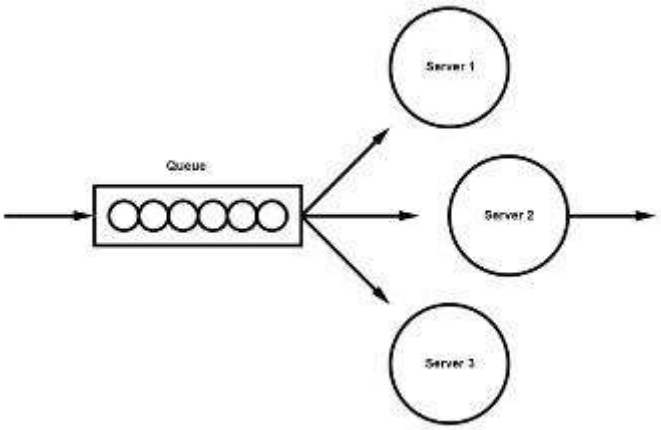
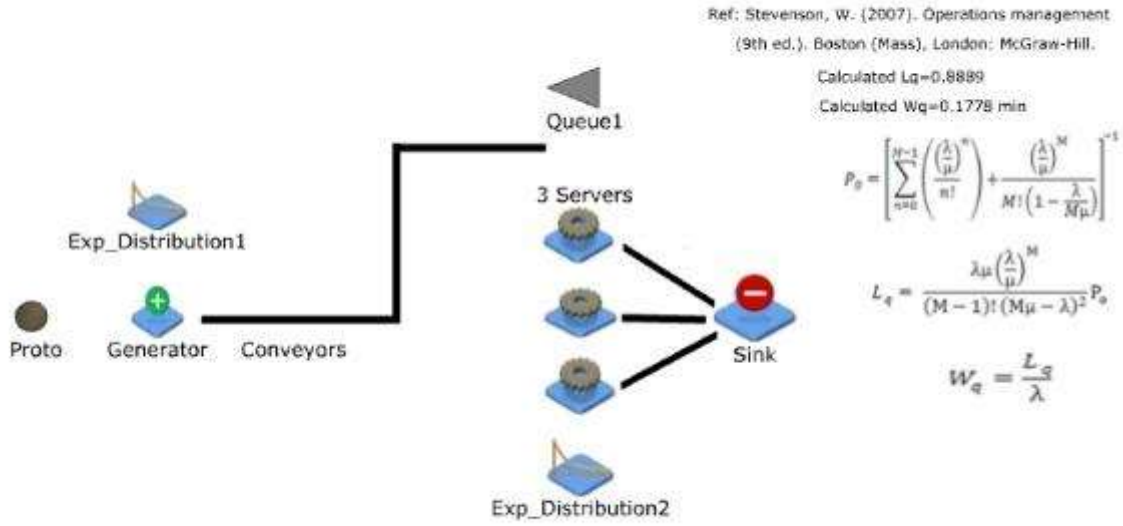


Simulation of Queuing Systems Part 1

Using JAAMSIM

Mike Halls

MM3 Ave InterArrival time 0.2min & Service time 0.4min



2014-Jan-01 00:00:00.000

29.05.2016

Disclaimer

By making use of any information within this text, you agree to the following:

No warranties: All of the information provided in this text is provided "AS-IS" and with No warranties. No express or implied warranties of any type, including for example implied warranties of fitness for a particular purpose, are made with respect to the information, or any use of the information, in this text. The author makes no representations and extends no warranties of any type as to the accuracy or completeness of any information or content of this text.

Disclaimer of liability: The author specifically disclaims liability for incidental or consequential damages and assumes no responsibility or liability for any loss or damage suffered by any person as a result of the use or misuse of any of the information or content of this text. The author assumes or undertakes no liability for any loss or damage suffered as a result of the use, misuse or reliance on the information and content of this text.

Use at your own risk: This text is for informational purposes only. It is your responsibility to check and validate any information you decide to use from this text.

Check for updates at:-

http://jaamsim.com/

The software is updated almost on a weekly basis.

Table of Contents

1.0 Introduction 7
2.0 Useful videosand YouTube 7
3.0 Queue Classification 8
4.0 The JaamSimTutorial 9
5.0 M/D/1 Constant Service Time Model 9
5.1.1 Assumptions 10
5.1.2 Flow Chart 1 Representation of the M/D/1 model 10
5.1.3 M/D/1/∞ Model of a single server queue with constant service times 11
5.1.4 M/D/1/∞ Results of the single server queue with constant service time 12
5.2 M/D/1/∞ The finished model 13
6.0 M/M/1/∞ Single server queue 14
6.1.1 Assumptions 14
6.1.2 Flow chart 2 Representation of the M/M/1 model 15
6.1.3 M/M/1/∞ Results of the single server Queue 17
6.2 M/M/1/∞ The finished model 18
7.0 M/M/3/∞ Multi-server single queue 19
7.1.1 Assumptions 20
7.1.2 Flow chart 3 Representation of M/M/3 model 20
7.2 M/M/3/∞ The model 21
7.2.1 M/M/3/∞ Results of the multi-server Queue 22
7.2.2 The time spent in the system Ws 23
7.3 M/M/3/∞ The finished Model 24
8.0 M/M/1 An introduction to Flow lines in series 25
8.1 M/M/1 SeriesExample 26
8.1.1 Assumptions 26
8.2 M/M/1 SeriesMachine utilization ρ 27
8.2.1 Machine 1. 27
8.2.2 Machine 2. 27
8.2.3 Machine 3. 27
8.3 M/M/1 Series Average numberof customers in the queue Lq. 27
8.3.1 Machine 1 27
8.3.2 Machine 2 27

8.3.3 Machine 3 28
8.4 M/M/1 Series Length of time spent in the queue Wq 28
8.4.1 Machine 1 28
8.4.2 Machine 2 28
8.4.3 Machine 3 28
8.5 M/M/1 SeriesBuild a model using JaamSim to simulate this Flow Line. 29
8.6 M/M/1 Resultsof theFlow line in series 30
8.7 M/M/1 + M/M/1 + M/M/1 flow line model in series 32
9.0 M/M/1 + M/M/1 +M/M/3.An Introduction to Flow lines 33
9.1 M/M/1 + M/M/1 + M/M/3Flowline Model 34
9.2 M/M/1 + M/M/1 + M/M/3 Results of the flow line model 35
9.3 M/M/1 + M/M/1 + M/M/3 The finished flow line model 37
10.0 M/M/1/K A single server Queue with finite length 38
10.1 M/M/1/K The modelof a single server queue with finite length 39
10.1.1 ExpressionThreshold 40
10.1.2 The Entity Generator 40
10.2 M/M/1/K Calculations - A single server Queue with finite length 41
10.2.1 Assumptions 41
10.2.2 Formulae 41
10.3 M/M/1/K Resultsof a single server queue with finite length 42
10.3.1 M/M/1/K Calculated results 42
10.3.2 M/M/1/K Simulated results 42
10.4 M/M/1/K The finishedmodel of a single server queue with finite length 43
11.0 M/M/1/K A finite queue with baulking 44
11.1.1 The Branch 45
11.2 M/M/1/K The finished model of a finite queue with baulking 46
12.0 M/M/2/3 Finite Queue 47
12.1 M/M/2/3 The model 48
12.2 M/M/2/3 Calculations. A single server Queue with finite length 48
12.2.1 Assumptions 48
12.2.2 M/M/2/3 Formulae 49
12.3 M/M/2/3 Results. Finite Queue 50
12.3.1 M/M/2/3 Calculated results 50

12.3.2 M/M/2/3 Simulated results 50
12.4 M/M/2/3 The finished model. Finite Queue 51
13.0 Reneging 52
13.1 Renege and Baulk Explanation. 56
13.1.1 Run Duration 56
13.1.2 Generator (Gen) 56
13.1.3 Exponential Distribution1 56
13.1.4 Branch1 56
13.1.5 Queue1 and Queue2 56
13.1.6 CustomOutputList 57
13.1.7 Renege 59
13.1.8 Queue1 and Queue2 59
13.1.9 Branch2 60
13.1.10 Fraction Added - How does it work? 60
13.1.11 ExpressionEntity 61
13.1.12 Tracing the values of indexOfMin 62
13.1.13 Conveyors 63
13.2 Reflections of the Renege and Baulking Model 63
14.0 Batch Versus One Piece Flow 64
14.1 Introduction - One Piece Flow line Versus Batch Production 65
14.2 Creating the batch size of 9 Flow line 66
14.2.1 SimEntity1 68
14.2.2 Assign1 68
14.2.3 SetGraphics1 70
14.2.4 Container 70
14.2.5 Pack 70
14.2.6 Unpack 70
14.2.7 Creating the Batches of 3Flow line 70
14.3 Creating the One Piece Flow Line 70
15.0 Further Models 71
15.1 Simple Pull System 71
16.0 Changing the graphics 72
16.1 Change the entity colour based on the attribute value assigned by an assign module. By Harry King. 73

17.0 Example of selected outputs for multiple runs summarized in a single table using the RunOutputList keyword - By Harry King. http://jaamsim.com/ 74
17.1 RunOutputList Example by Harry King http://jaamsim.com/ 75
18.0 Hints and Tips 76
18.1 Hints and Tips 1 - Saving 76
18.2 Hints and Tips 2 - Backup Copies 76
18.3 Hints and Tips 3 - Simulation times 76
18.4 Hints and Tips 4 - Anti-Virus programmes 76
19.0 Equation nomenclature 77
20.0 Other useful open source OR free software 78
20.1 XMind 78
20.2 Geogebra 78
20.3 LibreCAD 78
20.4 Artweaver 78
20.5 Debut Video Capture software 78
20.6 VideoPad Video Editor 78
20.7 Handbrake 78
21.0 Bibliography 79

1.0 Introduction

I found JaamSim quite by accident. - http://jaamsim.com/

To get started I decided to look at queuing models with known mathematical theory, which can be tested against simulated models created with JaamSim. This allows a degree of confidence in the models constructed and knowledge that they are valid.

This booklet starts very simply with Single Server queues, a Multiple Server queue M/M/3 queue, simple Flow lines and Finite queues as an introduction to Queuing theory and Simulation.

Acknowledgements

Harry King, Harvey Harrison and the JaamSim team.

Thank you for an excellent program and all your past and continued help to enable me to get started as a user.

http://jaamsim.com/

2.0 Useful videos and YouTube

Most of the models in this document can be viewed on YouTube:-

www.youtube.com/channel/UCiYvthn9V1BzVTtXF8VSpmg

The JaamSim YouTube channel has some of these and more examples:

www.youtube.com/c/jaamsim

3.0 Queue Classification

The simplest form of Kendall's notation for the classification of queues:-

The notation consists of letters and numbers.

A/B/C/D/E

Where:-

A = The distribution of the inter arrival times of customers

B = The distribution of the service times

C = Number of servers

D = Maximum total number of customers accommodated by the system

E = Calling population size

A and B could be any one of the following:

M – Exponential Distribution

D – Degenerate or deterministic distribution

E_k = Erlang distribution (k = shape parameter)

G = General distribution

Examples

M	**D**	**1**
Inter Arrival Times are exponential	Service Times are constant	Number of servers is 1
M	**M**	**1**
Inter Arrival Times are exponential	Service Times are exponential	Number of servers is 1
M	**M**	**3**
Inter Arrival Times are exponential	Service Times are exponential	Number of servers is 3

M	**M**	**s**	**K**	**N**
Inter Arrival Times are exponential	Service Times are exponential	Number of servers	Maximum number of customers in the queuing system	Number of potential customers in the calling population

There are plenty of explanations of Kendall's notation on the internet if you want to know more.

https://www.andrewferrier.com/oldpages/queueing_theory/Andy/kendall.html

http://nptel.ac.in/courses/117103017/5

4.0 The JaamSim Tutorial

Work through the tutorial in the JaamSim User Manual, and **read the Hints and Tips section of this document.**

The JaamSim tutorial is self-explanatory and will help you find your way around the user interface and enable you to build models quickly.

5.0 M/D/1 Constant Service Time Model

Where the conveyors times are not quoted in this tutorial enter travel times for the conveyors that allow you to see what is happening in the model.

This is essentially the same model you created from the JaamSim Manual.

Sometimes the services offered have a constant service time instead of exponentially distributed time. Any process that is automated will have a fixed cycle time for example a CNC programmed cycle of events, where the time to manufacture each component is the same.

Constant service times are fixed and the values for the average queue length Lq, average waiting time in the queue Wq and average number in the system Ls are always less than they would be if the service time was exponentially distributed.

Lq and Wq are half that of an equivalent M/M/1 model.

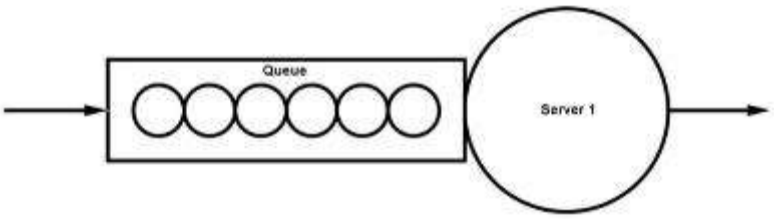


Figure 1. M/D/1/∞ Single server Queue with constant service times

Edit the model you created using the JaamSim Manual

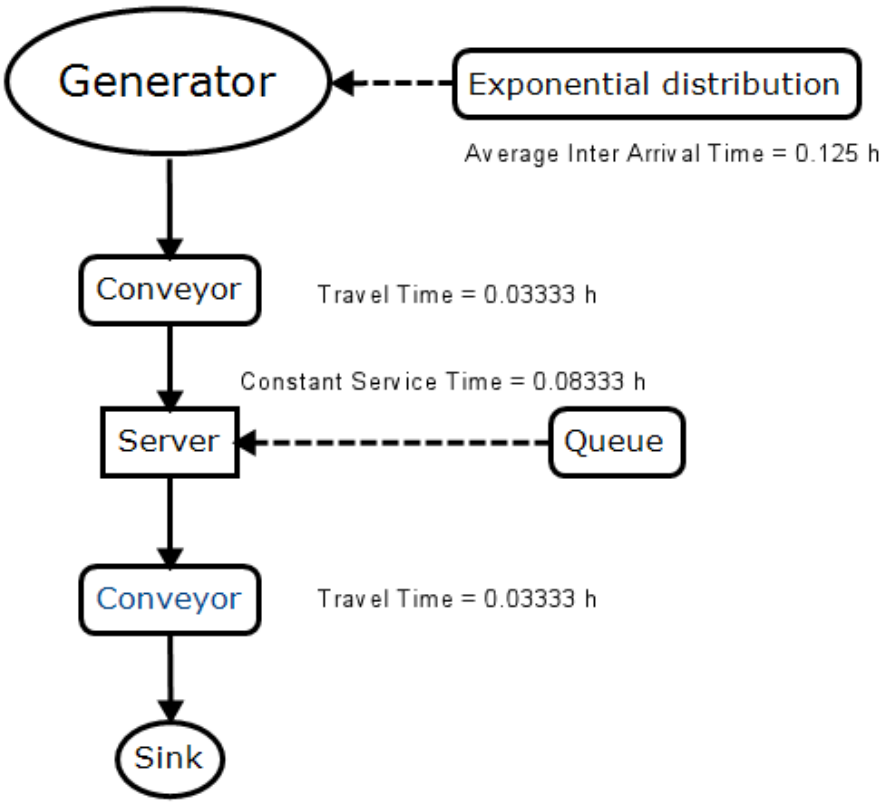
Entity	Inter arrival time Exponentially Distributed	Constant Service time	Travel time
Arrival Distribution Mean	0.125 hours		
Service Distribution Mean		0.08333 hours	
Conveyor			0.03333 hours

Run Duration	**90000 hours** **This can be changed in the:-** **Object Selector, Simulation, Run Duration**
Number of Samples	**721126**

Remove the 10 s from the MaxValue in the arrival distribution.

5.1.1 Assumptions

1. The Poisson distribution measures the number of arrivals per unit time. The inter-arrival time distribution for a Poisson process is the negative exponential distribution.
2. Constant service times, (degenerate service-time distribution).
3. Utilisation factor ρ<1
4. The system is in a steady state
5. Customers are served on a First In, First Out (FIFO) basis
6. The queue has an infinite capacity
7. Inter arrival time = 0.125 hours
8. Service time = 0.08333 hours



5.1.2 Flow Chart 1 Representation of the M/D/1 model

5.1.3 M/D/1/∞ Model of a single server queue with constant service times

Figure 2. M/D/1/∞ Model of a single server queue with constant service times

5.1.4 M/D/1/∞ Results of the single server queue with constant service time

M/D/1 Calculated Queue Characteristics									
		λ	**μ**	$\rho = \frac{\lambda}{\mu}$	**Po**	$Lq = \frac{\lambda^2}{2\mu(\mu-\lambda)}$	$Wq = \frac{\lambda}{2\mu(\mu-\lambda)}$	$Ls = Lq + \frac{\lambda}{\mu}$	$Ws = Wq + \frac{1}{\mu}$
Average Inter arrival time (h)	**Constant Service time (h)**	**λ Average Arrival rate (Customers per h)**	**μ Constant Service rate (Customers per h)**	**Utilisation fraction of the time the server is busy**	**Po Fraction of the time the server is idle**	**Lq Average No of customers in the queue**	**Wq Average time spent in the queue (h)**	**Ls Average No of customers in the system**	**Ws Average time spent in the system (h)**
0.125	0.08333	8	12.000	0.66667	0.33333	**0.66666**	**0.08333**	1.3333	0.16666

Simulated values	
Lq Average No of customers in the queue	**Wq Average time spent in the queue (h)**
0.676116	**0.0843825**

This simulation ran for 90000 hours.

Click onto the Queue; look in Output viewer to see the Average Queue Length, the Average Queue Time and the Maximum Length of the Queue.

