Universitat Autònoma
de Barcelona

Heterogeneous Models

AnyLogic

Author: Alejandro RodrIguez Grau

Tutor: Prof. Dr. Angel A. Juan

Full Course Index

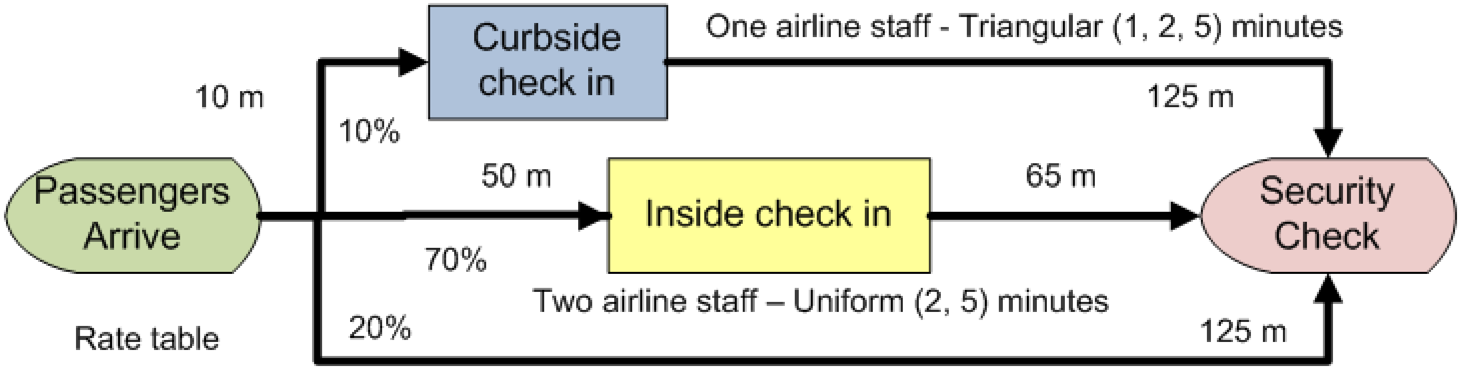
- Introduction to AnyLogic
- Basic models
- Heterogeneous model
- Advanced model

Heterogeneous Models

- **Exercise 1**
  - A Realistic Check-in Point
    - ❑ Basic Main Layout
    - ❑ Input Values
      - ❑ Source & SelectOutput (Individual Object)
      - ❑ Speed of Entities
      - ❑ Delays
      - ❑ Resource Pool (Worker Object)
    - ❑ Adding Result variables
      - ❑ Count Entities
      - ❑ Mean Time in Curbside
      - ❑ Mean Time in Checkpoint
      - ❑ Simulation

1. A realistic check in point

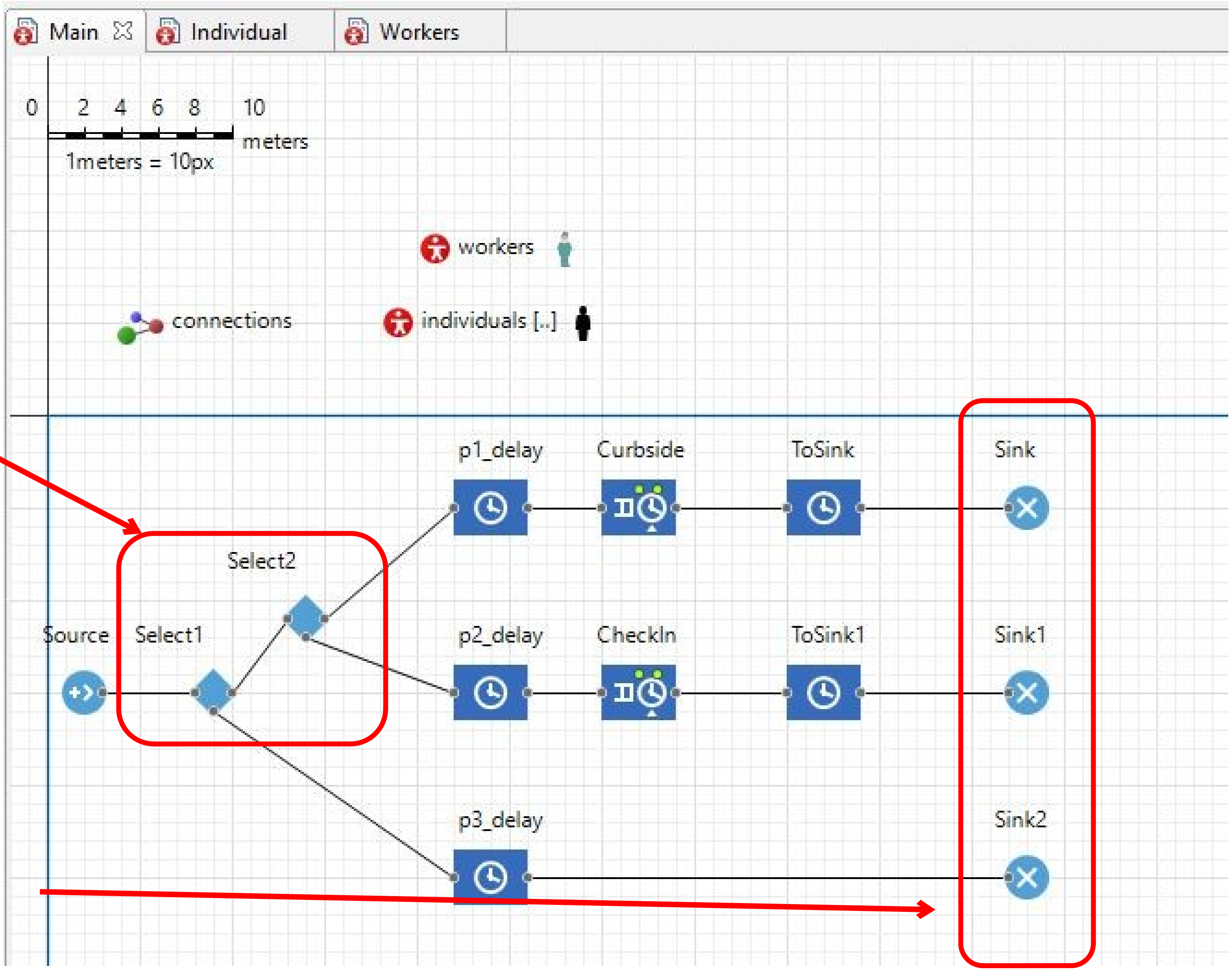
- Process overview: A realistic check-in model allows a choice of three different routes: (a) passengers who need to check bags and get tickets can either check in at the curbside station (10%), or (b) check in at the main station inside the airport (70%); and (c) those who have checked in online (20%) can proceed directly to the security check point.
- Assumptions: (a) Processing times at curbside check-in (only one airline staff) follow a Triangular(1,2,5) minutes; (b) processing times at main station (two airline staff) follow a Uniform(3,10) minutes; and (c) arrival rate is dynamic throughout the day (data table).



