Beastly Heis v1.5

Kolbjørn Austreng, Andreas Våge

January 29, 2017

Diagrams

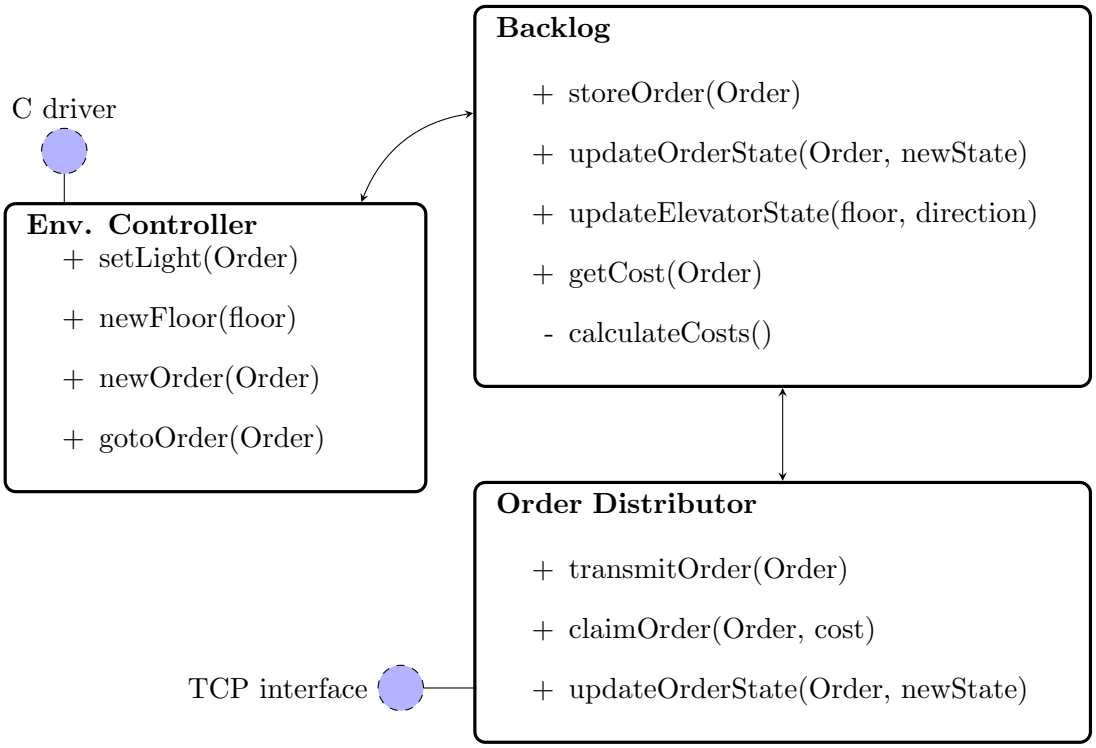

Figure 1: System module diagram.

