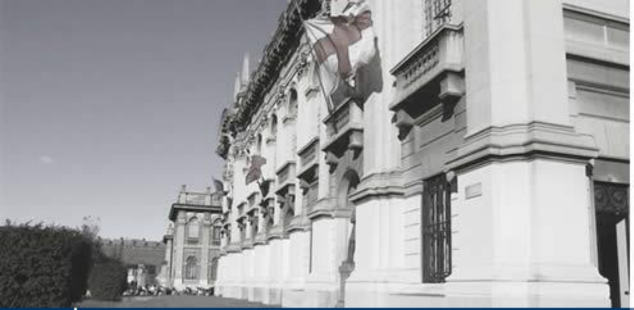


Performance Evaluation and Applications

POLITECNICO DI MILANO

Closed models

Motivating example

An ***automated warehouse*** has 3 robots that takes the goods from the shelves to the delivery area. Each robot has its own charging station, that requires a total of 30min to fully recharge its battery. Only one robot at a time can operate either in the shelves or delivery area, requiring respectively 6min and 4min to complete their tasks. Travelling between the zones can be done in parallel, and takes an average of 2min. Each robot has to return to the charging station on the average every 25 deliveries.

Management is interested in determining:

1. Average time between recharges for each robot.
2. System throughput (how many goods are delivered per hour).

The response time law

In time-sharing systems, the "think time" is usually not considered in the system response time.

$$R_{Tot} = R_{Sys} + Z \qquad N = X \cdot R_{Tot} \qquad N = X \cdot \left(R_{Sys} + Z\right)$$

In particular, Little's law becomes the so-called "*Response Time Law*" which explicitly excludes the think time from the usual response time.

The Response Time Law: $R = \dfrac{N}{X} - Z$

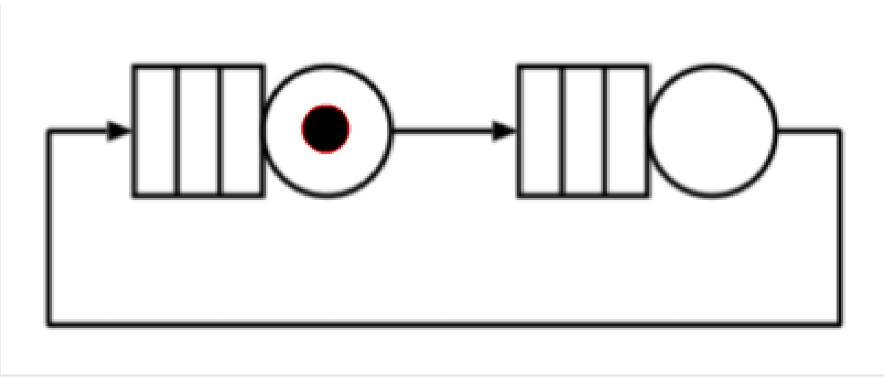
Analysis of closed models

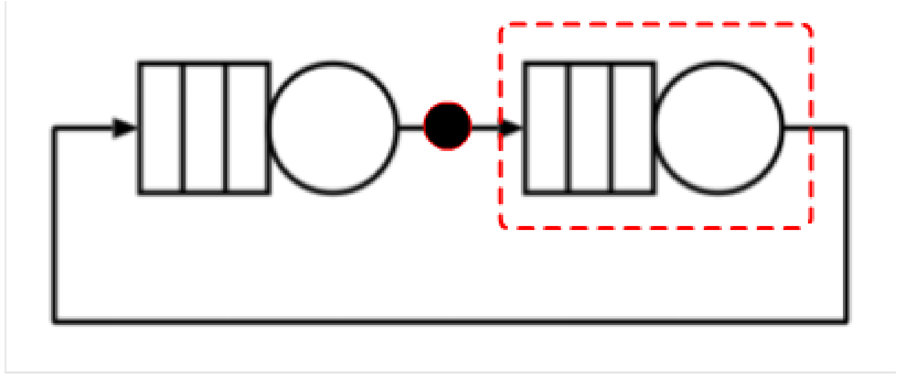
As for separable open queueing network models, the solution of closed ones is based on the computation of $A_k(N)$, the average number of jobs at the arrival.

However, the expression of $A_k(N)$ in *closed models* is different from the one used in open models.

If we consider a network with two stations, identical demand and a single job, we can see immediately that the average number of jobs is equal to one half: $N_k(N) = 0.5$.

However, since there is just one job, the number of jobs that are found when the costumer enters a station is always $A_k(1) = 0$.





Analysis of closed models

In closed models we can use the *"Arrival Theorem"* which states *that the number of jobs that a costumer finds in the queue at its arrival is equal to the average queue length of the system with one less job*.

$$A_k(N) = N_k(N-1)$$

Mean Value Analysis (MVA)

The performance indices can then be computed in an iterative way, starting from an empty system and adding one job per iteration.

This technique is called "*Mean Value Analysis*" (*MVA*).

Let us imagine that the system has been studied up to a workload of *N-1* jobs. Using the arrival theorem we can determine the residence time at each station when the population increases of one job (*N* jobs).

$$\mathrm{R}_k(N) = (1 + A_k(N)) \cdot D_k = \big(1 + N_k(N-1)\big) \cdot D_k$$

Mean Value Analysis (MVA)

Summing up the residence time at all the stations, we can determine the *system response time*.

$$R(N) = \sum_k \mathrm{R}_k(N)$$

Inverting the *response time law*, the system throughput can be determined.

$$R(N) = \frac{N}{X(N)} - Z$$

$$\Downarrow$$

$$X(N) = \frac{N}{R(N) + Z}$$

Mean Value Analysis (MVA)

Using *Little's law*, the queue length of each station can finally be determined.

$$N_k(N) = X(N) \cdot R_k(N)$$

The algorithm can then increase the population to *N+1* jobs, since for the arrival theorem we have that $A_k(N+1) = N_k(N)$, and the process can be repeated.

$$A_k(N+1) = N_k(N)$$

$$\mathrm{R}_k(N+1) = \left(1 + A_k(N+1)\right) \cdot D_k = \left(1 + N_k(N)\right) \cdot D_k$$

Mean Value Analysis (MVA)

The starting point considers an empty system.

$$N_k(0) = 0$$

$$A_k(1) = N_k(0) = 0$$

$$\mathrm{R}_k(1) = \big(1 + A_k(1)\big) \cdot D_k = D_k$$

Mean Value Analysis (MVA)

To summarize we have:

for $k \leftarrow 1$ to K do $Q_k \leftarrow 0$

for $n \leftarrow 1$ to N do

begin

for $k \leftarrow 1$ to K do $R_k \leftarrow \begin{cases} D_k & \text{(delay centers)} \\ D_k(1 + Q_k) & \text{(queueing centers)} \end{cases}$

$$X \leftarrow \frac{n}{Z + \sum_{k=1}^{K} R_k}$$

for $k \leftarrow 1$ to K do $Q_k \leftarrow XR_k$

end

Mean Value Analysis complexity

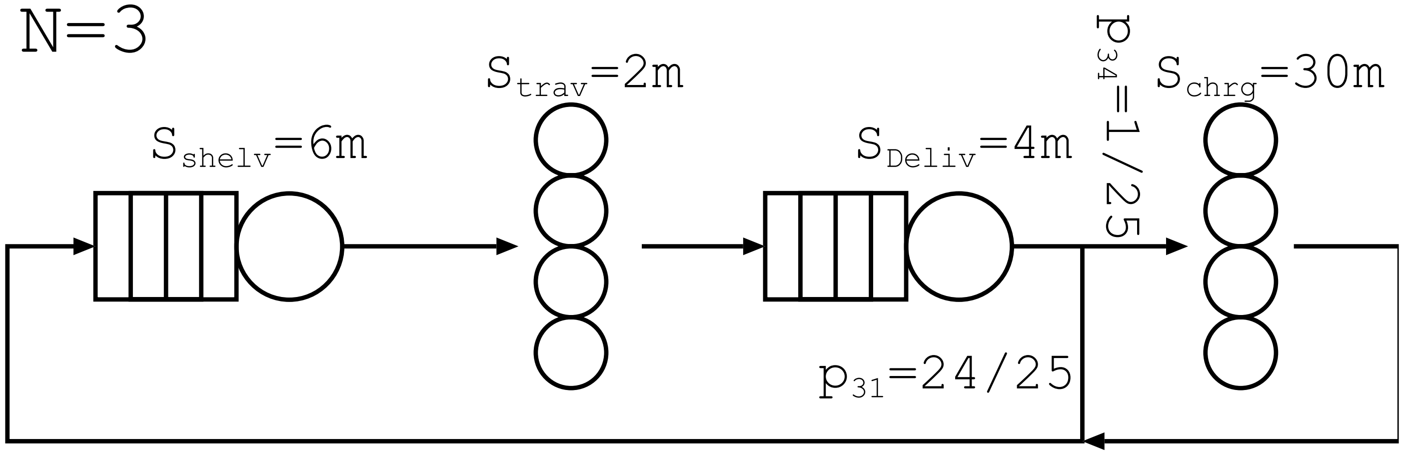
The mean value analysis computes the solution of a model with complexity $O(N \cdot K)$.

However, as a by-product, it computes the solution for the models with population sizes *1* to *N-1*.

This can be particularly effective when performing sizing studies, where the evolution of the performance against the population is required.

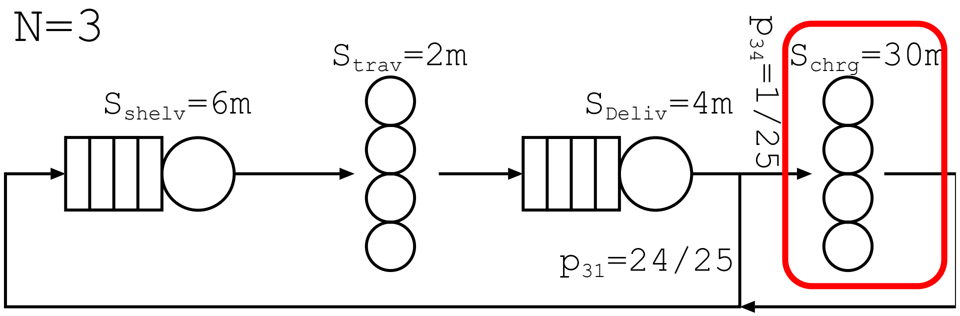
Analysis of Motivating Example

The considered automated warehouse can be modelled with a closed queuing model, with N=3 jobs circulating inside.



Analysis of Motivating Example

To determine the average time between charges, the model is a time-sharing system where station 4 is both the reference and the terminal station, and we compute the system response time with MVA.

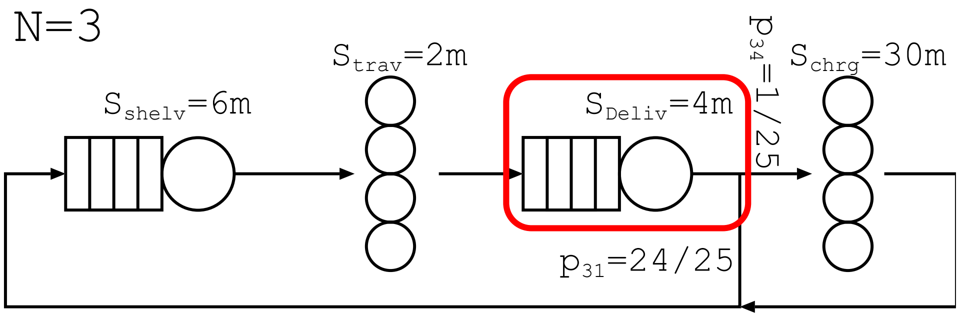


$$\begin{cases} v_1 = 25 \\ v_2 = 25 \\ v_3 = 25 \\ v_4 = 1 \end{cases} \qquad \begin{cases} D_1 = 150m \\ D_2 = 50m \\ D_3 = 100m \\ D_4 = Z = 30m \end{cases}$$

D	150	50	100		Z	30		
N	R1	R2	R3	R	X	N1	N2	N3
0						0	0	0
1	150	50	100	300	0,0030303	0,45454545	0,15151515	0,3030303
2	218,181818	50	130,30303	398,484848	0,00466761	1,01838755	0,23338048	0,60820368
3	302,758133	50	160,820368	513,578501	0,00551898	1,67091671	0,2759491	0,88756473
				8,55964168	H			

Analysis of Motivating Example

To determine the system throughput, the model is a batch system where station 3 is the reference station. With MVA we determine X.

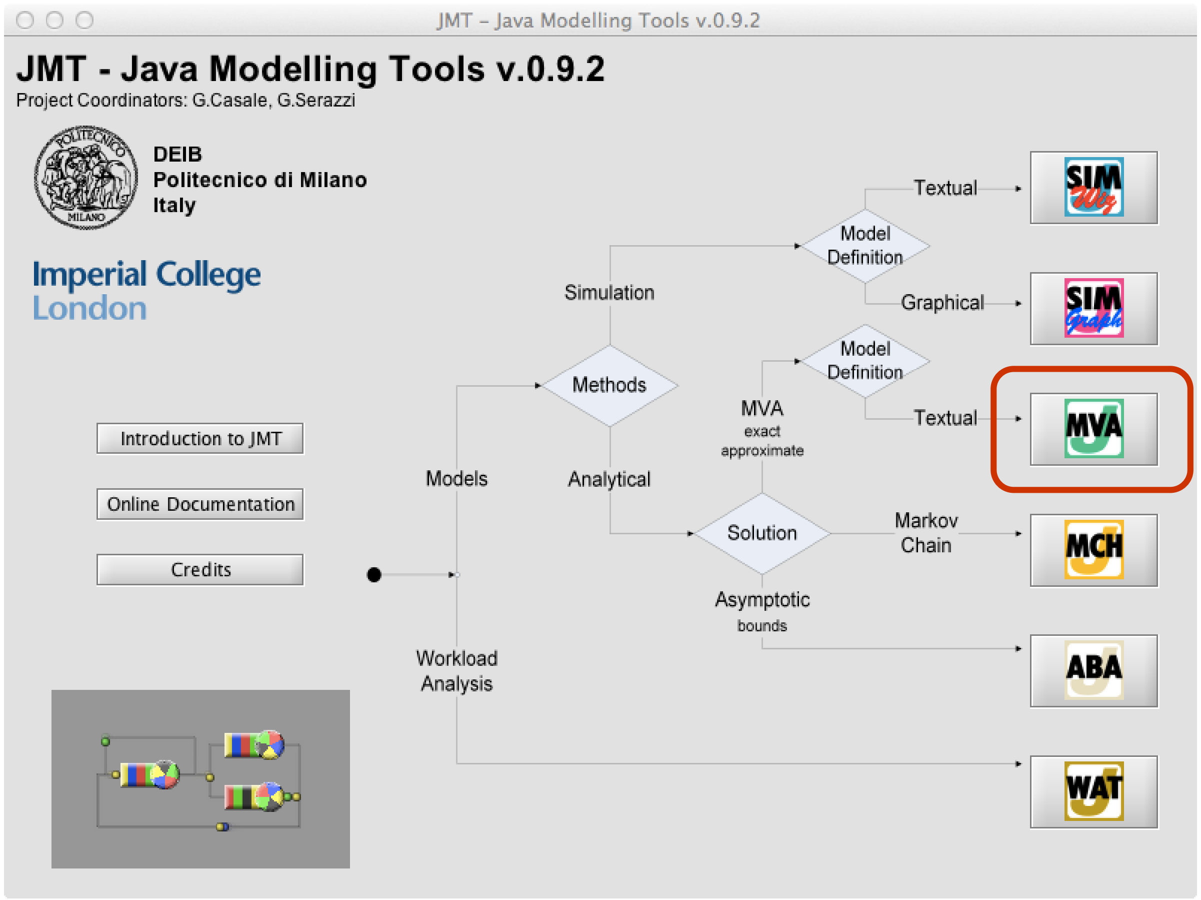


$$\begin{cases} v_1 = 1 \\ v_2 = 1 \\ v_3 = 1 \\ v_4 = 0.04 \end{cases} \qquad \begin{cases} D_1 = 6m \\ D_2 = 2m \\ D_3 = 4m \\ D_4 = 1.2m \end{cases}$$

D	6	2	4	1,2						
N	R1	R2	R3	R4	R	X	N1	N2	N3	N4
0							0	0	0	0
1	6	2	4	1,2	13,2	0,07575758	0,45454545	0,15151515	0,3030303	0,09090909
2	8,72727273	2	5,21212121	1,2	17,1393939	0,11669024	1,01838755	0,23338048	0,60820368	0,14002829
3	12,1103253	2	6,43281471	1,2	21,74314	0,13797455	1,67091671	0,2759491	0,88756473	0,16556946
						8,27847311 jobs/h				

JMVA

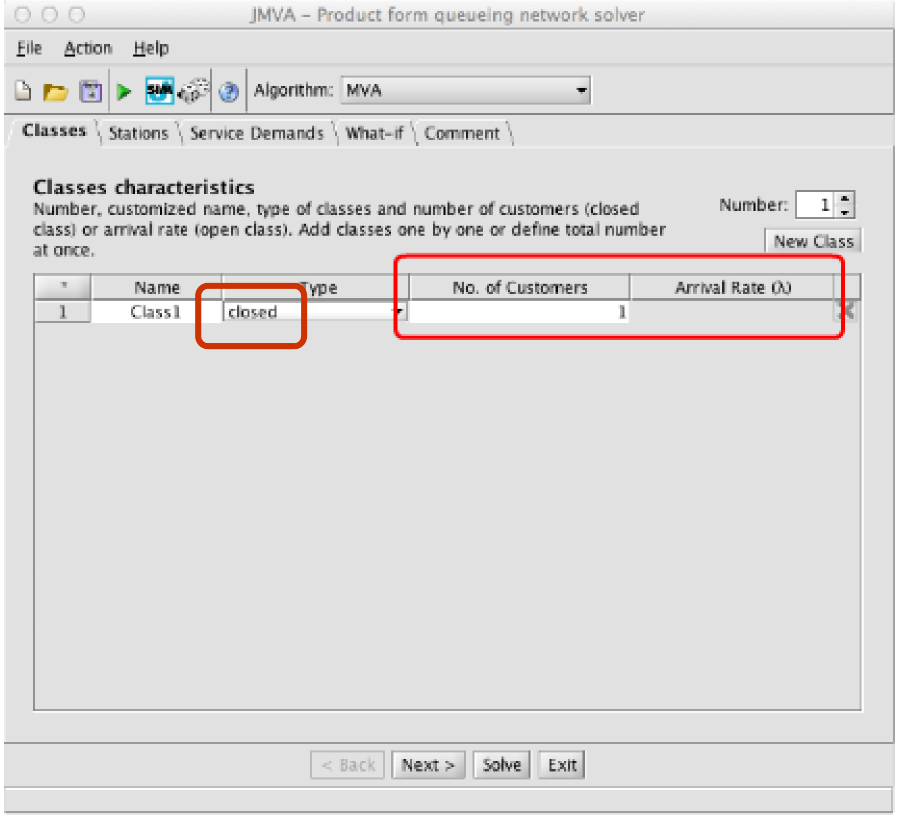
The JMT tool includes a component to perform analysis of open and closed separable models called *JMVA*.



JMVA

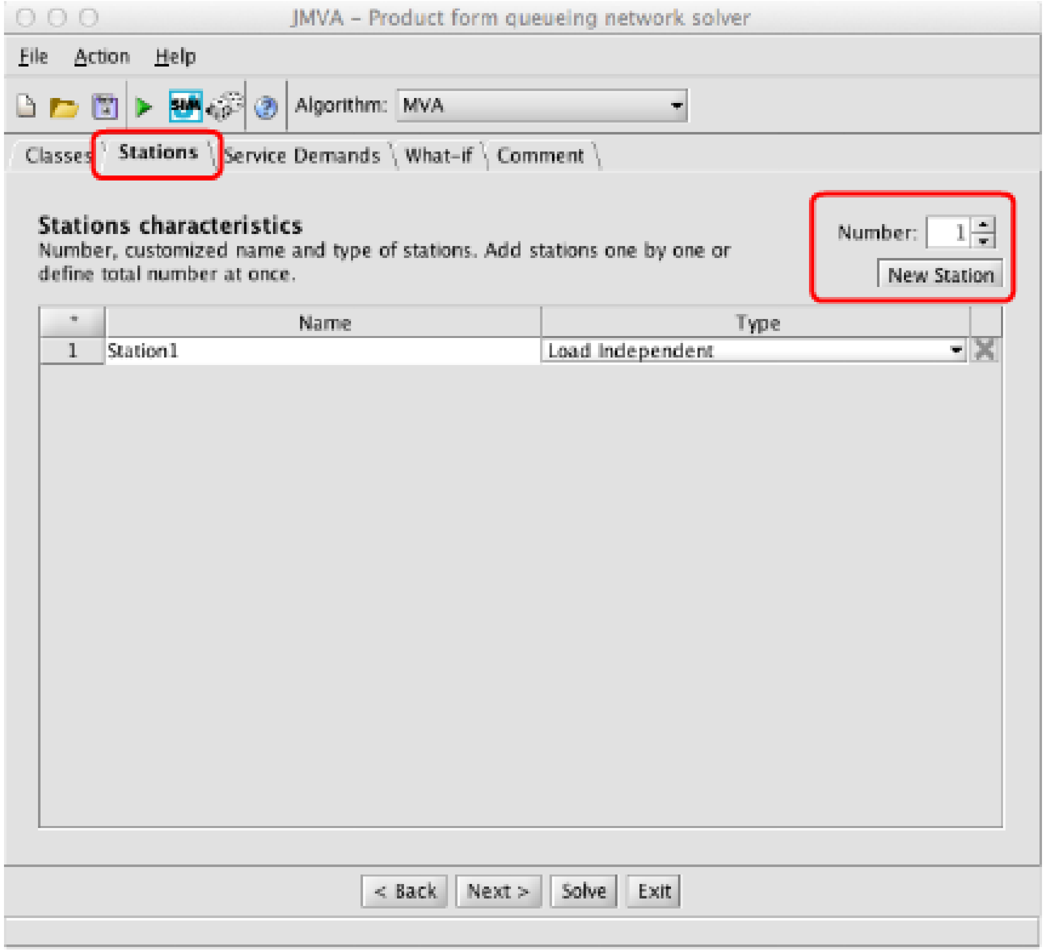
The user starts selecting the classes used in the model (we will return on this later), and defining whether they are open or closed.

In the same page the user specifies either the arrival rate (open models), or the total population (closed models).



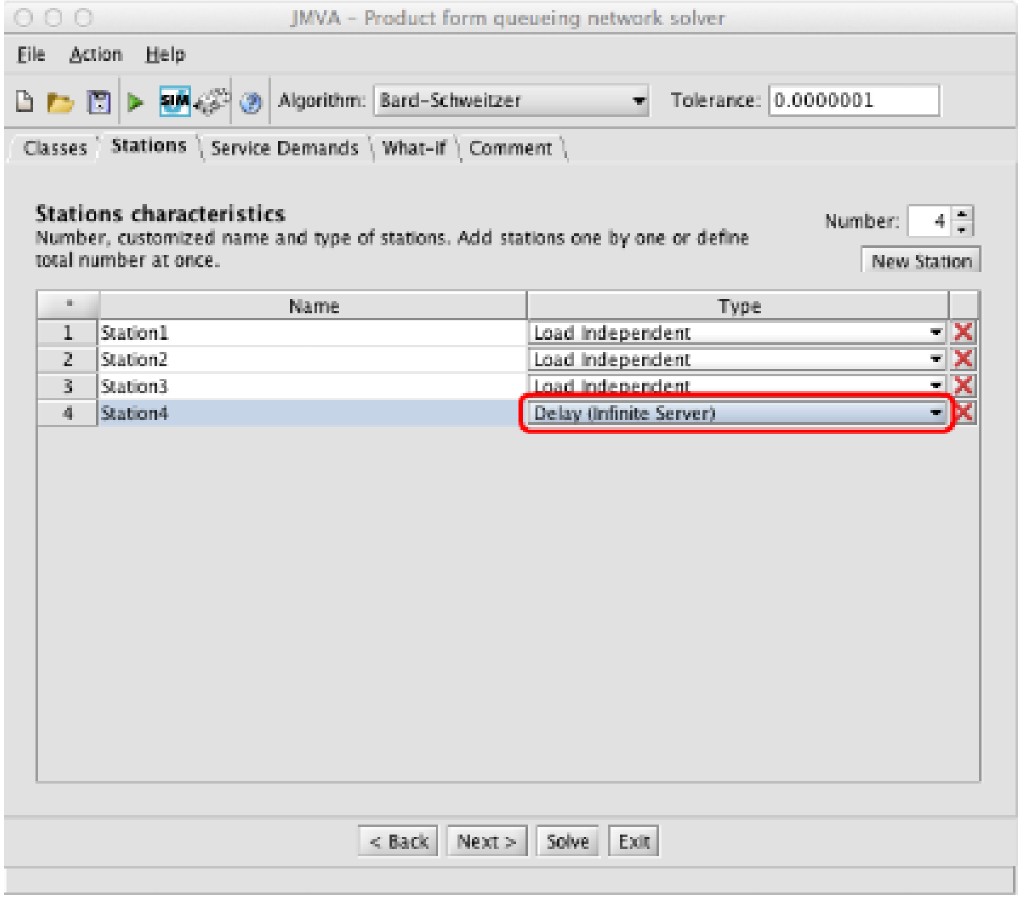
JMVA

Since separable models are fully characterized by their demand, the tool does not ask us the topology of the network, but it just allows us to insert as many stations as needed.



