Universitat Autònoma de Barcelona

Basic Models

AnyLogic

Author: Alejandro RodrIguez Grau

Tutor: Prof. Dr. Angel A. Juan

Full Course Index

- Introduction to AnyLogic
- <u>Basic models</u>
- Heterogeneous model
- Advanced model

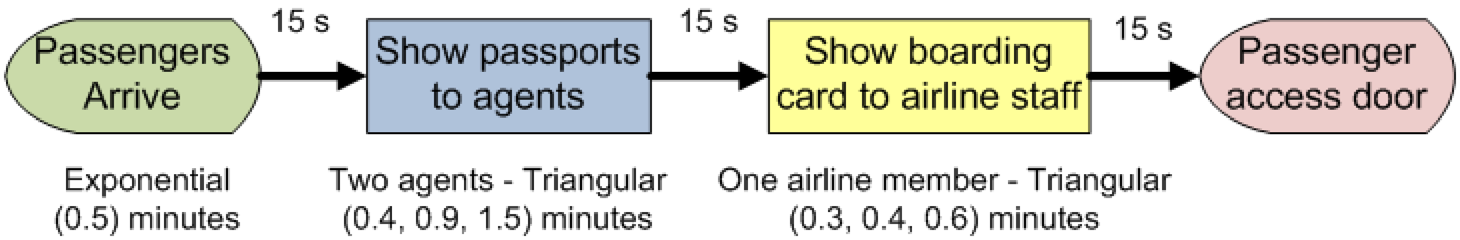
Basic Models

- **Exercise 1**
  - Access Control Point
    - Overview of the problem
    - Create the Model
    - Run and Results
- **Exercise 2**
  - Check-in Point
    - Overview of the problem
    - Create the Model
    - Run and Results

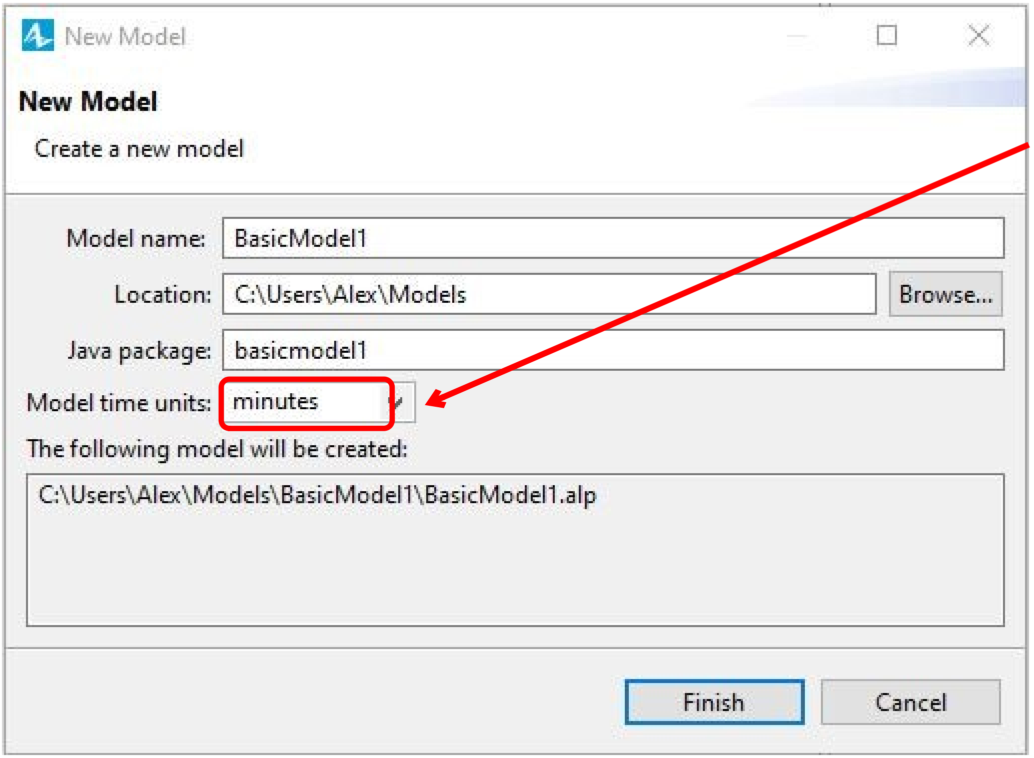
1. A simple access control point

- Process overview: Passengers arrive to a passport control point. They wait in line and show their passports to any of two agents. Then, passengers move to the boarding card control. After validating their cards, they can proceed to the boarding door.

- Assumptions: (a) Passengers interarrival times follow an Exponential(0.5) minutes; (b) the time an agent takes to interact with the passenger follows a Triangular(0.4, 0.9, 1.5) minutes; (c) the time the airline staff member takes to interact with the passenger follows a Triangular (0.3, 0.4, 0.6) minutes; (d) the travel time between each process is 15 seconds; (e) passengers will wait in a single waiting line on a FIFO basis; (f) there is no limit to the length of the waiting lines.



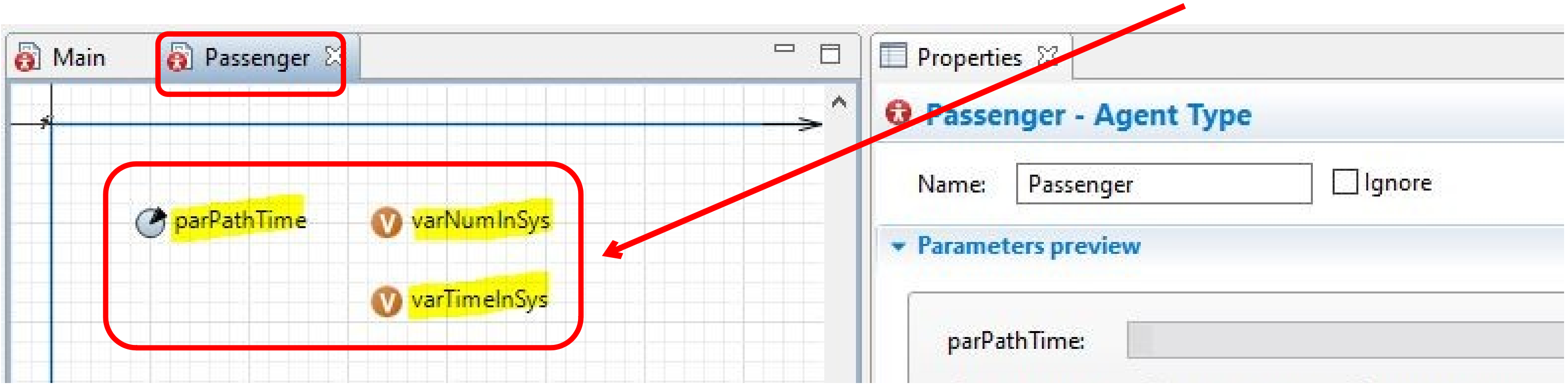
2. Create a new Model, Add Agent Type



This model will work best if each step represents 1 minute inside the simulation.

