ECE 18-649
Final Project Report

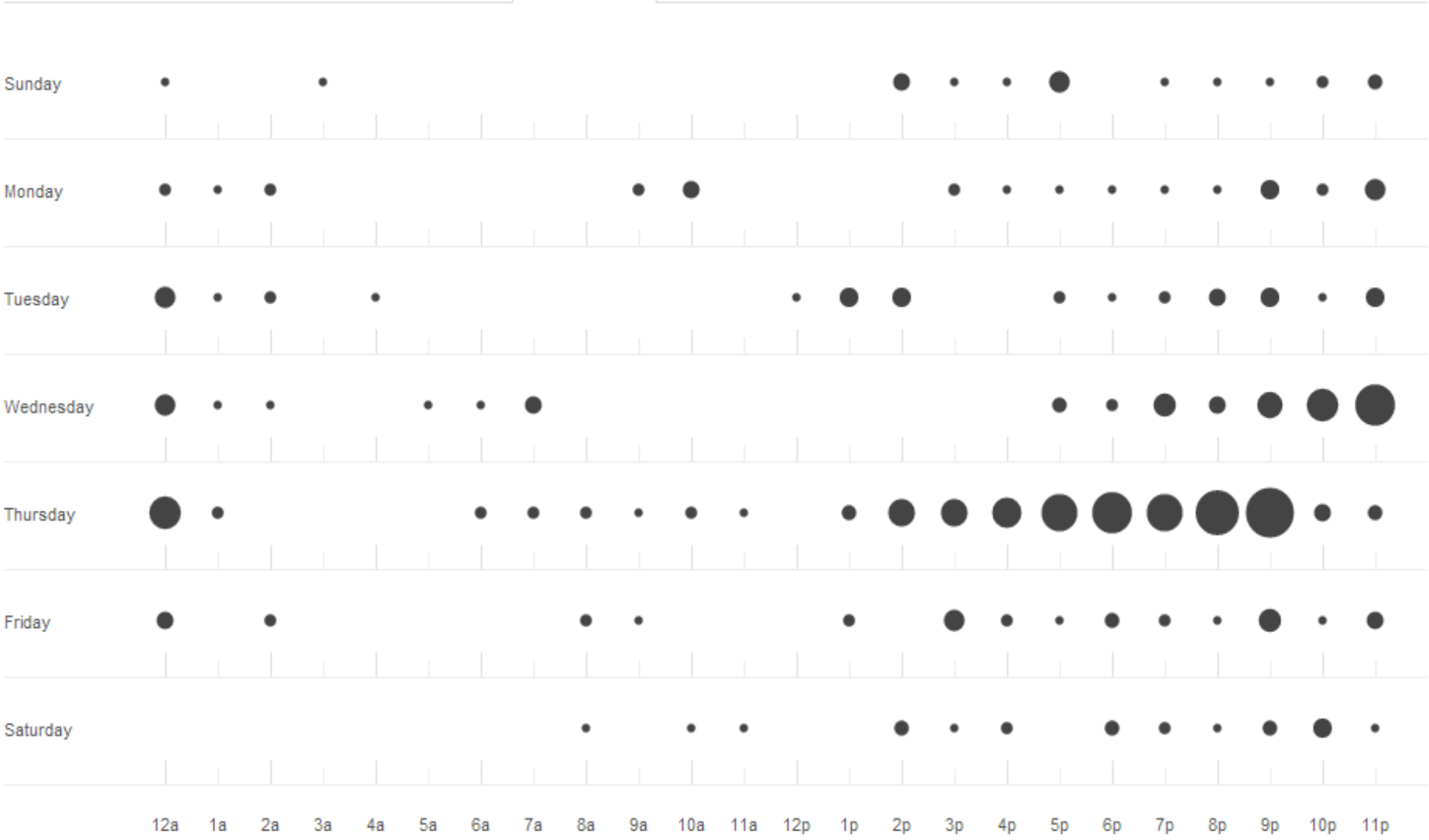
December 02, 2013

Group # 18
Sairam Krishnan (sbkrishn)
Yi Huo (yhuo)
Mustafa Bilgen (mbilgen)
Abhijit Lele (alele)