JMVA

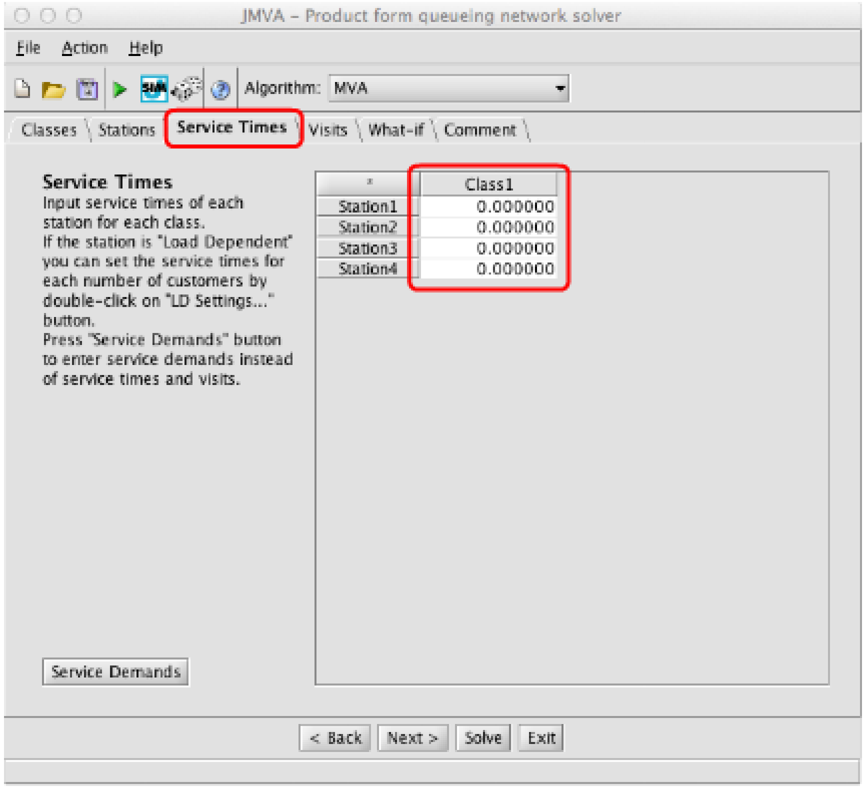
The tool supports both single server queues (Load independent), and infinite server stations. The latter can be used to insert the terminal station in time-sharing models.

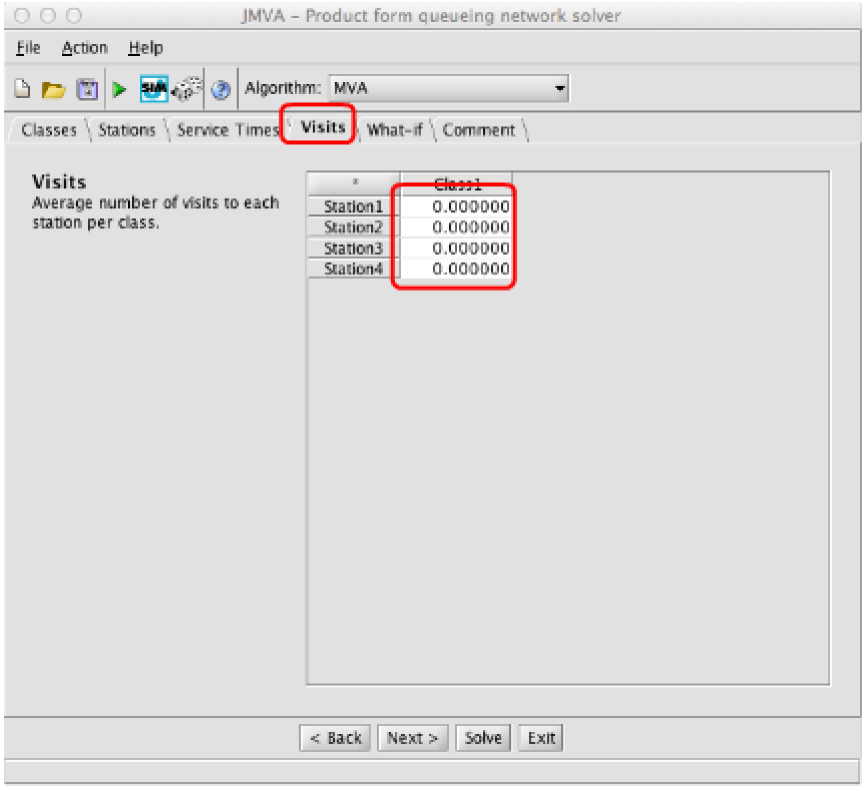


Load dependent stations can be used to add multiple server queues. However, since their definition is a bit complex, we will not consider them in this course.

JMVA

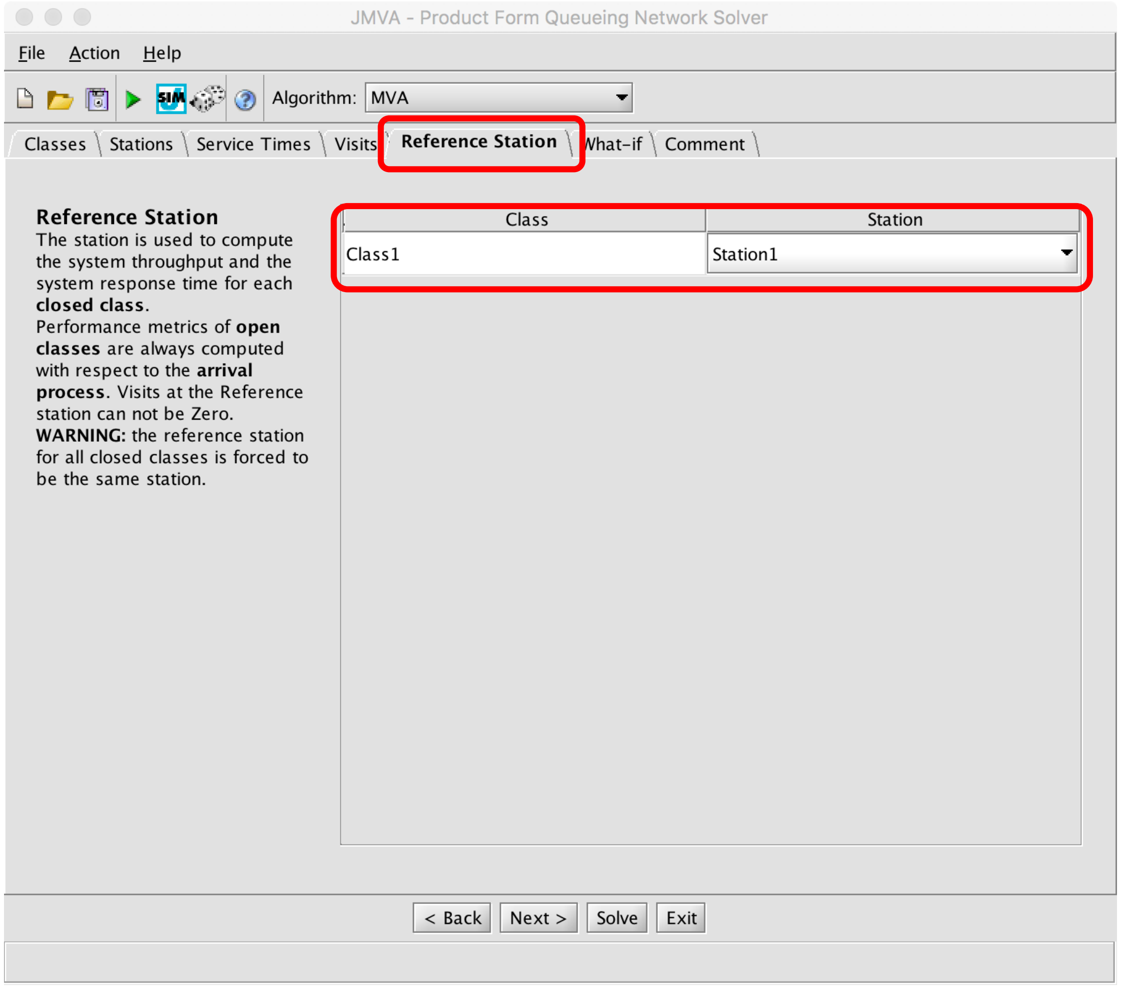
For each station we then have to specify the average service time, and the visits in the corresponding tabs. Note that in time sharing models, this is also used to specify the think time of the terminal station.





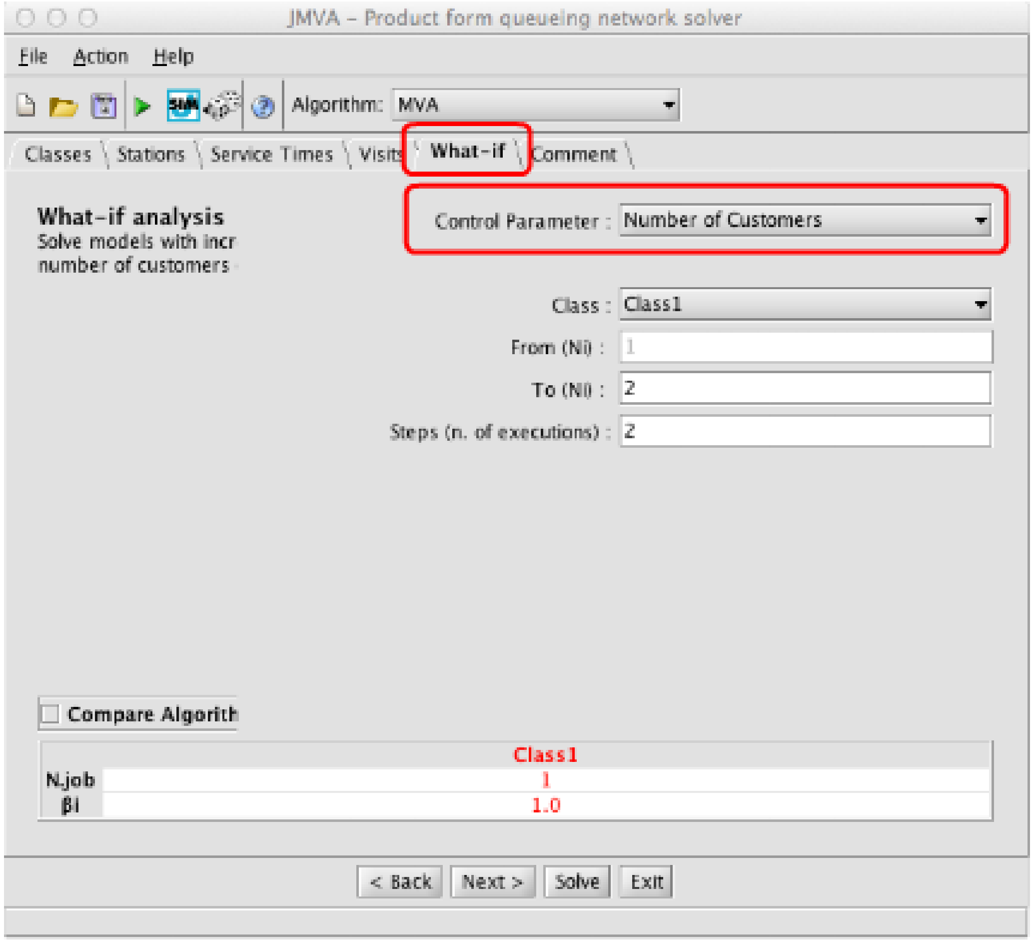
JMVA

We also have to define the reference station in the corresponding panel.



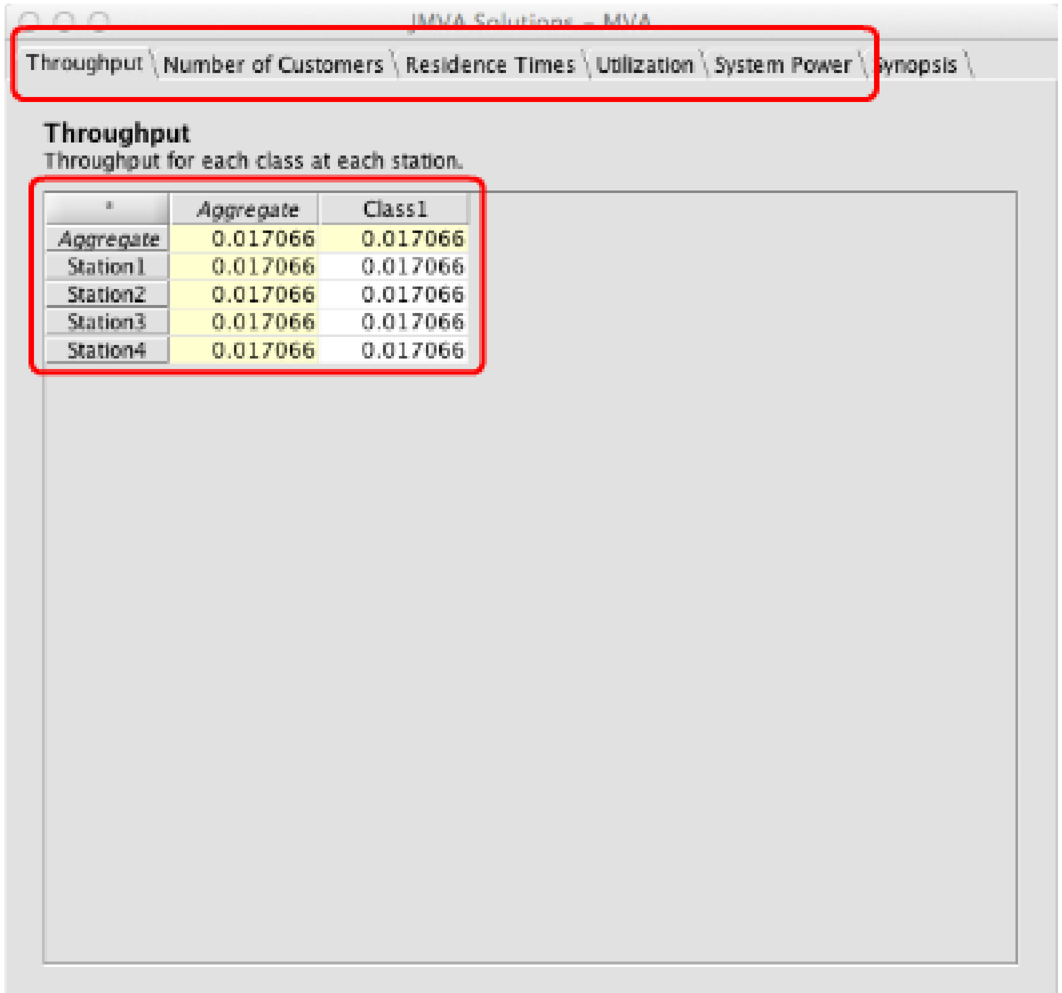
JMVA

If we want to perform several studies changing the workload, we can setup a range of experiments in the *What-if* tab.



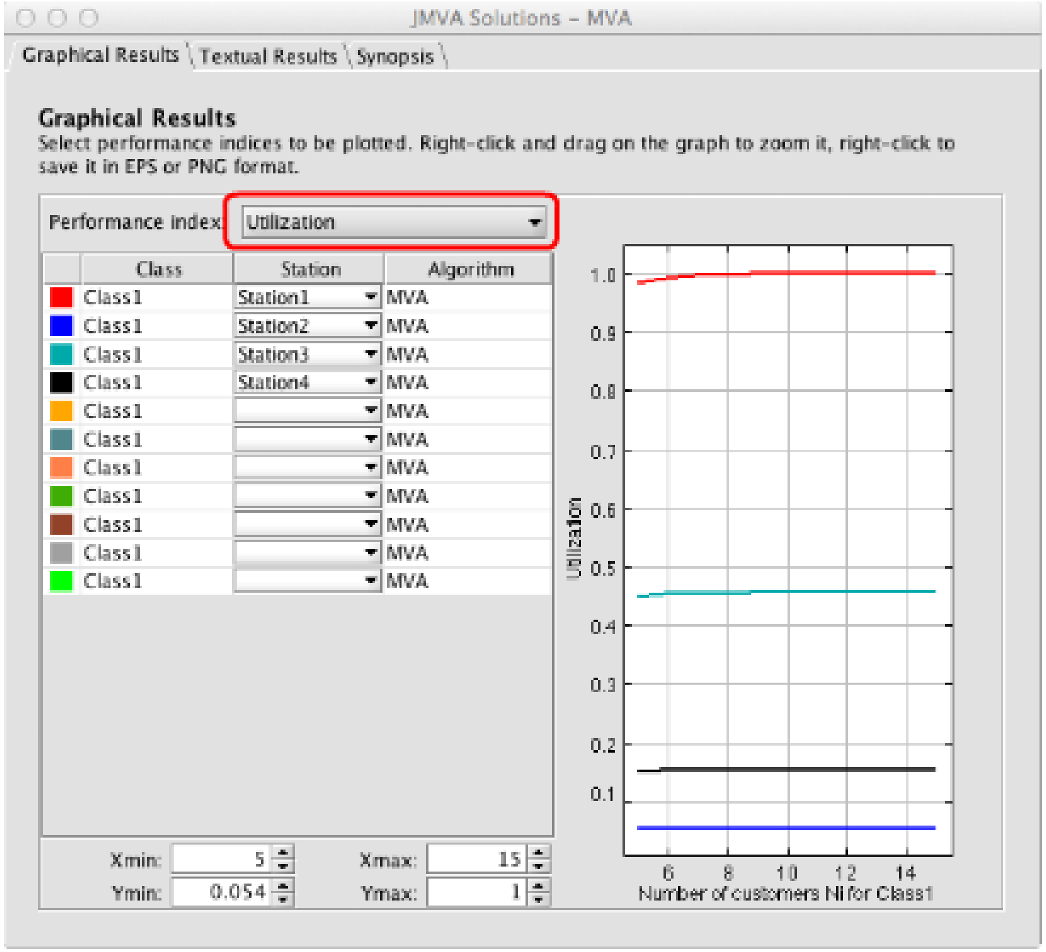
JMVA

JMVA always computes all the performance indices, and present them in a set of panels.



JMVA

If *What-if* analysis has been enabled, the tool allows to plot and compare the various performance indices.



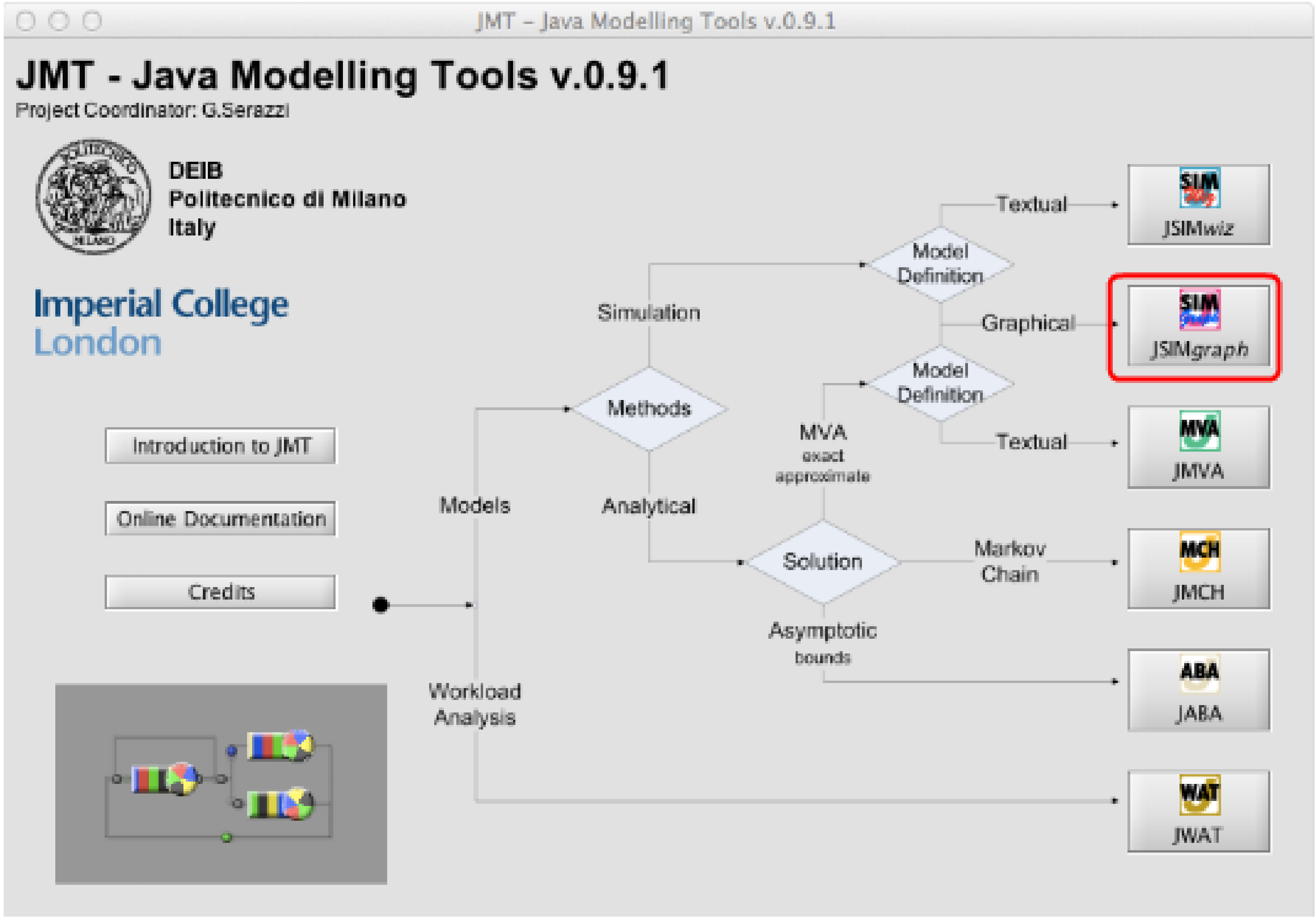
Advanced Queuing Network Features

When a model is not separable, it very rarely enjoys analytical solutions or numerical techniques to compute the relevant performance indices.

In most of the cases, models exploiting these properties are solved via *discrete event simulation*.

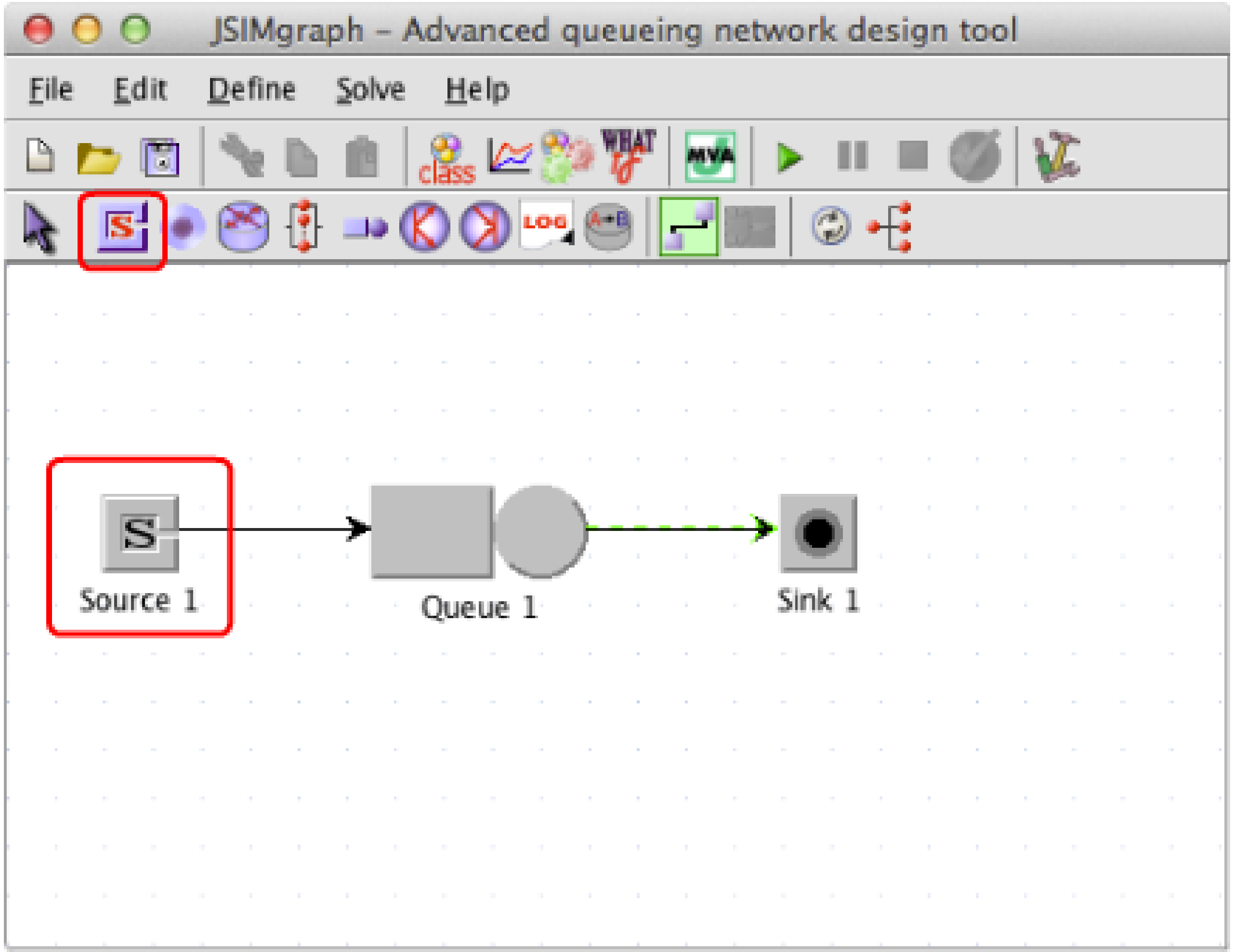
Advanced Queuing Network features in JMT

We will show these features in the *JSIMgraph* tool of JMT.