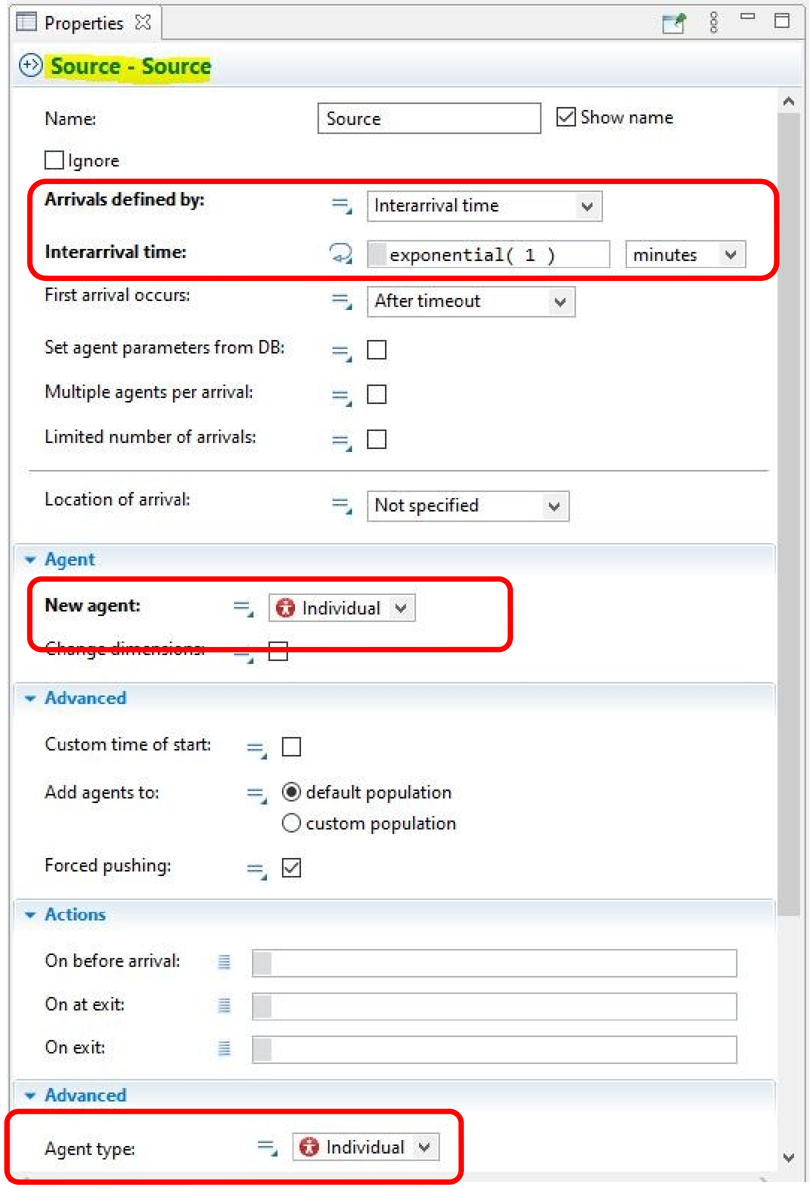
Basic Main Layout

This example has an important feature, there are different paths for the entity to go, and each one has a percentage that tells us how often passengers use each path.
To solve this we will use a ***Select Output*** twice (there is a ***Select Output*** with 5 ways to go, but for only 3, we must use two times the individual one)**.**

Dividing the sinks will allow us to see how many entities have gone through each path in the simulation, but we can use just one sink if preferred.



Input values (Source & SelectOutput)



Properties

Source - Source

Name: Source ☑ Show name

□ Ignore

Arrivals defined by: Interarrival time

Interarrival time: exponential(1) minutes

First arrival occurs: After timeout

Set agent parameters from DB: □

Multiple agents per arrival: □

Limited number of arrivals: □

Location of arrival: Not specified

Agent

New agent: Individual

Change dimensions: □

Advanced

Custom time of start: □

Add agents to: ● default population ○ custom population

Forced pushing: ☑

Actions

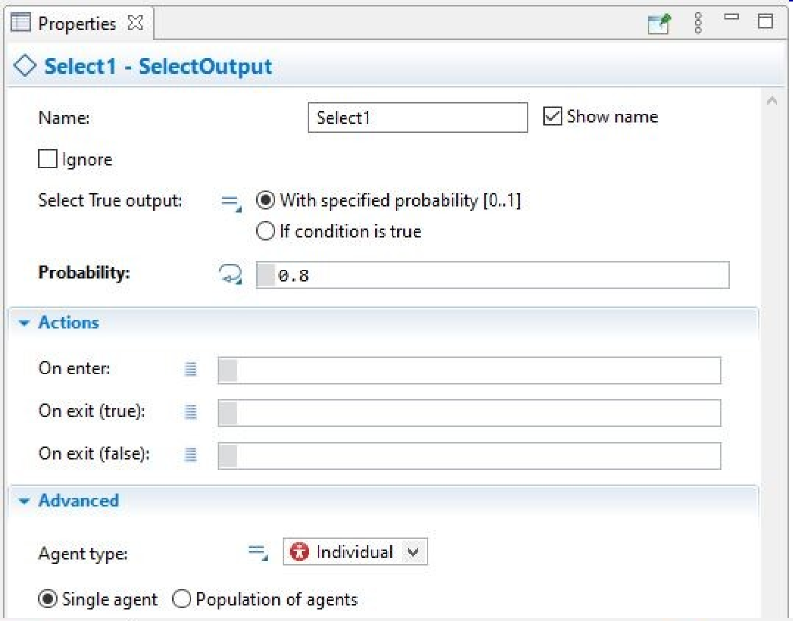
On before arrival:

On at exit:

On exit:

Advanced

Agent type: Individual



Properties

Select1 - SelectOutput

Name: Select1 ☑ Show name

□ Ignore

Select True output: ● With specified probability [0..1] ○ If condition is true

Probability: 0.8

Actions

On enter:

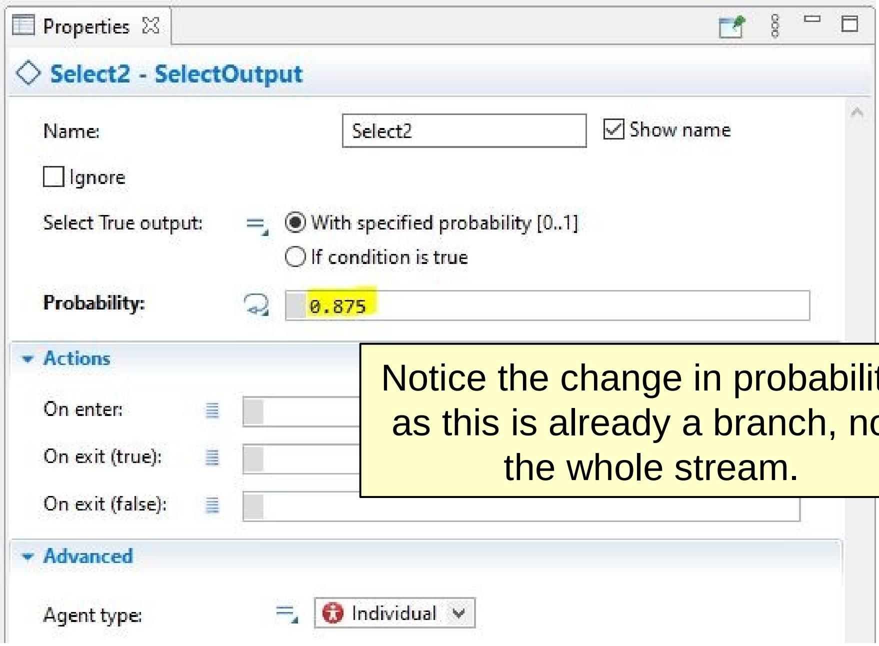
On exit (true):

