

Homework 6

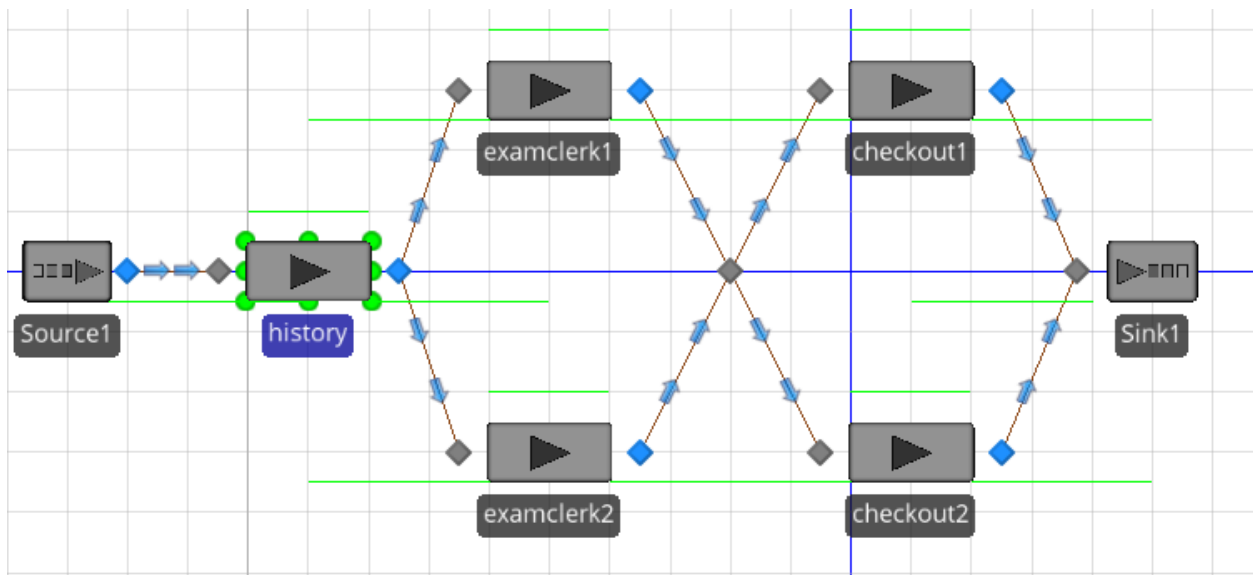
Max Wagner

April 12, 2016

My other classes put off due dates an extra week due to break, I assumed the same. Didn't mean it to be late.

1a.

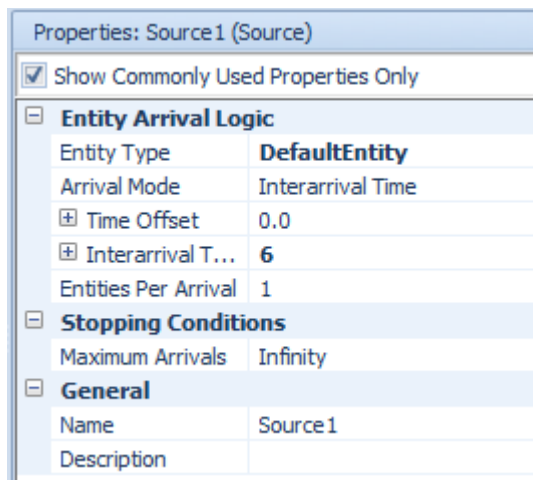
In this case, there is one source, 5 servers (a single clerk getting the driving history, two clerks each giving a written exam in parallel, and two checkout computers in parallel), and 1 sink.



1b.

Source:

Interval arrival time set to 6 minutes



Path Source -> History:

Properties: Path1 (Path)	
<input type="checkbox"/> Show Commonly Used Properties Only	
Travel Logic	
Type	Unidirectional
Initial Traveler ...	Infinity
Entry Ranking ...	First In First Out
Drawn To Scale	True
Allow Passing	True
+ Speed Limit	Infinity
Routing Logic	
Selection Weight	1.0
+ State Assignments	
+ Add-On Process Triggers	
+ Advanced Options	
+ General	

History:

A process time of 5, and a outemp2ound rule to link weight, so there is a 50/50 chance of where they go.