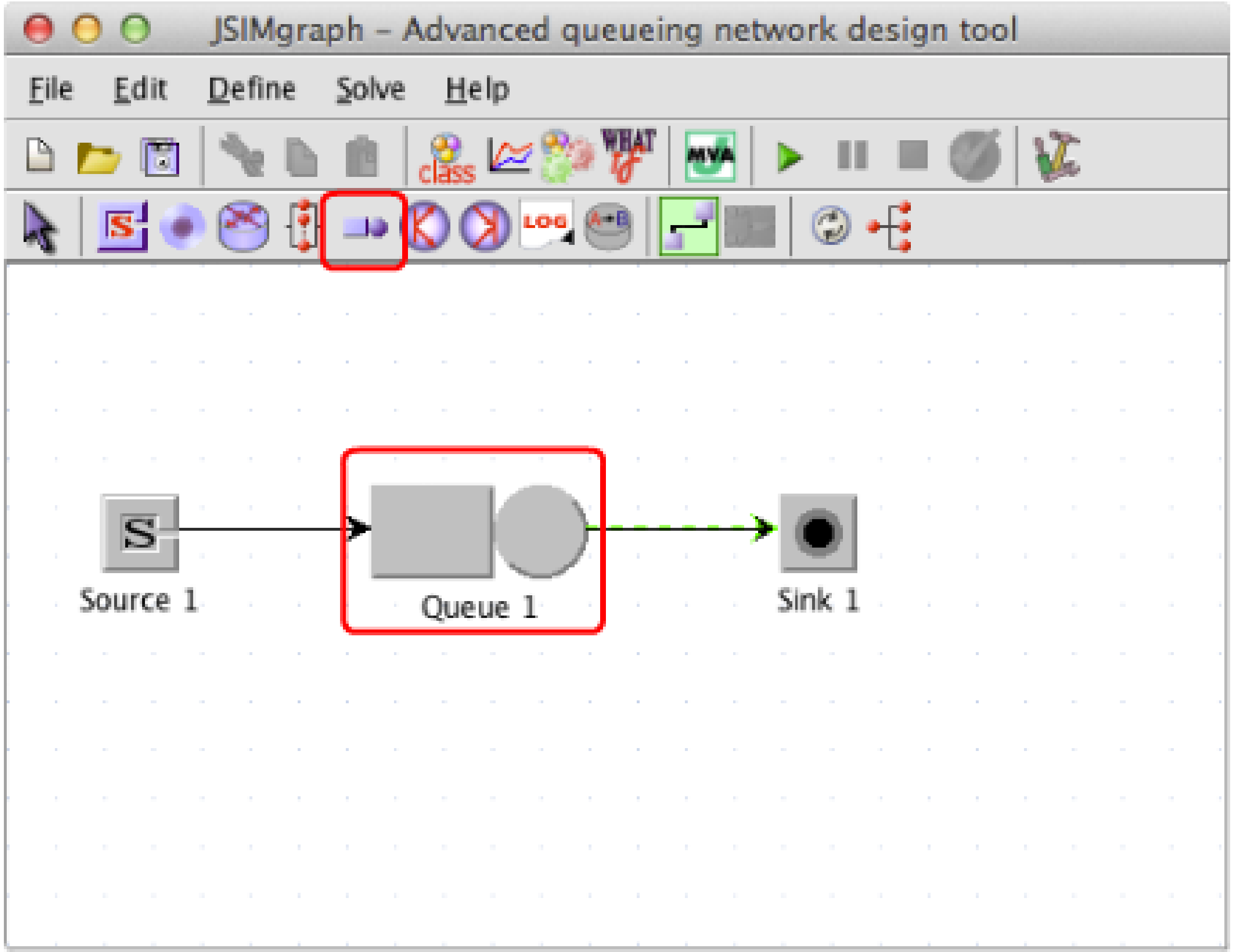
Single queues in JMT

Let's start with a simple (open) single queue model. It is composed by three elements: *a Source node*, representing the arrivals.



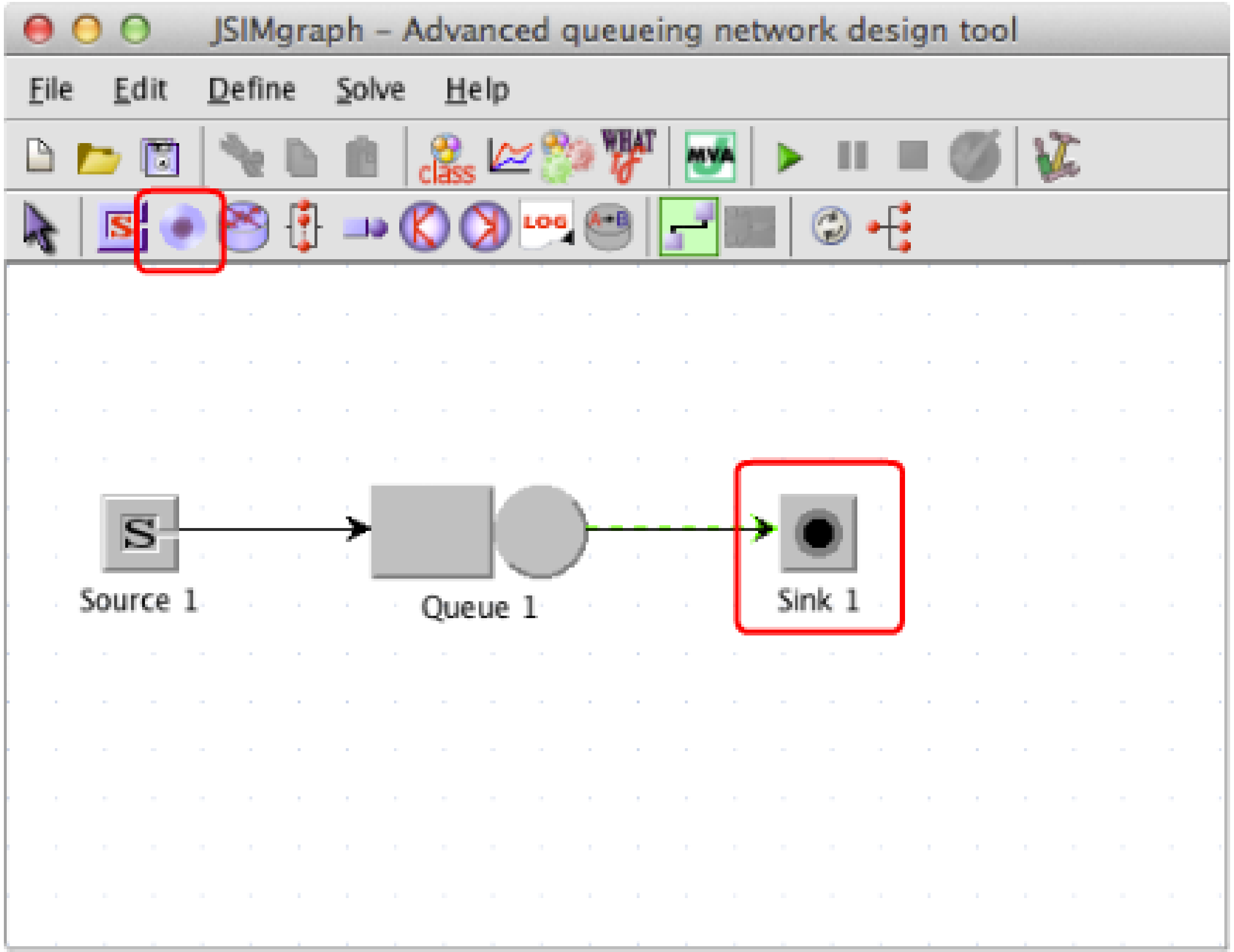
Single queues in JMT

A *Queue node* represents the single queuing station.



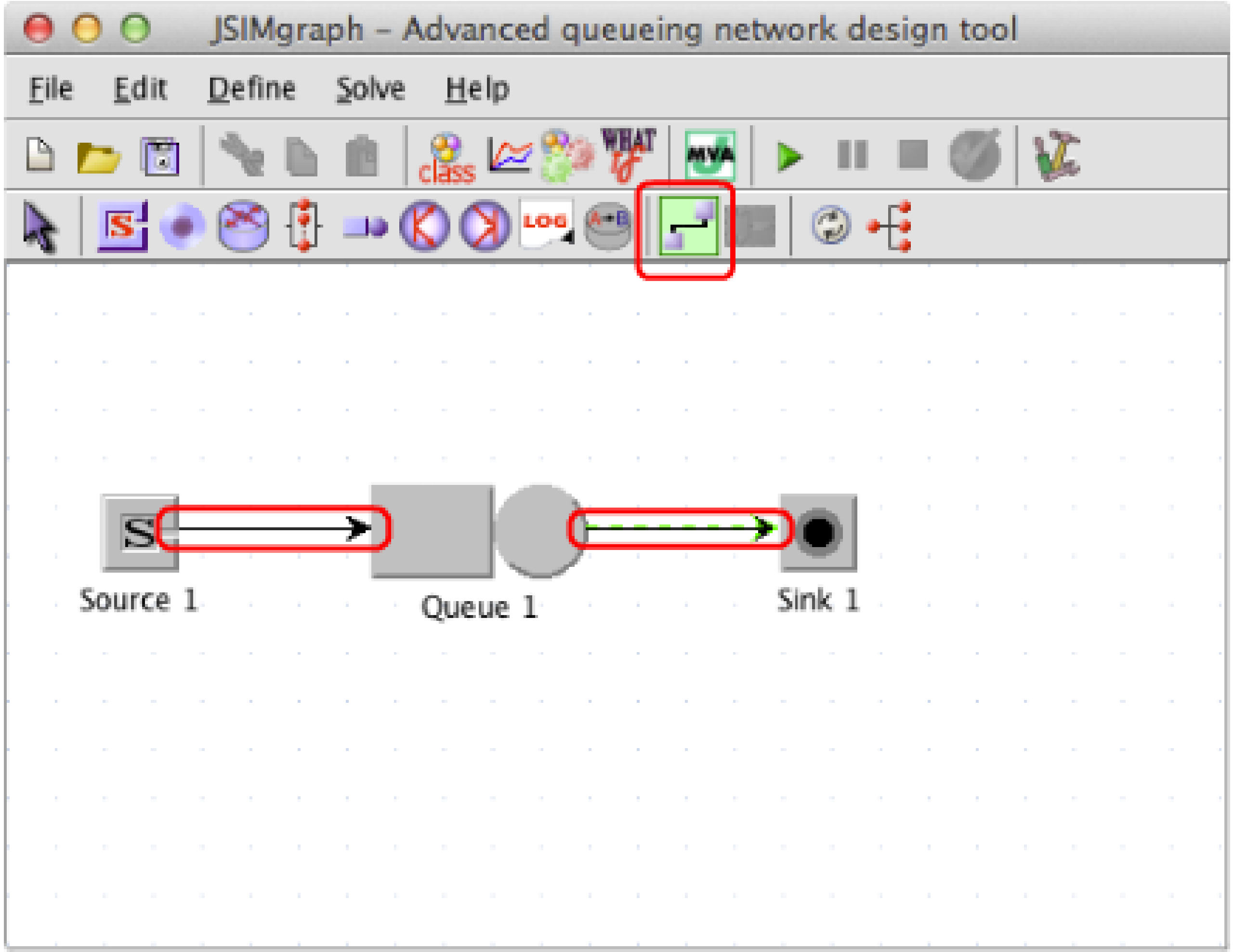
Single queues in JMT

Finally, a Sink node is required to allow jobs, which have finished their service, to leave the system.



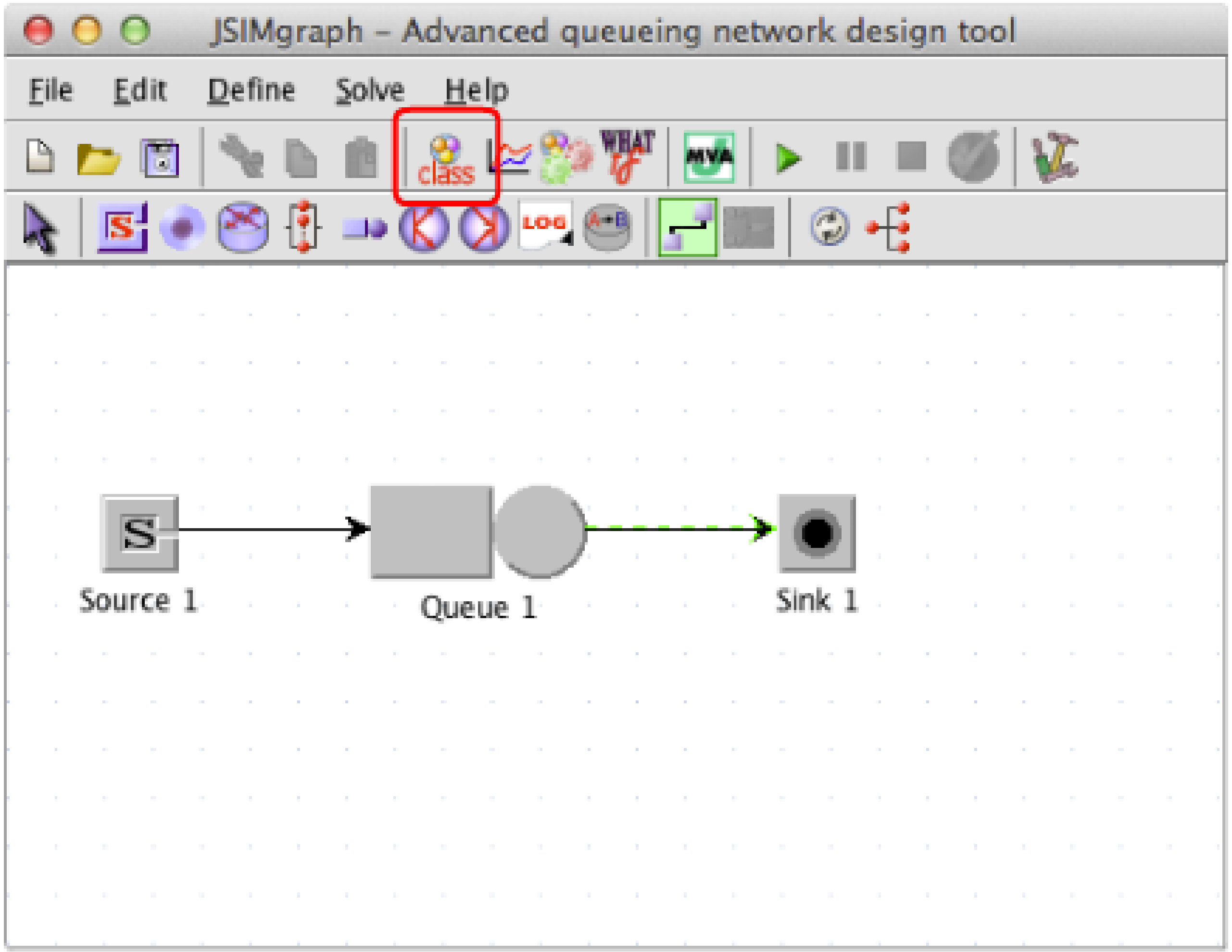
Single queues in JMT

Connections define the flow of jobs from the source to the queue, and from the queue to the sink.



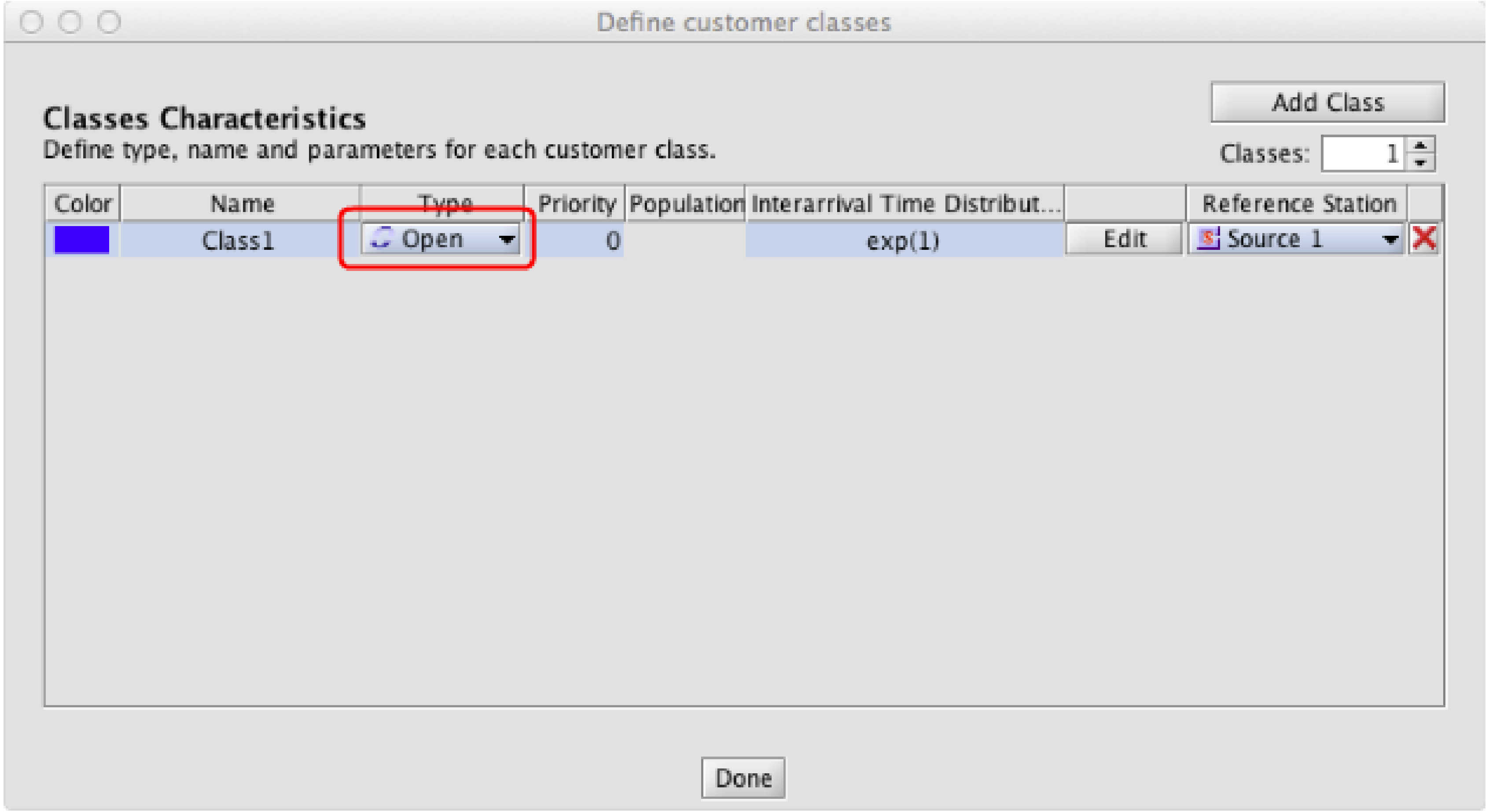
Single queues in JMT

Since JMT can support different types of jobs (we will return on this later), a new customer class must be defined.



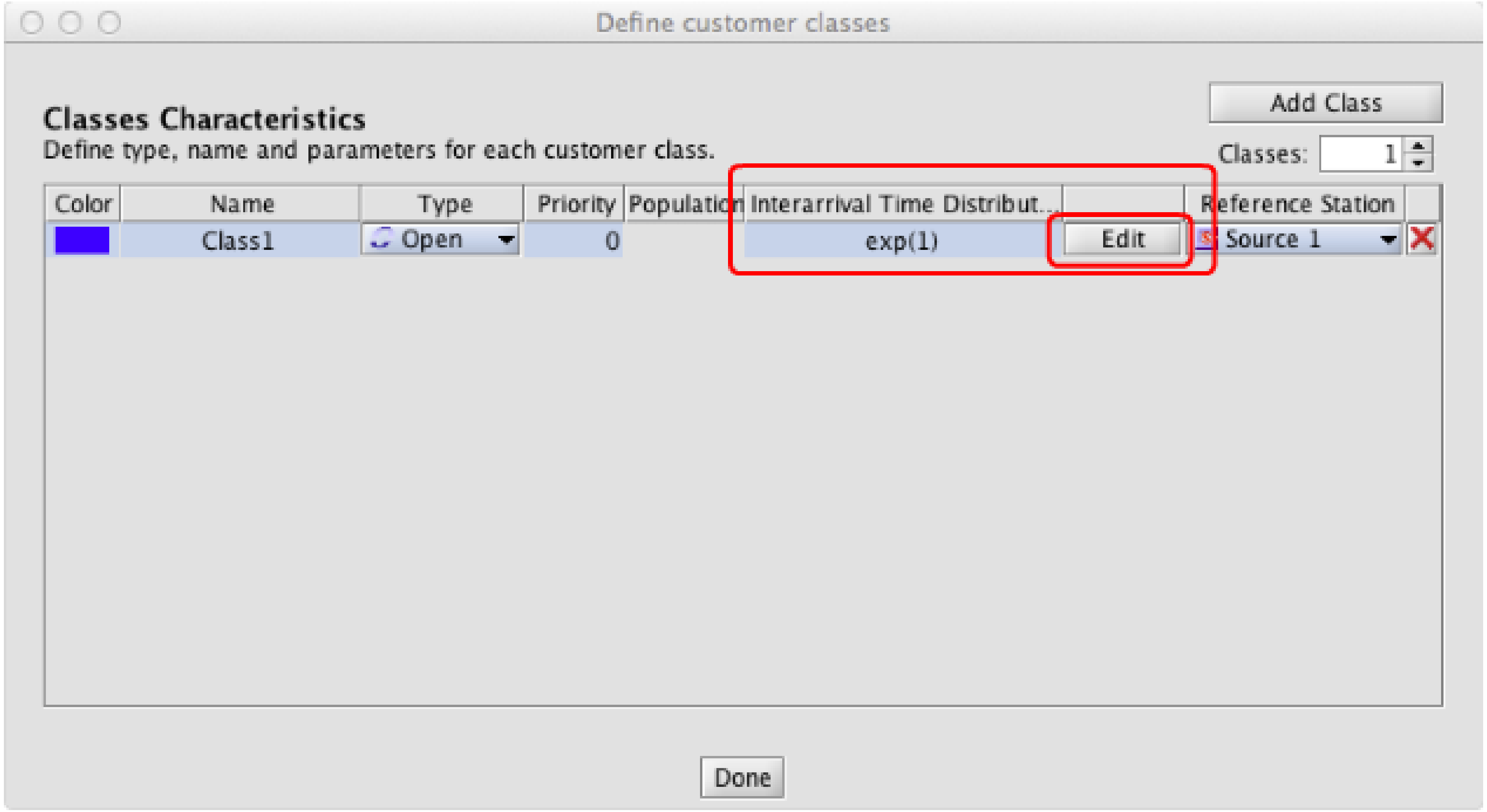
Single queues in JMT

For jobs arriving from an external source, the type of this class must be defined as *Open*.



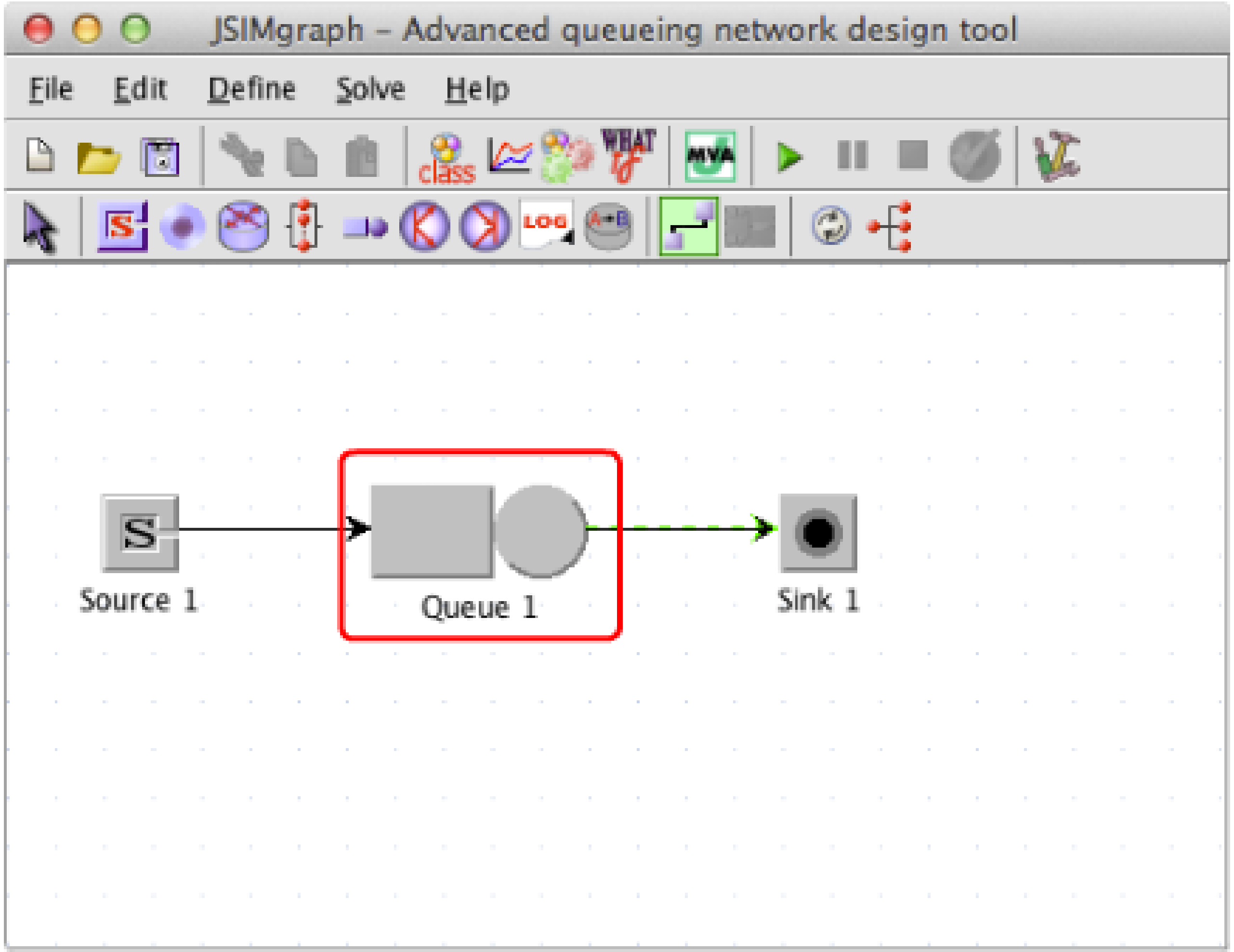
Single queues in JMT

The properties of the arrival process must be defined in the *Inter-arrival time distribution* column of the corresponding class. We will see how to define the arrival process pressing the Edit button in a few minutes.



