MoSIS Assignment 5 – Statechart

05.12.2018

Alp Tunçay

Homa Hassannia

Introduction

In this assignment, we were given a GUI that represents the dashboard of a train and we were asked to implement a statechart that controls the behaviour of a train. In this statechart we implemented acceleration of a train, train entering a station and emergency break. While implementing these states, we had to consider the limitations such as maximum speed limit and traffic lights on the railway. Train is limited to go at 100 km/h maximum so the train cannot accelerate further. Another limitations is that, whenever passing a yellow light occurs, the speed should be less than or equal to 50 km/h otherwise train would go to the emergency break state. When entering a station, we had to limit the speed to 20 km/h, otherwise again it would go to the emergency break state. If train passes a red light, then it directly goes to emergency break and stops. At the emergency break, the dashboard of the train is non-responsive to any of the inputs coming from the driver. After 5 seconds of cooldown, dashboard becomes responsive again. Also, there is a dead man's button that prompts the driver to press "Poll" button every 30 seconds after it starts moving. If the driver fails to press it within 5 seconds of prompt, then the train goes to emergency break.

In the next section we would like to discuss and analyse the statechart that we implemented.

Statechart Implementation

We started implementing the statechart with the part that is responsible for the acceleration and checking the maximum speed of the train. The section that handles this can be seen in the figure below.

