Model of the OC Transpo O-Train Using DEVS Formalism to Determine Average Passenger Trip Time

By: Heather Morris 100686492

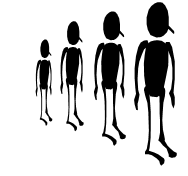
Submitted: October 28th, 2010

**1.0 Introduction**

The Ottawa transit system uses a light transport train, the O-Train, which runs on a track with five stations, one of them being at Carleton. The train schedule is such that the train will arrive at each station every 15 minutes. This project will investigate the average time it takes commuters to wait for the train and then disembark to their final destination. This knowledge would be useful for Carleton students who need to have an estimated travel time to make it to class on time. Also, investigation of the travel time for one and two trains running could be investigated.

1.1 Model Behaviour Description

To model this system there will be a series of generators and queues. The stations can be modelled as being a generator, which will generate people arriving, and a queue, where the people will wait until the train arrives. The generator will have to provide an ID number for the person and the destination stop. The people will be arriving to the station at a random time determined by the distribution model. After the people are generated they will have to be placed in the queue. The queue will be split into five separate sections; one queue for each station. This will ensure that the people go on the train that is going to their stop because the train goes to the same station but can be going one of two directions as well as keeping the final destination known when entering the train. The queue will look at the destination of the person and place them in the appropriate queue. The train itself will also be modelled as a series of queues. The train will be filled on a first come first serve basis. When the train arrives at the next station, the model will take off all the people that want to get off at that station, and inform the station queue that the train has arrived.

C:\Users\Aytchmo\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\WWH015RN\MC900078708[1].wmf  

The people get on the train FCFS

The train arrives at the station – People get off and people start getting on

People waiting for train

The train as well will have to have five different queues, one for each of the possible desired destinations. This saves having to search the queue and take out only the people that want to get off at that stop. This means that the train can be modelled using a coupled model with one input, the people wanting to come on the train, and two outputs, the people coming off the train and the station that the train is arriving.

Bayview Generator

Bayview Queue

Carling Generator

Carling Queue

Carleton Generator

Carleton Queue

Confederation Generator

Confederation Queue

Greenboro Generator

Greenboro Queue

O-Train Model

Bayview passengers

Carling passengers

Carleton passengers

Confederation passengers

Greenboro passengers

Station Arrival

Figure 1: First Draft of the Model Structure in Graphical Form

**2.0 Formal Definition of Model Components**

This model can be represented by using four different atomic models and one coupled model. The atomic models are those for the passenger generators, the stations queues, the station stop generator, and the on train queues. The coupled model represents the O-Train as passengers coming in and passengers coming out. First the Atomic models will be defined, then the couple model.

2.1 Station Arrival Generator Model

This model outputs the station that the train is currently stationed as well as the direction the train is going. The output will be a parsed number where the first number corresponds to the direction and the second the station. The station will have to change such that the train arrives at Bayview every 30 minutes. There are no inputs or external transition function.

out : StationNumber

States : Bayview, CarlingUp, CarlingDown, CarletonUp, CarletonDown, ConfedUp, ConfedDown, Greenboro

δint: **if**(Bayview)

State -> CarlingDown; holdIn( Bayview,Time(0,3,45,0));

**if**(CarlingDown)

State-> CarletonDown; holdIn( CarlingDown,Time(0,3,45,0));

**if** (CarletonDown)

State->ConfedDown; holdIn( CarletonDown,Time(0,3,45,0));

**if**(ConfedDown)

State->Greenboro; holdIn( ConfedDown,Time(0,3,45,0));

**if**(Greenboro){

State->ConfedUp; holdIn(Greenboro,Time(0,3,45,0));

**if**(ConfedUp)

State->CarletonUp; holdIn(ConfedUp,Time(0,3,45,0));

**if**(CarletonUp)

State->CarlingUp; holdIn(CarletonUp,Time(0,3,45,0));

**if**(CarlingUp)

State->Bayview; holdInCarlingUp,Time(0,3,45,0));

λ: **if**(Bayview)

sendOutput( msg.time(), StationNumber, 01 ) ;

**if**(CarlingDown)

sendOutput( msg.time(), StationNumber, 02);

**if**(ConfedDown)

sendOutput( msg.time(), StationNumber, 04);

**if**(CarletonDown){

sendOutput( msg.time(), StationNumber, 03);

**if**(Greenboro){

sendOutput( msg.time(), StationNumber, 15);

**if**(ConfedUp){

sendOutput( msg.time(), StationNumbert, 14);

**if**(CarletonUp)

sendOutput( msg.time(), StationNumber, 13);

**if**(CarlingUp){

sendOutput( msg.time(), StationNumber, 12);

The Arrival Generator model will be tested by setting a simulation time and seeing that the outputs are sent at the appropriate times. That would mean that it is at Bayview on the multiples of 30 minutes.