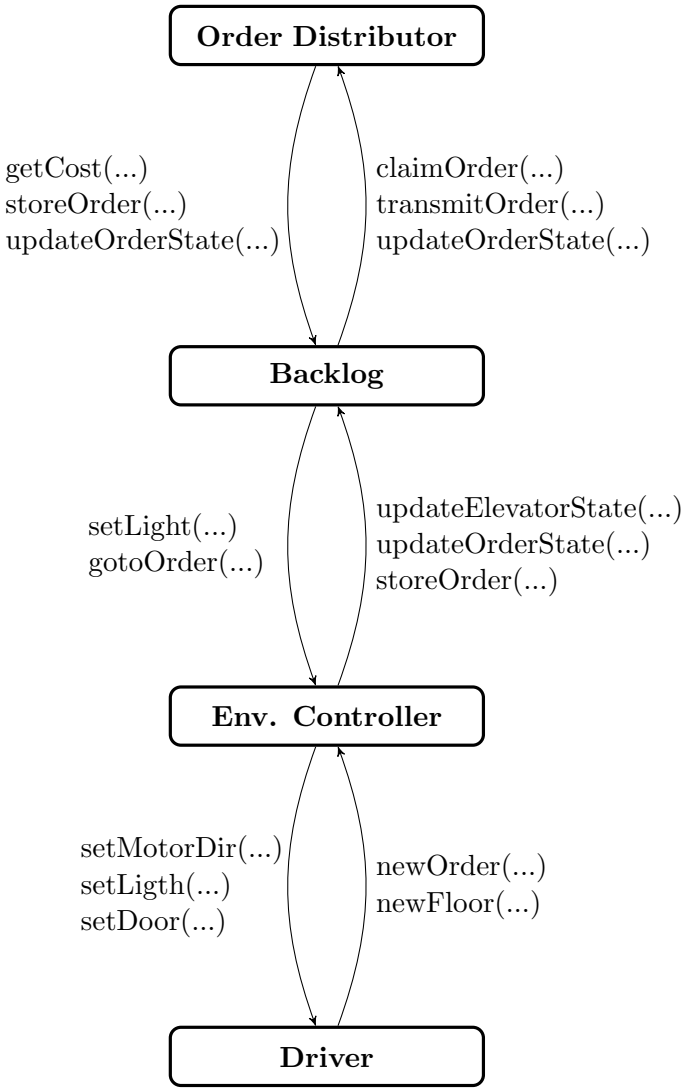

Figure 2: Function call between modules

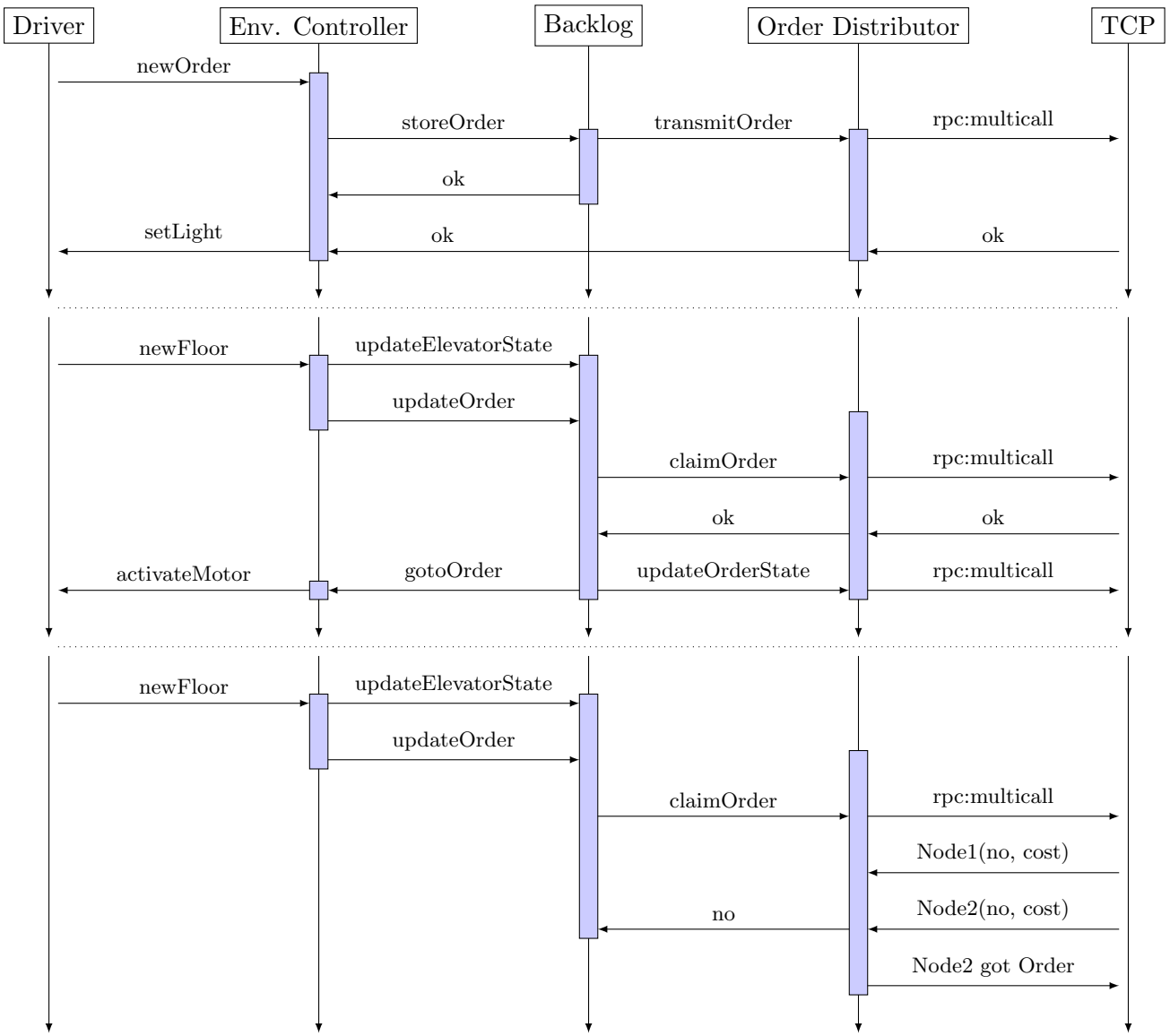

Figure 3: Sequence diagram between modules

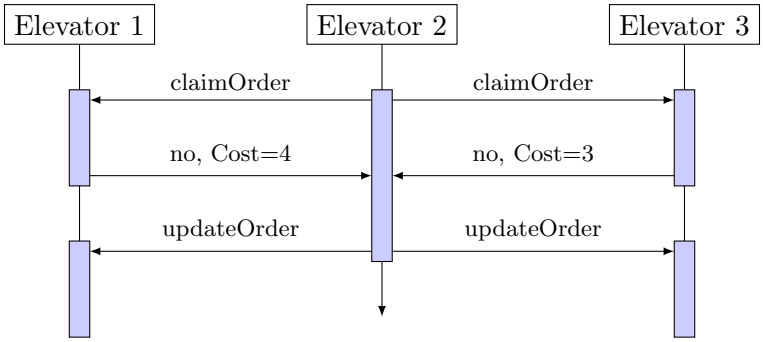

Figure 4: Sequence diagram between three elevators

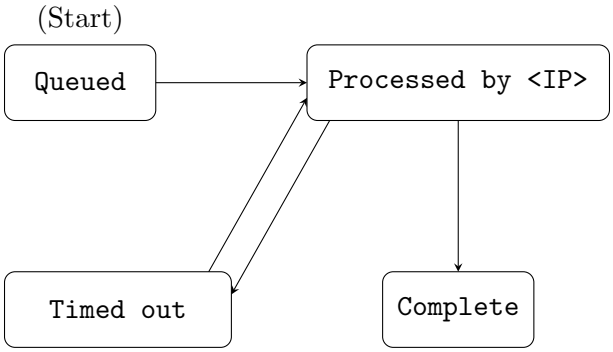

Figure 5: Order life stages.

Order object

Order	Comment
+ type	Internal / External
+ floor	Destination floor
+ timestamp	Set by computer that first received order
+ origin IP	Set by computer that first received order
+ state	Queued, In progress, Timed out, Complete

Environment Controller

+ setLight(Order)

Sets the light corresponding to the floor of an Order object.

+ newFloor(int floor)

Call from C driver to communicate to *Environment Controller* that a new floor has been reached.

+ newOrder(Order)

Call from C driver to communicate to *Environment Controller* that a new order has been created.

+ gotoOrder(Order)

Sends the elevator to the floor of a specific order.

Backlog

+ storeOrder(Order) ok

Saves an order from either *Environment Controller* or *Order Distributor* to the backlog. Returns acknowledgement.

+ updateOrderState(Order, newState) ok

Changes the state of a specific order. Returns acknowledgement.

+ updateElevatorState(int floor, enum direction)

Communicates the position and direction of the elevator to the *Backlog*.

+ getCost(Order) cost

Returns the cost of taking a specific order for this elevator.

- calculateCosts()

Calculates the costs of all the orders in the backlog for this elevator.

Order Distributor

+ transmitOrder(Order) ok