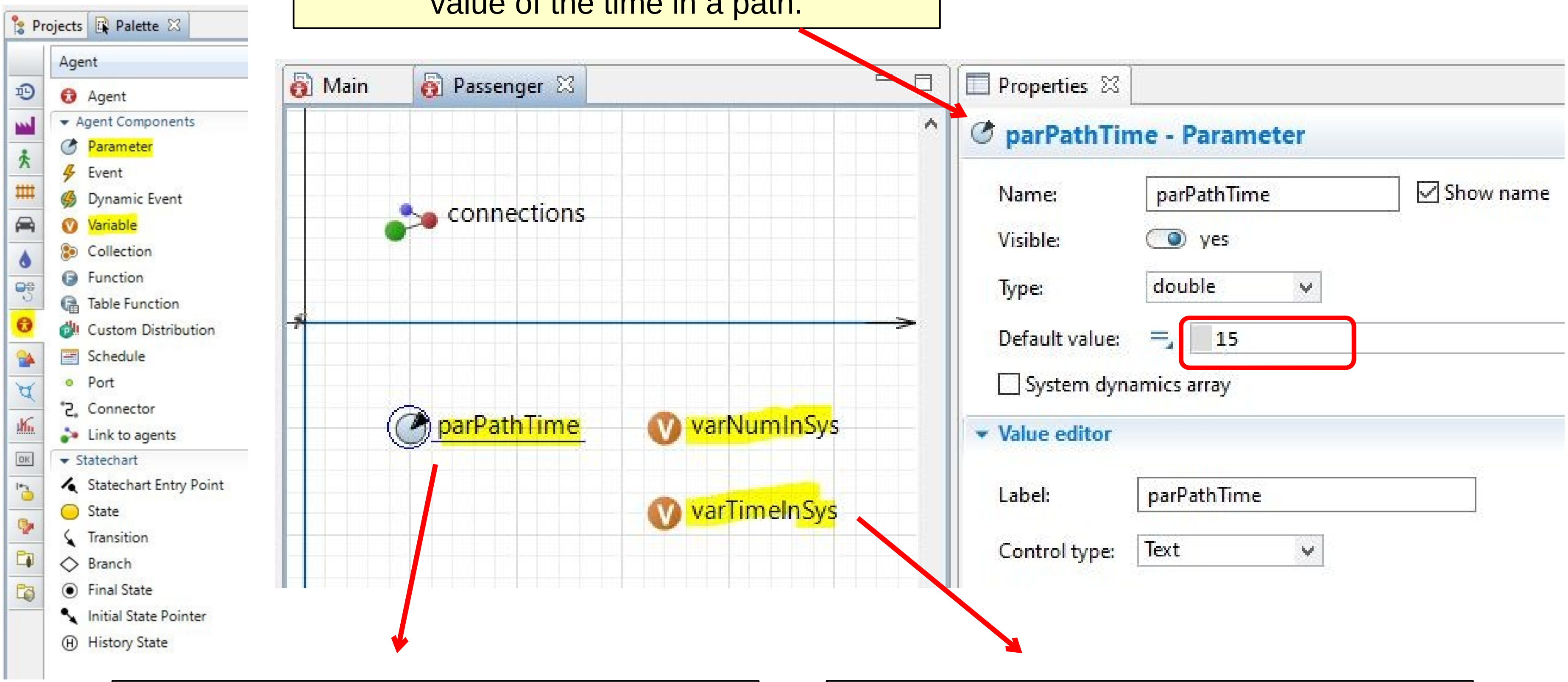
Drag & Drop an *Agent Type* from the Agent *Pallete.*

Add Parameters and Variables on new Agent Type: **Passenger**



3.Parameters & Variables for optional use

We could define a parameter to fix the value of the time in a path.

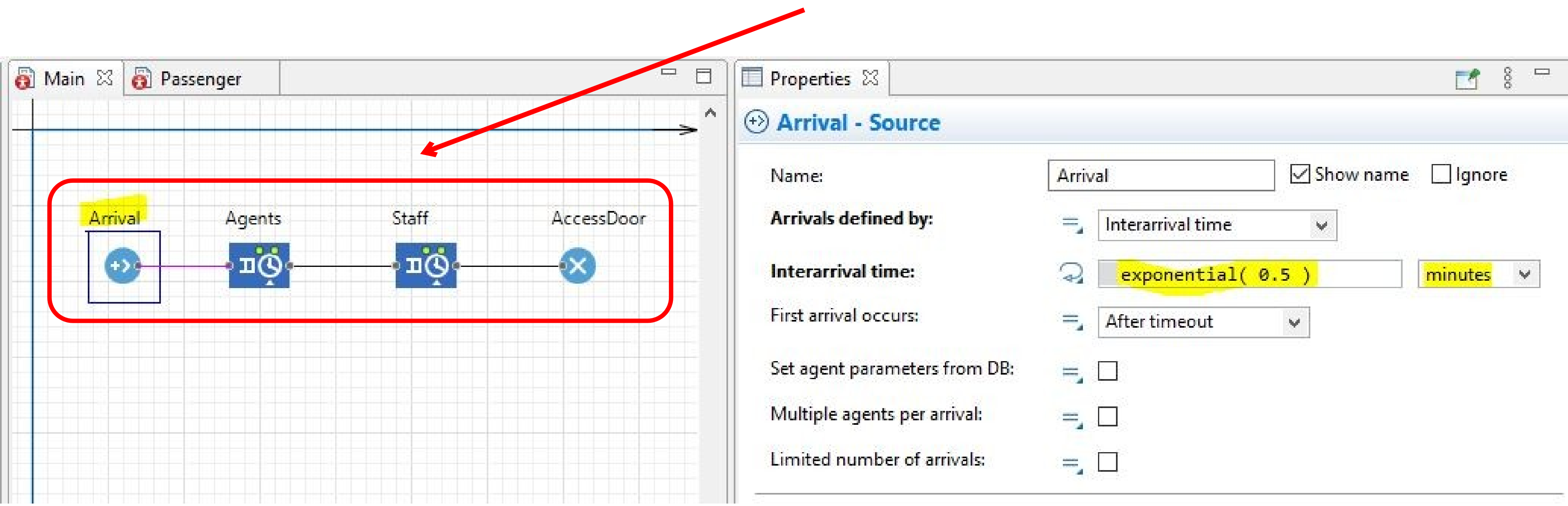


But in our case, we will use connectors into a *Delay* block, where *Entities* will wait those 15 virtual seconds, before proceeding to the next step.

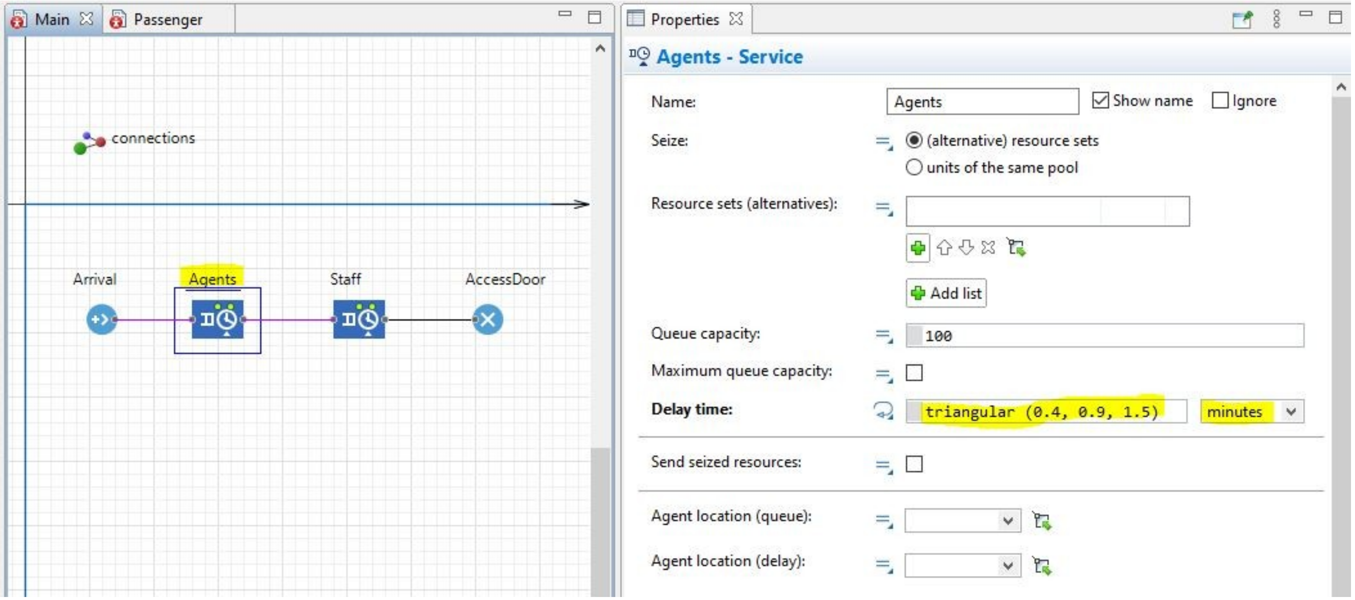
Our Variables will be used, as in the first model, to obtain and print results. So for now, lets go back to the *Main* tab, and create the proposed model.