2.2 The In Train Queues Model

The queues in the O-Train will only accept people that want to get off at the station associated with queue. For example one queue is going to hold only the people that want to get off at Carling. This means that the queue must be able to know which station it is associated to and only accept the passenger numbers that want to get off at that stop. It will also take in the station arrival number to initial the emptying of the queue.

in : StationNumberIn, PassengerIn

out : PassengerOut

States: Passive, FillQueue, EmptyQueue

δext: **if**(PassengerIn )

**if**(value(PassengerIn)%10 == StationDestNumber){

State->FillQueue;

FillQueue( msg.value() ) ;

holdIn(active, Time(0,0,0,0));

**if**( StationNumberIn)

**if**(value(StationNumberIn)%10 == StationDestNumber && Queue Not Empty){

State->EmptyQueue;

holdIn(active, Time(0,0,0,25));

δint: **if**(EmptyQueue)

Delete First Queue Person;

**if** (#PeopleInQueue > 0)

holdIn(active, Time(0,0,0,25));

State->EmptyQueue;

**else**

passivate();

**if**(FillQueue)

passivate();

λ: **if**(EmptyQueue && #PeopleInQueue >0)

sendOutput(msg.time(), PassengerOut, FirstPersonInQueue);

The in train queue model will be tested by sending passenger numbers to the passenger input port and checking that the model only accepts those that are associated with the station are put in the queue. Also, different station numbers will be input to make sure that the queue only outputs when the proper station is input. Also, the case of a person coming and leaving at the same time should never occur since the queue will not be accepting passengers because the train is already at that stop.

2.3 Station Person Generator

The station person generator will create a passenger number, starting at 1 and incrementing upwards. This model will also put a stamp on it to say which station it started at, since there will eventually be five generators creating people. For example, all the people being generated by the Carling generator will have a passenger ID that ends in 2. The generator will also randomly assign a destination for the passenger, which will be added as the last digit of the passenger total number. There are also no inputs and external transition function for this model

out : PassengerInitial

States: active, passive

δinitial: holdIn( active, Time(0, 0, 0, 0));

δint: WaitTime = getdistribution;

Station = getdistribution;

PassengerNumber++;

**if**(StationRangeBayview)

PassengerOutput=PassengerNumber\*100+StationStartNumber\*10+1;

**if**(StationRangeCarling)

PassengerOutput=PassengerNumber\*100+StationStartNumber\*10+2;

**if**(StationRangeCarleton)

PassengerOutput=PassengerNumber\*100+StationStartNumber\*10+3;

**if**(StationRangeConfed)

PassengerOutput=PassengerNumber\*100+StationStartNumber\*10+4;

**else**

PassengerOutput=PassengerNumber\*100+StationStartNumber\*10+5;

holdIn(active, Time(0,0,WaitTime,0));

λ: **if**(PassengerOutput%10 == StationStartNumber)

**else**

sendOutput( msg.time(), PassengerInitial, PassengerOutput);

The person generator only needs to be tested to ensure that the passenger numbers are being generated and output and that the second last number corresponds to the associated station number.

2.4 Station Queues

The station queues will hold all the people until the train arrives at their station. It will then empty the queue and output the passenger values for the O-Train model. The queue will also check to make sure that there are no people wanting to arrive at the starting station when the passengers are input.

in : StationNumber, PassengerIn

out : PassengerOut

States: passive, EmptyQueue, FillQueue

δext: **if**(PassengerIn && !EmptyQueue)

**if**(value(PassengerIn)%10 != StationDestNumber){

State->FillQueue;

Put person at back of Queue ;

holdIn(active, Time(0,0,0,0));

**if**( (StationNumber )){

**if**(value(StationNumber)%10 == StationDestNumber && People in Queue){

State->EmptyQueue;

holdIn(active, Time(0,0,0,25));

δint: **if**(EmptyQueue){

Take out first person

**if** (elements.size() > 0){

holdIn(active, Time(0,0,0,25));

State->EmptyQueue;

**else**

passivate();

**if**(FillQueue)

passivate();

λ: **if**(EmptyQueue && People in Queue)

sendOutput(msg.time(), PassengerOut, First Person in Queue);

The station queue model will be tested by sending passenger numbers to the passenger input port and checking that the model only accepts those that are associated with the station are put in the queue. Also, different station numbers will be input to make sure that the queue only outputs when the proper station is input. Also, the case of a person coming and leaving at the same time must be investigated, since the person getting on should fill the queue and the person getting off should be output.