On exit (false):

Advanced

Agent type: Individual

● Single agent ○ Population of agents



Properties

Select2 - SelectOutput

Name: Select2 ☑ Show name

□ Ignore

Select True output: ● With specified probability [0..1] ○ If condition is true

Probability: 0.875

Actions

On enter:

On exit (true):

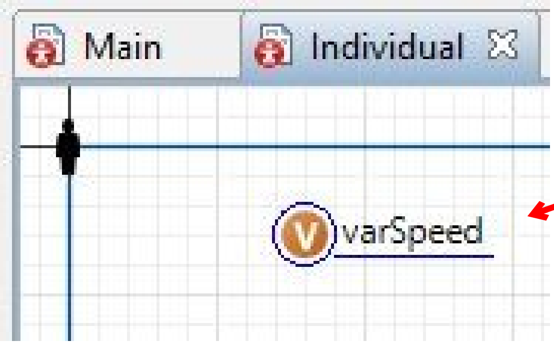
On exit (false):

Advanced

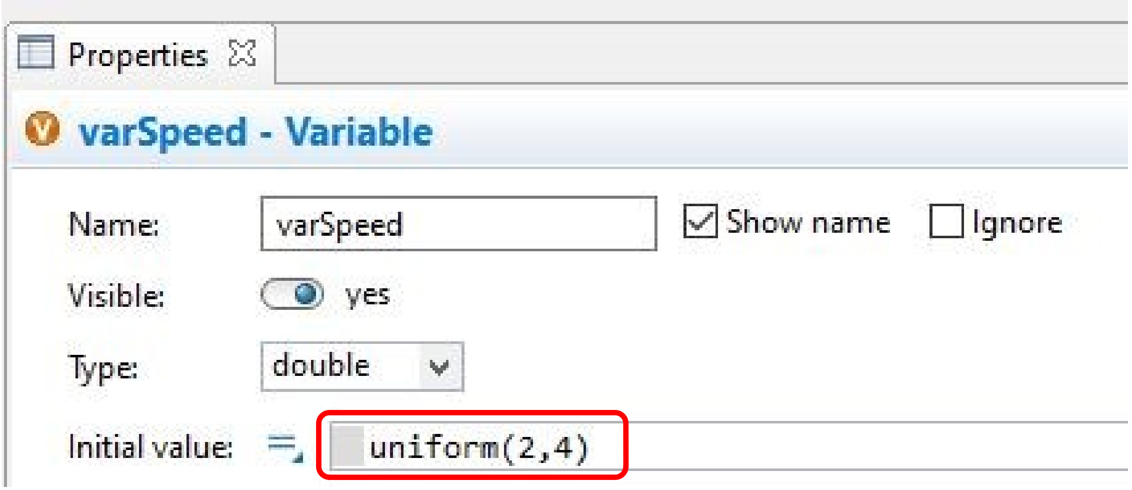
Agent type: Individual

Notice the change in probability, as this is already a branch, not the whole stream.

Input Values (Speed for paths)



This variable is needed to adjust the speed of the entities, the variables for results will come later.


Properties

varSpeed - Variable

Name: varSpeed ☑ Show name □ Ignore

Visible: yes

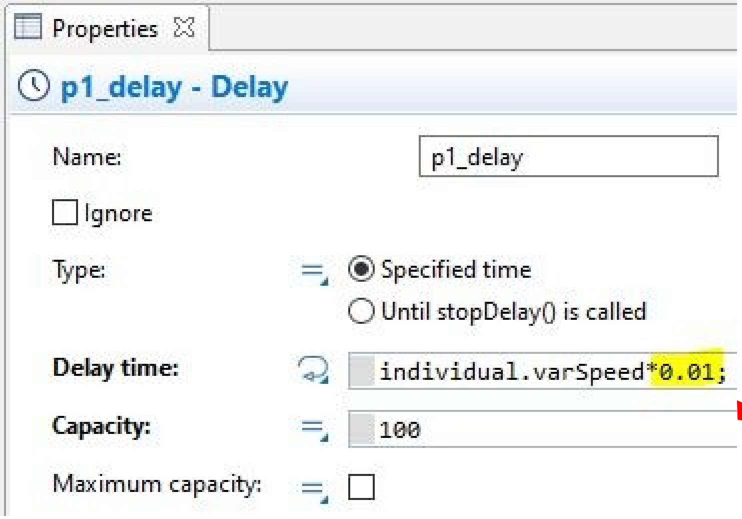
Type: double

Initial value: uniform(2,4)

Initial Value = uniform(2,4)

[km/h]

Input Values (Delays 1/2)



Properties

p1_delay - Delay

Name: p1_delay

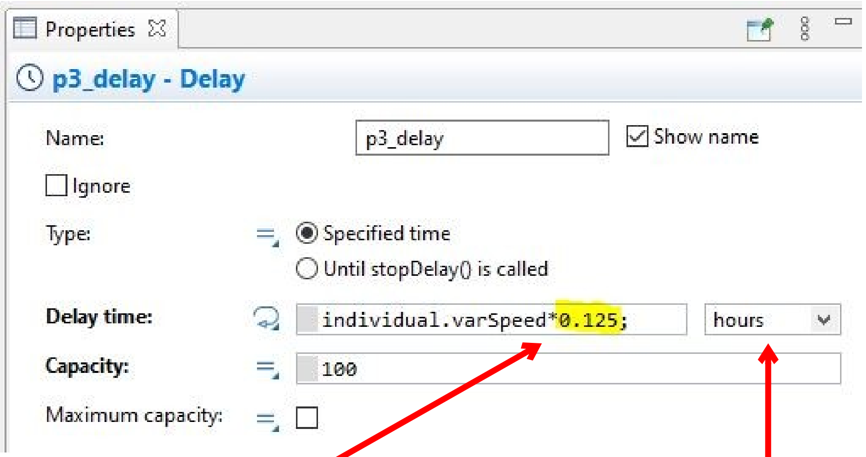
□ Ignore

Type: ● Specified time ○ Until stopDelay() is called

Delay time: `individual.varSpeed*0.01;`

Capacity: 100

Maximum capacity: □



Properties

p3_delay - Delay

Name: p3_delay ☑ Show name

□ Ignore

Type: ● Specified time ○ Until stopDelay() is called

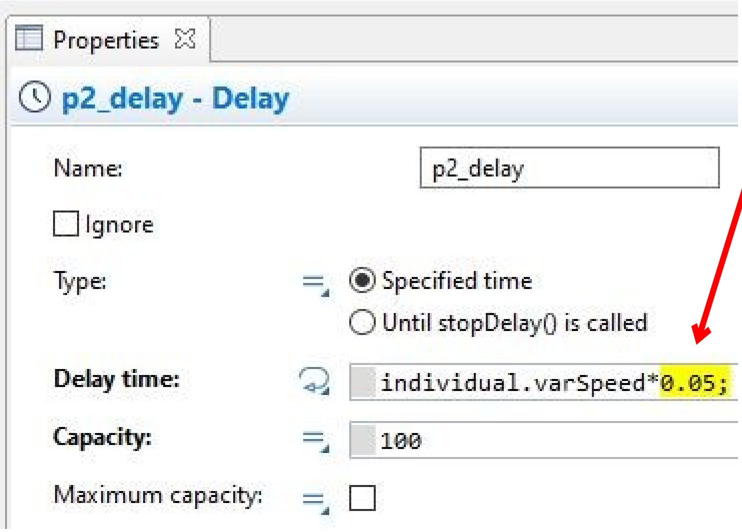
Delay time: `individual.varSpeed*0.125;` hours