Overview

- Project statistics
  - GitHub Commit Trends
  - Now vs. Midsemester
- Door Control
  - Design
  - Implementation
- Lessons learned
- Open issues
- Suggestions

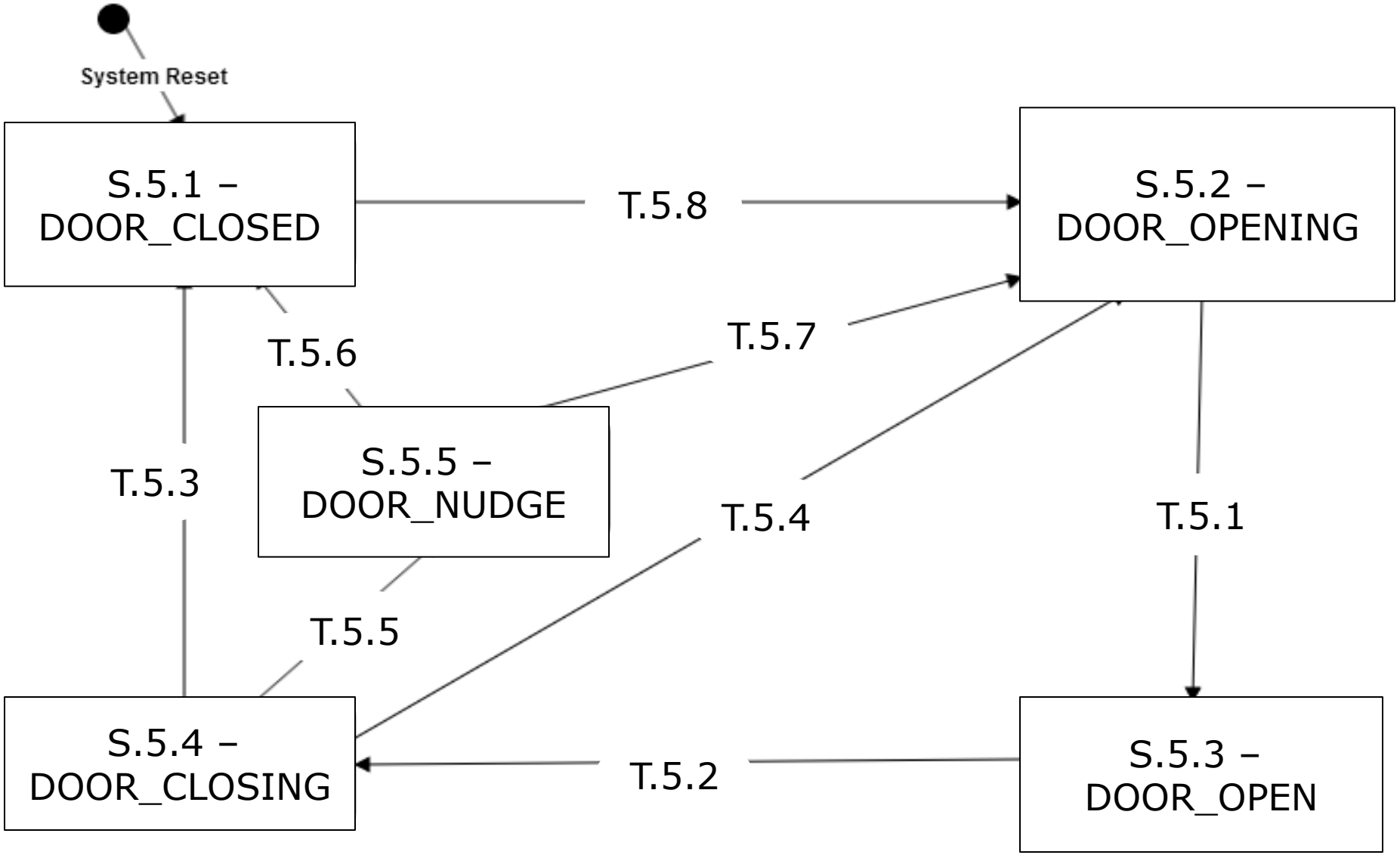
Project Statistics



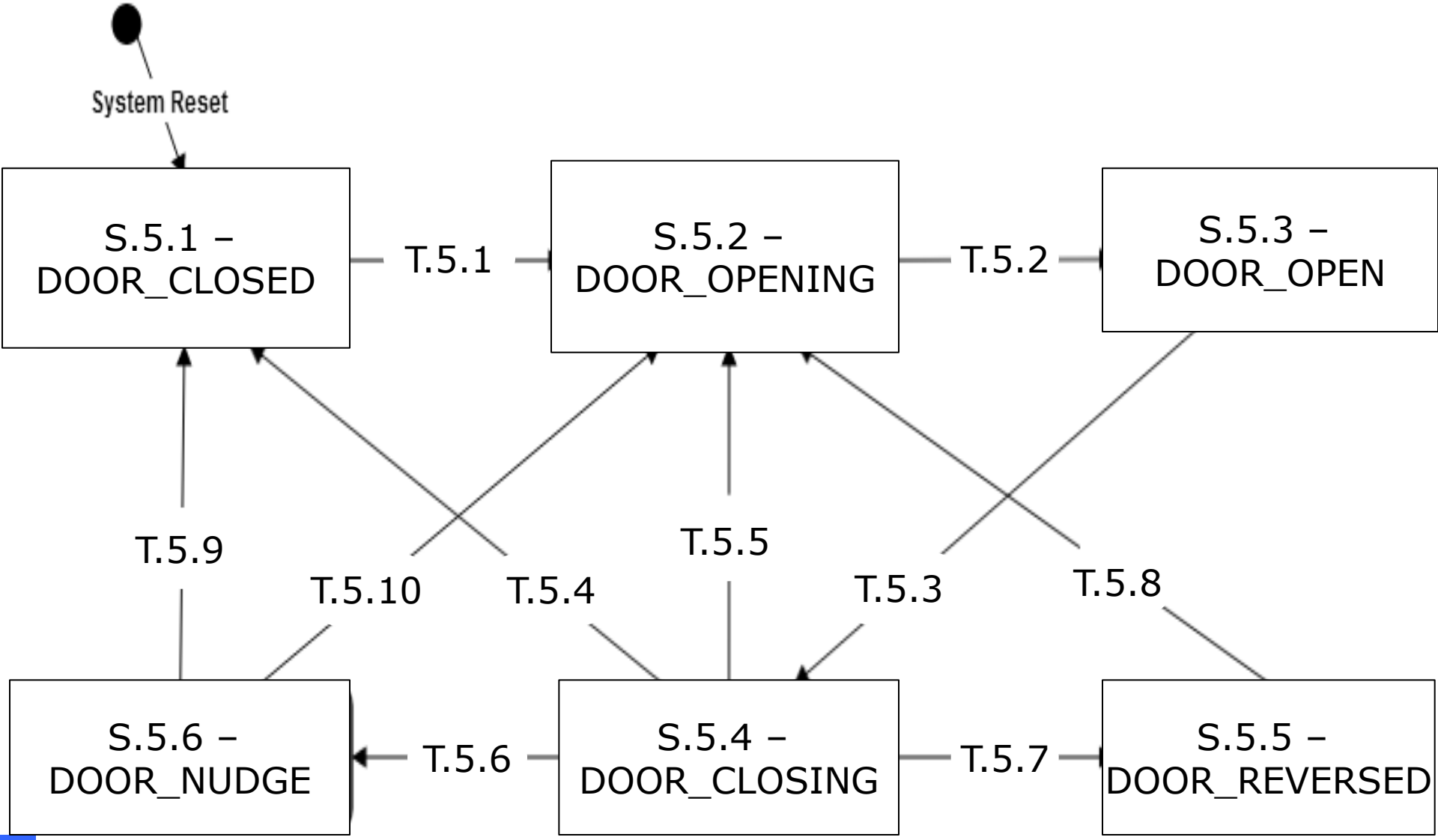
Project Statistics

	Midsemester	Now
(# of sequence diagrams; # of arcs)	(18; 146)	(19; 163)
# of lines of requirements	58	75
(# of state charts; # of states; # of arcs)	(7; 31; 24)	(7; 48; 30)
# of lines of non-comment code	1302	1798
# of test files written (incl. ".cf" files and the custom acceptance test for P8)	54	60
# of peer reviews	52	81
total # of defects found & fixed via peer reviews	40	64
total # of defects found & fixed via test & other	12	20
# of revisions	20	48
# of GitHub commits	308	453

Design of Door Control - Old



Design of Door Control – New



Design of Door Control - New

State	Description	Actions