2.5 O-Train Coupled Model

The O-Train coupled model will have an input which is the passengers from the stations and output the passengers that wanted to get off at that station.

in : PassengerIn

out : PassengerOut StationNumber

components : BayviewPassengers@InTrainQueue CarlingPassengers@InTrainQueue

components : CarletonPassengers@InTrainQueue ConfedPassengers@InTrainQueue

components : GreenboroPassengers@InTrainQueue

components : StationArrival@StationArrivalGenerator

IC: Link : StationNumberOut@StationArrival StationNumberIn@BayviewPassengers

Link : StationNumberOut@StationArrival StationNumberIn@CarlingPassengers

Link : StationNumberOut@StationArrival StationNumberIn@CarletonPassengers

Link : StationNumberOut@StationArrival StationNumberIn@ConfedPassengers

Link : StationNumberOut@StationArrival StationNumberIn@GreenboroPassengers

EC: Link : StationNumberOut@StationArrival StationNumber

Link : PassengerIn PassengerIn@BayviewPassengers

Link : PassengerIn PassengerIn@CarlingPassengers

Link : PassengerIn PassengerIn@CarletonPassengers

Link : PassengerIn PassengerIn@ConfedPassengers

Link : PassengerIn PassengerIn@GreenboroPassengers

Link : PassengerOut@BayviewPassengers PassengerOut

Link : PassengerOut@CarlingPassengers PassengerOut

Link : PassengerOut@CarletonPassengers PassengerOut

Link : PassengerOut@ConfedPassengers PassengerOut

Link : PassengerOut@GreenboroPassengers PassengerOut

2.6 Top Coupled Model

The top model will incorporate all the atomic models and the O-Train coupled model to simulate the O-Train over a certain time.

out : PassengerArrived PassengerInitial

components : BayviewStation@StationGenerator CarlingStation@StationGenerator

components : CarletonStation@StationGenerator ConfedStation@StationGenerator

components : GreenboroStation@StationGenerator

components : BayviewQueue@StationQueue CarlingQueue@StationQueue

components : CarletonQueue@StationQueue ConfedQueue@StationQueue

components : GreenboroQueue@StationQueue

components : OTrain

IC: Link : PassengerInitial@BayviewStation PassengerIn@BayviewQueue

Link : PassengerInitial@CarlingStation PassengerIn@CarlingQueue

Link : PassengerInitial@CarletonStation PassengerIn@CarletonQueue

Link : PassengerInitial@ConfedStation PassengerIn@ConfedQueue

Link : PassengerInitial@GreenboroStation PassengerIn@GreenboroQueue

Link : StationNumber@OTrain StationNumberIn@BayviewQueue

Link : StationNumber@OTrain StationNumberIn@CarlingQueue

Link : StationNumber@OTrain StationNumberIn@CarletonQueue

Link : StationNumber@OTrain StationNumberIn@ConfedQueue

Link : StationNumber@OTrain StationNumberIn@GreenboroQueue

Link : PassengerOut@BayviewQueue PassengerIn@OTrain

Link : PassengerOut@CarlingQueue PassengerIn@OTrain

Link : PassengerOut@CarletonQueue PassengerIn@OTrain

Link : PassengerOut@ConfedQueue PassengerIn@OTrain

Link : PassengerOut@GreenboroQueue PassengerIn@OTrain

EC: Link : PassengerInitial@BayviewStation PassengerInitial

Link : PassengerInitial@CarlingStation PassengerInitial

Link : PassengerInitial@CarletonStation PassengerInitial

Link : PassengerInitial@ConfedStation PassengerInitial

Link : PassengerInitial@GreenboroStation PassengerInitial

Link : PassengerOut@OTrain PassengerArrived

This simulation will be run over a certain set time; two hours should be enough to get enough data for an average travel time. The average time will be found from the output to the top model called passenger initial which will have the initial passenger time and comparing it to the output passenger arrived.

**3.0 Testing of the Atomic Models**

First the station generation model was checked. The model was run for 45 minutes in order to ensure that the output values were on a 15 minute cycle from Bayview to Greenboro. The output values can be seen in figure 2, which is an animation of the log file after running the StationNumberGenerator.ma file. It can be seen that the output values are as expected. The train is going from up to down and is cycling every station every 15 minutes.

