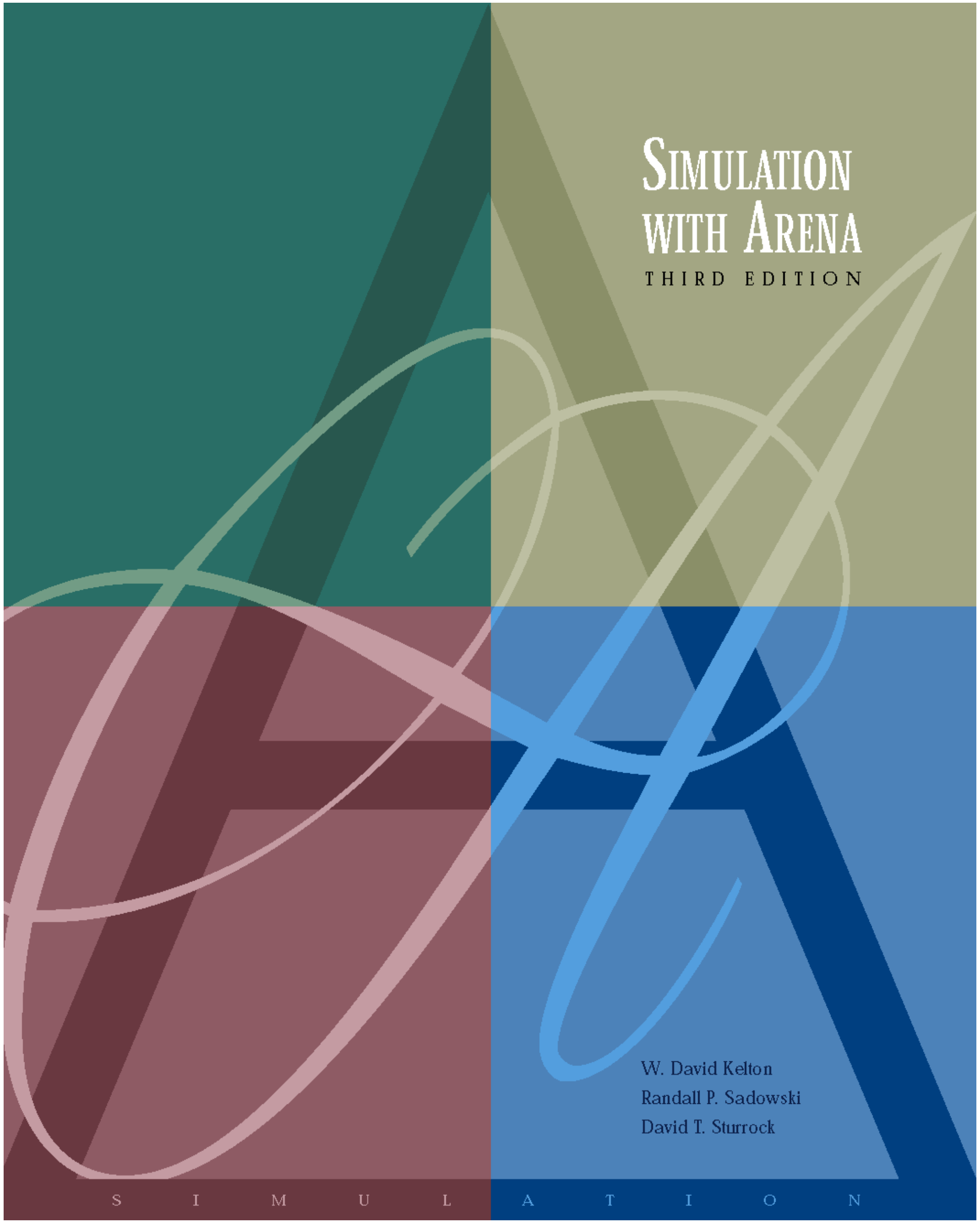
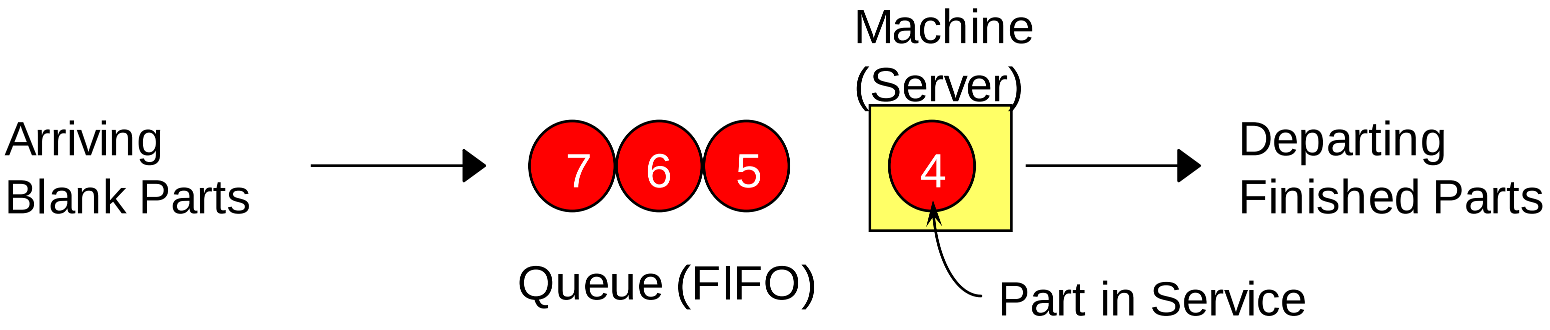


Fundamental Simulation Concepts

Chapter 2

Last revision June 7, 2003

The System: A Simple Processing System



- **General intent:**
  - Estimate expected production
  - Waiting time in queue, queue length, proportion of time machine is busy
- **Time units**
  - Can use different units in different places … must declare
  - Be careful to check the units when specifying inputs
  - Declare *base time units* for internal calculations, outputs
  - Be reasonable (interpretation, roundoff error)

Model Specifics

- **Initially (time 0) empty and idle**
- **Base time units: minutes**
- **Input data (assume given for now …), in minutes:**

Part Number	Arrival Time	Interarrival Time	Service Time
1	0.00	1.73	2.90
2	1.73	1.35	1.76
3	3.08	0.71	3.39
4	3.79	0.62	4.52
5	4.41	14.28	4.46
6	18.69	0.70	4.36
7	19.39	15.52	2.07
8	34.91	3.15	3.36
9	38.06	1.76	2.37
10	39.82	1.00	5.38
11	40.82	.	.
.	.	.	.
.	.	.	.

- **Stop when 20 minutes of (simulated) time have passed**

Goals of the Study: Output Performance Measures

- ***Total production*** **of parts over the run (*P*)**
- ***Average waiting time*** **of parts in queue:**

$$\frac{\sum_{i=1}^{N} WQ_i}{N}$$

N **= no. of parts completing queue wait**
WQ_i **= waiting time in queue of *i*th part**
Know: *WQ_1* = 0 (why?)
$N \geq 1$ **(why?)**

- ***Maximum waiting time*** **of parts in queue:**

$$\max_{i=1,\ldots,N} WQ_i$$

Goals of the Study: Output Performance Measures (cont'd.)

- *Time-average number of part in queue*:

$$\frac{\int_0^{20} Q(t)\,dt}{20}$$

$Q(t)$ = number of parts in queue at time t

- *Maximum number of parts in queue*: $\max_{0 \le t \le 20} Q(t)$
- *Average* and *maximum total time in system* of parts (a.k.a. *cycle time*):

$$\frac{\sum_{i=1}^{P} TS_i}{P}, \quad \max_{i=1,\ldots,P} TS_i$$

TS_i = time in system of part i

Goals of the Study: Output Performance Measures (cont'd.)

- *Utilization* of the machine (proportion of time busy)

$$\frac{\int_0^{20} B(t)\,dt}{20}, \quad B(t) = \begin{cases} 1 & \text{if the machine is busy at time } t \\ 0 & \text{if the machine is idle at time } t \end{cases}$$

- Many others possible (information overload?)

Pieces of a Simulation Model

- ***You need to know the basic concepts of simulation such as entities, resources, variables***

Arrival , departure Etc   These are found in data definition, descrete, input output etc in the software

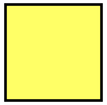
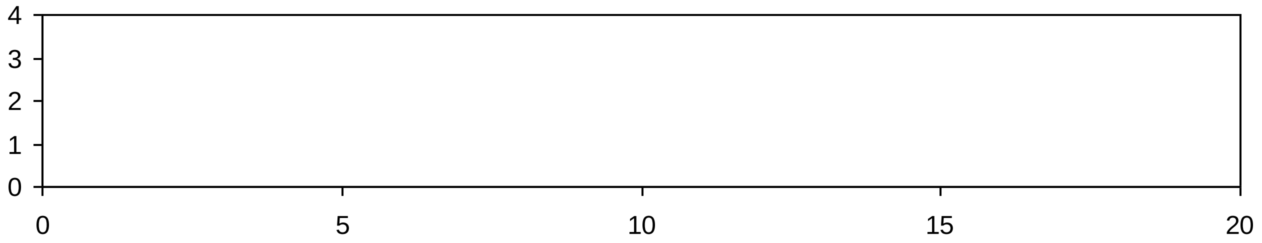
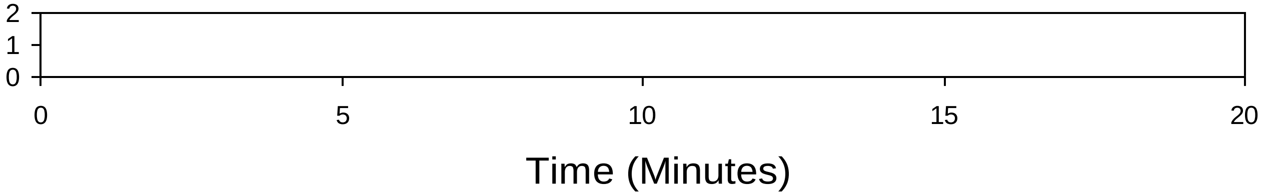
When yo use a variable and when to use attributes

Why variables are called global

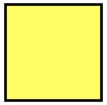
Simulation by Hand

- Manually track state variables, statistical accumulators
- Use “given” interarrival, service times
- Keep track of event calendar
- “Lurch” clock from one event to the next
- Will omit times in system, “max” computations here (see text for complete details)

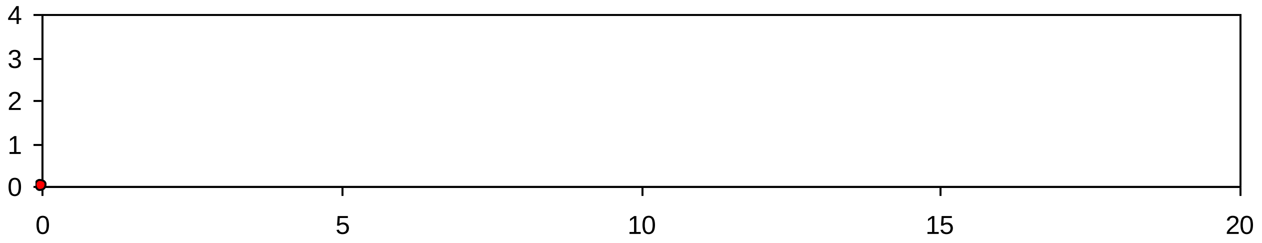
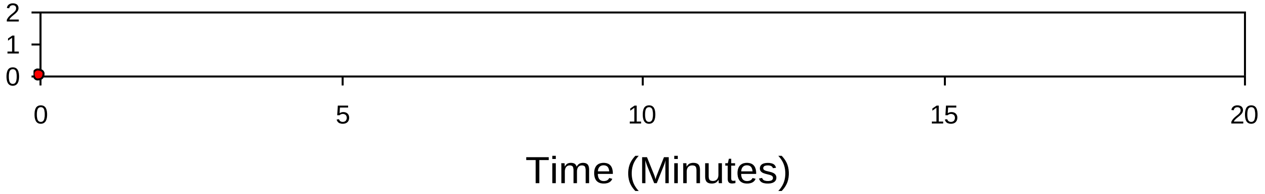
Simulation by Hand: Setup

<table>
<tr><td>System
</td><td>Clock</td><td><i>B</i>(<i>t</i>)</td><td><i>Q</i>(<i>t</i>)</td><td>Arrival times of custs. in queue</td><td>Event calendar</td></tr>
<tr><td>Number of completed waiting times in queue</td><td colspan="3">Total of waiting times in queue</td><td>Area under <i>Q</i>(<i>t</i>)</td><td>Area under <i>B</i>(<i>t</i>)</td></tr>
<tr><td><i>Q</i>(<i>t</i>) graph</td><td colspan="5"> </td></tr>
<tr><td><i>B</i>(<i>t</i>) graph</td><td colspan="5"> </td></tr>
<tr><td>Interarrival times</td><td colspan="5">1.73, 1.35, 0.71, 0.62, 14.28, 0.70, 15.52, 3.15, 1.76, 1.00, ...</td></tr>
<tr><td>Service times</td><td colspan="5">2.90, 1.76, 3.39, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...</td></tr>
</table>

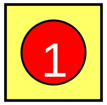
Simulation by Hand:
t = 0.00, Initialize

System	Clock	$B(t)$	$Q(t)$	Arrival times of custs. in queue	Event calendar
	0.00	0	0	<empty>	[1, 0.00, Arr] [–, 20.00, End]

Number of completed waiting times in queue	Total of waiting times in queue	Area under $Q(t)$	Area under $B(t)$
0	0.00	0.00	0.00

$Q(t)$ graph	
$B(t)$ graph	
Interarrival times	1.73, 1.35, 0.71, 0.62, 14.28, 0.70, 15.52, 3.15, 1.76, 1.00, ...
Service times	2.90, 1.76, 3.39, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...

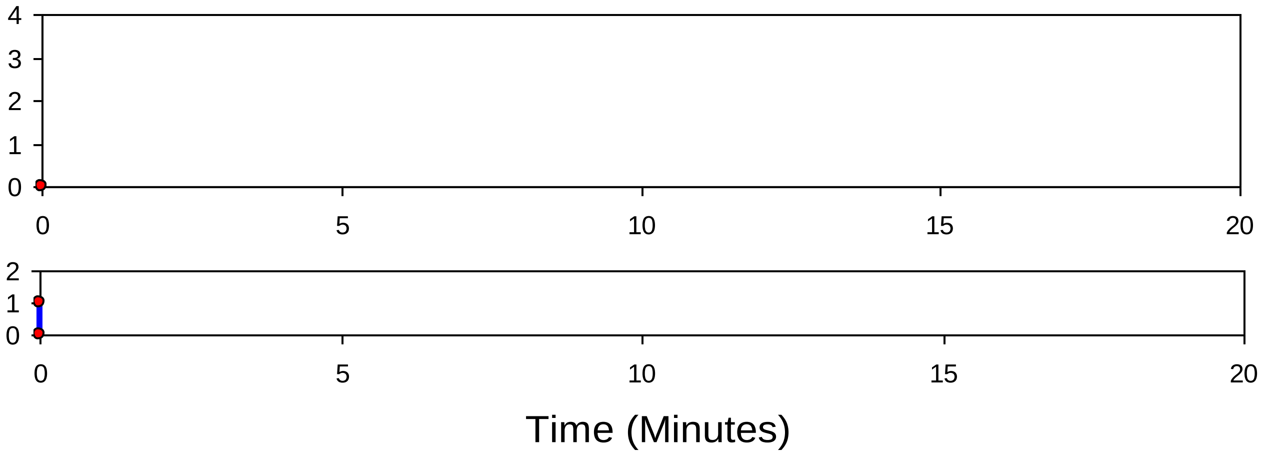
Simulation by Hand: *t* = 0.00, Arrival of Part 1

System	Clock	*B*(*t*)	*Q*(*t*)	Arrival times of custs. in queue	Event calendar
	0.00	1	0	<empty>	[2, 1.73, Arr] [1, 2.90, Dep] [–, 20.00, End]

Number of completed waiting times in queue	Total of waiting times in queue	Area under *Q*(*t*)	Area under *B*(*t*)
1	0.00	0.00	0.00

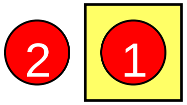
Q(*t*) graph

B(*t*) graph



Interarrival times	~~1.73~~, 1.35, 0.71, 0.62, 14.28, 0.70, 15.52, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, 1.76, 3.39, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...

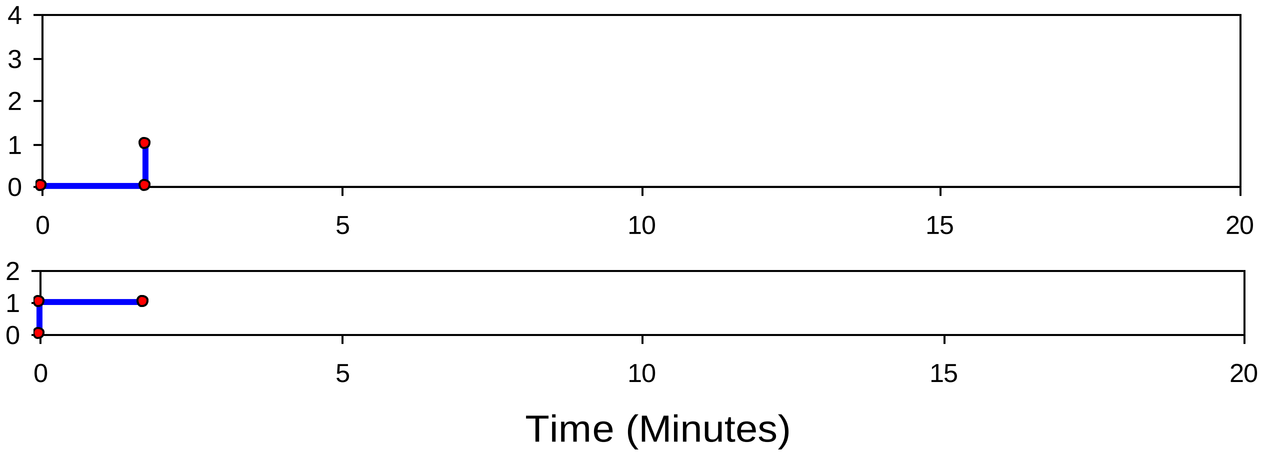
Simulation by Hand:
t = 1.73, Arrival of Part 2

System	Clock	*B*(*t*)	*Q*(*t*)	Arrival times of custs. in queue	Event calendar
	1.73	1	1	(1.73)	[1, 2.90, Dep] [3, 3.08, Arr] [–, 20.00, End]

Number of completed waiting times in queue	Total of waiting times in queue	Area under *Q*(*t*)	Area under *B*(*t*)
1	0.00	0.00	1.73

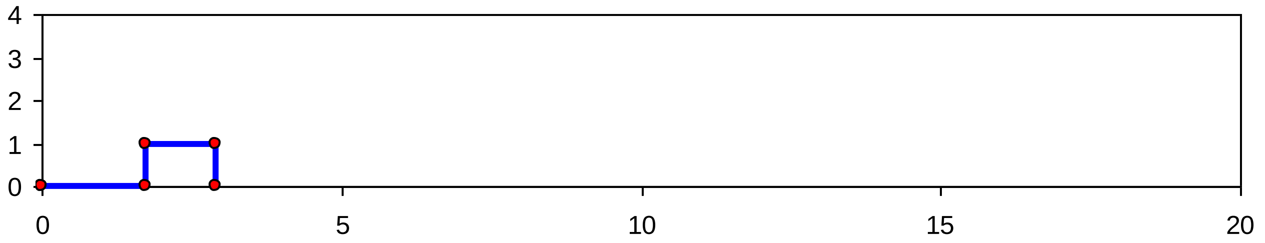
Q(*t*) graph

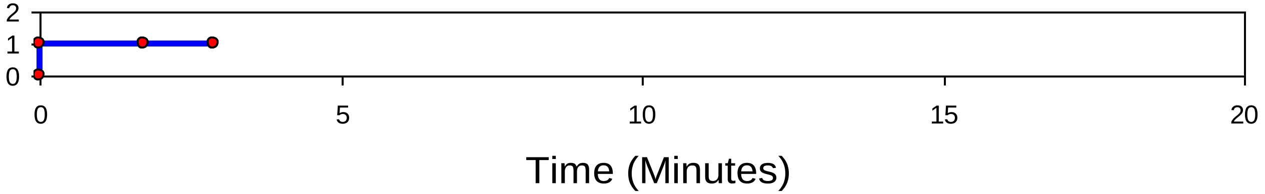
B(*t*) graph



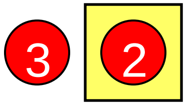
Interarrival times	~~1.73~~, ~~1.35~~, 0.71, 0.62, 14.28, 0.70, 15.52, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, 1.76, 3.39, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...

Simulation by Hand:
t = 2.90, Departure of Part 1

<table>
<tr><td>System
2</td><td>Clock
2.90</td><td><i>B</i>(<i>t</i>)
1</td><td colspan="2"><i>Q</i>(<i>t</i>)
0</td><td>Arrival times of custs. in queue
<empty&gt;</td><td>Event calendar
[3, 3.08, Arr]
[2, 4.66, Dep]
[–, 20.00, End]</td></tr>
<tr><td>Number of completed waiting times in queue
2</td><td colspan="3">Total of waiting times in queue
1.17</td><td colspan="2">Area under <i>Q</i>(<i>t</i>)
1.17</td><td>Area under <i>B</i>(<i>t</i>)
2.90</td></tr>
<tr><td><i>Q</i>(<i>t</i>) graph
<i>B</i>(<i>t</i>) graph</td><td colspan="6">

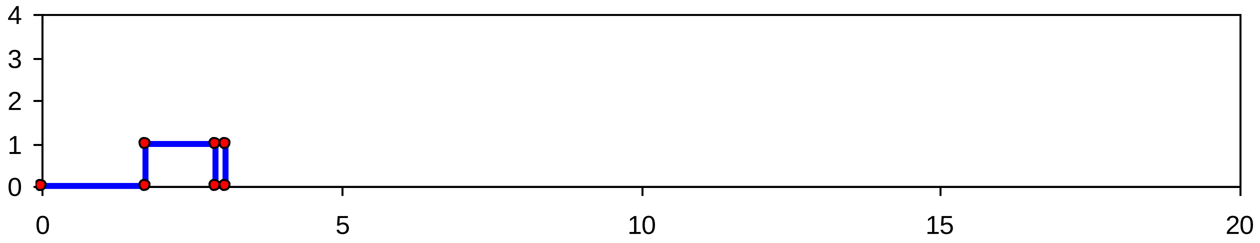

</td></tr>
<tr><td>Interarrival times</td><td colspan="6">~~1.73~~, ~~1.35~~, 0.71, 0.62, 14.28, 0.70, 15.52, 3.15, 1.76, 1.00, ...</td></tr>
<tr><td>Service times</td><td colspan="6">~~2.90~~, ~~1.76~~, 3.39, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...</td></tr>
</table>

Simulation by Hand:
t = 3.08, Arrival of Part 3

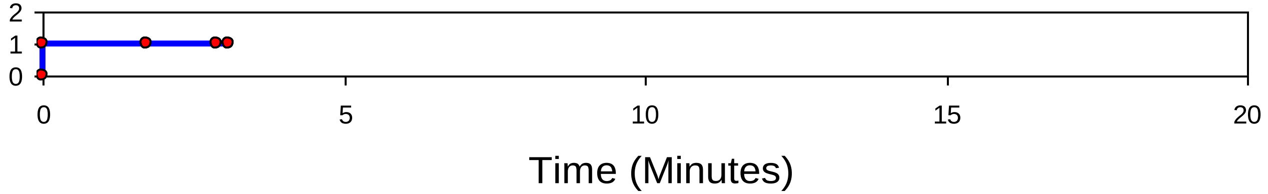
System	Clock	*B*(*t*)	*Q*(*t*)	Arrival times of custs. in queue	Event calendar
	3.08	1	1	(3.08)	[4, 3.79, Arr] [2, 4.66, Dep] [–, 20.00, End]

Number of completed waiting times in queue	Total of waiting times in queue	Area under *Q*(*t*)	Area under *B*(*t*)
2	1.17	1.17	3.08

Q(*t*) graph

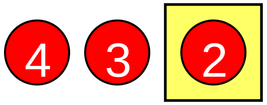


B(*t*) graph

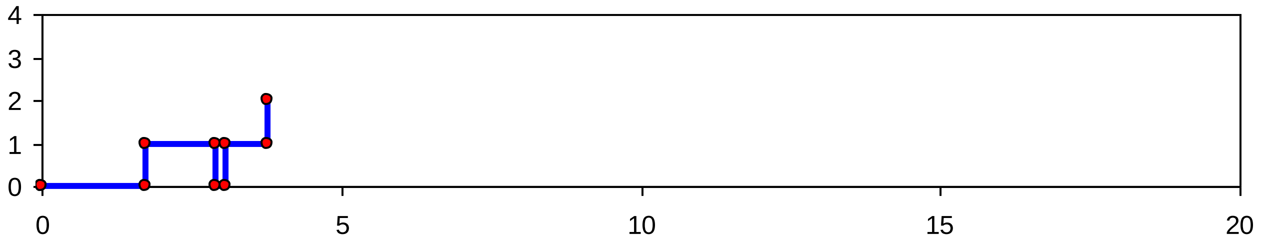
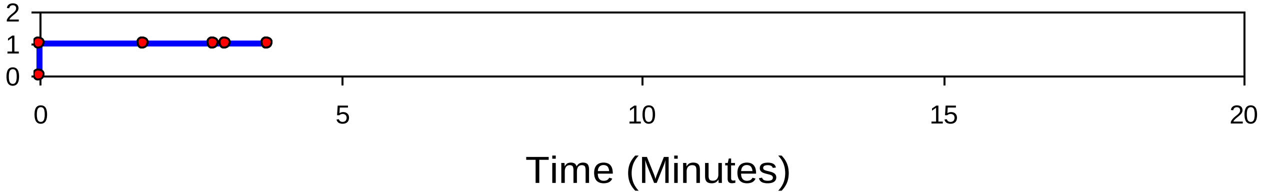


Interarrival times	~~1.73~~, ~~1.35~~, ~~0.71~~, 0.62, 14.28, 0.70, 15.52, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, ~~1.76~~, 3.39, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...

Simulation by Hand: *t* = 3.79, Arrival of Part 4

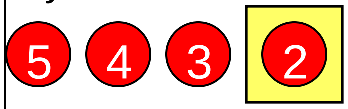
System	Clock	*B*(*t*)	*Q*(*t*)	Arrival times of custs. in queue	Event calendar
	3.79	1	2	(3.79, 3.08)	[5, 4.41, Arr] [2, 4.66, Dep] [–, 20.00, End]

Number of completed waiting times in queue	Total of waiting times in queue	Area under *Q*(*t*)	Area under *B*(*t*)
2	1.17	1.88	3.79

Q(*t*) graph	
B(*t*) graph	
Interarrival times	~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, 14.28, 0.70, 15.52, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, ~~1.76~~, 3.39, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...

Simulation by Hand: *t* = 4.41, Arrival of Part 5

System

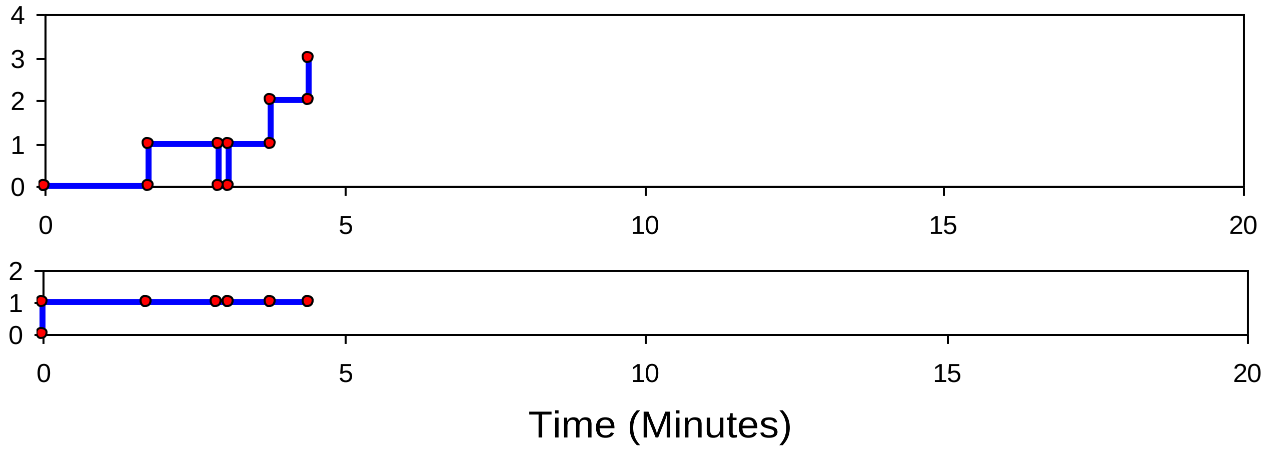


Clock	*B*(*t*)	*Q*(*t*)	Arrival times of custs. in queue	Event calendar
4.41	1	3	(4.41, 3.79, 3.08)	[2, 4.66, Dep] [6, 18.69, Arr] [–, 20.00, End]

Number of completed waiting times in queue	Total of waiting times in queue	Area under *Q*(*t*)	Area under *B*(*t*)
2	1.17	3.12	4.41

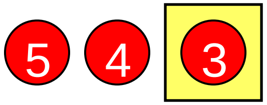
Q(*t*) graph

B(*t*) graph



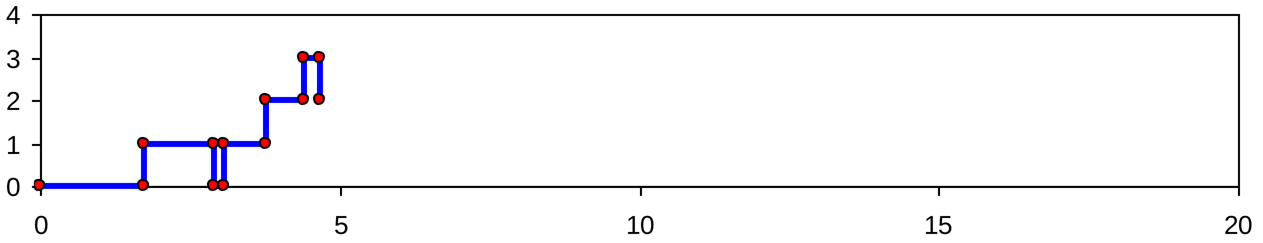
Interarrival times	~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, ~~14.28~~, 0.70, 15.52, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, ~~1.76~~, 3.39, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...