5.2 M/D/1/∞ The finished model

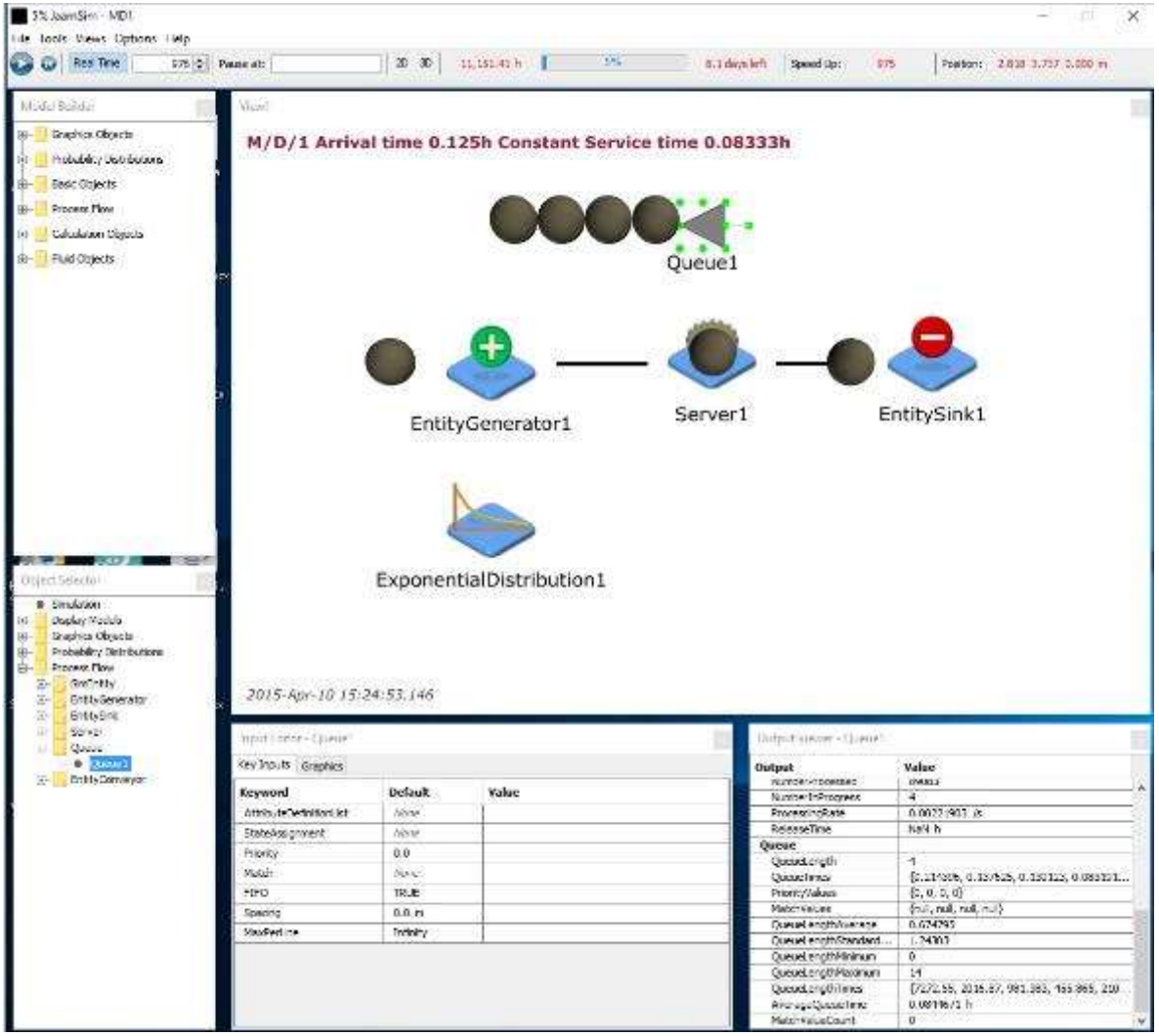


Figure 3.M/D/1/∞ The finished model

6.0 M/M/1/∞ Single server queue

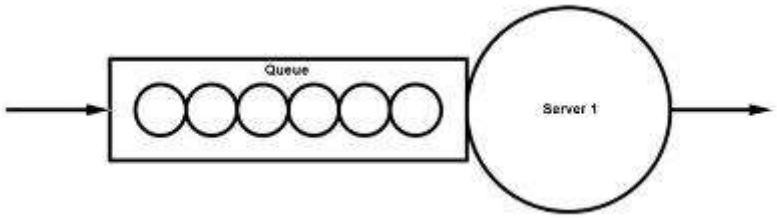


Figure 4. M/M/1/∞ Single server Queue

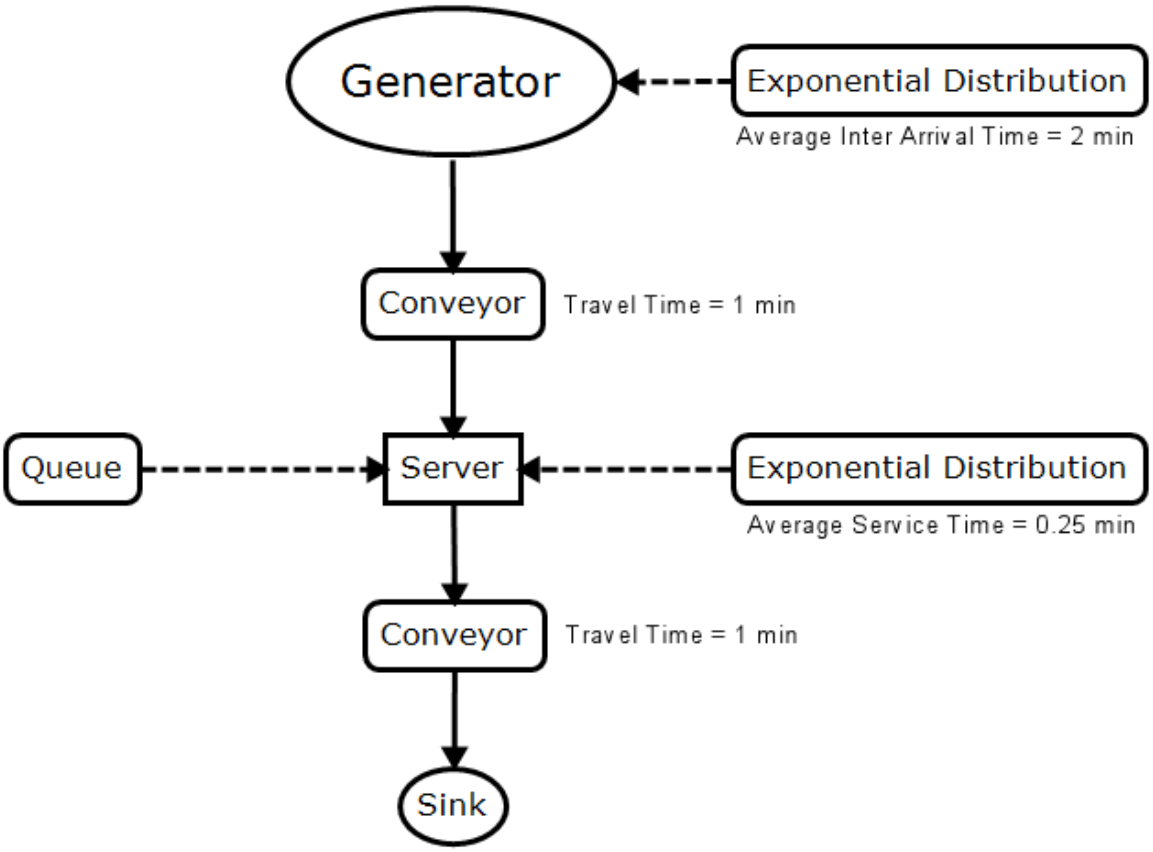
Edit the M/D/1 model to create a single server queue M/M/1 (See Figure 5.).

Entity	Inter arrival time Exponentially Distributed	Service time Exponentially Distributed	Travel time
Arrival Distribution Mean	2 minutes		
Server Distribution Mean		0.25 minutes	
Conveyor			1 minute

Run Duration	**2000000 minutes**
Number of Samples	**999168**

6.1.1 Assumptions

1. Inter-arrival and service times are independent and distributed exponentially.
   The Poisson distribution measures the number of arrivals per unit time.
   The inter-arrival time distribution for a Poisson process is the negative exponential distribution.
2. The system is in a steady state
3. Utilisation factor $\rho<1$
4. Customers are served on a First In, First Out (FIFO) basis
5. The queue has an infinite capacity
6. The inter-arrival time is 2 min
7. The Service time is 0.25 min



6.1.2 Flow chart 2 Representation of the M/M/1 model

Compare your simulated results of Lq and Wq with the mathematical results.

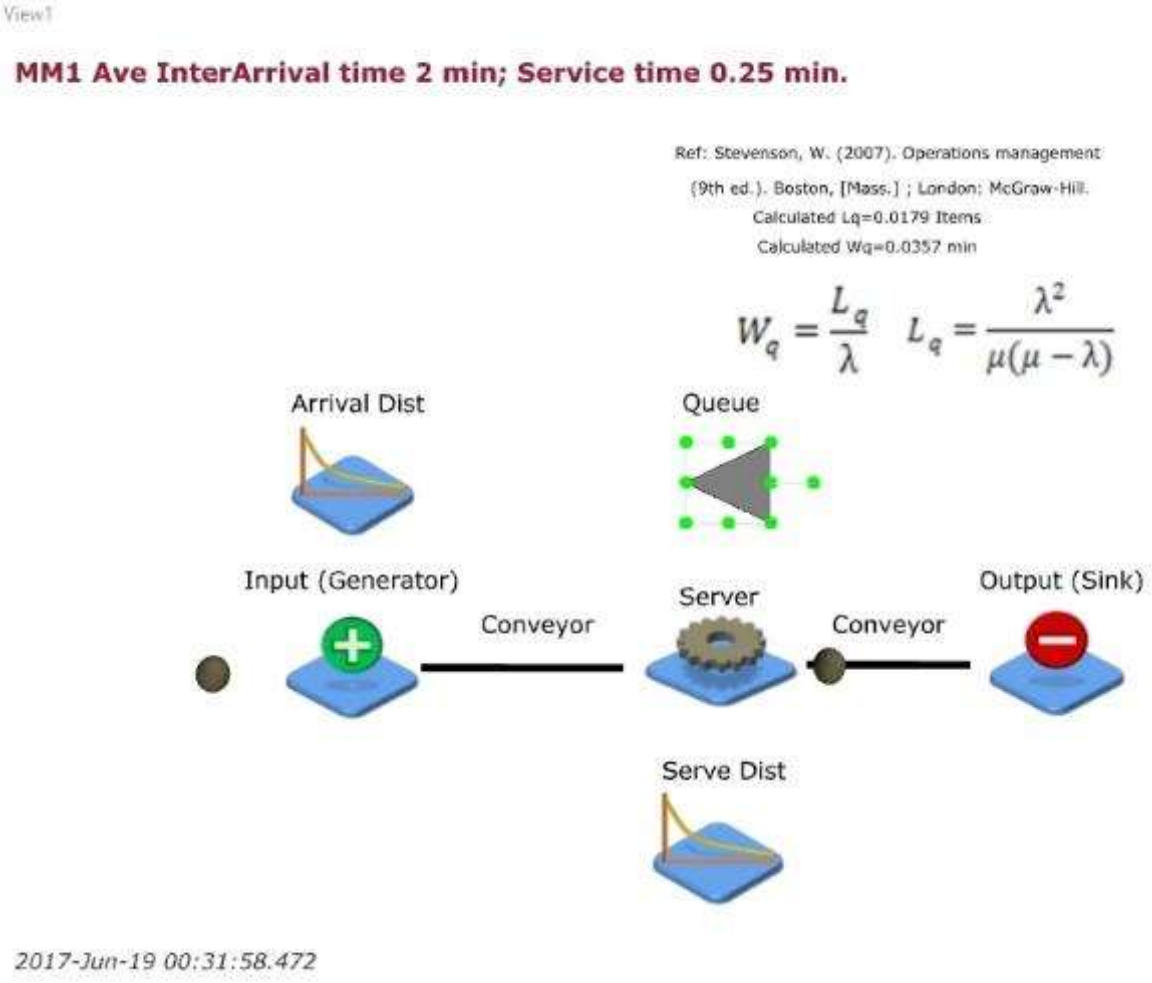


Figure 5. M/M/1/∞ Model of a single server queue

6.1.3 M/M/1/∞ Results of the single server Queue

M/M/1/∞ Calculated Queue characteristics									
λ	**μ**	$\rho = \frac{\lambda}{\mu}$		$L_q = \frac{\lambda^2}{\mu(\mu-\lambda)}$	$L_s = L_q + \rho$	$W_q = \frac{L_q}{\lambda}$	$W_s = W_q + \frac{1}{\mu}$	$P_o = 1 - \frac{\lambda}{\mu}$	$P_n = P_o\left(\frac{\lambda}{\mu}\right)^n$
Average Arrival rate per min	**Average Service rate per min**	**Utilisation Fraction of the time that all the servers are busy**	**Utilisation (%)**	**Lq Average No of customers in the queue (Customers)**	**Ls Average No of customers in the system (Customers)**	**Wq Average time spent in the queue (min)**	**Ws Average time spent in the system (min)**	**Po Probability of zero customers in the system (%)**	**Pn Probability of n customers in the system n = 4 (%)**
0.5	**4**	0.1250	12.5000	**0.0179**	0.1429	**0.0357**	0.2857	87.5000	0.0214

Simulated values	
Lq Average No of customers in the queue (Customers)	**Wq Average time spent in the queue (min)**
0.0178555	**0.0357406**

6.2 M/M/1/∞ The finished model

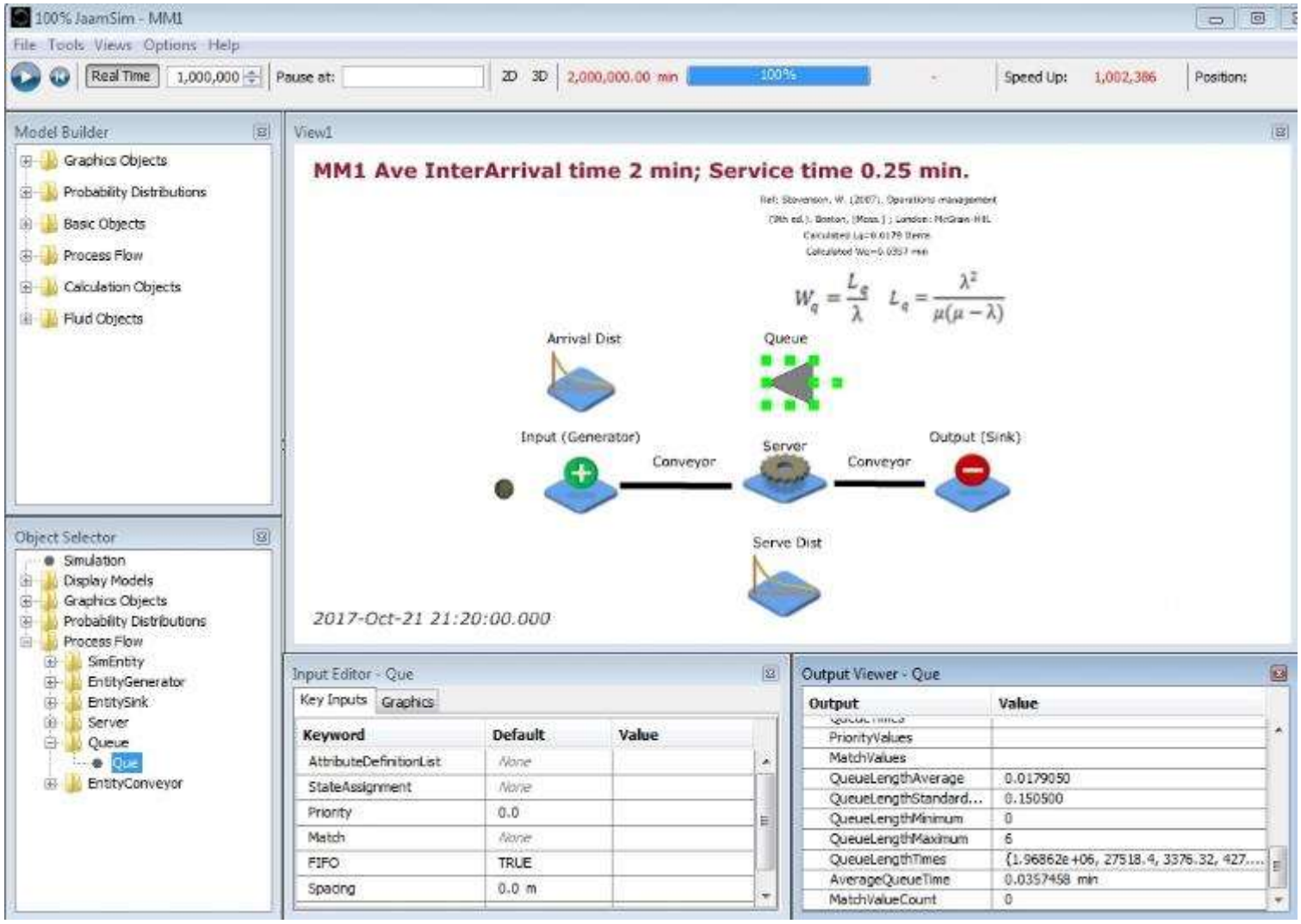


Figure 6.M/M/1/∞ The finished model of a single server queue

7.0 M/M/3/∞ Multi-server single queue

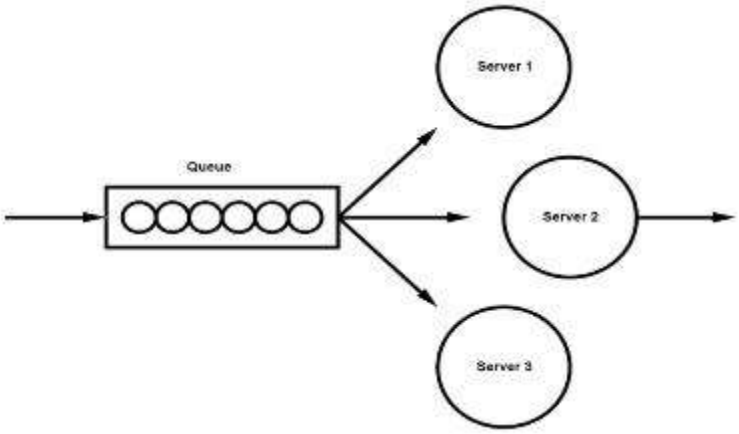


Figure 7.M/M/3/∞ Multi-server queue

Entity	Inter arrival time Exponentially Distributed	Service time Exponentially Distributed	Travel time
Arrival Distribution Mean	0.2 minutes		
Service Distribution Mean		0.4 minutes	
Conveyor			1 minute

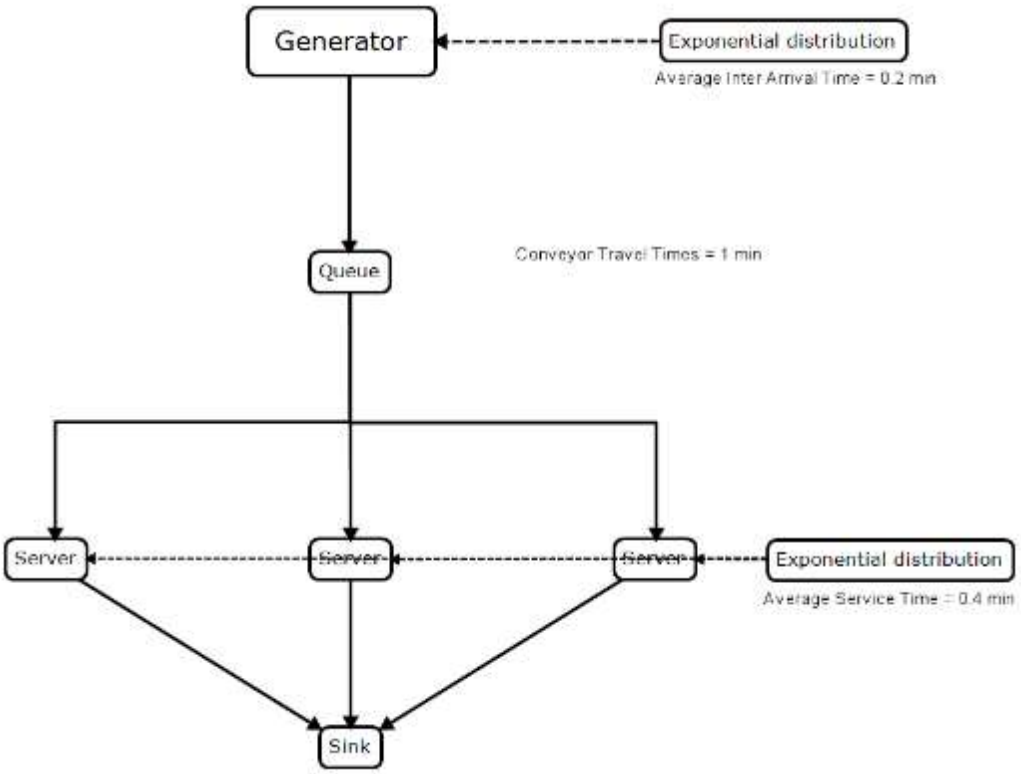
Run Duration	**3000000 minutes**
Number of Samples	**15000639**

7.1.1 Assumptions

1. Inter-arrival and service times are independent and distributed exponentially. The Poisson distribution measures the number of arrivals per unit time. The inter-arrival time distribution for a Poisson process is the negative exponential distribution.
2. The system is in a steady state
3. Utilisation factor ρ<1
4. Customers are served on a First In, First Out (FIFO) basis
5. The queue has an infinite capacity
6. Inter arrival time = 0.2 minutes
7. Service time = 0.4 minutes

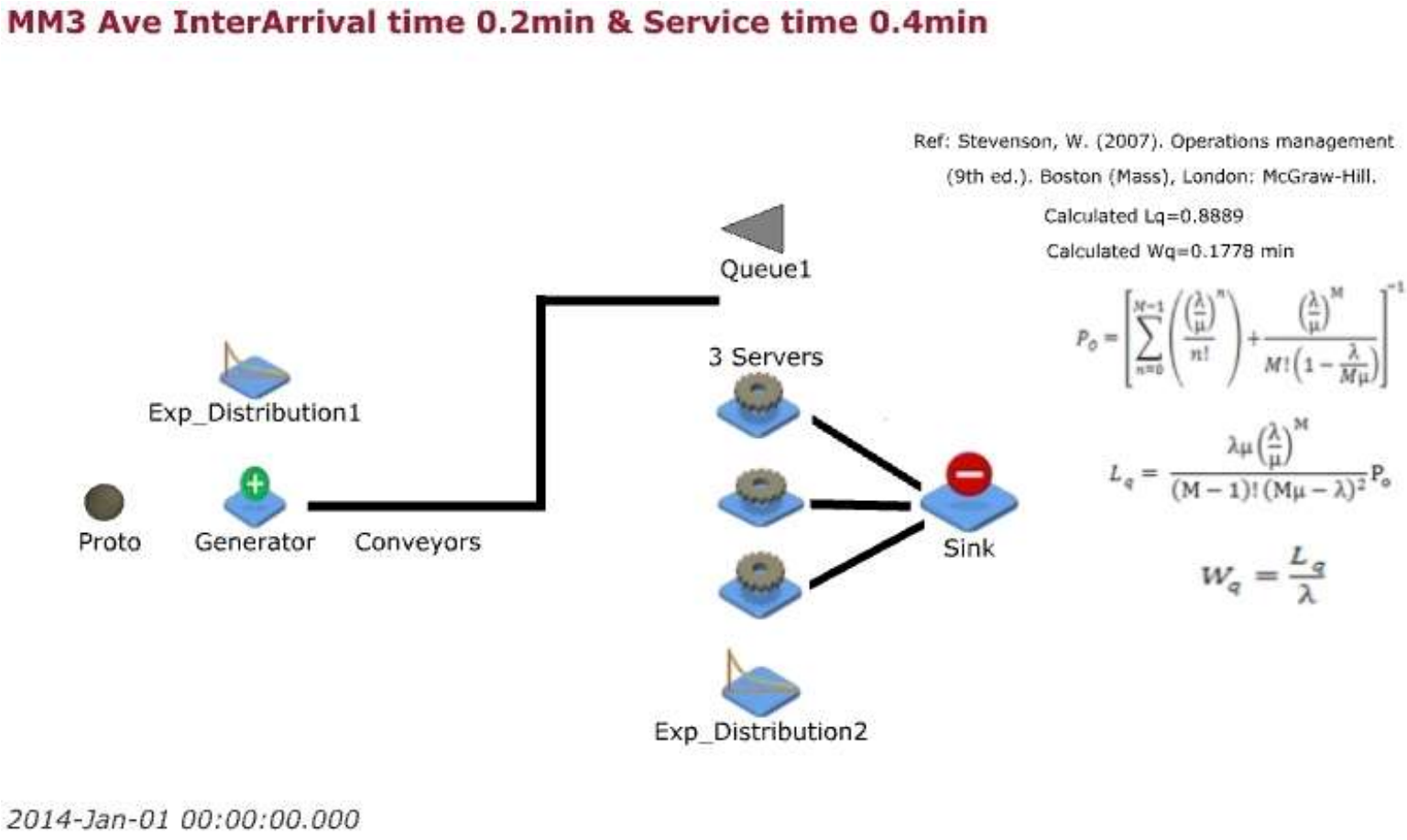
Build a new model to simulate a multi-server queue as shown in Figures7 and 8.