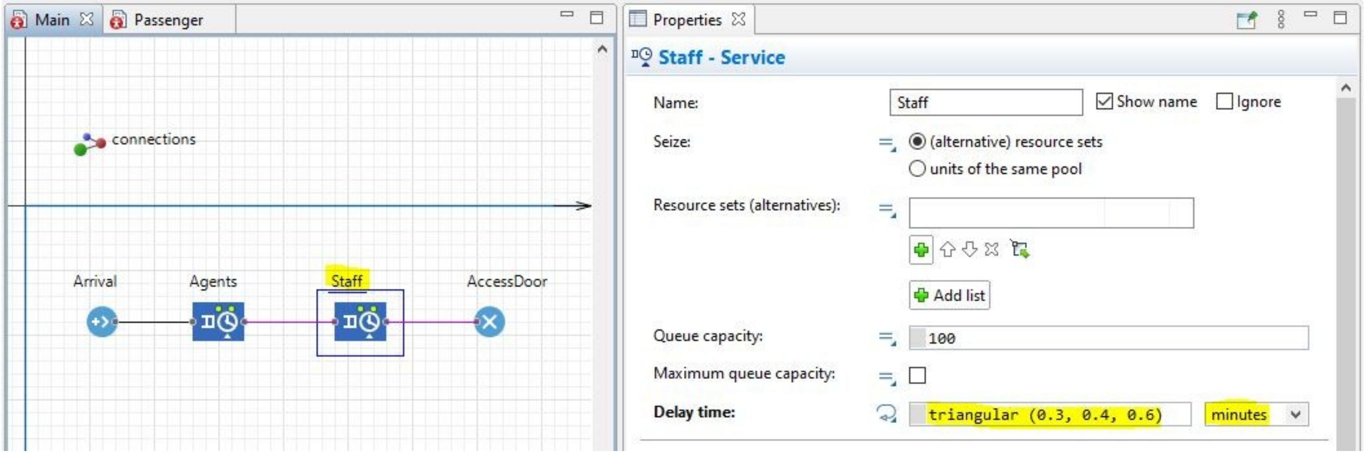
4. Implement your Model (1/2)

Drag & Drop the *Source*, *Services* (2) and *Sink*, change their names accordingly and set up the defined parameters on the first slide of this exercise



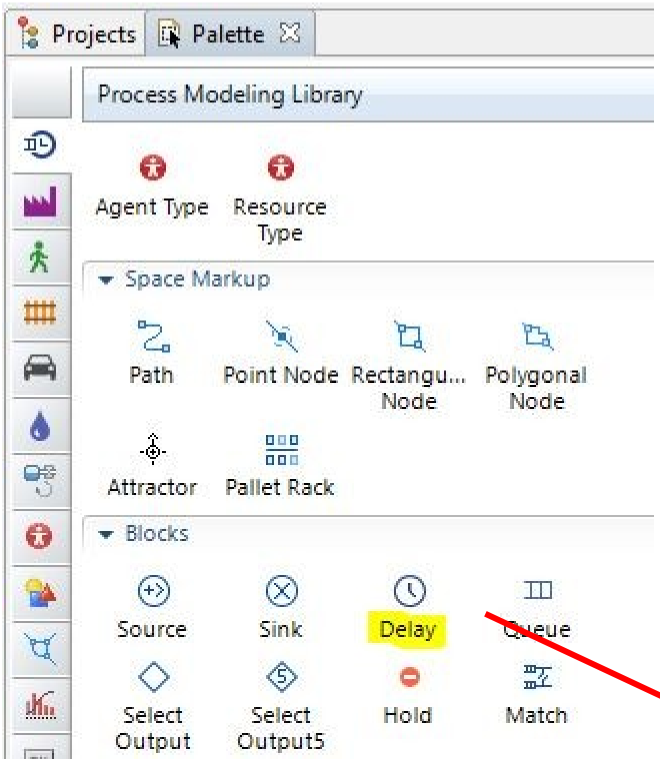
4. Implement your Model (2/2)