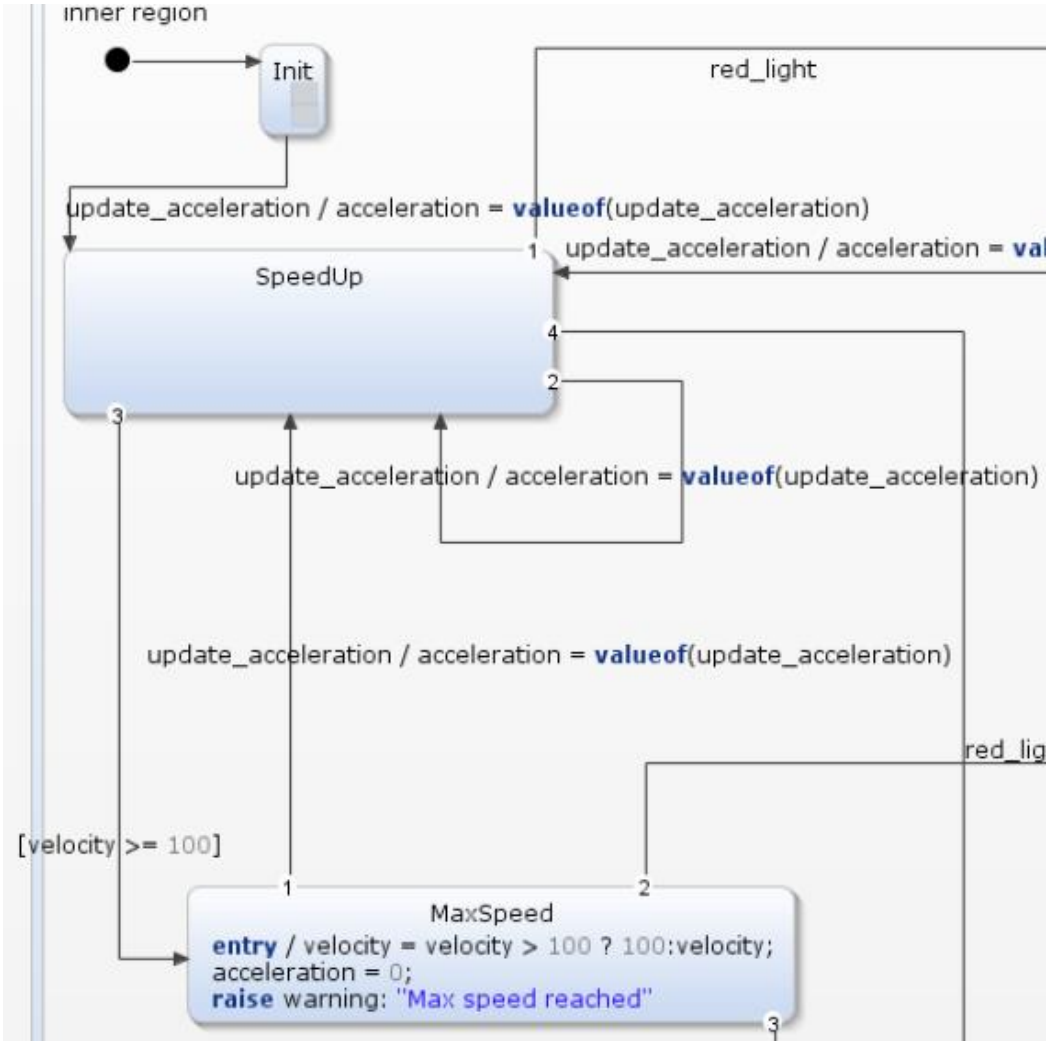

Figure 1: Acceleration and Maximum Speed States

Here after entering the Init state, we can start accelerating by changing the slider on the virtual dashboard. By adding a transition to itself in SpeedUp state, we can update the acceleration each time we change the slider. The state SpeedUp does not necessarily mean that the train would be always increasing its speed whenever it is in that state. It is the state that train reaches whenever there is a change in acceleration, therefore the name chosen for that state does not reflect its real purpose. We have a transition from SpeedUp to MaxSpeed state which is only triggered whenever the speed exceeds 100 km/h. Here, we limit the top speed of the train to 100 km/h. Driver is still able to accelerate but further positive acceleration does not affect the speed of the train in this state.

After handling the acceleration and checking the speed of the train, we moved on to the part which is responsible for controlling the behaviour with respect to changes in the traffic lights. The requirements are:

1. If the train passes red light, it has to go to emergency break.
2. If train is passing a yellow light and its speed is greater than 50 km/h, it must go to emergency break.

3. If the train is cruising at a speed which is less than or equal to 50 km/h while passing a yellow light, then the speed must be limited to 50 km/h. Train should not be able surpass 50 km/h until it sees green light.

The part that handles the controls can be seen below.