The model is built in the same way as the M/M/1 system.



7.1.2 Flow chart 3 Representation of M/M/3 model

7.2 M/M/3/∞ The model



$$P_0 = \left[\sum_{n=0}^{M-1}\left(\frac{\left(\frac{\lambda}{\mu}\right)^n}{n!}\right) + \frac{\left(\frac{\lambda}{\mu}\right)^M}{M!\left(1-\frac{\lambda}{M\mu}\right)}\right]^{-1}$$

$$L_q = \frac{\lambda\mu\left(\frac{\lambda}{\mu}\right)^M}{(M-1)!\,(M\mu-\lambda)^2}P_0$$

$$W_q = \frac{L_q}{\lambda}$$

Figure 8. M/M/3/∞ Multi-server queue

7.2.1 M/M/3/∞ Results of the multi-server Queue

M/M/3/∞ Calculated Queue characteristics										
λ	**μ**	**M**	$\rho = \frac{\lambda}{M\mu}$		$P_o = \left[\sum_{n=0}^{M-1}\left(\frac{\left(\frac{\lambda}{\mu}\right)^n}{n!}\right) + \frac{\left(\frac{\lambda}{\mu}\right)^M}{M!\left(1-\frac{\lambda}{M\mu}\right)}\right]^{-1}$		$L_q = \frac{\lambda\mu\left(\frac{\lambda}{\mu}\right)^M}{(M-1)!\,(M\mu-\lambda)^2}P_o$	$W_q = \frac{L_q}{\lambda}$	$W_s = W_q + \frac{1}{\mu}$	$L_s = L_q + \rho$
Arrival rate per min	**Service rate per min**	**Number of Servers**	**Utilisation Fraction of the time that all the servers are busy**	**Utilisation Fraction of the time that all the servers are busy (%)**	**Po Probability of no customers in the queue**	**Po (%)**	**Lq The average No of customers in the queue**	**Wq The average waiting time in the queue (min)**	**Ws Average time spent in the system (min)**	**Ls Average No of customers in the system**
5	2.5	3	0.67	66.67	0.111	11.11	**0.8889**	**0.178**	0.578	2.89

Simulated values	
Lq **Average number in the queue (Customers)**	**Wq** **Average time in queue (min)**
0.891868	**0.178366**

7.2.2 The time spent in the system Ws

Ws = Wq + Service time

Ws = 0.178 + 0.4 = 0.578 as in the table above.

The time spent in the system as calculated does not take into account any travelling time on the conveyor, hence:

Ws = Wq + Service time + Travel time

Ws = 0.178 + 0.4 + 2 = 2.578 minutes

7.3 M/M/3/∞ The finished Model

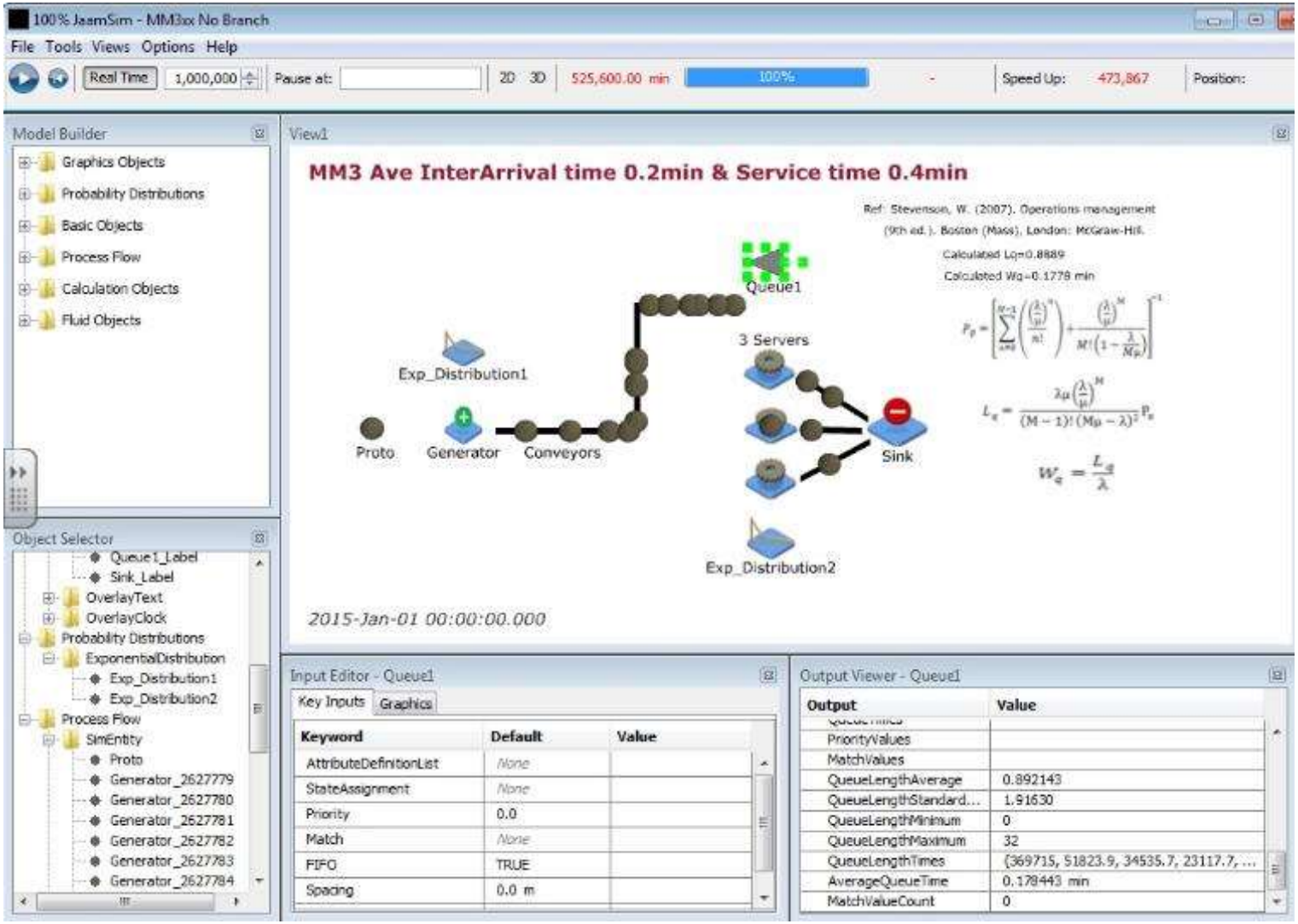


Figure 9The finished Model M/M/3/∞

8.0 M/M/1 An introduction to Flow lines in series

The following is based upon the work I found on the following web site:

http://homepages.cae.wisc.edu/~ie642/content/Applications/1-FlowLines/FlowLines2.htm

A flow line is a production line where a large number of parts are processed sequentially on a number of different machines or servers:

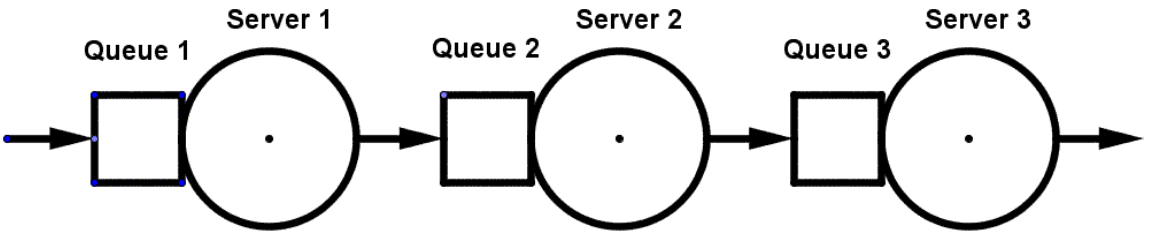


Figure 10 A simple three station flow line.

Flow lines are designed to break the operations down into a number of smaller tasks which are performed sequentially at each of the work stations as can be seen in Figure 10.

The production rate of any flow line is limited to that of the slowest part of the system, so ideally each work station should have the same production rate; this is called ‘line balancing’.

8.1 M/M/1 Series Example

A workshop consists of three machines arranged in a flow line as Figure 10.

It receives components to be processed at random intervals, the workshop is always busy, and it can be assumed that the system operates in a steady state.

Each station consists of a single machine and a storage area (the queue) which is assumed to have an infinite capacity for "work in progress".

Entity	Inter Arrival time Exponentially Distributed	Service time Exponentially Distributed	Travel time
Arrival Distribution Mean	10 minutes		
Lathe Distribution Mean		4 minutes	
Milling Distribution Mean		5 minutes	
Drilling Distribution Mean		8 minutes	
Conveyor			1 minute

Run Duration	**5000000 minutes**
Number of Samples	**498713**

The components feed into the assembly line and the workshop manager wants to make sure that the time the parts spend in the system is as short as possible.

The workshop presently uses a First-In-First-Out (FIFO) priority rule.

8.1.1 Assumptions

1. Inter-arrival and service times are independent and distributed exponentially. The Poisson distribution measures the number of arrivals per unit time. The inter-arrival time distribution for a Poisson process is the negative exponential distribution.
2. Utilisation factor $\rho<1$
3. The system is in a steady state
4. Customers are served on a First In, First Out (FIFO) basis
5. The queue has an infinite capacity
6. Samples enter each machine on the average every 10 minutes, (the arrival rate is the same for each machine).
7. On average each machine cycle is 4, 5 and 8 minutes respectively.

8.2 M/M/1 Series Machine utilization ρ

8.2.1 Machine 1.

Service rate μ_1 = 1/4 = 0.25 per minute

Arrival rate λ = 1/10 = 0.1 per minute

$$\rho = \frac{\lambda}{\mu}$$

$$\rho = \frac{0.1}{0.25} = 0.4$$

8.2.2 Machine 2.

Service rate μ_2= 1/5 = 0.2 per minute

$$\rho = \frac{0.1}{0.2} = 0.5$$

8.2.3 Machine 3.

Service rate μ_3 = 1/8 = 0.125 per minute

$$\rho = \frac{0.1}{0.125} = 0.8$$

8.3 M/M/1 Series Average number of customers in the queue Lq.

We can consider each work station as an independent single server queue M/M/1 and can be calculated from:

$$L_q = \frac{\lambda^2}{\mu(\mu - \lambda)}$$

Lq = average number of customers in the queue

8.3.1 Machine 1

$$L_q = \frac{0.1^2}{0.25 \times (0.25 - 0.1)} = 0.266667\ Components$$

8.3.2 Machine 2

$$L_q = \frac{0.1^2}{0.2 \times (0.2 - 0.1)} = 0.5\ Components$$

8.3.3 Machine 3

$$L_q = \frac{0.1^2}{0.125 \times (0.125 - 0.1)} = 3.2\ Components$$

8.4 M/M/1 Series Length of time spent in the queue Wq

The average time the samples spend in these buffers can now be estimated from Little's law.

$$W_q = \frac{L_q}{\lambda}$$

8.4.1 Machine 1

$$W_q = \frac{0.266667}{0.1} = 2.666667\ Minutes$$

8.4.2 Machine 2

$$W_q = \frac{0.5}{0.1} = 5\ Minutes$$

8.4.3 Machine 3

$$W_q = \frac{3.2}{0.1} = 32\ Minutes$$

8.5 M/M/1 Series Build a model using JaamSim to simulate this Flow Line.

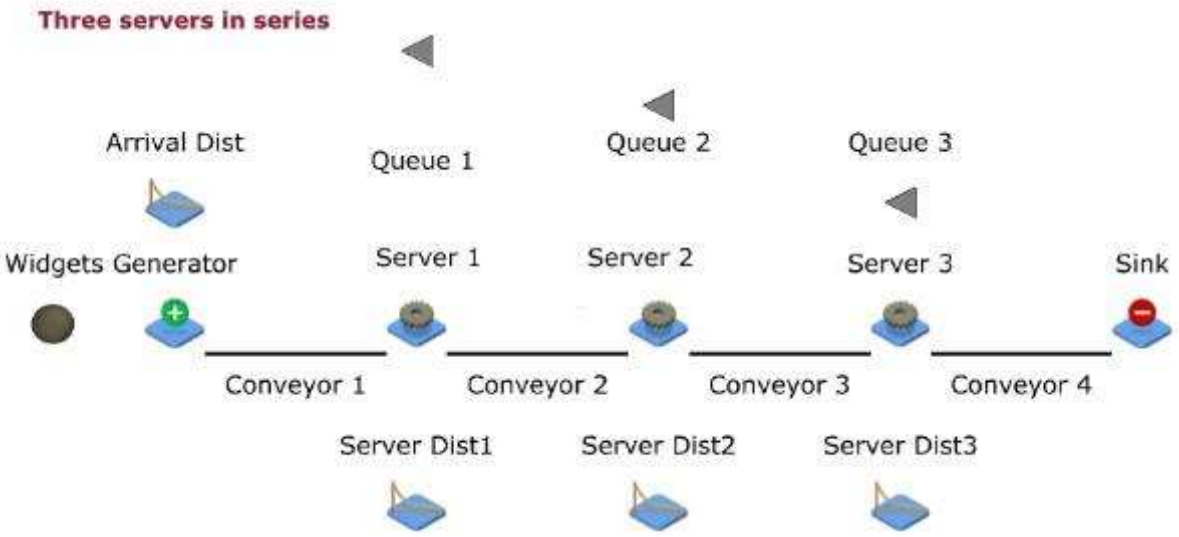


Figure 11. Flow line three machines in series.

Build the model just as you did for the M/M/1 model, but with three servers, three queues and three exponential distributions.

If the manager introduced a fixture to speed up the service time for station 3, by how much must the service time be reduced (to the nearest whole minute) to halve the length of the queue? (Need to solve a quadratic equation).

Show your calculations and compare these answers with your simulation.

8.6 M/M/1 Results of the Flow line in series

If we treat each machine as a separate queue then the following results will be valid for each of the three machines (Only), not for the whole system.

M/M/1 in Series - Calculated Queue characteristics												
					$\rho = \frac{\lambda}{\mu}$		$L_q = \frac{\lambda^2}{\mu(\mu-\lambda)}$	$L_s = L_q + \rho$	$W_q = \frac{L_q}{\lambda}$	$W_s = W_q + \frac{1}{\mu}$	$P_o = 1 - \frac{\lambda}{\mu}$	$P_n = P_o\left(\frac{\lambda}{\mu}\right)^n$
Machine	**Arrival time (min)**	**Service time (min)**	**λ Arrival rate per min**	**μ Service rate per min**	**Utilisation Fraction of the time that all the servers are busy**	**Utilisation (%)**	**Lq Average No of customers in the queue (Widgets)**	**Ls Average No of customers in the system (waiting and in service) (Widgets)**	**Wq The average time spent in queue (min)**	**Ws The average time spent in the system (min)**	**Po Probability of zero customers in the system (%)**	**Pn Probability of n customers in the system n = 4 (%)**
1	**10**	**4**	0.1	0.25000	0.40	40.0	**0.26667**	0.66667	**2.66667**	6.66667	60.00000	1.53600
2	**10**	**5**	0.1	0.20000	0.50	50.0	**0.50000**	1.00000	**5.00000**	10.00000	50.00000	3.12500
3	**10**	**8**	0.1	0.12500	0.80	80.0	**3.20000**	4.00000	**32.00000**	40.00000	20.00000	8.19200
3	**10**	**7**	0.1	0.14286	0.70	70.0	**1.63333**	2.33333	**16.33333**	23.33333	30.00000	7.20300

Simulated values- Machine 3 has an average serving time of 8 minutes					
Lq Average number in the queue (Customers)			**Wq The average time in queue (min)**		
Machine 1	**Machine 2**	**Machine 3**	**Machine 1**	**Machine 2**	**Machine 3**
0.262401	0.499308	3.16307	2.63074	5.00590	31.7119
Simulated values- Machine 3 has an average serving time of 7 minutes					
0.2686	0.5073	1.64208	2.6678	5.0573	16.3696

To calculate the time spent in the whole system we need to add the times together:

	Travelling time (min)	Service time at machine1 (min)	Time spent in the Queue at machine1 (min)	Travelling time (min)	Service time at machine2 (min)	Time spent in the Queue at machine2 (min)	Travelling time (min)	Service time at machine3 (min)	Time spent in the Queue at machine3 (min)	Travelling time (min)	Total (min)
Ws =	1	4	2.6667	1	5	5	1	8	32	1	60.6667

If the manager introduced a fixture to speed up the service time for station 3, by how much must the service time be reduced (to the nearest whole minute) to halve the length of the queue?

	Travelling time (min)	Service time at machine1 (min)	Time spent in the Queue at machine1 (min)	Travelling time (min)	Service time at machine2 (min)	Time spent in the Queue at machine2 (min)	Travelling time (min)	Service time at machine3 (min)	Time spent in the Queue at machine3 (min)	Travelling time (min)	Total (min)
Ws =	1	4	2.6667	1	5	5	1	7	16.333333	1	44.000

8.7 M/M/1 + M/M/1 + M/M/1 flow line model in series

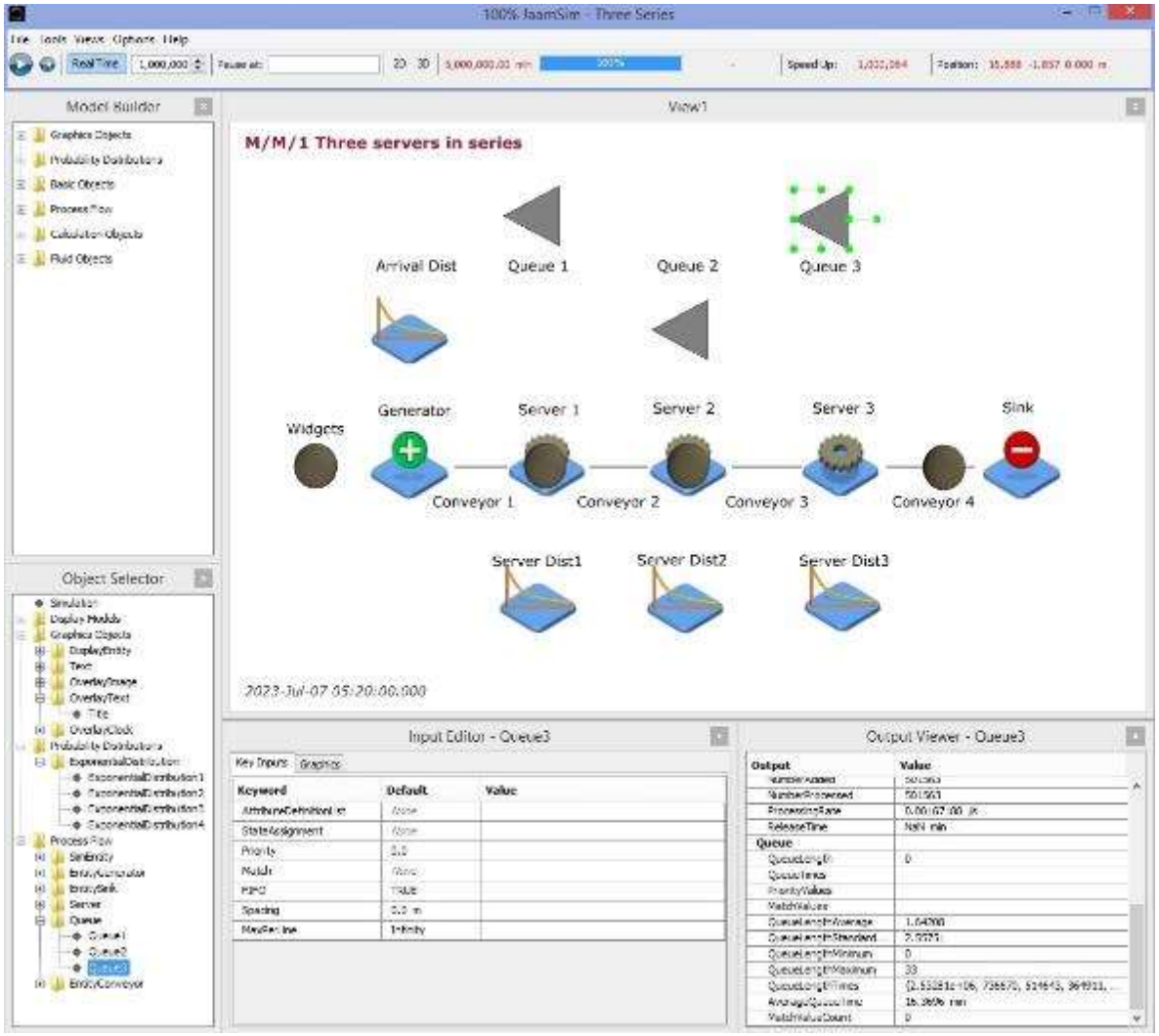


Figure 12. The finished model of a flow line in series

9.0 M/M/1 + M/M/1 +M/M/3.An Introduction to Flow lines

The following is based upon the work I found on the following web site:

http://homepages.cae.wisc.edu/~ie642/content/Applications/1-FlowLines/FlowLines2.htm

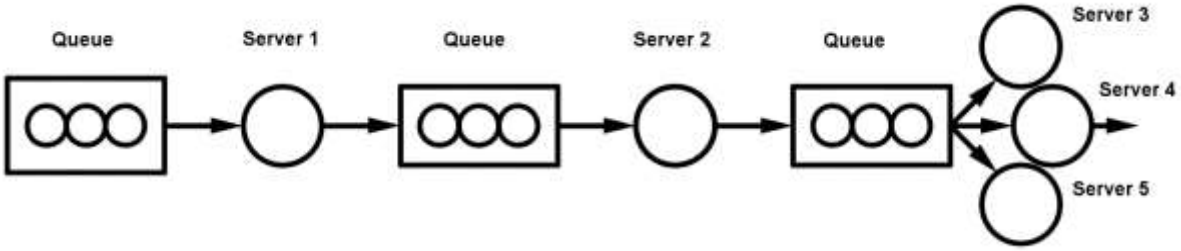


Figure 13. An Introduction to Flow lines

Build the model just as you did for the M/M/1 Series model, but with two single server queues and a three server queue.