Properties: history (Server)	
<input type="checkbox"/> Show Commonly Used Properties Only	
Process Logic	
Capacity Type	Fixed
Initial Capacity	1
Ranking Rule	First In First Out
Dynamic Selecti...	None
+ Transfer-In ...	0.0
Process Type	Specific Time
- Processing Ti...	5
Units	Minutes

Properties: Output@history (TransferNode)	
<input type="checkbox"/> Show Commonly Used Properties Only	
Crossing Logic	
Initial Traveler ...	Infinity
Entry Ranking ...	First In First Out
Routing Logic	
Outbound Trav...	Continue
Outbound Link ...	Any
Outbound Link ...	By Link Weight
Entity Destinati...	Continue
+ Other Routing Out Options	
Transport Logic	
Ride On Transp...	False

Path History -> ExamClerk:

Properties: Path2 (Path)

☐ Show Commonly Used Properties Only

Travel Logic	
Type	Unidirectional
Initial Traveler ...	Infinity
Entry Ranking ...	First In First Out
Drawn To Scale	True
Allow Passing	True
<input type="checkbox"/> Speed Limit	Infinity
Routing Logic	
Selection Weight	1.0

ExamClerk 1/2:

Set process time to 8.8 minutes

Properties: examclerk1 (Server)

☐ Show Commonly Used Properties Only

Process Logic	
Capacity Type	Fixed
Initial Capacity	1
Ranking Rule	First In First Out
Dynamic Selecti...	None
<input type="checkbox"/> Transfer-In ...	0.0
Process Type	Specific Time
<input type="checkbox"/> Processing Ti...	8.8
Units	Minutes

Path ExamClerk -> Node:

Properties: Path4 (Path)

☐ Show Commonly Used Properties Only

Travel Logic	
Type	Unidirectional
Initial Traveler ...	Infinity
Entry Ranking ...	First In First Out
Drawn To Scale	True
Allow Passing	True
<input type="checkbox"/> Speed Limit	Infinity
Routing Logic	
Selection Weight	1.0

Node:

Again set by link weight

Properties: BasicNode1 (BasicNode)

☐ Show Commonly Used Properties Only

Crossing Logic

Initial Traveler ...	Infinity
Entry Ranking ...	First In First Out

Routing Logic

Outbound Trav...	Continue
Outbound Link ...	Any
Outbound Link ...	By Link Weight

☐ Other Routing Out Options

Path Node -> Checkout 1/2:

Properties: Path7 (Path)

☐ Show Commonly Used Properties Only

Travel Logic

Type	Unidirectional
Initial Traveler ...	Infinity
Entry Ranking ...	First In First Out
Drawn To Scale	True
Allow Passing	True
<input type="checkbox"/> Speed Limit	Infinity

Routing Logic

Selection Weight	1.0
------------------	-----

Checkout 1/2:

Process time of 9 minutes

Properties: checkout1 (Server)

☐ Show Commonly Used Properties Only

Process Logic

Capacity Type	Fixed
Initial Capacity	1
Ranking Rule	First In First Out
Dynamic Selecti...	None
<input type="checkbox"/> Transfer-In ...	0.0
Process Type	Specific Time
<input type="checkbox"/> Processing Ti...	9

Path Checkout 1/2 to Sink:

Properties: Path8 (Path)

☐ Show Commonly Used Properties Only

Travel Logic

Type	Unidirectional
Initial Traveler ...	Infinity
Entry Ranking ...	First In First Out
Drawn To Scale	True
Allow Passing	True
<input type="checkbox"/> Speed Limit	Infinity

Sink:

Properties: Sink1 (Sink)