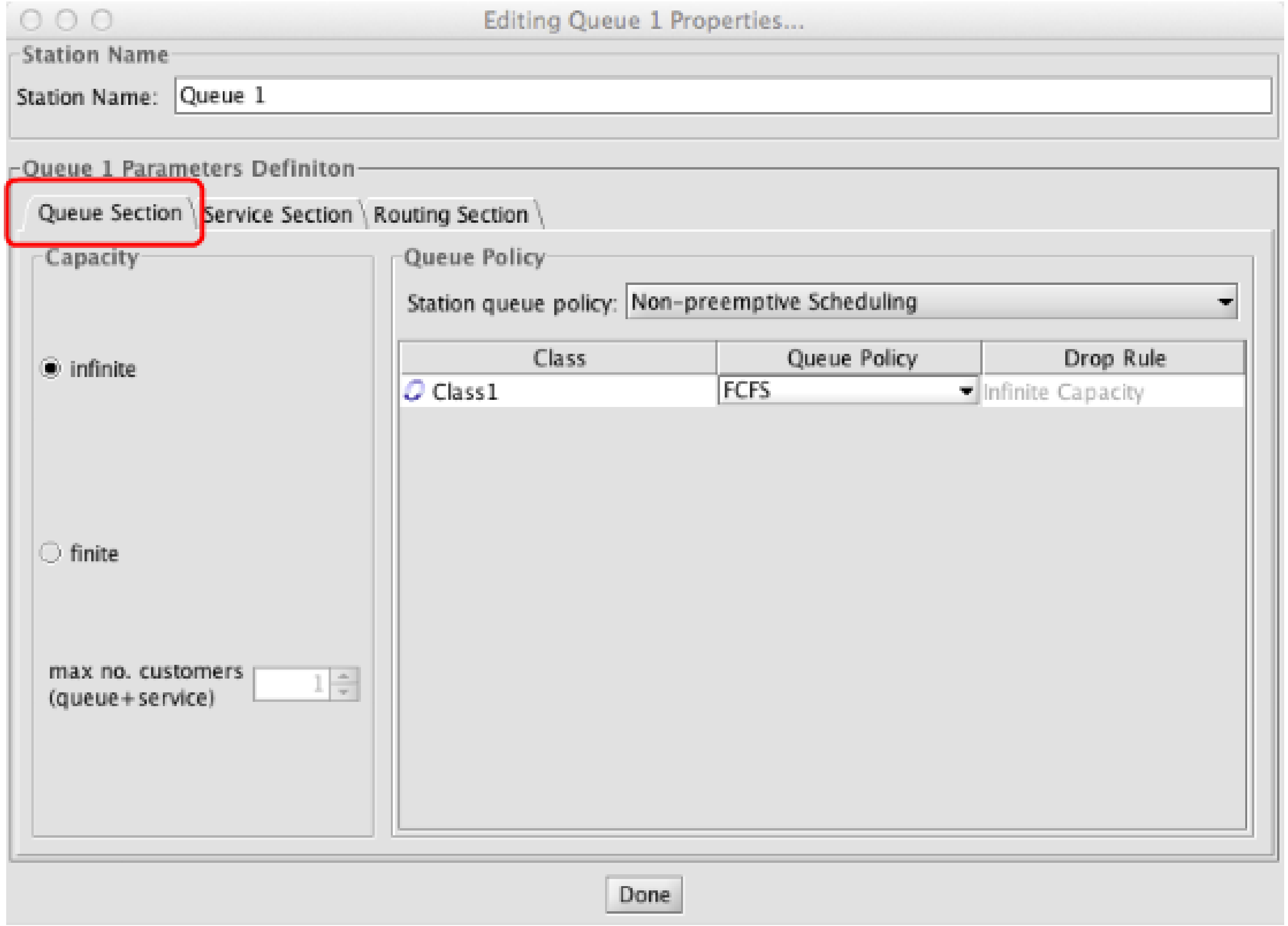
Single queues in JMT

Properties for both the queue and service section can be defined by double-clicking on the queue ...



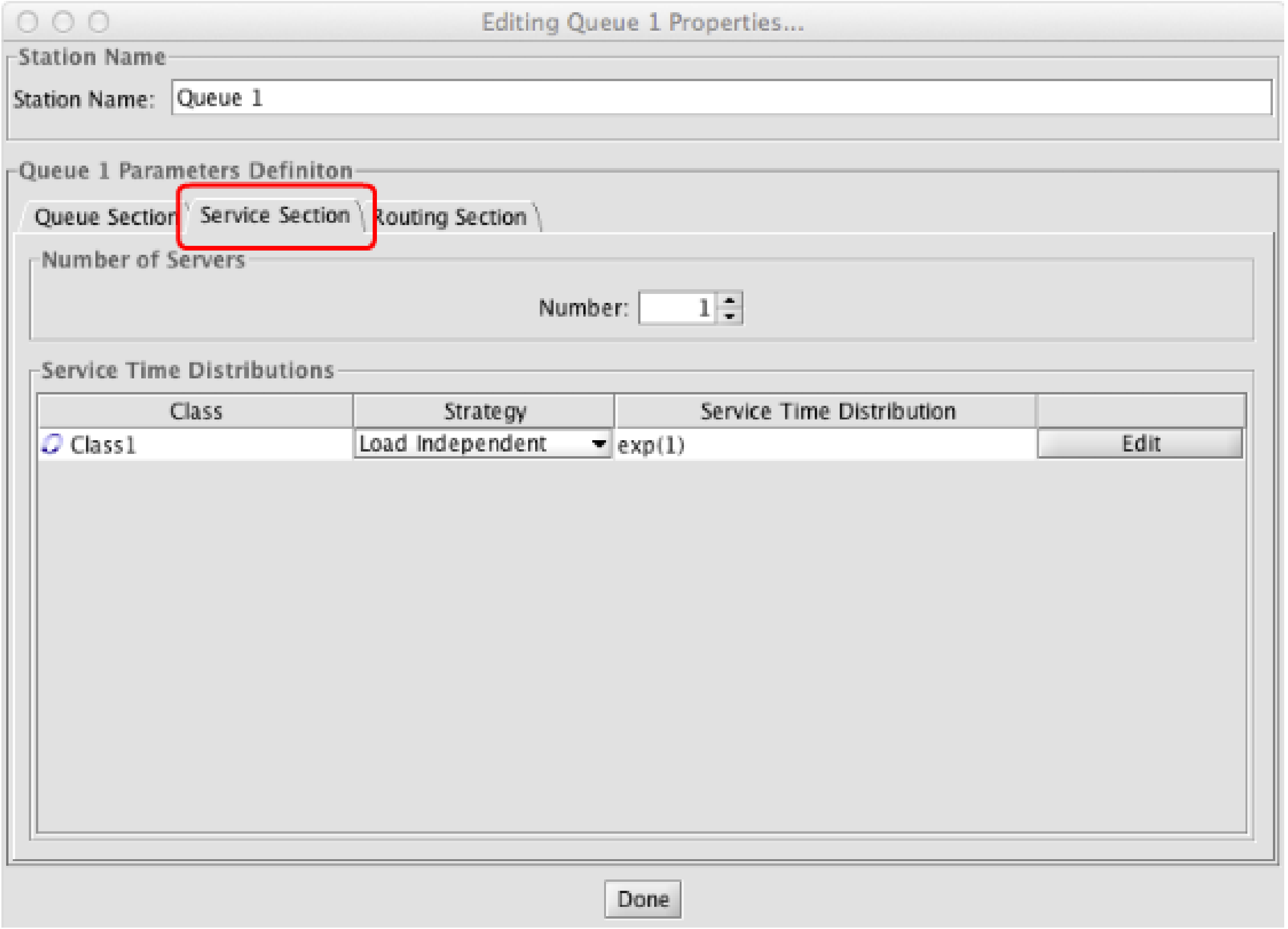
Single queues in JMT

then working respectively in the *Queue Section*



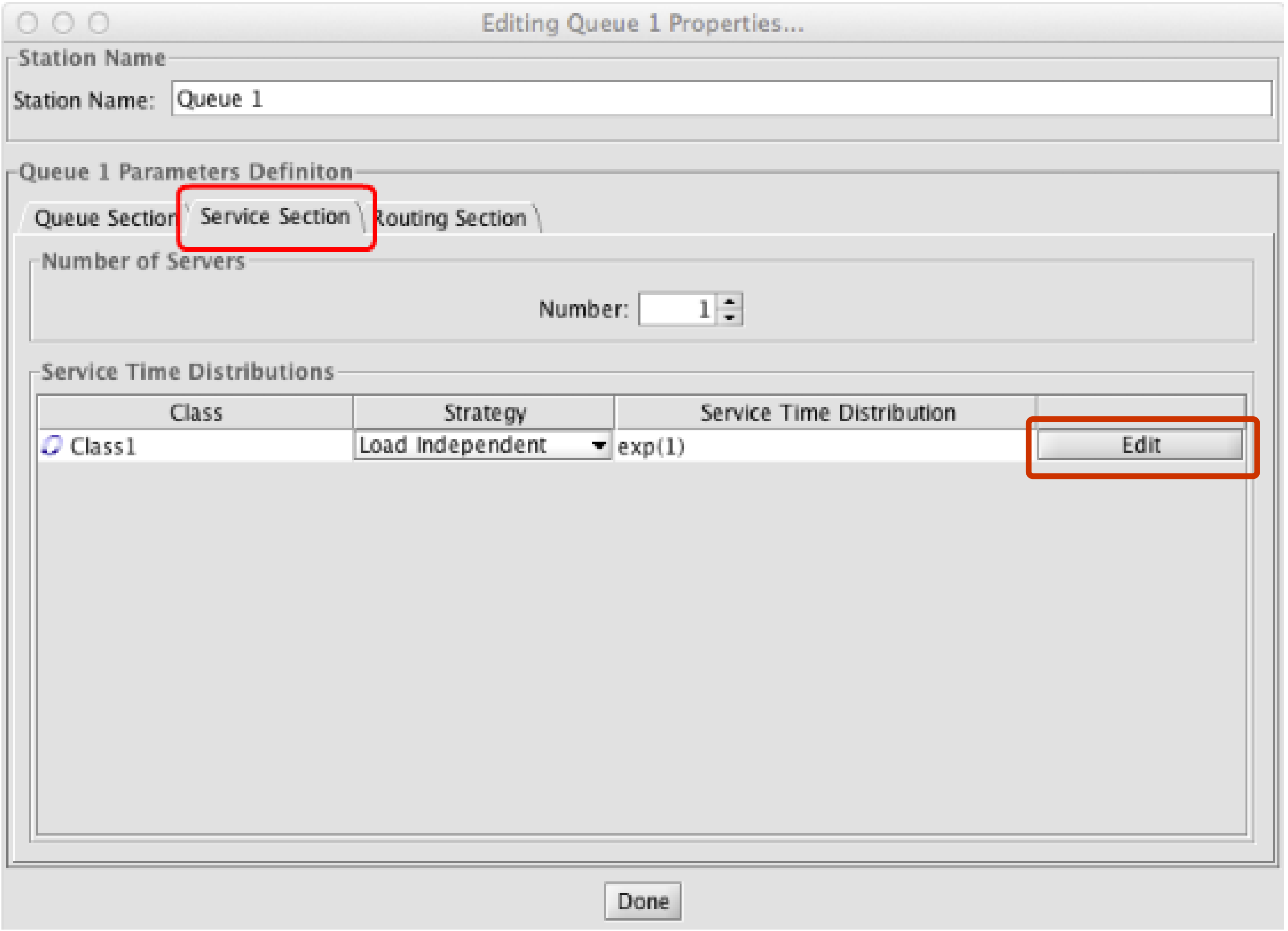
Single queues in JMT

and in the *Service Section*.



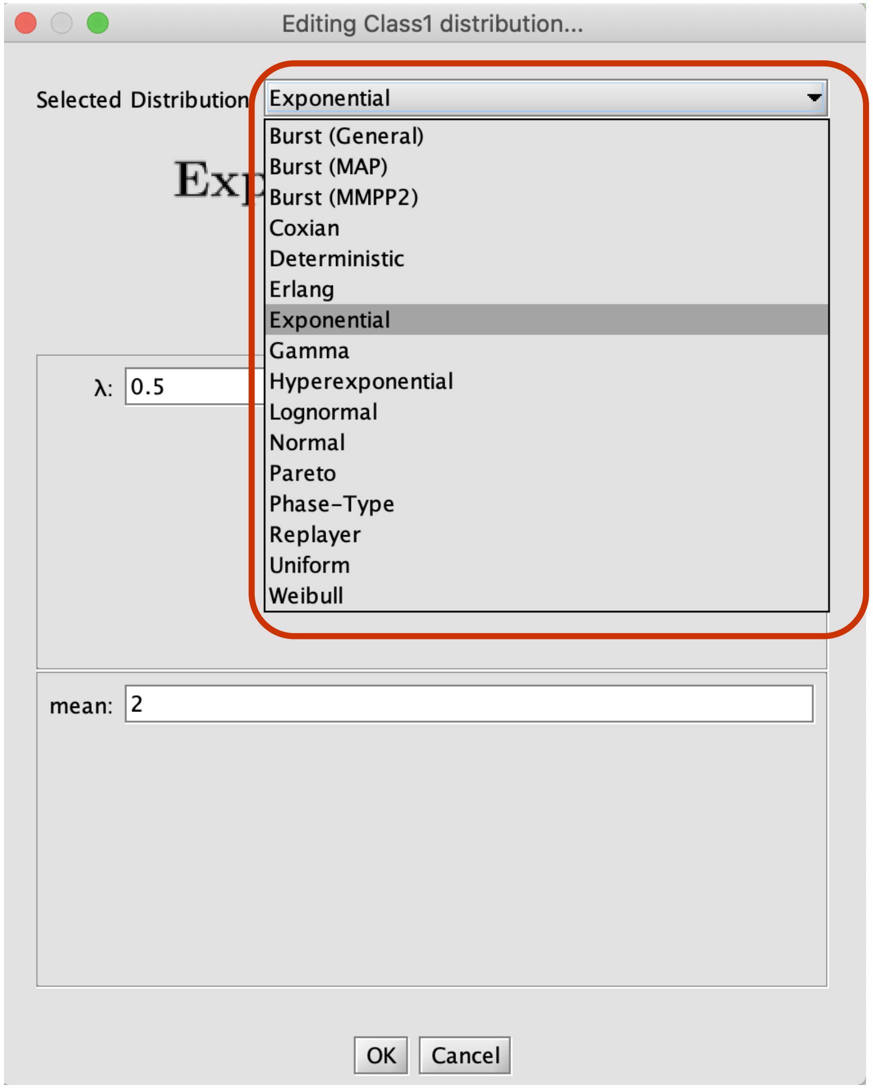
Single queues in JMT

Service time distribution can be defined pressing the *Edit* button.



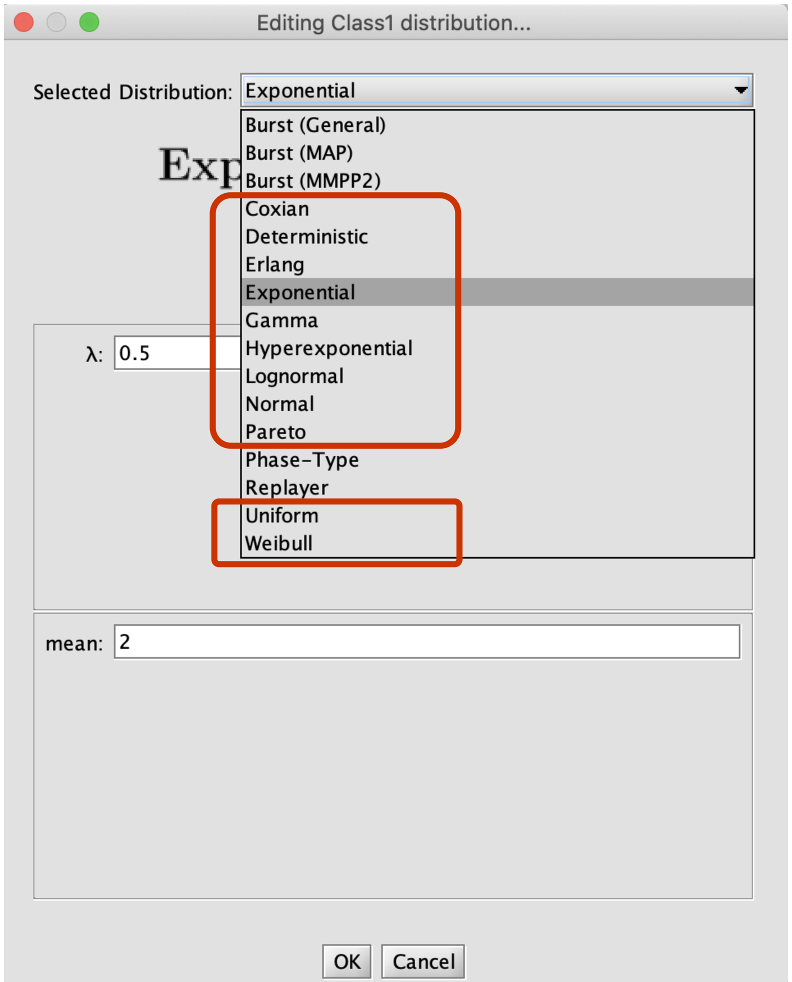
Distributions

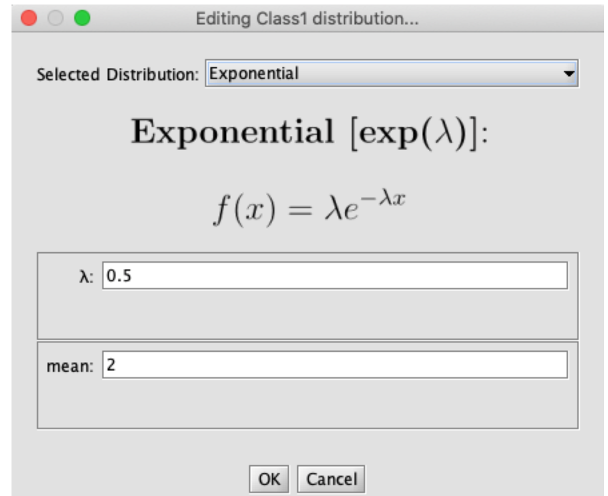
Several inter-arrival time and service distributions are possible:

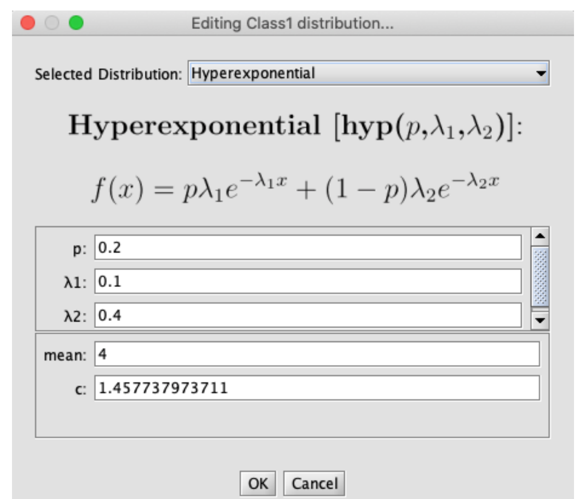


Distributions

Conventional distributions are characterized by their parameters, and are the following.

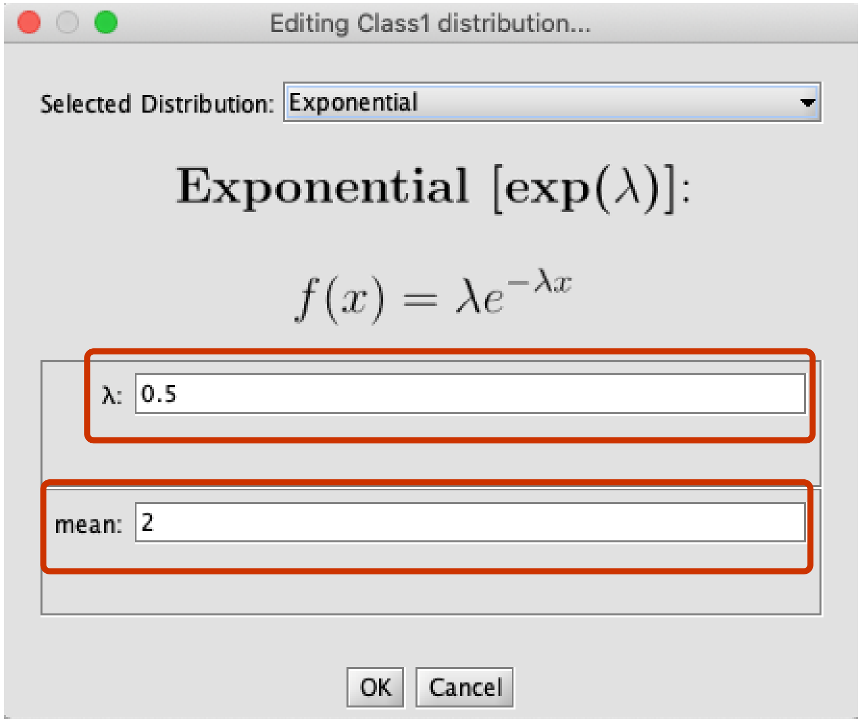






Distributions

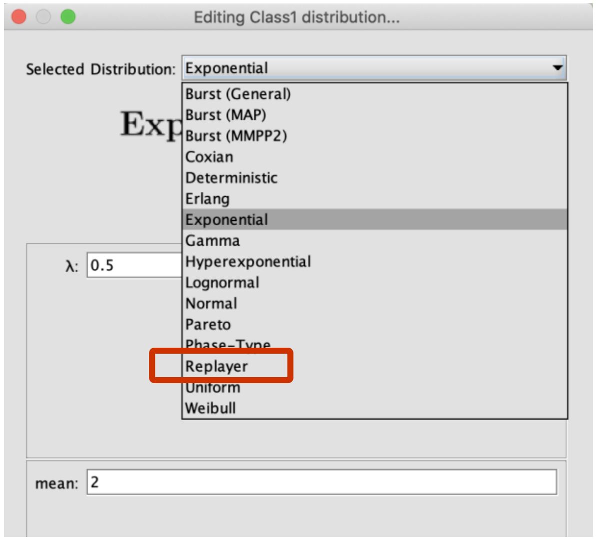
Some of them allow alternative parameterizations: for example the Exponential distribution can be defined through either the rate (useful for the arrivals), or with their average (better suited for services).

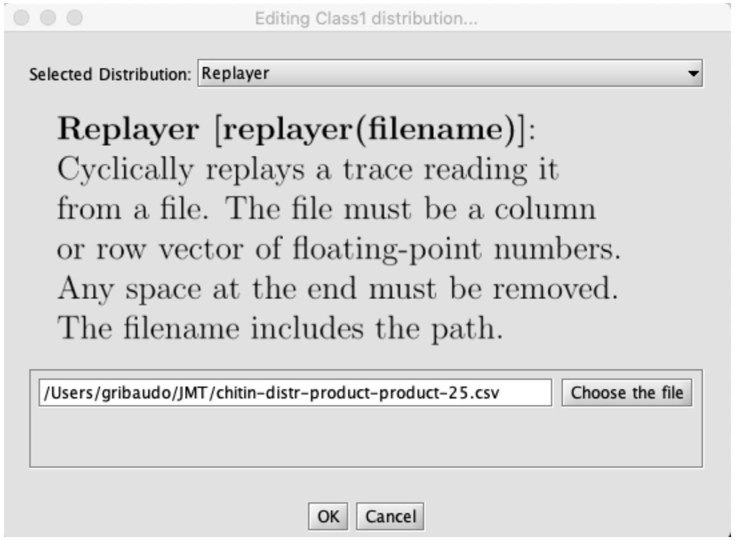


Distributions

Repeater allows to read samples from a file. It can be used for:

- Simulating a real system using data directly taken from its logs.
- Including unsupported distributions which have been previously generated by an external tool, and that have been saved in a file as a large set lot of samples.