Capacity: 100

Maximum capacity: □

Distance (km)

UNITS
Remember the units you are using!
(Speed comes in km/h)



Properties

p2_delay - Delay

Name: p2_delay

□ Ignore

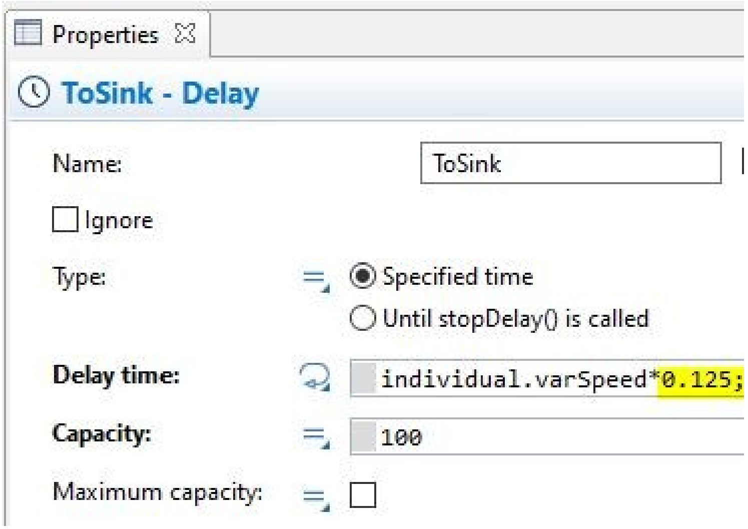
Type: ● Specified time ○ Until stopDelay() is called

Delay time: `individual.varSpeed*0.05;`

Capacity: 100

Maximum capacity: □

Input Values (Delays 2/2)



ToSink - Delay

Name: ToSink

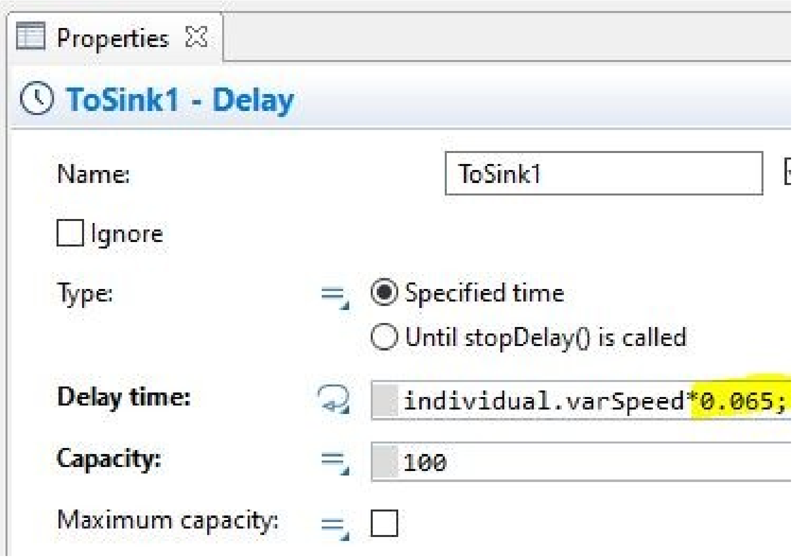
□ Ignore

Type: ● Specified time ○ Until stopDelay() is called

Delay time: `individual.varSpeed*0.125;`

Capacity: 100

Maximum capacity: □



ToSink1 - Delay

Name: ToSink1

□ Ignore

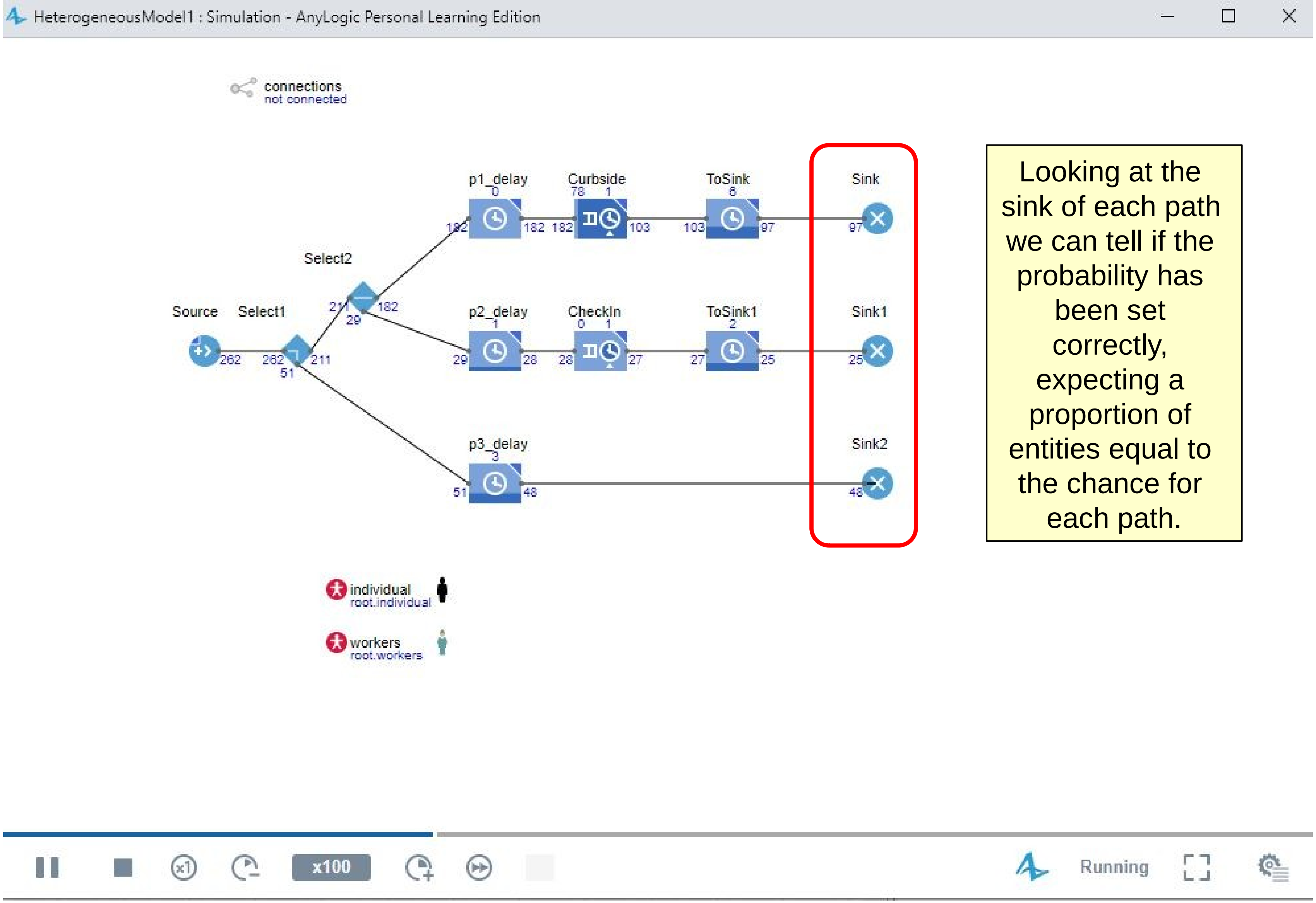
Type: ● Specified time ○ Until stopDelay() is called

