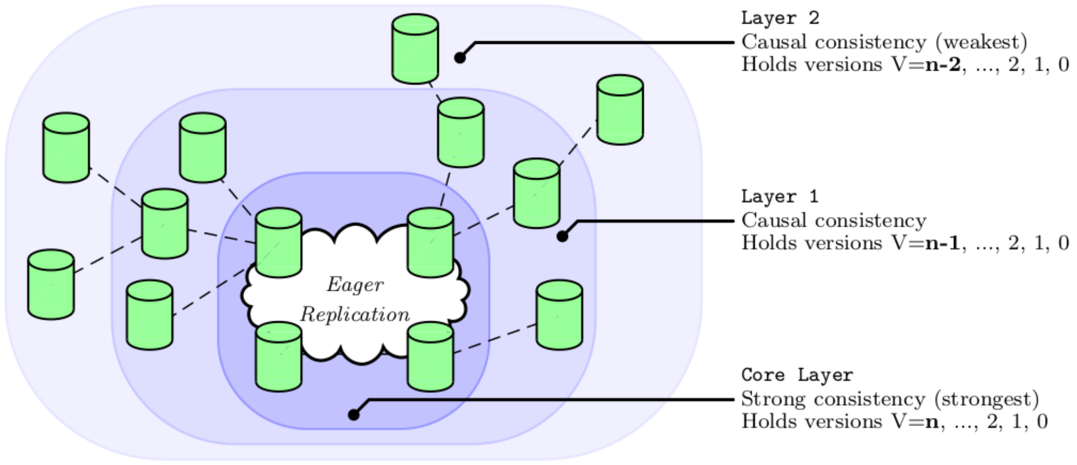
EXERCISE 4: EPIDEMIC REPLICATION

Learning Objectives
By the end of this exercise, you should be able to:

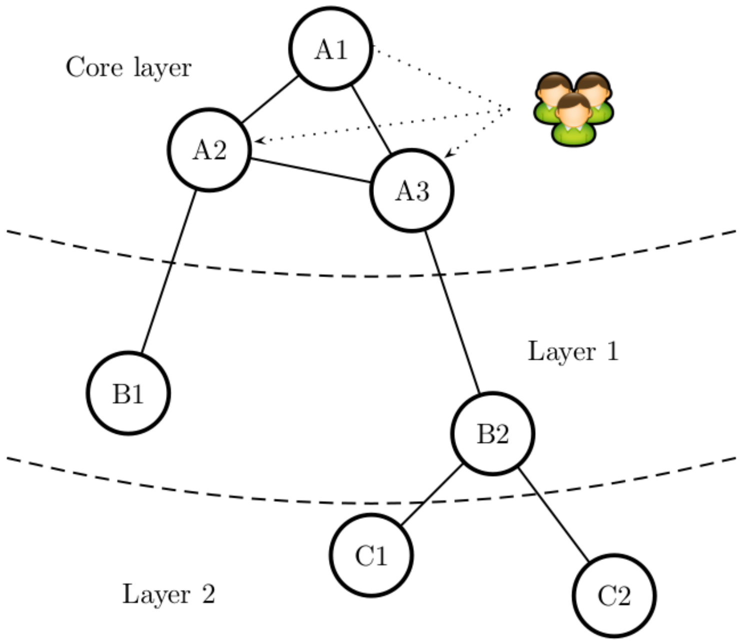
- Understand data replication strategies.
- Implement a multi-layered distributed system.

Exercise statement
Design and implement a distributed application in charge of replicating the data in an epidemic manner to achieve a multi-versioned data architecture similar to the following:


Layer 2
Causal consistency (weakest)
Holds versions V=**n-2**, ..., 2, 1, 0
Layer 1
Causal consistency
Holds versions V=**n-1**, ..., 2, 1, 0
Core Layer
Strong consistency (strongest)
Holds versions V=**n**, ..., 2, 1, 0
Eager Replication

In this system, the nodes that belong to the *core layer* have the most modern versions of the object. However, nodes belonging to the second layer have the oldest versions. Note that (1) the different nodes of a same layer will always have the same version (maybe old, but the same!), (2) by time = ∞ all nodes will converge towards the same version if the *core layer* does not propagate more updates (aka *eventual consistency*), (3) the clients issue update transactions to the *core layer*, and (4) read-only transactions can performed on any layer.
Specifically, we want to build a system like that of the figure *below* in which there are 7 nodes and 3 layers. the characteristics of the system are the following:

1. Communication between nodes must be via sockets.
2. The *core layer* uses *update everywhere*, *active*, and *eager replication*.
3. The nodes B1 and B2 from the second layer, receive the data every 10 updates (*lazy*) through passive replication, *primary backup*.
4. Nodes C1 and C2 receive the data every 10 seconds (lazy) through passive replication, *primary backup*.
5. Each node must maintain a local text file the log of all the versions it keeps.
6. Clients launch transactions specified in a local file. Transactions can be of two types:
   - ***Read-only***: `b<f>, r(30), r(49), r(69), c`. Where <f> is an integer that is 0, 1 or 2 and indicates the layer on which the transaction must be executed.
   - ***Not read-only***: `b, r(12), w(49,53), r(69), c`. These transactions will always be executed in one of the nodes of the *core layer*.
7. The status of each node must be monitored in real time through *web sockets* Create a web application for this purpose.


Core layer
A1
A2
A3
Layer 1
B1
B2
Layer 2
C1
C2