Entity	Inter Arrival time Exponentially Distributed	Service time Exponentially Distributed	Travel time
Arrival Distribution Mean	10 minutes		
Lathe Distribution Mean		4 minutes	
Milling Distribution Mean		5 minutes	
Drilling Distribution Mean		8 minutes	
Conveyor			1 minute

Run Duration	**3000000 minutes**
Number of Samples	**299376**

9.1 M/M/1 + M/M/1 + M/M/3Flowline Model

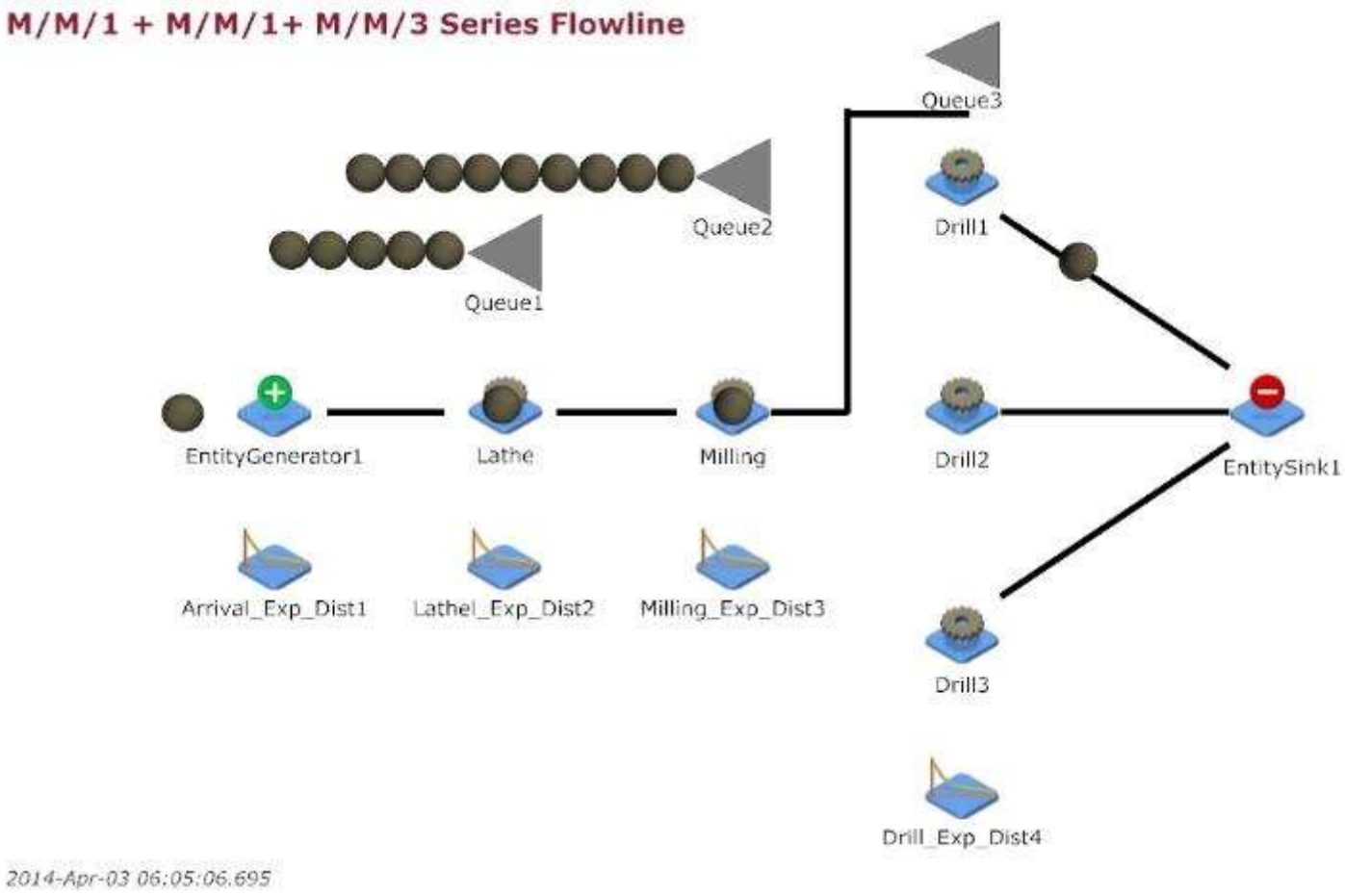


Figure 14. Flow line model.

9.2 M/M/1 + M/M/1 + M/M/3 Results of the flow line model

The Lathe and the Milling machine.

					$\rho = \frac{\lambda}{\mu}$		$L_q = \frac{\lambda^2}{\mu(\mu-\lambda)}$	$L_s = L_q + r$	$W_q = \frac{L_q}{\lambda}$	$W_s = W_q + \frac{1}{\mu}$	$P_o = 1 - \frac{\lambda}{\mu}$	$P_n = P_o \left(\frac{\lambda}{\mu}\right)^n$
Machine	**Arrival Time (min)**	**Service Time (min)**	**λ Arrival rate per (min)**	**μ Service rate per (min)**	**ρ Utilisation Fraction of the time that all the servers are busy**	**Utilisation %**	**Lq Average number in the queue**	**Ls Average number of customers in the system (waiting and in service)**	**Wq The average time in line (min)**	**Ws The average time in the system (min)**	**Po Probability of zero customers in the system %**	**Pn Probability of n customers in the system n = 4 %**
Lathe	10	4	0.1	0.25	0.4000	40.0000	0.2667	0.6667	2.6667	6.6667	60.0000	1.5360
Milling	10	5	0.1	0.2	0.5000	50.0000	0.5000	1.0000	5.0000	10.0000	50.0000	3.1250

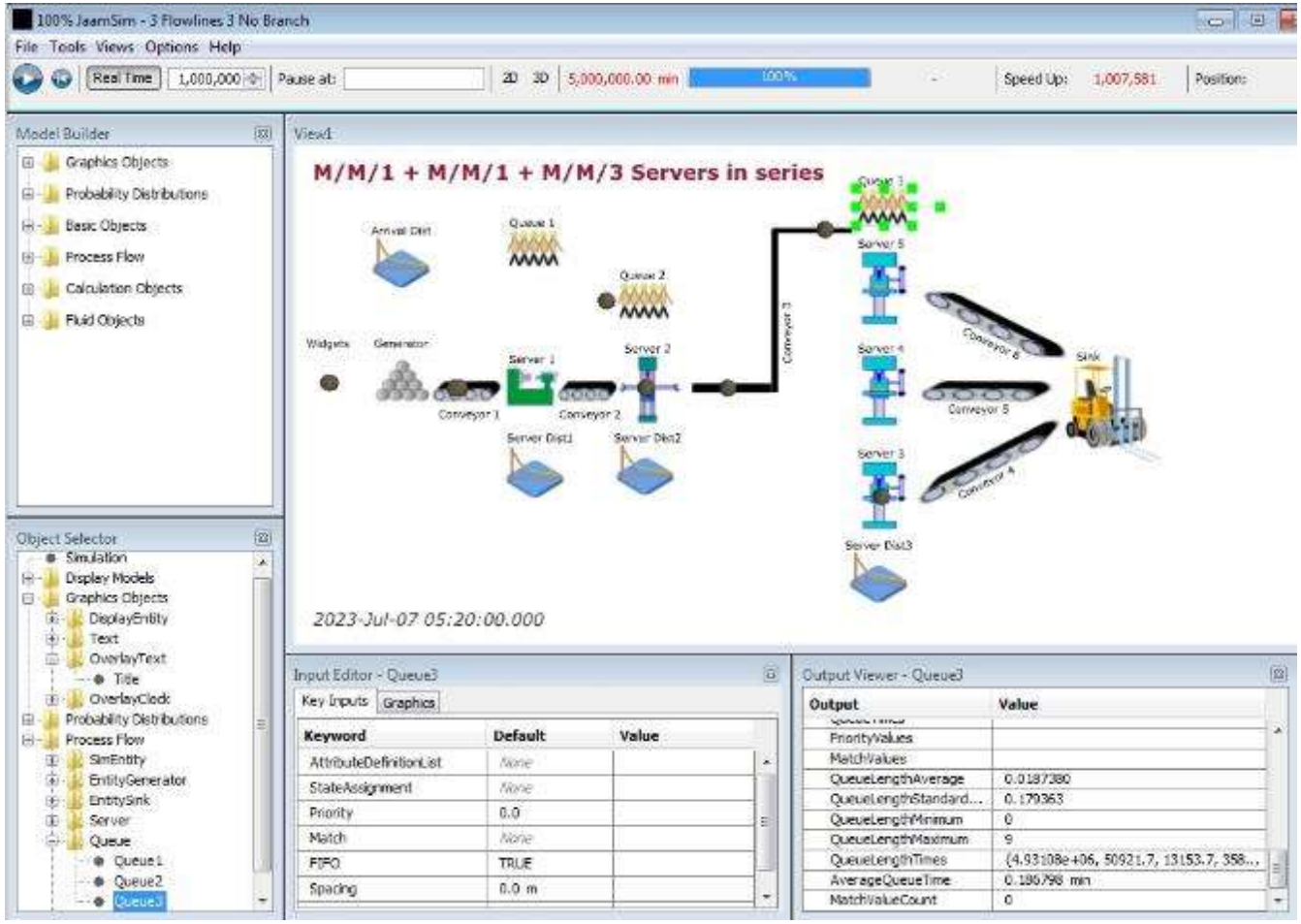
3 Drilling machines serving in parallel as part of a simple flow line.

					$\rho = \frac{\lambda}{M\mu}$		$P_o = \left[\sum_{n=0}^{M-1}\left(\frac{\left(\frac{\lambda}{\mu}\right)^n}{n!}\right) + \frac{\left(\frac{\lambda}{\mu}\right)^M}{M!\left(1-\frac{\lambda}{M\mu}\right)}\right]^{-1}$		$L_q = \frac{\lambda\mu\left(\frac{\lambda}{\mu}\right)^M}{(M-1)!(M\mu-\lambda)^2}P_o$	$W_q = \frac{L_q}{\lambda}$	$W_s = W_q + \frac{1}{\mu}$	$L_s = L_q + r$
Arrival Time (min)	**Service Time (min)**	**λ Arrival rate per (min)**	**μ Service rate per (min)**	**M Number of Servers**	**ρ Utilisation Fraction of the time that all the servers are busy**	**Utilisation - Time that all the servers are busy %**	**Po Probability of no customers in the queue**	**Po %**	**Lq The average number of customers in the queue**	**Wq The average waiting time in the queue (min)**	**Ws Average time spent in the system (min)**	**Ls Average number of customers in the system**
10	8	0.1	0.125	3	0.27	26.67	0.447	44.72	0.0189	0.189	8.189	0.82

Simulated Results		
Machine	**Lq Average number in the queue**	**Wq The average time in the line (min)**
Lathe	0.262443	2.62989
Milling	0.495754	4.96786
Drilling	0.0190278	0.190675
Run Duration	3000000 minutes	

Longer run durations may give closer results to the mathematical theory

9.3 M/M/1 + M/M/1 + M/M/3 The finished flow line model



You can change the graphics of a model to represent the machines etc, you are modelling, see later in this document.

Figure 15. M/M/1 + M/M/1 + M/M/3 The finished flow line model

10.0 M/M/1/K A single server Queue with finite length

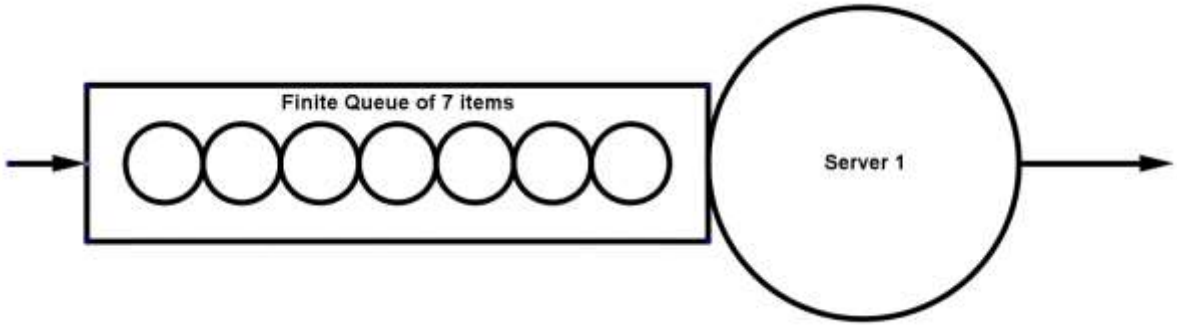


Figure 16. A single server queue with finite length

The most common example of a finite queue such as this might be at a hair dressers where there is only one chair in which to get your hair cut (1 server) and say six chairs for customers to sit and wait their turn (the queue).

When all six chairs are filled with customers there is no more room in the shop and any further arrivals are turned away until a space in the queue becomes vacant.

This begs the question of how much is it costing the business in lost income by turning customers away and how many servers should the shop have? I will leave that for you to think about.

Example

Suppose items arrive at a queue every 0.025 minutes and the service time is 0.02 minutes, the queue is limited to 6 items, that means the maximum number of items in the system at any one time is 7 items, 6 in the queue and one being served.

Entity	Inter Arrival time Exponentially Distributed	Service time Exponentially Distributed	Travel time
Arrival Distribution Mean	0.025 minutes		
Service Distribution Mean		0.02 minutes	
Conveyor			0.025 minutes

Max Queue Length	6
Run Duration	300000 minutes

10.1 M/M/1/K The model of a single server queue with finite length

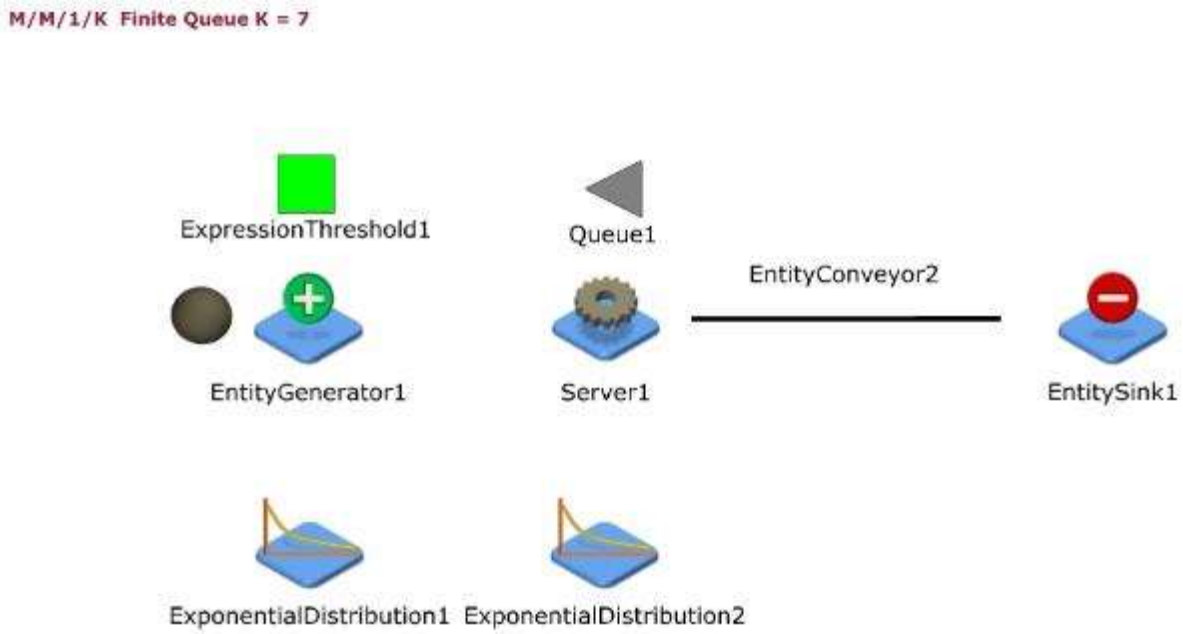


Figure 17. M/M/1/K A single server Queue with finite length

Build the model the same as the M/M/1, but without the conveyor between the generator and server and add an **ExpressionThreshold**.

There is no conveyor between the Generator and the server; it took me a while to figure this one out. The first model I made had a conveyor between the generator and the server, and I could not figure out why the **QueueLengthMaximum** was exceeding the set limit. The **ExpressionThreshold** should close entry to the queue when **QueueLengthMaximum** is reached.

It did not because the items were on the conveyor and not in the queue, hence entry to the queue was not restricted until items left the conveyor and entered the queue and the **QueueLengthMaximum** condition was met.

This meant more items were let through than should have been for a finite queue.

This problem could be solved with the clever use of an expression, we will consider expressions later.

10.1.1 ExpressionThreshold

The green box is an **ExpressionThreshold**, this will limit the size of the queue.

It could be Green, Yellow, Magenta or Red, each colour denotes what it's doing, Green = Open; *Yellow = Pending Open; *Magenta = Pending Closed; Red = Closed.

*The pending states are intended for debugging. If you see them, it means that the underlying threshold object has not been updated to match the value for the expression. This can happen if the expression's value changes without an event occurring.

Set up the expression threshold as shown in Figure 18.

The expression '[Queue1].QueueLength< 6' will limit the number of items in the system. Flow to the queue will be stopped when the number in the queue reaches 6 and the number in the system will be 7, since there will be one item being served.

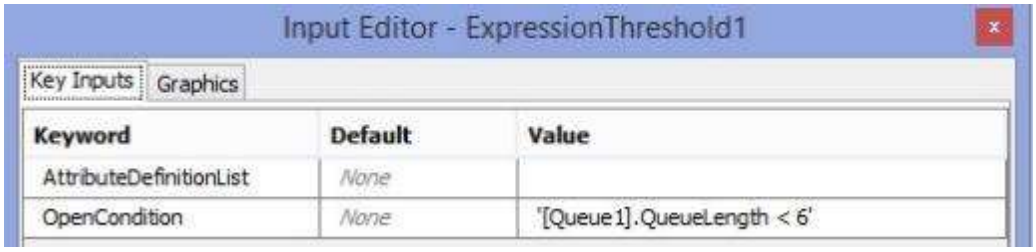


Figure 18. The ExpressionThreshold input.

10.1.2 The Entity Generator

Set up the Entity Generator as shown in Figure 19.

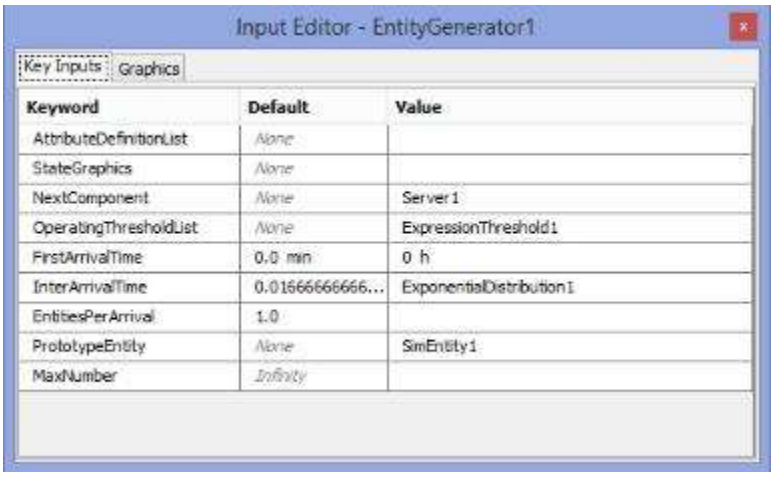


Figure 19. Setting for the Entity Generator

The **OperatingThresholdList** links the Entity Generator to the **ExpressionThreshold** we have set as '[Queue1].QueueLength< 6'.