Delay time: `individual.varSpeed*0.065;`

Capacity: 100

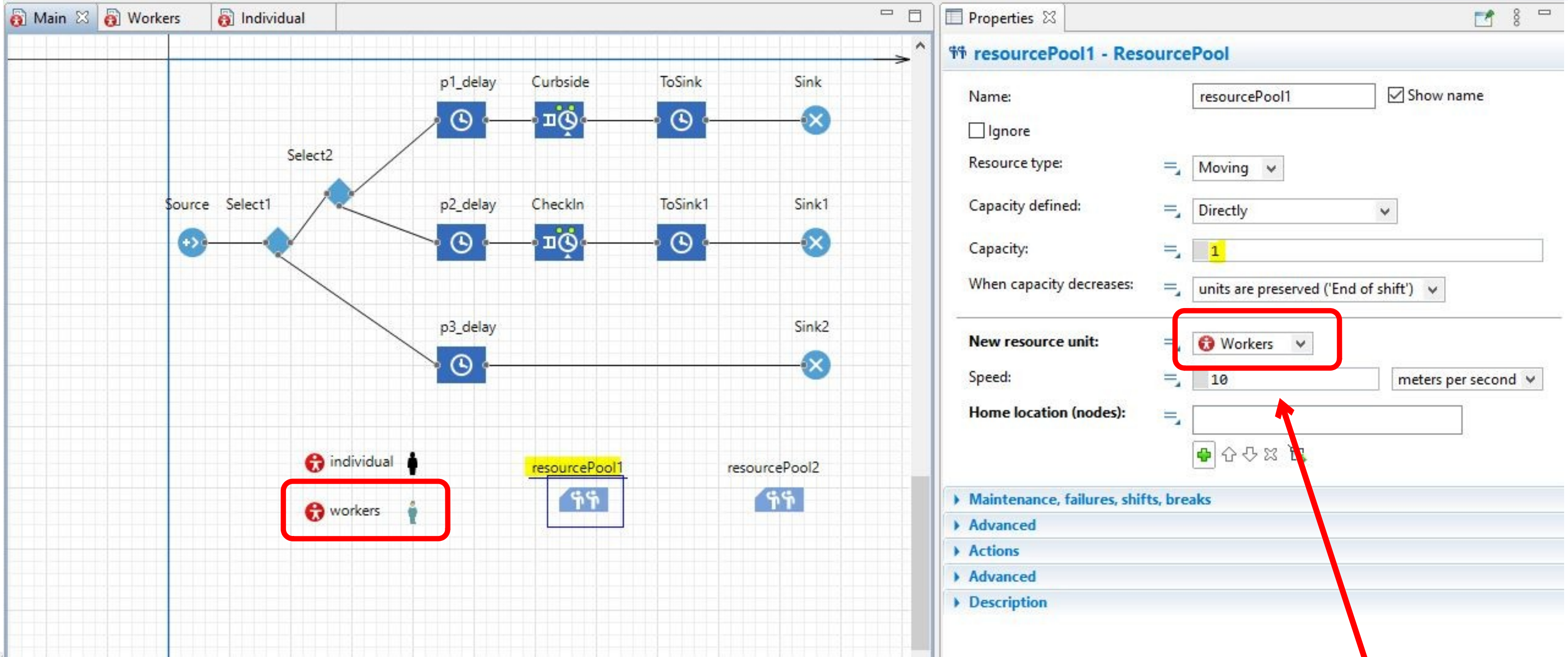
Maximum capacity: □

First Simulation



Looking at the sink of each path we can tell if the probability has been set correctly, expecting a proportion of entities equal to the chance for each path.

Resource Pools (Workers)



Properties

resourcePool1 - ResourcePool

Name: resourcePool1 ☑ Show name

☐ Ignore

Resource type: Moving

Capacity defined: Directly

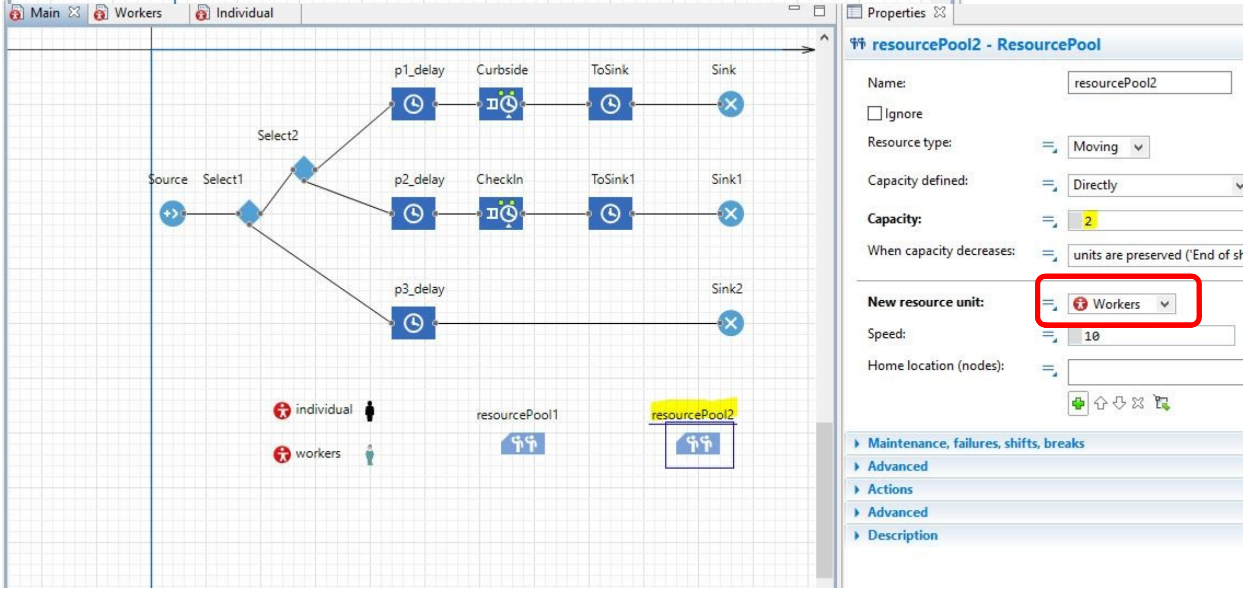
Capacity: 1

When capacity decreases: units are preserved ('End of shift')

New resource unit: Workers

Speed: 10 meters per second

Home location (nodes):