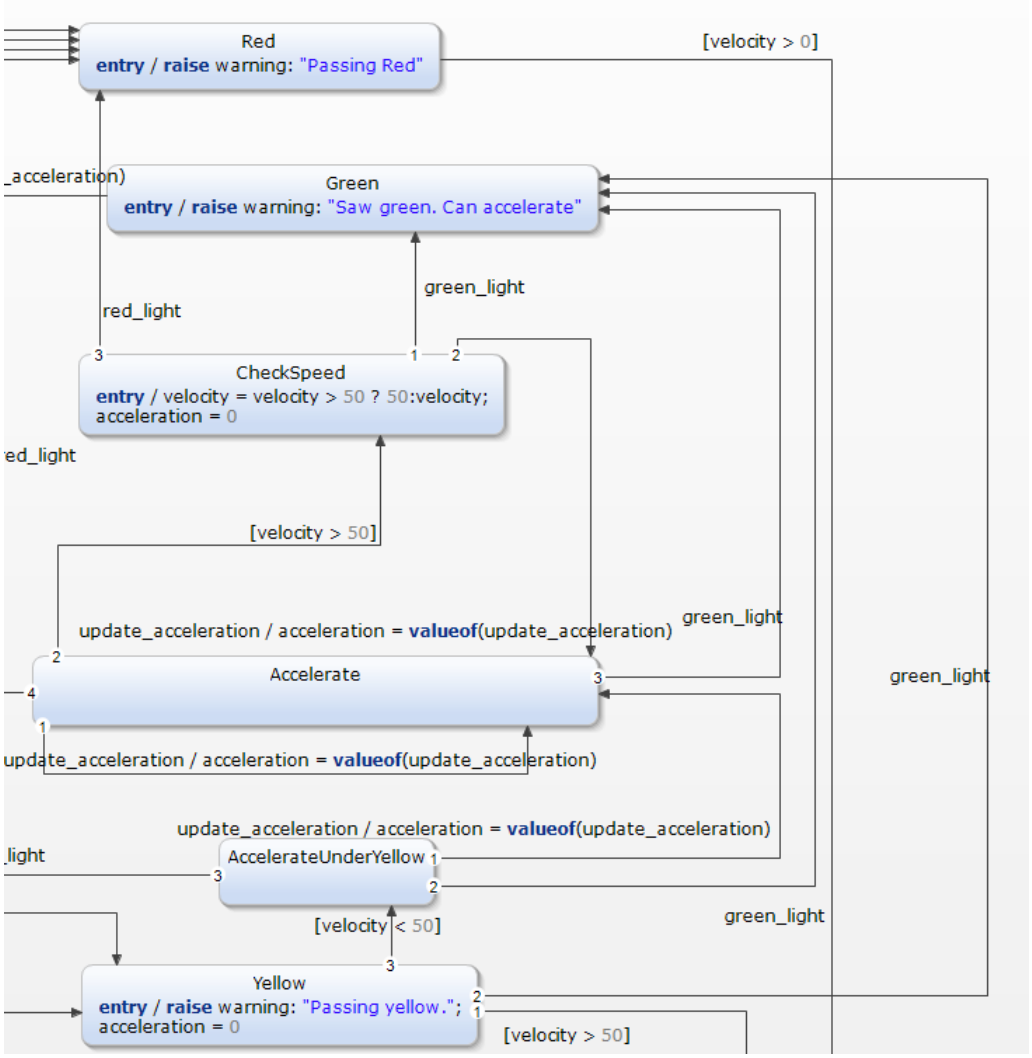

Figure 2: Checking the Speed and Traffic light

This snapshot of the statechart shows all the components that are responsible for checking the requirements for the lights. In the state named Yellow, train can only come to that state whenever there is a yellow light. There is a transition from Yellow state that is triggered only when the speed of the train is over 50 km/h and that transition goes to the Emergency state which we will discuss later. Under the yellow light the maximum speed is limited to 50 km/h but that does not mean that the train cannot accelerate. If the speed is lower than 50 km/h, it goes to the state which is called AccelerateUnderYellow. The reason this state has been added here is because we have to check the speed before allowing the train to accelerate. When the driver changes the slider on the virtual dashboard, it goes to the Accelerate state which actually enables the acceleration. In this state, we disallow the driver by adding the transition which is enabled whenever the speed exceeds 50 km/h. Under that condition it moves over to the CheckSpeed state and limits the velocity to 50 km/h. If the driver wants to slow down in this state, it is allowed but further positive acceleration does not affect the speed. The Green state

is used only under the conditions of yellow light because we have to let the driver speed up whenever there is green light after yellow light. When the train sees the red light it goes to the state Red. The speed must be 0, if the speed is greater than 0, the outgoing transition will be triggered and the train go to Emergency state again.

Following the requirements for lights, we went onto implementing the states where the train comes to a station. First of all, a train does not necessarily have to stop at a station. In our statechart, train is able to pass the station but with a speed which is limited to 20km/h. When the train arrives and stops at the station, it is able to open its doors, and after 5 seconds, the doors should be closed with pressing the "close" button. Then, it will leave the station. If the speed of the train is more than 20 km/h, it will go to emergency break. The snapshot can be seen below.