Simulation by Hand:
t = 4.66, Departure of Part 2

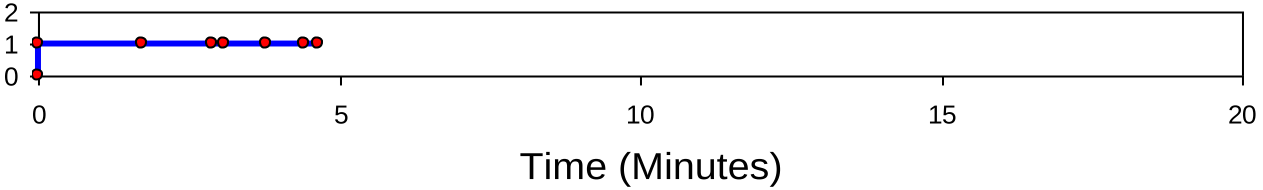
System	Clock	$B(t)$	$Q(t)$	Arrival times of custs. in queue	Event calendar
	4.66	1	2	(4.41, 3.79)	[3, 8.05, Dep] [6, 18.69, Arr] [–, 20.00, End]

Number of completed waiting times in queue	Total of waiting times in queue	Area under $Q(t)$	Area under $B(t)$
3	2.75	3.87	4.66

$Q(t)$ graph

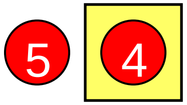


$B(t)$ graph



Interarrival times	~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, ~~14.28~~, 0.70, 15.52, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, ~~1.76~~, ~~3.39~~, 4.52, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...

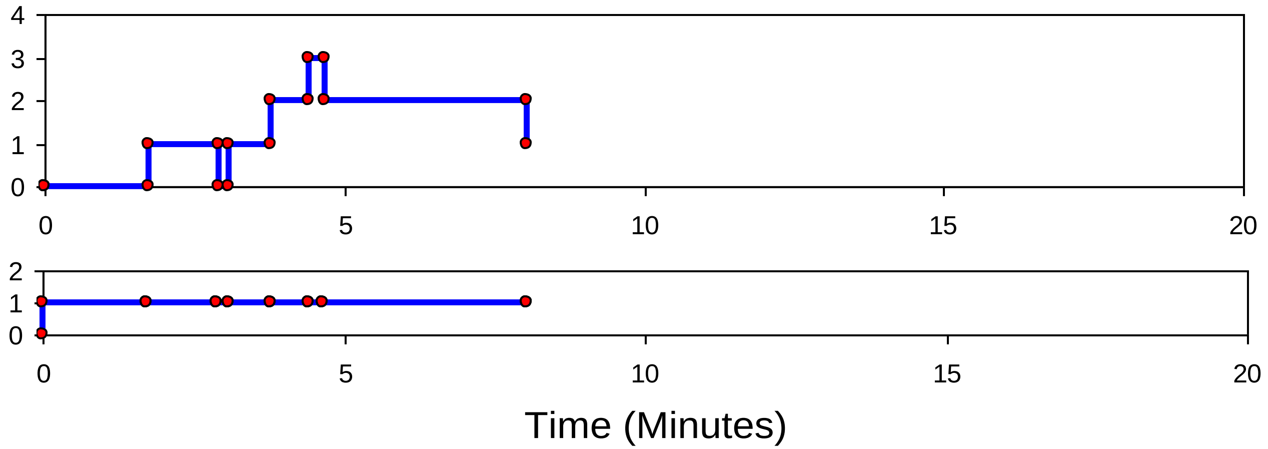
Simulation by Hand:
t = 8.05, Departure of Part 3

System	Clock	$B(t)$	$Q(t)$	Arrival times of custs. in queue	Event calendar
	8.05	1	1	(4.41)	[4, 12.57, Dep] [6, 18.69, Arr] [–, 20.00, End]

Number of completed waiting times in queue	Total of waiting times in queue	Area under $Q(t)$	Area under $B(t)$
4	7.01	10.65	8.05

$Q(t)$ graph

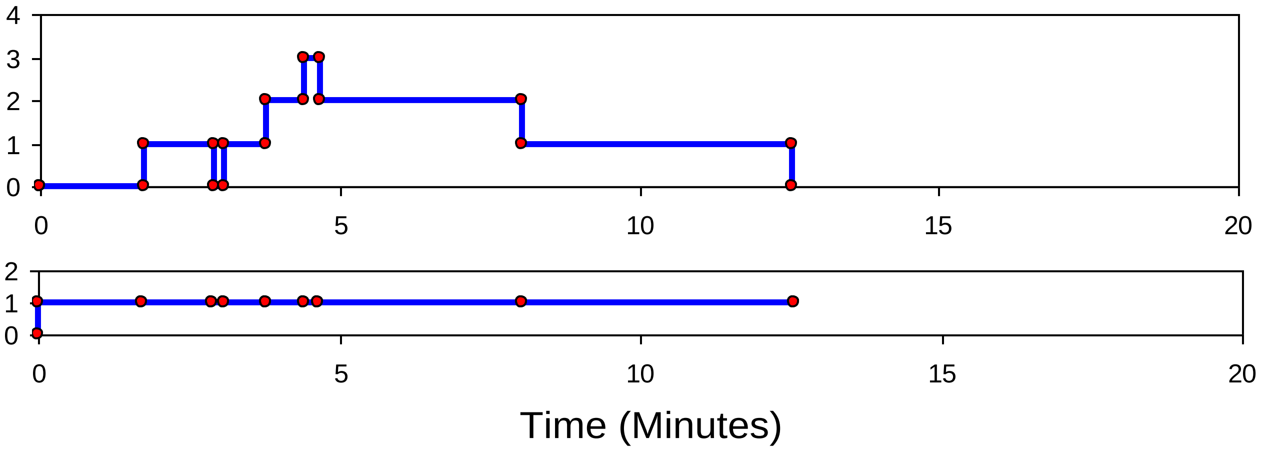
$B(t)$ graph



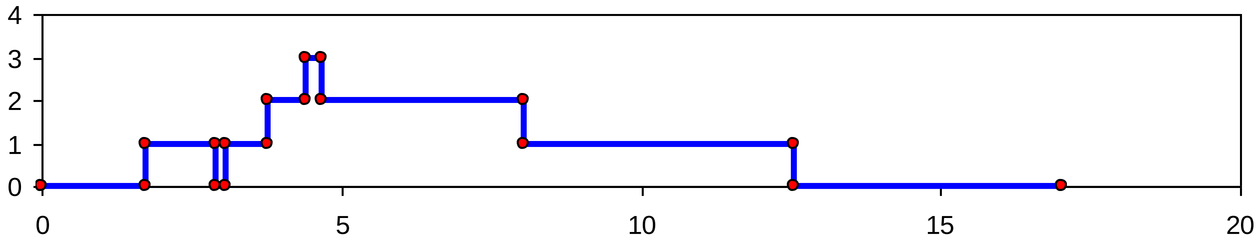
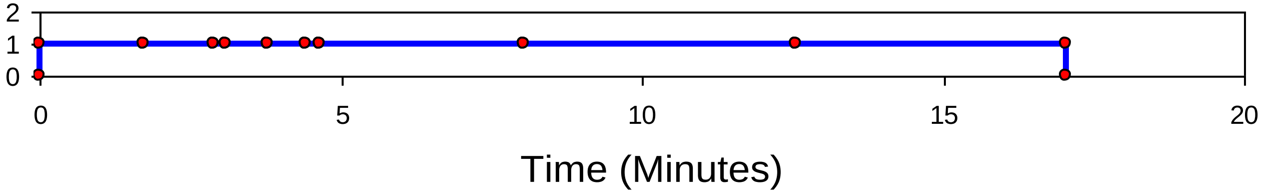
Interarrival times	~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, ~~14.28~~, 0.70, 15.52, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, ~~1.76~~, ~~3.39~~, ~~4.52~~, 4.46, 4.36, 2.07, 3.36, 2.37, 5.38, ...

Simulation by Hand:
t = 12.57, Departure of Part 4

<table>
<tr><td>System

 </td><td>Clock
12.57</td><td>B(t)
1</td><td colspan="2">Q(t)
0</td><td>Arrival times of custs. in queue
()</td><td>Event calendar
[5, 17.03, Dep]
[6, 18.69, Arr]
[–, 20.00, End]</td></tr>
<tr><td>Number of completed waiting times in queue
5</td><td colspan="3">Total of waiting times in queue
15.17</td><td colspan="2">Area under Q(t)
15.17</td><td>Area under B(t)
12.57</td></tr>
<tr><td>Q(t) graph
B(t) graph</td><td colspan="6">
 </td></tr>
<tr><td>Interarrival times</td><td colspan="6">~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, ~~14.28~~, 0.70, 15.52, 3.15, 1.76, 1.00, ...</td></tr>
<tr><td>Service times</td><td colspan="6">~~2.90~~, ~~1.76~~, ~~3.39~~, ~~4.52~~, ~~4.46~~, 4.36, 2.07, 3.36, 2.37, 5.38, ...</td></tr>
</table>

Simulation by Hand:
t = 17.03, Departure of Part 5

<table>
<tr><td>System
🟨</td><td>Clock
17.03</td><td colspan="2"><i>B</i>(<i>t</i>)
0</td><td><i>Q</i>(<i>t</i>)
0</td><td colspan="2">Arrival times of custs. in queue
()</td><td>Event calendar
[6, 18.69, Arr]
[–, 20.00, End]</td></tr>
<tr><td>Number of completed waiting times in queue
5</td><td colspan="3">Total of waiting times in queue
15.17</td><td colspan="3">Area under <i>Q</i>(<i>t</i>)
15.17</td><td>Area under <i>B</i>(<i>t</i>)
17.03</td></tr>
<tr><td><i>Q</i>(<i>t</i>) graph</td><td colspan="7"> </td></tr>
<tr><td><i>B</i>(<i>t</i>) graph</td><td colspan="7"> </td></tr>
<tr><td>Interarrival times</td><td colspan="7">~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, ~~14.28~~, 0.70, 15.52, 3.15, 1.76, 1.00, ...</td></tr>
<tr><td>Service times</td><td colspan="7">~~2.90~~, ~~1.76~~, ~~3.39~~, ~~4.52~~, ~~4.46~~, 4.36, 2.07, 3.36, 2.37, 5.38, ...</td></tr>
</table>

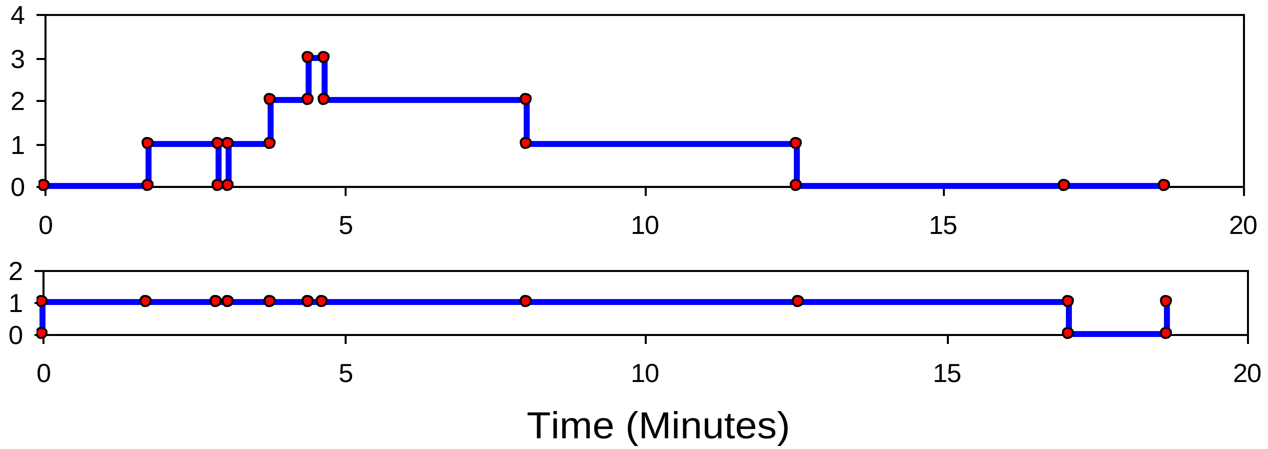
Simulation by Hand: *t* = 18.69, Arrival of Part 6

System	Clock	$B(t)$	$Q(t)$	Arrival times of custs. in queue	Event calendar
6	18.69	1	0	()	[7, 19.39, Arr] [–, 20.00, End] [6, 23.05, Dep]

Number of completed waiting times in queue	Total of waiting times in queue	Area under $Q(t)$	Area under $B(t)$
6	15.17	15.17	17.03

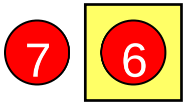
$Q(t)$ graph

$B(t)$ graph



Interarrival times	~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, ~~14.28~~, ~~0.70~~, 15.52, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, ~~1.76~~, ~~3.39~~, ~~4.52~~, ~~4.46~~, ~~4.36~~, 2.07, 3.36, 2.37, 5.38, ...

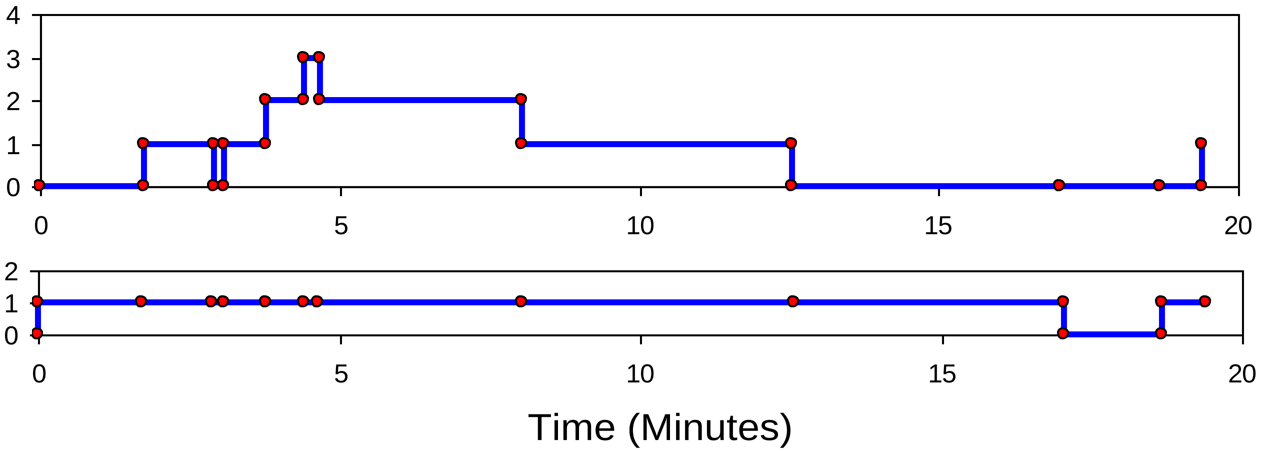
Simulation by Hand:
t = 19.39, Arrival of Part 7

System	Clock	*B*(*t*)	*Q*(*t*)	Arrival times of custs. in queue	Event calendar
	19.39	1	1	(19.39)	[–, 20.00, End] [6, 23.05, Dep] [8, 34.91, Arr]

Number of completed waiting times in queue	Total of waiting times in queue	Area under *Q*(*t*)	Area under *B*(*t*)
6	15.17	15.17	17.73

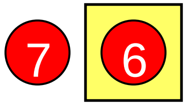
Q(*t*) graph

B(*t*) graph

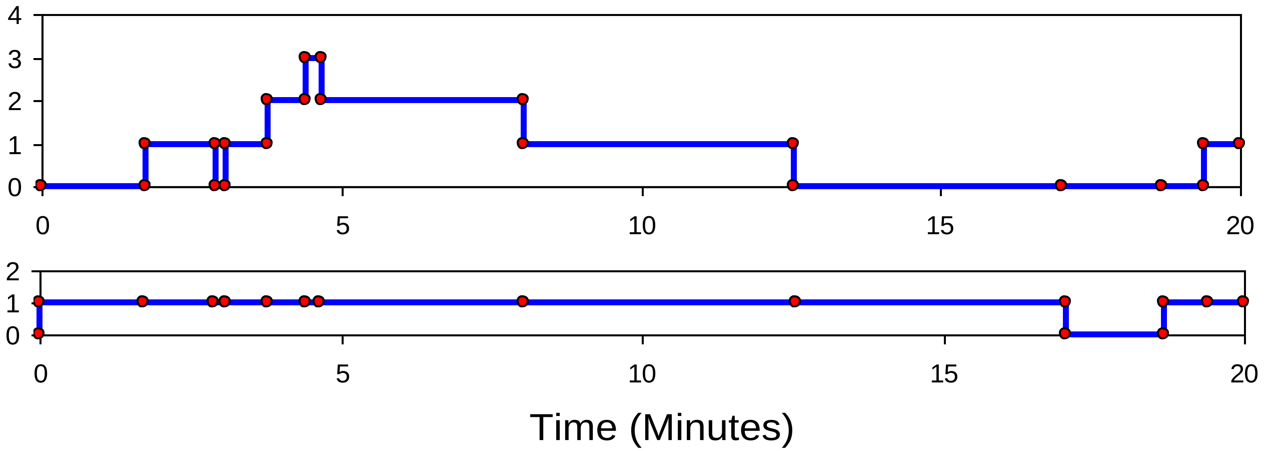


Interarrival times	~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, ~~14.28~~, ~~0.70~~, ~~15.52~~, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, ~~1.76~~, ~~3.39~~, ~~4.52~~, ~~4.46~~, ~~4.36~~, 2.07, 3.36, 2.37, 5.38, ...

Simulation by Hand: *t* = 20.00, The End

System 	Clock 20.00	*B*(*t*) 1	*Q*(*t*) 1	Arrival times of custs. in queue (19.39)	Event calendar [6, 23.05, Dep] [8, 34.91, Arr]

Number of completed waiting times in queue 6	Total of waiting times in queue 15.17	Area under *Q*(*t*) 15.78	Area under *B*(*t*) 18.34

Q(*t*) graph *B*(*t*) graph	
Interarrival times	~~1.73~~, ~~1.35~~, ~~0.71~~, ~~0.62~~, ~~14.28~~, ~~0.70~~, ~~15.52~~, 3.15, 1.76, 1.00, ...
Service times	~~2.90~~, ~~1.76~~, ~~3.39~~, ~~4.52~~, ~~4.46~~, ~~4.36~~, 2.07, 3.36, 2.37, 5.38, ...

Simulation by Hand: Finishing Up

- **Average waiting time in queue:**

$$\frac{\text{Total of times in queue}}{\text{No. of times in queue}} = \frac{15.17}{6} = 2.53 \text{ minutes per part}$$

- **Time-average number in queue:**

$$\frac{\text{Area under } Q(t) \text{ curve}}{\text{Final clock value}} = \frac{15.78}{20} = 0.79 \text{ part}$$

- **Utilization of drill press:**

$$\frac{\text{Area under } B(t) \text{ curve}}{\text{Final clock value}} = \frac{18.34}{20} = 0.92 \text{ (dimensionless)}$$

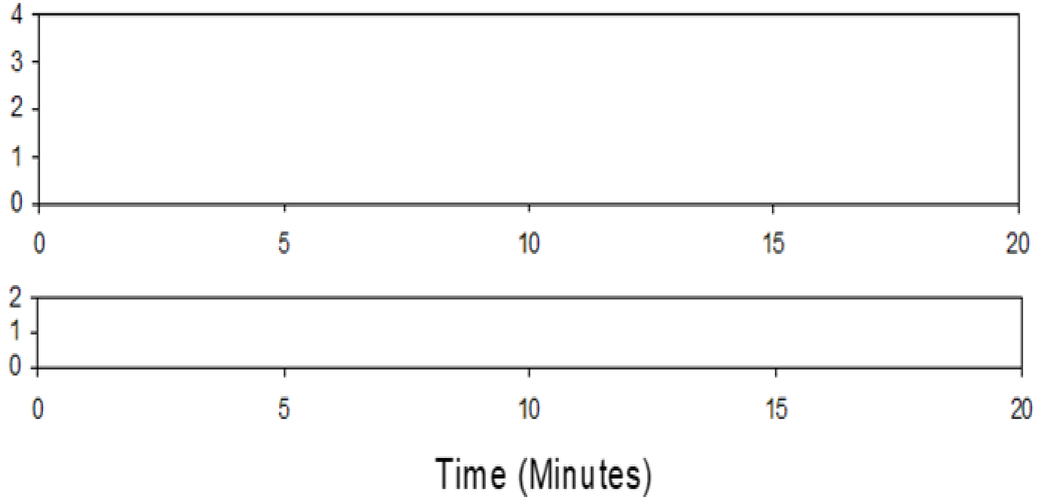
Practice this quiz

System 🟨	Clock	$B(t)$	$Q(t)$	Arrival times of custs. in queue	Event calendar

Number of completed waiting times in queue	Total of waiting times in queue	Area under $Q(t)$	Area under $B(t)$

$Q(t)$ graph

$B(t)$ graph



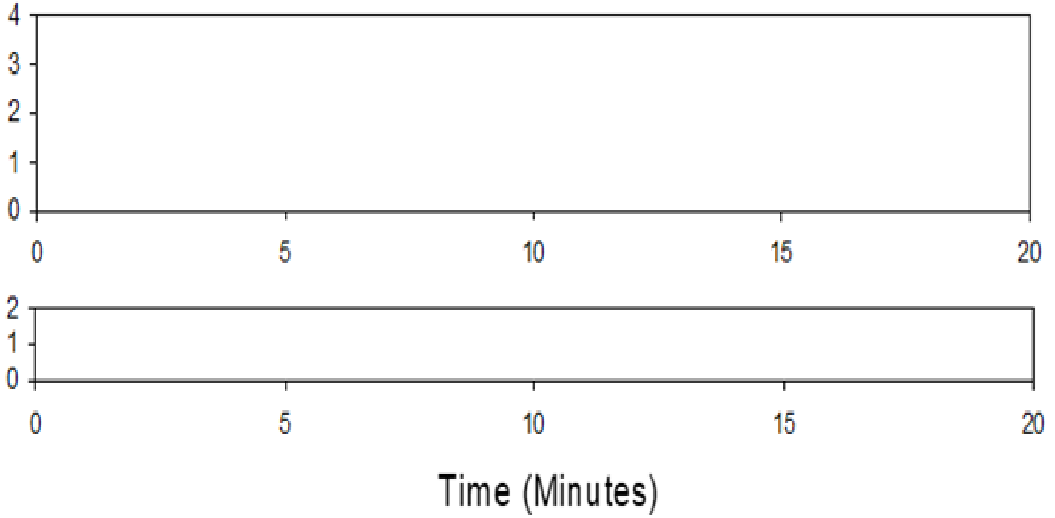
Interarival	1.6	4.8	5.9	4.7	1.3	1.0	0.7	11.3	8.3	1.2	1.9	7.2	3.9	4.8
Service	3.2	1.3	3.0	6.6	5.2	8.4	14.6	7.8	7.8	9.4	6.8	10.5	0.0	0.0

Practice this quiz

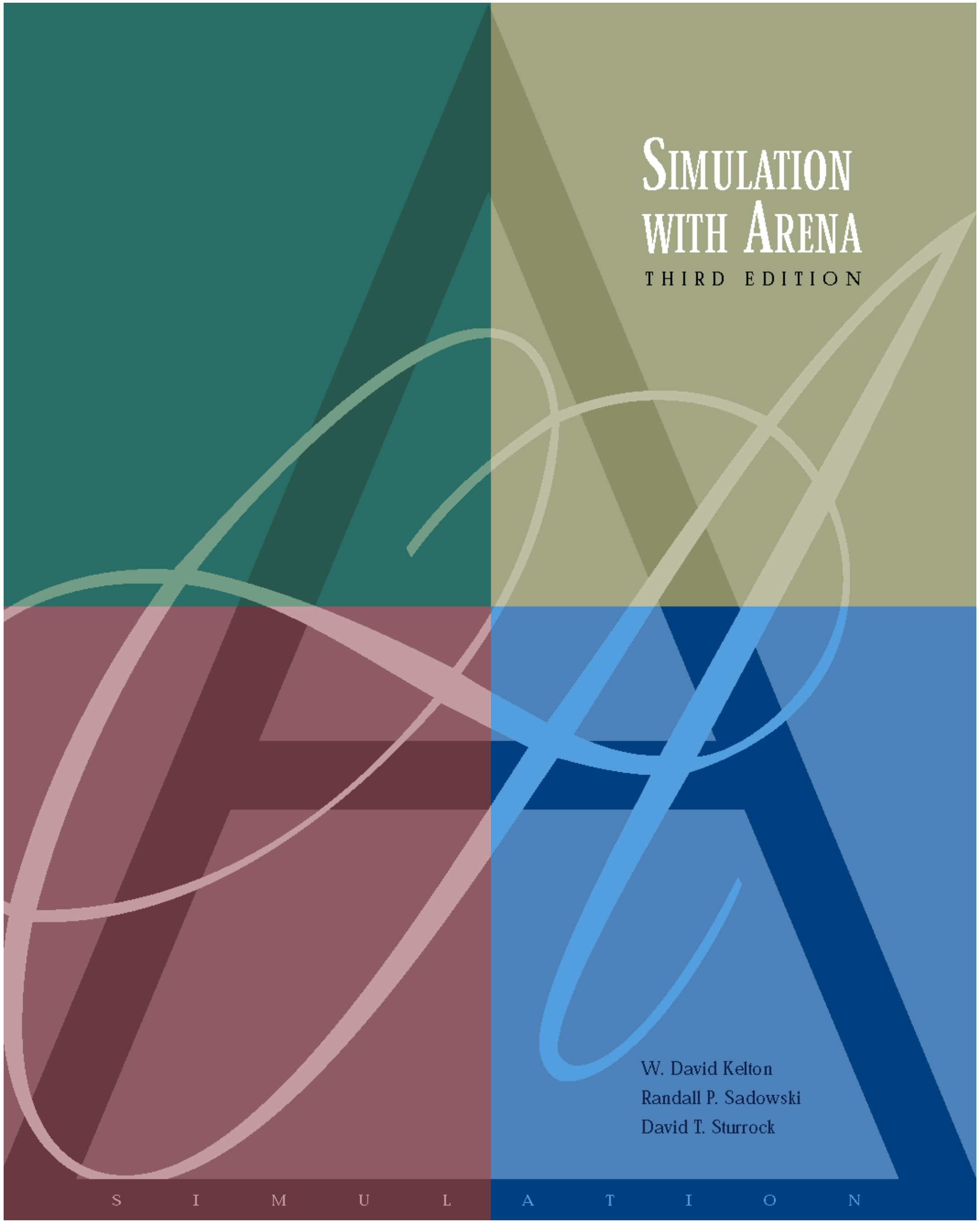
System 🟨	Clock	*B*(*t*)	*Q*(*t*)	Arrival times of custs. in queue	Event calendar
Number of completed waiting times in queue	Total of waiting times in queue	Area under *Q*(*t*)	Area under *B*(*t*)		

Q(*t*) graph

B(*t*) graph



Interarival	0.8	2.4	3.0	2.3	1.0	3.4	3.8	1.5	0.9	0.4	2.3	7.5	16.4	3.6
Service	4.4	7.1	3.4	0.1	1.0	5.6	2.2	7.4	1.5	7.1	11.8	4.3	1.9	1.2

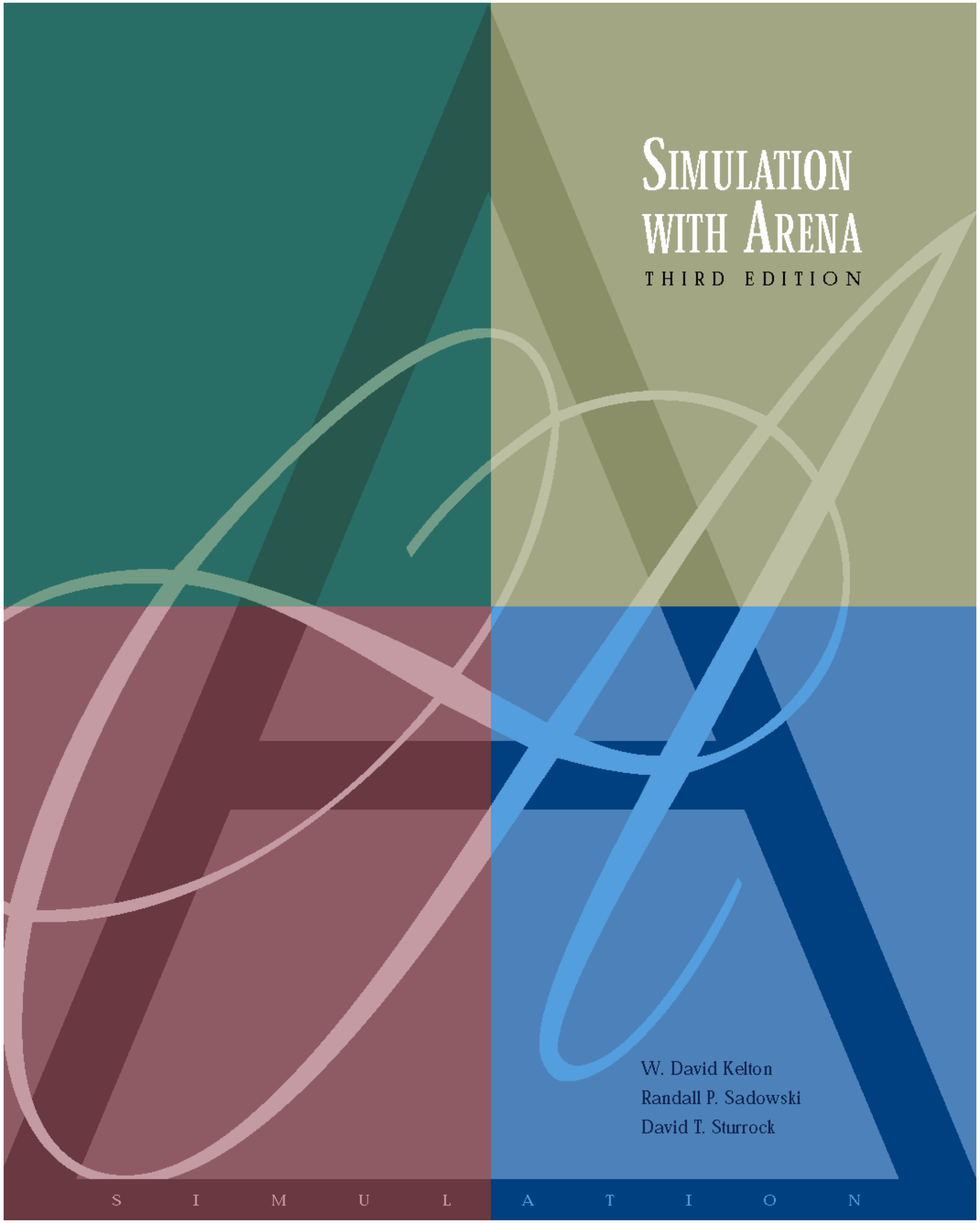


A Guided Tour Through Arena

Chapter 3 from the software what is the function of every menu item, button ribbon item, the chapter is just a reference

Read it but it is better to practice every aspect in the software.

