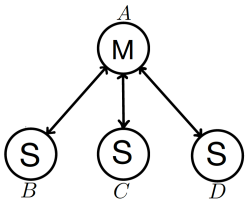
Group ID: 57
Timeslot Availability: Wednesday 15:15 - 19:00
Group Members: Adrian Niazi, Jonas I. Selfors, Henrik Elde
Emails: adriania@stud.ntnu.no, jonasise@stud.ntnu.no, henrikae@stud.ntnu.no

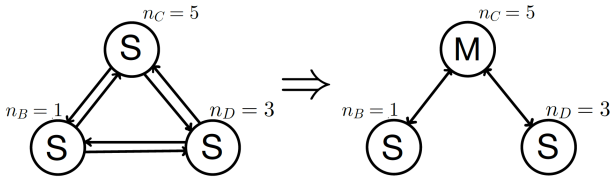
Network Topology

The system mainly consists of two modes of operations, in which the first one uses a master-slave topology and the second peer-to-peer topology. The behaviour of the system is as follows:

1. In the first mode of operation the responsibility of the master is to distribute all the incoming cab and hall calls in an efficient manner. In this mode of operation all the slaves accept the responsibility and commands given by the master, and it follows a hierarchical structure illustrated in figure 1a.
2. When the current master goes offline, the network enters the second mode of operation. This changes the structure from master-slave to peer-to-peer. The sole purpose of this mode of operation is to identify a new master. The tentative idea is to use the IP-number, i.e. the last digit of the IP-address, of the nodes as an identifier. The nodes will broadcast a message to the other nodes containing the suggested new master identified by the IP-number. The nodes will accept a new suggested master only if the IP-number is bigger than what the node currently holds, otherwise it will disregard it. Along with the IP-address the node contain a list of all the nodes that has broadcasted that message. When all the nodes in the network has accepted the highest IP-number among the nodes, the node who originally had the highest identifier will signal that it has taken over as the new master. The algorithm for choosing a new master follows the same principle as a non-monotonic cyclic counter. This avoids the possible traps that self-reference and negation poses in a distributed network. When a new master has been chosen, the mode of operation returns back to 1.



(a) Mode of operation 1. The two-way arrows signifies that the master both receive information from the slaves and transmits information back to them acting as a single source of truth.



(b) Showing the transfer from mode of operation 2 back to 1 through peer-to-peer communication. Notice the node with the biggest IP-address $n_c = 5$ becomes the new master, where n_x represents the last digit of the local IP-address.

Figure 1: Illustration of mode of operation 1 & 2 respectively.

Strategy for fault tolerance

Fault tolerance achieved through choice of topology as described in the previous section, in combination with a "distributed ledger of orders" managed by the master node. The slave-master topology avoids inconsistency issues by acting as a single source of truth. The ledger contains all current orders and the corresponding node responsible for servicing each order. This guarantees that:

1. As long as the system is in mode of operation 1, the master acts as a single source of truth and most of the problems associated with faults can be fixed by the master redistributing tasks that a slave node is unable to do (due to e.g. disconnect, motor stop, software crash etc.).
2. Data-loss prevention in the event of a master disconnecting (mode of operation 2) is avoided by preventing the slave nodes to signify a service guarantee light before the master has written it to the ledger and redistributed it back to the slave nodes. Thus, in the case of a master disconnecting, the latest version of the ledger as stored by the former slave-made-master becomes the starting point for the new master. Since the ledger is distributed to all nodes, and the slave nodes cannot guarantee service before it is written to the ledger, the order-data is always preserved. In addition, no new orders will be accepted while in mode of operation 2, which prompts the user to press the button until the system is back in mode of operation 1.

Another way this design avoids the problem of inconsistencies is to extract information from the lack thereof; when the current master disconnects, the slaves will use the lack of messages received from the master as information of its disappearance, which prompts mode of operation 2. In order to avoid the problem of trying to merge two masters, every node that reconnects to the network will be set as a slave with a value of "unknown" by default, regardless of whether the node was a master or a slave prior to its disappearance.

Choice of programming language & division of modules

The Go programming language was chosen mainly for its built in go-routines, which makes it simple to create concurrent processes, and the corresponding channels that makes synchronization between processes seamless.

The system takes advantages of a shell and core structure, where the core exclusively consists of pure functions. All communication with "the outside world" is handled by the shell, and the intermediate actions are handled by the core. Furthermore, the system is divided into modules and structured in a hierarchical fashion. There's no cross-referencing between modules, and the different modules has a single defined purpose.