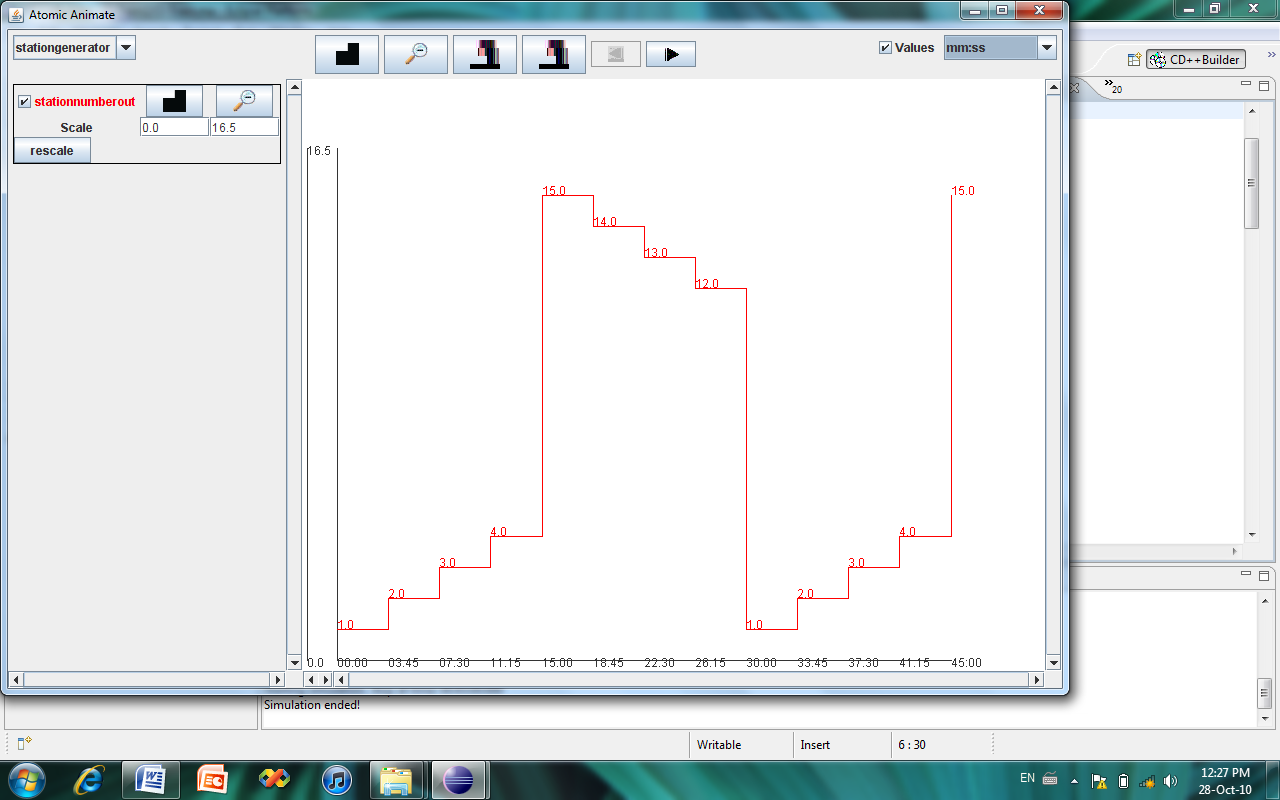


Figure 2: Station Number Output Test over 45 minutes

The in train queue models were then checked using an event file. The event file is that as attached in appendix A. The event file was sending values to the passenger in port as well as various values for the station arrival. The simulation results are those as seen in figures 3 and 4, also animated from a log file. As you can see the station will only admit people if the last digit of the passenger number is the same as the associated number for the queue. Also, when the station number associated to that queue is input all the people in the queue are exiting one after the other in quick succession.