Last revision June 7, 2003



Modeling Basic Operations and Inputs

Chapter 4

Last revision June 7, 2003

What We'll Do ...

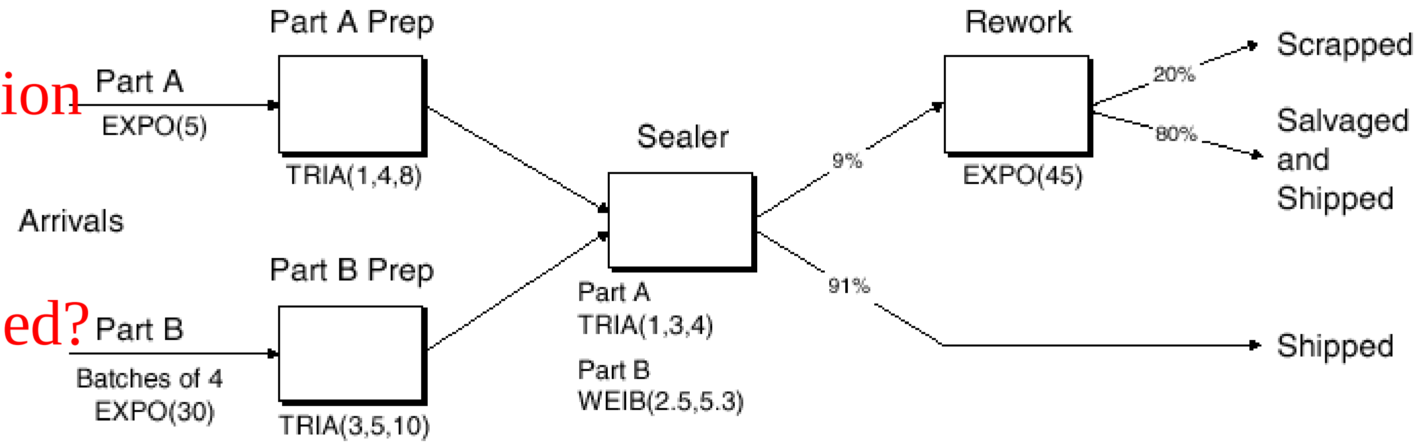
- **Model 4-1: Electronic assembly/test system**
  - Modeling approaches
  - New Arena modules (Decide, Record)
- **Model 4-2: Enhanced electronic assembly/test**
  - Resource Schedules, States, and Failures
  - Frequency outputs
  - More on utilizations
- **Model 4-3: Enhancing the animation**
  - Queues, Entity Pictures, Resource Pictures
  - Adding Plots and Variables

What We'll Do … (cont'd.)

- **Model 4-4: Adding entity travel times**
  - Modify with Stations, Transfers, Routes, animation of entity movement
- **Input analysis**
  - Specifying input distributions, parameters
  - Deterministic vs. random input
  - Collecting and using data
  - Fitting input distributions via the Input Analyzer
  - No data?
  - Nonstationary arrival processes
  - Multivariate and correlated input data

Electronic Assembly/Test System (Model 4-1)

Show the arena simulation for the model described?



- **Produce two different sealed elect. units (A, B)**
- **Arriving parts: cast metal cases machined to accept the electronic parts**
- **Part A, Part B – separate prep areas**
- **Both go to Sealer for assembly, testing – then to Shipping (out) if OK, or else to Rework**
- **Rework – Salvaged (and Shipped), or Scrapped**

Part A

- **Interarrivals: expo (5) minutes**
- **From arrival point, proceed immediately to Part A Prep area**
  - Process = (machine + deburr + clean) ~ tria (1,4,8) minutes
- **Go immediately to Sealer**
  - Process = (assemble + test) ~ tria (1,3,4) min.
  - 91% pass, go to Shipped; Else go to Rework
- **Rework: (re-process + testing) ~ expo (45)**
  - 80% pass, go to Salvaged; Else go to Scrapped

Part B

- **Interarrivals: *batches* of 4, expo (30) min.**
- **Upon arrival, batch separates into 4 individual parts**
- **From arrival point, proceed immediately to Part B Prep area**
  - Process = (machine + deburr +clean) ~ tria (3,5,10)
- **Go to Sealer**
  - Process = (assemble + test) ~ weib (2.5, 5.3) min. , *different* from Part A, though at same station
  - 91% pass, go to Shipped; Else go to Rework
- **Rework: (re-process + test) = expo (45) min.**
  - 80% pass, go to Salvaged; Else go to Scrapped

Run Conditions, Output

- **Start empty & idle, run for four 8-hour shifts (1,920 minutes)**
- **Collect statistics for each work area on**
  - Resource utilization
  - Number in queue
  - Time in queue
- **For each exit point (Shipped, Salvaged, Scrapped), collect total time in system (a.k.a. cycle time)**

Developing a Modeling Approach

- **Define pieces of model, modules, data structures, control logic**
- **Appropriate level of detail – judgment call**
- **Often multiple ways to model, represent logic**
- **This model:**
  - Entities are the individual parts (two types)
  - Separate Create modules for two part types
  - Separate Process modules for each Prep area
  - Process modules for Sealer and Rework, each followed by a Decide module (2-way by Chance)
  - Depart modules for Shipped, Salvaged, Scrapped
  - Attribute **`Sealer Time`** assigned after Creates in Assign modules (parts have *different* times at *the* Sealer)
  - Record modules just before Departs for time in system

Building the Model

- **New model window**
- **Attach Basic Process panel (if needed)**
- **Place modules**
  - Create (x 2)
  - Assign (x 2)
  - Process (x 4)
  - Decide (x 2)
  - Record (x 3)
  - Dispose (x 3)
- **Right click — repeat last action (place module)**
- **Auto-Connect, or manually connect via**

Alternate strategy – place one module at a time, fill it out completely

Part A Create Module

- **Name: `Part A Arrive`**
- **Entity Type: `Part A`**
- **Time Between Arrivals**
  - Type: **`Random (Expo)`**
    - Pull-down list with options
  - Value: **`5`**
  - Units: **`Minutes`**
    - Pull-down list with options
- **Default what's not mentioned above**

Once these entries are made, they are placed on the list for names of that type (Module Name, Entity Type, etc.) and will appear on future pull-down lists for that type of name.

Part B Create Module

- **Name: `Part B Arrive`**
- **Entity Type: `Part B`**
- **Time Between Arrivals**
  - Type: **`Random (Expo)`**
    - Pull-down list with options
  - Value: **`30`**
  - Units: **`Minutes`**
    - Pull-down list with options
- **Entities per Arrival: `4`**

How is the batch size modelled?

Part A Attributes Assign Module

- **Name:** **`Assign Part A Sealer and Arrive Time`**
- **Add button:**
  - Type: **`Attribute`**
  - Attribute Name: **`Sealer Time`**
  - New Value: **`TRIA(1, 3, 4)`**
- **Add button:**
  - Type: **`Attribute`**
  - Attribute Name: **`Arrive Time`**
  - New Value: **`TNOW`** (to compute time in system on exit)

TNOW is the internal Arena variable name for the simulation clock; see *Help > Arena Help > Contents >*
Variables, Functions, and Distributions > Date and Time Variables

Part B Attributes Assign Module

- **Name: `Assign Part B Sealer and Arrive Time`**
- **Add button:**
  - Type: **`Attribute`**
  - Attribute Name: **`Sealer Time`**
  - New Value: **`WEIB(2.5, 5.3)`**
- **Add button:**
  - Type: **`Attribute`**
  - Attribute Name: **`Arrive Time`**
  - New Value: **`TNOW`**

Names for things in Arena
– Default names usually suggested
– Names placed on appropriate pull-down lists for future reference
– All names in a model must be unique (even across different kinds of objects)

Process Module *Actions*

- ***Delay***

  Entity just sits here for the specified time; no Resource involved, so multiple entities could be undergoing this Delay simultaneously

- ***Seize Delay***

  Entity must first Seize the specified number of units of a Resource (possibility for Queueing if they're not available), then undergoes the Delay … assume that the entity will Release the Resource units at another downstream module

- ***Seize Delay Release***

  Like Seize Delay, but entity releases Resource units after Delay (*what we want in this model*)

- ***Delay Release***

  Assumes entity had already Seized Resource units at another upstream module, now Delays and Releases Resource units

Prep A Process Module

- **Name:** `Prep A Process`
- **Action:** `Seize Delay Release`
- **Resources subdialog (Add button):**
  - Type: `Resource` (a pull-down option)
  - Resource Name: `Prep A`
  - Quantity: `1` (default)
- **Delay Type:** `Triangular`
- **Units:** `Minutes`
- **Minimum:** `1`
- **Value (Most Likely):** `4`
- **Maximum:** `8`

If several Resources were named (Add button), entity would have to Seize them all before the Delay could start.

Prep B Process Module

- **Name:** `Prep B Process`
- **Action:** `Seize Delay Release`
- **Resources subdialog (Add button):**
  - Type: `Resource` (a pull-down option)
  - Resource Name: `Prep B`
  - Quantity: `1` (default)
- **Delay Type:** `Triangular`
- **Units:** `Minutes`
- **Minimum:** 3
- **Value (Most Likely):** 5
- **Maximum:** 10

Sealer Process Module

- **Name:** `Sealer Process`
- **Action:** `Seize Delay Release`
- **Resources subdialog (Add button):**
  - Type: `Resource` (a pull-down option)
  - Resource Name: `Sealer`
  - Quantity: `1` (default)
- **Delay Type:** `Expression`
- **Units:** `Minutes`
- **Expression:** `Sealer Time`

Recall – `Sealer Time` attribute was defined upstream for both Parts A and B … now its value is being used … allows for different distributions for A and B.

Sealer Inspection-Result *Decide* Module

- **Decide module provides branch points**
  - *By Condition* (entity Attributes, global Variables)
  - *By Chance* (multi-sided, possibly-biased hypercoin flip)
- **Name: `Failed Sealer Inspection`**
- **Type: `2-way by Chance` (default)**
- **Percent True: `9`**
- **Different exit points for True, False results – connect appropriately downstream**

> – Note it's *percent* true, not *probability* of true … so "9" means probability of 0.09
> – We arbitrarily decided "true" meant part failed inspection … could have reversed (but would change numerical results ... why? ... does this upset you? ... why?)
> – This is a rich, deep, versatile module … explore its Help button

Rework Process Module

- **Name: `Rework Process`**
- **Action: `Seize Delay Release`**
- **Resources subdialog (Add button):**
  - Type: **`Resource`** (a pull-down option)
  - Resource Name: **`Rework`**
  - Quantity: **`1`** (default)
- **Delay Type: `Expression`**
- **Units: `Minutes`**
- **Expression: `EXPO(45)`**

Had to use the general **`Expression`** choice for Delay Type since what we want (**`EXPO`**) is not directly on the Delay Type pull-down list.

Rework Inspection-Result Decide Module

- **Name:** `Failed Rework Inspection`
- **Type:** `2-way by Chance` **(default)**
- **Percent True:** `20`

We arbitrarily decided “true” meant part failed inspection

Record Modules

- **Arena collects and reports many output statistics by default, but sometimes not all you want**
- **We want time in system (average, max) of parts sorted out by their exit point (Shipped, Salvaged, Scrapped)**
  - It’s this sorting that Arena doesn’t do by default ... it would automatically sort *by Entity Type* if we had Entities checked in Run > Setup > Project Parameters (which we don’t)
- **Record module can be placed in the flowchart to collect and report various kinds of statistics from within the model run as entities pass through it**
- **For Tally-type output performance measures (see Chapter 3)**

Shipped Parts *Record* Module

Describe how to record time in system? And the parts produced?

- **Name:** **`Record Shipped Parts`**
- **Type:** **`Time Interval`**
  - This option records the length of time that elapsed up to now (**`TNOW`**) from when an entity attribute was marked with a time "stamp" upstream ... Attribute Name is below ...
  - There are several other options for Type … explore via Record module's Help button!
- **Attribute Name:** **`Arrive Time`**
  - Was defined upstream as the clock value in the Assign modules instantly after each entity was Created
- **Tally Name:** **`Record Shipped Parts`**
  - Determines the label in the reports

Other two Record modules – just like this except for Name and Tally Name.

Dispose Modules

- **Three separate exit points for three separate part disposition (Shipped, Salvaged, Scrapped)**
- **Could have directed all three exit types to a single Dispose module**
  - But having separate ones produces animation counts of the three dispositions
- **Also, having separate Dispose modules allows for differentially checking the boxes to Record Entity Statistics**
  - Produces flow statistics separated by entity type (*if* Entities Statistics Collection is checked in *Run > Setup > Project Parameters*), *not* by final disposition of part … so we *did* need our Record modules and Arrive Time attribute

Vhat are the
eplications for? How
o add them?

Run > Setup for Run Control

- **Without this, model would run forever – no defaults for termination rule**
  - That’s part of modeling, and generally affects results!
- **Project Parameters tab:**
  - Fill in Project Title, Analyst Name
  - Defaults for Statistics Collection, but we cleared the check box for Entities – not needed for what we want (we installed our own Record modules), and would slow execution
- **Replication Parameters tab:**
  - Replication length: **`32`**, accept **`Hours`** default for Time Units
  - Base Time Units: **`Minutes`** for inputs without Time Units option, internal arithmetic, and units on output reports

Different Part A, B Entity Pictures

- **Entity data module (just single-click on it in Project Bar, edit via spreadsheet only)**
- **Row for each Entity Type (`Part A`, `Part B`)**
- **Pull down Initial Picture pull-down menu, select different pictures for each Entity Type**
  - *Edit > Entity Pictures* to see, change the list of pictures that's presented here … more later

Describe how change the entity picture before run and during the replicate?

Running the Model

Describe the job of the different buttons?

- **Check ✓ (if desired)**
  - Find button to help find errors
- **Go ▶ (will automatically pre-Check if needed)**
  - Some graphics don’t show during run … will return when you End your run … control via *View > Layers*
  - Status Bar shows run progress – replication number, simulation time, simulation status
- **Animation speed – increase (>), decrease (<)**
- **Pause (❚❚) or Esc key; ▶ to resume**
- ***Run > Step* (▶❚) to debug**
- ***Run > Fast-Forward* (▶▶) to turn off animation**
  - *Run > Run Control > Batch Run (No Animation)* is fastest

Viewing the Results

- **Counters during animation for modules**
  - Create, Dispose, Decide – incremented when entity leaves
  - Process – number of entities currently in the module
- **Asked at end if you want to see reports**
  - What you get depends on *Run > Setup > Project Parameters*
    - *Looks like* the Rework area is a bottleneck ... more later
  - Navigate through report with browsing arrows, tree at left
  - Tally, Time-Persistent, and Counter statistics
  - Avg, Min, Max, and 95% Confidence Interval half-widths
    - Confidence intervals are for steady-state expectations … more later
    - May not be produced if run is not long enough for reliable stats
- **Generally difficult/unreliable to draw conclusions from just one run … more later**

Model 4-2: The Enhanced Electronic Assembly and Test System

- **A Story**
  - Original model shown to production manager
  - Pointed out that this is only the first shift of a two-shift day — on second shift there are two operators at Rework (the bottleneck station) … 16-hour days
  - Pointed out that the Sealer fails sometimes
    - Uptimes ~ exponential, mean 2 hours
    - Repair times ~ exponential, mean 4 minutes
  - Wants to buy racks to hold rework queue
    - A rack holds 10 parts
    - How many racks should be bought?
  - Run for 10 days
- **Need:** ***Resource Schedules***, ***Resource States***, ***Resource Failures***

Modify the model to include the following ?

Given the report make the decision on the number of racks?

Change Run Conditions

- Redefine a “day” to be 16 hours – *Run > Setup > Replication Parameters*
- Change Replication Length to 10 (of these) days

How can you make the above settings?

Describe how to model the lunch hour without failres?

Schedules

Describe how to model workers available for work fro 8-12 and 1 till 4?

- **Vary Capacity (number of units) of a resource over time**
- **In Resource Data module (spreadsheet view)**
  - For Rework Resource, change Type from **`Fixed Capacity`** to **`Based on Schedule`**
  - Two new columns – Schedule Name and Schedule Rule
  - Type in a schedule name (**`Rework Schedule`**)
  - Select a Schedule Rule – details of capacity decrease if the Resource is allocated to an entity
    - *Ignore* – Capacity goes down immediately for stat collection, but work goes on until finished … "break" could be shorter or gone
    - *Wait* – Capacity decrease waits until entity releases Resource, and "break" will be full but maybe start/end late
    - *Preempt* – Processing is interrupted, resumed at end of "break"

Describe and illustrate the schedule rules?

Schedules (cont'd.)

- **Define the actual Schedule the Resource will follow – Schedule data module (spreadsheet)**
  - Row already there since we defined **`Rework Schedule`**
  - Format Type is Duration for entries based on time past simulation beginning
  - Type is Capacity, for Resource schedule (more later on Arrival Type)
  - Click in Durations column, get Graphical Schedule Editor
    - *X*-axis is time, *Y*-axis is Resource Capacity
    - Click and drag to define the graph
    - Options button to control axis scaling, time slots in editor, whether schedule loops or stays at a final level forever
    - Can use Graphical Schedule Editor only if time durations are integers, and there are no Variables or Expressions involved

Schedules (cont'd.)

- Alternatively, right-click in the row, select Edit via Dialog
  - Enter schedule Name
  - Enter pairs for Capacity, Duration … as many pairs as needed
    - If all durations are specified, schedule repeats forever
    - If any duration is empty, it defaults to infinity
  - Can involve Variables, Expressions
- Another alternative – right-click in the row, select Edit via Spreadsheet
  - Enter capacity Value, Duration pairs

Resource Failures

- Usually used to model unplanned, random downtimes
- Can start definition in Resource or Failure module (Advanced Process panel) … we'll start in Failure
- Attach Advanced Process panel if needed, single-click on Failure, get spreadsheet view
- To create new Failure, double-click – add new row
- Name the Failure
- Type – Time-based, Count-based (we'll do Time)
- Specify Up, Down Time, with Units

Resource Failures (cont'd.)

- **Attach this Failure to the correct Resource**
  - Resource module, Failures column, Sealer row – click
  - Get pop-up Failures window, pick Failure Name **`Sealer Failure`** from pull-down list
  - Choose Failure Rule from **`Wait`**, **`Ignore`**, **`Preempt`** (as in Schedules)
- **Can have multiple Failures (separate names)**
- **Can re-use defined Failures for multiple Resources (operate independently)**

Describe how to collect queue size in categories of 10?

Frequencies

Describe how to measure time in queue and time in system and time prior to rework?

- **Record time-persistent occurrence frequency of variable, expression, or resource state**
  - Use here to record % of time rework queue is of length 0, (0, 10], (10, 20], … to give info on number of racks needed
- **Statistic data module (Advanced Process panel)**
  - Five Types of statistics, of which Frequencies is one
  - Specify Name (**`Rework Queue Stats`**), Frequency Type (**`Value`**)
  - Specify Expression to track and categorize
    - Right-click in field to get to Expression Builder
  - Report Label (**`Rework Queue Stats`**)
  - Pop-up secondary spreadsheet for Categories (browse file)

Describe the how to get the average, min and max for the queue size?

Frequencies (cont'd.)

- **Add another Frequency (in Statistic module) to give a finer description of the Sealer states**
  - Will produce statistics on proportion of time Sealer is in each of its *three* possible states – Busy, Idle, and Failed
- **Frequencies are not part of default Category Overview report – open Frequencies report from Project Bar (get a separate window for them)**

Results of Model 4-2

- **Differ from those of Model 4-1 since this is a longer run, modeling assumptions are different**
  - All of which causes underlying random-number stream to be used differently (Chapter 12)
- **Prep A/B didn't change (other than run length and random variation) … need statistical analysis of simulation output (Chapters 6, 7, 12)**
- **Sealer is more congested (it now fails)**
- **Rework is less congested (50% higher staffing)**
- **Frequencies report suggests one rack suffices about 95% of the time, two racks all the time**
  - See text for discussion of Standard, Restricted Percents

Utilizations – Some Fine Points

- **Two utilizations reported for each Resource**
  - *Instantaneous Utilization* is the time-average of the ratio of the number of units that are busy to the number of units that are scheduled
    - By definition, counts periods when zero units are scheduled as zero-utilization periods
  - *Scheduled Utilization* is the average number busy divided by the average number available
    - No division-by-zero problem, assuming there were ever any units of the Resource scheduled at all (if not, it shouldn't be in the model)
- **Identical for fixed-capacity Resource**
- **Can differ for Resources on a variable Schedule**
  - If Resource capacity varies among several different positive values, it's better to use Scheduled Utilization
  - See text for discussion of issues and even finer points

Model 4-3: Enhancing the Animation

- **Get "Spartan" generic default animation for some things (queues, connector-animation movement)**
  - Usually sufficient for verification, validation
- **Often want to customize, enhance it a bit**
  - More realism, impact
- **Can pull animation away from model logic in model window**
  - Useful for big models, complex animation
  - Set up Named Views for model logic, animation, or close-ups of parts of animation
- **Animation objects are connected to model logic**
  - Identifiers, physical location (Shift-drag to decouple)

Changing Animation Queues

- **Lengthen (click, drag, maybe hold shift) to "hold" more entities**
  - Simulation logic, results still OK even if animated queue overflows
- **Rotate to re-orient for realism**
- **Change the "form" of the queue from *Line* (the default) to *Point* — fixed places for entities**
  - Double-click on the queue
  - Select Type to be Point
  - Click Points… button
  - Successively click Add for points, then OK
  - Drag them around on screen
  - *Check* Rotate box to show entities turning

Describe how to animate the queue , resourse and entity in atrena?

Changing the Entity Pictures

- **Earlier – used Entity data module to assign different Initial Pictures to different Entity Types**
- **Now – customize the list, or alter the pictures in it**
  - *Edit > Entity Pictures*
  - Left column – names, pictures currently on the list
  - Right column –picture libraries (.plb filename extension)
  - Add a hand-drawn picture – Add button on left, name it in Value field at top, double-click on blank depressed button, then artwork (or paste in a copied graphics image)
    - New name won't appear in Entity data module until you type it there
  - Edit an existing picture – double-click, artwork
  - Copy a picture over from picture library

Adding Resource Pictures

- **Animate a Resource – Resource button in animate toolbar – get Resource Picture Placement window**
- **Left column – default pictures for different Resource states**
  - Attach logically to a Resource by Identifier pull-down list
  - Double-click to edit the artwork by hand, or paste in previously copied graphics images
  - Seize area – where seizing entity will “reside”
  - Multiple seize areas for multi-capacity Resources
- **Right column – picture libraries (.plb files) – can copy over to selected (depressed) state pictures**
- **Accept window, cross hairs, click to place**
  - Resize, reposition later

Adding Variables and Plots

- **Variable animation – just show a value of something as a number, watch it change**
  - Variable object 0.0 from Animate toolbar
  - Double-click, specify Expression to be shown (Expression Builder), and cosmetics
  - Resize, reposition later
- **Dynamic animated plots – discussed in Chapter 3**
- **Other animation objects from Animate toolbar**
  - Clock (TNOW), variety of formats
  - Level (thermometer) animation
  - Others discussed later

Describe how to add an animation for TNOW variable in arena?

Model 4-4: The Electronic Assembly and Test System with Part Transfers

- **Generalize Model 4-3**
- **All part transfers now take 2 minutes (not instant) … want to model and animate**
  - Includes:
    - Arriving parts to prep areas
    - Departing parts to appropriate exit
    - All internal part transfers
  - Regardless of distance ... will fix this (unrealistic) assumption in Chapter 8

Describe the station marker?

New Arena Constructs

- ***Station*** **– location where some process occurs**
  - Arrivals, manufacturing cells, departures
  - Each Station given a unique name
  - Can serve as an entry point for a section of model logic
  - *Station marker* represents a logical station in the flowchart/animation
- ***Station Transfer*** **– entities move between Stations without direct connection**
  - Several different types – we'll use *Routes* here, which allow for positive transfer time, but no other delays like "room" on the transitway or transporters
  - *Route paths* represent Routes in the flowchart/animation

Adding the Route Logic – From Arrival

- **Stations and Station Transfers affect both the model logic and the animation**
- **Start with Model 4-3 … change to Model 4-4**
- **For incoming parts (A and B) delete connection from Assign modules to “Prep” Process modules**
    - Replace with Station/Route module pairs
        - Station module (Advanced Transfer panel) – define entity’s current location
            - Module Name vs. Station Name
        - Route module (Advanced Transfer panel) – send entity out
            - Route Time, Destination Station
    - No direct connections exiting from the Route modules – Route module’s Destination Station Name defines that

Adding the Remaining Route Logic

- **Add Station modules for entry to each Prep area**
  - Station names are `Prep A Station`, `Prep B Station`, and are the destination stations for Routes after arrivals
- **Process modules for Prep A, Prep B unchanged**
- **After prep, entities connected to Route module to send to next station (sealer)**
  - Don’t need a separate Station module for outgoing side
- **Similar changes for rest of model**
  - Station modules for incoming parts into sealer, rework, each of three Record modules (entity exit points)
  - Route modules for outgoing parts out of sealer inspection, rework inspection (two for each Decide module – pass/fail)
- **Could run model now, get correct results … but no animation of transfers …**

Why Not Just Add Delays?

- **Simpler approach than the above to getting the two-minute transfer times:**
  - Insert a Process module with Action = Delay for 2 minutes on each relevant connection
  - Alternatively, use Delay module from Advanced Process panel
- **Actually this *would* work fine from modeling, numerical-output viewpoints**
- **But it would not allow animation of part transfers, so we'll proceed with the Stations and Routes**

Altering the Animation – Station Markers, Routes

- **Add animation for Stations and Routes**
- **Station button ▭ , Animate Transfer toolbar**
  - Attach Identifier to it from pull-down list of station names
  - Get cross hairs, place (click) marker in animation
  - Can place several station markers for the same logical station (to represent incoming, outgoing sides)
  - Can drag station markers around later
- **Route button R from Animate Transfer toolbar**
  - Options for appearance of entities as they travel the route
  - Get cross hairs; click in origin, destination Station Markers
    - Intermediate clicks for corners along the route
  - Can drag around endpoints, corners later

Describe how to add route animation?

Altering the Animation – Entity Pictures

- **Part B arrivals are in batches of four parts/batch**
  - But constant travel time to Prep B implies they travel “on top of each other” so it looks like just one part B
  - Try – change Route time from **`2`** to **`EXPO(2)`**, see separation along the route
- **Create illusion to animate the batch**
  - Assign module just after **`Part B Arrive`**
  - Add assignment of Entity Picture to **`Picture.Batch B`**
  - *Edit > Entity Pictures* to draw the new picture
    - Copy **`Picture.Part B`** and rename it **`Picture.Batch B`**
    - Double-click on picture, use Picture Editor to get four circles
  - When batch arrives to Prep B, change to single circle
    - Add Assign module after **`Prep B Arrival Station`**

Input Analysis: Specifying Model Parameters, Distributions

- ***Structural*** **modeling: what we've done so far**
  - Logical aspects — entities, resources, paths, etc.
- ***Quantitative*** **modeling**
  - Numerical, distributional specifications
  - Like structural modeling, need to observe system's operation, take data if possible

Deterministic vs. Random Inputs

- ***Deterministic*: nonrandom, fixed values**
  - Number of units of a resource
  - Entity transfer time (?)
  - Interarrival, processing times (?)

Describe how to make the intearrival time deterministic or random? Same for process time?