Transmits an Order object to all the other nodes in the network. Acknowledges if at least one other elevator received the order transmit, or there are no other elevators in the network.

+ claimOrder(Order, cost) ok

Attempts to claim an order in the *Backlog*. Transmits own cost of taking on this order. Acknowledges if no other elevators have a lower cost on the specified order.

+ updateOrderState(Order, newState) ok

Broadcasts an order state update to ensure that the backlogs are identical.

Fault handling

Software crash

- Backlog is saved to a local file.
- All orders are distributed to all other nodes.
- Upon restart, open saved backlog, merge with network backlog.
- All claimed orders will time out; be claimed by other nodes.

Software hangs

- Time out on claimed orders.
- Supervisor restarts process.

Network cable disconnect, then reconnect

- Claimed orders time out when TCP marks it as dead.
- Syncs backlog from network on reconnect.

Troll user

- Cost function prioritizes continuing in current direction, and older orders. An elevator will never not serve an order, given enough time.

Elevator never arrives

- Order times out, different elevator takes over.

Cosmic rays

- TCP for detecting junked up messages.
- New "phantom" orders may be created, no big deal.
- No orders can be wiped from the system without being explicitly marked as complete.
- Supervisor restarts uncooperative processes.

Power out on all elevators

- Backlog saved to local file, will carry on where they left of on power up.

Heat Death of the Universe

- This is acceptable.