10.2 M/M/1/K Calculations - A single server Queue with finite length

10.2.1 Assumptions

1. Poisson arrivals and exponential services. The Poisson distribution measures the number of arrivals per unit time. The inter-arrival time distribution for a Poisson process is the negative exponential distribution.
2. The system is in a steady state.
3. The queue has a finite capacity of 6 items.
4. Utilisation factor ρ>1 - Arrival rate does not have to be <= the service rate.
5. System is full arrival rate λ= 0
6. Number of customers in system does not go to ∞
7. Customers that arrive when the Queue is full are turned away.
8. Samples enter the queue every 0.025 minutes.
9. On average the service takes 0.02 minutes.

10.2.2 Formulae

$$Ls = \frac{\rho}{(1-\rho)} - \frac{(K+1(\rho^{(K+1)})}{(1-\rho^{(K+1)})}$$

$$Lq = Ls - (1 - Po)$$

$$Po = \frac{(1-\rho)}{(1-\rho^{(K+1)})}$$

$$\rho^n = \left(\frac{\lambda}{\mu}\right)^n$$

$$P_n = \frac{\rho^n}{n!} Po$$

$$\bar{\lambda} = \lambda(1 - P_k$$

$$Wq = \frac{Lq}{\bar{\lambda}}$$

$$Ws = \frac{Ls}{\bar{\lambda}}$$

10.3 M/M/1/K Results of a single server queue with finite length

10.3.1 M/M/1/K Calculated results

Vary the value of K and see what happens.

Number of servers =1; Arrival time = 0.025 min; Service = 0.02 min							
K	2	3	4	5	6	7	10
lambda	40	40	40	40	40	40	40
Mu	50	50	50	50	50	50	50
rho	0.8	0.8	0.8	0.8	0.8	0.8	0.8
Ls Customers	0.8525	1.2249	1.5631	1.8683	2.1424	2.3872	2.9663
Lq Customers	0.2623	0.5637	0.8605	1.1394	1.3955	1.6276	2.1851
Ws min	0.0289	0.037	0.0445	0.0513	0.0574	0.0628	0.0759
Wq min	0.0089	0.0170	0.0245	0.0313	0.0374	0.0428	0.0559
Po	0.4098	0.3388	0.2975	0.2711	0.2531	0.2043	0.2188
K	2	3	4	5	6	7	10

10.3.2 M/M/1/K Simulated results

Vary the value of K and see what happens.

Number of servers =1; Arrival time = 0.025 min; Service = 0.02 min; Run time = 150000 min							
K	2	3	4	5	6	7	10
lambda	40	40	40	40	40	40	40
Mu	50	50	50	50	50	50	50
rho	0.8	0.8	0.8	0.8	0.8	0.8	0.8
Lq Customers	0.2624	0.5641	0.8613	1.1402	1.3964	1.6254	2.188
Wq min	0.0088	0.0170	0.0245	0.0312	0.0374	0.0428	0.056
K	2	3	4	5	6	7	10

10.4 M/M/1/K The finished model of a single server queue with finite length

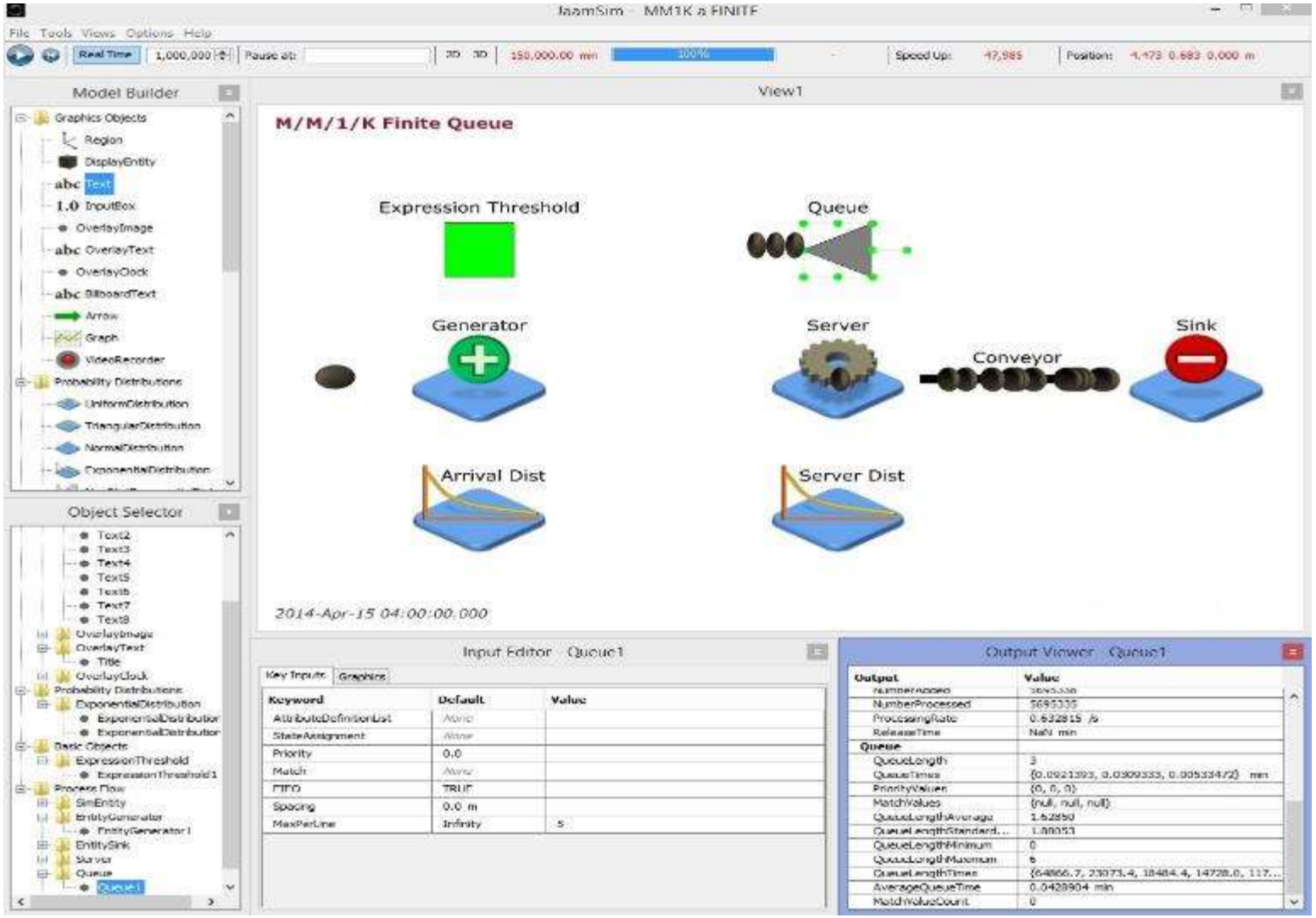


Figure 20. The finished model of a single server queue with finite length

11.0 M/M/1/K A finite queue with baulking

Modify the previous example:-

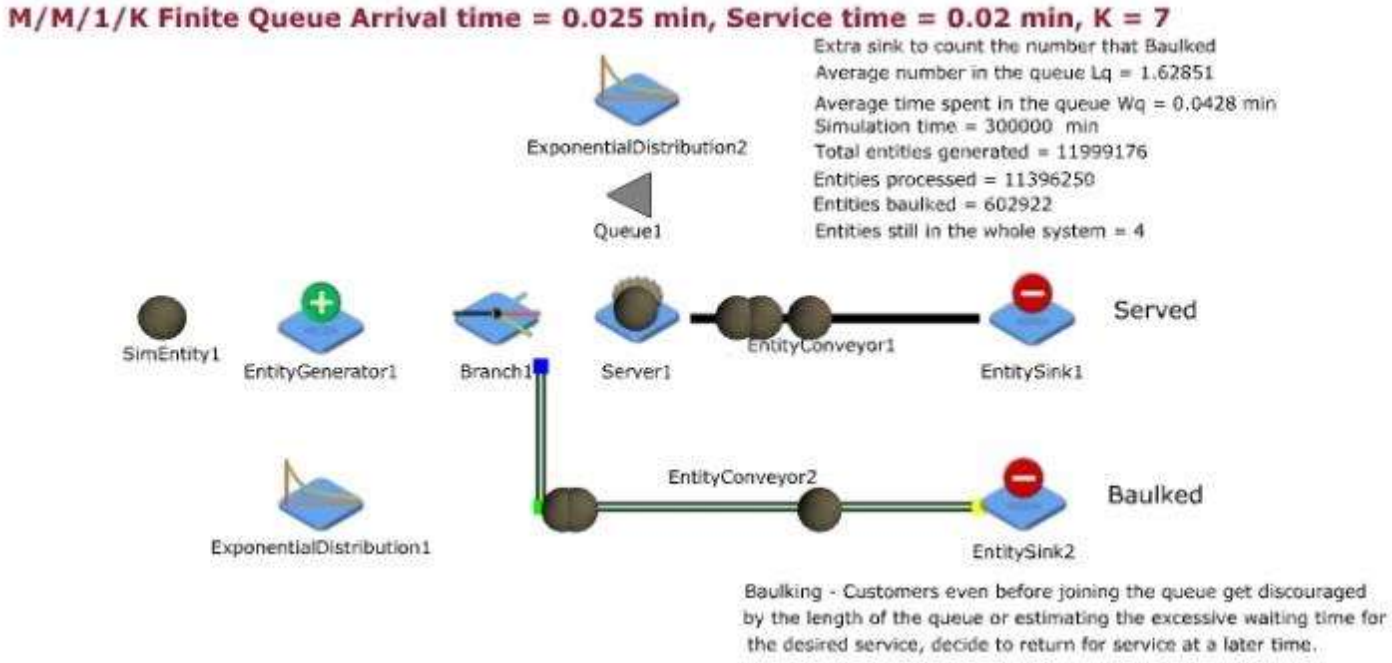


Figure 21. M/M/1/K Finite Queue With Baulking

Entity	Inter Arrival time Exponentially Distributed	Service time Exponentially Distributed	Travel time
Arrival Distribution Mean	0.025 minutes		
Service Distribution Mean		0.02 minutes	
Conveyors			0.025 minutes

Run Duration	300000 minutes

Run the model and add up the number of items baulked, processed and in the system, compare these with the number generated. The numbers should match as they did in my model above.

11.1.1 The Branch

Set up the Branch as follows:-

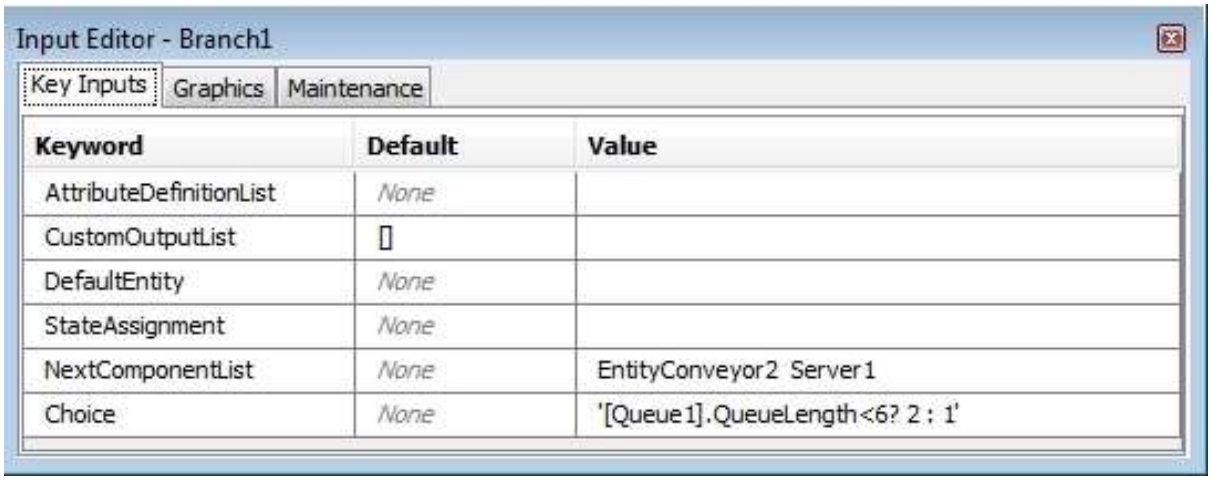


Figure 22. The Branch Baulking.

The branch Choice needs a condition to tell it where to send the items.

Condition 1. - Server1 = [Queue1].QueueLength<6

Condition 2. - Else send them to EntityConveyor2

11.2 M/M/1/K The finished model of a finite queue with baulking

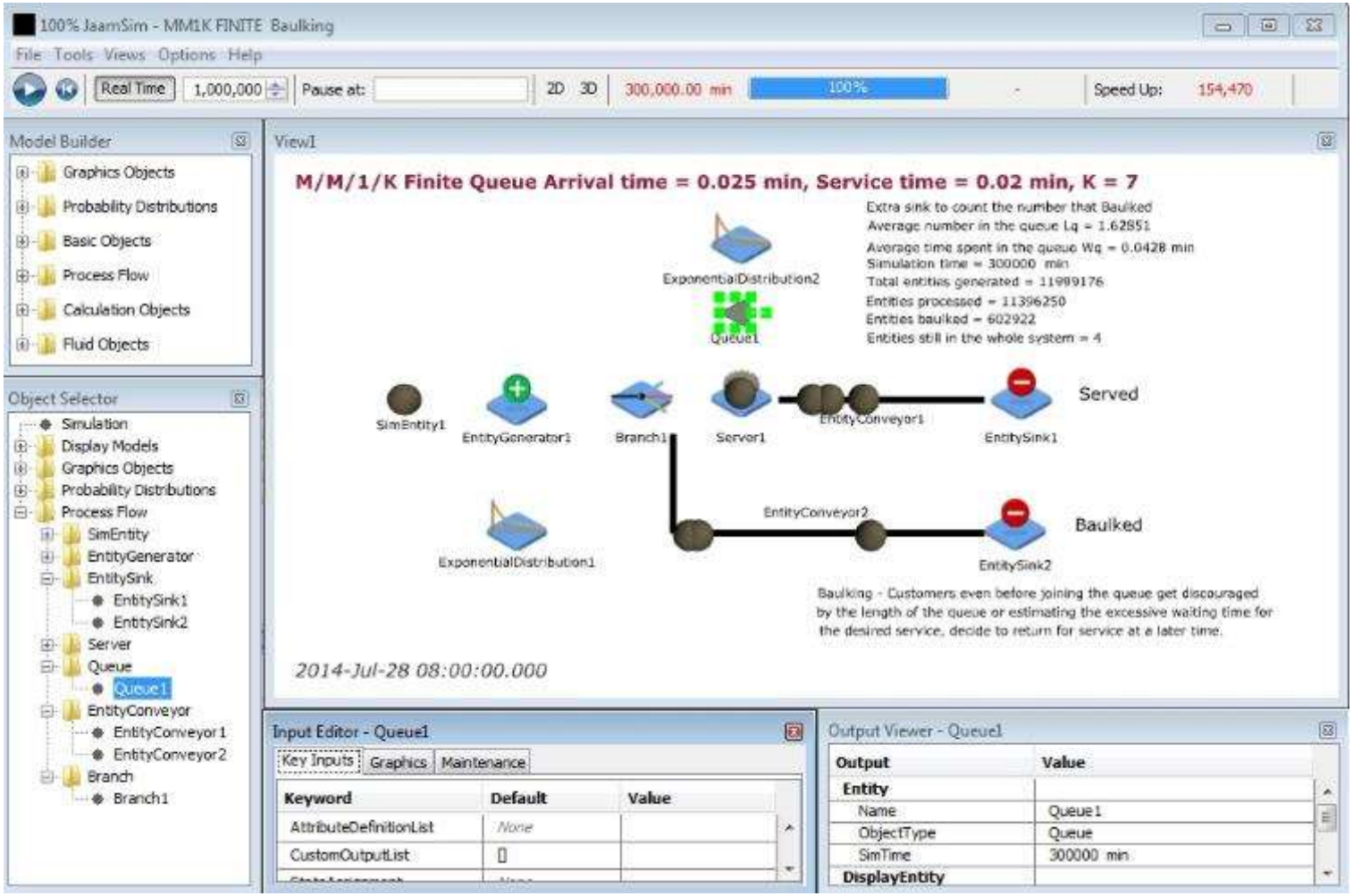


Figure 23. Finite Queue Measuring Baulking

12.0 M/M/2/3 Finite Queue

This model is similar to the M/M/1/K example, but with more servers, I have only considered 2 servers, as you will see the maths gets more laborious the more servers you have.

Build the model as before, but with 2 servers

Entity	Inter Arrival time Exponentially Distributed	Service time Exponentially Distributed	Travel time
Arrival Distribution Mean	5 minutes		
Service Distribution Mean		7 minutes	
Conveyors			2 minutes

Run Duration	5000000 minutes

The number of items in the system K = 3, this means that the **ExpressionThreshold** needs setting as:-

'[Queue1].QueueLength< 1'

So that when the queue is empty an item is let through.

One item in the queue and 2 items being served gives K = 3.

Once you have built the model and verified that it works you can experiment with the values or add more servers to see what happens.

12.1 M/M/2/3 The model

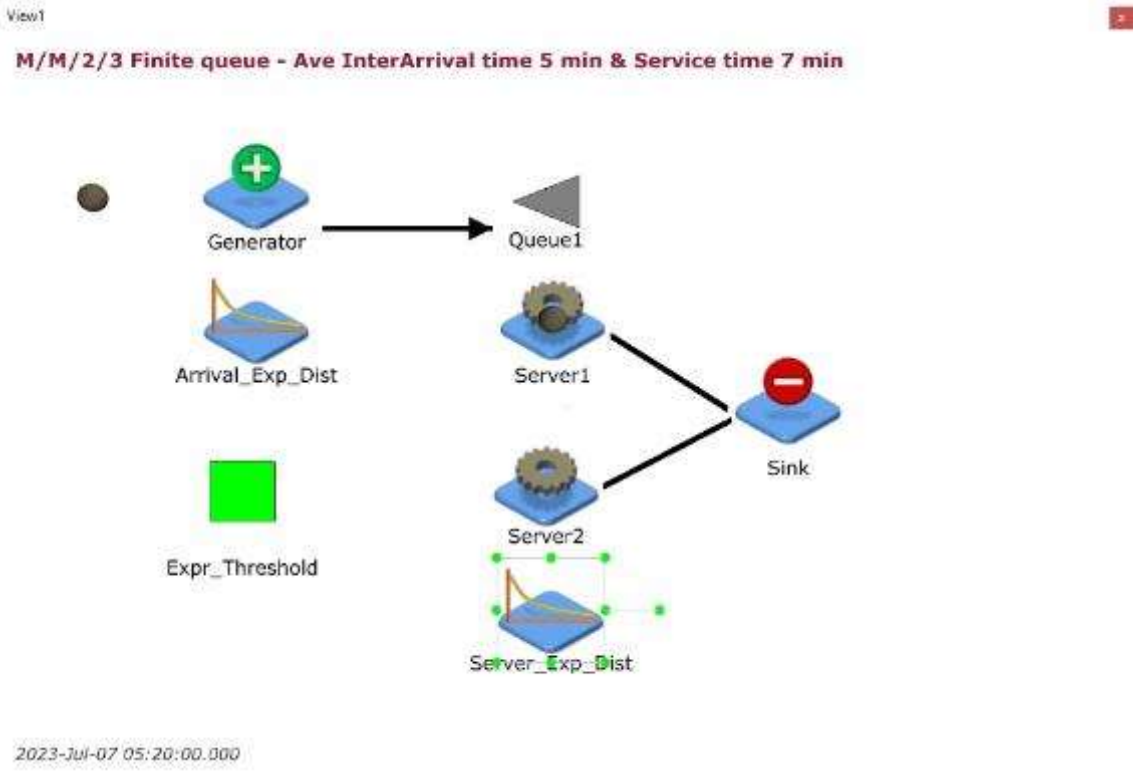


Figure 24 M/M/2/3 The model

We could try and count the number of items that Baulked with this model.

12.2 M/M/2/3 Calculations. A single server Queue with finite length

12.2.1 Assumptions

1. Poisson arrivals and exponential services. The Poisson distribution measures the number of arrivals per unit time. The inter-arrival time distribution for a Poisson process is the negative exponential distribution.
2. The system is in a steady state.
3. The queue has a finite capacity of 1 item.
4. Utilisation factor ρ>1 - Arrival rate does not have to be <= the service rate.
5. If the system is full - arrival rate λ= 0
6. Number of customers in system does not go to ∞
7. Customers that arrive when the Queue is full are turned away.
8. Samples enter the queue every 5 minutes.
9. On average the service takes 7 minutes.