- ***Random* (a.k.a. *stochastic*): model as a distribution, “draw” or “generate” values from to drive simulation**
  - Transfer, Interarrival, Processing times
  - What distribution? What distributional parameters?
  - Causes simulation output to be random, too
- **Don’t just assume randomness away — validity**

Collecting Data

- **Generally hard, expensive, frustrating, boring**
  - System might not exist
  - Data available on the wrong things — might have to change model according to what's available
  - Incomplete, "dirty" data
  - Too much data (!)
- **Sensitivity of outputs to uncertainty in inputs**
- **Match model detail to quality of data**
- **Cost — should be budgeted in project**
- **Capture variability in data — model validity**
- **Garbage In, Garbage Out (GIGO)**

Using Data:
Alternatives and Issues

- **Use data "directly" in simulation**
  - Read actual observed values to drive the model inputs (interarrivals, service times, part types, …)
    - Arena ReadWrite module … see Model 10-2
  - All values will be "legal" and realistic
  - But can never go outside your observed data
  - May not have enough data for long or many runs
  - Computationally slow (reading disk files)
- **Or, fit probability distribution to data**
  - "Draw" or "generate" synthetic observations from this distribution to drive the model inputs
  - We've done it this way so far
  - Can go beyond observed data (good and bad)
  - May not get a good "fit" to data — validity?

Describe how to fit a distribution to an interarrival data?

What is required here is how you can you model using input analyser ----in details (but in short)

Difference between empirical (data based from the data) distribution and theoretical distribution (who has an equation)

Fitting Distributions to Data with the Arena Input Analyzer

- **Assume:**
  - Have sample data: Independent and Identically Distributed (IID) list of observed values from the actual physical system
  - Want to select or fit a probability distribution for use in generating inputs for the simulation model
- **Arena Input Analyzer**
  - Separate application, also accessible via Tools menu in Arena
  - Fits distributions, gives valid Arena expression for generation to paste directly into simulation model

Fitting Distributions to Data with the Arena Input Analyzer (cont'd.)

- **Fitting = deciding on distribution form (exponential, gamma, empirical, etc.) and estimating its parameters**
  - Several different methods (Maximum likelihood, moment matching, least squares, …)
  - Assess goodness of fit via hypothesis tests
    - H_0: fitted distribution adequately represents the data
    - Get *p* value for test (small = poor fit)

What is the empirical distribution?

- **Fitted "theoretical" vs. empirical distribution**
- **Continuous vs. discrete data, distribution**
- **"Best" fit from among several distributions**

Data Files for the Input Analyzer

- **Create the data file (editor, word processor, spreadsheet, ...)**
  - Must be plain ASCII text (save as text or export)
  - Data values separated by white space (blanks, tabs, linefeeds)
  - Otherwise free format
- **Open data file from within Input Analyzer**
  - *File > New* or 🗋
  - *File > Data File > Use Existing* or 📊
  - Get histogram, basic summary of data
  - To see data file: *Window > Input Data*
- **Can generate "fake" data file to play around**
  - *File > Data File > Generate New*

The Fit Menu

- **Fits distributions, does goodness-of-fit tests**
- **Fit a specific distribution form**
  - Plots density over histogram for visual “test”
  - Gives exact expression to Copy and Paste (Ctrl+C, Ctrl+V) over into simulation model
  - May include “offset” depending on distribution
  - Gives results of goodness-of-fit tests
    - Chi square, Kolmogorov-Smirnov tests
    - Most important part: *p-value*, always between 0 and 1:
      - Probability of getting a data set that’s more inconsistent with the fitted distribution than the data set you actually have, if the the fitted distribution is truly “the truth”
      - “Small” *p* (< 0.05 or so): poor fit (try again or give up)

The Fit Menu (cont'd.)

- **Fit all of Arena's (theoretical) distributions at once**
  - *Fit > Fit All* or
  - Returns the *minimum square-error* distribution
    - Square error = sum of squared discrepancies between histogram frequencies and fitted-distribution frequencies
    - Can depend on histogram intervals chosen: different intervals can lead to different "best" distribution
  - Could still be a poor fit, though (check *p* value)
  - To see all distributions, ranked: *Window > Fit All Summary* or

The Fit Menu (cont'd.)

- **"Fit" Empirical distribution (continuous or discrete): *Fit > Empirical***
  - Can interpret results as a Discrete or Continuous distribution
    - Discrete: get pairs (*Cumulative* Probability, Value)
    - Continuous: Arena will linearly interpolate *within* the data range according to these pairs (so you can never generate values outside the range, which might be good or bad)
  - Empirical distribution can be used when "theoretical" distributions fit poorly, or intentionally

Some Issues in Fitting Input Distributions

- **Not an exact science — no “right” answer**
- **Consider theoretical vs. empirical**
- **Consider range of distribution**
  - Infinite both ways (e.g., normal)
  - Positive (e.g., exponential, gamma)
  - Bounded (e.g., beta, uniform)
- **Consider ease of parameter manipulation to affect means, variances**
- **Simulation model sensitivity analysis**
- **Outliers, multimodal data**
  - Maybe split data set (see textbook for details)

Nonstationary Arrival Processes

- **External events (often arrivals) whose rate varies over time**
  - Lunchtime at fast-food restaurants
  - Rush-hour traffic in cities
  - Telephone call centers
  - Seasonal demands for a manufactured product

Describe how to model non-stationary interarrival times? Example peak rush hour?

These are modelled as arrival schedules

- **It can be critical to model this nonstationarity for model validity**
  - Ignoring peaks, valleys can mask important behavior
  - Can miss rush hours, etc.
- **Good model:** ***Nonstationary Poisson process***

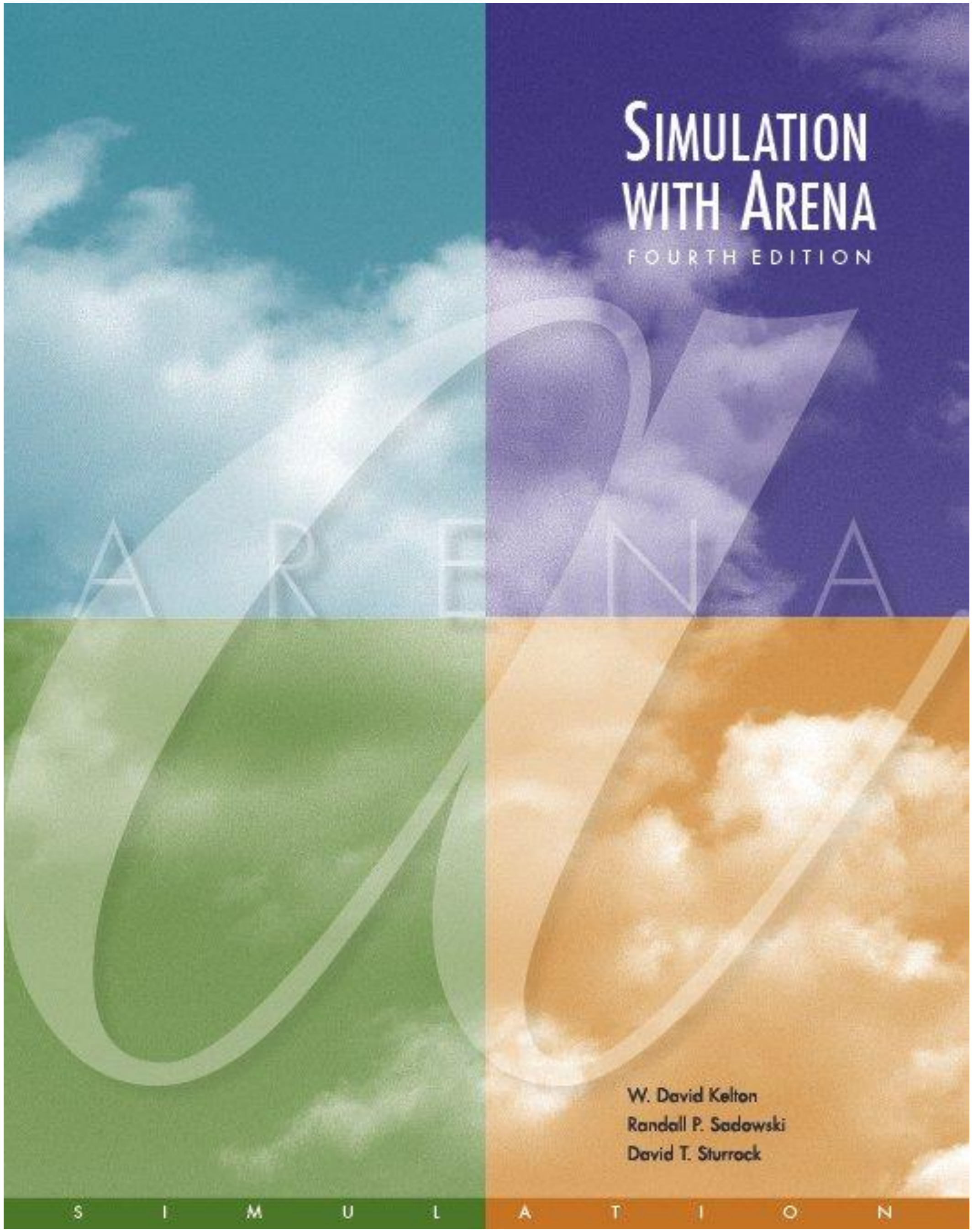
Nonstationary Arrival Processes (cont'd.)

- **Two issues:**
  - How to specify/estimate the *rate function*
  - How to generate from it properly during the simulation
- **Several ways to estimate rate function — we'll just do the *piecewise-constant* method**
  - Divide time frame of simulation into subintervals of time over which you think rate is fairly flat
  - Compute observed rate within each subinterval
  - In Arena, must convert to expected number of arrivals *per hour* on subintervals of time that need not be of one-hour length
    - Want expected 45 arrivals in a half hour; specify rate = 90 *per hour*
- **Example: Model 5-2 in Chapter 5**

When to use arrival schedule ----non-stationary data for example when people arrival is varied across the day ie it changes by the hour for example rush hour early morning then less and less

Now practice the following questions from the book
4-13, 4-15, 4-18, 4-27

I solved them earlier. Whether in lab or course and recorded them



Modeling Detailed Operations

Chapter 5

Last revision August 20, 2006

What We’ll Do ...

- **Model 5-1: Simple call center**
  - Lower-level modeling, Advanced Process panel
  - Three-way decisions, Variables, Expressions, Storages
  - Blocks panel
  - Terminating vs. steady-state operation
  - Logical (“fake”) entities
  - Terminating Condition in *Run > Setup*
- **Model 5-2: Enhanced call center**
  - Nonstationary Poisson arrival process
  - Sets – Resource, Counter
  - New Statistic data module Types
    - Counter, Time Persistent

What We'll Do ... (cont'd.)

- **Model 5-3: Enhanced call center with more output performance measures**
  - New Statistic data module Type
    - Output
  - Additional variable resources – look at staffing levels
- **Model 5-4: (*s*, *S*) inventory**
  - Not queueing
  - Choose to use low-level Blocks, Elements panels (SIMAN)
    - Can be done with higher-level panels

Model 5-1: Simple Call Center Setup

- **One phone number for customers to call in to**
  - 26 trunk lines, one needed for each call (incoming or outgoing, talking or on hold)
  - Arriving call finding no free trunk lines gets busy signal, goes away
    - Count number of such rejected calls
- **Calls arrive with interarrivals ~ EXPO (0.857) min.**
  - First call arrives at time 0
- **Three incoming call types**
  - Initial recording to decide ~ UNIF (0.1, 0.6) min.
  - Tech support (76%), sales (16%), order status (8%)

Model 5-1: Simple Call Center Setup (cont'd.)

- **Tech-support calls**
  - For product type 1 (25%), 2 (34%), or 3 (41%)
  - Needs qualified tech-support person
    - Two for type 1, three for type 2, three for type 3

      No crossover to another type ... change in Model 5-2
    - Separate FIFO queues for each type
    - Conversation time ~ TRIA (3, 6, 18) min. for all types
  - Then leaves system
- **Sales calls**
  - All the same
  - Four sales staff, all the same
    - One FIFO queue feeding all sales staff
  - Conversation time ~ TRIA (4, 15, 45)
  - Then leaves system

Model 5-1: Simple Call Center Setup (cont'd.)

- **Order-status calls**
  - All the same
  - Handled automatically by phone system
    - No limit on number in process at a time, except for trunk-line limit
  - "Conversation" time ~ TRIA (2, 3, 4)
  - After "conversation," 15% of callers opt to talk to a person
    - Routed to sales staff
    - Sales calls have higher priority (non-preemptive)
- **Center receives calls 8am – 6pm**
  - Must terminate arrival process at 6pm
  - Operate past 6pm if necessary to "flush out" all calls

Model 5-1: Simple Call Center Setup (cont'd.)

- **Output performance measures**
  - Number of calls attempted, rejected, completed
  - By call type – total time in system
  - By resource – time on hold, number of calls on hold
  - Resource utilization – personnel, trunk lines
- **Terminating or steady-state**
  - Time frame of interest for each replication
    - Terminating – specific starting, stopping conditions (this model)

      Stopping conditions could be of several forms – fixed time, count, condition (here)
    - Steady-state – output performance measures are a limit as simulated time $\rightarrow \infty$
    - Choice usually depends on intent of study, not on model logic

Model 5-1: Simple Call Center Modeling Panels

- **Basic Process**
  - Highest, fastest modeling level, usually the place to start
- **Advanced Process**
  - Smaller building elements, other functions, more detail
- **Advanced Transfer**
  - Entity movement, material handling
- **Blocks, Elements**
  - Lowest modeling level, SIMAN simulation *language*
  - Repeats some capabilities of higher-level panels
  - Some functions available only here
- **Other special-purpose panels**
  - License-dependent

Model 5-1: Simple Call Center Data Structure

- **Re-use data in several places**
  - Define once, global to whole model
  - Redefine once – modeling generality, user efficiency
- **Arena *Variables***
  - Store numbers (not formulas)
  - Define, initialize in Variable data module (Basic Process)
  - Can change during run (Assign module, other ways)
  - Scalar, 1-d array (vector), 2-d array (matrix)
- **Arena *Expressions* – generalize Variables**
  - Store formulas (as well as numbers)
    - Use math ops, numbers, random variates, Attributes, Variables, …
  - Define in Expression data module (Advanced Process)
  - Scalar, 1-d array (vector), 2-d array (matrix)

Model 5-1: Simple Call Center Arrivals, Direct to Service

- **Create attempted calls**
  - Entity type **`Incoming Call`**, change later
  - Max Arrivals = **`MaxCalls`**, Variable initialized to 999999
    - At 6pm (time 600 minutes) change this to 1 to cut off arrivals … later
- **Entity data module**
  - **`Incoming Call`** Entity Type already there
  - For Initial Picture, select **`Picture.Black Ball`**
- **Record module for an attempted call**
  - Add 1 to Counter Name **`Attempted Calls`**
  - Results – Category Overview report, User Specified

More detailed description – mouse over modules, read Data Tips that pop up

Model 5-1: Simple Call Center Arrivals, Direct to Service (cont'd.)

- **Decide module – Trunk Line Available?**
  - Type = 2-way by Condition
    - Select (logical) Expression for "If"
      - NR() is number of units of that resource that are busy now
      - MR() is number of units of that resource that exist now
  - False – Record rejected call counter, Dispose
  - True:
    - Seize a unit of **`Trunk Line`** Resource – Release later
      - Resources data module for Trunk Line and other Resource levels
    - Increment Variable **`Total WIP`** for number of active calls
      - Used in stopping rule at or after 6pm to sense if system is empty
    - Store module to enable entity animation during next Delay module
      - Add Storage animation separately, identify with this logical storage by name
      - Storage data module – entry made there by Store module
    - Delay module to listen to initial recording, make selection
      - Could have used Process module, but this is simpler, faster
    - Unstore module to make entity animation disappear

Alternate strategy – Queue module from Blocks panel ... details in text

Model 5-1: Simple Call Center Arrivals, Direct to Service (cont'd.)

- **Decide module – Determine Call Type**
  - Three-sided coin flip – Type = N-way by Chance
    - Add button for more sides of coin
    - Get new exit point for each Add, plus one for Else
    - Note that probabilities are entered as *percentages* (0-100, not 0-1)
    - Last entry is "else"
- **Direct call to one of tech support, sales, or order-status areas**

Backed each area with colored box
Alternative way to organize – Submodels

Model 5-1: Simple Call Center Tech-Support Calls

- **Assign module**
  - Change Entity Type for separating out in results
  - Change Entity Picture for animation
- **Store – Delay – Unstore for recording, product type selection**
- **Decide module for product type**
  - Different three-sided coin flip
  - Direct to appropriate Process module for that product type
- **Process modules for tech-support service**
  - Seize-Delay-Release
  - Seize a unit from appropriate multi-unit Resource
  - Use `Tech Time` defined in Expression data module
- **Proceed to system exit logic ... later**

Model 5-1: Simple Call Center Sales Calls

- **Assign module – change Entity Type, Picture**
- **Process module**
  - Seize-Delay-Release
  - Seize a unit of **`Sales`** Resource
- **Sales calls priority over order-status calls that seek a person?**
  - Queue data module, **`Process Sales Call.Queue`**
    - Type = Lowest Attribute Value

      Attribute Name = **`Sales Call Priority`**

      Undefined for sales calls, so has value 0 … will set to 1 for order-status calls that seek a person, putting sales calls ahead in the queue
    - Shared queue (with order-status calls seeking a person)
- **Proceed to system-exit logic**

Not the only way to do this

Model 5-1: Simple Call Center Order-Status Calls

- **Assign module – change Entity Type, Picture**
- **Delay block (Blocks panel) for robo-chat**
  - Includes Store/Unstore logic – alternative to earlier method
    - No automatic entry in Storage data module, so must enter manually
- **Decide module**
  - No sales person required – go directly to system-exit logic
  - Sales person required:
    - Assign module – set **`Sales Call Priority`** Attribute to 1 so these will have lower priority than real sales calls
    - Seize module for a unit of **`Sales`** resource
      - Define Queue Name = **`Process Sales Call.Queue`** – shared with sales calls
      - Process module does not allow for specifying a shared queue, so can't use here
    - Delay for conversation with sales person
    - Release the unit of **`Sales`** resource
- **Proceed to system-exit logic**

Model 5-1: Simple Call Center
System Exit

- **All calls of all types come here when finished**
- **Release module – release the unit of `Trunk Line` resource seized upstream**
- **Assign module – decrement `Total WIP` variable**
- **Record module – increment `Completed Calls` counter**
- **Dispose of call**

Model 5-1: Simple Call Center Arrival-Cutoff Logic

- **Used to "choke off" arrival stream at 6pm**
- **Create a single "logical" entity at time 600 min. (6pm)**
  - Overkill on making sure just one is created
    - Time Between Arrivals = 999999 min., Max Arrivals = 1
- **Assign module to set Variable `MaxCalls` to 1**
  - Recall use of **`MaxCalls`** for Max Arrivals in Create module for attempted calls
- **Dispose of this single logical entity**

Creative use of such "logical" (a.k.a. "fake") entities enhances modeling flexibility, power, detail

Model 5-1: Simple Call Center Run > Setup

- Replication Parameters tab (other tabs as usual)
- Base Time Units = Minutes
- Replication Length = Infinite (the default)
- Terminating Condition field:

$$\underbrace{\underbrace{\texttt{TNOW}}_{\substack{\textit{Arena}\\\textit{clock}\\\textit{Variable}}}\;\underbrace{\texttt{>= 600.0}}_{\substack{\textit{Greater than}\\\textit{or equal to 600}\\\underbrace{\textit{minutes}}_{\textit{Base Time Units}}\textit{, (6pm)}}}}_{\textit{It's 6pm or later}}\quad\underbrace{\underbrace{\texttt{\&\&}}_{\substack{\textit{Logical}\\\textit{``and''}}}}_{\textit{and}}\quad\underbrace{\underbrace{\texttt{Total WIP}}_{\substack{\textit{Variable we}\\\textit{maintained}\\\textit{in model}}}\;\underbrace{\texttt{== 0}}_{\substack{\textit{Equality}\\\textit{test for}\\\textit{zero}}}}_{\textit{there are no calls in the system.}}$$

Could have used ***`NR(Trunk Line)`*** *instead of* ***`Total WIP`***

Model 5-1: Simple Call Center Animation

- **Place three Storage animations**
  - Initial Recording Delay, Tech Call Recording Delay, Order Status Delay
  - Select proper Identifier in each from pull-down list
  - Graphic behaves like Queue animations
- **Four Queue animations**
  - Three tech-support call product types, sales
  - Came with four Process modules specifying Seize
- **Resource animations for three tech-support types, sales Resources**
  - Multi-unit Resource animations, as in Models 4-3, 4-4

Model 5-1: Simple Call Center Animation (cont'd.)

- **Variable animations for WIP at tech calls, sales**
  - For tech calls, Arena variable to animate is `Process Product Type 1 Tech Call.WIP`, etc. – pull-down list
  - For sales calls, must include order-status calls seeking a real person:

    `NR(Sales) + NQ(Process Sales Call.Queue)`
- **Plot number of trunk lines busy, `NR(Trunk Line)`**
- **Labeling, background boxes as in model logic**

Model 5-1: Simple Call Center Results *(one replication ... sample of size only one!!)*

- **Trunk-lines-busy plot**
  - Starts, ends at 0 – startup, termination logic
  - Capped at 26 during run
- **735 attempted calls (User Specified section)**
  - 644 completed, other 91 rejected
- **Sometimes see mixture of sales (green), order-status (blue) entities in sales queue**
- **Other "usual" outputs**
  - Times in system – separated out by call type
  - Queue lengths, times in queue – separated out by resource
  - Resource utilizations – normalized to [0, 1] by capacity

Model 5-2: Enhanced Call Center Changes

- **Incoming calls' arrival rate varies over day**
  - Probabilistic model – *Nonstationary Poisson process*
    - More in Section 12.3
  - Instead of a constant rate (= 1 / mean interarrival time), specify a rate *function*
    - Arena supports *piecewise-constant* rate function – "step" functions
      - Easy to specify, strong theoretical support
    - Rate-function specification:

In Arena, rates MUST be in arrivals per HOUR, regardless of base time units or time intervals

Caution – it's easy to generate this incorrectly ... see text for details

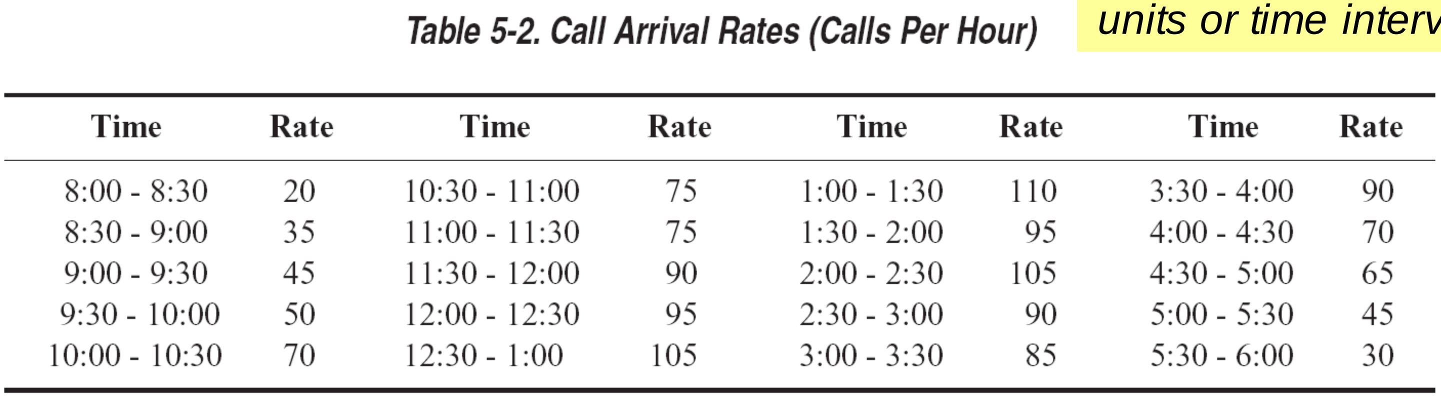


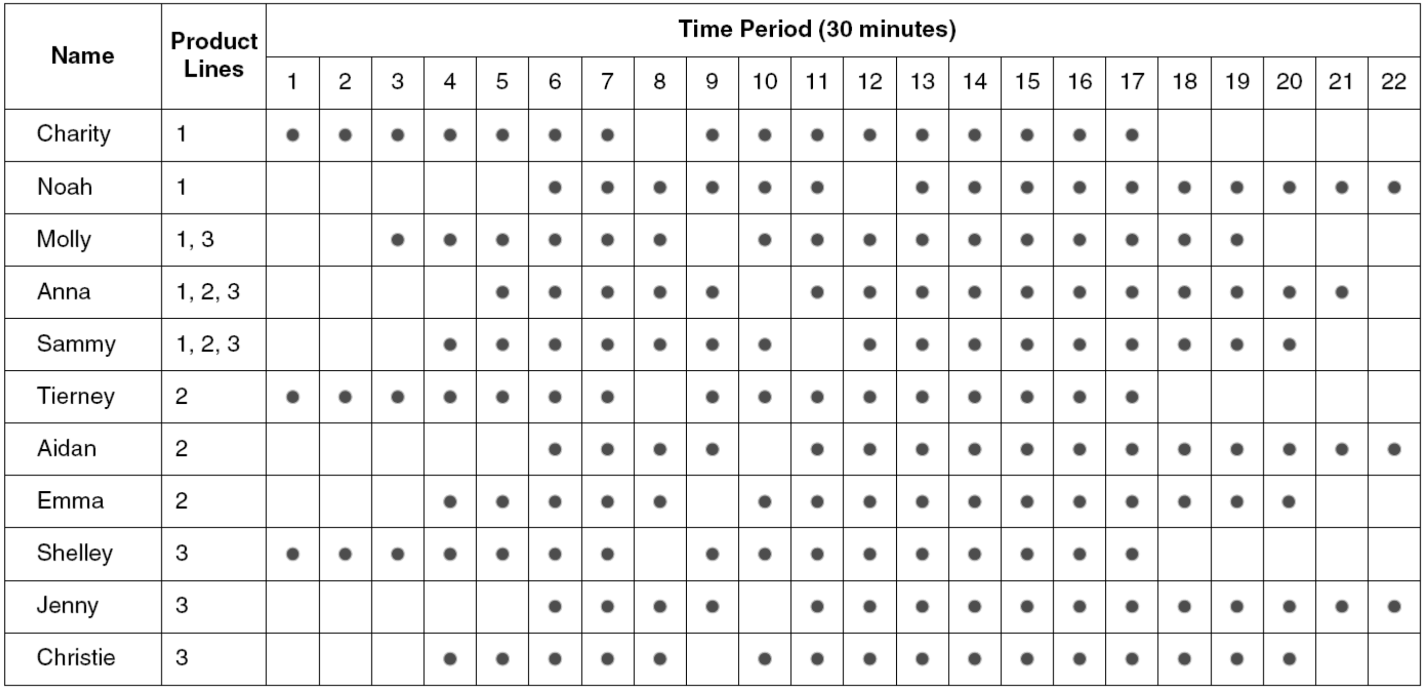
Table 5-2. Call Arrival Rates (Calls Per Hour)

Time	Rate	Time	Rate	Time	Rate	Time	Rate
8:00 - 8:30	20	10:30 - 11:00	75	1:00 - 1:30	110	3:30 - 4:00	90
8:30 - 9:00	35	11:00 - 11:30	75	1:30 - 2:00	95	4:00 - 4:30	70
9:00 - 9:30	45	11:30 - 12:00	90	2:00 - 2:30	105	4:30 - 5:00	65
9:30 - 10:00	50	12:00 - 12:30	95	2:30 - 3:00	90	5:00 - 5:30	45
10:00 - 10:30	70	12:30 - 1:00	105	3:00 - 3:30	85	5:30 - 6:00	30

Model 5-2: Enhanced Call Center Changes (cont'd.)

- **Sales-staff size varies over day**
  - Data in text, Schedule data module, `Sales Schedule`
- **Tech-support staff are partially cross-trained, work complicated schedule:**

Table 5-3. Technical Support Schedules



Name	Product Lines	Time Period (30 minutes)																					
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Charity	1	●	●	●	●	●	●	●		●	●	●	●	●	●	●	●	●					
Noah	1						●	●	●	●	●	●		●	●	●	●	●	●	●	●	●	●
Molly	1, 3			●	●	●	●	●	●		●	●	●	●	●	●	●	●	●	●			
Anna	1, 2, 3					●	●	●	●	●		●	●	●	●	●	●	●	●	●	●	●	
Sammy	1, 2, 3				●	●	●	●	●	●	●		●	●	●	●	●	●	●	●	●		
Tierney	2	●	●	●	●	●	●	●		●	●	●	●	●	●	●	●	●					
Aidan	2						●	●	●	●		●	●	●	●	●	●	●	●	●	●	●	●
Emma	2				●	●	●	●	●		●	●	●	●	●	●	●	●	●	●	●		
Shelley	3	●	●	●	●	●	●	●		●	●	●	●	●	●	●	●	●					
Jenny	3						●	●	●	●		●	●	●	●	●	●	●	●	●	●	●	●
Christie	3				●	●	●	●	●		●	●	●	●	●	●	●	●	●	●	●		

Will use Arena Sets concept to implement this cross training

Model 5-2: Enhanced Call Center Changes (cont'd.)

- **4% of tech-support calls cannot be handled during the call, need offline back-office research**
  - Original call ends, same original talk-time distribution, gives up its trunk line, but not counted (yet) as completed
  - Case sent to back office (outside model boundaries), takes EXPO (60) minutes to resolve
    - Offline research may be carried over night, completed on a later day
  - Answer goes back to *same* tech-support person who took original call, with higher priority than incoming calls, but still might have to queue for this person
  - This tech-support person requests a trunk line for outgoing call, higher priority than incoming calls, but still might have to queue, talks for TRIA (2 ,4 ,9) min., call is *now* completed
  - Track number of each product type after research is done

Model 5-2: Enhanced Call Center Data Structure

- **Resources, Schedules**
  - Resource, Schedule data modules
  - Trunk Line – fixed capacity at 26
  - Sales – on Schedule **`Sales Schedule`**
  - 11 individual tech-support people on individual schedules
    - Caution – must fill out each schedule to all 22 half-hour periods, with leading/trailing 0's if necessary … use Edit via Dialog or Spreadsheet, not graphical schedule editor
    - **`Ignore`** option to avoid shifting back schedule over multiple days
    - Include costing data for people in Resource data module
  - Define nonstationary arrival-rate function in Schedule module – **`Arrival Schedule`**
    - Enter trailing 0's in Edit via Dialog or Spreadsheet, not graphical schedule editor

Model 5-2: Enhanced Call Center Data Structure (cont’d.)