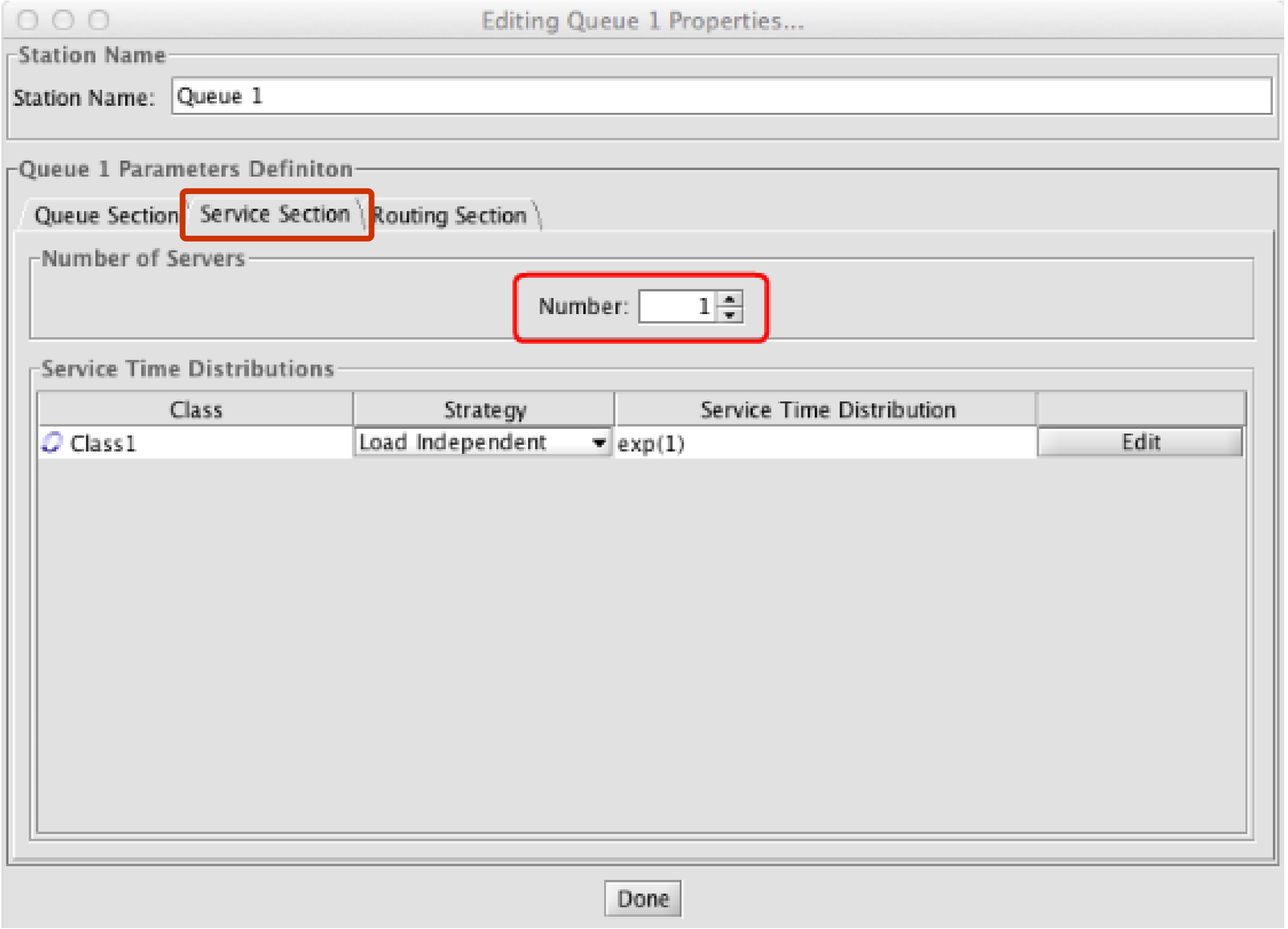




Please note that using directly log files, instead of fitting the distribution and generating an arbitrarily large set of samples, has a lot of limitations. It must thus be generally avoided and ***used only in very special cases*** (i.e. don't do it for the exam).

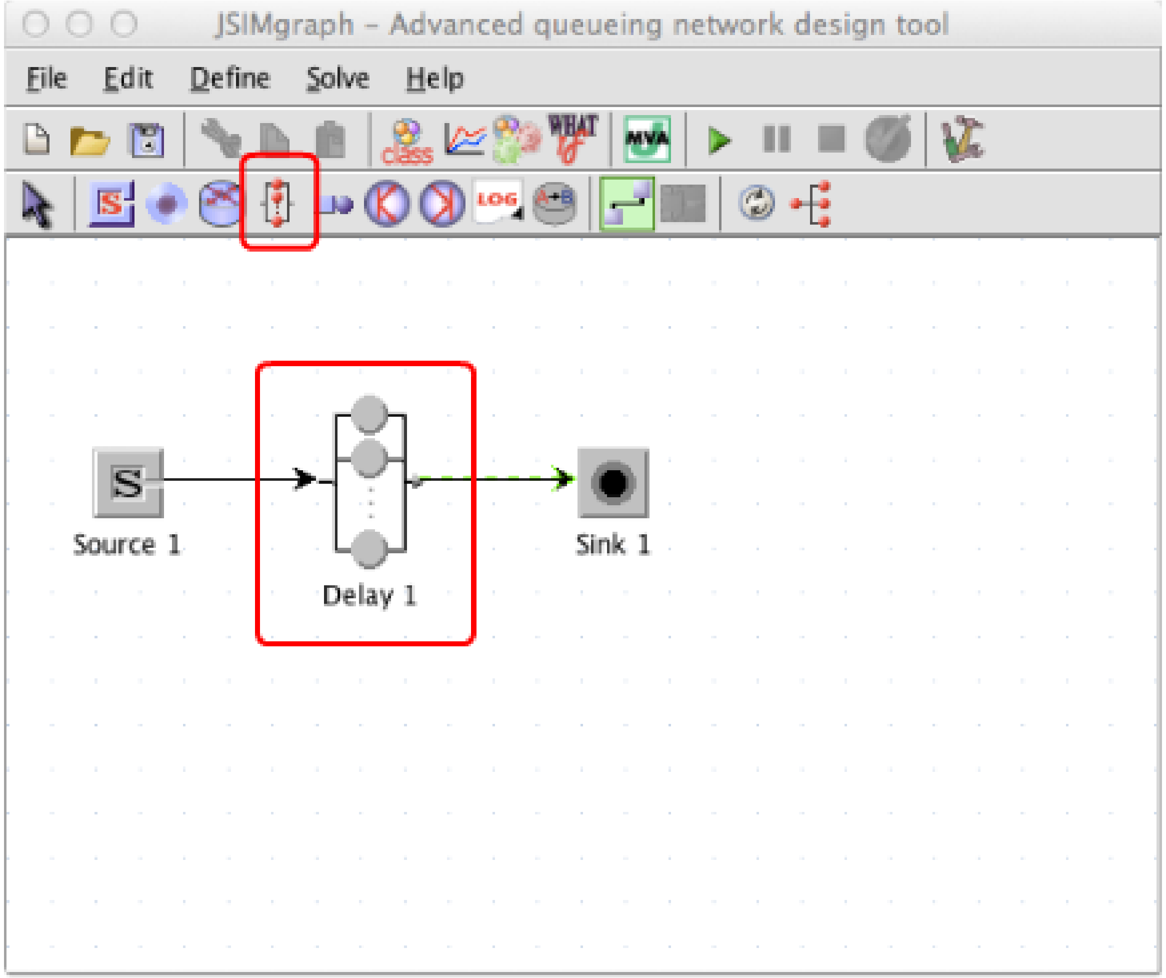
Multiple servers

In JMT, multiple servers are defined by an appropriate parameter in the service section of a station.



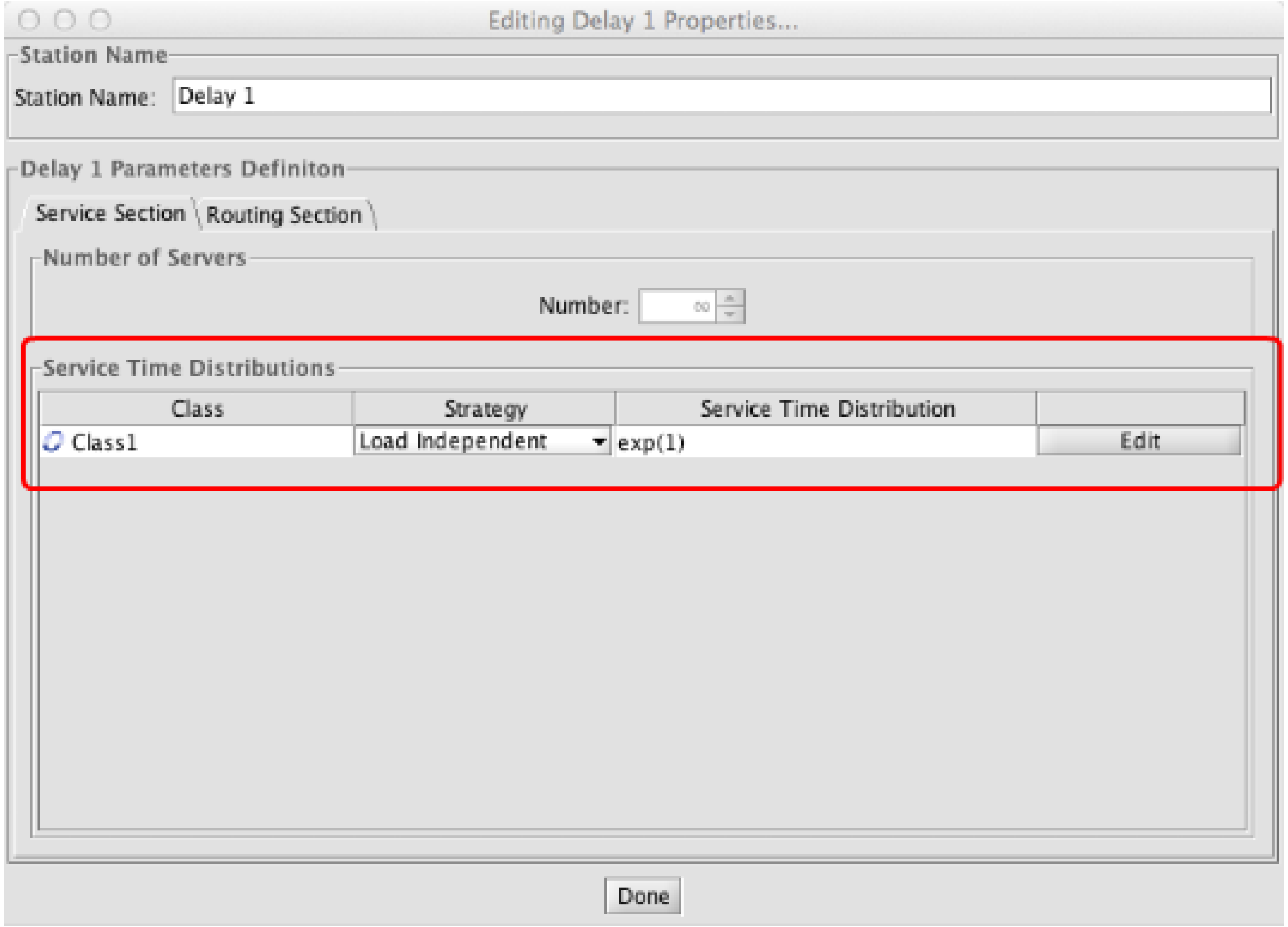
Infinite servers

Infinite servers are inserted using a specific primitive called "*Delay station*".



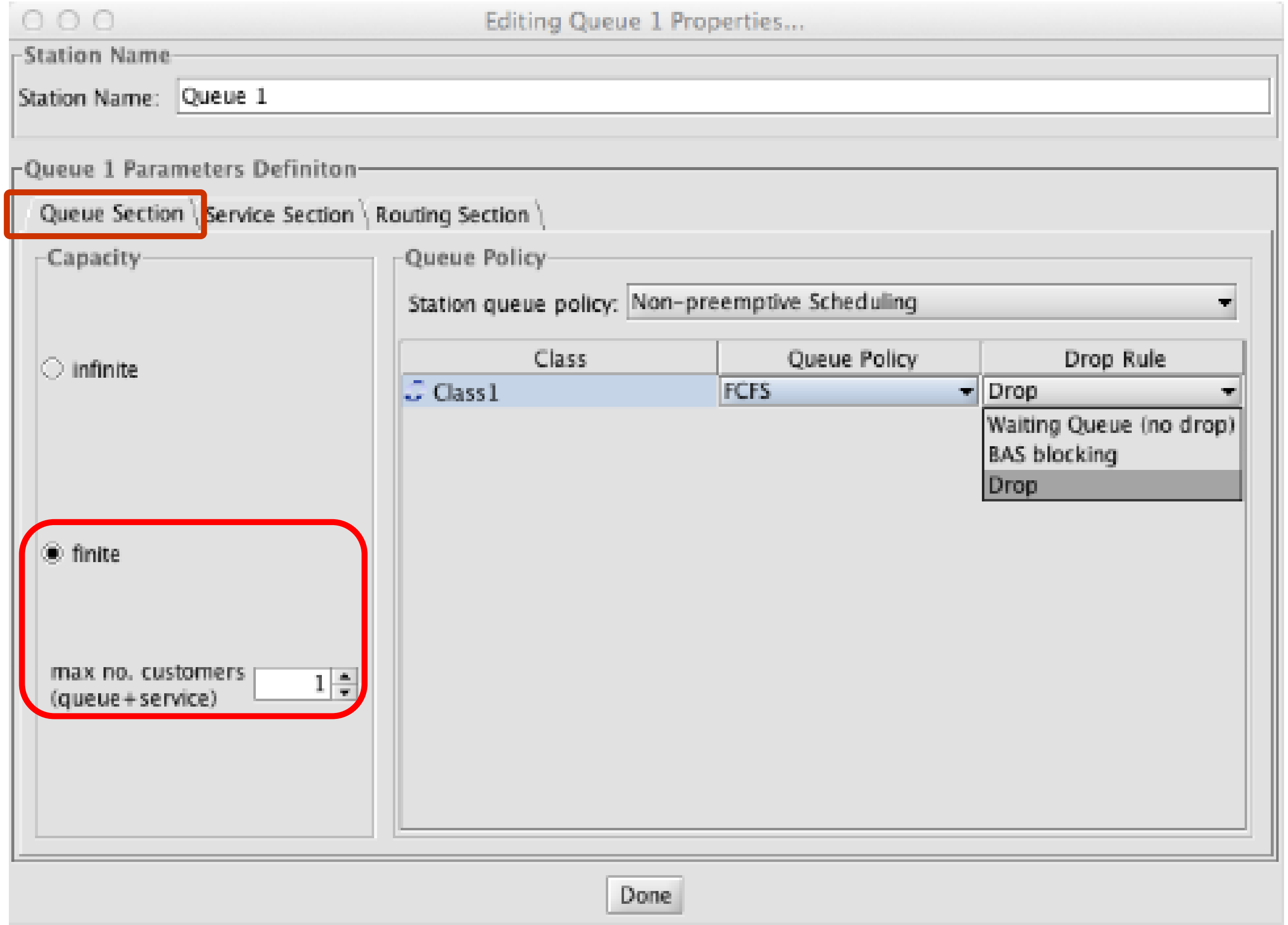
Infinite servers

Delay stations are characterized only by their service time (*waiting time*).



Finite capacity

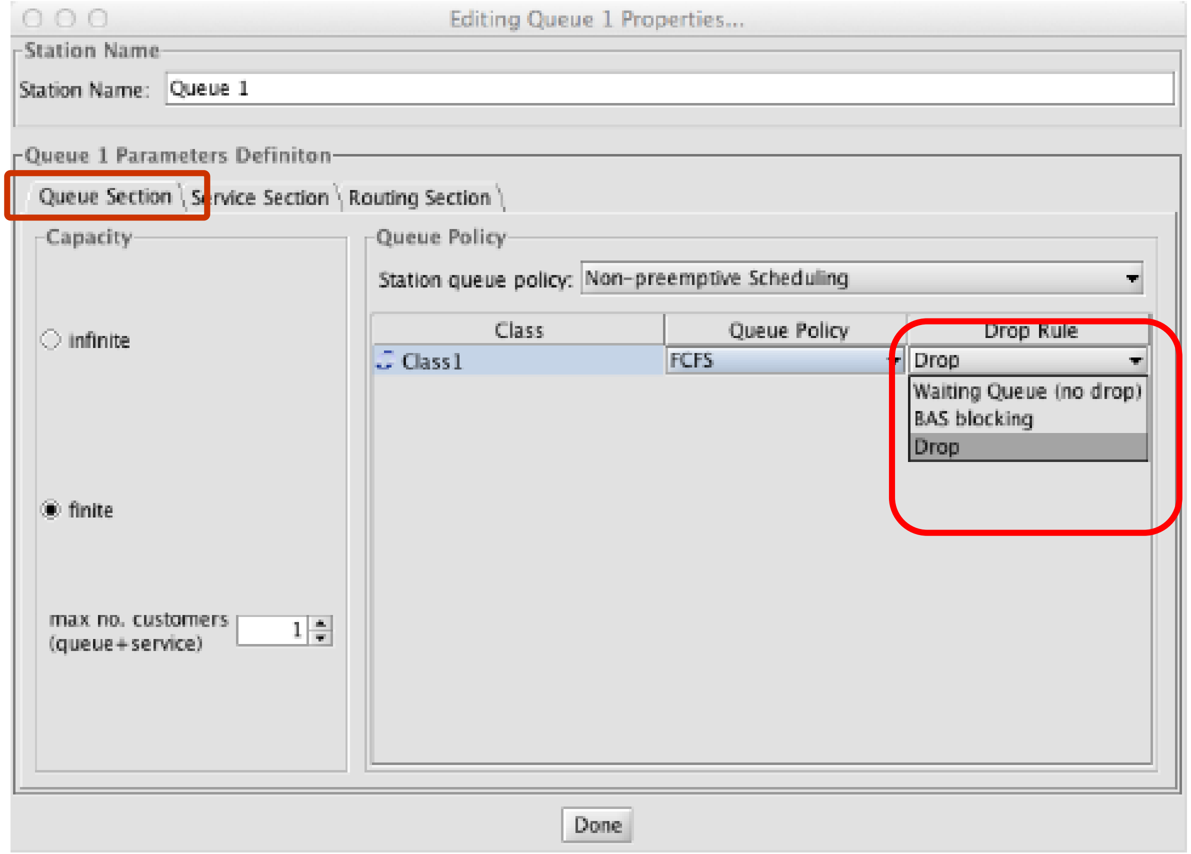
Losses and blocking are supported by specifying a finite capacity in the queuing section of node.



Finite capacity

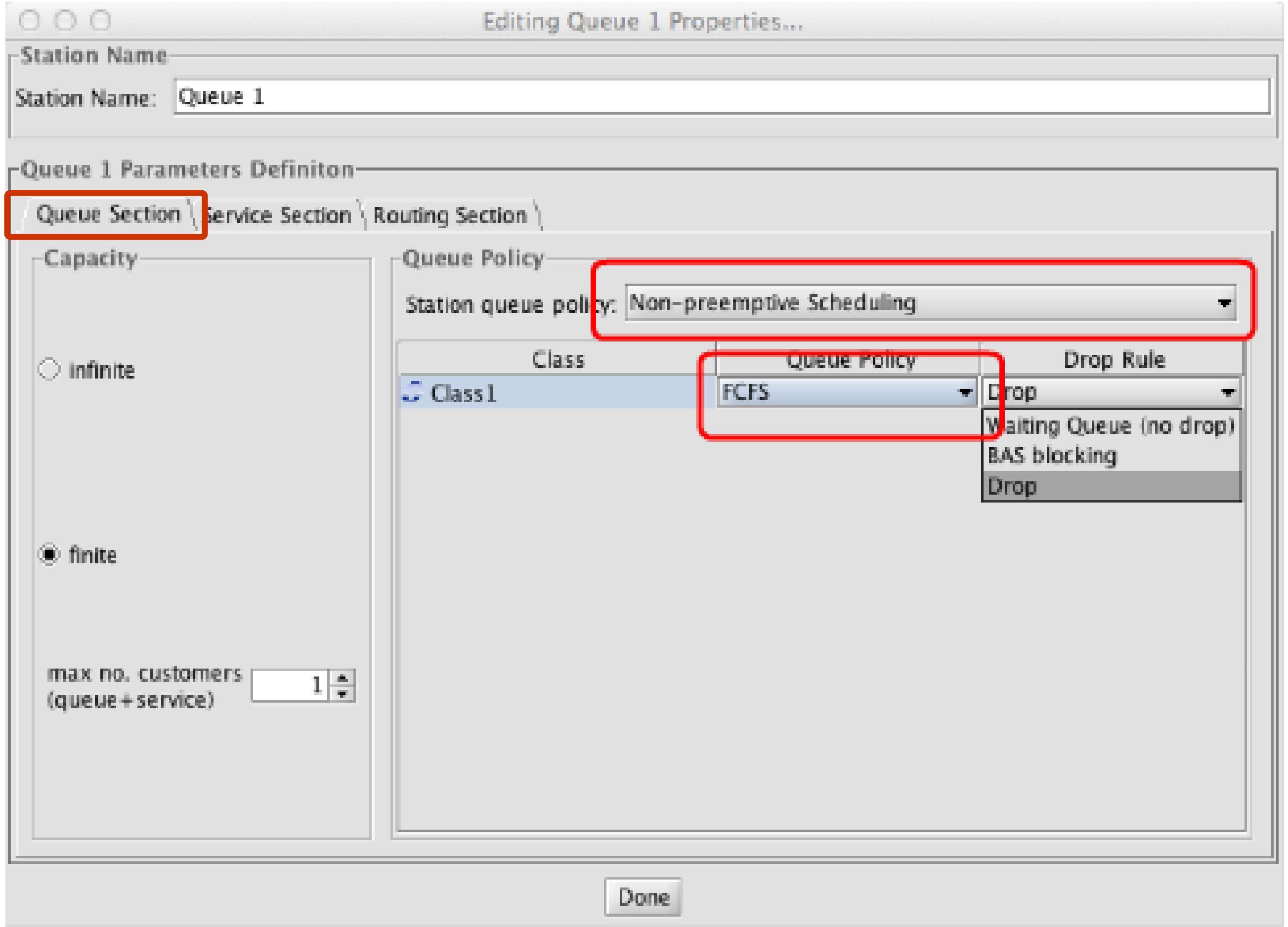
The choice between loss or blocking is performed in the appropriate column on the table in the central part of the station properties window.

JMT supports blocking for models in which the source is another station, and it mainly focus on the BAS mechanism.



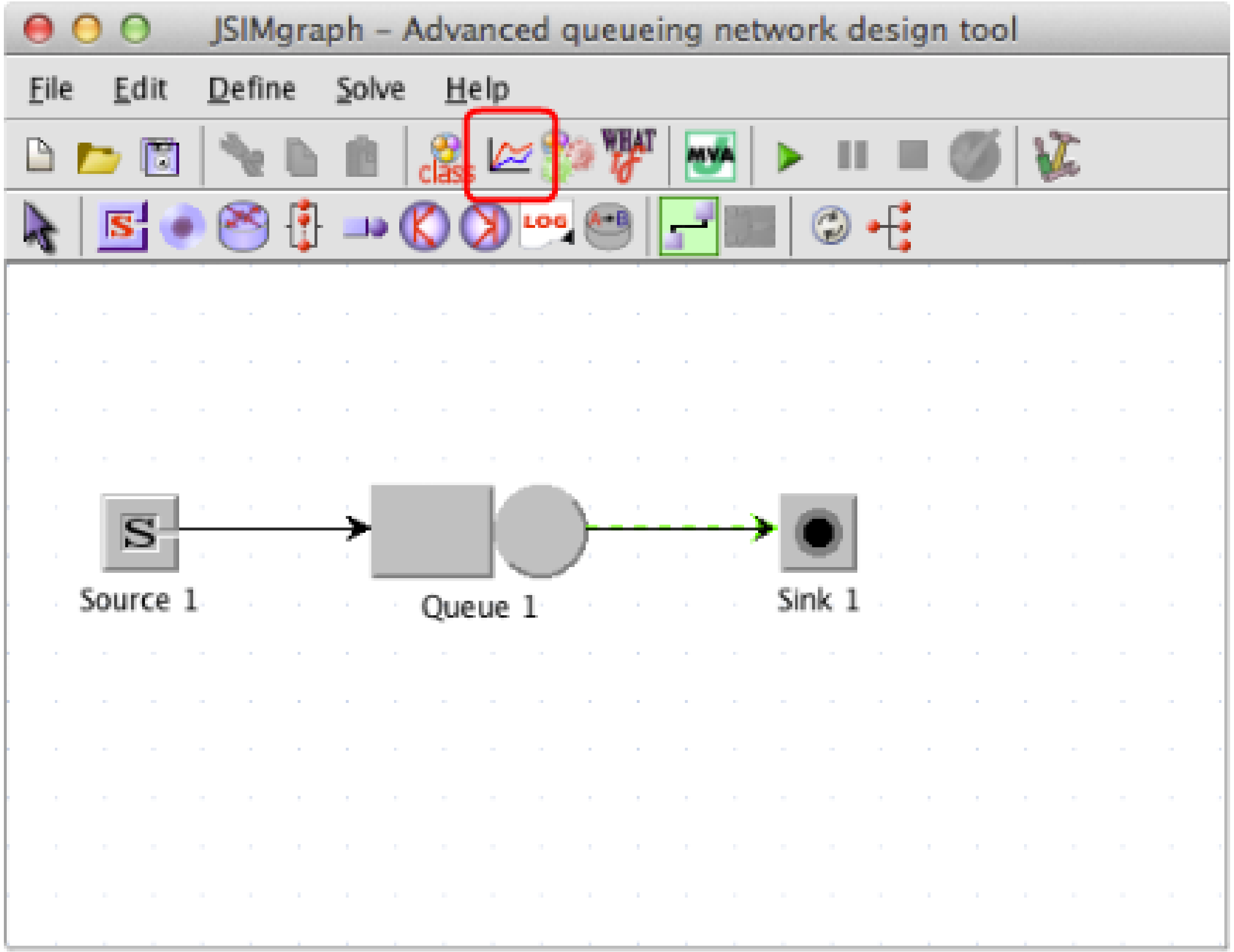
Tool support

The queuing disciplines supported by JMT includes FCFS, LCFS (both preemptive and non-preemptive) and various flavors of PS.