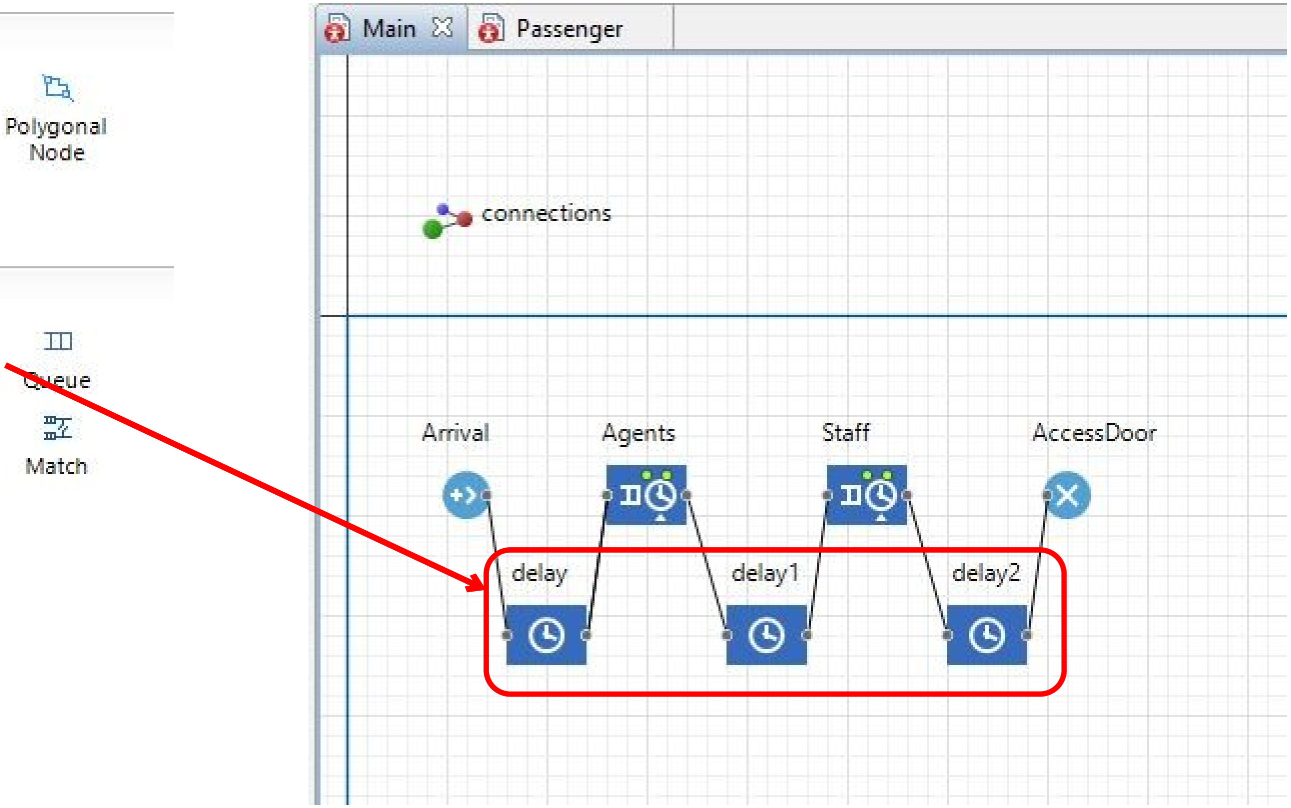




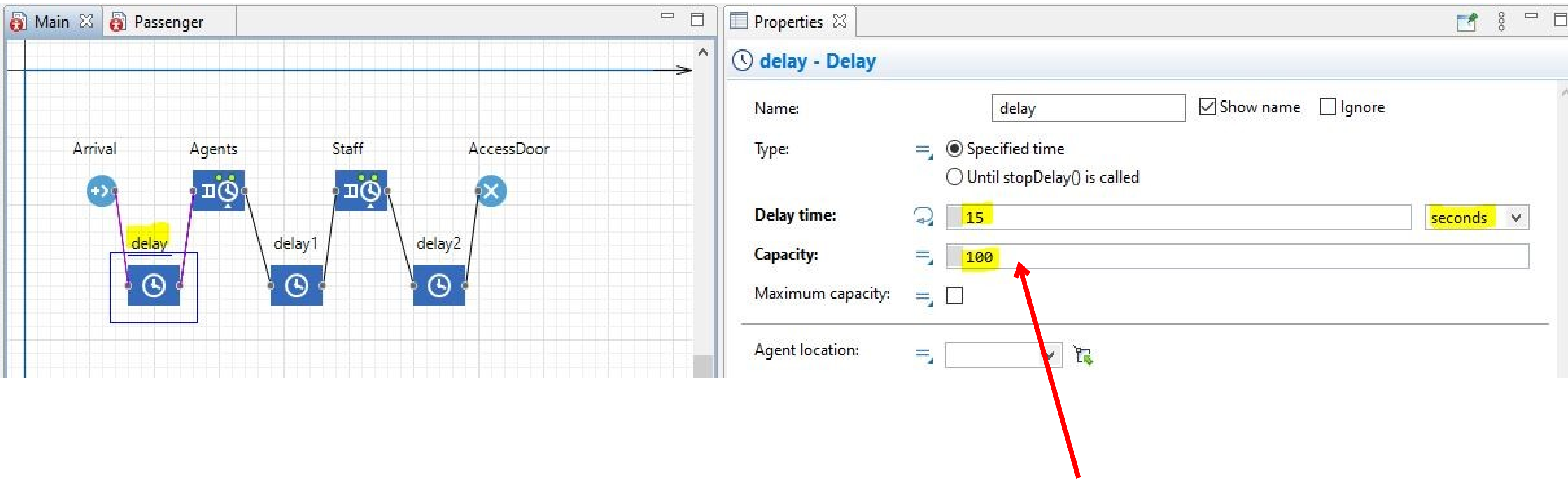
5. Add Delay (1/2)

Now we already have a simulation to run, entities have an arrival rate, processing time on the server and a sink to close the cycle.
Still, entities are **instantly** going from the "out" nodes to the "in" nodes. We must add a *Path* or a *Delay*, where we can specify the time that it takes for a passenger to go from one place to another (in this case, a fixed time).





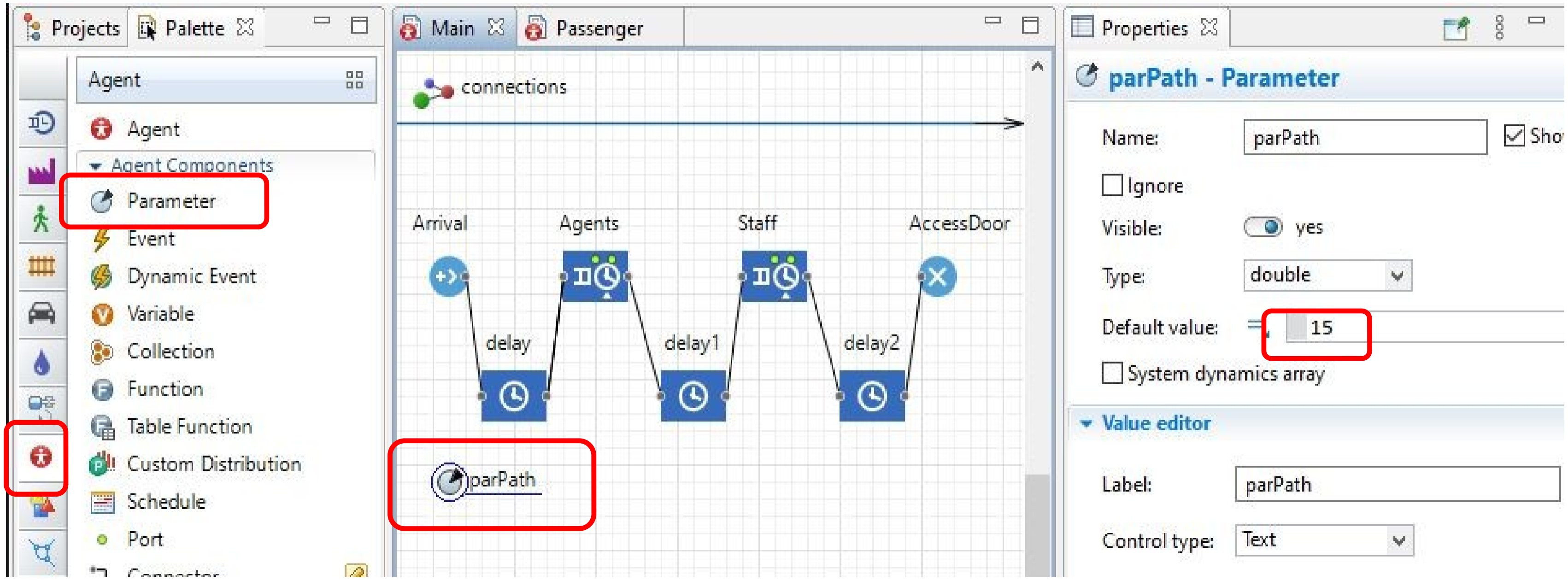
5. Add Delay (2/2)



Once defined the waiting time, we must set a high enough capacity, because if we leave only 1 (by default), no other entity will be able to do that path until the one in it reaches the exit.
Do this for every Delay block