- **Sets – collect same-type items together**
  - Set, Advanced Set data modules (Basic, Advanced Process panels, resp.)
    - Refer to items in set by original name, or index (subscript) in set
  - Resource set for each tech-support product type
    - Members are those tech-support resources qualified
    - Individual resources already defined – Resource data module
    - Overlapping membership – some resources in multiple sets
    - Sets are ordered – here, put most versatile tech-support people at bottom, to “save” them for other calls … Preferred Order in Seize
    - Will Seize from a set in model
  - Counter set – one for each hour
    - Count number of rejected calls in each hour
    - Individual counters already defined – Statistic data module
    - Use results later to decide when to increase staffing

Model 5-2: Enhanced Call Center Modifying the Model

- **Call-arrivals, termination, Run > Setup**
  - Create module
    - Type = Schedule, Schedule Name = **`Arrival Schedule`**
  - Delete the entire arrival-cutoff section from Model 5-1
    - **`Arrival Schedule`** cuts off arrivals at 6pm, via 0 rate
  - Delete **`Total WIP`** variable used to terminate Model 5-1
    - Use built-in NR(**`Trunk Line`**) instead in Terminating Condition
    - Delete Assign modules used to manage **`Total WIP`**
  - Record module for rejected calls
    - Index into Counter Set **`Rejected Calls`** with index

      `AINT((TNOW/60) + 1)`

      which is 1 for first hour, 2 for second hour, etc. (AINT truncates decimals toward zero)

Model 5-2: Enhanced Call Center Modifying the Model (cont'd.)

- **Tech-support calls**
  - Same through **`Determine Product Type`** Decide
  - Add Assign modules for each product type thereafter
    - Entity Type to distinguish product type in reports
    - Entity Picture to distinguish product type in animation
    - Attribute **`Tech Call Type`** (1, 2, or 3 by product type) for routing
  - Process modules, Resources subdialogs
    - Type = Set
    - Set Name = Product 1, etc.
    - Selection Rule = Preferred Order, to select earlier entries in set first

      Recall – we put more versatile tech-support people lower in the set list
    - Save Attribute = **`Tech Agent Index`**

      Entity attribute, carried along, in case of back-office research to send back to this same tech-support person for return call

Model 5-2: Enhanced Call Center Modifying the Model (cont'd.)

- **Back office, returned tech-support calls – all new**
  - Entry via True branch (4%) in Decide module **`Backoffice Research and Return Call?`**
  - Release this call's trunk line – going offline now
  - Delay (with storage) for EXPO (60) back-office research
  - Increment **`Tech Return WIP(Tech Call Type)`**
    - 1-dim. Variable array – defined in Variable data module
    - **`Tech Call Type`** is 1, 2, or 3, assigned in earlier Assign module
  - Decide module **`Product Type?`** based on Entity Type
  - Seize the same tech-support person – higher priority
  - *Then* seize a trunk line (higher priority), make return call
  - Then release this trunk line, tech-support person
  - Decrement **`Tech Return WIP(Tech Call Type)`**
  - Send entity to final Record, *after* trunk-line release there

Model 5-2: Enhanced Call Center Modifying the Model (cont'd.)

- **Statistic data module**
  - Ten Counter-type statistics, discussed earlier
  - Four Time-Persistent statistics to track expressions
    - **`Backoffice Research WIP`** to track total number of cases in research, via NSTO(**`Backoffice Research Storage`**)
    - **`Tech 1 Total Online WIP Stat`**, etc., to track number of that product type in back office via Expression **`Tech 1 Total Online WIP`**, etc., defined in Expression data module as

`Process Product Type 1 Tech Call.WIP + Tech Return WIP(1)`, etc.

- **No changes needed in sales-calls or order-status-calls section of Model 5-1**

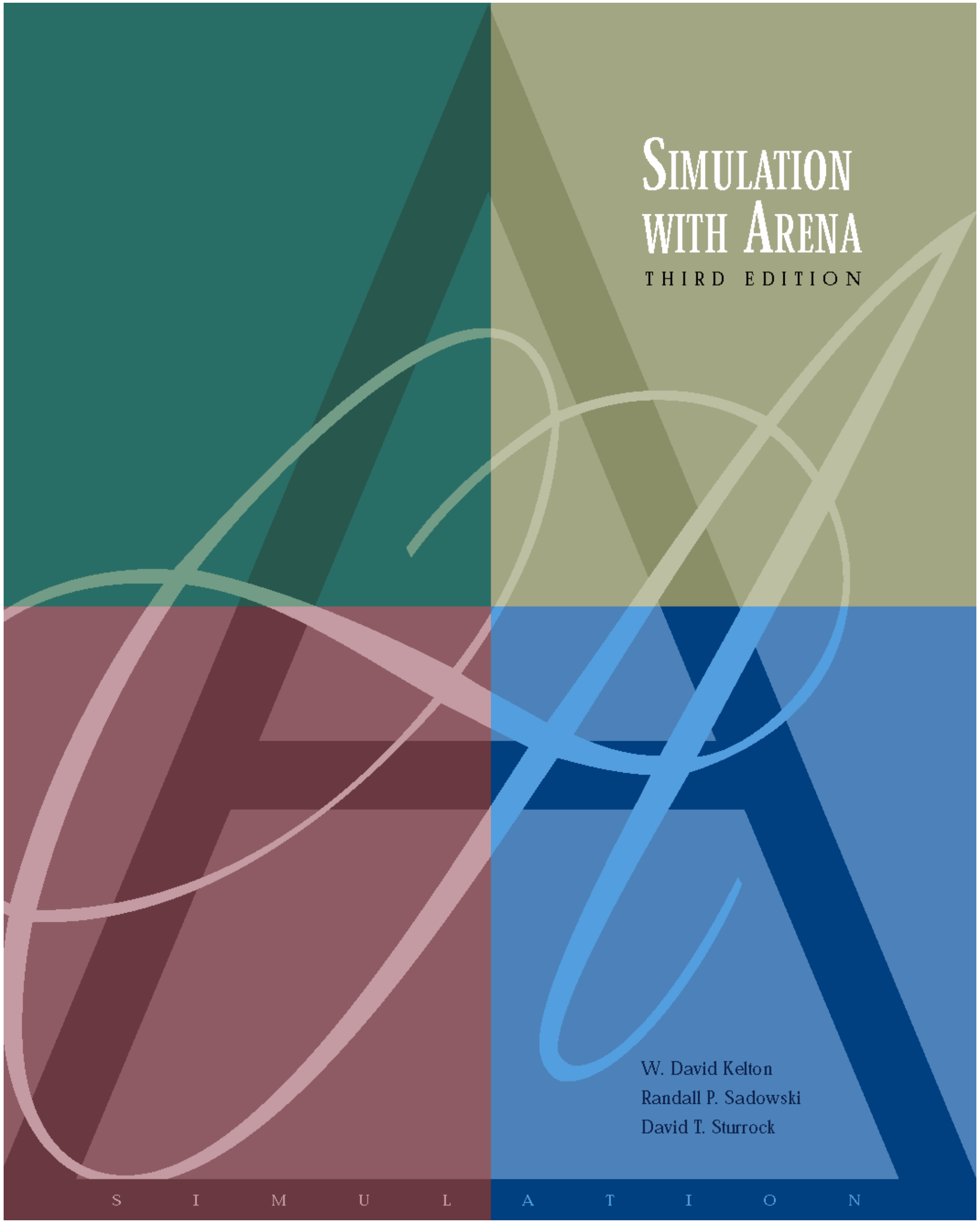
Model 5-2: Enhanced Call Center Modifying the Model (cont'd.)

- **Animation**
  - Delete **`Tech 1`**, **`Tech 2`**, and **`Tech 3`** resource animations
  - Change variables in three tech-support WIP displays to track total number of tech-support calls of that type present
  - New back-office storage animation, variable animation for number present
  - A new queue for each tech-support product type for return calls waiting for service
  - Added a resource animation (from a .plb library) for each individual tech-support person
    - Grouped by product type, colors for capabilities
- **Results**
  - Most rejected calls in hours 5-8 ... increase staff then?

Model 5-3: Overall Call-Center Stats Setup

- **Develop an overall operational-cost measure**
  - Two cost categories – staffing/resource, and poor service
- **Also develop overall measure of service, % of calls rejected**
- **Also add options for increased staffing, improvement**
- **Make 5 replications, focus on weekly costs**
  - IID replications, so will not carry over back-office research

Solve questions 5.2-5.14



Statistical Analysis of Output from Terminating Simulations

Chapter 6

Last revision June 8, 2003

What is required in this chapter is what are the software's associated with arena for output analysis, every software and what it its used for and

Important keywords in it such as controls scenarios, constraints objectives etc. every terminology

And how to used it step by step

What We’ll Do ...

- Time frame of simulations
- Strategy for data collection and analysis
- Confidence intervals
- Comparing two alternatives
- Comparing many alternatives via the Arena Process Analyzer (PAN)
- Searching for an optimal alternative with OptQuest

Introduction

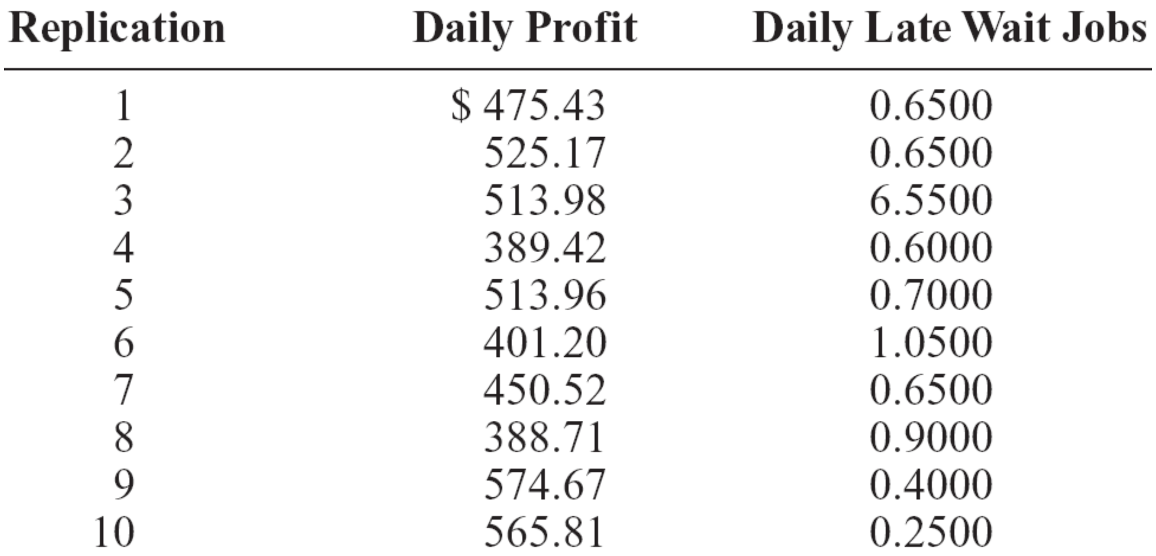
- **Random input leads to random output (RIRO)**
- **Run a simulation (once) — what does it mean?**
  - Was this run “typical” or not?
  - Variability from run to run (of the same model)?
- **Need statistical analysis of output data**
  - From a single model configuration
  - Compare two or more different configurations
  - Search for an optimal configuration
- **Statistical analysis of output is often ignored**
  - This is a big mistake – no idea of precision of results
  - Not hard or time-consuming to do this – it just takes a little planning and thought, then some (cheap) computer time

Time Frame of Simulations

- ***Terminating*: Specific starting, stopping conditions**
  - Run length will be well-defined (and finite)
- ***Steady-state*: Long-run (technically forever)**
  - Theoretically, initial conditions don't matter (but practically they usually do)
  - Not clear how to terminate a simulation run
- **This is really a question of intent of the study**
- **Has major impact on how output analysis is done**
- **Sometimes it's not clear which is appropriate**
- **Here: Terminating (steady-state in Section 7.2)**

Strategy for Data Collection and Analysis

- **For terminating case, make IID replications**
  - *Run > Setup > Replication Parameters*: Number of Replications field
  - Check both boxes for Initialize Between Replications
- **Separate results for each replication – Category by Replication report**
  - Model 5-2; Daily Profit, Daily Late Wait Jobs; 10 replications



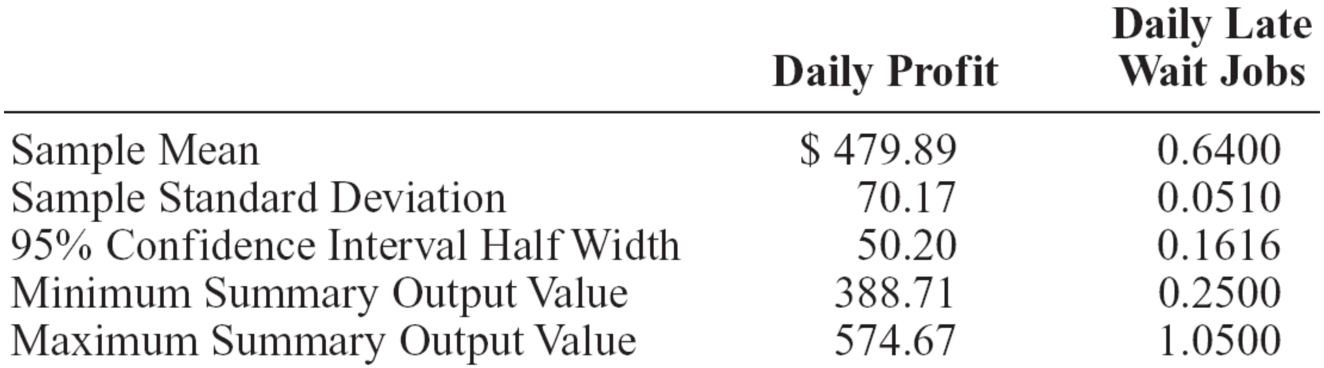
Replication	Daily Profit	Daily Late Wait Jobs
1	$ 475.43	0.6500
2	525.17	0.6500
3	513.98	6.5500
4	389.42	0.6000
5	513.96	0.7000
6	401.20	1.0500
7	450.52	0.6500
8	388.71	0.9000
9	574.67	0.4000
10	565.81	0.2500

Strategy for Data Collection and Analysis (cont'd.)

- **Category Overview report will have some statistical-analysis results of the output across the replications**
- **How many replications?**
  - Trial and error (now)
  - Approximate number for acceptable precision (below)
  - Sequential sampling (Chapter 12)
- **Turn off animation altogether for max speed**
  - *Run > Run Control > Batch Run (No Animation)*

Confidence Intervals for Terminating Systems

- Using formulas in Chapter 2, viewing the cross-replication summary outputs as the basic data:



	Daily Profit	Daily Late Wait Jobs
Sample Mean	$ 479.89	0.6400
Sample Standard Deviation	70.17	0.0510
95% Confidence Interval Half Width	50.20	0.1616
Minimum Summary Output Value	388.71	0.2500
Maximum Summary Output Value	574.67	1.0500

- Possibly most useful part – 95% confidence interval on expected values
- This information (except standard deviation) is in Category Overview report
  - If > 1 replication specified, Arena uses cross-replication data as above
  - Other confidence levels, graphics – Output Analyzer

Half Width and Number of Replications

- **Prefer smaller confidence intervals — *precision***
- **Notation:** $n = \text{no. replications}$

  $\overline{X} = \text{sample mean}$

  $s = \text{sample standard deviation}$

  $t_{n-1,1-\alpha/2} = \text{critical value from } t \text{ tables}$
- **Confidence interval:** $\overline{X} \pm t_{n-1,1-\alpha/2} \dfrac{s}{\sqrt{n}}$
- **Half-width =** $t_{n-1,1-\alpha/2} \dfrac{s}{\sqrt{n}}$

Want this to be "small," say ≤ h where h is prespecified

- **Can't control *t* or *s***
- **Must increase *n* — how much?**

Half Width and Number of Replications

(cont'd.)

- **Set half-width = *h*, solve for** $n = t^2_{n-1,1-\alpha/2}\,\dfrac{s^2}{h^2}$
- **Not really solved for *n* (*t*, *s* depend on *n*)**
- **Approximation:**
  - Replace *t* by *z*, corresponding normal critical value
  - Pretend that current *s* will hold for larger samples
  - Get $\boxed{n \cong z^2_{1-\alpha/2}\,\dfrac{s^2}{h^2}}$

_s_ = sample standard deviation from "initial" number n_0 of replications

- **Easier but different approximation:**

$$\boxed{n \cong n_0\,\frac{h_0^2}{h^2}}$$

h_0 = half width from "initial" number n_0 of replications

***n* grows quadratically as *h* decreases**

Half Width and Number of Replications

(cont'd.)

- **Application to automotive repair shop**
  - From initial 10 replications, 95% half-width on Daily Profit was ±\$50.20 ... let's get this down to ±\$20 or less
  - First formula: $n \cong 1.96^2(70.17^2/20^2) = 47.3$, so 48
  - Second formula: $n \cong 10(50.20^2/20^2) = 63.0$, so 63
  - Modified Model 5-2 into Model 6-1
    - Checked *Run > Run Control > Batch Run (No Animation)* for speed
    - In *Run > Setup > Replication Parameters*, changed Number of Replications to 100 (conservative based on above)
  - Got 492.63 ± 13.81, satisfying criterion

Interpretation of Confidence Intervals

- **Interval with random (data-dependent) endpoints that's supposed to have stated probability of containing, or covering, the expected valued**
  - "Target" expected value is a fixed, but unknown, number
  - Expected value = average of infinite number of replications
- **Not an interval that contains, say, 95% of the data**
  - That's a *prediction* interval … useful too, but different
- **Usual formulas assume normally-distributed data**
  - Never true in simulation
  - Might be approximately true if output is an average, rather than an extreme
  - Central limit theorem
  - Robustness, coverage, precision – see book (Model 6-2)

Comparing Two Alternatives

- **Usually compare alternative system scenarios, configurations, layouts, sensitivity analysis**
  - For now, just two alternatives ... more later
- **Model 6-3**
  - Model 6-1, but add file **`Daily Profit.dat`** to Statistic module, Output column, **`Daily Profit`** row
  - Saves this output statistic to this file for each replication
  - Two versions
    - *Base case* – all inputs as originally defined
    - *More-bookings case* – Change **`Max Load`** from 24 to 28 hours (allow more bookings per day ... increase utilization, profit? Maybe)

Comparing Two Alternatives (cont'd.)

- **Reasonable but not-quite-right idea**
  - Make confidence intervals on expected outputs from each alternative, see if they overlap
  - Base case:

    492.63 ± 13.81, or [478.82, 506.44]
  - More-bookings case:

    564.53 ± 22.59, or [541.94, 567.12]
  - But this doesn't allow for a precise, efficient statistical conclusion

No Overlap

Compare Means via the Output Analyzer

- **Output Analyzer is a separate application that operates on .dat files produced by Arena**
  - Launch separately from Windows, not from Arena
- **To save output values (Expressions) of entries in Statistic data module (Type = Output) – enter filename.dat in Output File column**
  - Just did for Daily Profit, not Daily Late Wait Jobs
  - Will overwrite this file name next time … either change the name here or out in Windows before the next run
  - .dat files are binary … can only be read by Output Analyzer

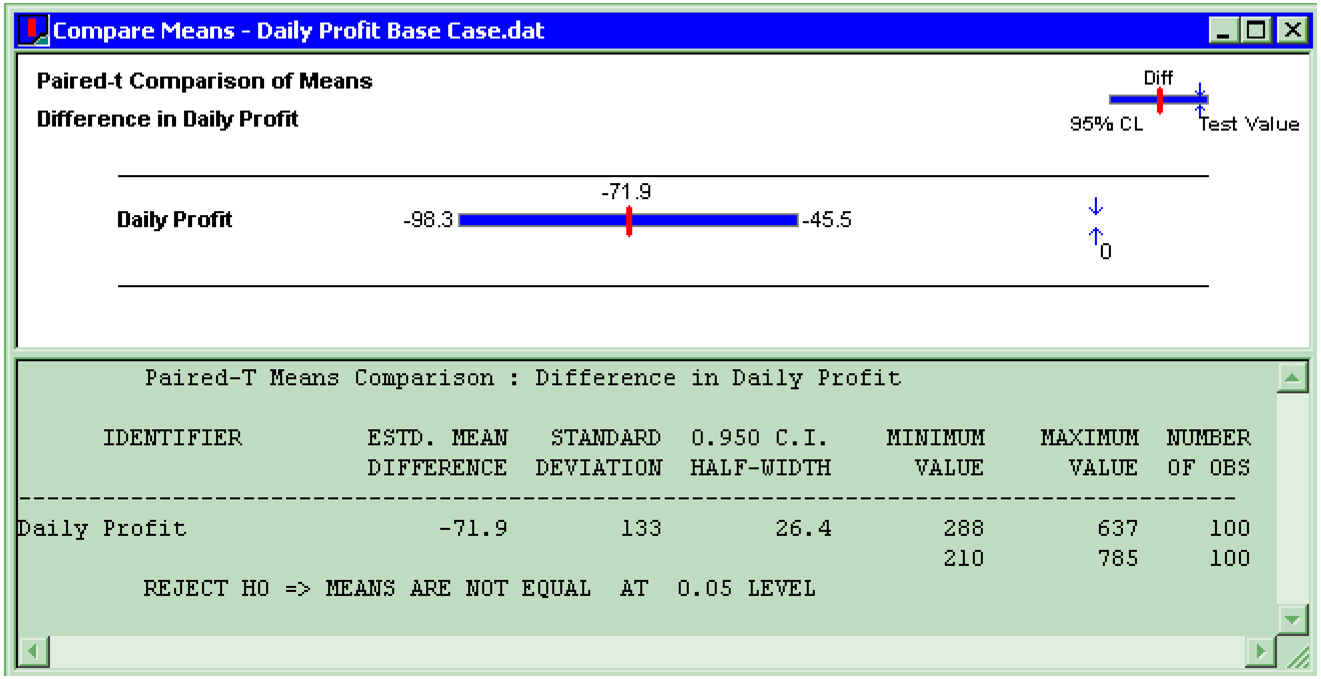
Compare Means via the Output Analyzer
(cont'd.)

- **Start Output Analyzer, open a new data group**
  - Basically, a list of .dat files of current interest
  - Can save data group for later use – .dgr file extension
  - Add button to select (Open) .dat files for the data group
- ***Analyze > Compare Means*** **menu option**
  - Add data files … "A" and "B" for the two alternatives
  - Select "Lumped" for Replications field
  - Title, confidence level, accept Paired-t Test, Scale Display

Compare Means via the Output Analyzer

(cont'd.)

- **Results:**



Paired-t Comparison of Means

Difference in Daily Profit

```
      Paired-T Means Comparison : Difference in Daily Profit

   IDENTIFIER         ESTD. MEAN  STANDARD  0.950 C.I.   MINIMUM    MAXIMUM  NUMBER
                      DIFFERENCE  DEVIATION HALF-WIDTH     VALUE      VALUE  OF OBS
------------------------------------------------------------------------------------
Daily Profit               -71.9        133       26.4       288        637     100
                                                             210        785     100
         REJECT H0 => MEANS ARE NOT EQUAL  AT  0.05 LEVEL
```

- **Confidence interval on difference misses 0, so conclude that there *is* a (statistically) significant difference**

Evaluating Many Alternatives with the Process Analyzer (PAN)

- **With (many) more than two alternatives to compare, two problems are**
  - Simple mechanics of making many parameter changes, making many runs, keeping track of many output files
  - Statistical methods for drawing reliable and useful conclusions
- **Process Analyzer (PAN) addresses these**
- **PAN operates on program (.p) files – produced when .doe file is run (or just checked)**
- **Start PAN from Arena (*Tools > Process Analyzer*) or via Windows**
- **PAN runs on its own, separate from Arena**

PAN Scenarios

- **A *scenario* in PAN is a combination of:**
  - A program (.p) file
  - Set of input *controls* that you choose
    - Chosen from Variables and Resource capacities – think ahead
    - You fill in specific numerical values
  - Set of output *responses* that you choose
    - Chosen from automatic Arena outputs or your own Variables
    - Values initially empty … to be filled in after run(s)
  - To create a new scenario in PAN, double-click where indicated, get Scenario Properties dialog
    - Specify Name, Tool Tip Text, .p file, controls, responses
    - Values of controls initially as in the model, but *you can change them in PAN* – this is the real utility of PAN
    - Duplicate (right-click, Duplicate) scenarios, then edit for a new one
  - Think of a scenario as a row

PAN Projects and Runs

- **A *project* in PAN is a collection of scenarios**
  - Program files can be the same .p file, or .p files from different model .doe files
  - Controls, responses can be the same or differ across scenarios in a project – usually will be mostly the same
  - Think of a project as a collection of scenario rows – a table
  - Can save as a PAN (.pan extension) file
- **Select scenarios in project to run (maybe all)**
- **PAN runs selected models with specified controls**
- **PAN fills in output-response values in table**
  - Equivalent to setting up, running them all “by hand” but much easier, faster, less error-prone

Model 6-4 for PAN Experiments

- **Same as Model 6-3 except remove Output File entry in Statistic module**
  - PAN will keep track of outputs itself, so this is faster
- **Controls – set up a formal 2^3 factorial experiment**

Control (*factor*)	"–" Level	"+" Level
`Max Load`	20	40
`Max Wait`	1	7
`Wait Allowance`	0.5	2.0

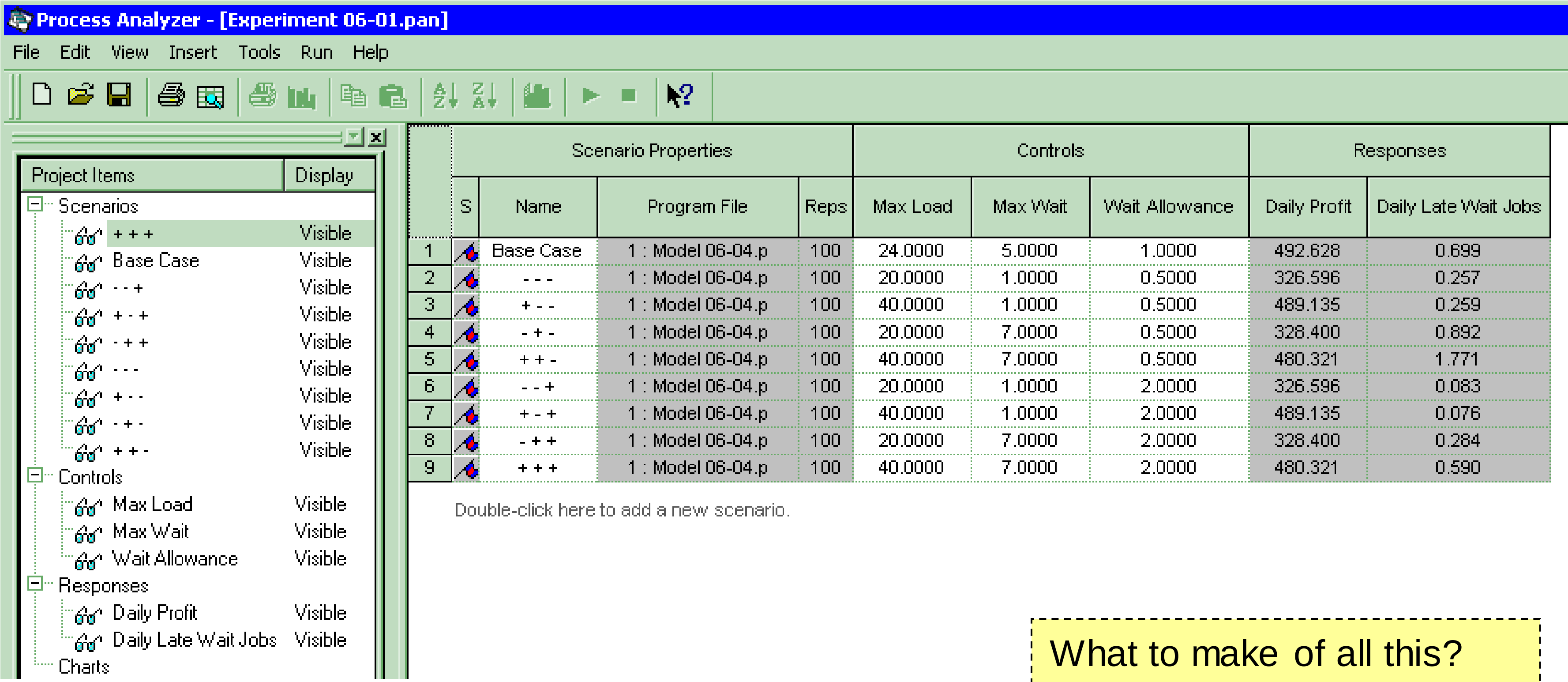
2^3 = 8 Scenarios
Also do Base Case

- **Responses**
  - `Daily Profit`
  - `Daily Late Wait Jobs`

Not *required* to do a designed experiment with PAN, but this is more informative than haphazard fooling around

Running Model 6-4 with PAN

- **Scenarios**
  - Select all to run (click on left of row, Ctrl-Click or Shift-Click for more)
  - To execute, ▶ or *Run* > *Go* or F5