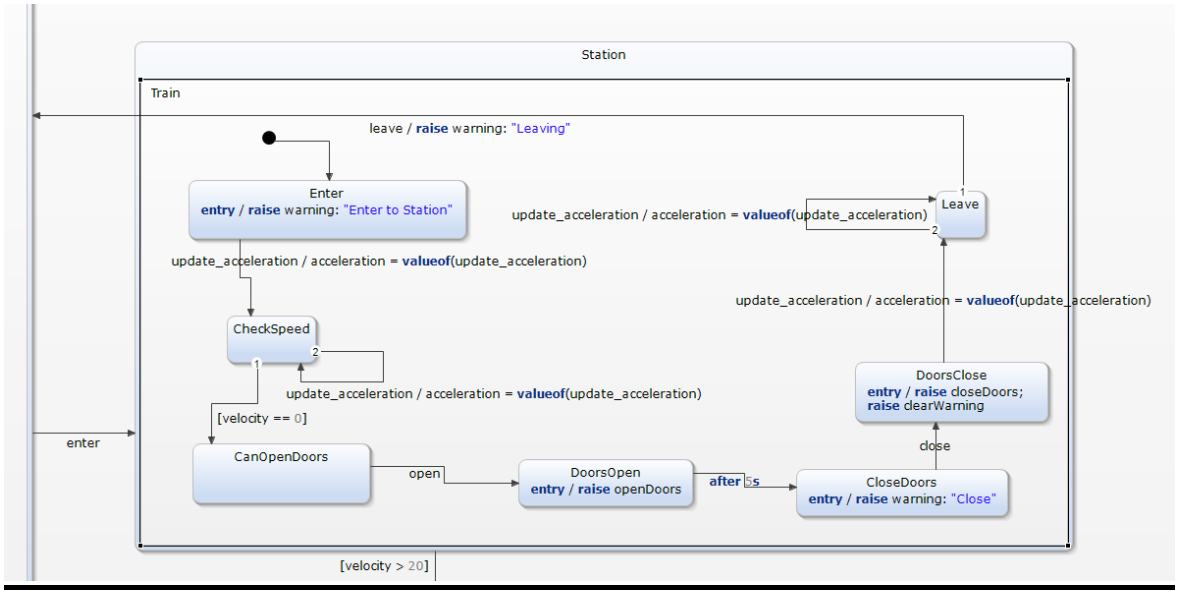

Figure 3: Station

We enter to this state with *enter* transition. Upon entrance, we give the driver to change acceleration in order to be able to stop the train. If the driver exceeds the speed limit of 20km/h, the system will go to Emergency state. If the driver chooses to stop at a station, then there is a transition from CheckSpeed to CanOpenDoors which is triggered when the speed is 0. In this state driver is able to open the doors by pressing the *open* button on the virtual dashboard. The transition from DoorsOpen state is triggered after 5 seconds which ensures that the doors will stay open for at least 5 seconds. After that, driver can close the doors by using the *close* button on the dashboard. Upon closing the doors train can accelerate and leave the station. If at any point the train exceeds 20 km/h in this composite state, then the system will go to the Emergency state.

Also, there is a Pause state. When we press the "pause" button, it will pause the simulation. And after pressing the "continue" button, it will continue with the same speed as it had before. For having the same condition in emergency situation, we added the EmergencyPause for that part too. When we press "pause" button in the emergency situation, it will stop and after pressing the "continue" button, it will come back to the emergency situation.

You can see these 2 parts in figures below.