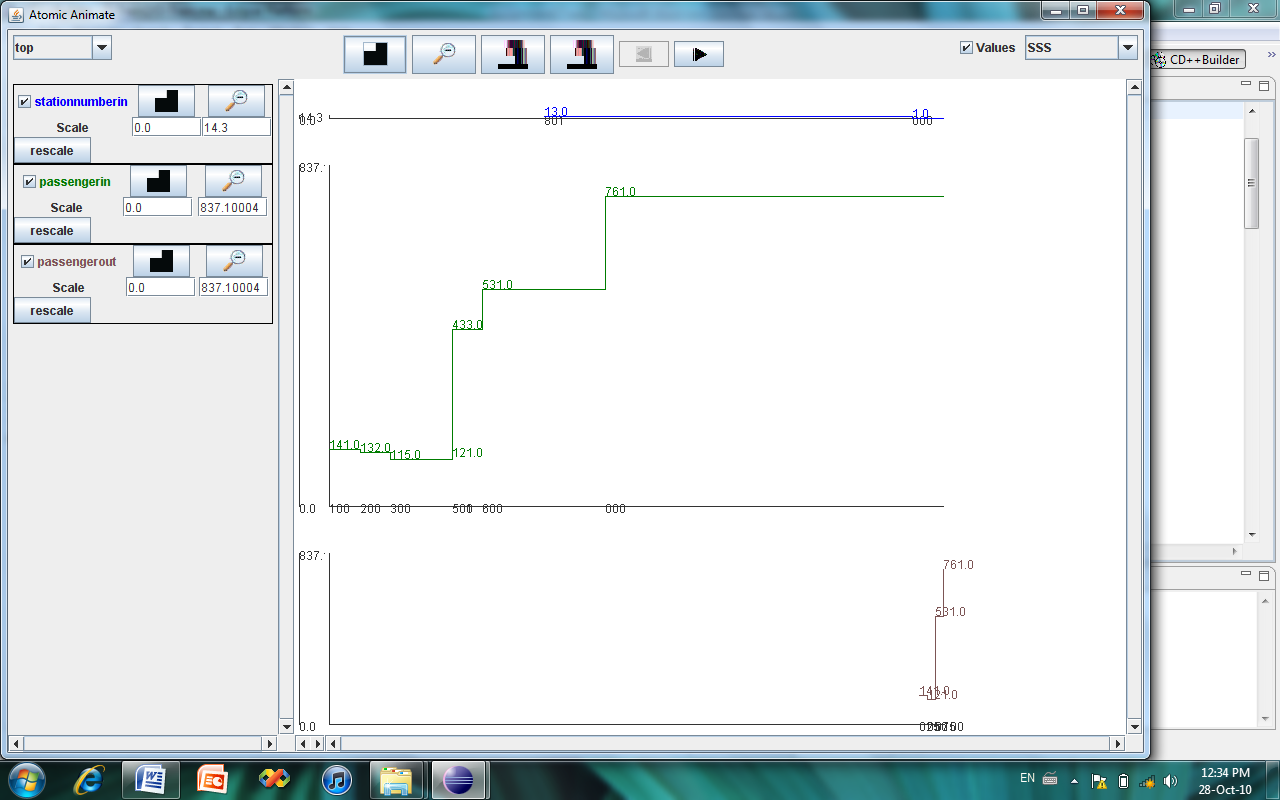


Figure 3: In Train queue with Station Value of 1

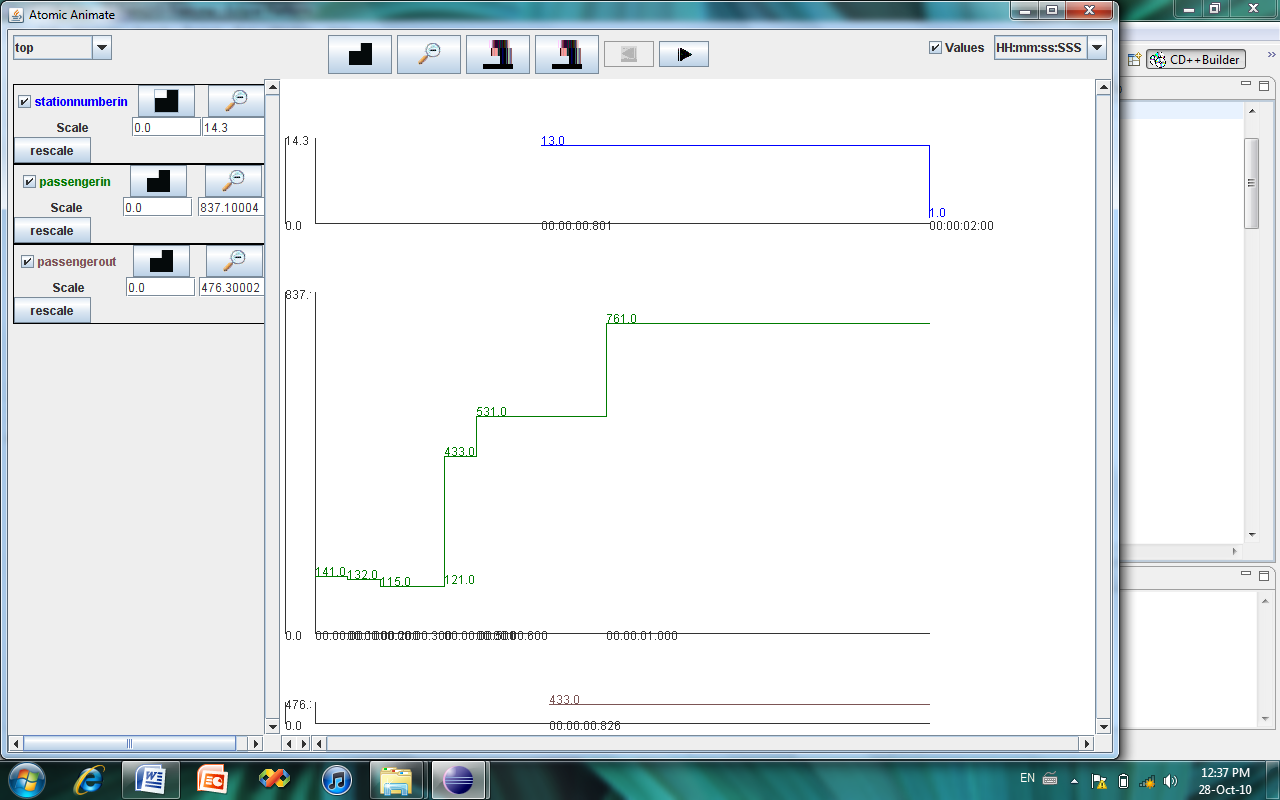


Figure 4: In Train Queue with Station Number 3

The station generator was tested over a certain time. It can be seen that the passenger output number is increasing sequentially, as well as the second last digit corresponds to the associated queue number. If the mean and standard deviation numbers were changed the people come less frequently. In this example the mean was set as 5 seconds with a standard deviation of 5 seconds.