Properties

resourcePool2 - ResourcePool

Name: resourcePool2

☐ Ignore

Resource type: Moving

Capacity defined: Directly

Capacity: 2

When capacity decreases: units are preserved ('End of sh

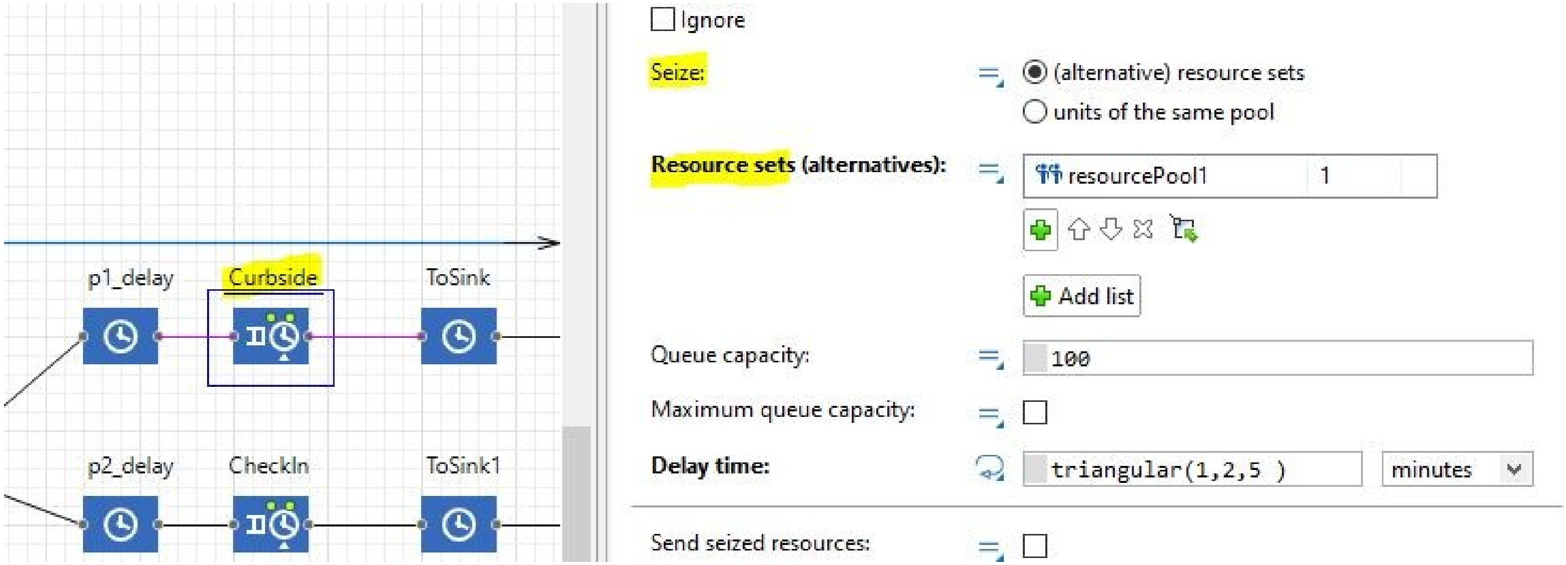
New resource unit: Workers

Speed: 10

Home location (nodes):

Up to this point, the "**Worker" object tab** is still empty, but further work can be done giving the resources (Workers) more properties.