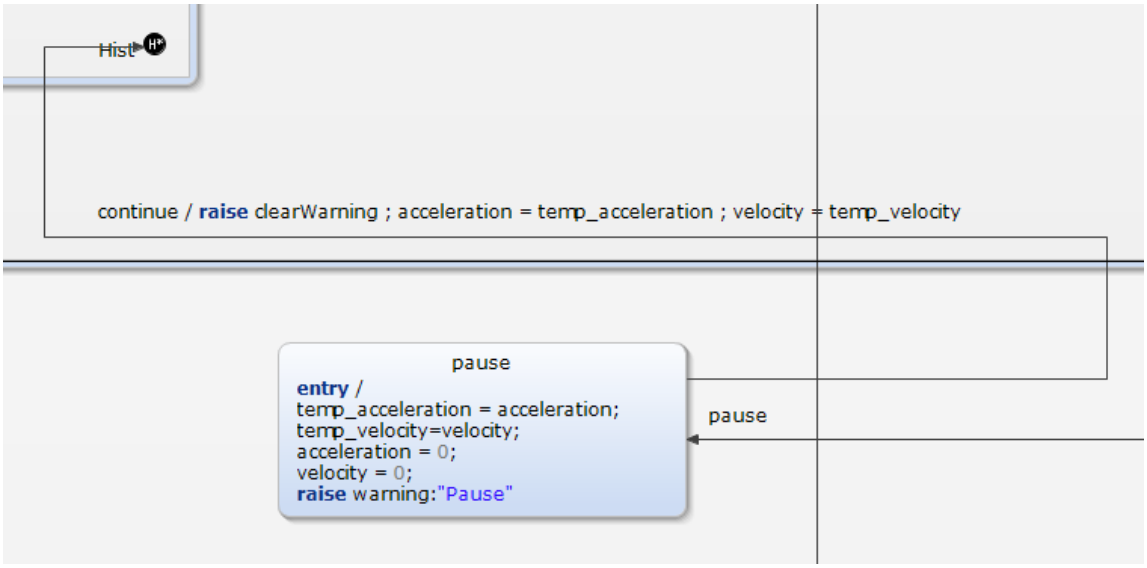

Figure 4: Pause State

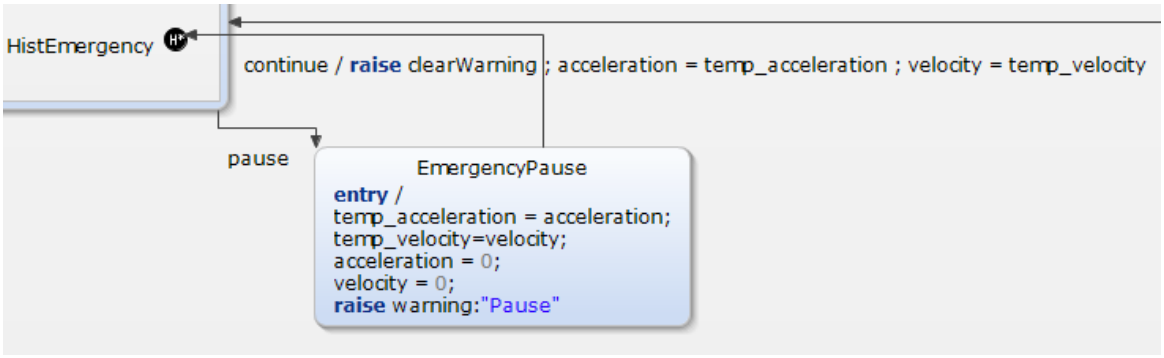

Figure 5: Pause State for Emergency Case

In the emergency states, we use Deep History. The main purpose here is to remember the state where we left off. Whenever the *pause* button is pressed and we are in either pause state, we store the velocity and acceleration values in the temporary variables, and then set those velocity and acceleration values to 0 in order to simulate the pausing. We get back to the state where we left off by pressing the *continue* button. We give the stored values as the new velocity and acceleration values to the state that we left off.

There is another part which is called “Dead Man’s Button”. The driver should press this button every 30 seconds, if it doesn’t happen within 5 seconds, it will go to the emergency break. But, after pressing that button it will have no effect, however, the warning will be cleared from the screen.

You can see this state in the figure below.

