Assignment 1

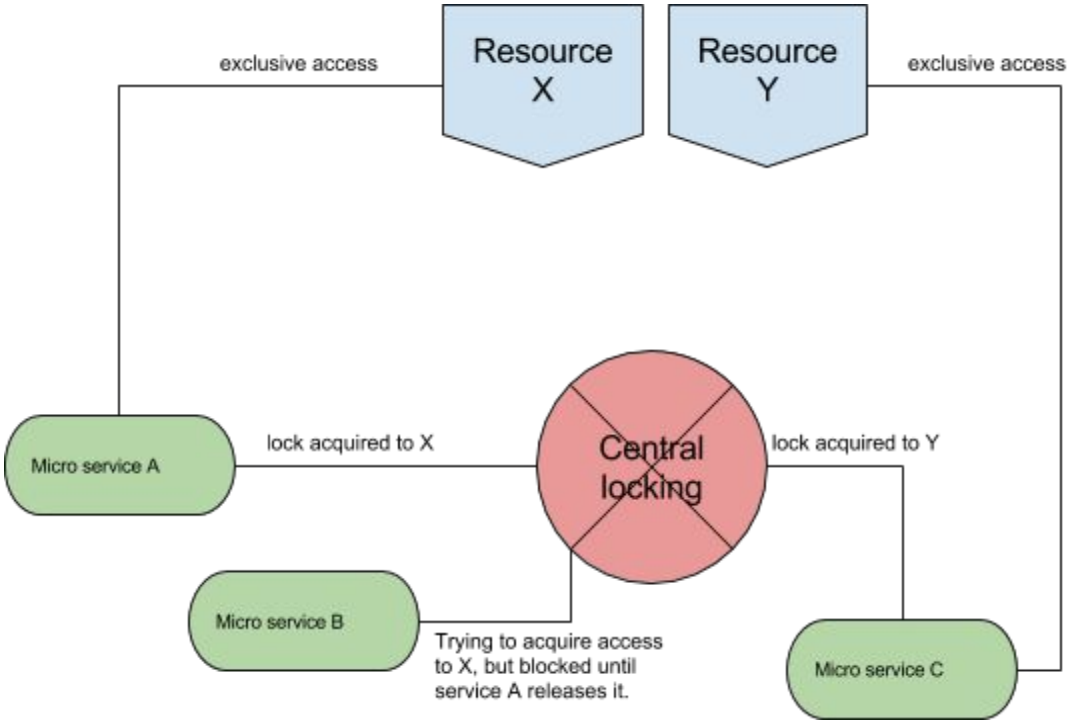
Centralized locking system

Description

Usually, during building a distributed, highly available system designers comes across the problem of shared resources. Local shared resources (like files, disks and attached hardware) is easy to handle with local locking mechanisms (like file locks and others). But in a more distributed system, acquiring an exclusive access to a remote shared resource, comes with lots of pitfalls and challenges. For example, a centralized locking system must provide the following requirements

Requirements

1. 100% guarantee of exclusive access. If 2 separate system managed to get access to the same resource, an unrecoverable corruption might occur.
2. If an exclusive access is granted to one of the services, and the service crashed or died without releasing the locks, the locking system must free the lock on that resource (eventually), so other services doesn't get blocked forever.
3. Support timeout, if a service can't get an exclusive lock to a resource after the given timeout.
4. Detect deadlocks. [Optional]
5. resource is defined by it's name. This name is used as a key to acquire the locking.



Implementation details

1. Central locking system must be accessible remotely (you choose the interface with the outer world [REST, Sockets, etc...]
2. Locking clients that can be used from any service to acquire the lock by the name.
3. If timedout, or another locking error occurred (deadlock detected for example) the error must be reported to the caller.
4. Well defined interface, so different clients in different languages can be implemented.
5. Test coverage