Project Items	Display
Scenarios	
+ + +	Visible
Base Case	Visible
- - +	Visible
+ - +	Visible
- + +	Visible
- - -	Visible
+ - -	Visible
- + -	Visible
+ + -	Visible
Controls	
Max Load	Visible
Max Wait	Visible
Wait Allowance	Visible
Responses	
Daily Profit	Visible
Daily Late Wait Jobs	Visible
Charts	

	Scenario Properties				Controls			Responses	
	S	Name	Program File	Reps	Max Load	Max Wait	Wait Allowance	Daily Profit	Daily Late Wait Jobs
1		Base Case	1 : Model 06-04.p	100	24.0000	5.0000	1.0000	492.628	0.699
2		- - -	1 : Model 06-04.p	100	20.0000	1.0000	0.5000	326.596	0.257
3		+ - -	1 : Model 06-04.p	100	40.0000	1.0000	0.5000	489.135	0.259
4		- + -	1 : Model 06-04.p	100	20.0000	7.0000	0.5000	328.400	0.892
5		+ + -	1 : Model 06-04.p	100	40.0000	7.0000	0.5000	480.321	1.771
6		- - +	1 : Model 06-04.p	100	20.0000	1.0000	2.0000	326.596	0.083
7		+ - +	1 : Model 06-04.p	100	40.0000	1.0000	2.0000	489.135	0.076
8		- + +	1 : Model 06-04.p	100	20.0000	7.0000	2.0000	328.400	0.284
9		+ + +	1 : Model 06-04.p	100	40.0000	7.0000	2.0000	480.321	0.590

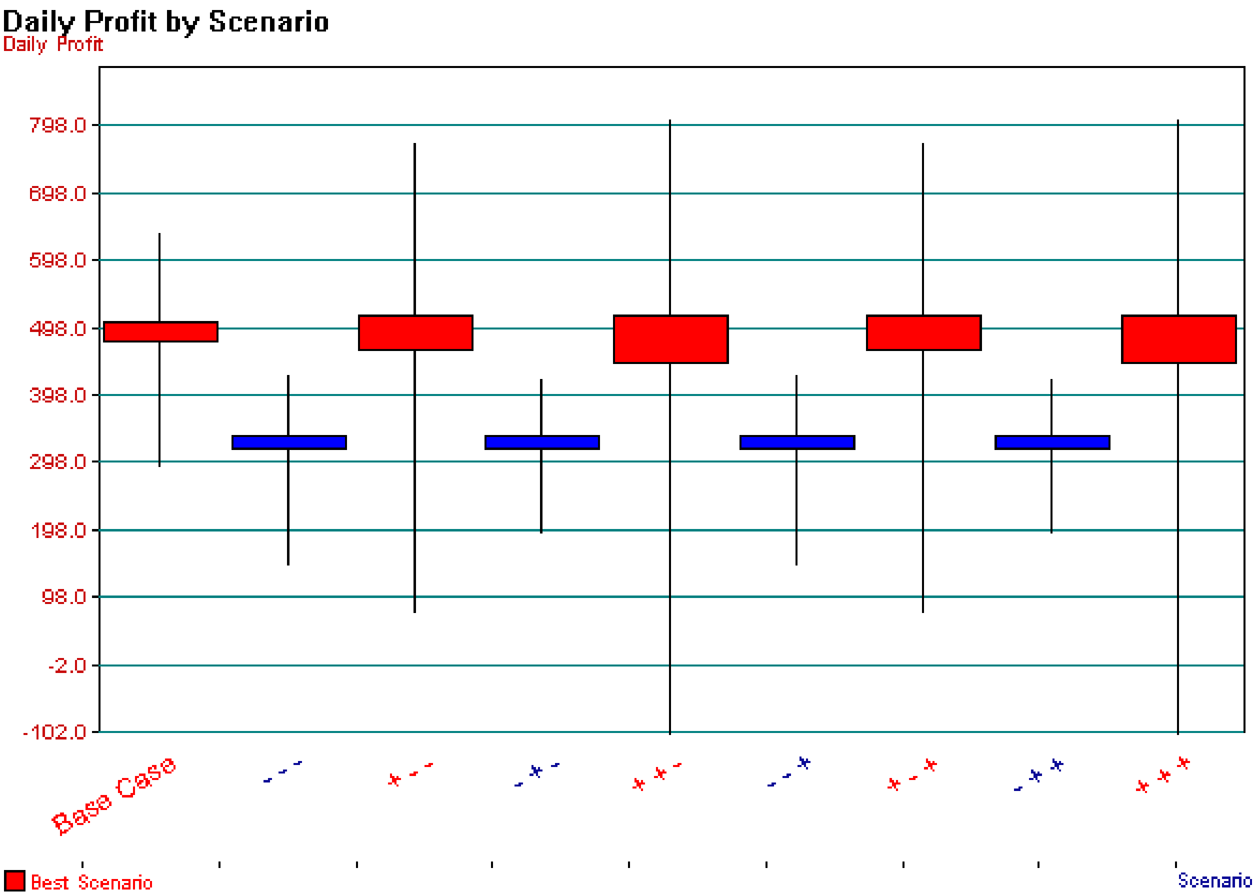
Double-click here to add a new scenario.

What to make of all this?
Statistical meaningfulness?

Statistical Comparisons with PAN

- **Model 6-4 alternatives were made with 100 replications each**
  - Better than one replication, but what about statistical validity of comparisons, selection of “the best”?
- **Select Total Cost column, *Insert > Chart* (or 📊 or right-click on column, then Insert Chart)**
  - Chart Type: Box and Whisker
  - Next, Total Cost; Next defaults
  - Next, Identify Best Scenarios
    - Bigger is Better, Error Tolerance = 0 (not the default)
    - Show Best Scenarios; Finish

Statistical Comparisons with PAN (cont'd.)

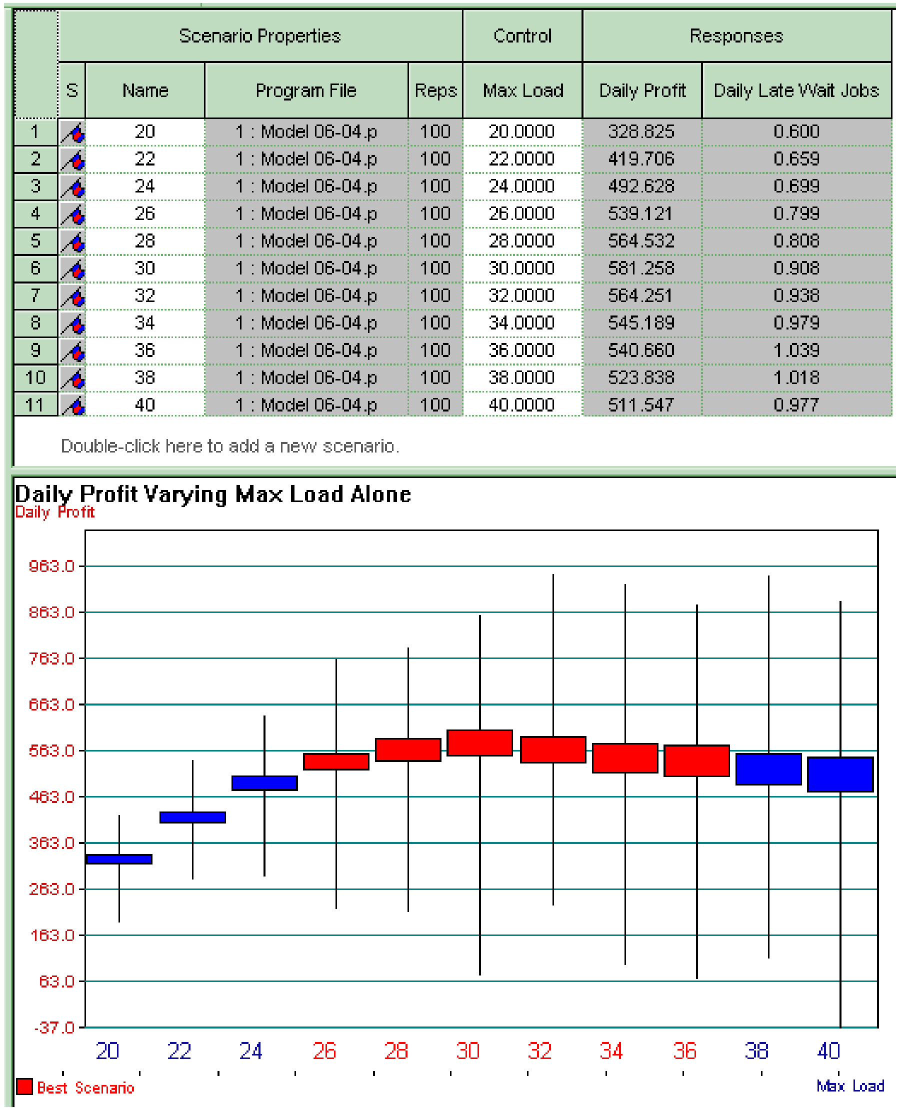


- **Vertical boxes: 95% confidence intervals**
- **Red scenarios statistically significantly better than blues**
  - More precisely, red scenarios are 95% sure to contain the best one
  - Narrow down red set – more replications, or Error Tolerance > 0
  - More details in book

A Follow-Up PAN Experiment

- **From 2^3 factorial experiment, it's clear that `Max Load` matters the most, and bigger appears better**
  - It's factor 1, varying between "−" and "+" in each scenario as ordered there, creating clear down/up/down/up pattern
  - Could also see this by computing *main effects* estimates
    - Consult an experimental-design text
- **Eliminate other two factors (fix them at their base-case levels) and study `Max Load` alone**
  - Let it be 20, 22, 24, ..., 40
  - Set up a second PAN experiment to do this, created chart as before

A Follow-Up PAN Experiment (cont’d.)



	Scenario Properties				Control	Responses	
	S	Name	Program File	Reps	Max Load	Daily Profit	Daily Late Wait Jobs
1		20	1 : Model 06-04.p	100	20.0000	328.825	0.600
2		22	1 : Model 06-04.p	100	22.0000	419.706	0.659
3		24	1 : Model 06-04.p	100	24.0000	492.628	0.699
4		26	1 : Model 06-04.p	100	26.0000	539.121	0.799
5		28	1 : Model 06-04.p	100	28.0000	564.532	0.808
6		30	1 : Model 06-04.p	100	30.0000	581.258	0.908
7		32	1 : Model 06-04.p	100	32.0000	564.251	0.938
8		34	1 : Model 06-04.p	100	34.0000	545.189	0.979
9		36	1 : Model 06-04.p	100	36.0000	540.660	1.039
10		38	1 : Model 06-04.p	100	38.0000	523.838	1.018
11		40	1 : Model 06-04.p	100	40.0000	511.547	0.977

Double-click here to add a new scenario.

- Here, profit-maximizing **`Max Load`** is about 30
- But **`Daily Late Wait Jobs`** keeps increasing (worsening) as **`Max Load`** increases
  - At profit-maximizing **`Max Load`** = 30, it’s 0.908 job/day, which seems bad since we only take 5 wait jobs/day
  - Would like to require that it be at most 0.75 job/day … still want to maximize **`Daily Profit`**
  - Allow other two factors back into the picture …

Searching for an Optimal Alternative with OptQuest

- **The scenarios we've considered with PAN are just a few of many possibilities**
- **Seek input controls maximizing `Daily Profit` while keeping `Daily Late Wait Jobs` ≤ 0.75**
- **Formulate as an optimization problem:**

Maximize `Daily Profit`

Subject to 20 ≤ `Max Load` ≤ 40
1 ≤ `Max Wait` ≤ 7
0.5 ≤ `Wait Allowance` ≤ 2.0

`Daily Late Wait Jobs` < 0.75

Objective function is the simulation model

Constraints on the input control (decision) variables

An output *requirement*, not an input *constraint*

Could also have constraints on linear combinations of input control variables (but we don't in this problem)

  - Reasonable starting place – best acceptable scenario so far: (the base case, actually)
  - Where to go from here? Explore all of feasible three-dimensional space exhaustively? ***No***.

OptQuest

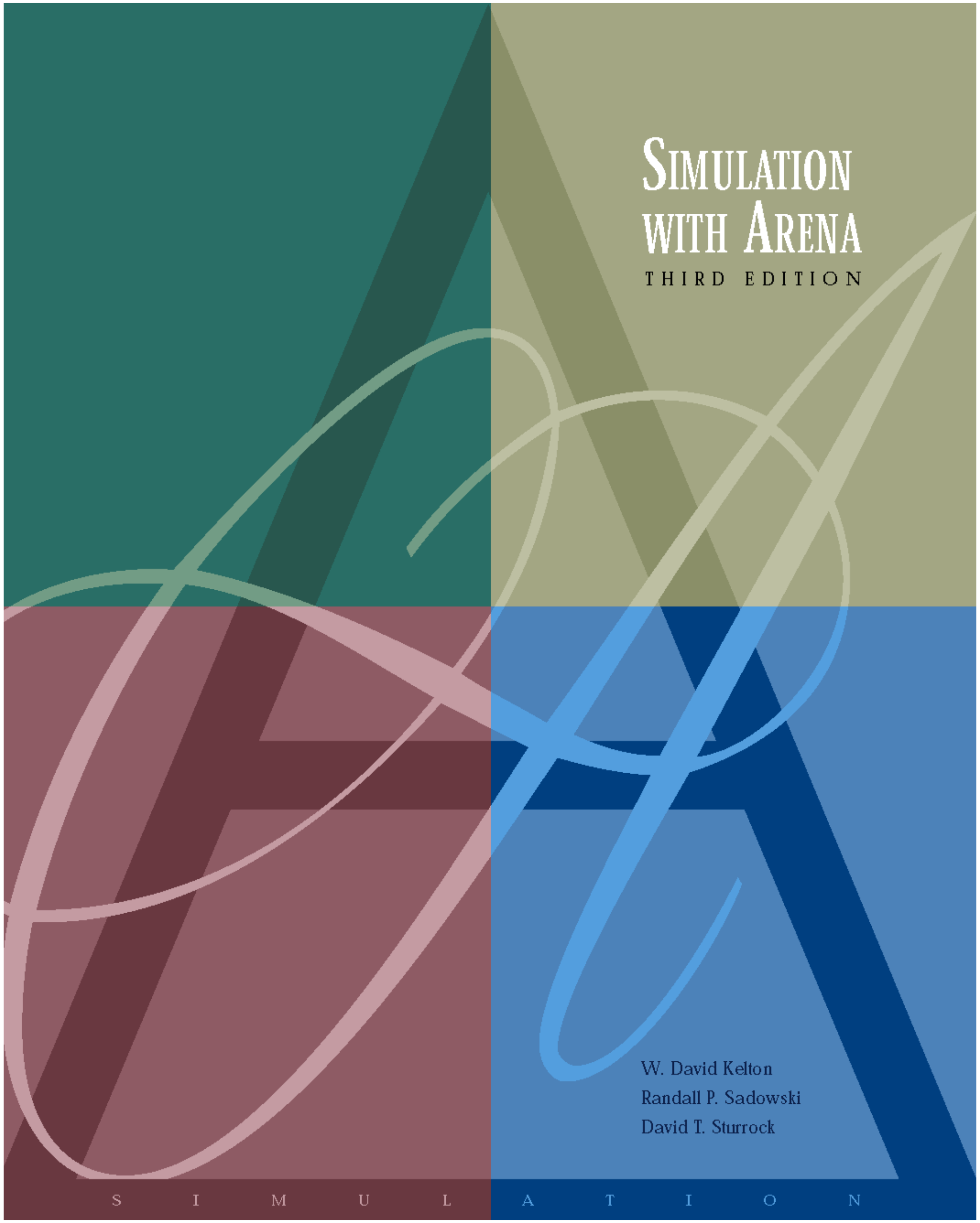
- **OptQuest searches intelligently for an optimum**
  - Like PAN, OptQuest
    - Runs as a separate application … can be launched from Arena
    - "Takes over" the running of your model
    - Asks that you identify the input controls and the output (just one) response objective
  - Unlike PAN, OptQuest
    - Allows you to specify constraints on the input controls
    - Allows you to specify requirements on outputs
    - Decides itself what input-control-value combinations to try
    - Uses internal heuristic algorithms to decide how to change the input controls to move toward an optimum configuration
- **You specify stopping criterion for the search**

Using OptQuest

- ***Tools > OptQuest for Arena***
- **New session (*File > New* or Ctrl+N or 🗋)**
  - Make sure the desired model window is active
- **Select controls – Variables, Resource levels**
  - **`Max Load`**, Lower Bound = 20, Upper Bound = 40, Conts.
  - **`Max Wait`**, Lower Bound = 1, Upper Bound = 7, Discrete (Input Step Size 1)
  - **`Wait Allowance`**, Lower Bound = 0.5, Upper Bound = 2, Conts.
- **Constraints – none here other than earlier Bounds**
- **Objective and Requirement**
  - **`Daily Profit`** Response – Select Maximize Objective
  - **`Daily Late Wait Jobs`** Response – Select Requirement, enter 0.75 for Upper Bound

Using OptQuest (cont'd.)

- **Options window – computational limits, procedures**
  - Time tab – run for 20 minutes
  - Precision tab – vary number of replications from 10 to 100
  - Preferences tab – various settings (accept defaults)
- **Can revisit Controls, Constraints, Objective and Requirements, or Options windows via** →▮ ⧗ ▮→ ◷
- **Run via wizard (first time through a new project), or *Run > Start* or** ▶
- ***View > Status and Solutions* and *View > Performance Graph* to watch progress**
- **Can't absolutely guarantee a true optimum**
  - Usually finds far better configuration than possible by hand



Intermediate Modeling and Steady-State Statistical Analysis

Chapter 7

Last revision June 9, 2003

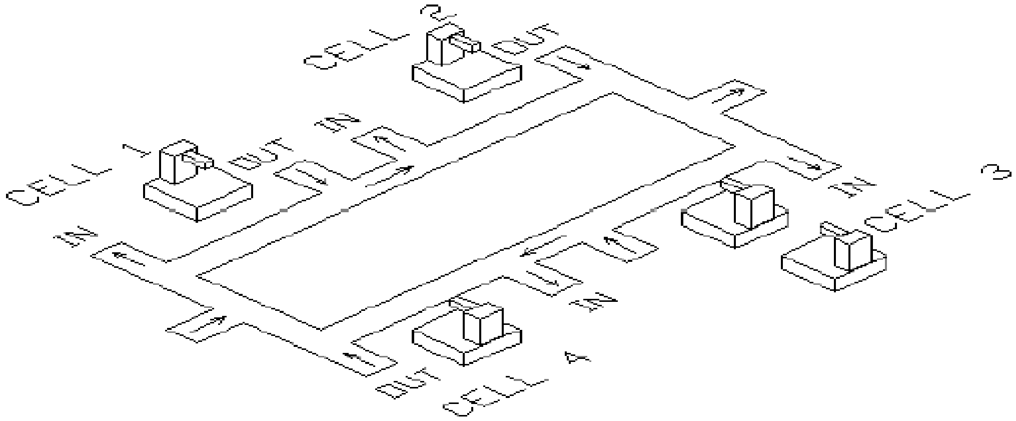
What is important here is to solve the model using the Route block and the sequences.

What is the replicate length and the warm up period

What We'll Do ...

- **Model 7-1: A small manufacturing system**
  - Entity-dependent Sequences
  - Data requirements and availability
  - Verification (debugging)
- **Statistical analysis of steady-state simulations**
  - Warmup and run length
  - Truncated replications
  - Batching
  - Other methods and goals

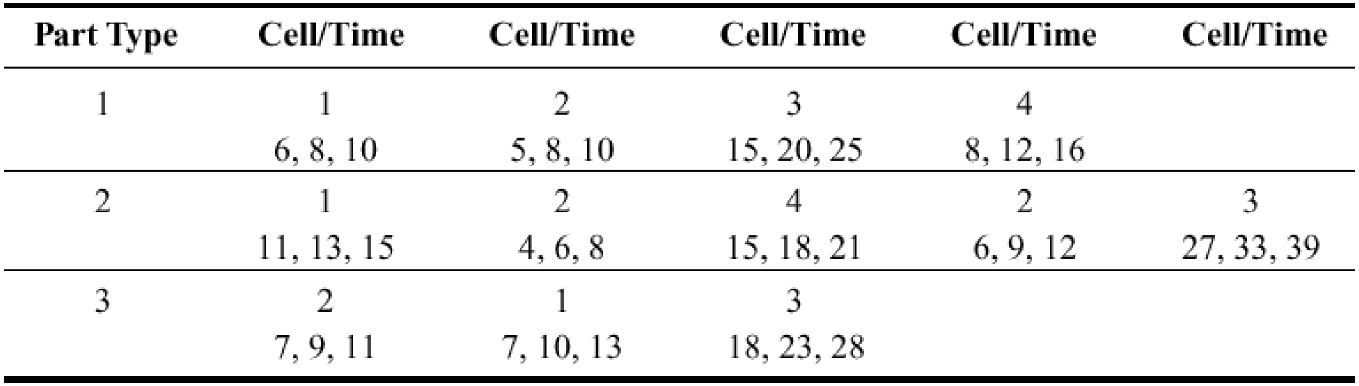
Model 7-1:
A Small Manufacturing System



- **Part arrivals, four cells, part departures**
- **Cells 1, 2, and 4: single machine each**
- **Cell 3: two machines — newer one 20% faster**
  - Need: way to model non-identical resource units
- **Circular layout of cells**
- **Parts enter at left, exit at right, travel only clockwise, all transfer times = 2 min. (realistic?)**

A Small Manufacturing System (cont'd.)

- **Three separate part types**
  - Interarrivals (all types merged) ~ expo(13) minutes
  - 26% type 1, 48% type 2, 26% type 3
- **Different part types follow different routes, have different (triangular) processing times:**



Part Type	Cell/Time	Cell/Time	Cell/Time	Cell/Time	Cell/Time
1	1 6, 8, 10	2 5, 8, 10	3 15, 20, 25	4 8, 12, 16	
2	1 11, 13, 15	2 4, 6, 8	4 15, 18, 21	2 6, 9, 12	3 27, 33, 39
3	2 7, 9, 11	1 7, 10, 13	3 18, 23, 28		

Parameters are for the slow machine at Cell 3.

- **Observe utilizations, time/number in queues, cycle times (times in system) by part type**
- **Run for 32 hours**

New Arena Concepts

- **Non-identical machines at Cell 3**
- **Different entity types follow different process plans**
  - Previous models – all entities went through same sequence of stations, maybe with Decides for branching
  - Now, need process plan with automatic routing by entity type – different *Sequence* assigned to each entity (like an attribute), and entity follows its own sequence
  - Won’t use direct Connect or Routes … instead we tell entities departing from modules to follow their own Sequence
  - Arena internally keeps track of where entity is, where it will go next

The Modeling Approach

- **Usually there are many ways to build a (correct) Arena model**
  - And also many ways to do so incorrectly …
- **Important to think about data structures**
  - What data are available?
  - How will they be stored in the model?
- **For this model …**
  - Use Sequence for part transfer (described below)
  - As part of Sequence definition, can define Attributes
    - Do for processing times at all cells but Cell 1
  - Use an Expression for processing times at Cell 1
  - Use Variables for new-machine speedup at Cell 3, part transfer times

Sequence Data Module

- **Advanced Transfer panel**
- **Double-click for new row for each process plan**
  - Name for each Sequence
  - Open Steps column for subdialog
    - Define ordered sequence of Stations to be visited in the Sequence … must have Station Names already defined
    - Double-click to add a new Station to the bottom of the Sequence list; right-click to insert/delete a row
    - Name for each step
    - Possible Assignments of Attribute, Variable, Pictures, etc. at each station in the Sequence … this is done *before* transferring the entity to this step in the sequence
    - In this model, Attribute assignment used to attach **Process Time** Attribute to entity for the next Cell (except for Cell 1)

Sequence Data Module (cont'd.)

- **Assign Sequence Name to entities that follow it**
- **In Route modules, select Sequence as Destination Type (rather than Station)**
  - Departing entity looks in its own sequence to know where to go next
- **Arena tracks Sequence-following entities via automatic attributes**
  - Sequence name, NS (or Entity.Sequence)
  - Station (where entity is or is going to), M (or Entity.Station)
  - JobStep along the sequence, IS (or Entity.JobStep)
- **Normally, entity is assigned a Sequence, travels its route, then exits**
  - Can interrupt this sequence, jump forward/backward (tricky)
- **Remember to define the "exit" station**

Expression Data Module

- **Advanced Process panel**
- **Use for processing times at Cell 1**
  - Could have done in Sequences, as for other Cells … done this way mostly to illustrate its use
- **Three different part types at Cell 1, so use a vector-valued Expression with three rows**
  - Name for the expression, **`Cell 1 Times`**
  - Rows, **`3`**
  - Expression Values subdialog
    - Cell 1 processing times for the three part types
    - Order matters, since index is part type … will reference as **`Cell 1 Times(Part Index)`** in model

Variable Data Module

- **Basic Process panel**
- **`Factor` variable**
  - Speed factor at Cell 3 – need a two-row vector
    - Assume new (faster) machine is #1, old (slower) machine is #2
    - Set to `0.8` for index 1; set to `1.0` for index 2
- **`Transfer Time` variable**
  - Holds transfer-time constant of 2 minutes between stations
  - Just a scalar, not a vector or matrix
  - Used for model generality – if all transfer times changed, this makes it easy to implement this change
- **These are the Initial Values of variables … any entity can change them**
  - But they're constant in this model

Set Data Module

- **Basic Process panel**
- **Define three sets**
  - Resource set, **`Cell 3 Machines`**
    - For new and old machine (in that order) at Cell 3
    - Resource Names – could have already defined them in Resource data module, or can define them here
  - Entity Picture set, **`Part Pictures`**
    - To attach to entities once their part type is determined
    - Picture Names – could have already defined them elsewhere (*Edit > Entity Pictures*), or can define them here
  - Entity Type set, **`Entity Types`**
    - To attach to entities once their part type is determined
    - Entity Types – define them here

Advanced Set Data Module

- **On Advanced Process panel**
- **Needed since Set data module does not have “Other” category for Type**
  - Need to form a set of Sequences to attach the right one to arriving entities once their part type is determined
  - Define Name of set to be **`Part Sequences`**
  - Set Type is “Other”
  - Members subdialog – Add rows, type in names in “Other” column (have to remember or look up the Sequence names)

Run > Setup and *Edit > Entity Pictures*

- ***Run > Setup* Dialog**
  - Replication Parameters Tab
    - Replication Length = 32 Hours
    - 24 Hours/Day
    - Base Time Units = Minutes
- ***Edit > Entity Pictures***
  - Create three custom pictures – **`Picture.Part 1`**, **`Picture.Part 2`**, **`Picture.Part 3`**
  - Copy blue, red, and green ball pictures
  - Rename them
  - Picture Editor to put white numbers inside via Text object

Part Arrivals

- **Create module for arrival of one part**
  - One-at-a-time, Time Between Arrivals is exponential with mean 13 minutes
  - Don’t know the part type yet …
- **Assign module for part attributes**
  - `Part Index` = draw from DISC probability distribution
    - Pairs *cumulative* probability, value
  - `Entity.Sequence` = `Part Sequences(Part Index)`
    - `Part Index` attribute already assigned … order matters
    - Index into `Part Sequences` (Advanced) Set
  - `Entity.Type` = `Entity Types(Part Index)`
  - `Entity.Picture` = `Part Picture(Part Index)`

Release Arriving Entity into System

- **Use previously defined Sequences, assigned to entity via (Advanced) Set of Sequences**
- **Send arriving entity through a Station module to define its current station location**
  - Station Name = **`Order Release`**
  - Other five station names already defined via Sequences
- **Route module to start it on its way**
  - Route Time = **`Transfer Time`** (a Variable previously defined) Minutes
  - Destination Type = Sequential
    - Arena will direct this entity according to its own sequence
    - It just arrived so Arena initializes its JobStep attribute

Logic for Cell 1

- **Station module to define the station location**
  - Station Name = `Cell 1`, on pull-down list for stations since it was previously defined in Sequences
- **Cell 1 Process module**
  - Action = Seize Delay Release
  - Resources subdialog
    - Type = Resource (not Set … yet)
    - Resource Name = `Cell 1 Machine,` Quantity to seize = 1
  - Delay Type = Expression
    - Expression = `Cell 1 Times(Part Index)` Minutes, using the previously-defined Expression `Cell 1 Times`
- **Route module from Cell 1**
  - Destination Type = Sequential
  - Station already defined (on incoming side)

Logic for Cells 2 and 4

- **Incoming Station module – similar to Cell 1**
  - Except for names of Module and Station
- **Process module**
  - Action, Resources, Delay Type – similar to Cell 1
  - Expression for Delay time = **`Process Time`**
    - Attribute defined in Sequence module for each job type at this point in its sequence for Cells 2 and 4
    - Note that Part Type 2 visits Cell 2 twice in its sequence, with *different* delay-time distributions … this data structure is general enough to handle this
- **Outgoing Route module – similar to Cell 1**
  - Except for name of Module

Logic for Cell 3

- **Station, Route modules – similar to Cells 1, 2, 4**
- **Process module**
  - Action, Delay Type – similar to Cells 1, 2, 4
  - Resources subdialog
    - Type = Set, Set Name = **`Cell 3 Machines`**
    - Selection Rule for set = Cyclical

      Maybe Preferred Order would have been better???
    - Save Attribute = **`Machine Index`** (will be 1 or 2)
  - Expression for Delay time =

    `Process Time * Factor(Machine Index)`

    to multiply by 0.8 if entity gets the new machine (#1), using the preciously-defined vector variable **`Factor`**
  - See book for alternative (cute) expression that avoids the need for the vector variable **`Factor`**

Digression: Data Structures

- **Why an Expression for processing times at Cell 1 rather than entity Attribute assigned in Sequences as for the other cells?**
  - Frank answer: Just to show the use of Expression
  - Could easily have treated Cell 1 like the others
- **Conversely, could have used Expression for processing times at Cells 3 and 4**
  - But there would be a problem with Cell 2
    - Part 2 visits it twice with different processing-time distributions, so would have to indicate which visit somehow
  - Moreover, this is a very small model
- **Moral: Think carefully about data structure!**

Logic for Exiting the System

- **Station module to define this location**
  - Station Name = **`Exit System`**
- **Dispose module**
  - Record Entity Statistics box is checked
  - Will generate one of the outputs we want, cycle time (time in system) separated out by part type, since they map onto the entity types for this model
  - So don’t need separate Record modules here to collect cycle times
- **Model would run at this point, give correct output results … but develop animation to show queues, resources, and movement …**

Animation

- **Pull animation away from logic, data modules**
- **Move, resize, reorient queues for realism**
- **Animate Routes (all movement possibilities)**
  - Thick “bundles” of routes — Shift key, Snap to Grid
  - Heed clockwise direction
  - Draw lines to define route “lanes”
- **Import, modify AutoCAD .dxf file for backdrop and resource pictures (see text)**
- **Fine-tune resource pictures**
  - Layers for seize point
- **In animation, note that entities travel at very different rates, pass each other … realistic???**

Verification

- **System → Model → "Code"**
- ***Validation*: Is Model = System?**
- ***Verification*: Is "Code" = Model? (debugging)**
- **The Truth: Can probably never completely verify, especially for large models**

Verification (cont'd.)