Link services with ResourcePool



□ Ignore

Seize: ● (alternative) resource sets
○ units of the same pool

Resource sets (alternatives): resourcePool1 1

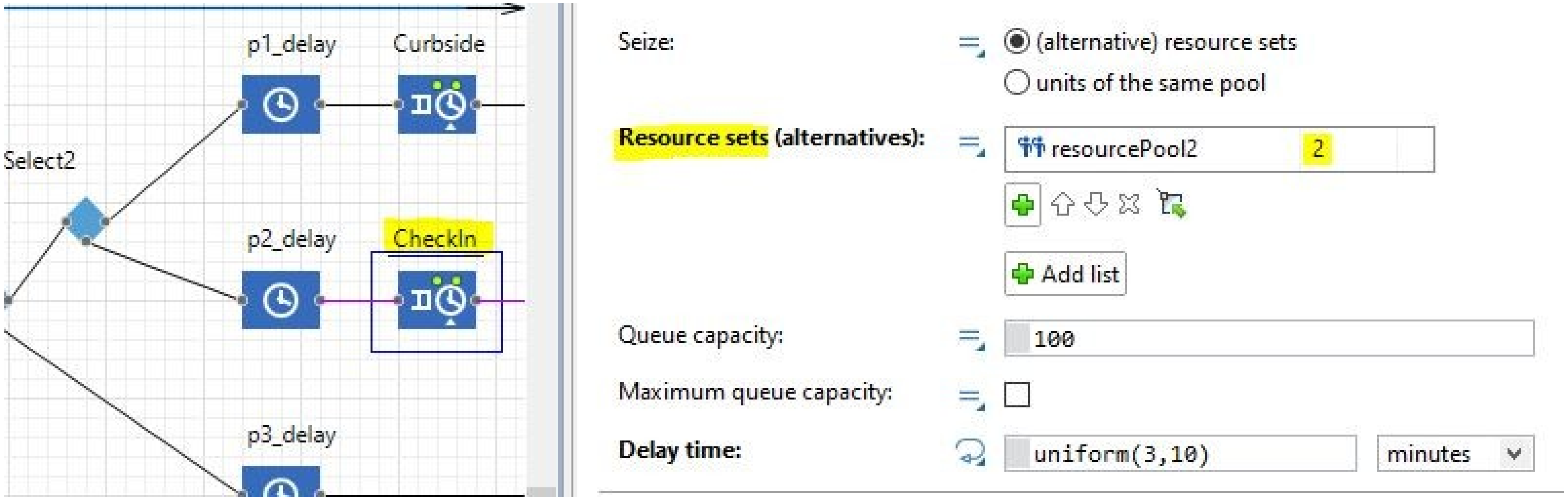
Add list

Queue capacity: 100

Maximum queue capacity: □

Delay time: triangular(1,2,5) minutes

Send seized resources: □



Seize: ● (alternative) resource sets
○ units of the same pool

Resource sets (alternatives): resourcePool2 2

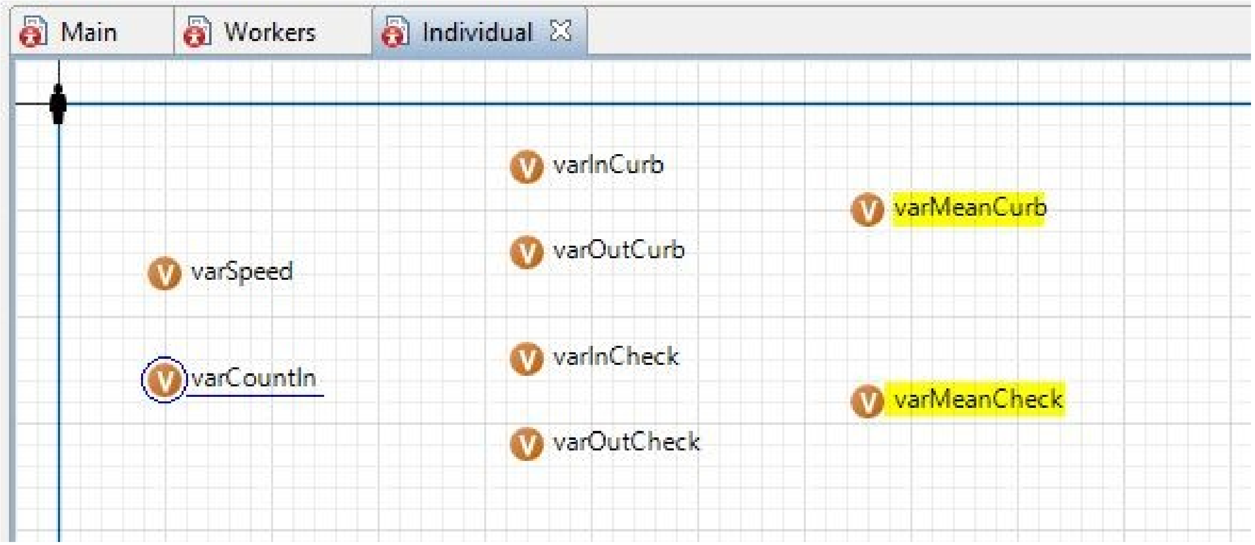
Add list

Queue capacity: 100

Maximum queue capacity: □

Delay time: uniform(3,10) minutes

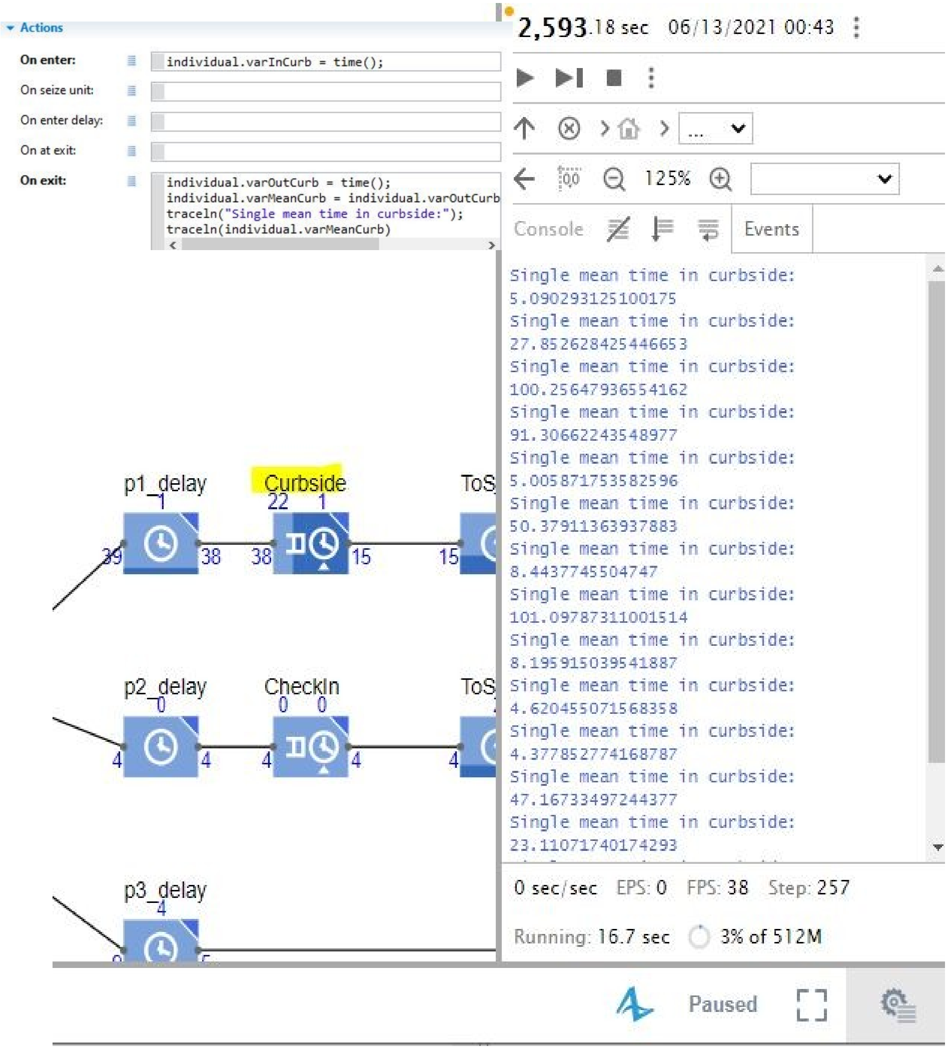
Variables for Results(1/2)



Define:
varCounIn = Total entities in system
varInCurb/check = Time in & time out of the services
VarMeanCurb/check = Time spent by an entity in a service

Variables for Results (2/2)

Mean Time in Curbside



Actions

On enter:
```
individual.varInCurb = time();
```

On seize unit:

On enter delay:

On at exit:

On exit:
```
individual.varOutCurb = time();
individual.varMeanCurb = individual.varOutCurb
traceln("Single mean time in curbside:");
traceln(individual.varMeanCurb)
```

2,593.18 sec 06/13/2021 00:43

125%

Console Events

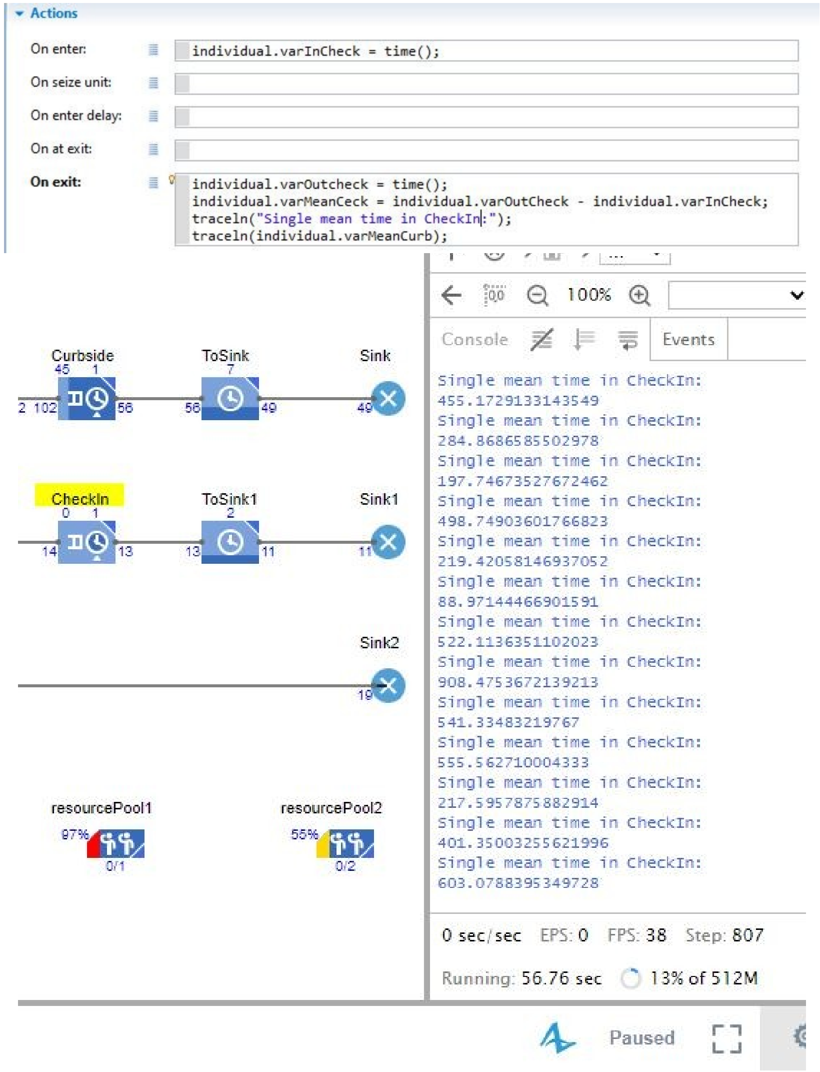
```
Single mean time in curbside:
5.090293125100175
Single mean time in curbside:
27.852628425446653
Single mean time in curbside:
100.25647936554162
Single mean time in curbside:
91.30662243548977
Single mean time in curbside:
5.005871753582596
Single mean time in curbside:
50.37911363937883
Single mean time in curbside:
8.4437745504747
Single mean time in curbside:
101.09787311001514
Single mean time in curbside:
8.195915039541887
Single mean time in curbside:
4.620455071568358
Single mean time in curbside:
4.377852774168787
Single mean time in curbside:
47.16733497244377
Single mean time in curbside:
23.11071740174293
```

