Steven Eisemann

9/25/18

Homework 05

1. As I worked with this code, the changes that I made were mostly to constructors and class functionality. Once I had it working passably this way, I tried to clean up the design and add classes that would remove some of the interdependency that most everything had and hopefully make something run smoother, but this was a much larger job and I was not able to make much headway. Overall, I certainly did not follow the SOLID principles that we learned in class the other day mostly due to lack of time, but even if we had known about them earlier, I feel it would have been a large job to refactor the entire code base. I think that working it into the next homework assignment will help reinforce the concepts, since I feel that currently, I still to not quite understand all of them.

For the first part of the assignment, I only needed to change the Train, CrossingGate, MapBuilder, and TracksDisplay. Of those, the only real functionality changes occurred in Train and CrossingGate. Since I did not want to add a train factory, I decided to change the constructor for the Train class to take speed and image path arguments. Both of these were necessary to allow the second train to move west to east and have the train face the correct direction. In the Gate class, I only had to touch the update class so that it considered which train triggered the update and whether there was a train in the crossing. I feel like I used too many conditionals in this update method, but I was not quite sure how else to go about changing it without having everything have a reference to everything else.

For the second part of the assignment I had to make more major changes. As I was unsure how to go about abstracting functionality away from the cars, car factories, and roads to new classes, I ended up giving roads, factories, and cars references to each other as needed. Beginning with roads, the only major change I made was to how the addCarFactory method was called. I added arguments to the call that controlled if the created factory should spawn cars, and a reference to the road that the factory was created for. This was to allow the notion of intersections to be implemented and to facilitate the passing of cars as they turned off of one road and onto another. I don't dislike how I went about implementing this, as I think it is important that factories know what road they are spitting cars out onto, however, I do not like that I needed to give a road a dead factory just to allow the cars on that road to move. Ideally, I would have changed the way roads and factories work so factories do not control the movement of cars on their road, and given that control to the road itself.

Since I changed the addCarFactory method in Road, I also needed to change the constructor of CarFactory to take the two new parameters. Since I now can have factories that don't produce cars, I needed to add an insertCar method so that cars turning onto the street are able to queue up properly. Properly is loosely used, as I was unable to make it so that the cars in the crossroad were able to "safely" enter the western road. What they do instead is enter the road, then wait for any cars that were in the intersection to get to the safe distance ahead of them and then they will continue. The majority of the time I worked on this homework, I was spent trying to isolate this issue and get it working properly. In the end I ran out of time, so I decided to submit it as is. If I had had more time, I think I might have been able to fix it. It may be that how I handle cars, roads, and factories would need to change before it works

correctly. On a similar note, since cars are told to move in removeOffScreenCars, I had to work with that substantially to ensure that cars that turned are properly removed from the observer chain and passed safely to the new road's factory's list of cars (if I had more time I would have tried to take the list of cars out of the factory…).

Finally, in the Car class I added functionality so that the X-value could be considered when checking lead car and the like. I also gave cars a sense of the direction they are heading and utilized this in their move method. These are things that I thought were missing in the original car class and would stay in there if I were to rework the code in the future. I also gave the cars a turn method, which would allow eligible cars to turn onto the crossroad.

In retrospect, I don't like how this homework turned out. My code is only scalable in some aspects, and would require a lot of digging into the code to add functionality to the rest. I felt like I was too crunched for time, and needed to stick with a lot of the original design decisions. This led to me needing to interconnect a lot of things instead of reworking the entire code base and ultimately led to a product that meets the minimum requirements and not much else.

2. Simply put, as is my code would not scale up nicely. Intersections behave strangely, adding trains, tracks, roads, and car factories all require adding code in multiple places, and plugging in new functionality, such as 4-way stops or streetlights would not be a straight forward process. There are also some places, such as where cars expect to stop of gates or what y-value gates look for trains at, that are hardcoded since the original code did not have a way to easily access such values. Without serious changes, this code could not be used for such an endeavor.