☐ Show Commonly Used Properties Only

Process Logic

☐ Transfer-In ... 0.0

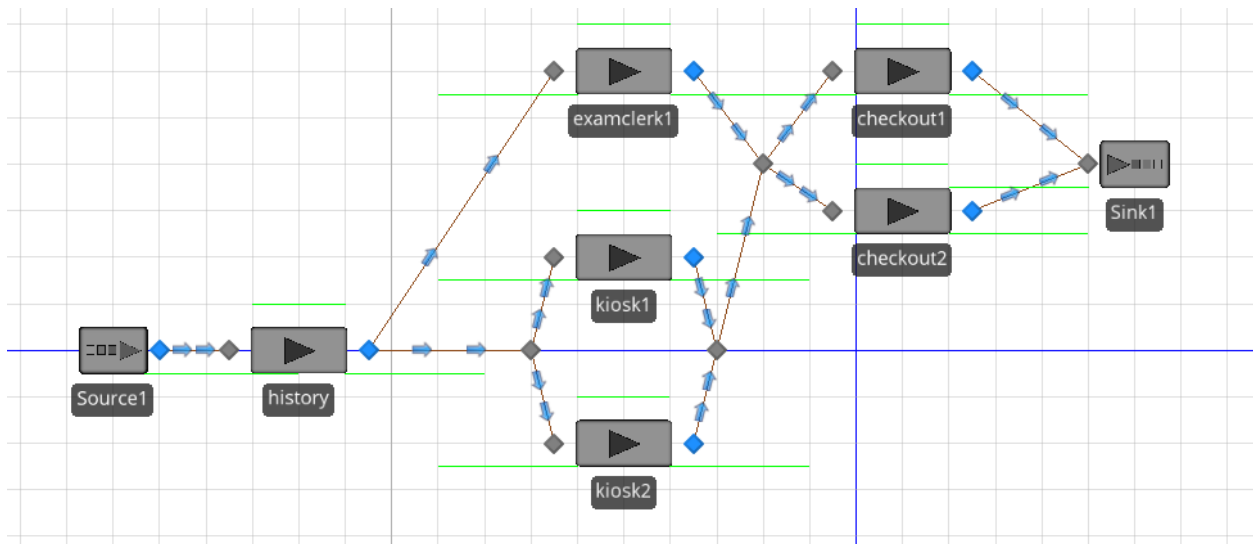
1c.

Some average run times from the system.

Object Type ▲ ▼	Object Name ▲ ▼	Data Source ▲ ▼	Category ▲ ▼	Data Item ▲ ▼	Statistic ▲ ▼	Average Total
ModelEntity	DefaultEntity	[Population]	FlowTime	TimeInSystem	Maximum (Ho...	0.8620
					Minimum (Ho...	0.3820
Server	checkout1	[Resource]	Capacity	ScheduledUtilization	Percent	61.4443
					Average	1.0000
				UnitsScheduled	Maximum	1.0000
					Average	0.6144
	checkout2	[Resource]	Capacity	ScheduledUtilization	Percent	77.5612
					Average	1.0000
				UnitsScheduled	Maximum	1.0000
					Average	0.7756
	examclerk1	[Resource]	Capacity	ScheduledUtilization	Percent	79.0511
					Average	1.0000
				UnitsScheduled	Maximum	1.0000
					Average	0.7905
	examclerk2	[Resource]	Capacity	ScheduledUtilization	Percent	63.8810
					Average	1.0000
				UnitsScheduled	Maximum	1.0000
					Average	0.6388
	history	[Resource]	Capacity	ScheduledUtilization	Percent	83.4689
					Average	1.0000
				UnitsScheduled	Maximum	1.0000
					Average	0.8347
				UnitsUtilized	Average	0.8347
					Maximum	1.0000

1d.

It gets rid of one of the human clerks, and adds in two new kiosks in parallel. The paths can be set to be equal lengths even though they aren't the same in the picute.



2.