12.2.2 M/M/2/3 Formulae

$$P_o = \left[\sum_{n=0}^{M} \frac{\frac{\lambda}{\mu}}{n!} + \frac{\frac{\lambda}{\mu}}{M!} \sum_{n=M+1}^{K} \left(\frac{\lambda}{M\mu}\right)^{n-M}\right]^{-1}$$

$$Lq = \frac{Po\left(\frac{\lambda}{\mu}\right)^{M}\rho}{M!\,(1-\rho)^2}[1 - \rho^{K-M} - (K-M)\rho^{K-M}(1-\rho)]$$

$$Where\ \ \rho = \frac{\lambda}{M\mu} \qquad and \qquad \lambda \neq M\mu$$

$$L_s = \sum_{n=0}^{M-1} nP_n + L_q + M\left(1 - \sum_{n=0}^{M-1} P_n\right)$$

$$Ws = \frac{Ls}{\bar{\lambda}}$$

$$Wq = \frac{Lq}{\bar{\lambda}}$$

Where

$$\bar{\lambda} = \lambda(1 - P_k)$$

From the following we can get P_k; When n = K

$$P_n = \frac{\left(\frac{\lambda}{\mu}\right)^{n} P_o}{M!\,M^{(n-M)}} WhereM \leq n \leq K$$

12.3 M/M/2/3 Results. Finite Queue

12.3.1 M/M/2/3 Calculated results

K Finite queue Number customers in the system	M No Servers	Arrival time (Customers per min)	Service time (Customers per min)	λ Arrival rate (min)	μ Service rate (min)	$\frac{\lambda}{\mu}$	$\rho = \frac{\lambda}{M\mu}$	Po	Lq The average number in the queue (Customers)	Lambda Bar $\bar{\lambda}$	Wq The average waiting time in the queue (min)	Ls Average number of in the system (Customers)	Ws Average time spent in the system (min)
3	2	5	7	0.20	0.14	1.40	0.70	0.2459	**0.1687**	0.1663	**1.0148**	1.3325	8.0148

12.3.2 M/M/2/3 Simulated results

Simulated values	
Lq Average number in the queue (Customers)	**Wq Average time in queue (min)**
0.168194	**1.01234**

12.4 M/M/2/3 The finished model. Finite Queue

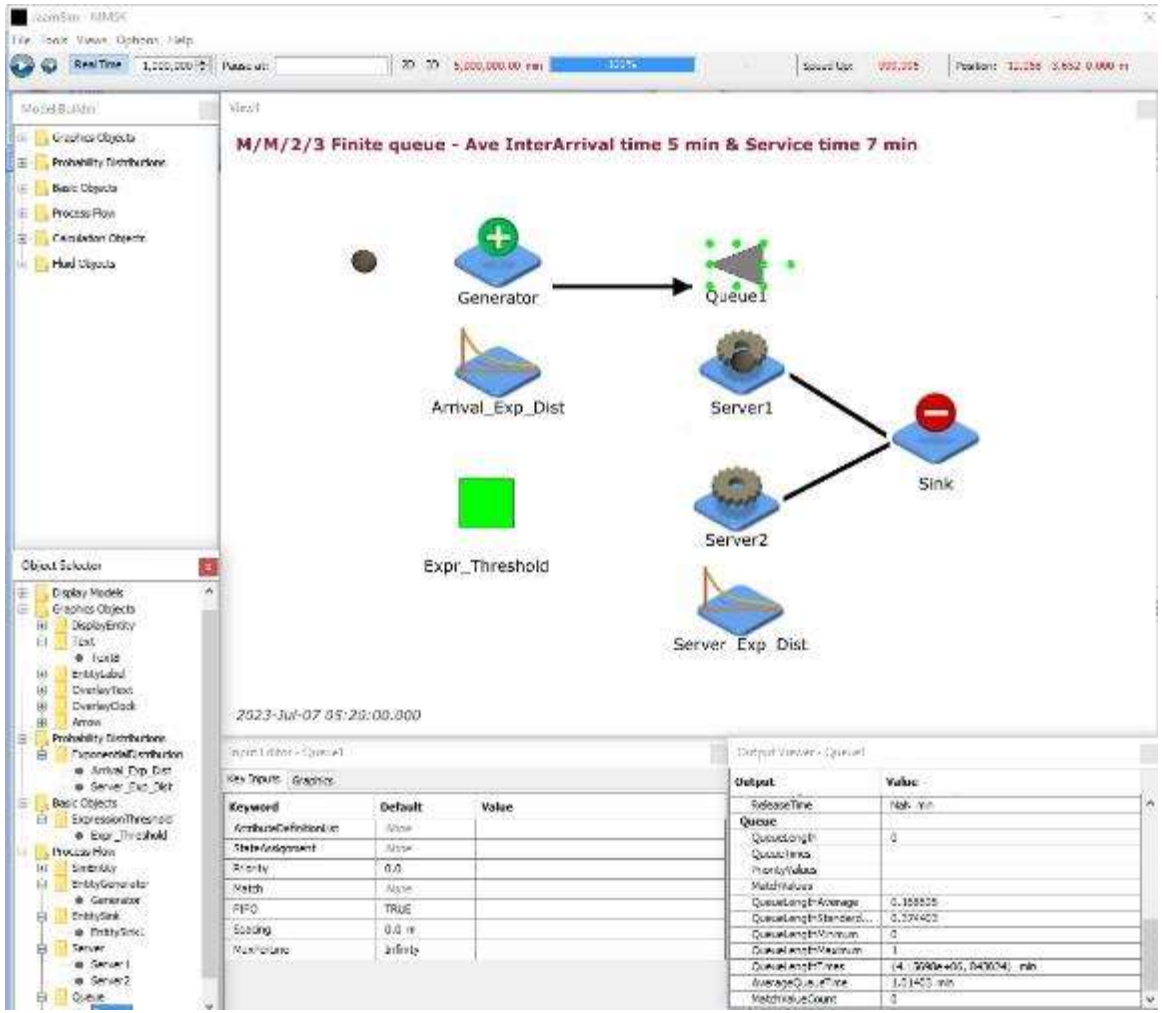


Figure 25 M/M/2/3 The finished model. Finite Queue

13.0 Reneging

A new feature recently added allows models to display reneging behaviour, and it follows on nicely from the M/M/2/K model in the previous example.

Edit the previous model to include the entities shown in **Figure 26.**

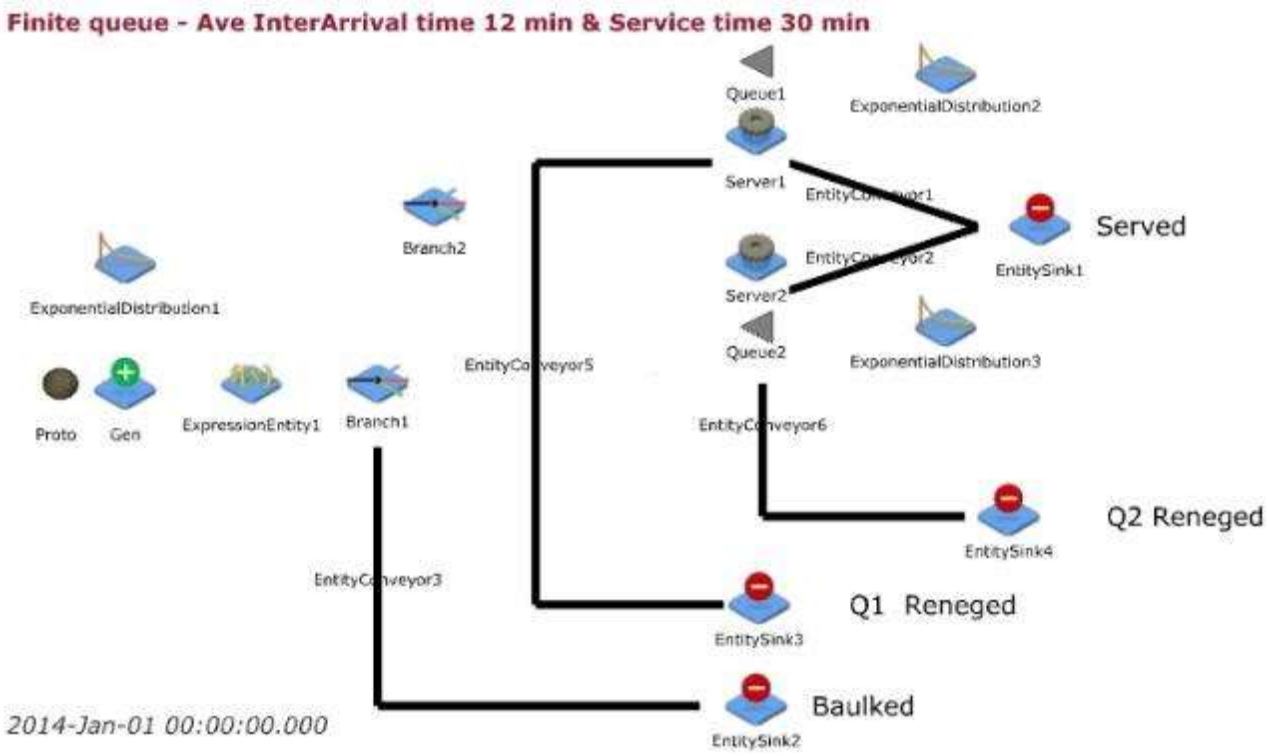
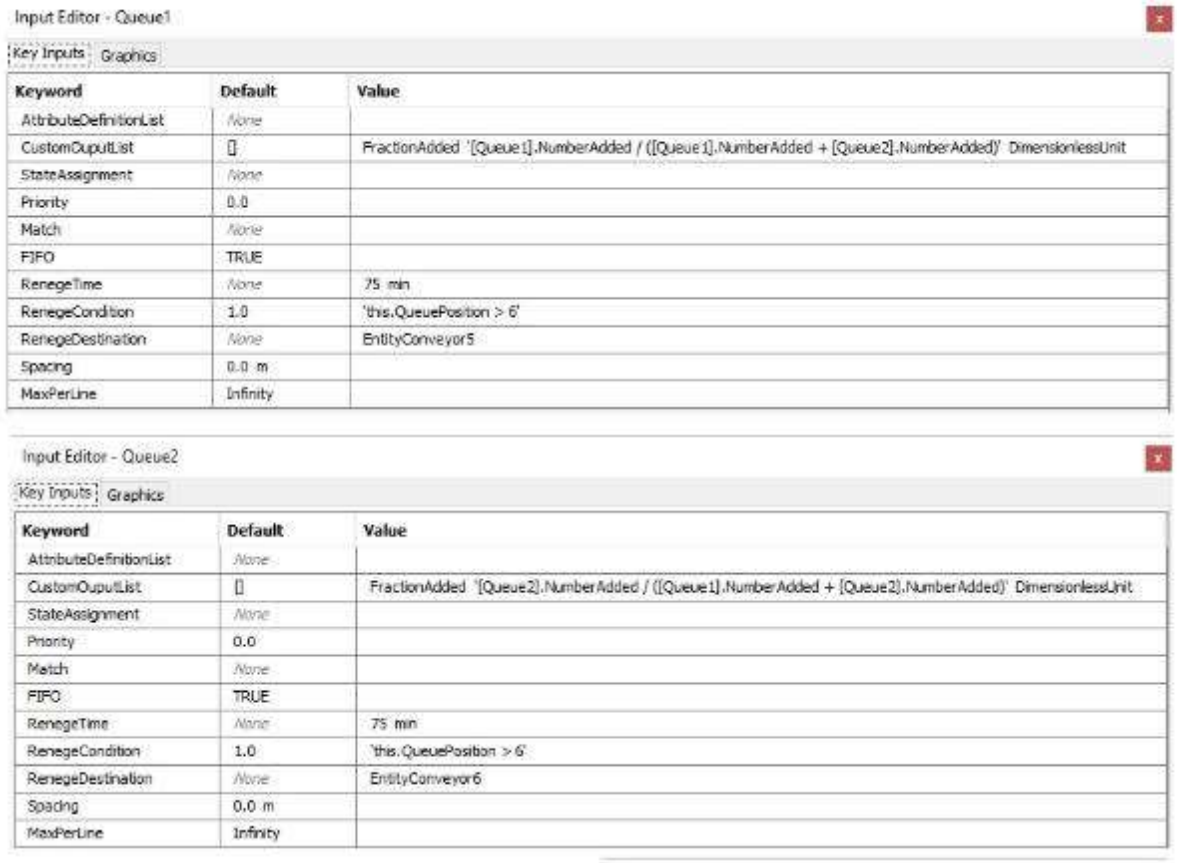


Figure 26. Renege and Baulking model

Set up each entity as follows:-



Input Editor - Queue1

Key Inputs | Graphics

Keyword	Default	Value
AttributeDefinitionList	*None*	
CustomOutputList	[]	FractionAdded '[Queue1].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded)' DimensionlessUnit
StateAssignment	*None*	
Priority	0.0	
Match	*None*	
FIFO	TRUE	
RenegeTime	*None*	75 min
RenegeCondition	1.0	'this.QueuePosition > 6'
RenegeDestination	*None*	EntityConveyor5
Spacing	0.0 m	
MaxPerLine	Infinity	

Input Editor - Queue2

Key Inputs | Graphics

Keyword	Default	Value
AttributeDefinitionList	*None*	
CustomOutputList	[]	FractionAdded '[Queue2].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded)' DimensionlessUnit
StateAssignment	*None*	
Priority	0.0	
Match	*None*	
FIFO	TRUE	
RenegeTime	*None*	75 min
RenegeCondition	1.0	'this.QueuePosition > 6'
RenegeDestination	*None*	EntityConveyor6
Spacing	0.0 m	
MaxPerLine	Infinity	

Figure 27. Renege and Baulk Settings.

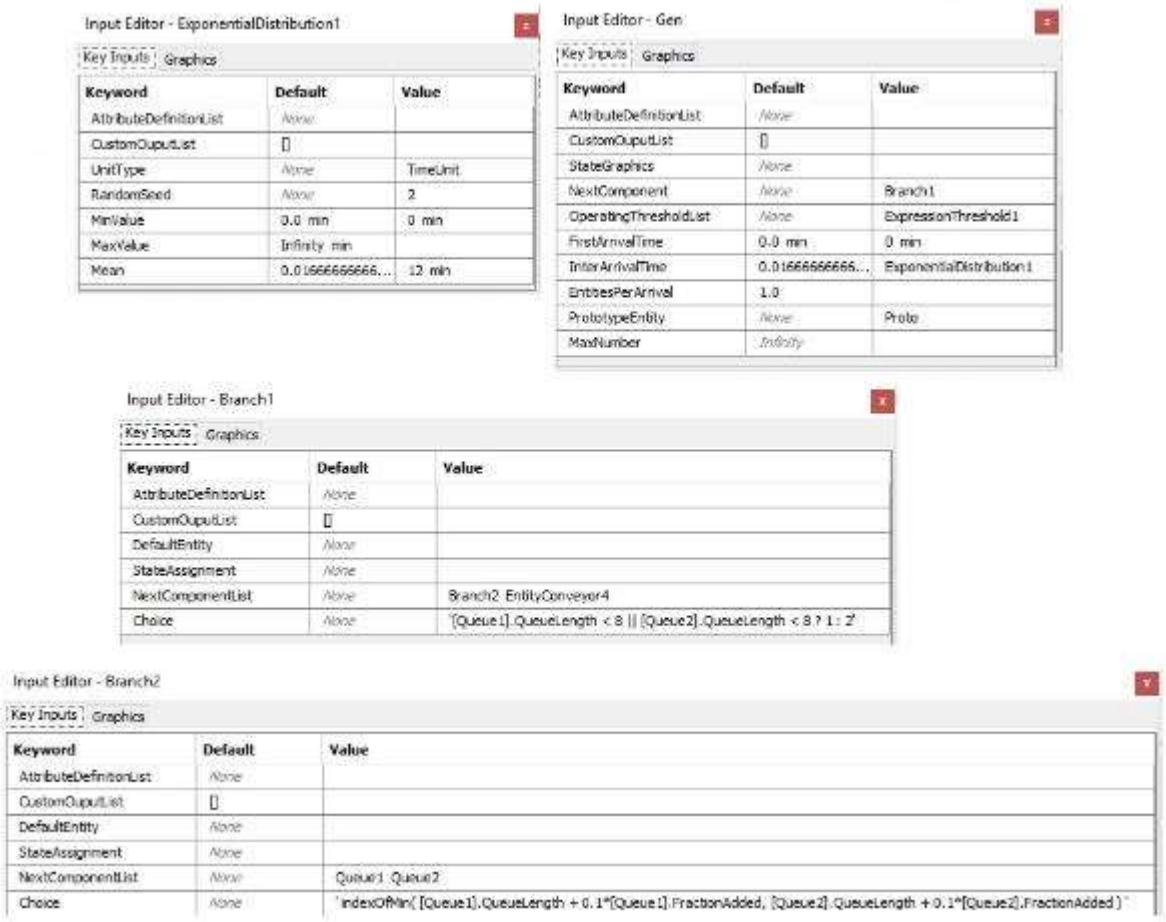


Figure 28. Renege and Baulk Settings.

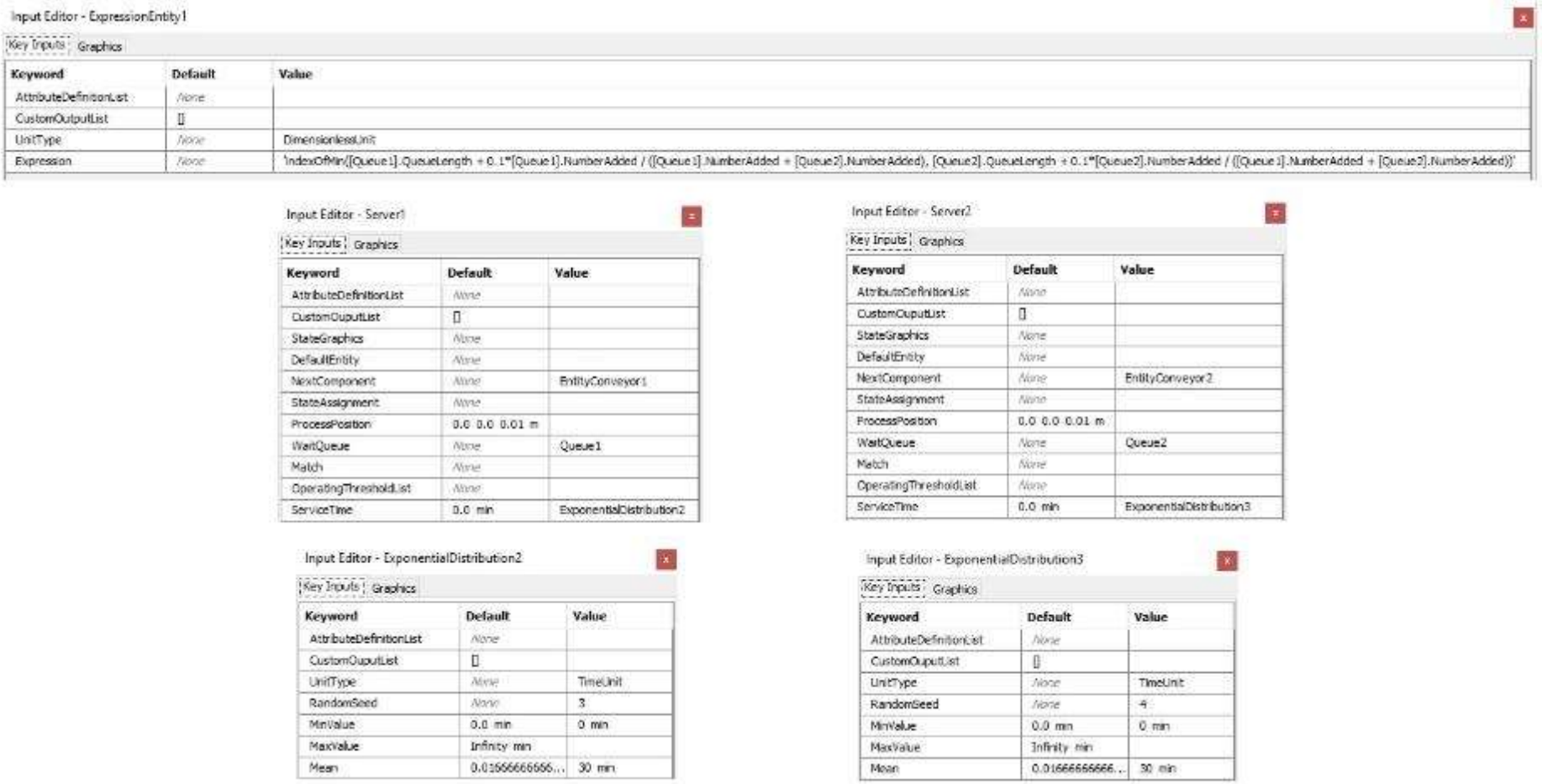


Figure 29. Renege and Baulk Settings.

13.1 Renege and Baulk Explanation.

13.1.1 Run Duration
Run Duration = 5000000 minutes

13.1.2 Generator (Gen)
The Generator is setup much as in the previous models

13.1.3 Exponential Distribution1
The Exponential Distribution is setup much as in the previous models

13.1.4 Branch1
Branch1 determines where items can either go to, Branch2 If Queue1 OR Queue2 are less than 8.

If Queue1 and Queue2 are equal or greater than 8, the items are passed to the Entity Conveyor4 and onwards to the Baulking EntitySink2.

NextComponentListBranch2 EntityConveyor4

Choice'[Queue1].QueueLength< 8 || [Queue2].QueueLength<8 ?1 : 2'

13.1.5 Queue1 and Queue2
When the queues are empty or of equal length the Branch will normally pass items to the first queue in the list - hence Queue1 would be favoured over Queue2.

I did not think this was realistic behaviour, if the queues are empty or equal in length I would expect items to join the queues over a period of time in an equal fashion.

Ok, I can hear you say that at Asda or Walmart, if Queue1 is nearer the exit, then Queue1 might be favoured, or for some other reason one queue might be favoured over the other. This could only be determined by observing a real queue.

For this exercise we will assume the queues are on average favoured equally.

13.1.6 CustomOutputList

The following is the message given by JaamSim.