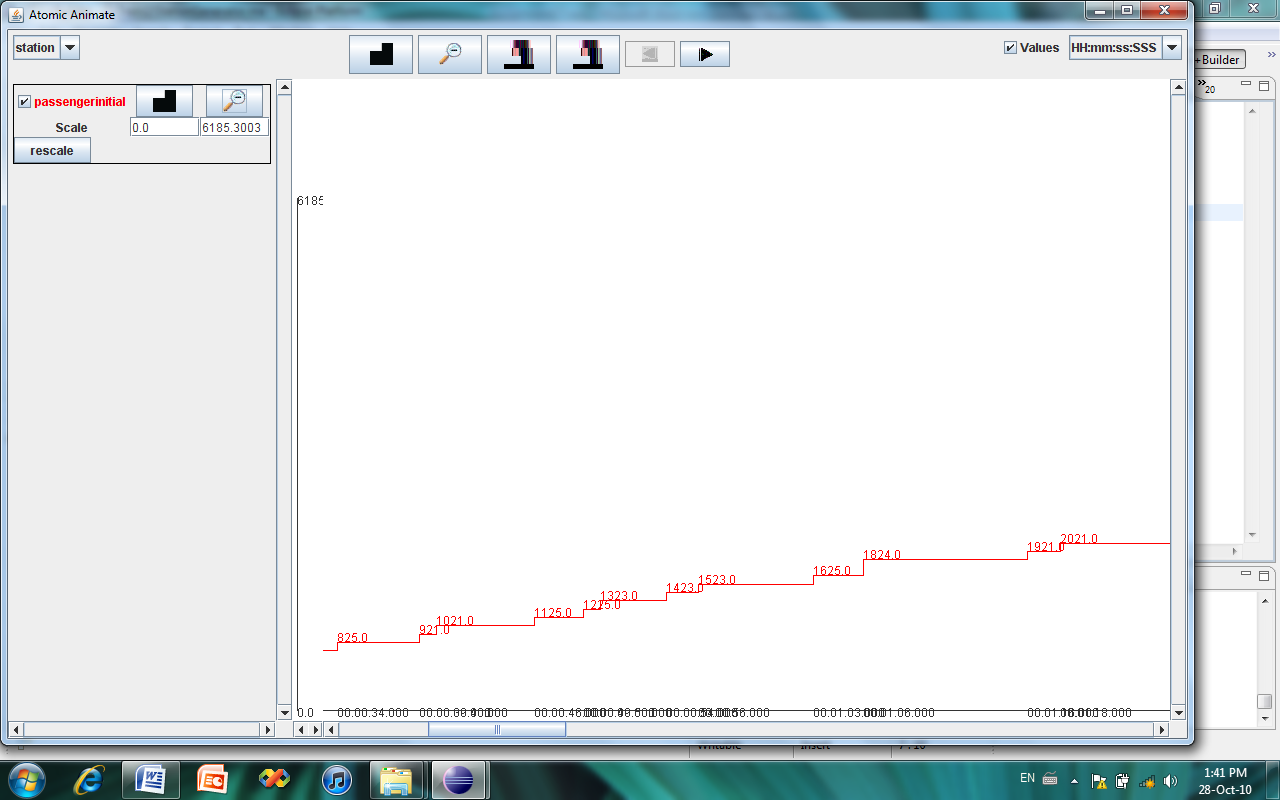


Figure 5: Station Generator with a Mean Time of 5 and Standard Deviation of 5, Station Number 2

One point to note in the figure is that there are some numbers in the sequence missing, such as passenger number 17. This is due to the fact that it had the same destination as where it was being created, meaning that it was being created at station 2 and wanted to get off at station 2. This is not a desired result and by not outputting the passenger data from the start it doesn’t clutter the results with people coming in to just leave.