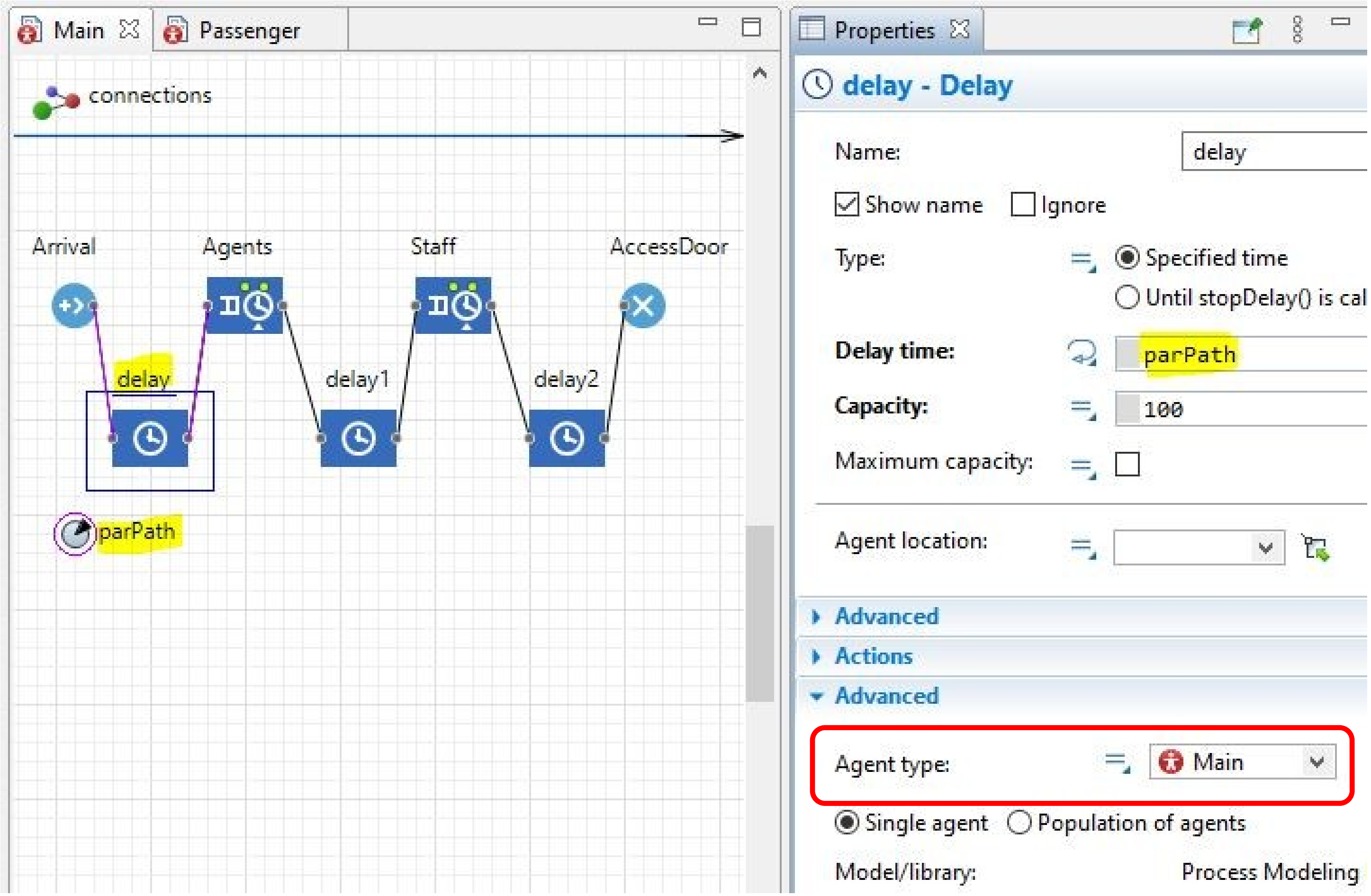
6. Use *Parameters* in Main (1/2)

Now that we know how to input data, lets learn how to substitute a number that appears several times and can by compressed into a Parameter inside the "*Main*" *Agent* screen.



6. Use *Parameters* in Main (2/2)

Use that parameter instead of the numbers in each Delay block.

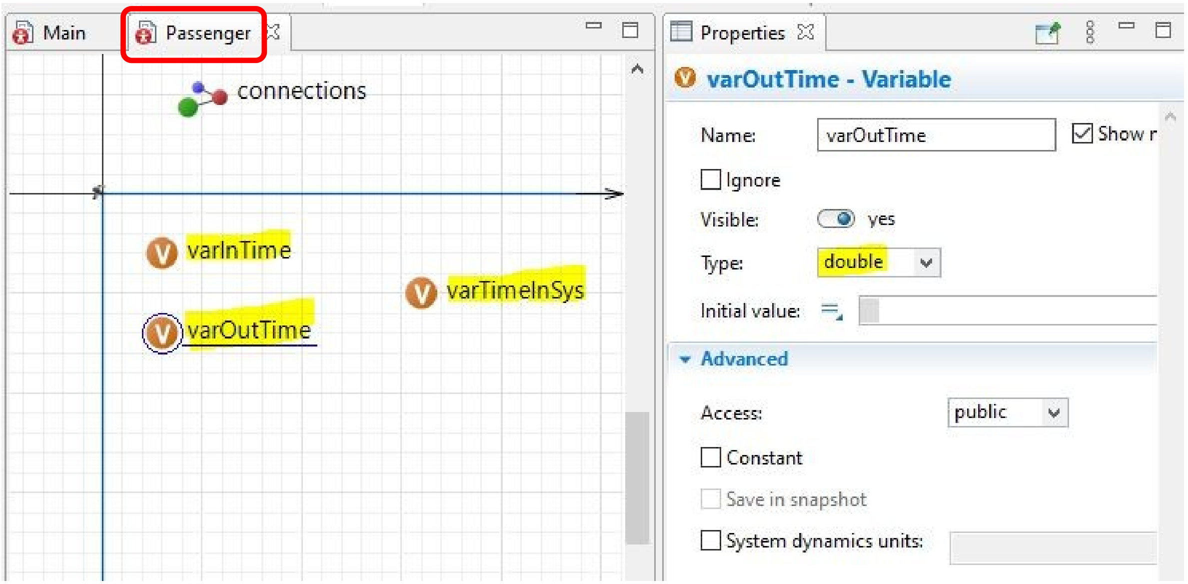


Remember to change the *Agent Type* on the *Delay* blocks so the program can find its way to the parameter!