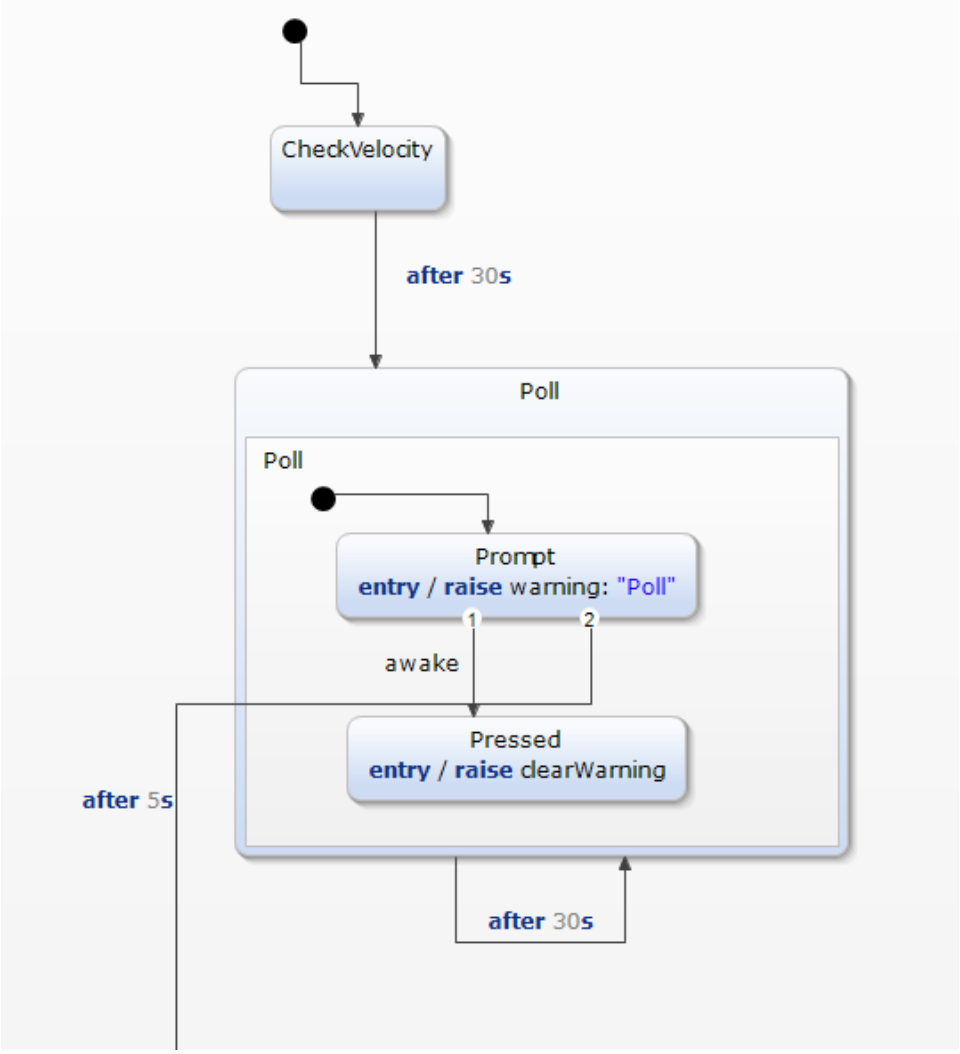

Figure 6: Dead man's Button

Since we implemented the whole system in an orthogonal state, whenever the system starts we are also in this Dead Man's Button. The name CheckVelocity is mistakenly left here. Before, we were checking if the velocity was greater than 0. In that case, the driver would be prompted to press the *poll* button immediately after starting to move. In order to fix that, we trigger that transition after 30 seconds of starting the system. Following that, every 30 seconds the driver would be prompted. If the driver fails to push the *poll* button within 5 seconds of prompting, then the system would go to Emergency state.

The last part of the system is the emergency break. Emergency state is visited whenever a dangerous situation such as passing at red light, disobeying the speed limits, etc. occurs. The aim here is to stop the train as fast as possible and then wait for a certain time for the system to become responsive again.

The snapshot of the system can be seen below.

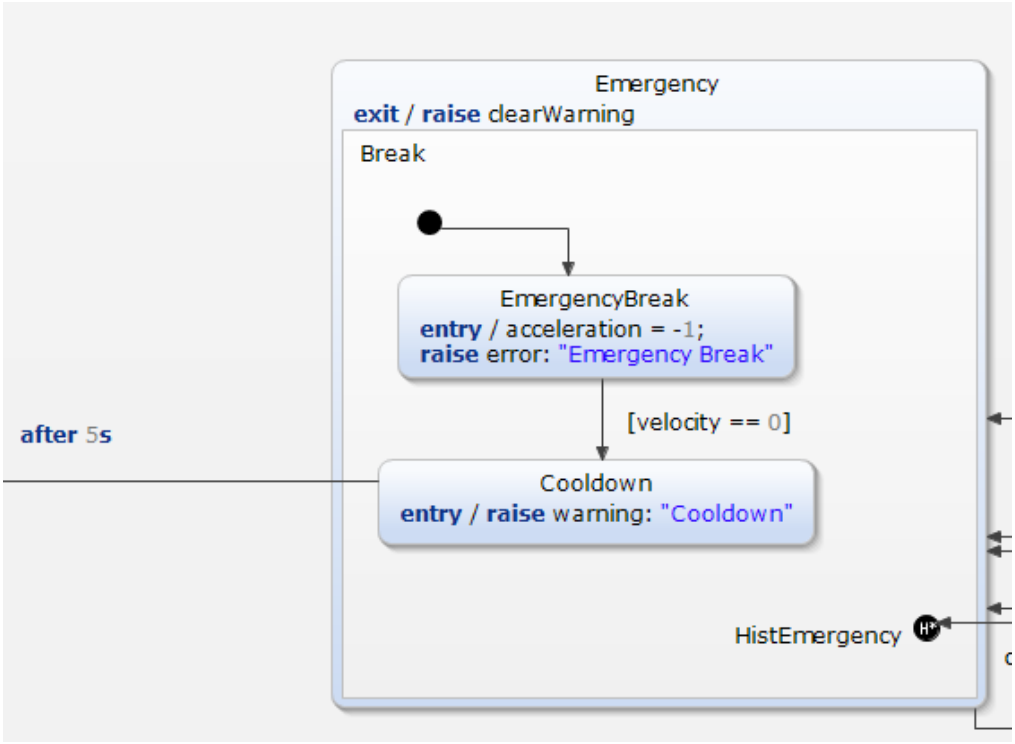

Figure 7: Emergency Break

When system enters this state, initially the acceleration is set to -1 for slowing down. When the velocity reaches 0, the system is in Cooldown period which takes 5 seconds. Within that 5 seconds system does not respond to any of the actions. After the 5 seconds has passed, system goes back to initial state and becomes responsive again.

Also, there is an image from the complete statechart in PNG format which is names model in the folder.