The station queues were tested using a similar event file to that for the in train queues. The event file data is attached to the report in Appendix B. The queue is supposed to input passenger data into the queue, with a second check that there aren’t people that want to get off at this stop. When the station number input value is received it is to initiate the emptying of the queue, which should have everyone other than those with the same station number. Also, if the queue is receiving an input from the passenger in port it is to ignore the person coming in and continue emptying the queue. As you can see from the figure 6, the model fills the queue with everyone but those that want the same station. When an input to passenger in was initiated the passenger data was dropped, and when the station number input was that of the station number the queue emptied.

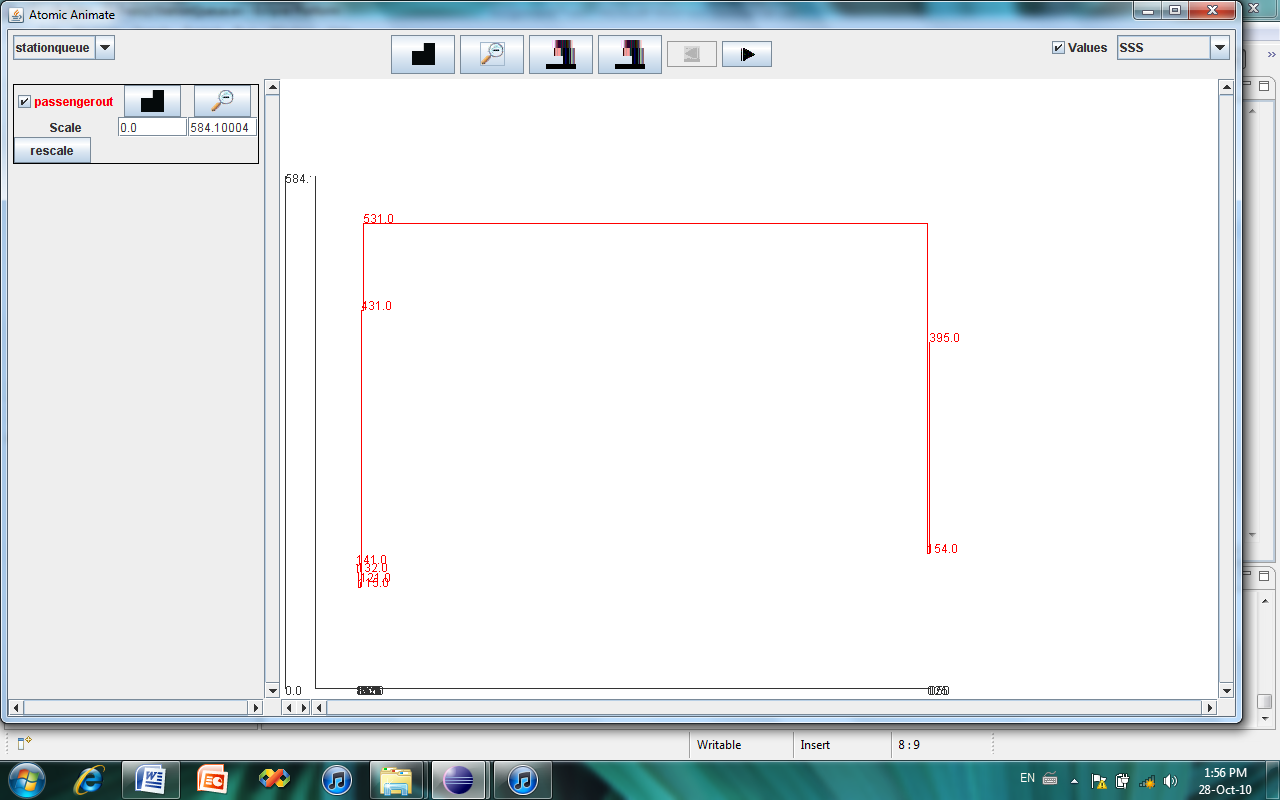


Figure 6: Station Queue of Value 3 Log Data from Event File

**4.0 O-Train Elapse Time Analysis**

This model was created to view the average time it takes a passenger on the O-Train from arriving at the station to getting to their desired destination. First, an analysis of the behaviour of the model will be investigated to ensure that the passenger data is getting transferred correctly. What one would expect to happen is when the train arrives at a station, there is a bunch of passenger data with the same last number, the destination number, being sent to the output port of the O-Train coupled model. Also, there should be a clump of passenger in data to the O-Train coupled model with the same second last number as the station number. These behaviours can be seen in figure 7, where the simulation was performed over a 60 minute elapse time.