7. Add Results

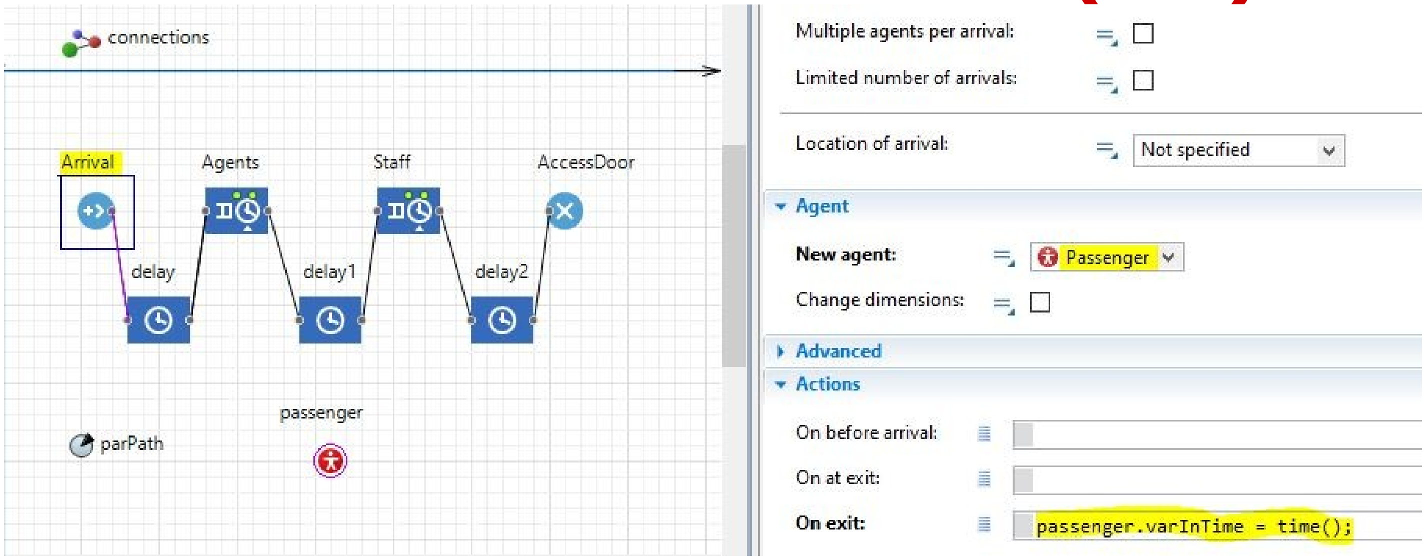
Time to revisit the *Passenger Agent.*

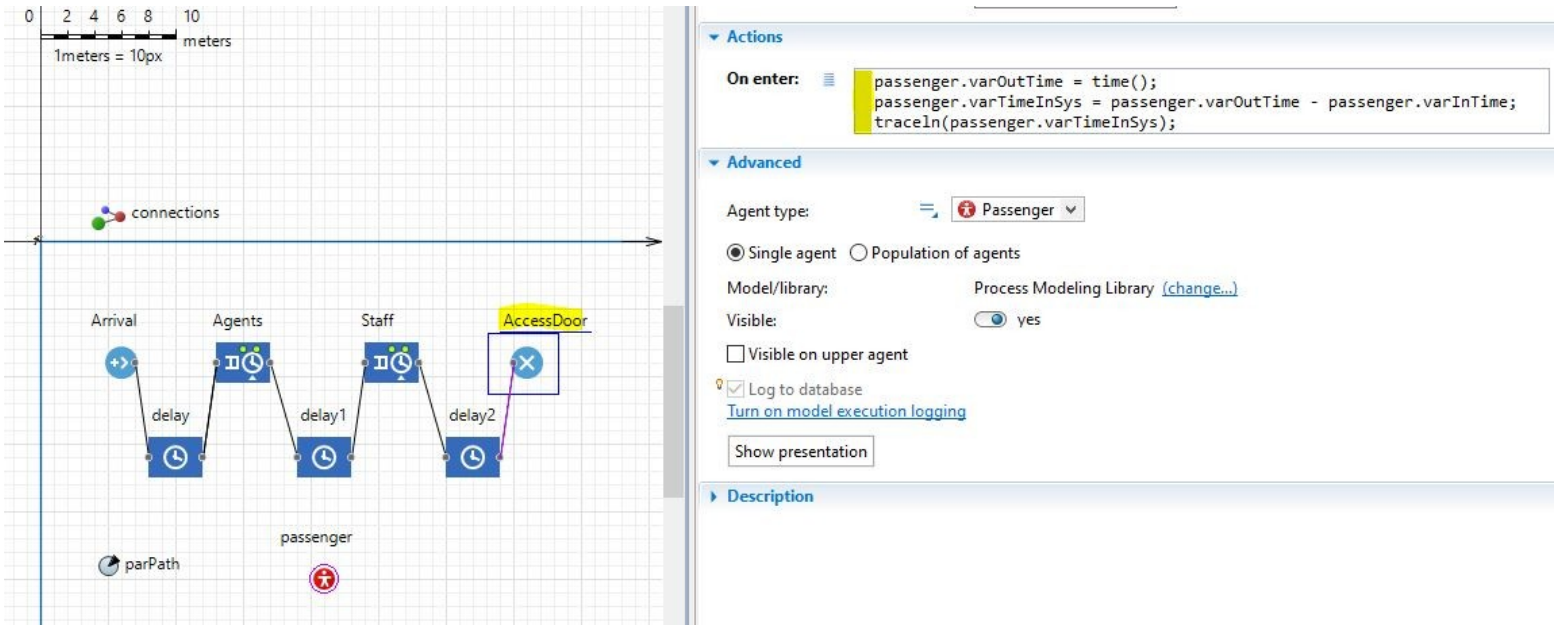
Reorder parameters and variables according to what is needed, some can go in Main, others must go in the Agent in question.



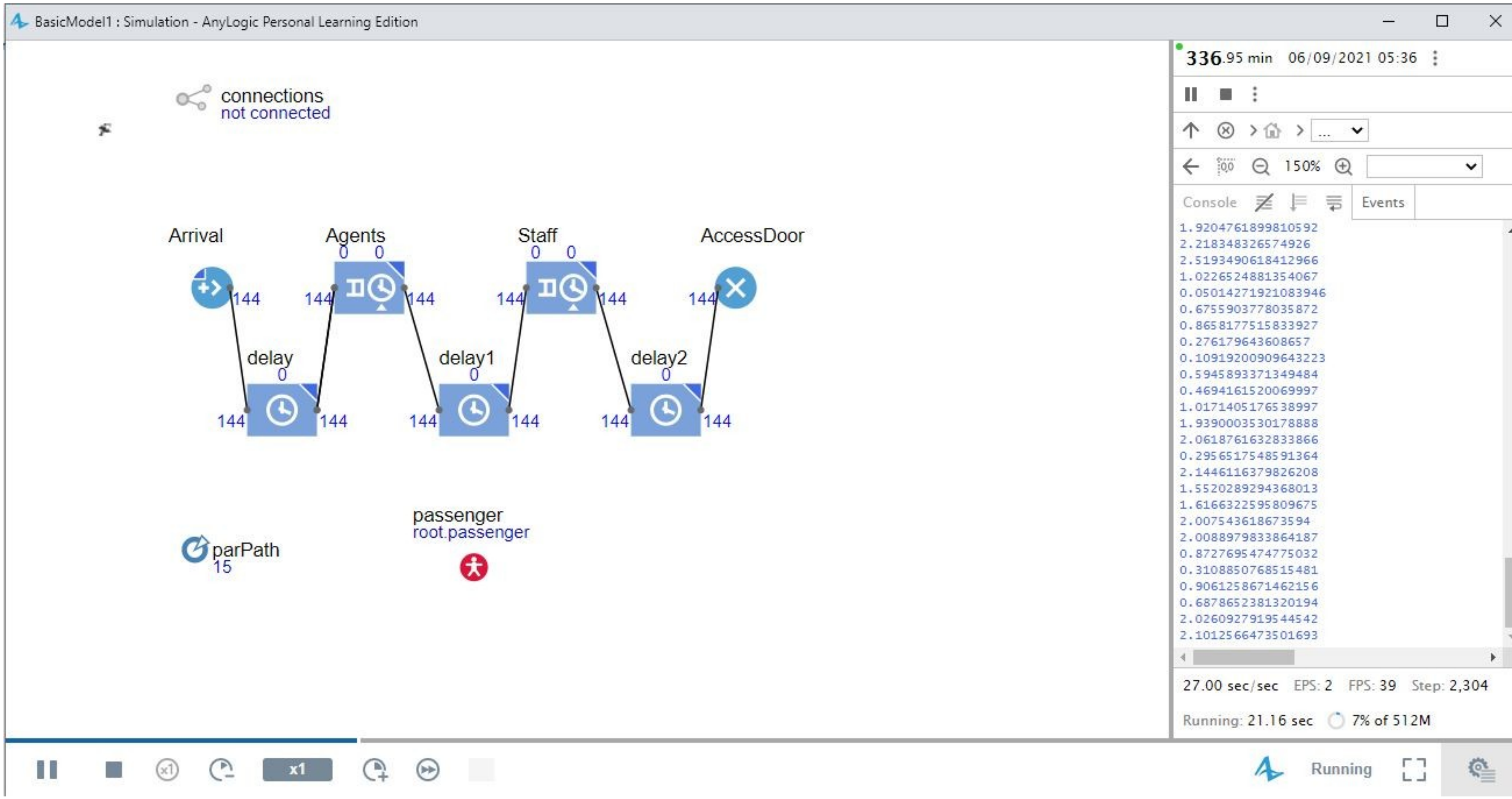
Notice that it's no longer as seen in slide number 5, this is not unusual, as when you program, you sometimes find better ways or improvements for the system performance along the way.

