David Pick

Eric Stokes

Pete Brousalis

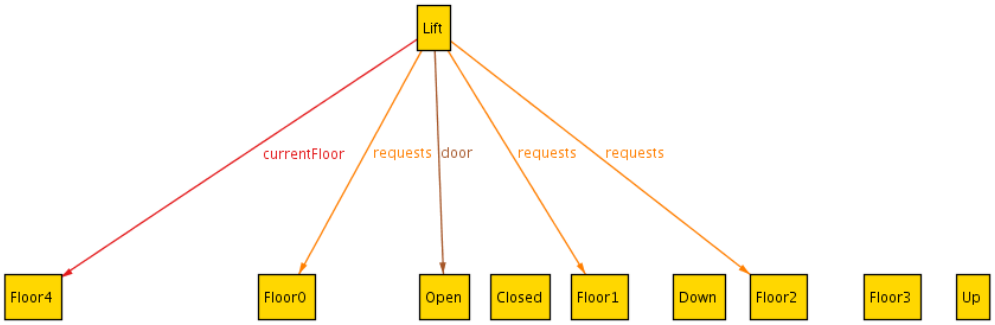
Elevator Project

Introduction

For this project, we were required to specify a control system for an installation of elevators including 3 lifts and 5 floors. In order to start modeling our system, we created some very basic signatures. We first modeled the physical aspects of the lift by creating the Floor, Time, and Lift signatures. In order to ensure the order of the floor signatures, we used the package ordering. By using the package, we made sure that Floor0 pointed to Floor1 and Floor1 pointed to Floor2 and etc... This also allowed us to use the next relation when trying to determine which floor would come after a given one. We added a DoorState signature extended by Open and Close for keeping track of the state of the lift's doors. We also added a Direction signature extended by Up and down to keep track of the intended direction of each lift. We did this so a lift can always be defined as open or closed.

In our model, instead of having buttons and tracking pressed and unpressed states, we have a set of requests. We created a requests field in the Lift signature, of type Time -> Floor. By doing this, we know that if a floor is in the set of requests, it must be pressed. A depiction of the set of requests on a current floor that is not in the set is located below in Figure 1.

Figure 1.



Facts

The next part of our model deals with specific facts that constrain our system. The facts are outlined below:

- The first fact we chose to create was doorAlwaysOpen. This makes sure that since there is no time when an elevator is in between floors, the door is always open.

- We added a floorDoesNotChangeRequestsDoNotChange fact which ensures that if the floor did not change between Time1 and Time2, then the requests floors did not change between Time1 and Time2.

- We then added a requestCannotChange fact. This makes sure that if the floor changes from t1 to t2 and neither floor is in the requests set, then the requests set does not change from t1 to t2.

- We added a mustBeAValidTransition fact, which checks to make sure that the floor didn't "teleport", meaning it did not go up or down by more than 1 floor. This also calls the validTransition predicate, which ensures that the new floor was removed from the set of requests, if it was in the set.

- And finally, we added a cannotBeOnRequestedFloor fact which ensures that the current floor is not in the set of requested floors.

Predicates

Our model contains several predicates in charge of the movement of the lift. The moveUp predicate moves the lift up one floor by setting currentFloor to the next currentFloor. The moveDown predicate moves the lift down one floor and works similarly to the moveUp predicate. We also a serviceFloor predicate, this predicate checks to see if the current floor is in the set of requested floors, and if so removes it from the set.

Assertions

We wrote an assertion, requestsServiced, to check if the current floor was in the set of requested floors, it was then removed.

Visual Representations

We used the Alloy to preview our model and export the images for analysis. Each predicate depicted below is projected over the Time signature.

moveUp

These images demonstrate the moveUp predicate. They show a model at Time t and t'. At Time t, the currentFloor is Floor0, then at Time t' the currentFloor is Floor1. These models are depicted below in Figure 2 and Figure 3.

Figure 2.

This diagram shows the Time t model for the moveUp predicate.

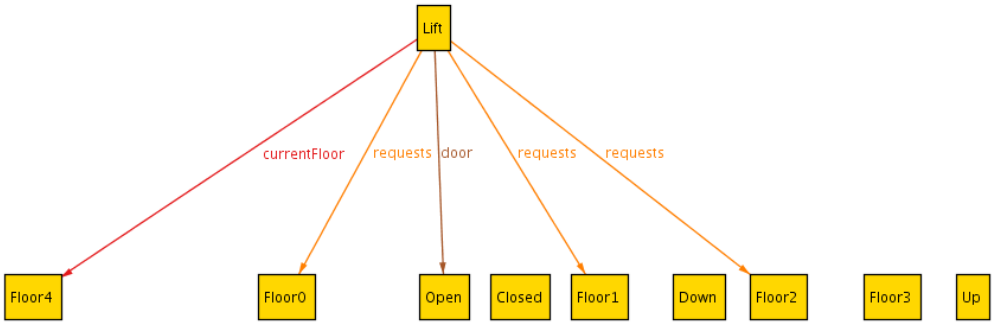
Figure 3.

This diagram shows the Time t' model for the moveUp predicate.

modeDown

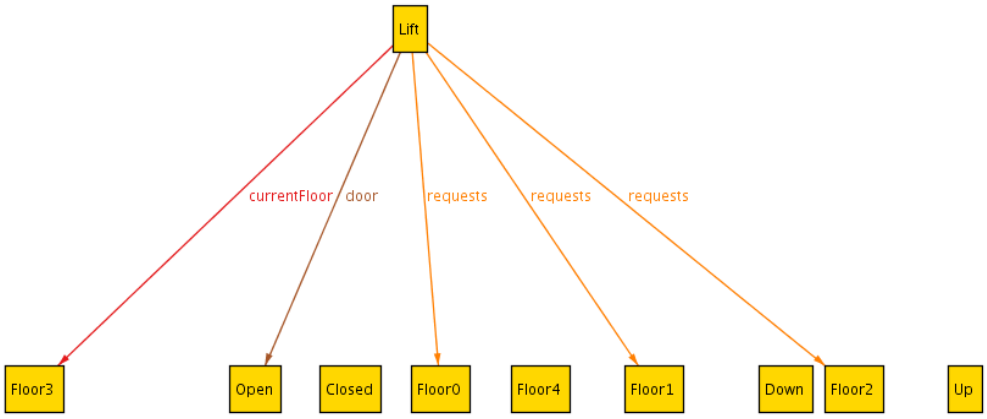
These images demonstrate the moveDown predicate. They show a model at Time t and t'. At Time t, the currentFloor is Floor4, then at time t' the currentFloor is Floor3. These models are depicted below in Figure 4 and Figure 5.

Figure 4.



This diagram shows the Time t model for the moveDown predicate.

Figure 5.



This diagram shows the Time t' model for the moveDown predicate.

Time Spent

David Pick	**6 hours**
Pete Brousalis	**5 hours**
Eric Stokes	**5 hours**