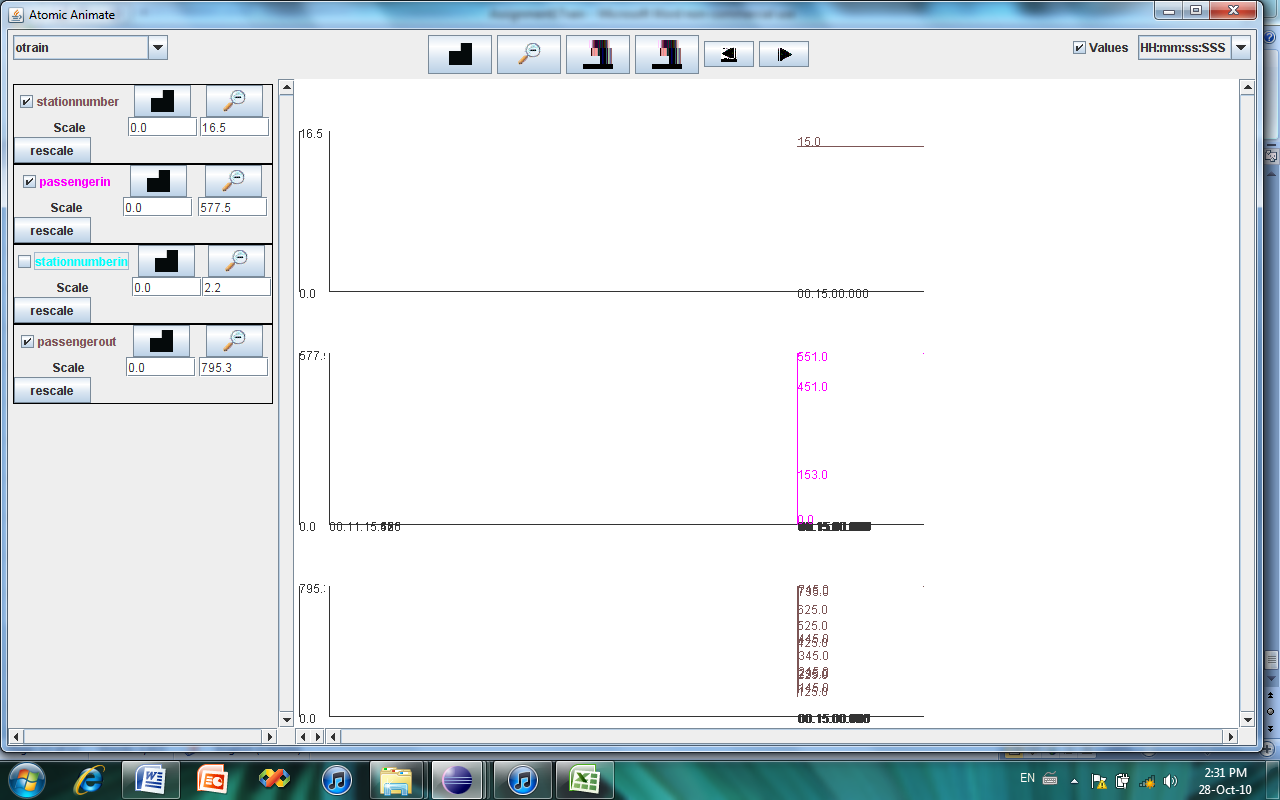


Figure 7: Passenger Data Transfer upon Station Arrival

After confirming that the model is transferring data in the correct times, the average elapse time can be investigated. The initial conditions of the model were such that there were more passengers appearing at Bayview, Carleton, and Greenboro. This was done to reflect the real situation. To do this the output data for the top model was exported into excel and placed in a spreadsheet, in which a section is presented in Table 1.



Table 1: Output Data from the O-Train Simulation

Another check on the model is that the passenger values have been matched and none of the passenger data is getting lost in the simulation. From this table, the passenger initial port is connected to the output port of the station generators, so this is the first instance of the passenger data. The passenger arrived port is connected to the output port of the O-Train coupled model. This would be the last instance of the passenger data. The difference between these two times is the total time it took to get from their start location to their desired location. It is also important to note that this model is prepared for only one O-Train at the same time, where actually there are two travelling in opposite directions. The average total time for a passenger was calculated to be about 17 minutes, which is just above the time it takes for the train to go one length of the track.