S.5.1	DOOR_CLOSED	DoorMotor[b,r] = Stop, CountDown = 0, ReversedOnce = False
S.5.2	DOOR_OPENING	DoorMotor[b,r] = Open, Dwell = mDesiredDwell[b], CountDown = Dwell
S.5.3	DOOR_OPEN	DoorMotor[b,r] = Stop, Decrement CountDown
S.5.4	DOOR_CLOSING	DoorMotor[b,r] = Close
S.5.5	DOOR_REVERSED	ReversedOnce = True
S.5.6	DOOR_NUDGE	DoorMotor[b,r] = Nudge

Design of Door Control

Transition #	Guard Condition
T.5.1	mAtFloor[mDesiredFloor.f, b] == True && (mDesiredFloor.b == b || mDesiredFloor.b == BOTH) && (mDriveSpeed(0,d) == True || mDriveSpeed(s, Stop) == True)
T.5.2	mDoorOpened[b,r] == True
T.5.3	CountDown <= 0 && mCarWeight(x) <= MaxCarCapacity
T.5.4	mDoorClosed[b,r] == True && mCarWeight(x) <= MaxCarCapacity
T.5.5	mCarWeight(x) > MaxCarCapacity
T.5.6	mDoorReversal[b,*] == True && mCarWeight(x) <= MaxCarCapacity && ReversedOnce == True && mDoorClosed[b,r] == False
T.5.7	mDoorReversal[b,*] == True && mCarWeight(x) <= MaxCarCapacity && ReversedOnce == False && mDoorClosed[b,r] == False
T.5.8	None
T.5.9	mDoorClosed[b,r] == True && mCarWeight(x) <= MaxCarCapacity
T.5.10	mDoorOpened[b,r] == False && mCarWeight(x) > MaxCarCapacity

Door Control Implementation

```
switch (state) {
   //#state '5.1'
   case DOOR_CLOSED:
      //#state '5.1' actions
      localDoorMotor.set(DoorCommand.STOP);
      countDown = SimTime.ZERO;
      reversedOnce = false;
      int desiredFloor = mDesiredFloor.getFloor();
      if (desiredFloor == 0) { break; }

      //#transition 'T.5.1'
      Hallway desiredHallway = mDesiredFloor.getHallway();
      boolean atFloor = mAtFloor.isAtFloor(desiredFloor, hallway) &&
               (desiredHallway == hallway || desiredHallway == Hallway.BOTH);

       if (atFloor && (mDriveSpeed.getSpeed() == 0.0 || DriveSpeed.getDirection()
          == Direction.STOP)) {
          newState = State.DOOR_OPENING;
      }
      else {  newState = state;  }
      break;
```