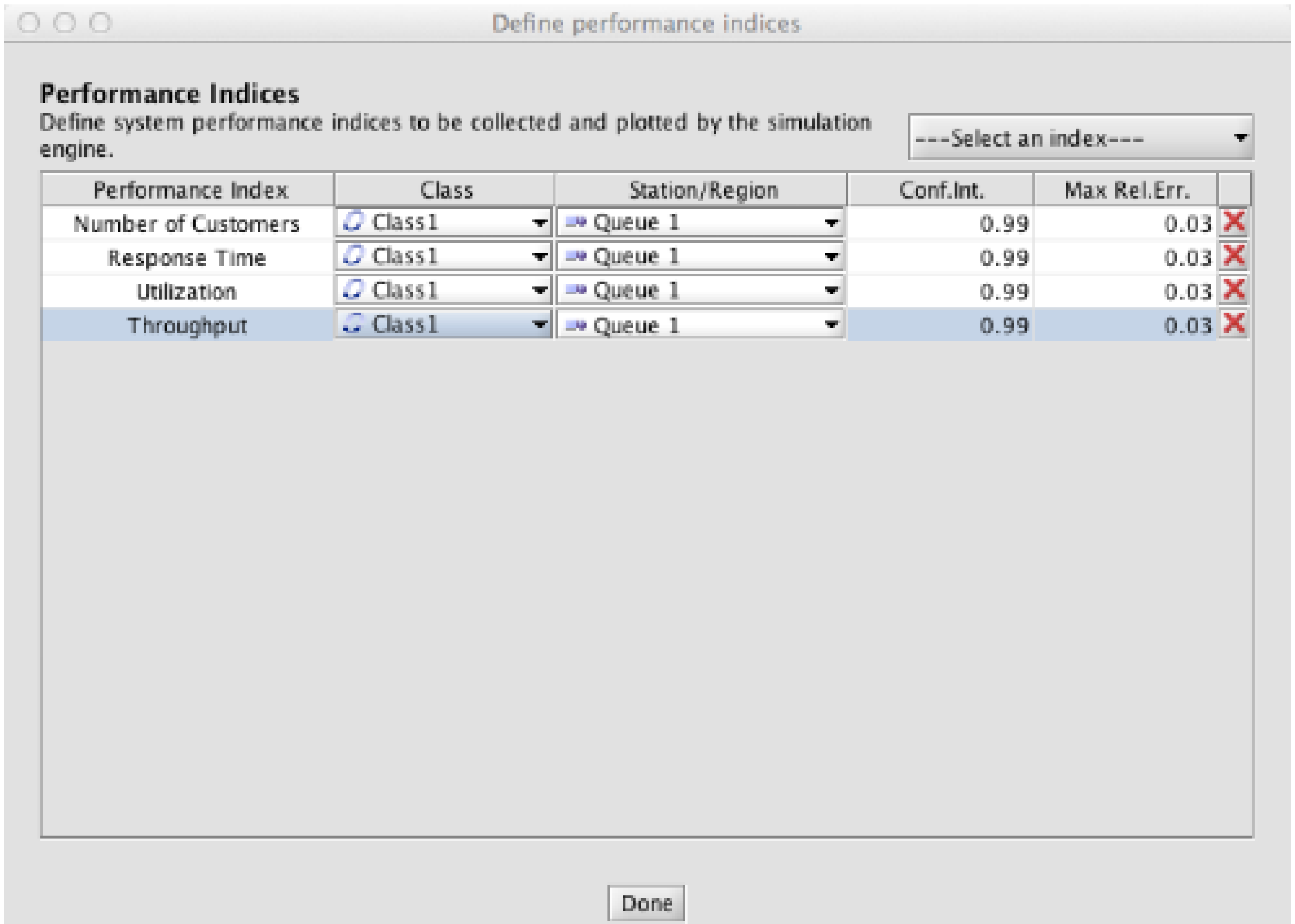
Single queues in JMT

In order to produce the results, *Performance Indices* must be defined.



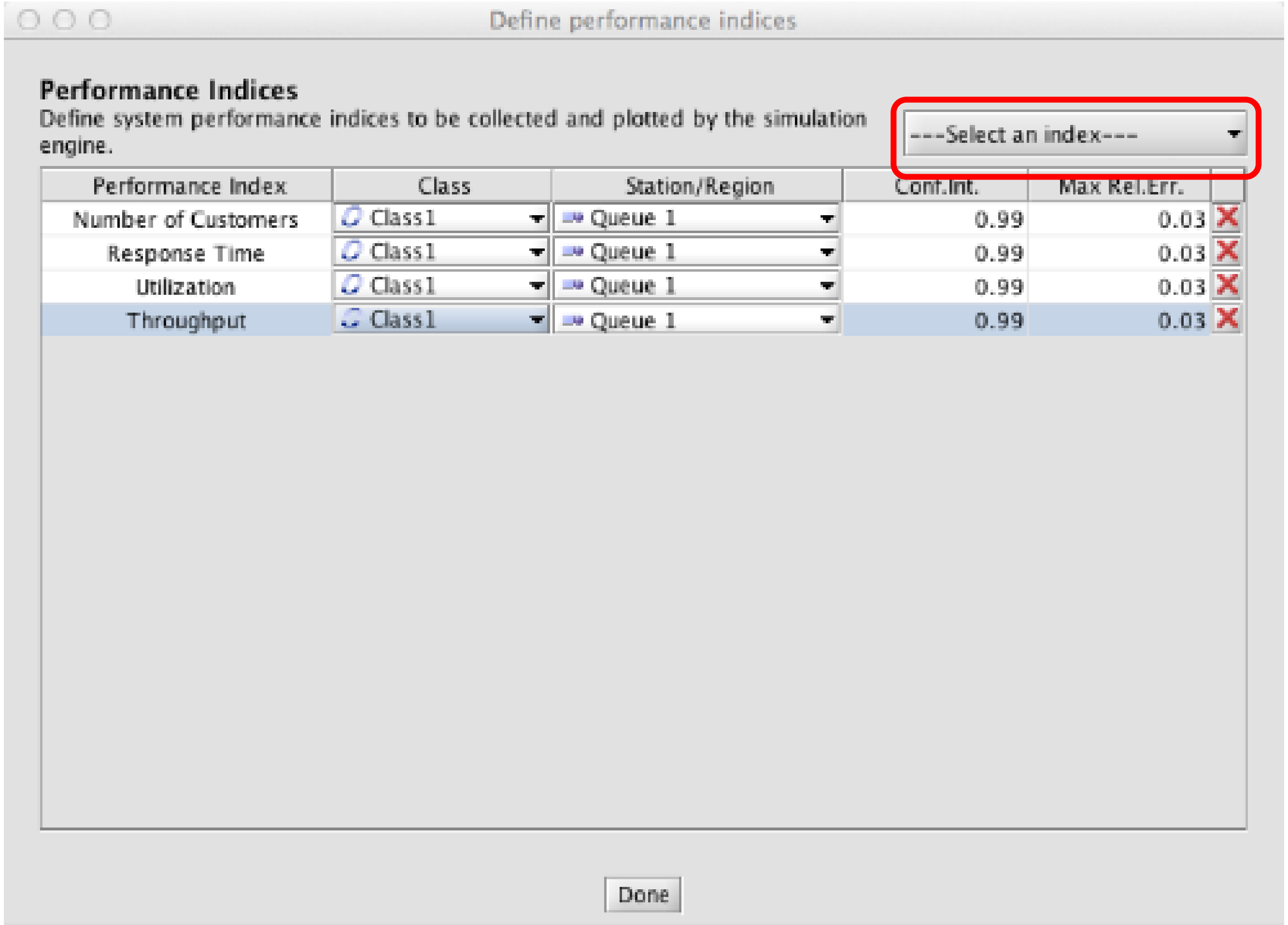
Single queues in JMT

Performance indices include average number of jobs, response time, utilization and throughput.



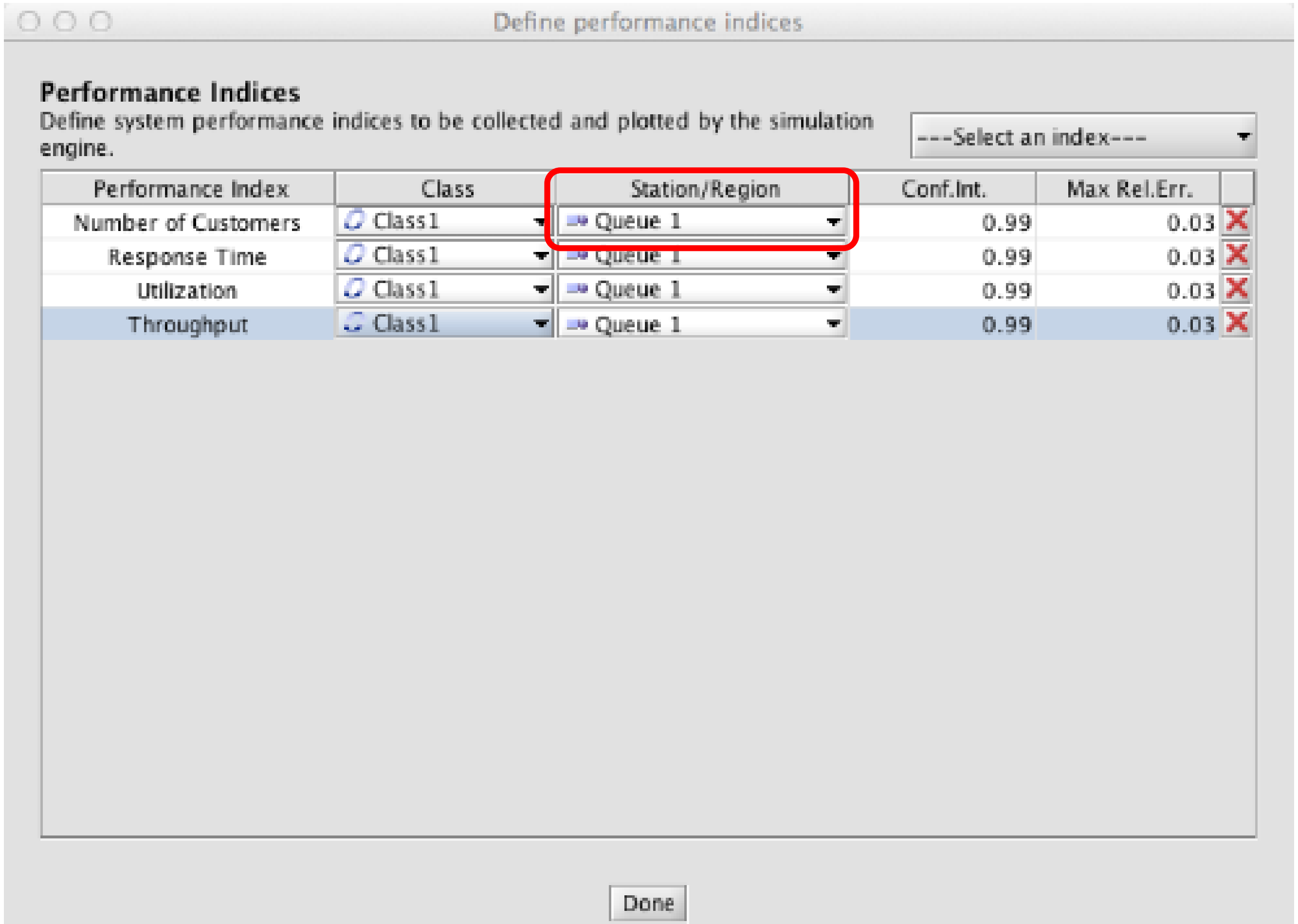
Single queues in JMT

They are selected from the drop down menu on the top-right corner of the window…



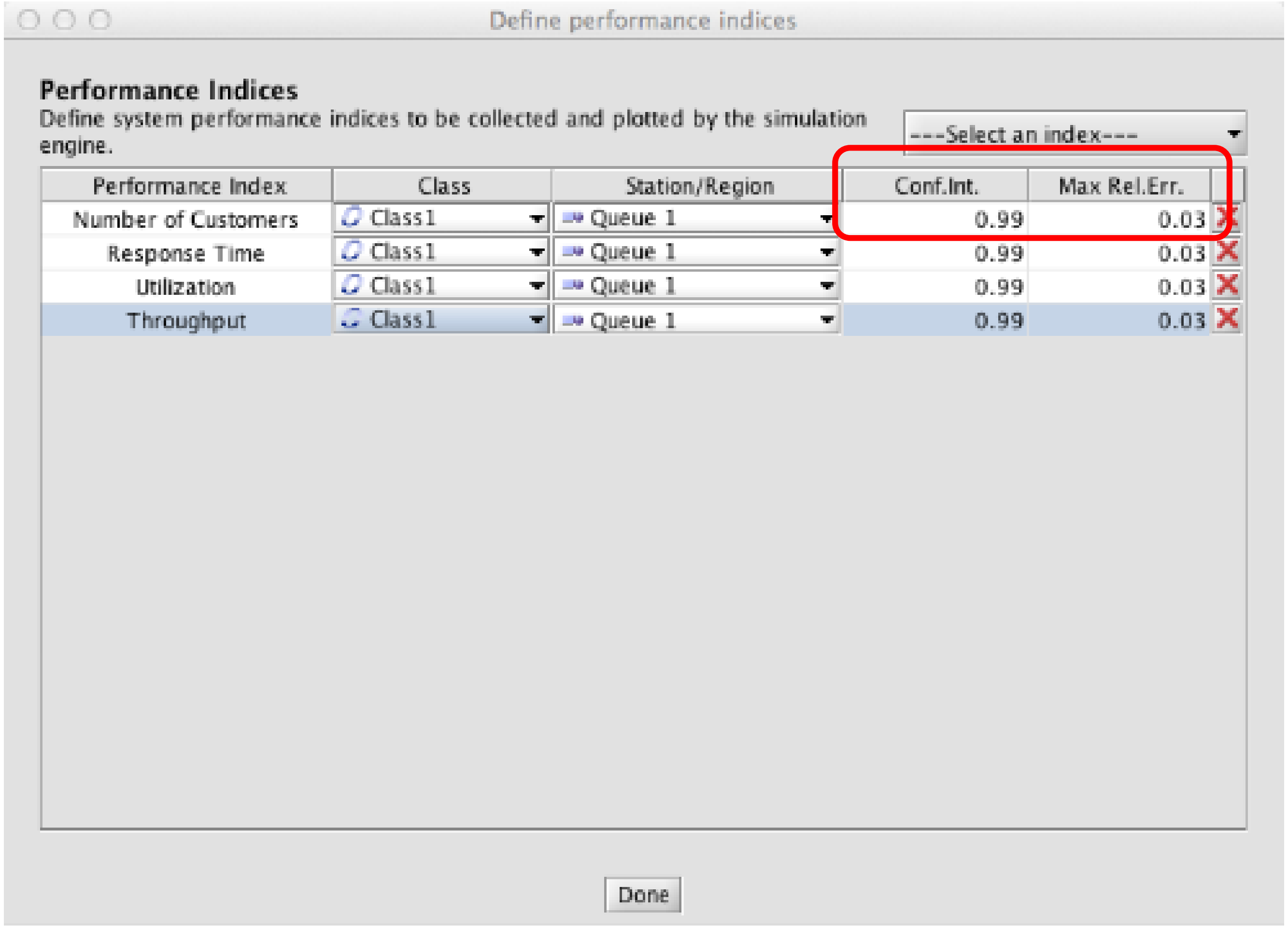
Single queues in JMT

… and further configured in the central columns. In particular, the centre column can limit the metrics to the selected nodes only.



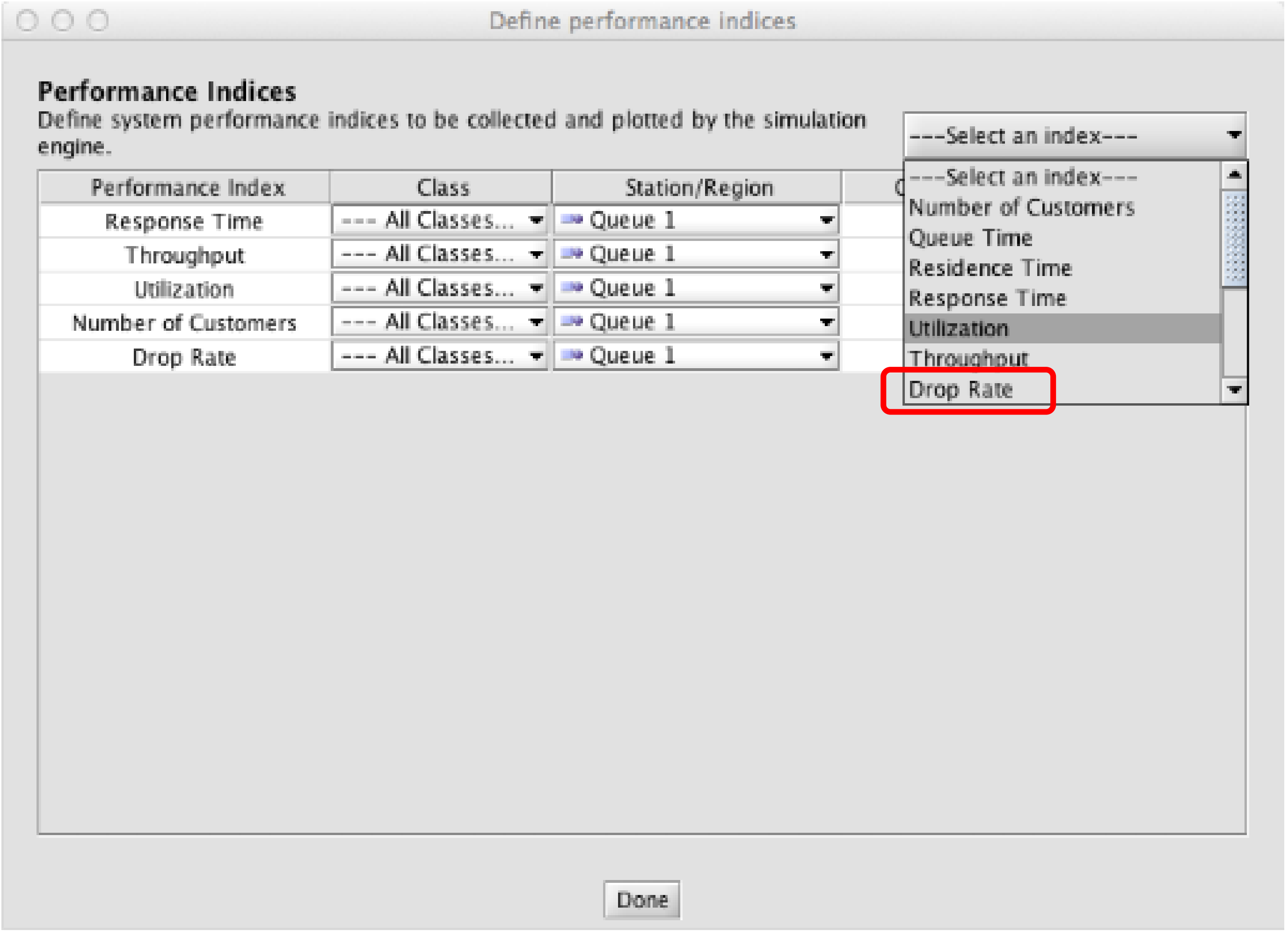
Single queues in JMT

For each index, a different *Confidence Level* and *Maximum Relative Error* can be defined.



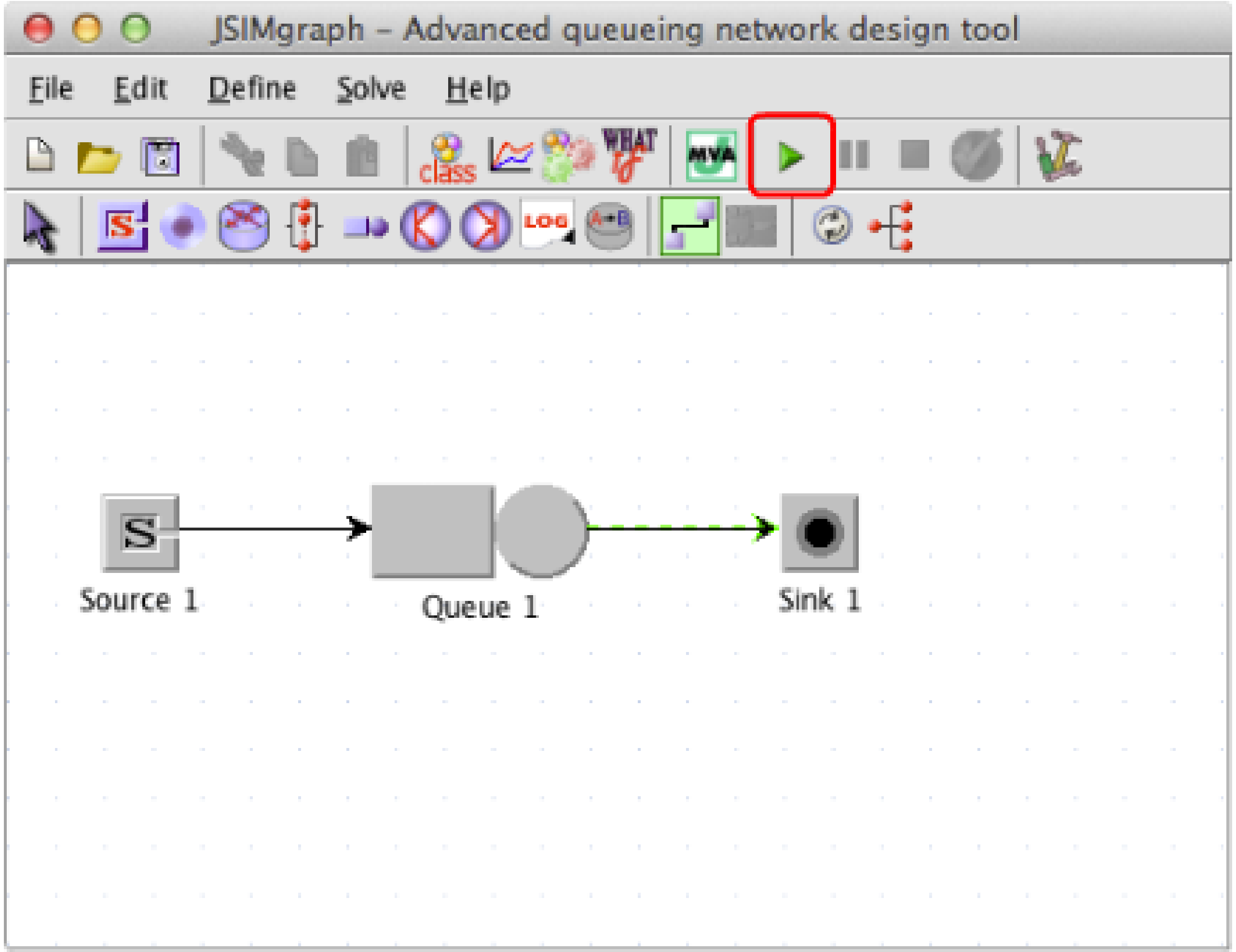
Finite capacity: losses

For model model with losses, the **drop rate** can be computed as well.