- **Some techniques to *attempt* verification**
  - Eliminate error messages (obviously)
  - Single entity release, Step through logic
    - Set Max Batches = 1 in Arrive
    - Replace part-type distribution with a constant
  - "Stress" model under extreme conditions
  - Performance estimation — like slide-rule decimal placement
  - Look at generated SIMAN .mod and .exp files
    - *Run > SIMAN > View*

Statistical Analysis of Output from Steady-State Simulations

- **Recall: Difference between terminating, steady-state simulations**
  - Which is appropriate depends on goal of study, and not so much on the model structure
    - Most models could be used for terminating or steady-state analysis
- **Now, assume steady-state is desired**
  - Be sure this is so, since running and analysis is a lot harder than for terminating simulations
- **Naturally, simulation run lengths can be long**
  - Opportunity for different internal computation order
  - Can change numerical results
  - Underscores need for statistical analysis of output

Warm Up and Run Length

- **Most models start *empty and idle***
  - *Empty*: No entities are present at time 0
  - *Idle*: All resources are idle at time 0
  - In a terminating simulation this is OK if realistic
  - In a steady-state simulation, though, this can bias the output for a while after startup
    - Bias can go either way
    - Usually downward (results are biased low) in queueing-type models that eventually get congested
    - Depending on model, parameters, and run length, the bias can be very severe

Warm Up and Run Length (cont'd.)

- **Remedies for initialization bias**
  - Better starting state, more typical of steady state
    - Throw some entities around the model
    - Can be inconvenient to do this in the model
    - How do you know how many to throw and where?

      This is what you're trying to estimate in the first place!
  - Make the run so long that bias is overwhelmed
    - Might work if initial bias is weak or dissipates quickly
  - Let model *warm up*, still starting empty and idle
    - *Run > Setup > Replication Parameters*: Warm-up Period

      Time units!
    - "Clears" all statistics at that point for summary report, any Outputs-type saved data from Statistic module of results across replications

Warm Up and Run Length (cont'd.)

- **Warm-up and run length times?**
  - Most practical idea: preliminary runs, plots
  - Simply "eyeball" them
  - Be careful about variability — make multiple replications, superimpose plots
  - Also, be careful to note "explosions"
- **Possibility – different Warm-up Periods for different output processes**
  - To be conservative, take the max
  - Must specify a single Warm-up Period for the whole model

Warm Up and Run Length (cont'd.)

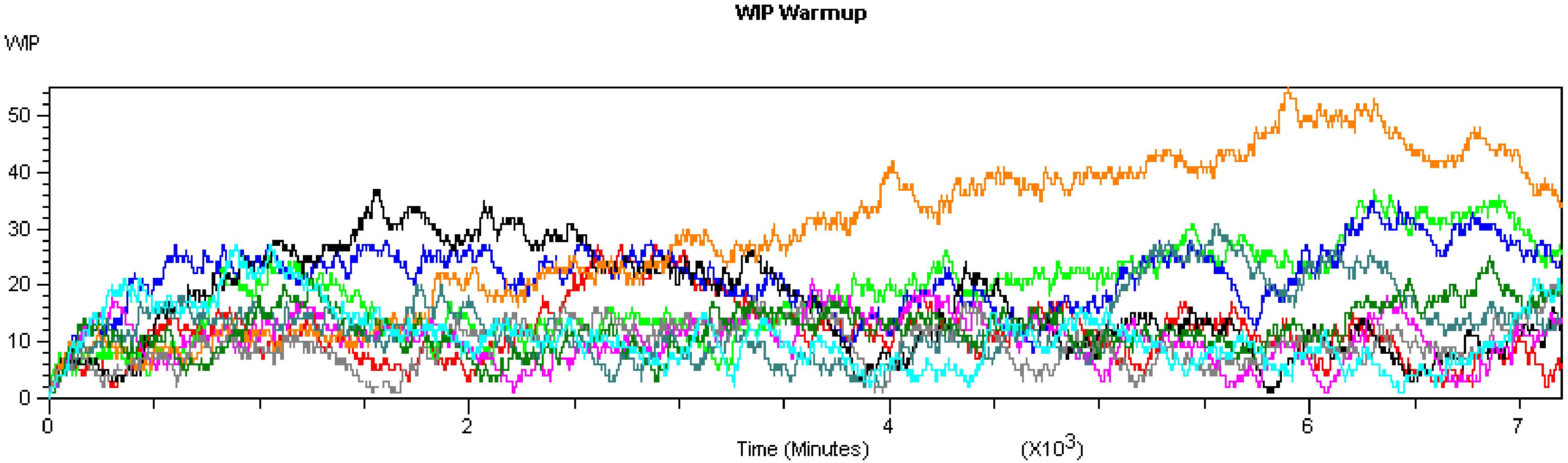
- **Create a single overall output performance measure for Model 7-1 … modify it into Model 7-2**
  - Measure is time-average total number of parts in system
  - Statistic module
    - Time-Persistent type, Name and Report Label **`Total WIP`**
    - Expression (via Expression Builder … details in book)

      `EntitiesWIP(Part 1) + EntitiesWIP(Part 2) + EntitiesWIP(Part 3)`
    - Output File **`Total WIP History.dat`** to save within-run data

      Animated plots disappear, can't overlay plots from multiple replications … will use Output Analyzer to plot the saved data
  - Speed up the run
    - Check *Run > Run Control > Batch Run (No Animation)*
    - Uncheck boxes in *Run > Setup > Project Parameters*, Dispose module
  - Lengthen Replications to 5 days, do 10 Replications

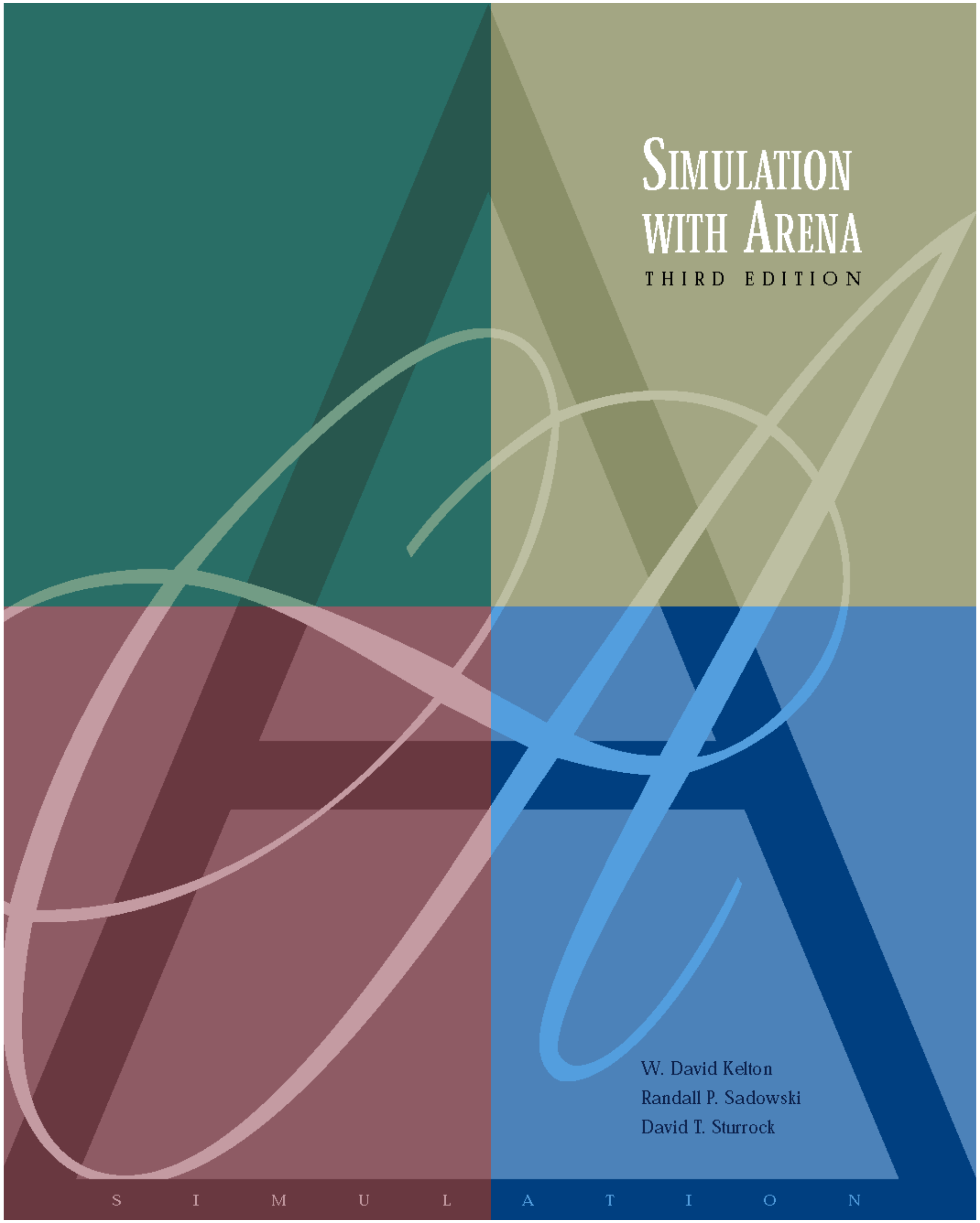
Warm Up and Run Length (cont'd.)

- **In Output Analyzer**
  - New data group, Add the file **`Total WIP History.dat`**
  - *Graph > Plot* or 📈
  - Add **`Total WIP History.dat`**, Replications = All, enter Title, axis labels



  - No apparent explosion
  - Warm-up about 2000 min.; round up to 2 days (2880 min.)

Solve examples   7-4, 7-8,



Entity Transfer

Chapter 8

Last revision June 9, 2003

What is important here is the steps for the solution of the example in chapter 7 while adding transporters and conveyor, how to define them distances, segments stations etc.

What We'll Do ...

- **Types of Entity Transfers**
- **Model 8-1: Resource-Constrained Transfers**
- **Models 8-2, 8-3: Transporters**
- **Conveyors**
  - Model 8-4: Non-accumulating
  - Model 8-5: Accumulating

Types of Entity Transfers So Far

- **Connect**
  - Zero-delay
  - Connection graphic vs. module Labels (no graphic)
- **Route**
  - Non-zero-delay — constant, r.v., expression
  - Stations, animated Routes
  - Fixed routes vs. entity-dependent Sequences
- **Connect and Route both assume:**
  - No limit on number in transit at a time
  - Entities have their own feet

New Types of Entity Transfers

- ***Resource-constrained* transfers**
  - Limit total number of entities in transit at a time
  - Entities still have their own feet
  - Telecommunications (number of packets), logistics (number of vehicles)
- **Material-handling devices**
  - *Transporters* – fork lifts, trucks, carts, wheelchairs
    - Usually place limits on numbers, capabilities of transporters
    - Like a Resource, except moveable
  - *Conveyors*
    - Belts, hook lines, escalators
    - Usually limit space on conveyor, speed
    - Non-accumulating vs. accumulating

Model 8-1: Small Manufacturing System with Resource-Constrained Transfers

- **Original system (Model 7-1)**
  - Assumed all transfer times = 2 minutes … keep (for now)
  - Parts have their own feet … keep (for now)
  - No limit on number of parts in transit at a time … dump
    - Now – no more than 2 parts can be in motion at a given time
    - If other parts are ready to go, they must wait until there's room to go
- **Model via existing constructs — think creatively**
  - Model "space" on the "road" as a Resource
  - Limit the number of Units of this Resource
  - Entity must Seize unit of "space" resource before beginning trip, Release it at end of trip

Two Ways to Model Resource-Constrained Transfers

- **Both use a new `Transfer` Resource representing space on the transitways**
  - Capacity set to **2** in Resource data module
- **Maybe the most obvious way (but won't do) …**
  - Before each Route module insert a Seize module to Seize one unit of **`Transfer`** (queue, priority details … see text)
  - After each Station module (except **`Order Release`**) insert a Release module to free up one unit of **`Transfer`**

Two Ways to Model Resource-Constrained Transfers (cont'd.)

- **Different way (will do, to illustrate new modules, set up for transporters and conveyors) …**
  - Replace Route modules with *Leave* modules (Advanced Transfer panel)
    - Transfer Out: Seize unit of **`Transfer`** resource before leaving station
      - Resource, Resource Set, particular member of a Resource Set
      - Can specify priorities
    - Also contains the Route operation
    - Get individual queues, with animation, for parts waiting to go
  - Replace Station modules with *Enter* modules (Advanced Transfer panel)
    - Defines the Station
    - Option of an unload Delay time (`0` for this model)
    - Transfer In: Release **`Transfer`** resource
- **Effect – slight increase in cycle times in system**

Transporter Concepts

- **Carts, fork lifts, trucks, wheelchairs, people, …**
- **When entity is ready to go somewhere, it needs to be "picked up" and moved**
- **Use *Transporters* — "moveable" resources**
- **Activities: *Request*, *Transport*, *Free***
  - Transporter Selection Rule: If > 1 transporter is available when Requesting
  - When freed and > 1 entity is waiting: Priorities, closest one
- **Two types of Transporters**
  - *Free-Path* (we'll do)
    - Travel time depends only on velocity, distance
    - Ignore "traffic jams" and their resulting delays
  - *Guided* (won't do)
    - AGVs, intersections, etc.

The Small Manufacturing System with Transporters

- **Have two carts to transport parts**
  - A cart can carry one part at a time
  - Carts move at 50 feet/minute
    - Will need to specify accurate distances between Stations
  - It takes 0.25 minute to load part on a cart, 0.25 minute to unload it from a cart
- **Modify Model 8-1 to Model 8-2**

The Small Manufacturing System with Transporters (cont'd.)

- **Create Transporter in Transporter data module (Advanced transfer panel)**
  - Name = `Cart`, Capacity = `2`, Velocity = `50`
  - Default the Distance Set (later), Units = Per Minute, Initial Positions
    - *Mind the units* – consistency here, in Distance Set (later)
- **Animation picture for `Cart` Transporter**
  - Transporter button , Animate Transfer toolbar
  - Identifier = `Cart`, pictures for Idle, Busy, Inactive states
  - Draw or copy from .plb picture libraries
  - Ride point (details in book)
  - Drop it anywhere in flowchart view (hidden during run)

The Small Manufacturing System with Transporters (cont'd.)

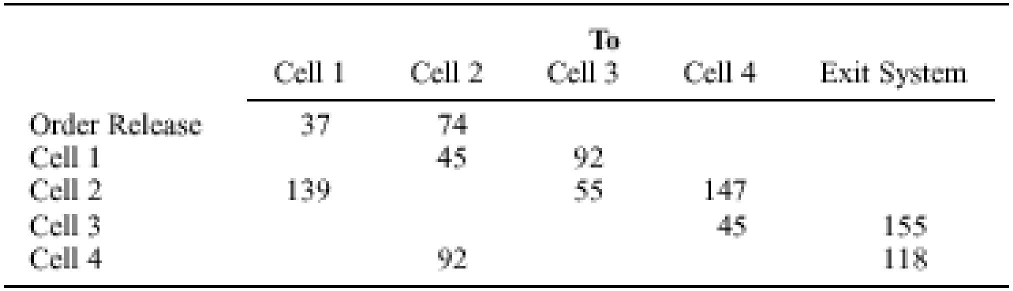
- **Request a `Cart` – modify existing Leave modules**
  - Delay = **`0.25 Minute`** for load time
  - Transfer Out = **`Request Transporter`**
  - Transporter Name = **`Cart`**
  - Selection Rule = **`Smallest Distance`**
    - Applies when > 1 transporter is available
    - Others: Cyclic, Random, Preferred Order, Largest Distance (???)
  - Save Attribute = **`Cart #`** (remember which cart … for later)
  - Connect Type = **`Transport`**
    - Move Time disappears … determined by Velocity, Distances (later)
  - Station Type = **`Sequence`**
- **Instead of Leave: Request-Delay-Transport**
  - More complex, more flexible – book has details, examples

The Small Manufacturing System with Transporters (cont'd.)

- **Free the Cart – modify existing Enter modules**
  - Delay = **`0.25 Minute`** for unload time
  - Transfer In = **`Free Transporter`**
  - Transporter Name = **`Cart`**
  - Unit Number = **`Cart #`** attribute of part entity
- **Instead of Enter: Station-Delay-Free**
  - More complex, more flexible – book has details, examples

Distances for Transporters

- **Define contents of Distance Set `Cart.Distance`**
- **Distances (in feet) moved by parts:**



			To		
	Cell 1	Cell 2	Cell 3	Cell 4	Exit System
Order Release	37	74			
Cell 1		45	92		
Cell 2	139		55	147	
Cell 3				45	155
Cell 4		92			118

Units!!

  - Blank cells: part movements that don't occur
- **Enter these data in Distance data module (Advanced Transfer panel)**
  - Name = **`Cart.Distance`**
  - Stations button, add 11 rows with Beginning Station, Ending Station, Distance for above data
  - Direction is implied; could be asymmetric

Units!!

Why are there 25 rows??

Animating Transporter Movement

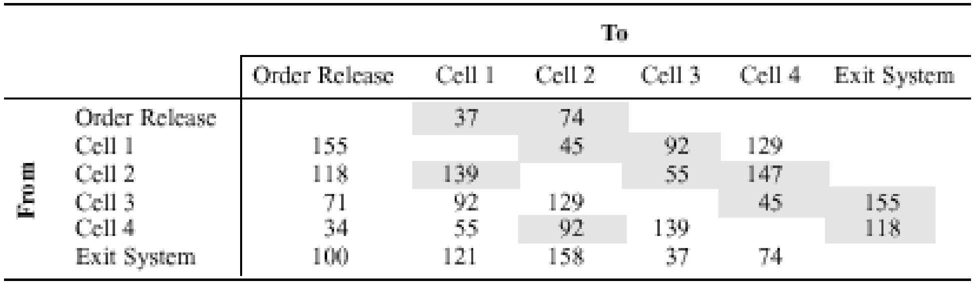
- **Add distances to animation**
- **Delete all the old Route Path animation objects**
  - But leave the Station animations
- **Add animated transporter distances with Distance button , Animate Transfer toolbar**
  - Dialog, placement similar to Route Paths
  - Identifier = **`Cart.Distance`**
  - Click in Beginning Station marker, intermediate clicks, Ending Station marker
  - Options for Rotate, Flip
  - Grid, Snap to help place animated transporter distances

Parking Areas for Transporters

- **Animate transporters when they're free**
- **Parking button , Animate Transfer toolbar**
  - Like a Queue animation – Point vs. Line, Shift, Rotate
  - Cursor becomes cross hairs, click near lower left of Station marker to start, click for first Point or head of Line
  - More clicks for more Points (double-click to end), or second click to end Line
  - Want enough points/space for all transporters (2 here)
  - Repeat for all Stations where Transporters could be freed

More Distances — Empty Transporters

- **Above Distances incomplete — only for part movements along their sequences**
- **Transporters must also move when empty (*deadheading*)**
  - In general, *n*(*n* - 1) distances need definition for network with *n* nodes
  - Some not possible — Order Release to Exit System
- **14 more distances to define in Distances data module (not grayed):**



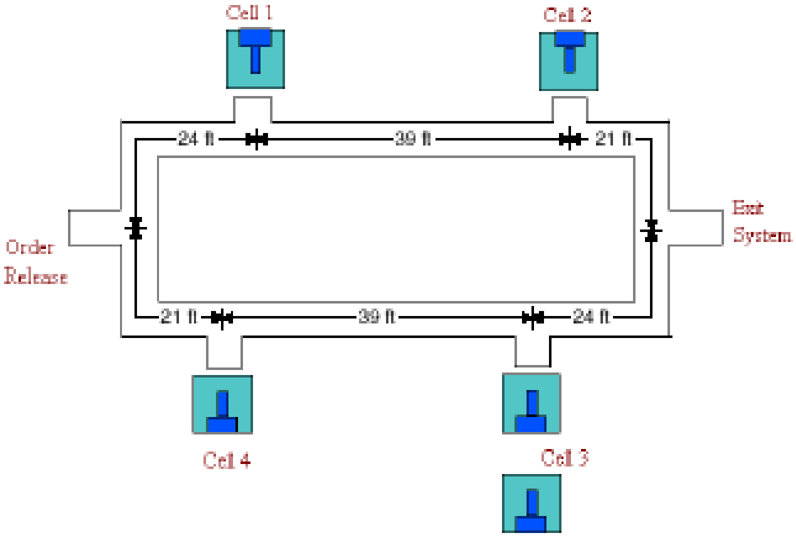
		To					
		Order Release	Cell 1	Cell 2	Cell 3	Cell 4	Exit System
From	Order Release		37	74			
	Cell 1	155		45	92	129	
	Cell 2	118	139		55	147	
	Cell 3	71	92	129		45	155
	Cell 4	34	55	92	139		118
	Exit System	100	121	158	37	74	

Model 8-3: Refining the Animation for Transporters

- **As it stands, `Part` Entities disappear from animation when waiting to be picked up by a `Cart` Transporter**
  - Model logic OK … get right answers … animation is flawed
- **Solution – *Storage* for entity to reside in, be animated, while it waits for something (here, a `Cart` Transporter)**
  - Can get statistics on numbers in Storages
- **But Storages not available with modules from Advanced Transfer panel**
  - Use lower-level SIMAN modules from Blocks panel … see book for specific details

Conveyors

- **Replace Transporters with a conveyor**
- **Loop conveyor to follow main path, clockwise**
- **Six entrance/exit points**
  - Load, Unload takes 0.25 minute
  - Each part is 4 feet per side, but want 6 feet of conveyor space for clearance on corners
- **Speed = 20 feet/minute** *Units!!*
- **Distances:**



Conveyor Concepts

- **Entity to be conveyed must wait for space**
- **Conveyor consists of *cells***
  - Equal size, constantly moving – think of a narrow escalator
- **Entities might require multiple contiguous cells**
- **Must define *cell size*; tradeoff involved:**
  - Small cells: accurate model but slow execution
  - Large cells: just the opposite!
- **Entities *Access* space, *Convey*, and *Exit***
- **Conveyor = series of linear *Segments***
  - Each segment starts and ends at a Station
  - Link to form loops, diverge points, converge points

Types of Conveyors

- **Both travel in a single, irreversible direction**
- ***Nonaccumulating*: belt, bucket line, escalator**
  - Spacing between entities on it doesn't change
  - Entire conveyor stops for entity Access/Exit if Load/Unload time is > 0
- ***Accumulating*: rollers, freeway**
  - Conveyor never stops moving
  - If entity on it stops to Exit, other entities behind it are blocked and bunch up (entities ahead of it keep moving)
  - When blockage ends, blocked entities go on but maybe not all at once (spacing requirements)

Model 8-4: Small Manufacturing System with Nonaccumulating Conveyors

- **Modify Model 8-1 (resource-constrained transfer)**
- **Define new Variables `Load Time` and `Unload Time`, each with initial value 0.25**
- **Delete all the Route Paths**
- **Define Conveyor via Conveyor data module, Advanced Transfer panel**
  - Conveyer = **`Loop Conveyor`**
  - Segment Name = **`Loop Conveyor.Segment`**
  - Type = Non-Accumulating
  - Velocity = **`20`** (feet), Units = Per Minute *Units!!*
  - Cell Size = **`3`** (feet) *Units!!*
  - Max Cells Occupied = **`2`** (cells per entity)

Leave, Enter Modules for Conveyor

- **Change each Leave module**
  - Delay = **`Load Time`**, Units = Minutes
  - Transfer Out = Access Conveyor
  - Conveyor Name = **`Loop Conveyor`**
  - # of Cells = **`2`**
  - Connect Type = Convey
- **Change each Enter module**
  - Delay = **`Unload Time`**, Units = Minutes
  - Transfer In = Exit Conveyor
  - Conveyor Name = **`Loop Conveyor`**

Conveyor Segments

- **Define one-way lengths (in feet) of segments**
- **Segment data module, Advanced Transfer panel**
  - Name = `Loop Conveyor.Segment`
  - Beginning Station = `Order Release`
  - Next Stations button
    - Name Next Station in correct sequence
    - Give distance (in feet) to this next station
- **Segment animation**
  - Put Station markers in front of each Resource picture
  - Segment button , Animate Transfer toolbar
  - Dialog, crosshairs, clicking just like Distances for Transporters
    - Except here, have to place only 6 Segment animations

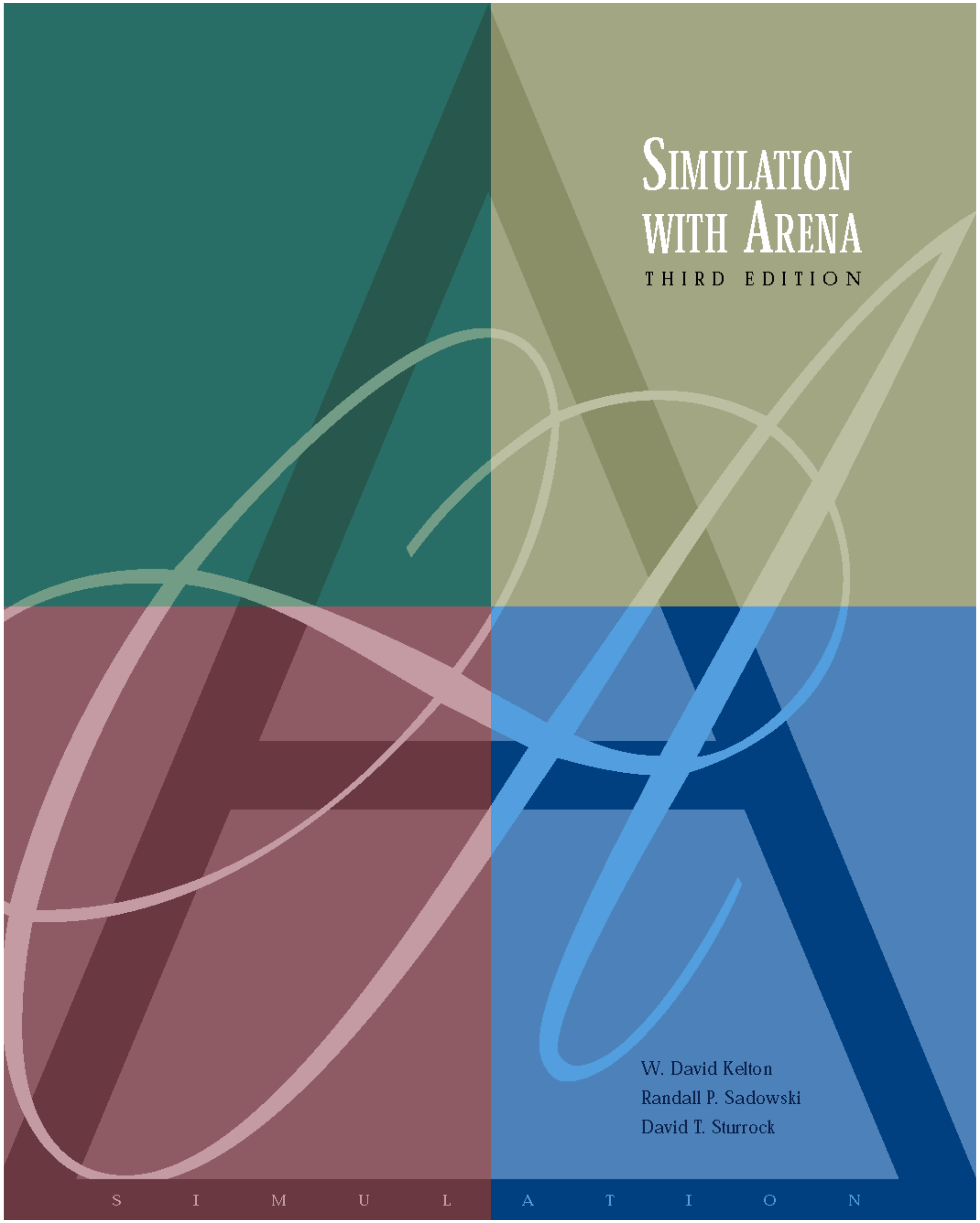
Conveyor Statistics

- ***Run > Setup > Project Parameters* to check Conveyor Statistics**
- **Get percent of time blocked (stopped)**
- **Utilization statistic is average percent of space occupied on conveyor (not percent of time that a part was on the conveyor)**
- **To see conveyor stop (it's nonaccumulating) more clearly, change `Load Time` and `Unload Time` to much greater values than `0.25`**
  - Could do this during run with VBA (Chapter 10), or Run Controller – see text for details – but makes output statistics nearly impossible to interpret

Model 8-5: Change Conveyors to Accumulating

- **Conveyor module**
  - Change Conveyor Type to Accumulating
  - Accumulation Length = 4 (in feet), amount of space the accumulated parts need on the conveyor
- **Running, see very little accumulation in animation**
  - To see more, increase **`Load Time`** and **`Unload Time`**

Solve 8-7 question



Arena Integration and Customization

Chapter 10

Last revision July 14, 2003

What We'll Do ...

