource tries to be accessed at the same time from various apps has no effect on the execution of the app.

SCALABILITY

Modularity, having dense connections between the nodes inside the modules but sparse connections between nodes in different modules, is one of the characteristics of DS. This allows flexibility and enables scalability, what means that the system can grow in an efficient way and without increasing its complexity.
Scalability presents two aspects: Namespaces and Complexity.

- Namespaces: Namespaces, can identify objects of different nature, such as files. In the case of linear spaces, such as memory, there is an inherent **limitation** associated with the **size** of the name, so that today it is reasonable to raise the insufficiency of 32-bit address spaces.
- Complexity/Performance: The growth of a DS can introduce **bottlenecks** and **latencies** that degrade its performance. Also, adding new systems generates more complexity, making the communication between systems more complex, and therefore, **affecting** the overall **performance**.

RELIABILITY AND FAULT TOLERANCE

Reliability of a system is defined as the ability to perform its tasks and functions correctly. Reliability is specified in two aspects:

- Availability: We call availability to the fraction of time the system is operative. The main parameter to measure the availability is the mean time between failures (from now on MTBF), baut we must also consider the repair time. Availability can be increased in two ways: (***Note:*** *both alternatives increase the cost of the system*)
    - Using components of **higher quality**.
    - With a **design** which **replicates** the **components** that allows the system to continue operating, even when one of them fails.
- Fault tolerance: Even with high availability, a failure at a certain time can have disastrous consequences. Fault tolerance expresses the ability of the system to continue operating correctly when any of its components fail. Thus, fault tolerance involves:
    - **Detecting** the failure.
    - **Continuing** the service in a transparent manner.

CONSISTENCY

The distribution of resources introduces importante benefits, but also problems. About the **benefits**, the distribution increases the **performance** through **parallelism** and promotes access to local copies of the resources, which **decreases communication costs**. Also, replication increases availability which provides fault tolerance.
However, we also find problems when distributing resources. First, the interconnection **network** is a new source of **failures** and the security of the system is more **vulnerable to unauthorized access**. But the problem of greater complexity is the management of the global state to avoid situations of **inconsistency** between the components of the system.

The main problem lies in the need of **maintaining a global state** of a system with several components, where each one has its one local state. The nodes of the system are physically distributed, so that the management of the global state depends strongly on the communication mechanisms, in turn supported by a network subject to failures.

In summary, the maintenance of a strict consistency requires a strong support, which implies a great load of additional communication between nodes of the system. To avoid this, is preferable to relax the consistency level to maintain an acceptable performance level according to the needs of the application.

OTHER PROPERTIES

- **DISTRIBUTED FILE SYSTEM**: In a distributed file system, one or more central servers store files that can be accessed, with proper authorization rights, by any number of remote clients in the network.

Exercises:

A) What properties of those defined for a distributed file system meets the service of the SAR/SZA subject?

1. Consistency: The system doesn't have replication among its functions, as a result it will be completely consistent between accesses.
2. Transparency: Even if you need the port number and the address of the server, once you are finally connected, you don't need any kind of external help to upload or download your files.

B) what properties it lacks in relation to a cloud system as defined in Activity 1.2.?

1. Transparency: our system requires technical knowledge of the addresses of the servers and the ports which leads to a cumbersome system where there is no seamlessness in the access to resources.
2. Scalability: Our system is a simple design of a cloud system, and it's not prepared for advanced use with or an elevated user number. Adding new systems generates more complexity, making the communication between systems more complex, and therefore, affecting the overall performance of the system.
3. Availability: Our system is designed to connect to a unique server. In case the server we are willing to connect is not available, we are not going to be able to use the service correctly. To avoid this problem, we should deploy our system on multiple machines.
4. Confidentiality | Data Security: A cloud system should be able to guarantee that only the one that uploads the files to the server has access to it, it's called confidentiality. For it, two aspects should be developed: first, encryption of communications. Anyone that intercepts the communications between the user and the server should not be able to access the information. Second, storaged files encryption, only the owner of those files should have access to the information stored on his account.
5. Data integrity: Our system does not have a backup place to make a copy of the files that are uploaded. This means that in case that our system suffers any kind of problem, we may lose the information. To avoid this, a mirror system should be deployed.

Sources:

- https://www.webopedia.com/TERM/D/distributed_file_system.html#:~:text=Distributed%20file%20system%20(DFS)%20is,remote%20clients%20in%20the%20network.
- Topic 1 notes.(UPV/EHU University)