Single queues in JMT

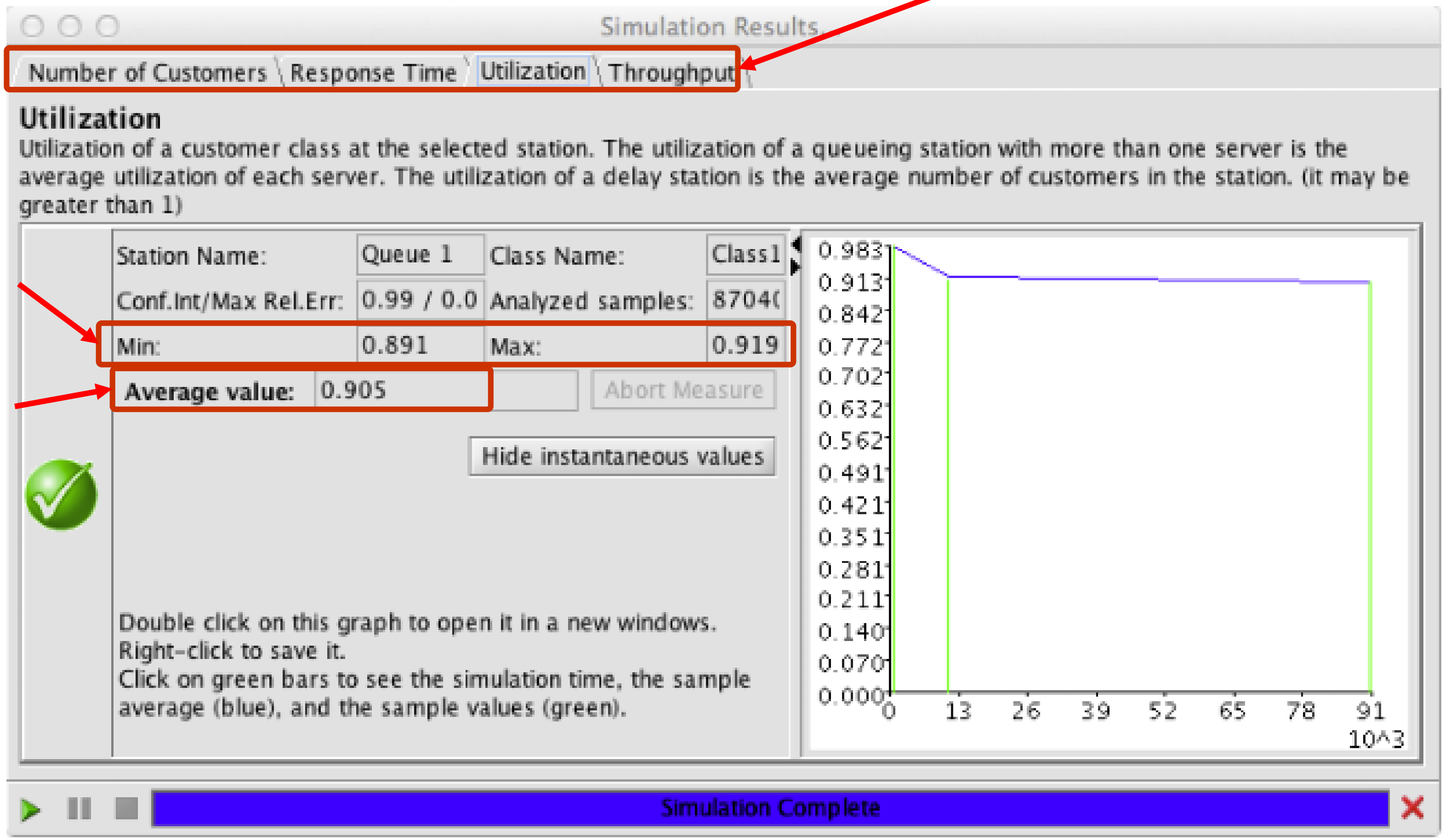
Simulation can be run with the *"Play"* button.



Single queues in JMT

At the end of the simulation, JMT outputs the values of the selected performance indices.

Tabs on the top selects the different types of performance indices



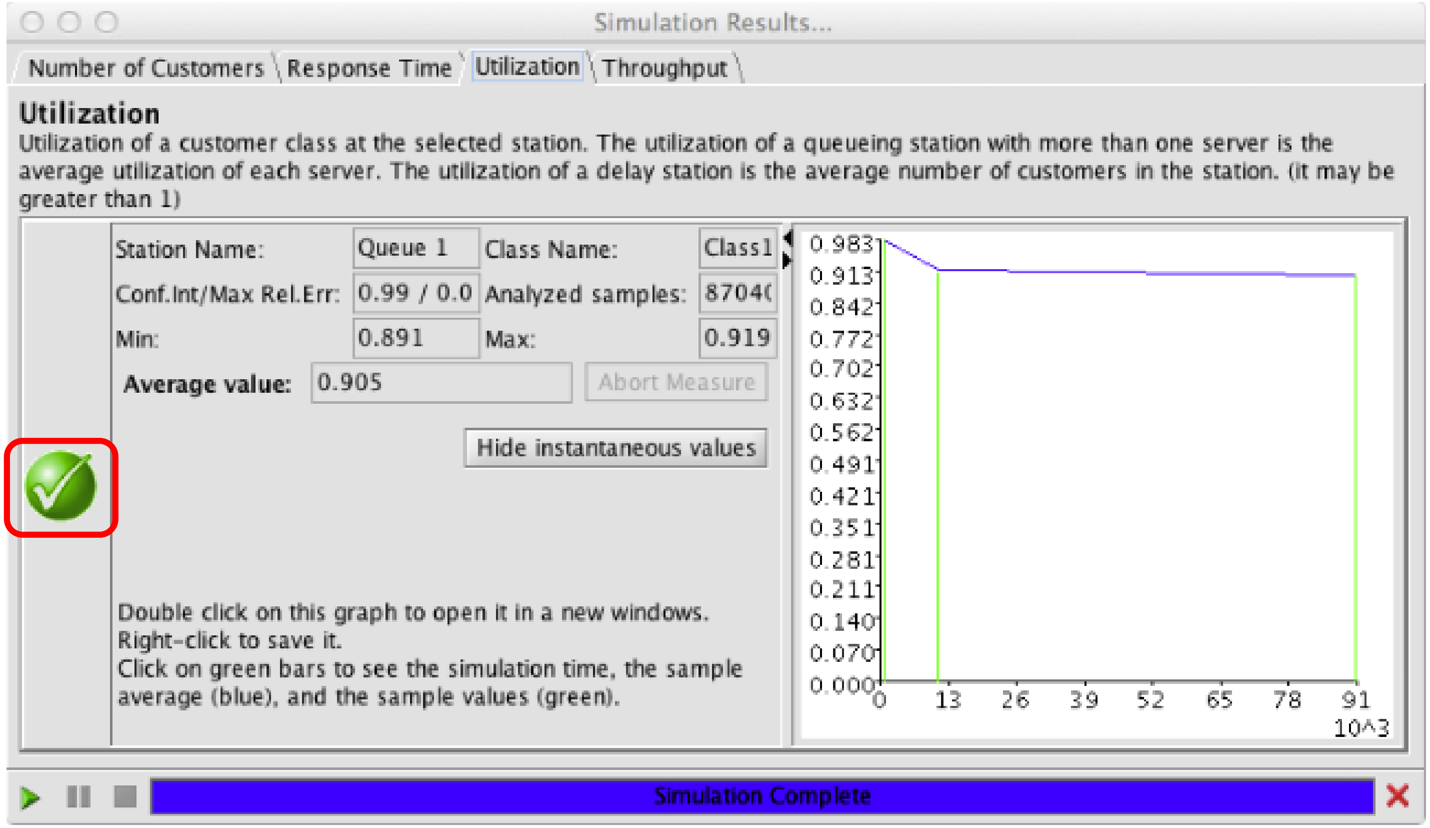
Confidence interval

Average value

Single queues in JMT

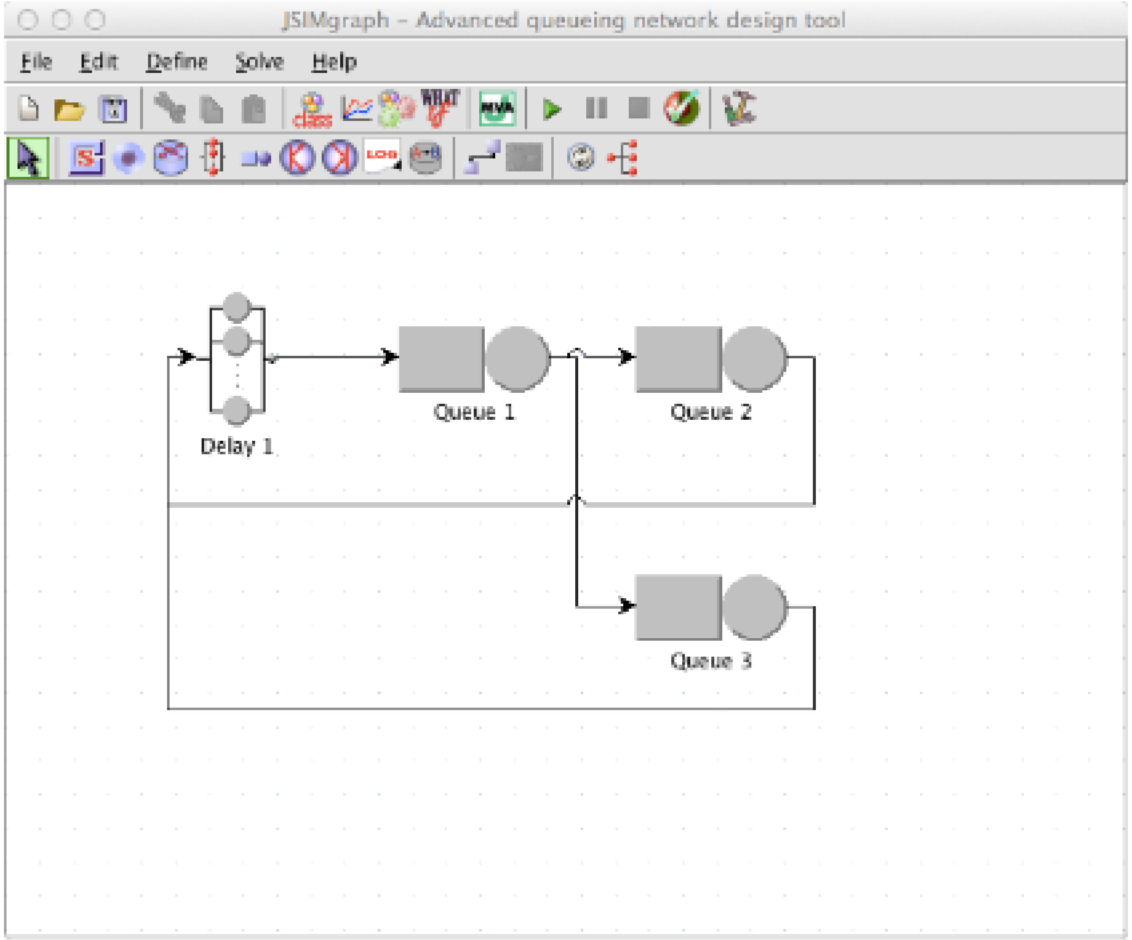
Simulation ends either when all the confidence intervals have reached the desired level of accuracy, or when a limit condition (user definable) has been reached.

A green icon denotes that the desired accuracy was reached.



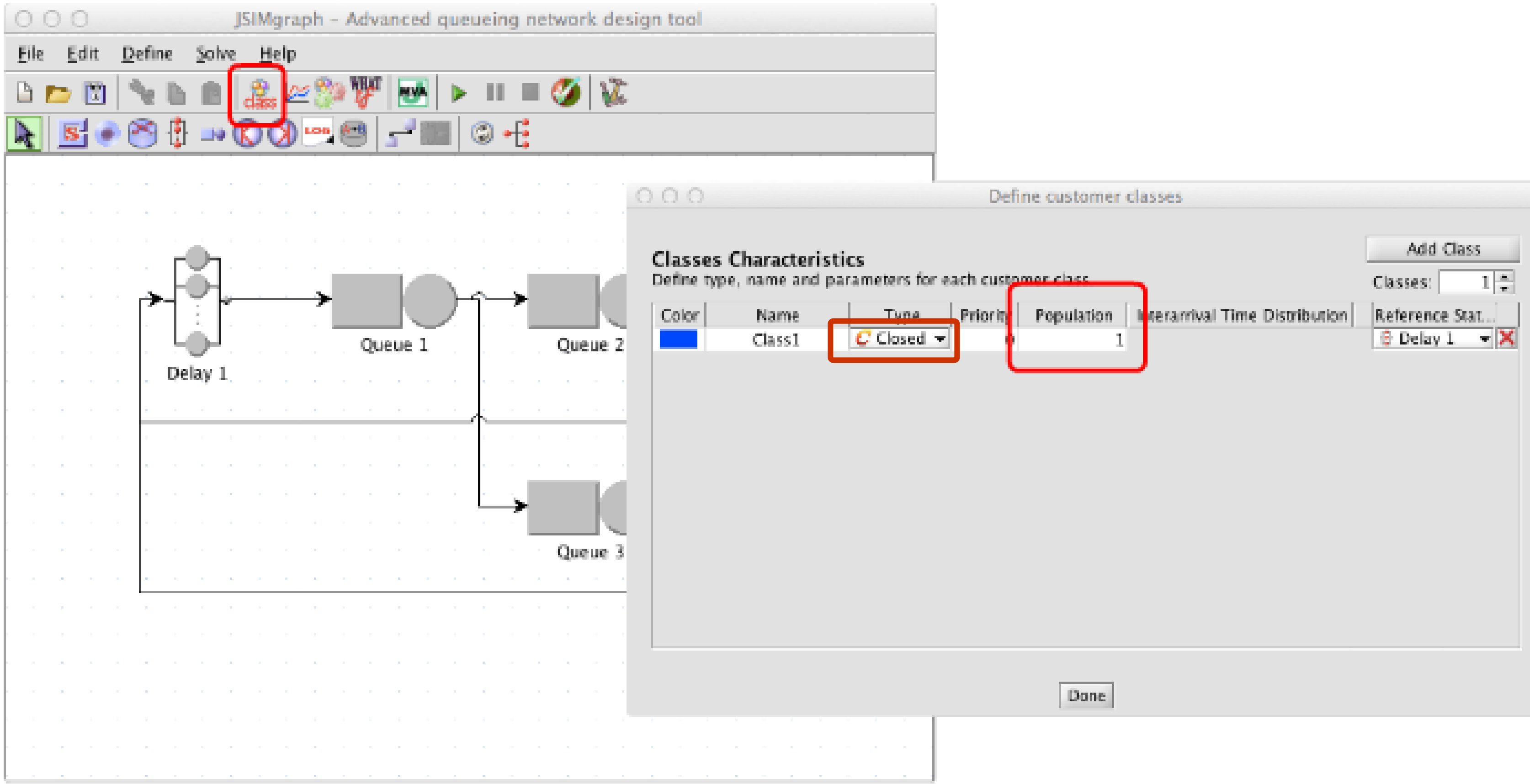
Closed models in JMT

Closed models do not use sources or sinks. Instead they are characterized by arcs that creates loops between the stations.



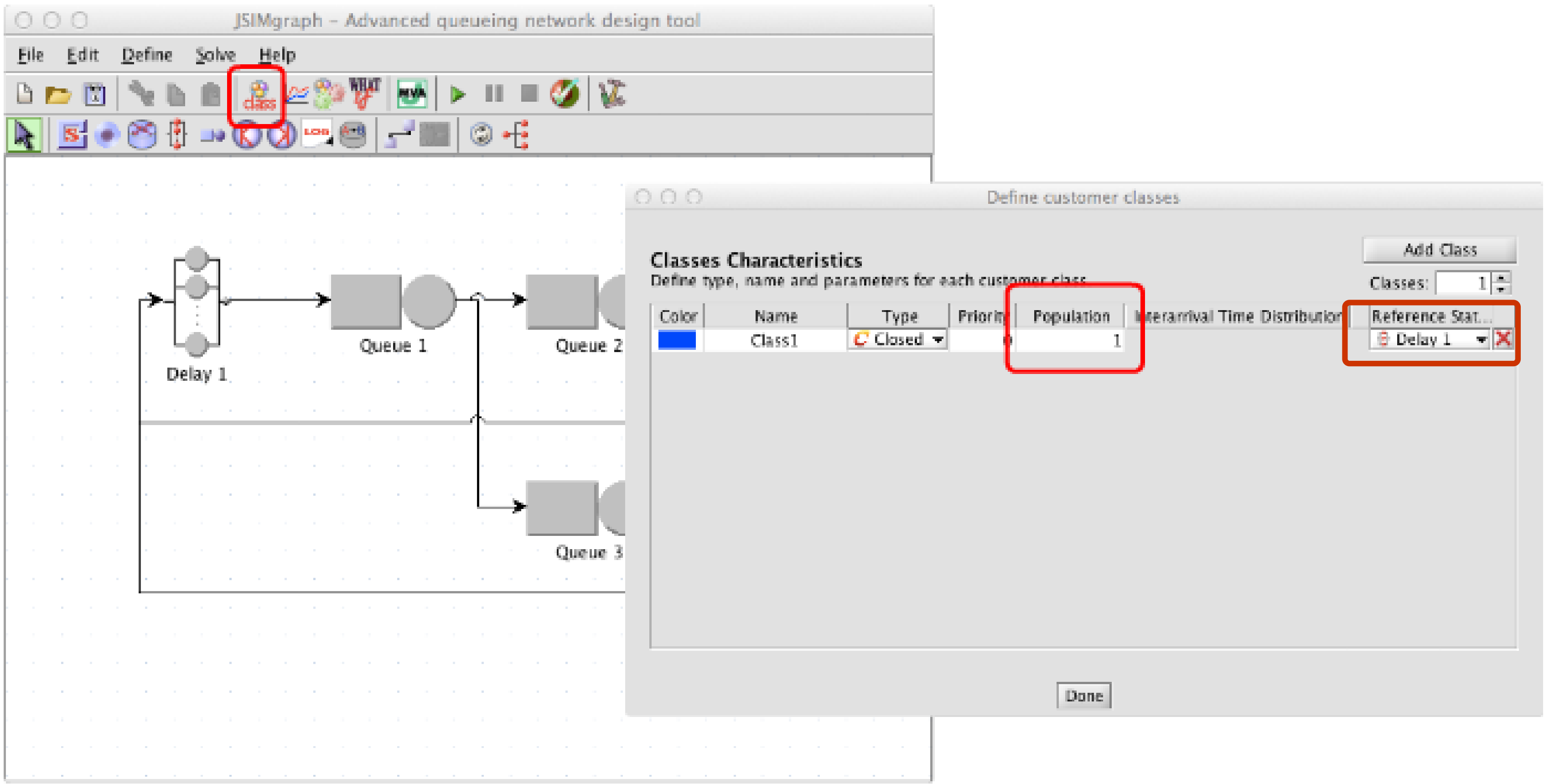
Closed models in JMT

In this case, the type "closed" should be specified for their corresponding class, and the total population is defined in the class definition tab of the model.



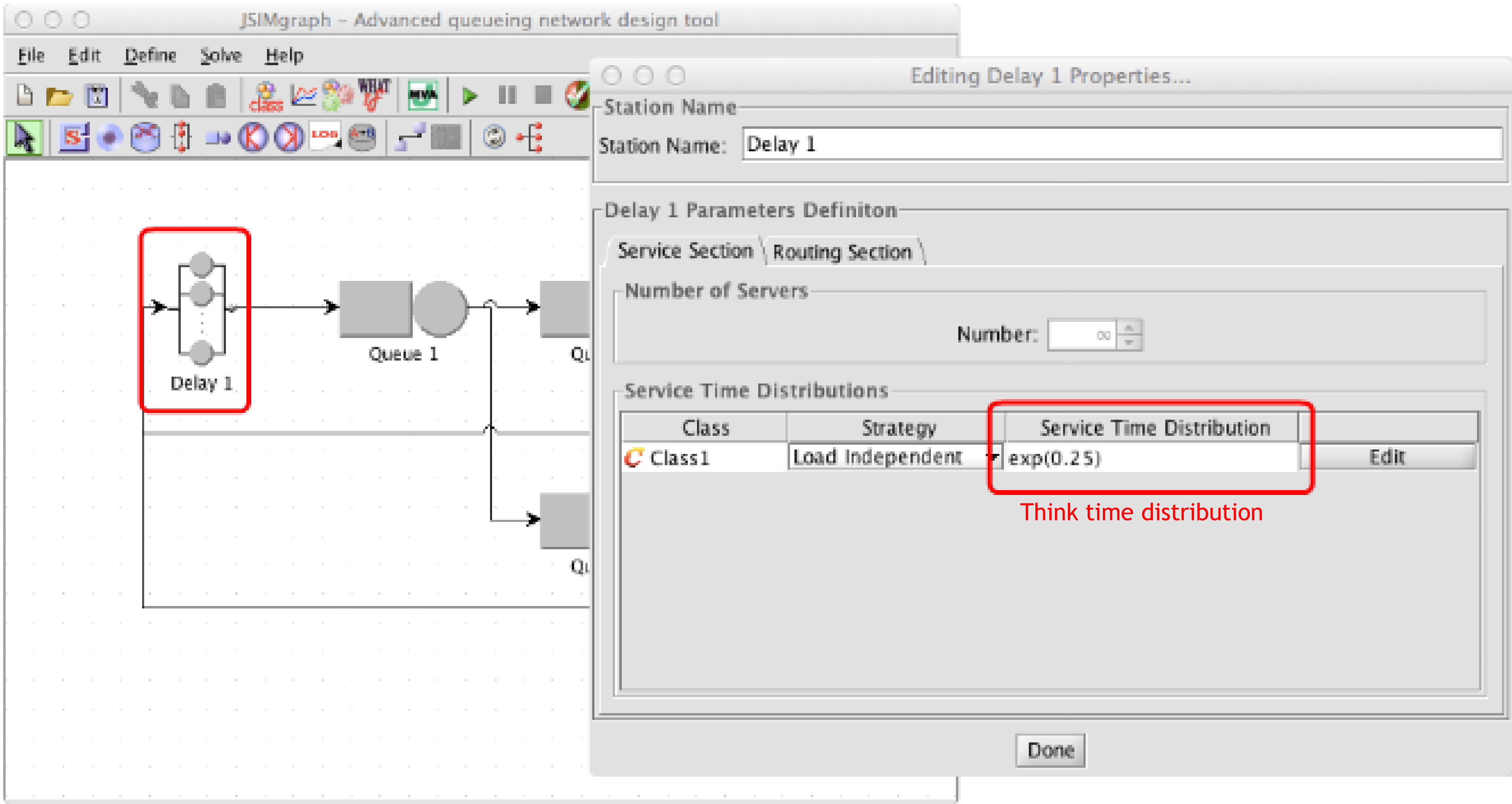
Closed models in JMT

Reference station must also be defined in the corresponding column of the class panel.



Closed models in JMT

For time-sharing systems, the "terminal station" is inserted by a delay station, whose service time corresponds to the think time *Z*.



Response times in closed models

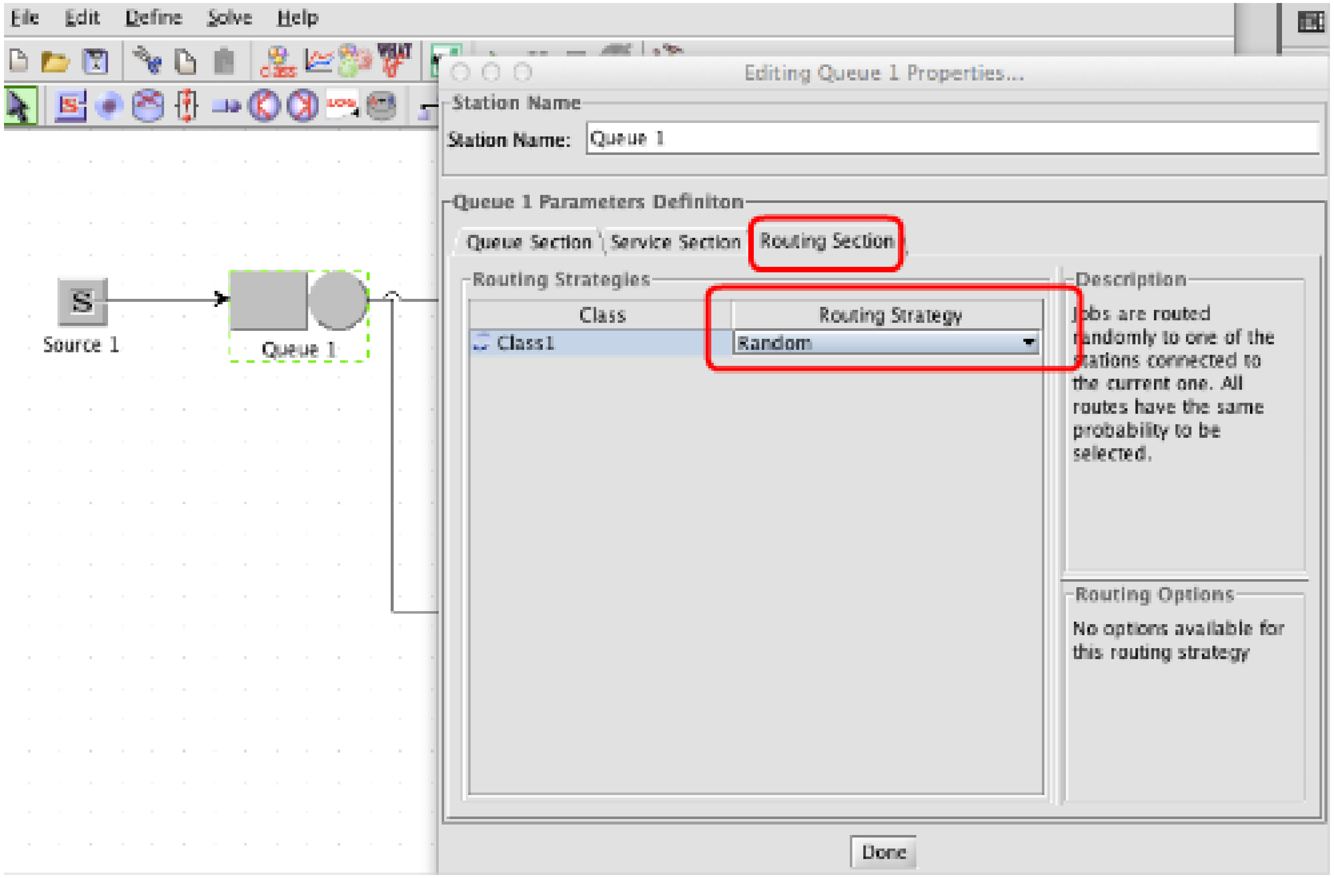
Please note that system response times in time-sharing closed models, always include the time spent in the terminal station. To obtain the same system response time proposed in the *Response Time Law* from the book of Lazoswka, you have to manually remove the average think time.