0 sec/sec EPS: 0 FPS: 38 Step: 257

Running: 16.7 sec 3% of 512M

Paused

Mean time in CheckIn



Actions

On enter:
```
individual.varInCheck = time();
```

On seize unit:

On enter delay:

On at exit:

On exit:
```
individual.varOutcheck = time();
individual.varMeanCeck = individual.varOutCheck - individual.varInCheck;
traceln("Single mean time in CheckIn:");
traceln(individual.varMeanCurb);
```

100%

Console Events

```
Single mean time in CheckIn:
455.1729133143549
Single mean time in CheckIn:
284.8686585502978
Single mean time in CheckIn:
197.74673527672462
Single mean time in CheckIn:
498.74903601766823
Single mean time in CheckIn:
219.42058146937052
Single mean time in CheckIn:
88.97144466901591
Single mean time in CheckIn:
522.1136351102023
Single mean time in CheckIn:
908.4753672139213
Single mean time in CheckIn:
541.33483219767
Single mean time in CheckIn:
555.562710004333
Single mean time in CheckIn:
217.5957875882914
Single mean time in CheckIn:
401.35003255621996
Single mean time in CheckIn:
603.0788395349728
```

0 sec/sec EPS: 0 FPS: 38 Step: 807

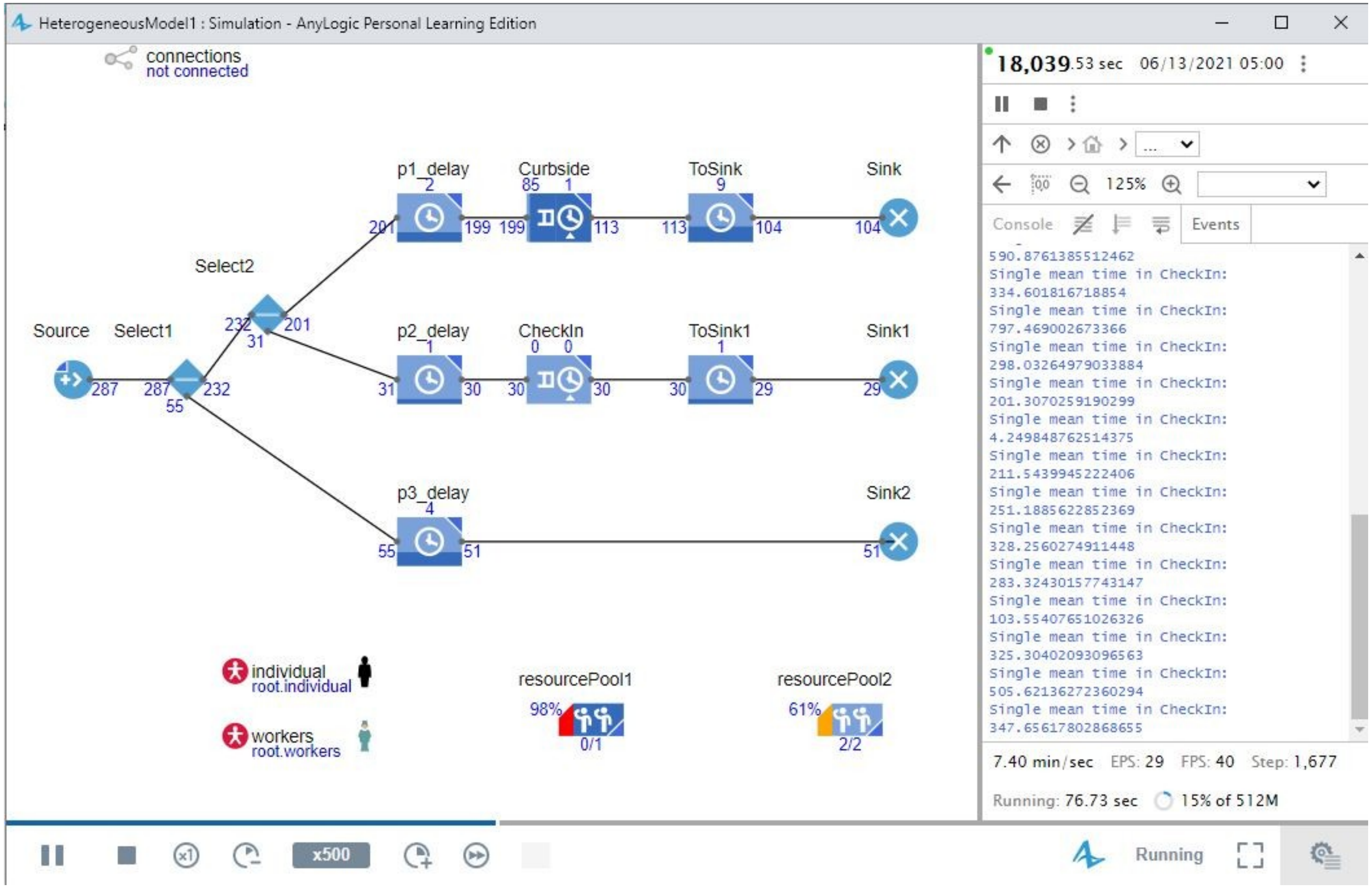
Running: 56.76 sec 13% of 512M

Paused

Simulation

Notice that in this case, both results of curbside and checkpoint are in seconds, as the simulation uses that measure (set in the first model screen).

This is the last simulation for this exercise:



HeterogeneousModel1 : Simulation - AnyLogic Personal Learning Edition

18,039.53 sec 06/13/2021 05:00

```
590.8761385512462
Single mean time in CheckIn:
334.601816718854
Single mean time in CheckIn:
797.469002673366
Single mean time in CheckIn:
298.03264979033884
Single mean time in CheckIn:
201.3070259190299
Single mean time in CheckIn:
4.249848762514375
Single mean time in CheckIn:
211.5439945222406
Single mean time in CheckIn:
251.1885622852369
Single mean time in CheckIn:
328.2560274911448
Single mean time in CheckIn:
283.3243015774314 7
Single mean time in CheckIn:
103.55407651026326
Single mean time in CheckIn:
325.30402093096563
Single mean time in CheckIn:
505.6213627236029 4
Single mean time in CheckIn:
347.6561780286865 5
```

7.40 min/sec EPS: 29 FPS: 40 Step: 1,677

Running: 76.73 sec 15% of 512M