8. Show Results on console (1/2)

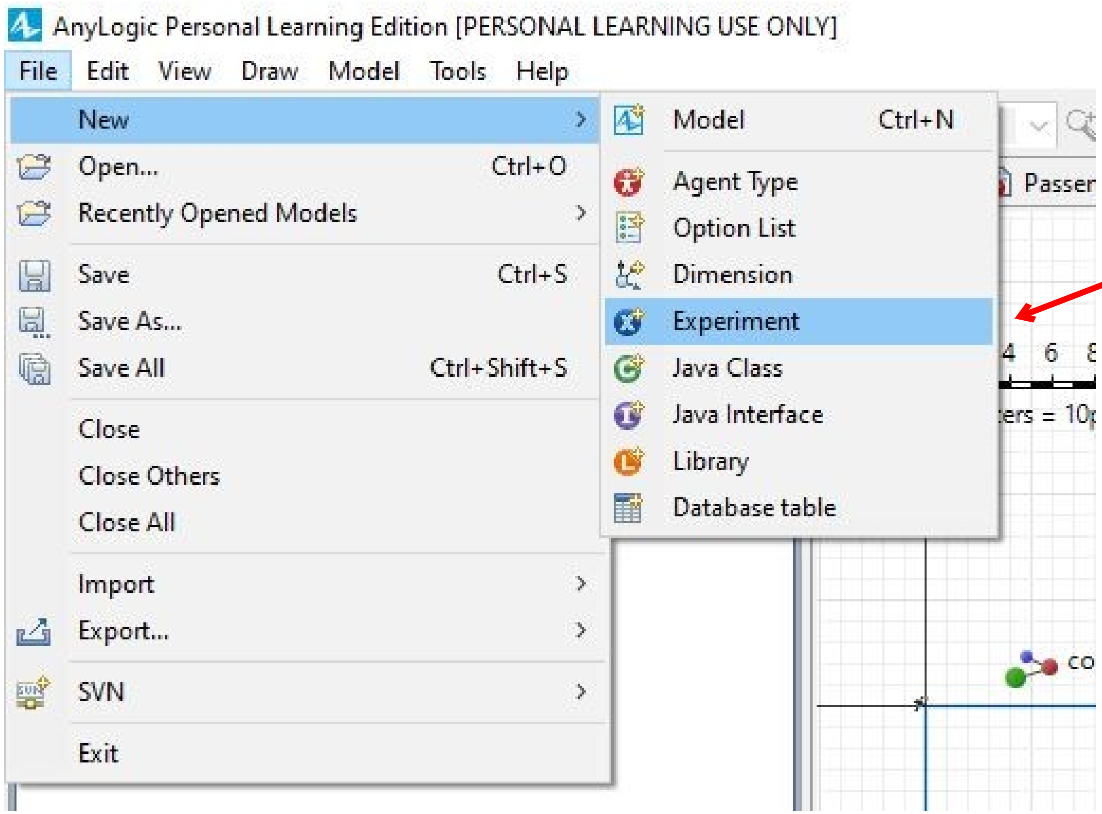




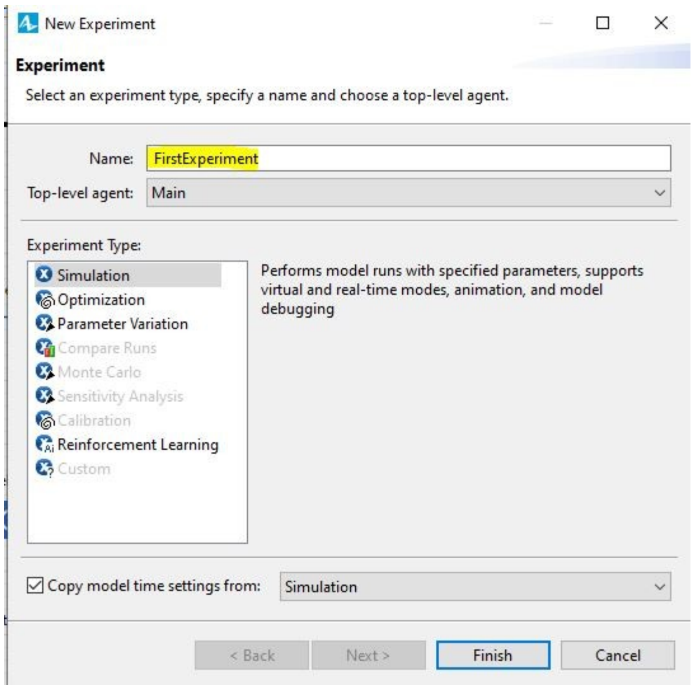
8. Show Results on console (2/2)



9. Add other simulations



Add an Experiment to your model workspace.

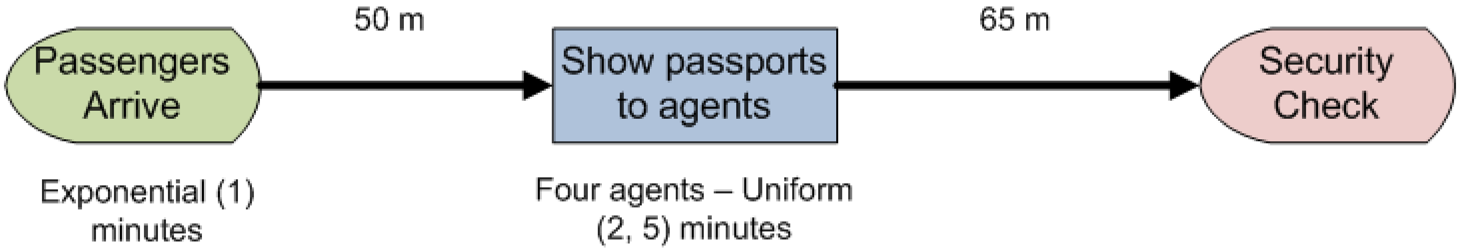


Adding a new experiment is basically adding a new mode of simulation for the program.
We can change things as randomness, repetition, speed, or others present in the properties when selecting the new simulation on *Projects*.

10. A simple check in point

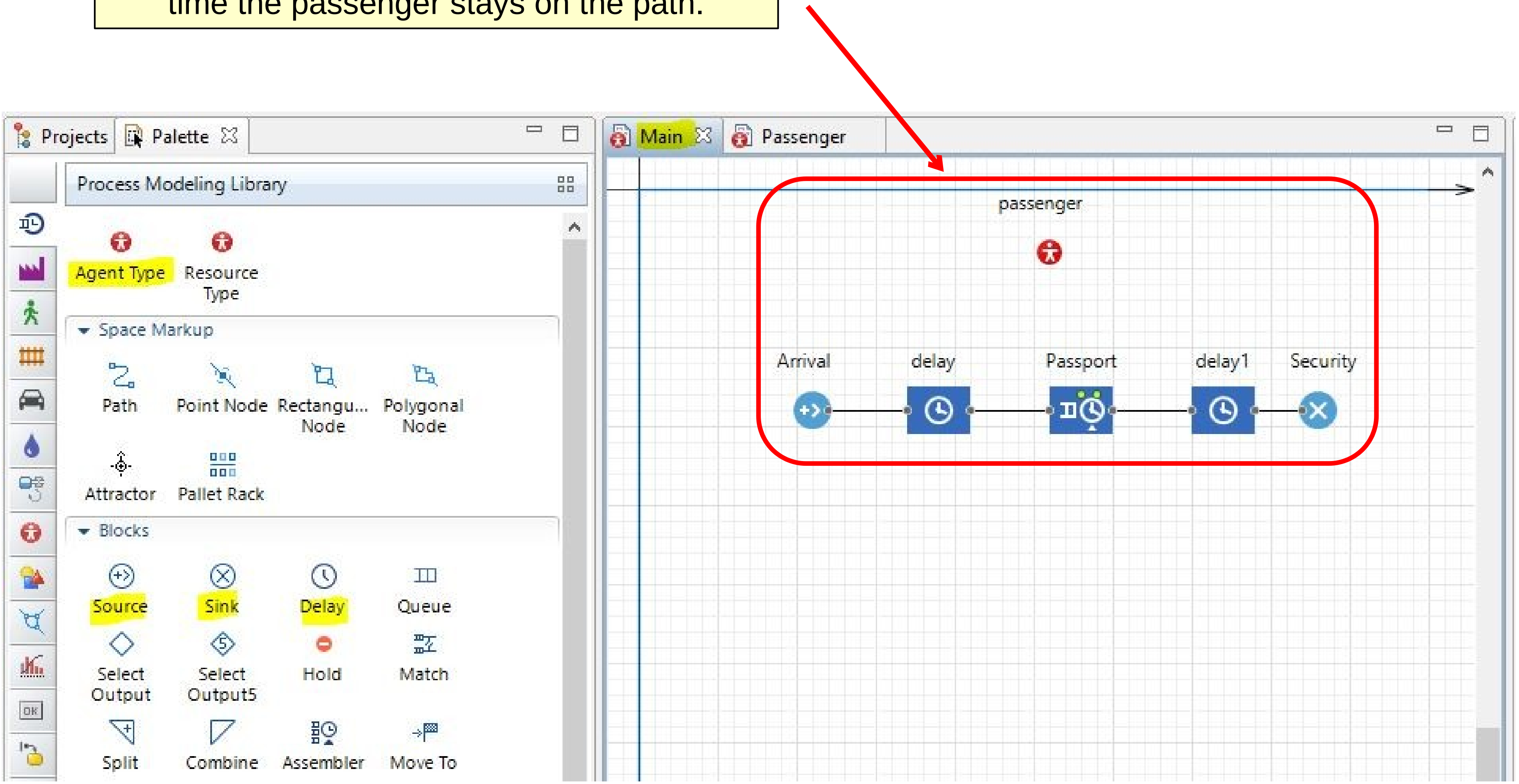
- Process overview: People arrive to an airport and go through the check in process to get their tickets. After check-in the passengers proceed to the security check point.