$$R_{Tot} = R_{Sys} + Z \qquad N = X \cdot R_{Tot} \qquad N = X \cdot \left(R_{Sys} + Z\right)$$

$$\boxed{R_{Sys} = R_{Tot} - Z}$$

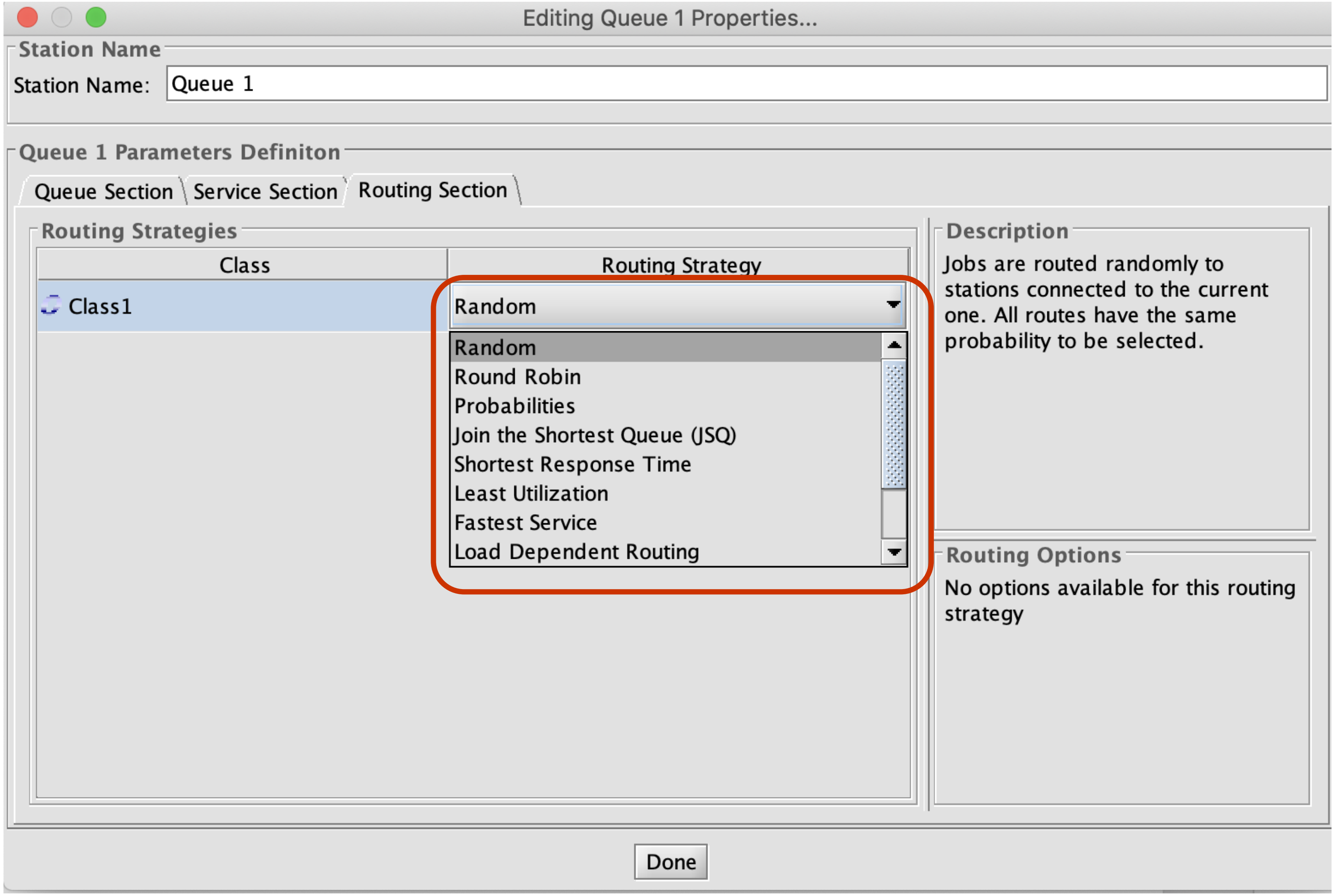
Routing in JMT

Every node has a property page called *routing section*. This allows embedding the definition of the routing policy in queues and other modeling elements, and it is used whenever more than one output arc is connected.



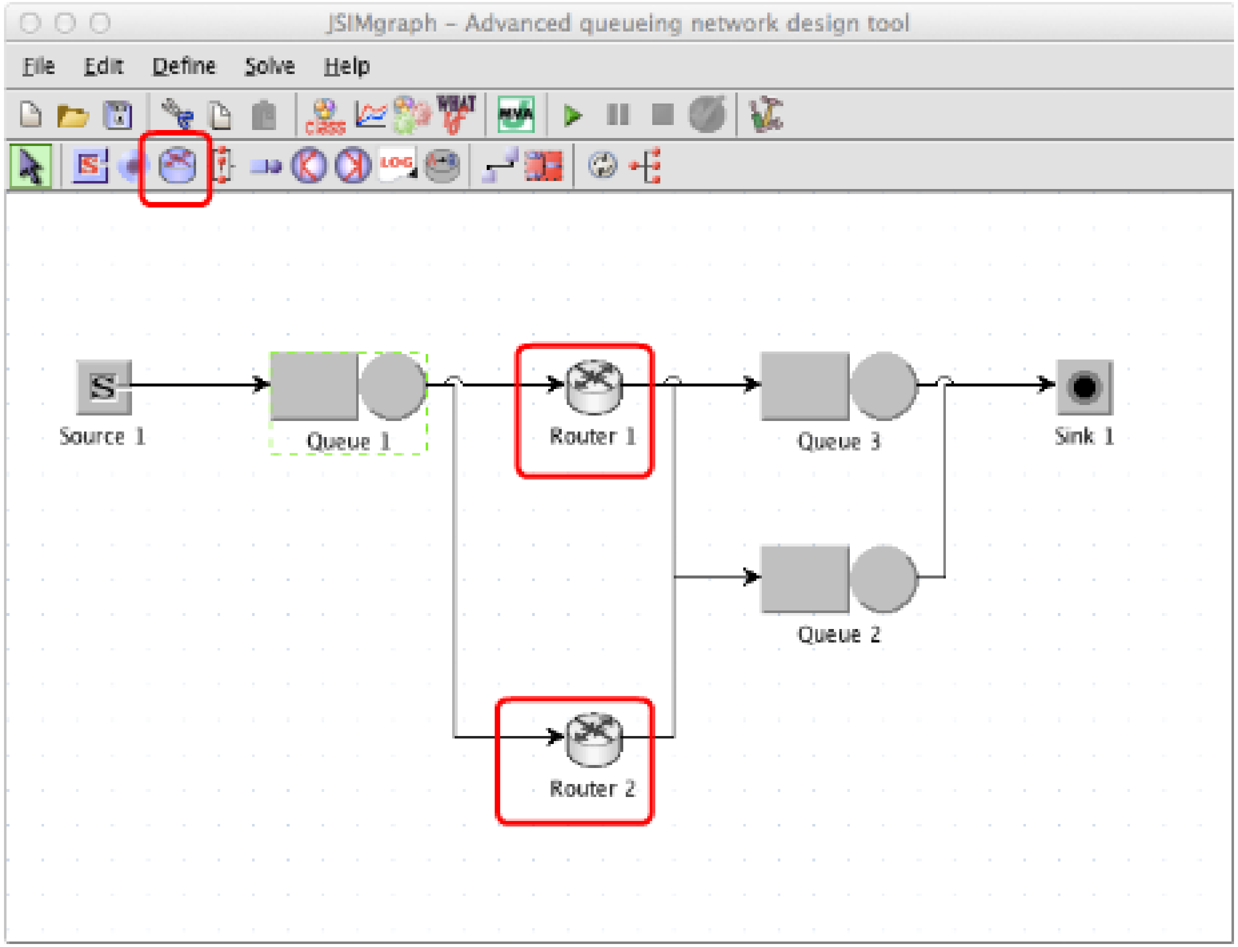
Routing in JMT

Several routing mechanisms are available:



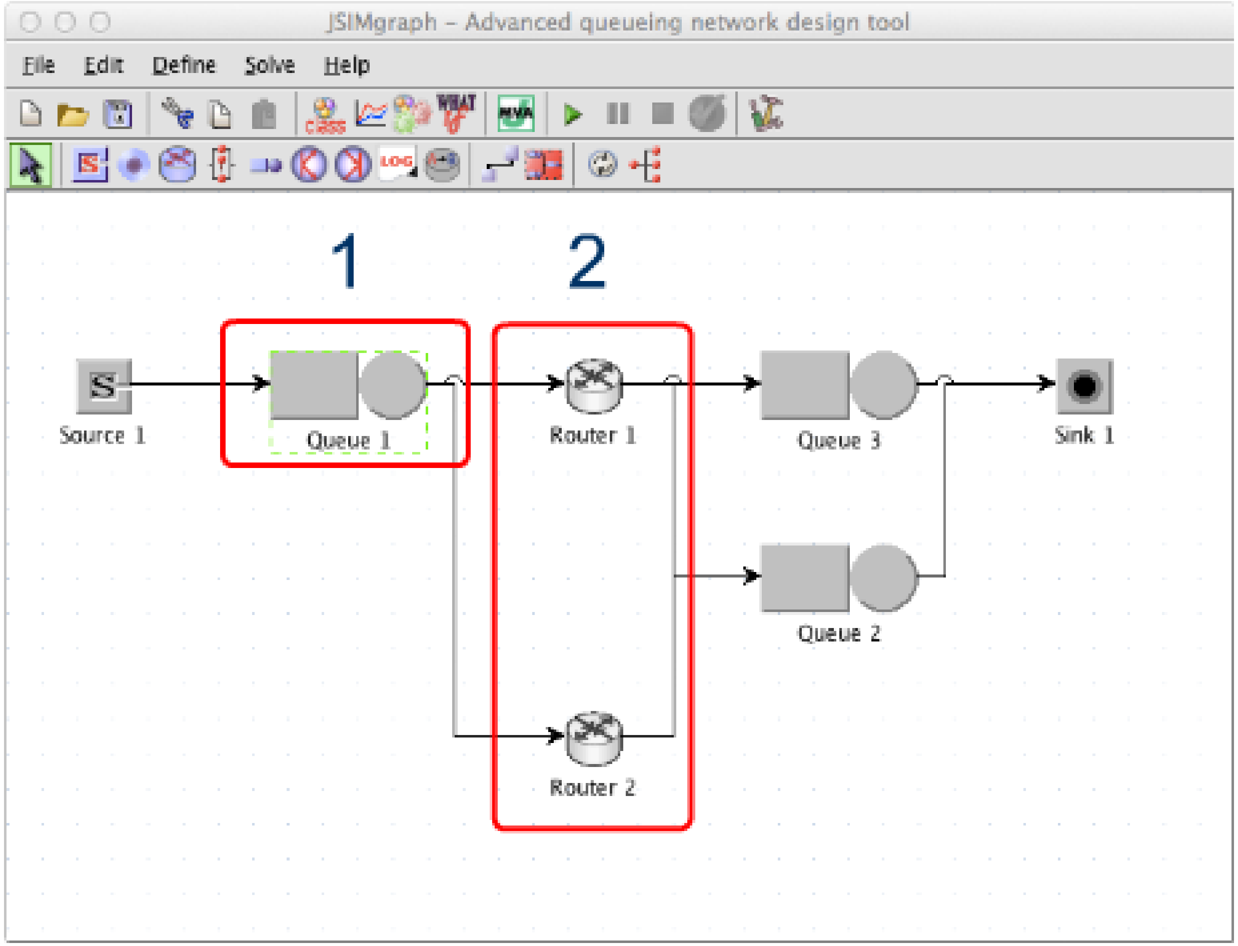
Routing in JMT

JMT also allows including in a model specific components called *Routers*, whose properties are characterized only by the routing component page.



Routing in JMT

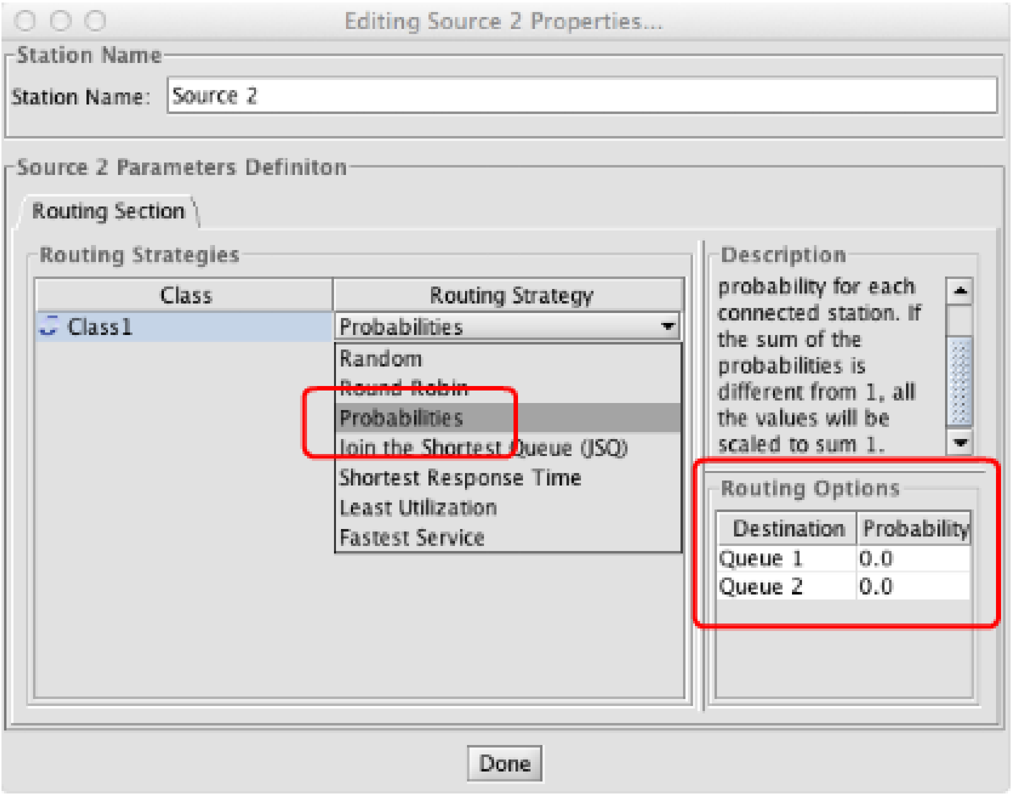
The inclusion of the routing component allows creating hierarchical policies, where different algorithms can be combined to create more complex job assignments.



Probabilistic routing

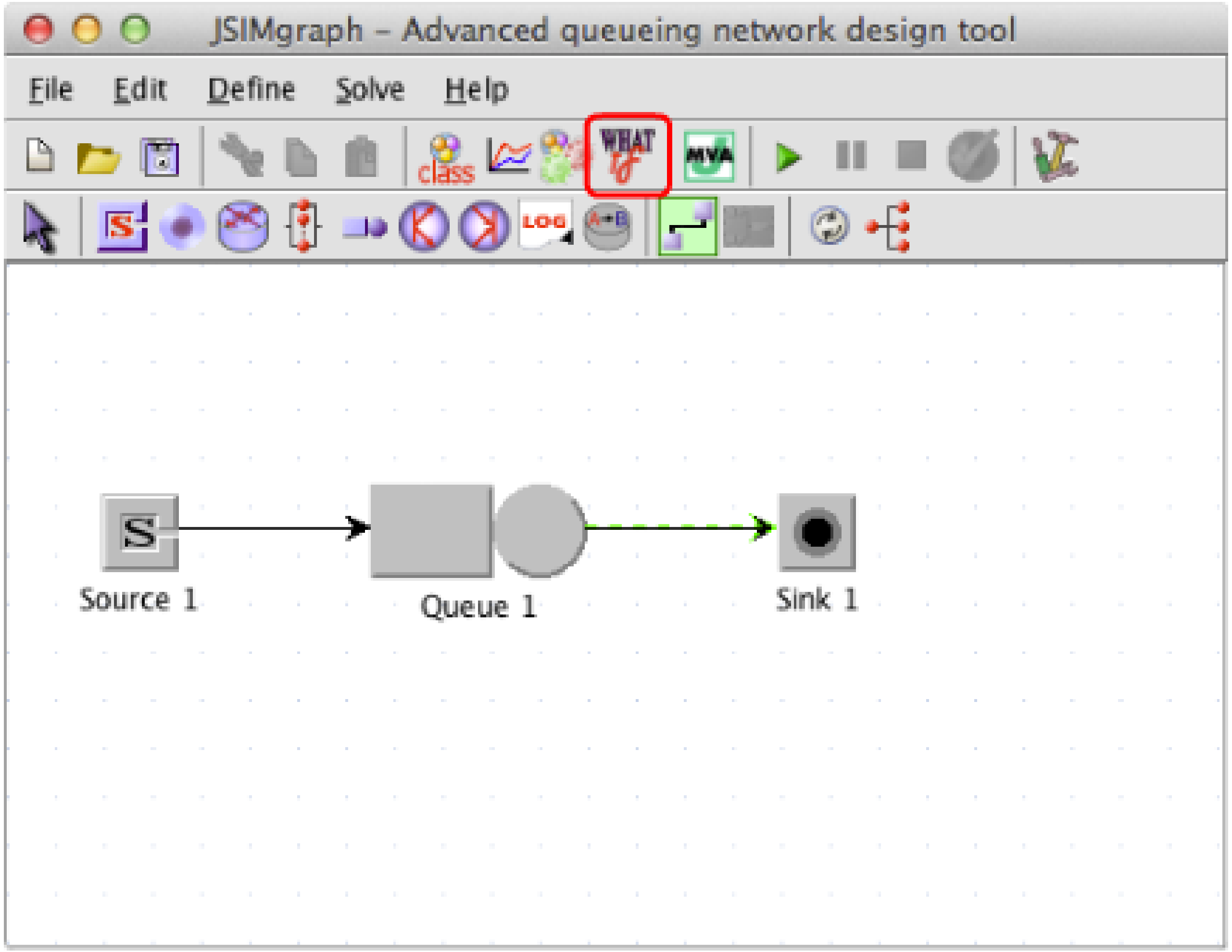
The "probabilities" routing strategy corresponds to probabilistic routing.

An additional panel, which appears in the bottom right of the window, allows the user to associate a different probability to each of the nodes connected to the output of the selected queue.



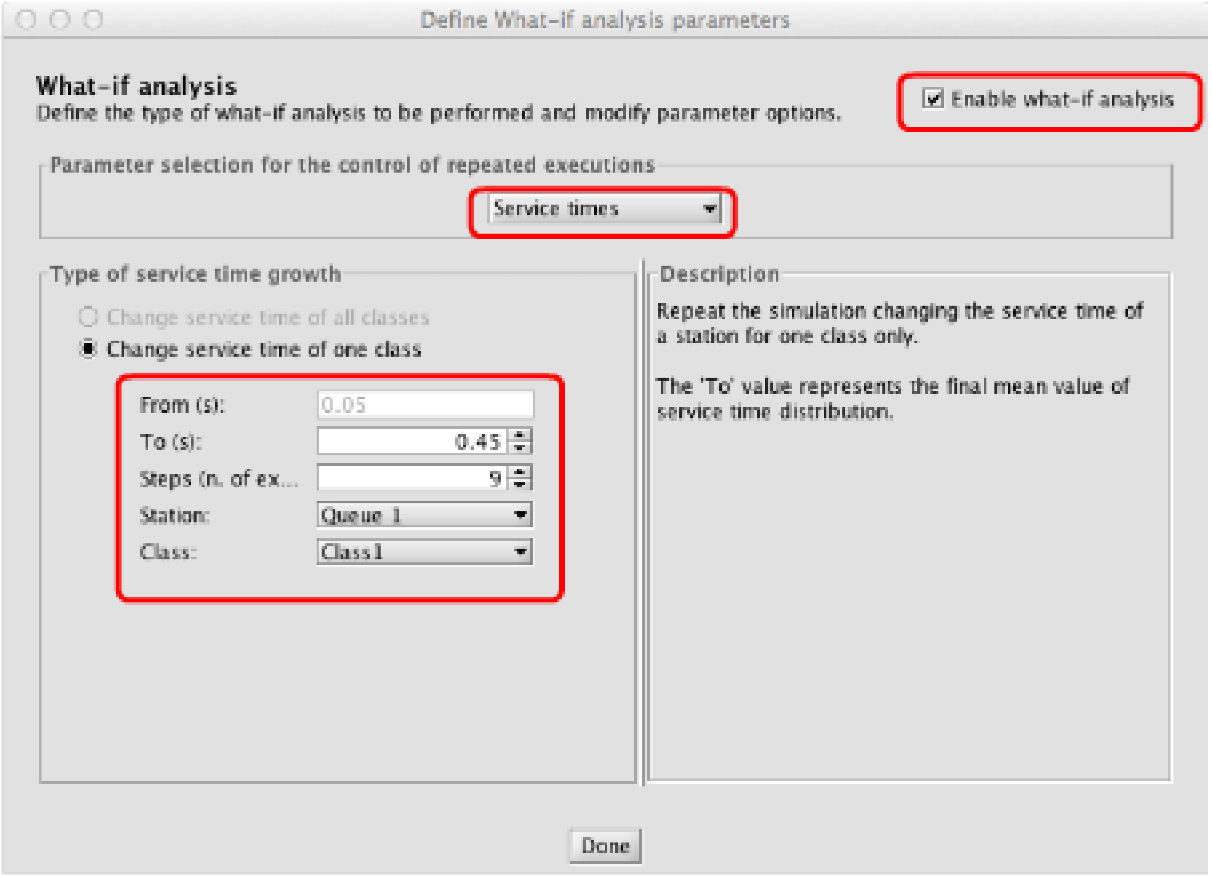
What-if analysis

What-if analysis is available also in *JSimGraph*, with a specific button.



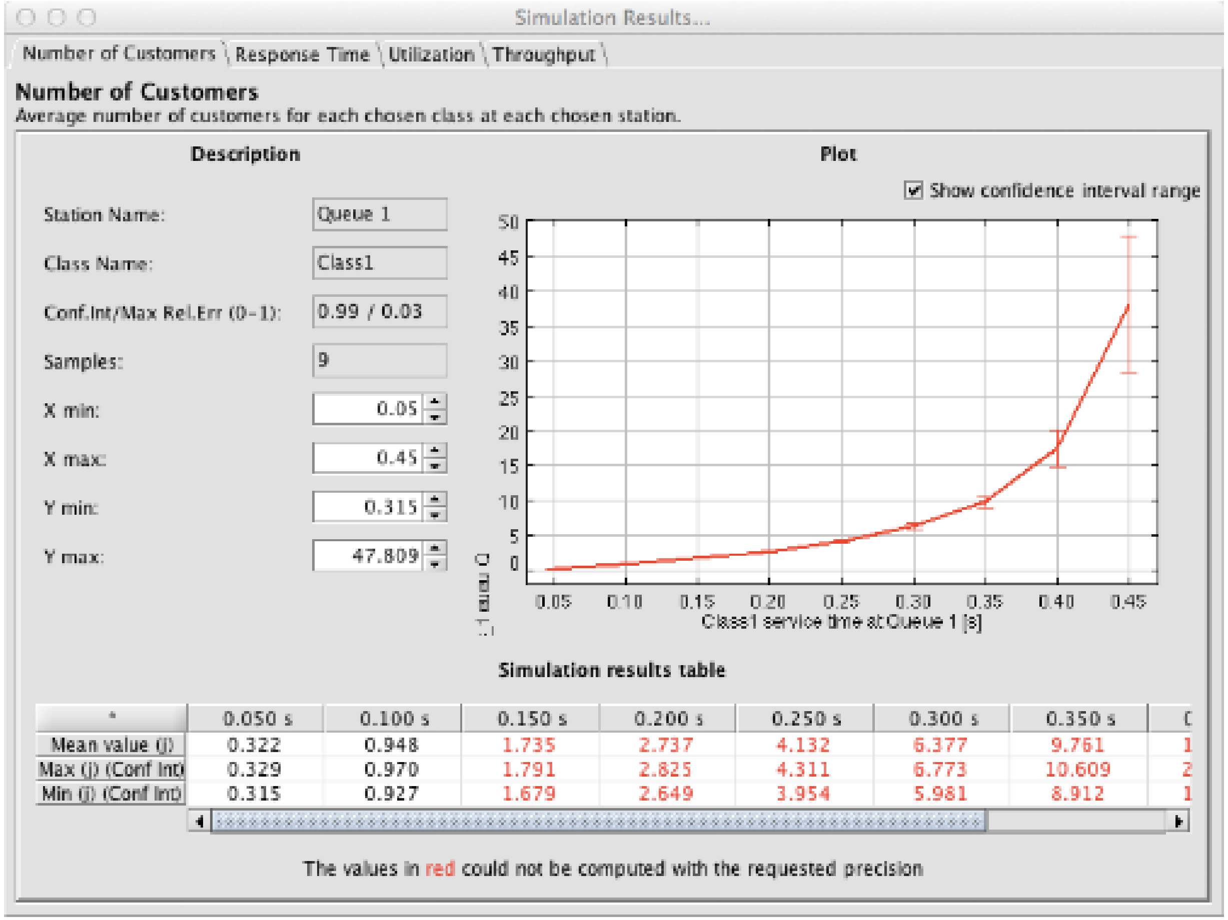
What-if analysis

What-if must be activated, and the user must select: the value that will be varied, its range and the number of steps that will be performed.



What-if analysis

In this case results can be plotted against the parameter that has been varied during the experiment.



*	0.050 s	0.100 s	0.150 s	0.200 s	0.250 s	0.300 s	0.350 s
Mean value (J)	0.322	0.948	1.735	2.737	4.132	6.377	9.761
Max (J) (Conf Int)	0.329	0.970	1.791	2.825	4.311	6.773	10.609
Min (J) (Conf Int)	0.315	0.927	1.679	2.649	3.954	5.981	8.912