Keyword	Description
CustomOutputList	The list of user defined custom output for this entity. The output name is followed by an expression to be evaluated as the output and the unit type of the expression. Examples: Queue1 CustomOutputList {{ TwiceSimTime '2*this.Sim.Time' TimeUnit } { CargoVolume 'this.Cargo/this.CargoDensity' VolumeUnit }}

Avoid using a custom output in the definition of another custom output, if you do you will likely get an error message at this present time 29.05.2016.

Queue1 we have:

FractionAdded '[Queue1].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded)' DimensionlessUnit

It's the fraction of items added to queue1 out of the total added to both queues:-

$$Fraction\ Added = \frac{Number Added\ to\ Queue\ 1}{Total\ number\ Added}$$

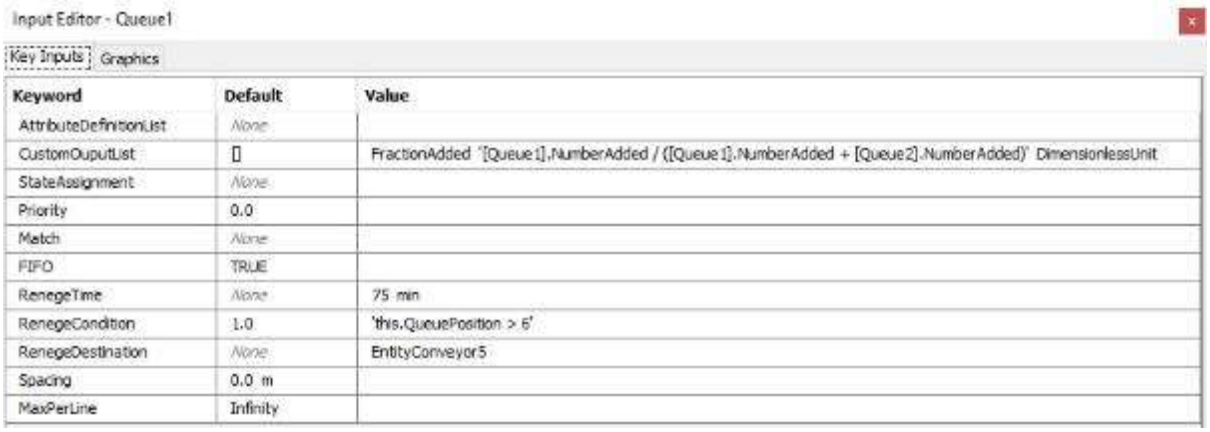


Figure 30. Queue1

Queue2 we have:

FractionAdded '[Queue2].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded)' DimensionlessUnit

It's the fraction of items added to queue2 out of the total added to both queues:-

$$Fraction\ Added = \frac{Number Added\ to\ Queue\ 2}{Total\ number\ Added}$$

Input Editor - Queue2

Key Inputs | Graphics

Keyword	Default	Value
AttributeDefinitionList	None	
CustomOutputList	[]	FractionAdded '[Queue2].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded)' DimensionlessUnit
StateAssignment	None	
Priority	0.0	
Match	None	
FIFO	TRUE	
RenegeTime	None	75 min
RenegeCondition	1.0	'this.QueuePosition > 6'
RenegeDestination	None	EntityConveyor6
Spacing	0.0 m	
MaxPerLine	Infinity	

Figure 31. Queue2

We did this because the value of Fraction Added will be used in Branch 2, and it helps to break down what would have been a very lengthy expression.

(See Branch2)

13.1.7 Renege

The following is the message given by JaamSim.

Keyword	Description
RenegeDestination	The object to which an entity will be sent if it reneges Examples Queue1 RenegeDestination { Branch1 }
RenegeCondition	A logical expression that determines whether an entity will renege after waiting for its RenegeTime value. Note that TRUE and FALSE are entered as 1 and 0, respectively. A constant value, a distribution to be sampled, a time series, or an expression can be entered. Examples Queue1 RenegeCondition {1} Queue1 RenegeCondition { 'this.QueuePosition> 1' } Queue1 RenegeCondition { 'this.QueuePosition> [Queue2].QueueLength' }
RenegeTime	The time an entity will wait in the queue before deciding whether or not to renege. Evaluated when the entity first enters the queue. A constant value, a distribution to be sampled, a time series, or an expression can be entered. Examples Queue1 RenegeTime { 3.0 h } Queue1 RenegeTime { NormalDistribution1 } Queue1 RenegeTime { '1[s] + 0.5*[TimeSeries1].PresentValue' }

13.1.8 Queue1 and Queue2

RenegeDestination

Queue1 EntityConveyor5 and onwards to EntitySink3 where the items are disposed of.

Queue2 EntityConveyor6 and onwards to EntitySink4 where the items are disposed of.

RenegeCondition

'this.QueuePosition> 6' - any item in position 7 or 8 is evaluated for reneging in accordance with the RenegeTime value.

QueuePosition can be seen in the OutputViewer for that queue.

RenegeTime

75 min - Any item that has been in the queue according to the RenegeCondition above for 75 min or more will renege.

13.1.9 Branch2

The original Choice expression for Branch2 without using the **CustomOutputList**in queue1 and queue2 would look like this:

'indexOfMin([Queue1].QueueLength + 0.1*[Queue1].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded), [Queue2].QueueLength + 0.1*[Queue2].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded))'

This is rather cumbersome?

CustomOutputList has other advantages since when the model is running we can see in the Output Viewer the values of Fraction Added for either Queue at any point in time, allowing you to evaluate whether the expression is behaving as expected.

We need the expression below in Branch2 as a tie breaker when the queues are empty or equal but less than 8 so that the items are allocated in an even manner.

' indexOfMin([Queue1].QueueLength + 0.1*[Queue1].FractionAdded, [Queue2].QueueLength + 0.1*[Queue2].FractionAdded) '

Items will be sent to the Queue that returns the lowest value.

13.1.10 Fraction Added - How does it work?

The Fraction Added is a measure of how many items have been through each queue to date, added to the length of the queue, we can then compare the value of each expression to see which has the least items allocated using the 'indexOfMin'.

We then allocate the next item to the smallest valued queue in the expression.

' indexOfMin([Queue1].QueueLength + 0.1*[Queue1].FractionAdded, [Queue2].QueueLength + 0.1*[Queue2].FractionAdded) '

The 0.1 factor - I was advised to include this. I tried running the model with and without this factor, and it did not seem to make any difference to the behaviour of the model.

"The factor of 0.1 is needed to make sure that NumberAdded is only used as a tie-breaker when it is equal to 1.0".

13.1.11 ExpressionEntity

When I was building the model I wanted to see how the complicated expression behaved. At first I included a CustomOuputList using the FractionAdded from the CustomOutputList in Queue1 and Queue2, so that I could see the values returned.

This will work, but each time you save the model the configuration file is saved with the CustomOutputList's in the wrong order, and hence we get error messages next time the file is opened. This means you have to open the configuration file in Notepad and move them into an order that will work.

This can be avoided by not using a custom output in the definition of another custom output, if you do you will get an error message at this present time 29.02.2016.

Instead I included an ExpressionEntity with the full equation entered.

This allows me to see the **Value** returned by the expression in the output Viewer for the ExpressionEntity1 **(See Figure 32)**.

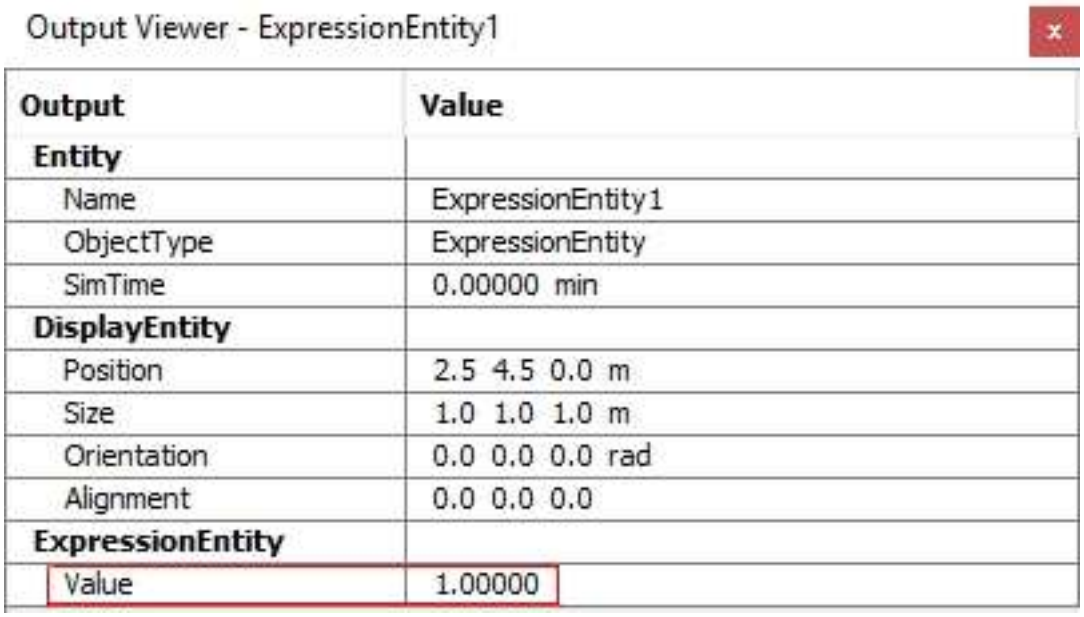


Figure 32. The Value from the expression in the Output Viewer.

The Expression:-

'indexOfMin([Queue1].QueueLength + 0.1*[Queue1].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded), [Queue2].QueueLength + 0.1*[Queue2].NumberAdded / ([Queue1].NumberAdded + [Queue2].NumberAdded))'

We could replace the expression in Branch2 Choice with:-

'[ExpressionEntity].Value'

I did not because I wanted to demonstrate the CustomOutputList feature.

13.1.12 Tracing the values of indexOfMin

Queue1 Number Added	Queue2 Number Added	Queue1 Fraction Added	Queue2 Fraction Added	Factor	Queue Length1	Queue Length2	Expression Value Queue1	Expression Value Queue2	Returned indexOfMin
1	0	1.00000	0.00000	0.1	0	0	0.100000	**0.000000**	**2**
1	1	0.50000	0.50000	0.1	0	0	**0.050000**	0.050000	**1**
1	0	1.00000	0.00000	0.1	0	0	0.100000	**0.000000**	**2**
2	1	0.66667	0.33333	0.1	0	0	0.066667	**0.033333**	**2**
2	2	0.50000	0.50000	0.1	0	0	**0.050000**	0.050000	**1**
2	2	0.50000	0.50000	0.1	1	0	1.050000	**0.050000**	**2**
3	3	0.50000	0.50000	0.1	1	0	1.050000	**0.050000**	**2**
3	3	0.50000	0.50000	0.1	0	0	**0.050000**	0.050000	**1**
135	119	0.53150	0.46850	0.1	8	8	8.053150	**8.046850**	**2**
1100	1095	0.50114	0.49886	0.1	6	8	**6.050114**	8.049886	**1**
8074	8068	0.50019	0.49981	0.1	8	6	8.050019	**6.049981**	**2**
27070	27096	0.49976	0.50024	0.1	8	7	8.049976	**7.050024**	**2**
60354	60355	0.50000	0.50000	0.1	8	8	**8.050000**	8.050000	**1**
122005	122073	0.49986	0.50014	0.1	0	0	**0.049986**	0.050014	**1**
180478	180412	0.50009	0.49991	0.1	8	7	8.050009	**7.049991**	**2**

If we run the model and pause it from time to time, we can calculate the values of the expression from the information in the Output Viewer for each of the queues. The Returned values **indexOfMin** are 1's and 2's, indicating which Branch will be chosen to send the items to, that is which Queue they will go to, Queue1 or Queue2.

13.1.13 Conveyors

Finally I added conveyors and set the travel time to 40 min so that I could observe the items and their behaviour.

13.2 Reflections of the Renege and Baulking Model

Although this has been a lengthy explanation about the model, this model has been well worth the effort. This has been the model that (for me) has helped to break through the wizardry of creating expressions and given me the chance to try out some new features of the modelling software.

Expressions are worth studying further to understand them, they are the key to creating more complicated models. Being from a Manufacturing Engineering background, some of the expressions used by some models rely on mathematical functions I would not normally use, they are perhaps more at home with the Software Engineer as we will see in the next example.

14.0 Batch Versus One Piece Flow

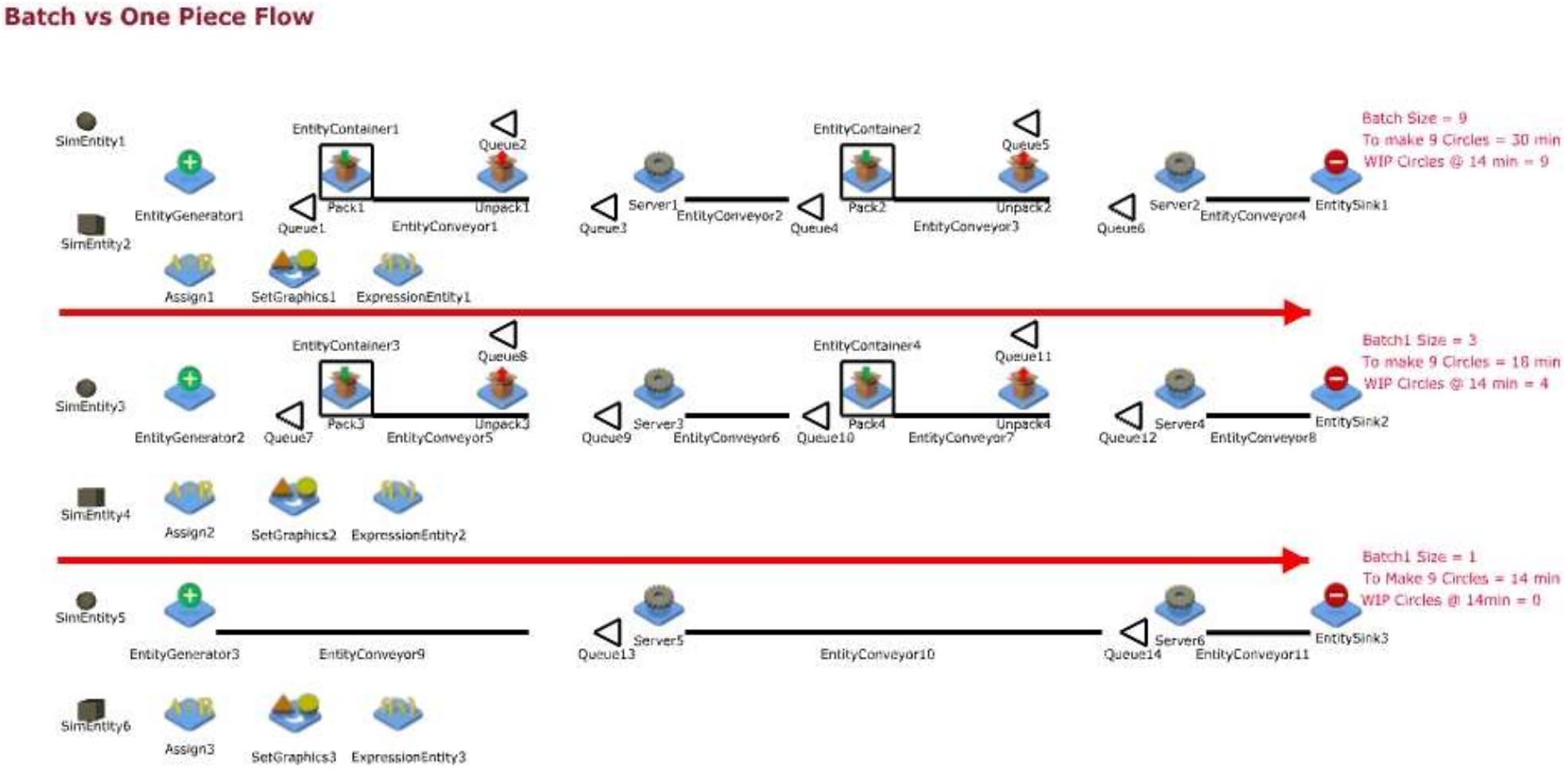


Figure 33. The Value from the expression in the Output Viewer.

14.1 Introduction - One Piece Flow line Versus Batch Production

This is quite a busy model, which compares One Piece Flow with Batches of 3 and 9 items. The flow lines alternate between the two different products being produced in quantities of 9 off each time.

It is quite a nice model, which introduces some new features of JaamSim namely:

Pack, Unpack, Containers, Assign and SetGraphics.

It also introduces some mathematical functions you may not be familiar with, we will discuss these shortly.

The model also demonstrates how you only need ONE EntityGenerator rather than TWO.

So here is your shopping list to build this model:-

Entity	**Quantity**
Queue	14
Server	6
Pack	4
Unpack	4
EntityContainer	4
EntityGenerator	4
Assign	3
SetGraphics	3
SimEntity	6
EntityConveyor	11
ExpressionEntity	3
EntitySink	3
Arrows (Optional)	2
Text	As Required

NOTE: do not place them all onto the screen at once, we will build the model up in stages and verify each stage works as we go along.

14.2 Creating the batch size of 9 Flow line

Drag and drop the required Entities for the flow line at the top of the screen for a batch size of 9 items.

Edit the Entities as follows:-

Entity Name	Keyword	Value
SimEntity1	AttributeDefinitionList	{ Type 1 }
SimEntity2		Nothing to Enter Here
EntityGenerator1	NextComponent	Assign1
	InterArrivalTime	1 min
	EntitiesPerArrival	1
	PrototypeEntity	SimEntity1
Assign1	NextComponent	SetGraphics1
	AttributeAssignmentList	'this.obj.Type = floor([Queue1].NumberAdded / 9) % 2+1'
SetGraphics1	NextComponent	Queue1
	GraphicsList	SimEntity1 SimEntity2
	Choice	this.obj.Type
ExpressionEntity1	UnitType	DimensionlessUnit
	Expression	'floor([Queue1].NumberAdded / 9) % 2+1'
Queue1/3/4/6/	MaxPerLine	1
Queue2/5		Nothing to Enter Here
Pack1	NextComponent	EntityConveyor1
	WaitQueue	Queue1
	PrototypeEntityContainer	EntityContainer1
	NumberOfEntities	9
EntityContainer1	MaxPerLine	3
EntityConveyor1	NextComponent	Unpack1
	TravelTime	1 min
Unpack1	NextComponent	Server1
	WaitQueue	Queue2

Creating the Batches of 9 Flow line Continued....

Entity Name	Keyword	Value
Server1	NextComponent	EntityConveyor2
	WaitQueue	Queue3
	ServiceTime	1 min
EntityConveyor2	NextComponent	Pack2
	TravelTime	1 min
Pack2	NextComponent	EntityConveyor3
	WaitQueue	Queue4
	PrototypeEntityContainer	EntityContainer2
	NumberOfEntities	9
EntityContainer2	MaxPerLine	3
EntityConveyor3	NextComponent	Unpack2
	TravelTime	1 min
Unpack2	NextComponent	Server2
	WaitQueue	Queue5
Server2	NextComponent	EntityConveyor4
	WaitQueue	Queue6
	ServiceTime	1 min
EntityConveyor4	NextComponent	EntitySink1
	TravelTime	1 min

You will see that the queues and the containers have had their colours changed, we can explore changing these later.

14.2.1 SimEntity1

SimEntity1 is given the attribute{ Type 1 }

14.2.2 Assign1

Assign allows you to give attributes to objects received by the Assign Entity, in this case we have Assigned a value to each SimEntity (this.obj.Type), a value of 1 or 2.

'this.obj.Type = floor([Queue1].NumberAdded / 9) % 2+1'

Let's see how this expression works:-

Floor is a mathematical function which returns the largest (closest to positive infinity) integer that is less than or equal to the argument.

Looking at the value in the ExpressionEntity when the NumberGenerated is zero.

$$\frac{NumberGenerated}{9} = \frac{0}{9} = 0$$

In this case the argument is 0, hence it returns 0.

$$\frac{NumberGenerated}{9} = \frac{3}{9} = 0$$

In this case the argument is 0.3333, hence it returns 0.

When the NumberGenerated is 9 we get:-

$$\frac{NumberGenerated}{9} = \frac{9}{9} = 1$$

In this case the argument is 1, hence it returns 1.

$$\frac{NumberGenerated}{9} = \frac{14}{9} = 1$$

In this case the argument is 1.556, hence it returns 1.

Modulus % (remainder) operator. Used as an operator, not a function.

If you divide 0 by 2, there is no remainder hence

$$Mod\left|\frac{0}{2}\right| = 0$$

However

$$Mod\left|\frac{1}{2}\right| = 1$$

Since 2 will not divide exactly into 1, the remainder is 1

Adding 1 to the values returned by the Modulus operation we get 1's and 2's which is assigned to the SimEntity.

(See Table Tracing 'this.obj.Type = floor([Queue1].NumberAdded / 9) % 2+1')

SetGraphics The value 1 or 2 Assigned by the Assign Entity will determine whether a Circle or a Cube is visible. Hence we can change the items being manufactured.

SimEntity1 (Circle) = 1; SimEntity2 (Cube) = 2.