Let's do it in R first.

```
mm1 <- function (){
  dura  <- 1220 # duration of sim
  start <- 0 # start time
  inter <- 10 # interarrival time
  serv  <- 7 # service time
  next.a <- 0 # next arrival
  next.d <- dura # next departure
  temp1 <- start # temps
  temp2 <- 0
  n <- 0
  s <- 0
  b <- 0
  c <- 0

  # run while clock hasn't reached completion
  while (start < dura) {
    if (next.a < next.d) { # arriving
      start <- next.a
      s <- s + n * (start - temp1)
      n <- n + 1
      temp1 <- start
      next.a <- start + rexp(1, 1/inter)
      if(n == 1) {
        temp2 <- start
        next.d <- start + rexp(1, 1/serv)
      }
    } else { # leaving
      start <- next.d
      s <- s + n * (start - temp1)
      n <- n - 1
    }
  }
}
```

```

temp1 <- start
c <- c + 1
if (n > 0) {
  next.d <- start + rexp(1, 1/serv)
} else {
  next.d <- dura
  b <- b + start - temp2
}
}
}
return(c(b,c))
}

queue <- replicate(1000, mm1())
paste('Utilization Rate:', mean(queue[1,] / 1220))

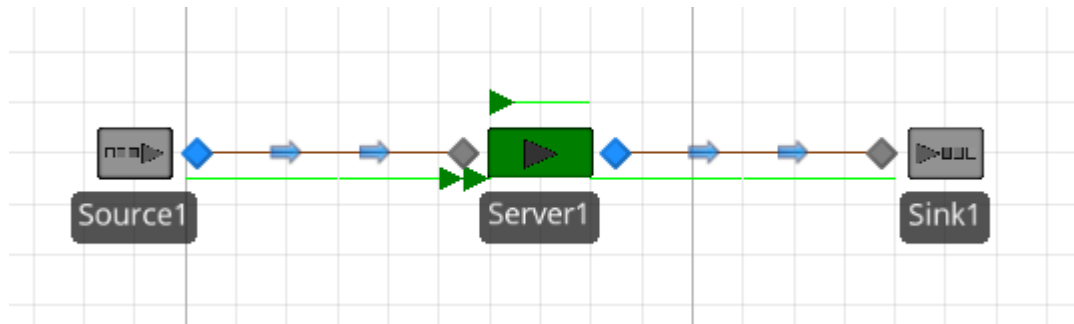
```

```
## [1] "Utilization Rate: 0.652115443252272"
```

```
paste('System Time:' , mean(queue[1,]/ mean(queue[2,] )), 'minutes')
```

```
## [1] "System Time: 6.54788267491705 minutes"
```

And the Simio version:



Drop Filter Fields Here						
Average		Drop Column Fields Here				
Object Type ▲ ▼	Object Name ▲	Data Source ▲	Category ▲ ▼	Data Item ▲	Statistic ▲ ▼	Average Total
ModelEntity	DefaultEntity	[Population]	Content	NumberInSystem	Average	1.0454
			FlowTime	TimeInSystem	Average (Hours)	0.2005
Server	Server1	[Resource]	Capacity	ScheduledUtilization	Percent	53.5540
				UnitsAllocated	Total	58.0000
				UnitsScheduled	Average	1.0000
				UnitsUtilized	Average	0.5355
			ResourceState	TimeProcessing	Average (Hours)	0.2295
					Occurrences	28.0000
					Percent	53.5540
					Total (Hours)	6.4265
			TimeStarved	Average (Hours)	Average (Hours)	0.1991
					Occurrences	28.0000
					Percent	46.4460
					Total (Hours)	5.5735
		InputBuffer	Content	NumberInStation	Average	0.5027
			HoldingTime	TimeInStation	Average (Hours)	0.0939
		Processing	Content	NumberInStation	Average	0.5355
			HoldingTime	TimeInStation	Average (Hours)	0.1117
Sink	Sink1	[DestroyedObjects]	FlowTime	TimeInSystem	Average (Hours)	0.2005
					Observations	57.0000

6.1

arrival times = every 4 minutes

service time = every 3 minutes

atten pay = 10/hour

mech pay = 15/hour

With one attendant, lets find number of mechanics and then cost to run it:

$$\frac{1/4}{(1/3) - (1/4)} = 3$$

$$15(3) + 10 = 55 \text{ dollars}$$

With two attendants, we can use a MMc queue. I'll leave out some of the equations but it comes out to a cost of roughly 33 dollars instead of the above 55. This is due to there being less mechanics in use at any given time.

6.2

We can use a MM1 queue in this case and come out with the equation:

$$3 \leq \frac{\lambda}{(2/3)(2/3 - \lambda)}$$

Solving this gives $\lambda = 4/9$ and $\lambda = 2/3$. From that, we can assume that rates lower than $\lambda = 2/3$ will satisfy the problem.