- Assumptions: (a) Passengers interarrival times follow an Exp(1) min.; (b) passengers walk rate follows a Uniform(2, 4) km/h; (c) passengers travel 50 m from the terminal entrance to the check in station; (d) following the check in process they must walk 65 m to the security check point; (e) the check-in station has 4 people assigned to process customers, who wait in a FIFO single line; (f) the check-in process follows a Uniform(2, 5) to completion; (g) the simulation model needs to be run for 24h.



11. Create the model

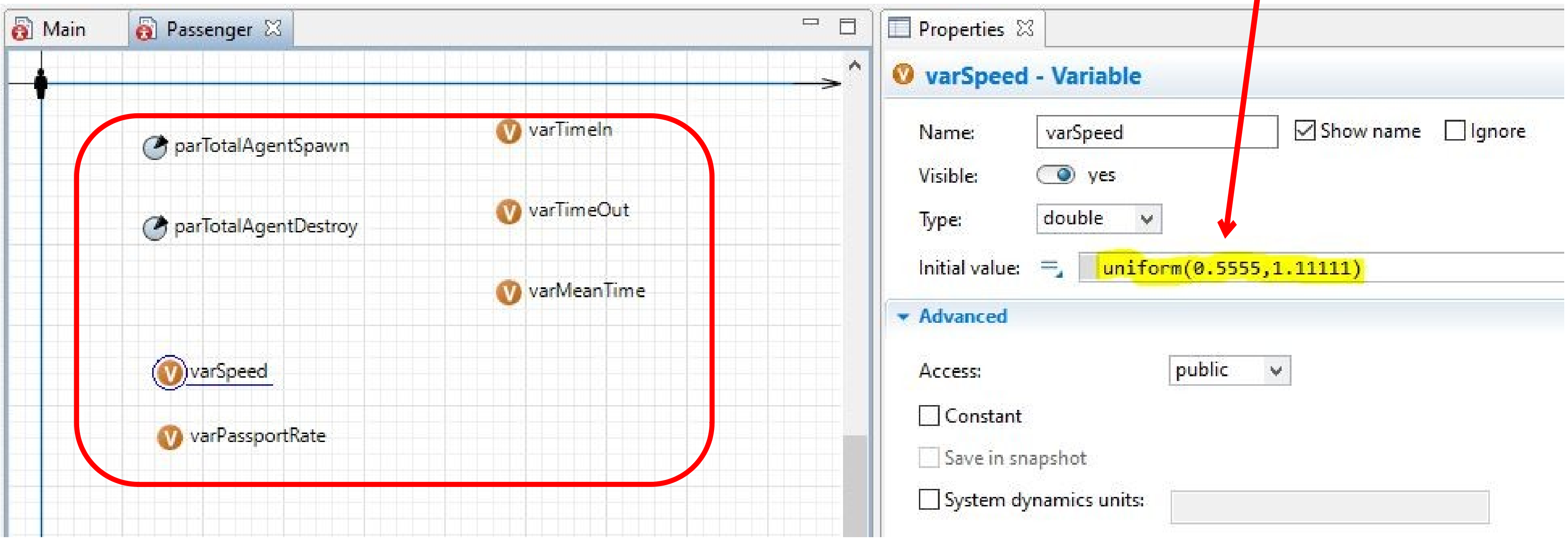
Follow the same steps as before, Drag & Drop objects into the model, connect them via a connector.
We will use Delay blocks to account for the time the passenger stays on the path.



12. Agent Passenger

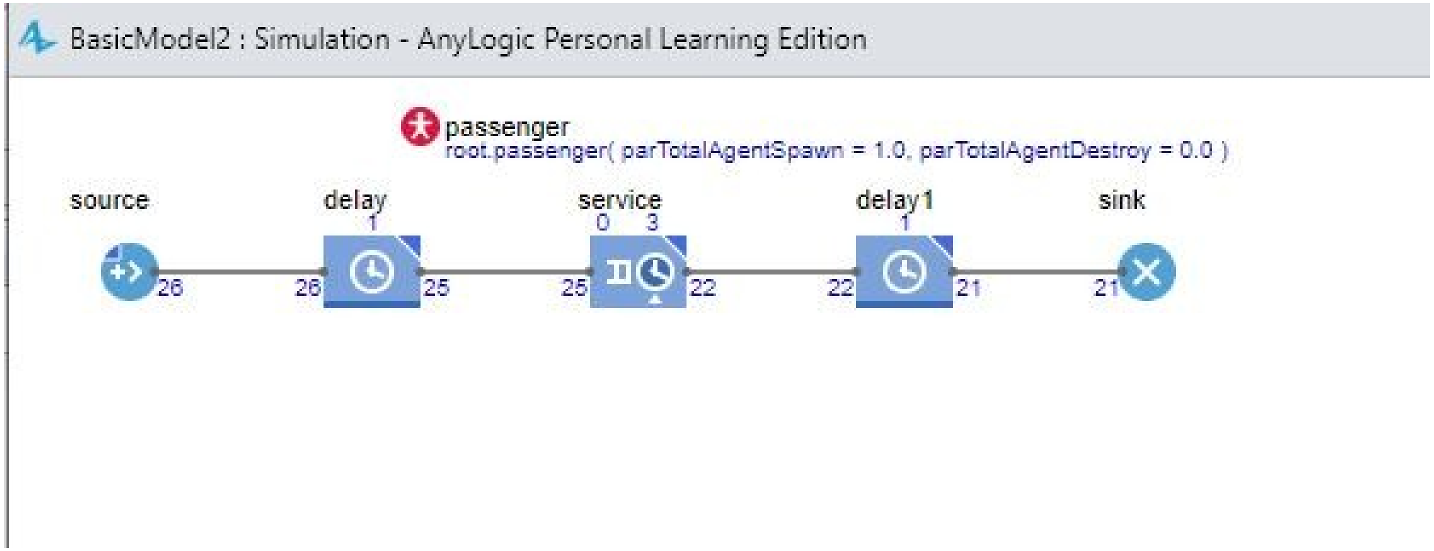
We create our Agent called Passenger, and we add variables and parameters that we need to show results.

Don't forget the metrics you are using! (This example shows the number for a simulation which each step equals 1 second)



13. First simulation

We can see how the simulation behaves, and we must check that it is not doing anything funny, like agents instantly skipping a block, travelling to fast, or not generating enough entities at the source.



With the simulation running we can set new parameters/variables to get data from the simulation on console.

14. Adding results

We must search for the Agent Actions on each block and use this spaces to order the program to produce results, it will be very helpful if you look first the example model, so you can see how this coding part looks like.