Number Generated	Number Generated / 9	Floor	Constant	Modulus % 2	+1	SimEntity	Number of SimEntities
0	0.000	0	2	0	1	Circle	1
1	0.111	0	2	0	1	Circle	2
2	0.222	0	2	0	1	Circle	3
3	0.333	0	2	0	1	Circle	4
4	0.444	0	2	0	1	Circle	5
5	0.556	0	2	0	1	Circle	6
6	0.667	0	2	0	1	Circle	7
7	0.778	0	2	0	1	Circle	8
8	0.889	0	2	0	1	Circle	9
9	1.000	1	2	1	2	Cube	1
10	1.111	1	2	1	2	Cube	2
11	1.222	1	2	1	2	Cube	3
12	1.333	1	2	1	2	Cube	4
13	1.444	1	2	1	2	Cube	5
14	1.556	1	2	1	2	Cube	6
15	1.667	1	2	1	2	Cube	7
16	1.778	1	2	1	2	Cube	8
17	1.889	1	2	1	2	Cube	9

Table Tracing 'this.obj.Type = floor([Queue1].NumberAdded / 9) % 2+1')

14.2.3 SetGraphics1

The SetGraphic object is used to change the graphical appearance of a specified entity. There are two Entities defined, SimEntity1 (Circle) and SimEntity2 (Cube).

If the value of the expression in the Assign Entity is equal to 1, then SimEntity1 (Circle) is used, if the expression is equal to 2, then SimEntity2 (Cube) is used

14.2.4 Container

The EntityContainer object is used to store a collection of entities for subsequent processing as a unit. A container is used to Pack items into, and could represent a Palette, a Truck, a Bin, or some other container used to hold a number of items. In our case it will hold a Batch size of 9 items.

14.2.5 Pack

The Pack object is used to generate EntityContainers and fill them with incoming entities. This “does what it says on the tin”, it packs the items into the Container.

14.2.6 Unpack

The Unpack object is used to remove the entities from incoming EntityContainers. Again it “does what it says on the tin” and passes the items onto the next Entity.

There are no other Entities or expressions in this part of the model we have not covered.

Complete the flow line for a batch size of 9 and make sure it works before moving onto create the next flow line for a batch of 3.

14.2.7 Creating the Batches of 3Flow line

We create the flow line for a batch size of 3 in the same way.

Do this and verify it works before moving on.

14.3 Creating the One Piece Flow Line

We create the One Piece Flow line in the same way, but since we are not passing batches greater than 1 from one Entity to another, there is no need for a Container, Pack and Unpack Entity.

How long does it take each flow line to process 9 components?

Change the Batch sizes to 6 and 12 and see the differences.

15.0 Further Models

15.1 Simple Pull System

A Manufacturing System in which production is based on actual daily demand (sales), and where information flows from market to management in a direction opposite to that in traditional (push) system.

http://www.businessdictionary.com/definition/pull-system.html

No parts are produced without an order for them, which is sent by a customer invoice. Since orders are unlikely to arrive at a constant rate I have assumed that their arrival times are exponentially distributed, however the manufacturing times are considered constant with one piece flow.

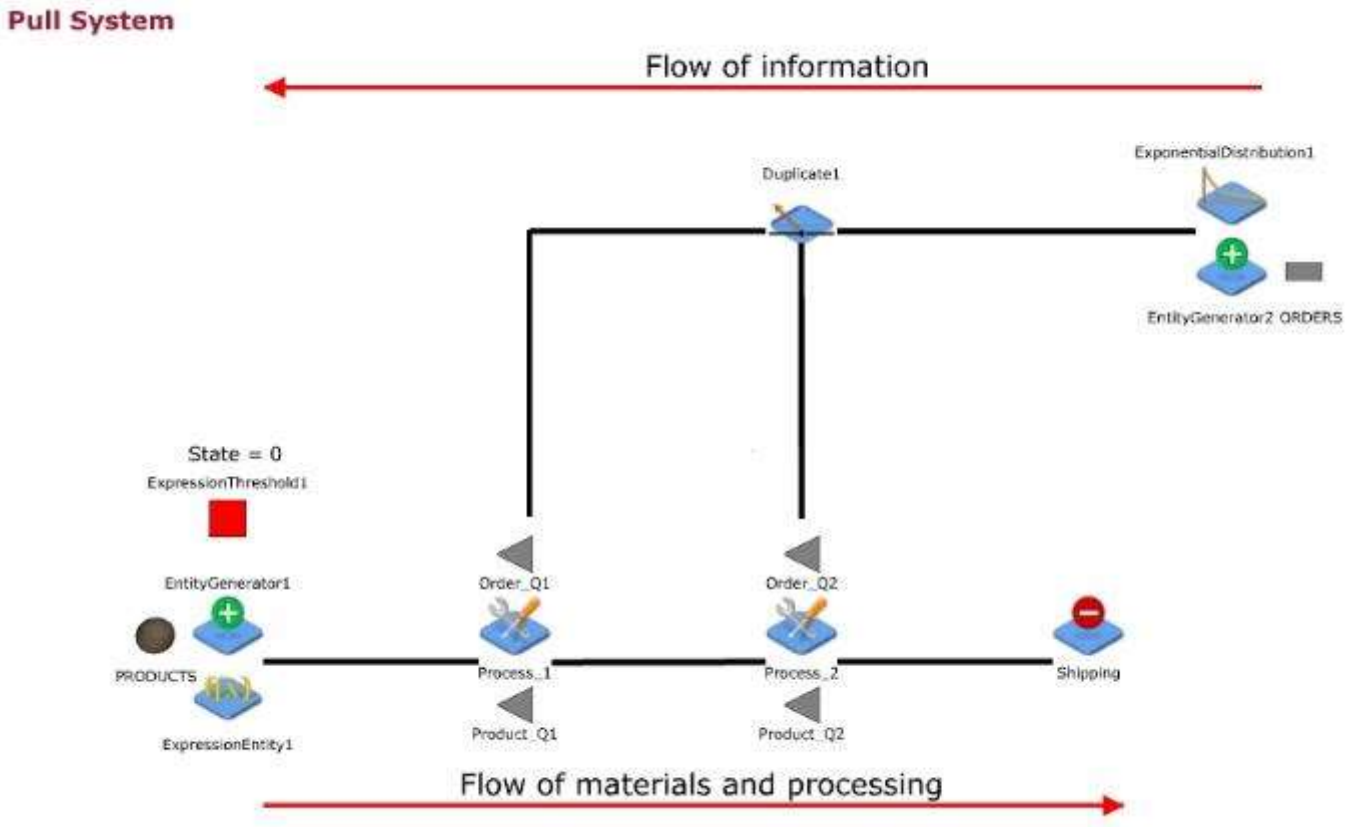


Figure 34 Simple Pull System

You should by now be able to build a model to represent this simple Pull System.

Try it out.

16.0 Changing the graphics

You can change the default images for the entities so that they represent the machines and items you are modelling. See the JaamSim Manual. However you have to create and import the images you want to use.

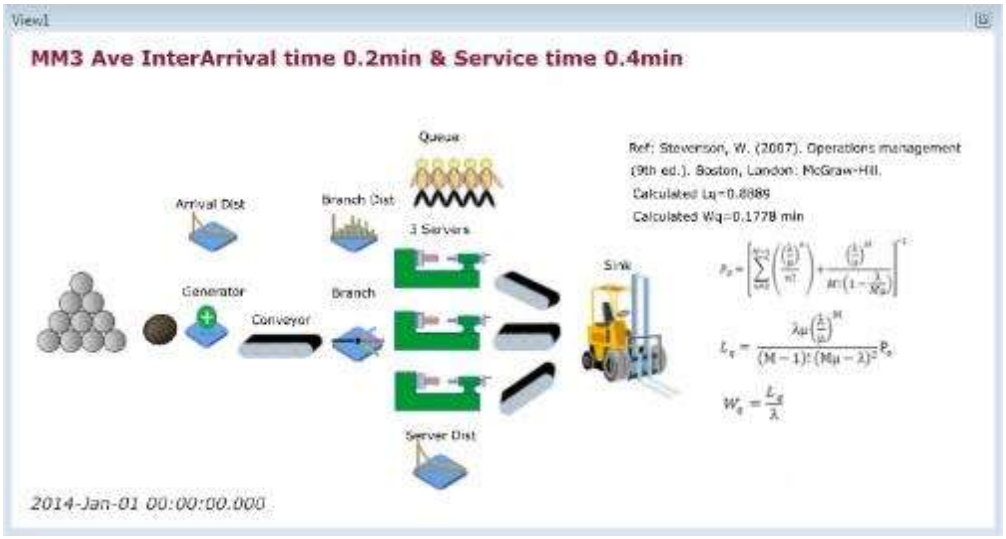
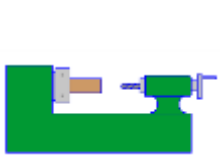
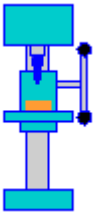


Figure 35. Changing the Graphics

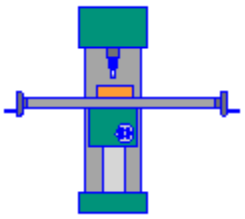


Lathe

Press



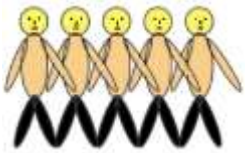
Drill



Mill

Generator

Conveyor



Queue

Sink

Some of these are animated Gifs, I created for another project I worked on a few years ago and have come in handy for this. If you get these from me do not expect them to animate in JaamSim, they will not, and they would slow the program down anyway, so there is not much point.

16.1 Change the entity colour based on the attribute value assigned by an assign module. By Harry King.

If your object is an imported image or 3D asset, then you must modify the colour externally and re-import separate versions. The SetGraphics object was intended to work with imported images, but it can work with other types of DisplayEntity.

If your object is one of the simple flat objects such as Rectangle or Circle, you can edit the colour through the Input Editor but you still need to create new DisplayEntities -- one for each colour. The following steps outline the procedure:

1. Find the shape you want in the Object Selector under DisplayModels>ShapeModel. The only ones you can use in this way are Rectangle, Circle, Arrow2D, and Triangle.

2. Right-click on the object and select "Duplicate"

3. Rename the duplicated object to something you can remember, say "Rectangle-Green"

4. Click on the renamed object in the Object Selector and select the Key Inputs tab in the Input Editor

5. Set the input for the FillColour keyword either by entering a named colour (e.g. green) from Section 14 of the User Manual or by selecting an RGB value using the drop down menu. There are other graphical options you can also set.

6. Drag and drop a new DisplayEntity from the Model Builder

7. Click on the new DisplayEntity and select the Graphics tab in the Input Editor

8. Click on the drop-down menu for the DisplayModel keyword and select the new ShapeModel you created (e.g. Rectangle-Green)

9. The DisplayEntity should now adopt the shape and colour settings you have made.

Taken from a post in the JaamSim Forum, a reply from Harry King.

"Change entity color based on entity attribute value" 30.03.2016

17.0 Example of selected outputs for multiple runs summarized in a single table using the RunOutputList keyword - By Harry King. http://jaamsim.com/

This is a very important new feature that allows a large number of simulation runs to be performed and their key outputs summarized in a single table.

The example used was a simple server and queue model with a run duration of one day. Ten replications are made by setting `RunIndexDefinitionList` and `EndingRunNumber` inputs to 10, and setting the `GlobalStreamSeed` input to `this.RunNumber`. The `RunOutputList` input is set to generate outputs for the run number, the maximum queue length, the average queue length, and the average queue time. The main output report is turned off by letting the `PrintReport` input default to FALSE.

With these inputs, a single output report "RunOutputList Example.dat" was generated that contains the following text:

[Simulation].RunNumber	[Queue1].QueueLengthMaximum	[Queue1].QueueLengthAverage	[Queue1].AverageQueueTime
			s
1	27	2.509058050	2.498013591
2	29	2.321671146	2.32630221
3	30	2.529669508	2.530343095
4	28	2.325720433	2.329630113
5	26	2.372417501	2.380186165
6	33	2.541593983	2.527173882
7	28	2.385389889	2.387738937
8	26	2.579446826	2.560921641
9	27	2.300238063	2.315675903
10	26	2.395205723	2.392353727

Taken from a post in the JaamSim Forum by Harry King.

"Example of selected outputs for multiple runs summarized in a single table using the RunOutputList keyword" 06.04.2016

17.1 RunOutputList Example by Harry King http://jaamsim.com/

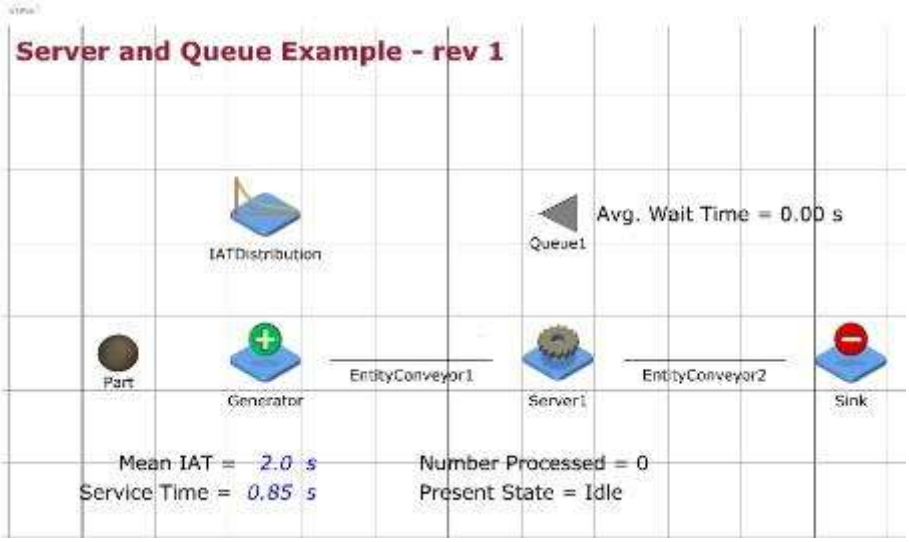


Figure 36a RunOutputList

Input Editor - Simulation

Key Inputs | Multiple Runs | GUI

Keyword	Default	Value
RunIndexDefinitionList	1	10
StartingRunNumber	1	1
EndingRunNumber	1	10

Figure 36b RunOutputList

Input Editor - Simulation

Key Inputs | Multiple Runs | GUI

Keyword	Default	Value
AttributeDefinitionList	None	
CustomOutputList	[]	
RunDuration	3.1536E7 s	1 d
InitializationDuration	0.0 s	
PauseCondition	None	
ExitAtPauseCondition	FALSE	
ExitAtStop	FALSE	
GlobalSubstreamSeed	0.0	this.RunNumber
PrintReport	FALSE	
ReportDirectory	Configuration File	
UnitTypeList	None	DimensionlessUnit DimensionlessUnit DimensionlessUnit TimeUnit
RunOutputList	None	{ [Simulation].RunNumber } { [Queue1].QueueLengthMaximum } { [Queue1].QueueLengthAverage } { [Queue1].AverageQueueTime }
TickLength	1.0E-6 s	

Figure 36c RunOutputList

18.0 Hints and Tips

18.1 Hints and Tips 1 - Saving

It is good practice to save your model on a regular basis.

18.2 Hints and Tips 2 - Backup Copies

Make backup copies of your models JaamSim does not do this automatically. I accidentally over wrote one of these models.

18.3 Hints and Tips 3 - Simulation times

When modelling using a Simulation, it can take a number of samples for the model to settle down into a steady state. Therefore the number of samples is important to ensure the results you get are valid. A better result might be obtained by running the simulation for longer.

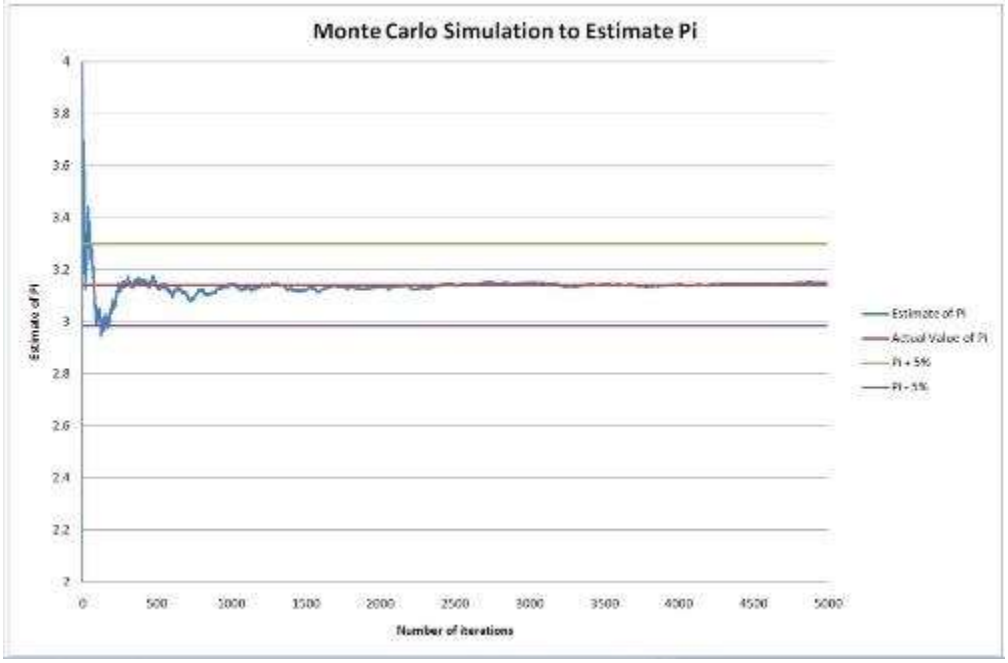


Figure 37. Monte Carlo Simulation to estimate the value of Pi

For example **Figure 37** shows a simulation to estimate the value of Pi, as can be seen the simulation does not settle to a steady state until about 2500 samples.

18.4 Hints and Tips 4 - Anti-Virus programmes

I use F-Secure as my Anti-Virus programme. Within F-Secure is an option called DeepGuard, if this is turned on, you will find that JaamSim takes 2 minutes to load. This is because DeepGuard is doing its job and protecting your computer by monitoring the activities of JaamSim. If you turn DeepGuard off, JaamSim will load faster.

19.0 Equation nomenclature

λ	=	The average arrival rate
μ	=	The average service rate
ρ	=	Utilisation factor, the probability the server is busy
Ls	=	The average number of items in the system
Lq	=	The average number of items in the queue
Ws	=	The average time spent in the system
Wq	=	The average time spent in the queue
Po	=	Probability of zero customers in the system
K	=	Number of items in the system of a finite queue.
M	=	Number of servers

20.0 Other useful open source OR free software

I use the following programs on a regular basis and would recommend them.

20.1 XMind

Mind mapping, Flow charts, Cause and effect diagrams and more.
http://www.xmind.net/

20.2 Geogebra

Dynamic Geometry and Algebra. This might be useful for analysing results, it has too many functions and applications to explain here. It is a mathematics program which will allow animation, plotting of mathematical functions, it also has a statistics capability and much, much more.
http://www.geogebra.org/

20.3 LibreCAD

A nice 2D CAD package, I used it to draw graphics to include in my models, which I then rendered (coloured in) using a programme similar to Paint
http://librecad.org/cms/home.html

20.4 Artweaver

A painting program to render your graphics.
http://www.artweaver.de/en

20.5 Debut Video Capture software

Record video from a webcam, recording device or screen.
A free version of Debut video capture software is available for Non-Commercial use only. Capture your video from the screen and edit the resulting AVI file using VideoPad.
http://www.nchsoftware.com/capture/

20.6 VideoPad Video Editor

A fully featured video editor for creating professional quality videos in minutes. VideoPad will allow you to save in MP4 format and others.
Free download for Non-Commercial use only.
http://www.nchsoftware.com/videopad/

20.7 Handbrake

Used for converting the .AVI video file created by JaamSim to .MP4 format.
https://handbrake.fr/

21.0 Bibliography

Chase, R., and Aquilano, N. (1991).Production and operations management: A life cycle approach (6th Ed.). Irwin.

Hillier, F., & Lieberman, G. (2004).Introduction to operations research (8th ed.; International Ed.). Boston, Mass.; London: McGraw-Hill.

JaamSim User manual Software Version: 2016-01January 20, 2016 -

http://jaamsim.com/

Render, Barry, Ralph M. Stair, and Michael E. Hanna. Quantitative Analysis for

Management. Upper Saddle River, NJ: Upper Saddle River, NJ: Prentice Hall, 2006.

Stevenson, W. (2007). Operations management (9th Ed.). Boston (Mass), London:

McGraw-Hill.

http://homepages.cae.wisc.edu/~ie642/content/Applications/1-

FlowLines/FlowLines2.htm

http://thebigleansimulationlibrary.com/

Formulae sheet

http://profs.degroote.mcmaster.ca/ads/parlar/courses/o711/documents/Formulas-HL-9e-2011-10-31.pdf

Lecture series on Advanced Operations Research by Prof. G. Srinivasan, Department of Management Studies, IIT Madras

Proofs of some of the queuing models

Queuing Models

https://www.youtube.com/watch?v=xGkpXk-AnWU

Single Server Queuing Models

https://www.youtube.com/watch?v=2aPlzhsEsIw

Multiple Server Queuing Models

https://www.youtube.com/watch?v=PavZX3hAL6I

Classification of Queues

https://www.andrewferrier.com/oldpages/queueing_theory/Andy/kendall.html

http://nptel.ac.in/courses/117103017/5