Design of Door Control

```
case DOOR_OPENING:
localDoorMotor.set(DoorCommand.OPEN);
countDown = Dwell;
if (mDoorOpened.getValue() == true) {
        newState = State.DOOR_OPEN;
 }

case DOOR_OPEN:
localDoorMotor.set(DoorCommand.STOP);
countDown = SimTime.subtract(countDown, period);
if (countDown.isLessThanOrEqual(SimTime.ZERO) &&
        mCarWeight.getWeight() <= Elevator.MaxCarCapacity) {
            newState = State.DOOR_CLOSING;
}

case DOOR_CLOSING:
localDoorMotor.set(DoorCommand.CLOSE);
//#transition 'T.5.4'
    if (mDoorClosed.getValue() == true &&
        mCarWeight.getWeight() <= Elevator.MaxCarCapacity) {
            newState = State.DOOR_CLOSED;
 }
```

Design of Door Control

```
//#transition 'T.5.5'
else if (mDoorOpened.getValue() == false &&
    mCarWeight.getWeight() > Elevator.MaxCarCapacity) {
        newState = State.DOOR_OPENING;
}
//#transition 'T.5.6'
else if (mDoorReversal.getEitherReversed() == true &&
    reversedOnce ==true && mCarWeight.getWeight() <=
    Elevator.MaxCarCapacity)
{
        newState = State.DOOR_NUDGE;
}
//#transition 'T.5.7'
else if (mDoorReversal.getEitherReversed() == true &&
    reversedOnce == false && mCarWeight.getWeight() <=
    Elevator.MaxCarCapacity)
{
    newState = State.DOOR_REVERSED;
}
case DOOR_REVERSED:
reversedOnce = true;
```

Design of Door Control

```
case DOOR_NUDGE:
localDoorMotor.set(DoorCommand.NUDGE);
        //#transition 'T.5.9'
if(mDoorClosed.getValue() == true &&
    mCarWeight.getWeight() <= Elevator.MaxCarCapacity) {
        newState = State.DOOR_CLOSED;
}
//#transition 'T.5.10'
else if(mDoorOpened.getValue() == false &&
    mCarWeight.getWeight() > Elevator.MaxCarCapacity) {
        newState = State.DOOR_OPENING;
}
```

Unit Testing – Door Control

- Unit Test Files – 3
- Total Assertions passed – 244
- Failing Assertions – 0

Lessons Learned

- Version control is your friend
- Consistency facilitates communication
  - Use a single naming/style convention
  - Use same software for state charts and sequence diagrams
- Most of your time should be spent on design rather than on implementation and testing

Lessons Learned

- Rerun all tests after changing code

- What worked well
  - Peer reviews
    - Helped catch almost all the major bugs, made sure we had a solid design
  - State charts
    - Great guide to implementation
  - Scenarios and Use Cases
    - Helped us visualize different elevator behaviors

Lessons Learned

- Use checklists
- What didn’t work well
  - Unit and Integration tests
    - Most errors were due to timing rather than implementation

Open Issues

- Even smarter Dispatcher design
  - Considered skipping HallCalls when car is overweight
  - Too complex, current version works well, not enough time

Open Issues

- Controllers assume all other controllers work perfectly
  - Network faults could cause very unpredictable behavior
  - Would need to redesign for reliability
  - Can’t be added on

Suggestions

- Run a script that runs acceptance tests
  - Run them overnight, check for any RT warnings/exceptions/stranded passengers
  - Great for figuring out weird, rare bugs
- Found a bug during acceptance testing?
  - Replace super(“name”, verbose) with super(“name”, true)
  - Easy way to get all log() outputs without adding in a bunch of System.out.println()
  - Make sure you undo this before handin!

Suggestions

- For simulator:
  - Add timing information in .stats outputs
  - Makes it possible to run all tests in bulk instead of running them one by one and checking stdout output