- **Reading and Writing Data Files (ReadWrite)**
- **ActiveX™ and Visual Basic® for Applications (VBA)**
- **Creating Modules with Arena Professional Edition**

Reading and Writing Data Files

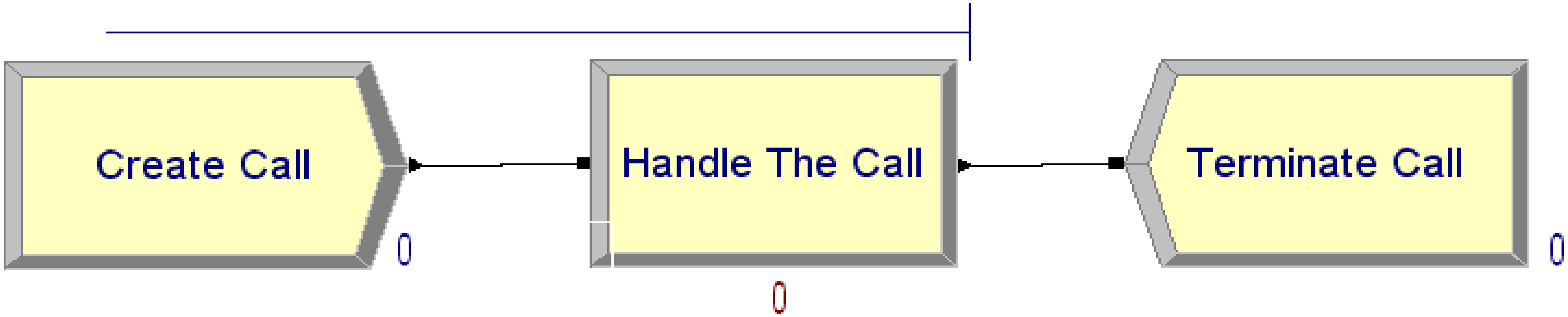
- Reading entity arrivals from a text file
- Reading and writing Microsoft Access and Excel files
- Advanced reading and writing

Reading Entity Arrivals From a Text File

- **Why data-driven simulations?**
  - Model validation
  - Evaluating how a particular scenario is handled
  - Modeling a specific arrival pattern
  - Assumes historical data exist and can be transformed for use in simulation

Simple Call Center Model

- Single call stream
- Single agent resource
- Random call processing time



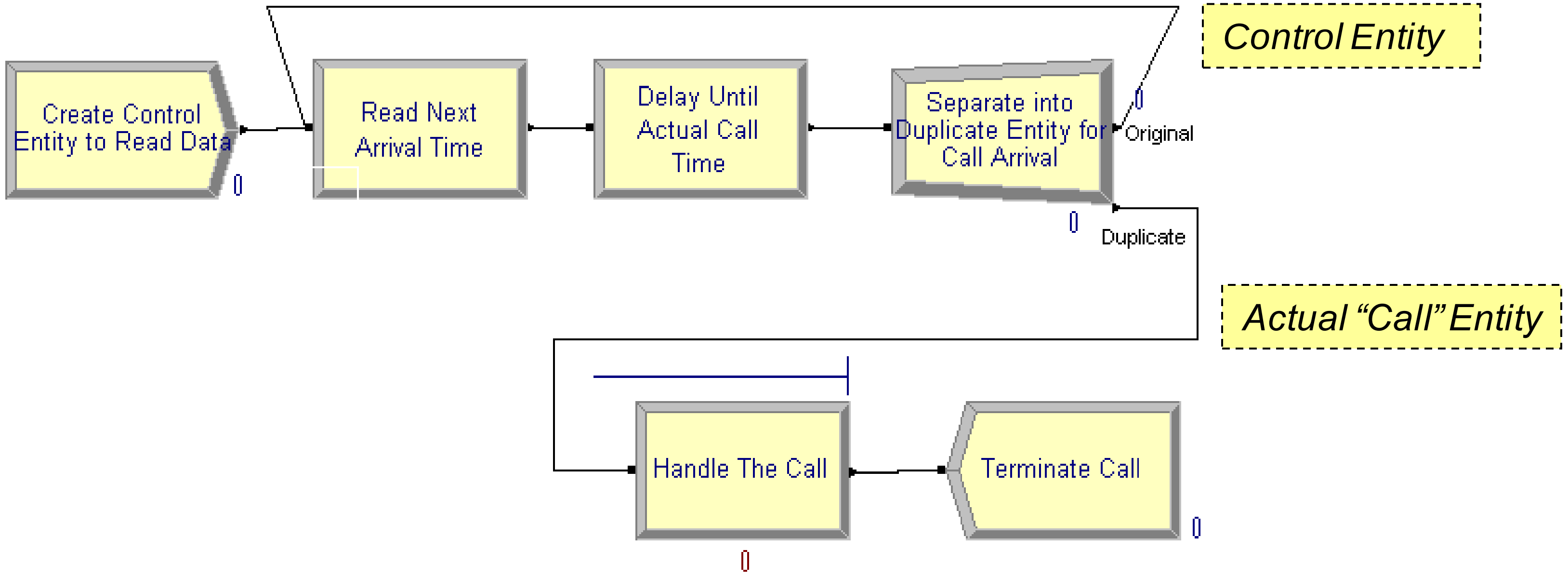
External Call Center Data

- **Historical call arrival times**
- **Model 10-02 Input.txt**
  - ASCII file (e.g., Notepad, saved as text from Excel)
  - Absolute simulation arrival times

```
1.038457
2.374120
4.749443
9.899661
10.525897
17.098860
```

Model Logic to Read Data

- **Can’t use simple time between arrivals**
- **Control entity**
  - Create only one
  - Duplicate to send actual “call” entity into model


Create Control Entity to Read Data
0
Read Next Arrival Time
Delay Until Actual Call Time
Separate into Duplicate Entity for Call Arrival
0
Original
0
Duplicate
Handle The Call
0
Terminate Call
0

Control Entity

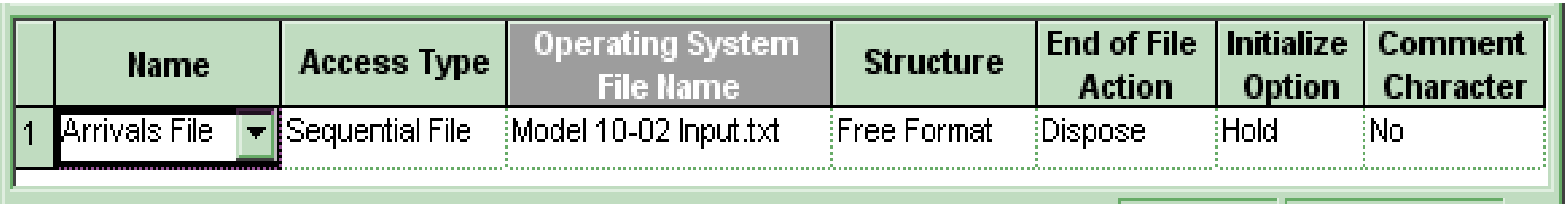
Actual “Call” Entity

Model Logic to Read Data (cont’d.)

- **ReadWrite module (Advanced Process)**
  - *Arena File Name*: description (actual disk filename is specified in File module)
  - *Assignments*: model variables/attributes to be assigned based on data read from file (*Call Start Time* attribute)
- **Delay/Duplicate Logic**
  - File contains “absolute” times; Delay module holds entity for a time *interval*
  - Delay control entity for interval until actual arrival time of call (*Call Start Time - TNOW*)
  - Create a duplicate (Separate module) to dispatch actual call into model. Original entity loops back to read next time.

Model Logic to Read Data (cont'd.)

- **File data module (Advanced Process)**
  - *Name*: Name referenced in other Arena modules.
  - *Access Type:* Sequential indicates to read in order.
  - *Operating System File Name*: The name used by file system. May be relative or fully qualified.
  - *End of File Action*: What to do when all records are read.



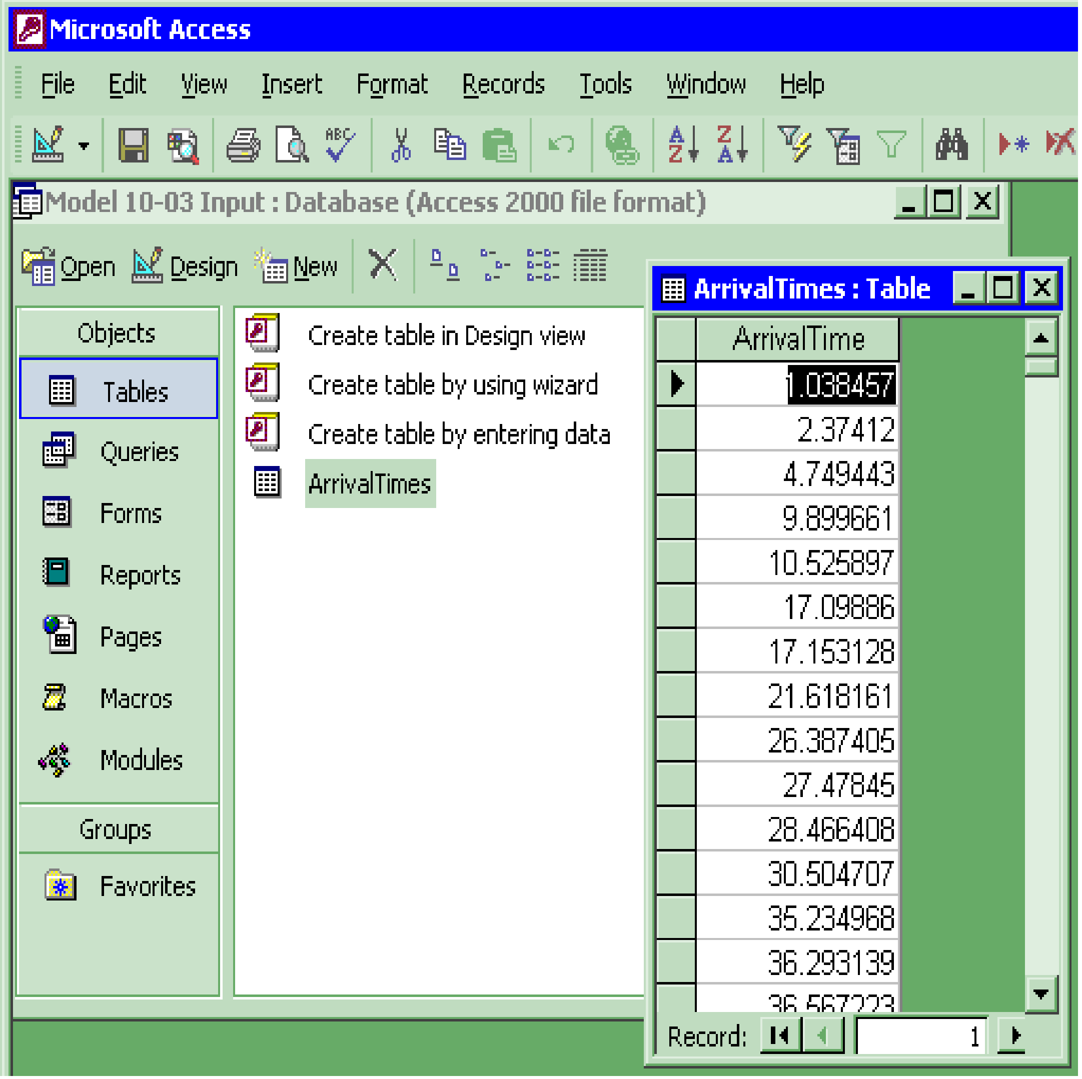
	Name	Access Type	Operating System File Name	Structure	End of File Action	Initialize Option	Comment Character
1	Arrivals File	Sequential File	Model 10-02 Input.txt	Free Format	Dispose	Hold	No

Run Termination

- **Run Setup options**
  - Maximum replications / simulation end time always terminates the simulation run.
- **System empties**
  - If no entities on calendar and no other time-based controls, run may terminate earlier than setup options dictate.
  - The control entity is disposed after it reads the last data value.

Reading Access Files

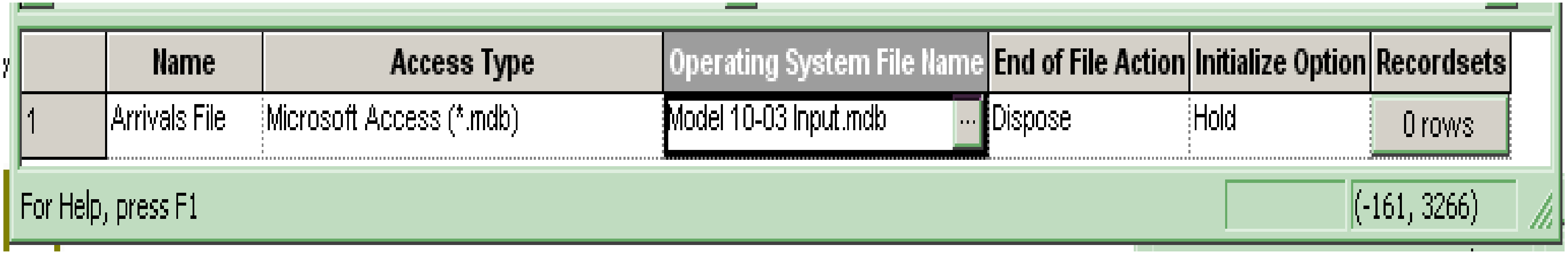
- **Sample Access data**
  - Model 10-03 Input.mdb
  - Table: ArrivalTimes



ArrivalTime
1.038457
2.37412
4.749443
9.899661
10.525897
17.09886
17.153128
21.618161
26.387405
27.47845
28.466408
30.504707
35.234968
36.293139
36.567223

Reading Access Files

- **File data module (Advanced Process)**
  - *Access Type*: Microsoft Access (*.mdb)
  - *Operating System File Name*: Model 10-03 Input.mdb
  - *Recordsets*: Click to load the Recordsets Editor
- **Important:**
  - Never name an access file the same as the model name or it will conflict with the automatic output database file.



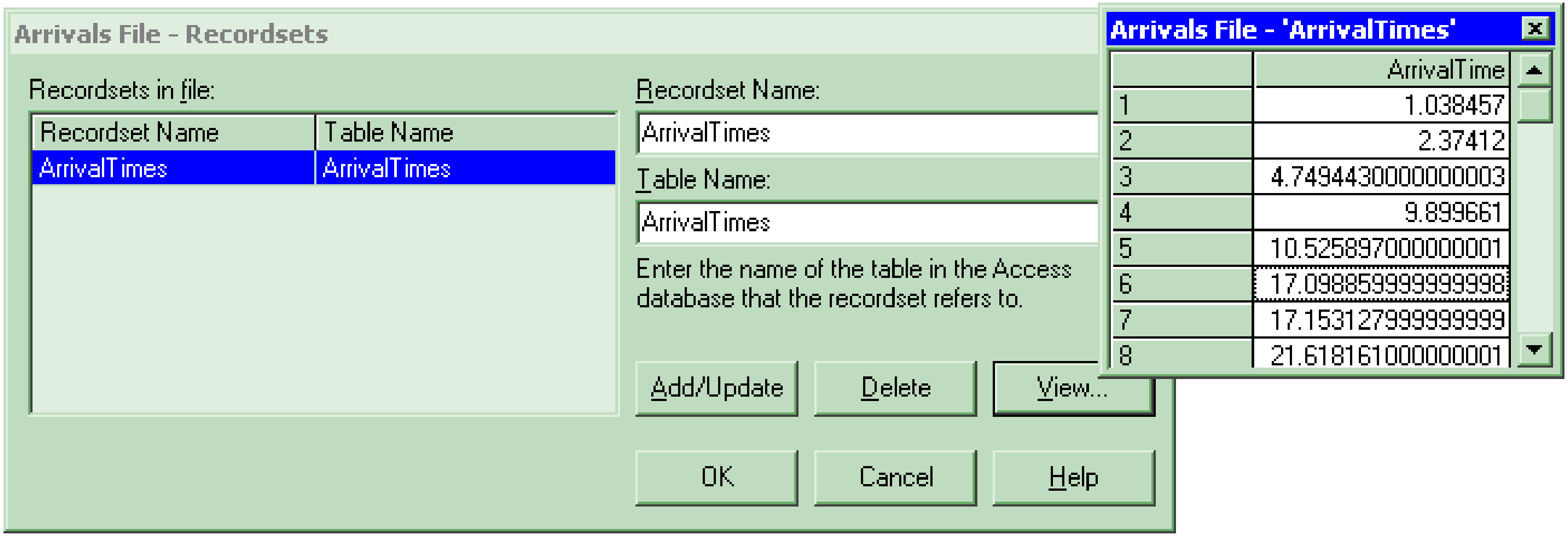
	Name	Access Type	Operating System File Name	End of File Action	Initialize Option	Recordsets
1	Arrivals File	Microsoft Access (*.mdb)	Model 10-03 Input.mdb	Dispose	Hold	0 rows

For Help, press F1

(-161, 3266)

Reading Access Files

- **Recordsets Editor**
  - Associates a recordset name with a table
  - Table must already exist
  - *View* allows you to see a sample of the real data



Arrivals File - Recordsets

Recordsets in file:

Recordset Name	Table Name
ArrivalTimes	ArrivalTimes

Recordset Name: ArrivalTimes

Table Name: ArrivalTimes

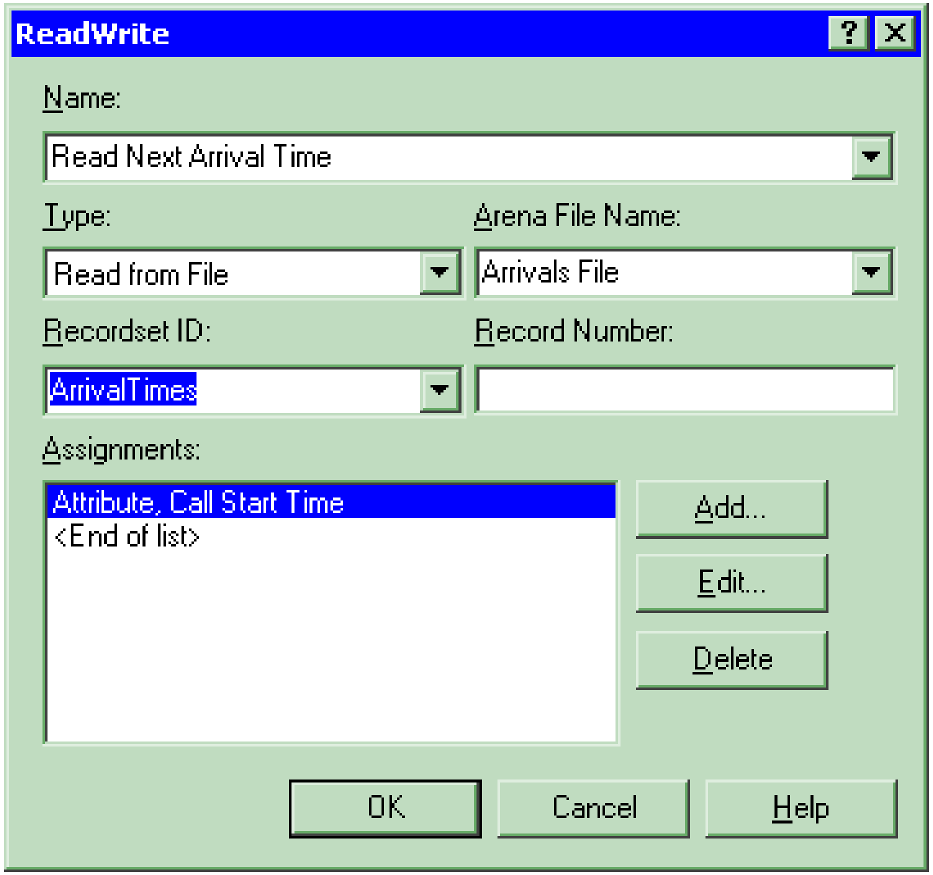
Enter the name of the table in the Access database that the recordset refers to.

Arrivals File - 'ArrivalTimes'

	ArrivalTime
1	1.038457
2	2.37412
3	4.7494430000000003
4	9.899661
5	10.525897000000001
6	17.098859999999998
7	17.153127999999999
8	21.618161000000001

Reading Access Files

- **ReadWrite module (Advanced Process)**
  - *Recordset ID*: Same as defined in Recordsets Editor



ReadWrite

Name: Read Next Arrival Time

Type: Read from File

Arena File Name: Arrivals File

Recordset ID: ArrivalTimes

Record Number:

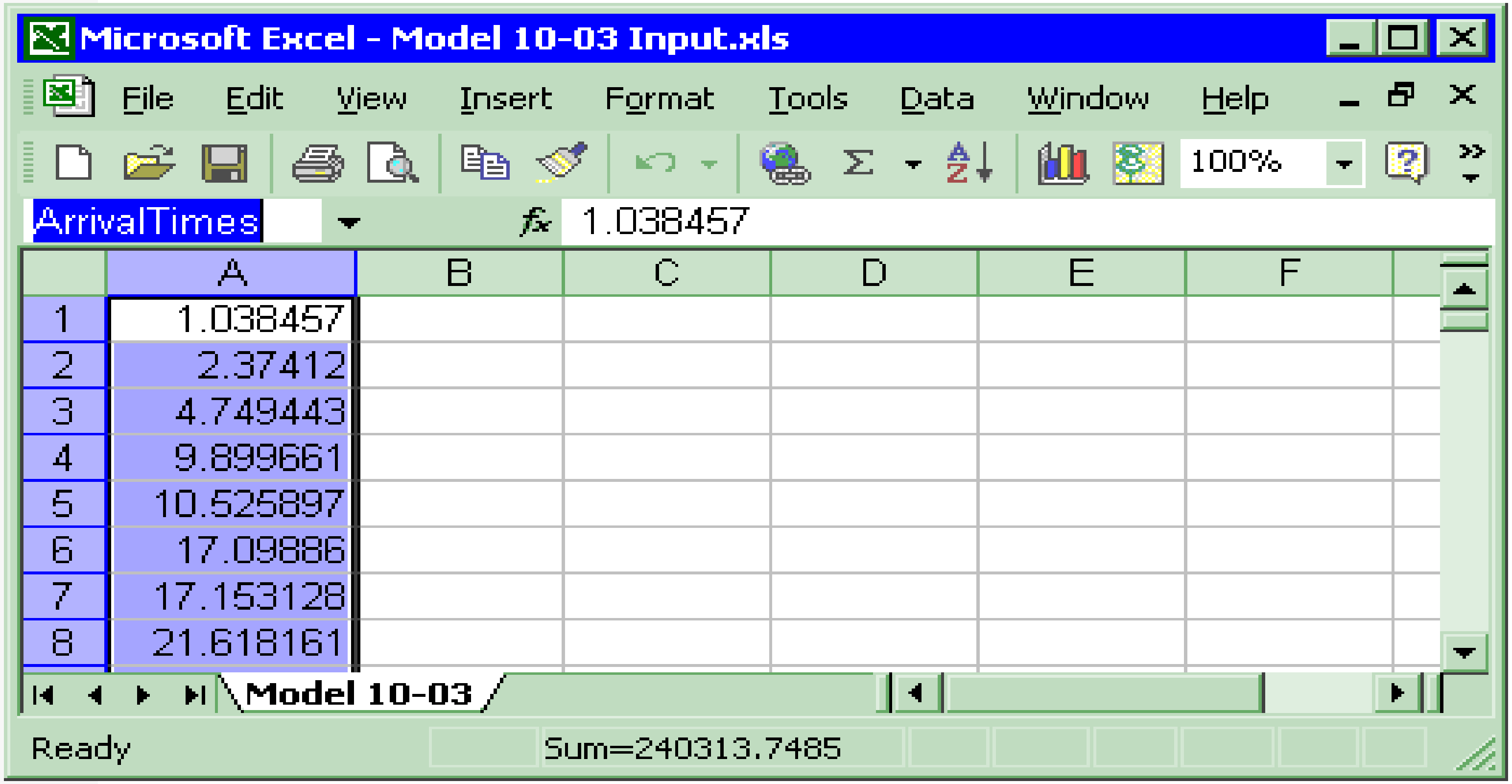
Assignments:
Attribute, Call Start Time
<End of list>

Reading Excel Files

- **Excel is not a relational database but has many similarities:**
  - *An Excel workbook is similar to an Access database file.*
  - *The rows and columns in a rectangular named range in an Excel worksheet are similar to the rows and columns of an Access table.*

Reading Excel Files

- **Sample Excel data**
  - Model 10-03 Input.xls
  - Named Range: ArrivalTimes



Microsoft Excel - Model 10-03 Input.xls
File Edit View Insert Format Tools Data Window Help
100%

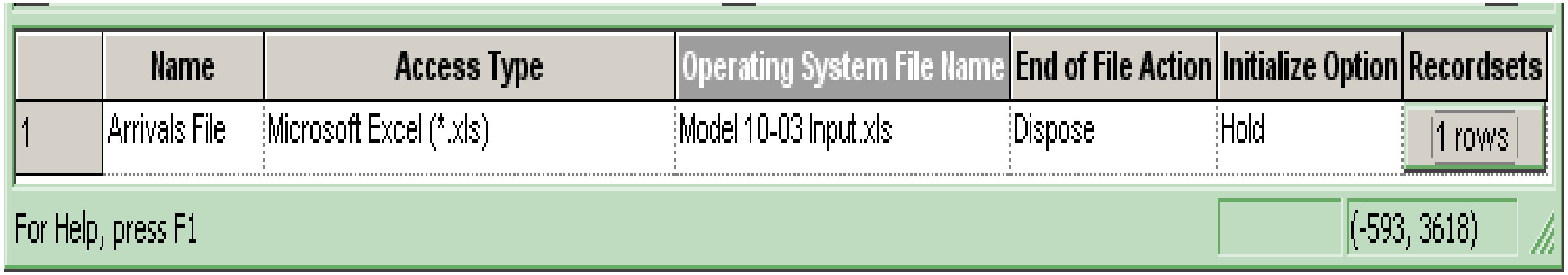
ArrivalTimes fx 1.038457

	A	B	C	D	E	F
1	1.038457					
2	2.37412					
3	4.749443					
4	9.899661					
5	10.525897					
6	17.09886					
7	17.153128					
8	21.618161					

Sum=240313.7485

Reading Excel Files

- **File data module (Advanced Process)**
  - *Access Type*: Microsoft Excel (*.xls)
  - *Operating System File Name*: Model 10-03 Input.xls
  - *Recordsets*: Click to load the Recordsets Editor



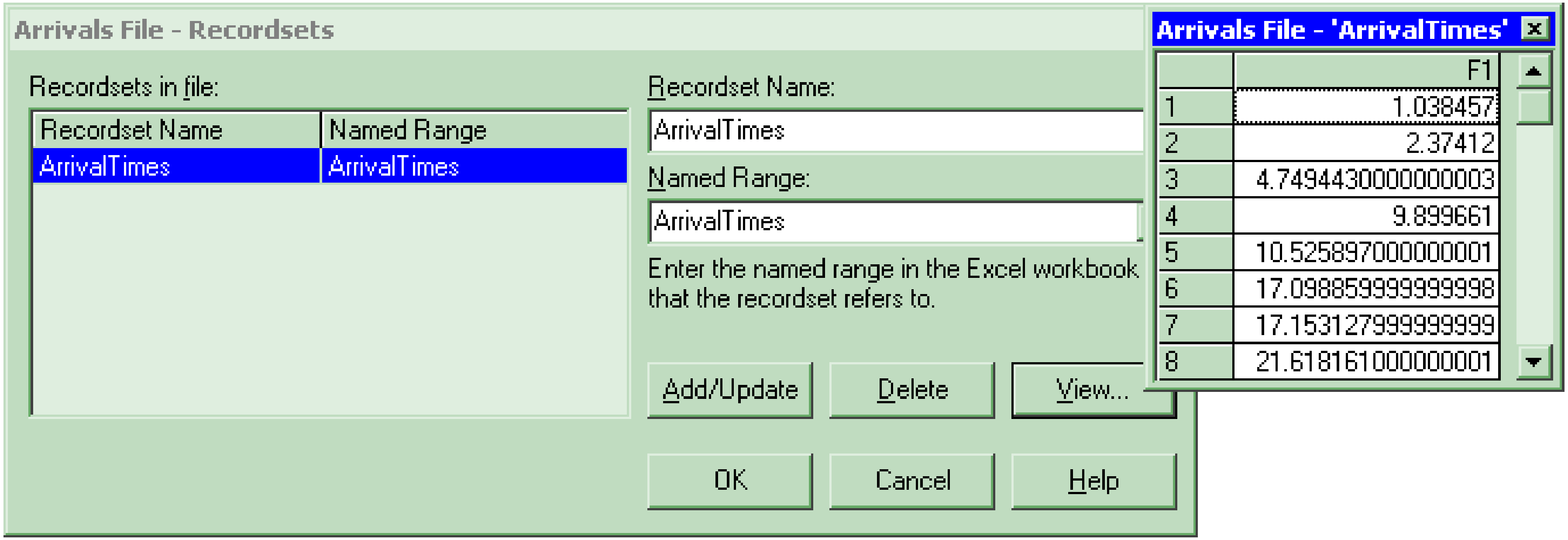
	Name	Access Type	Operating System File Name	End of File Action	Initialize Option	Recordsets
1	Arrivals File	Microsoft Excel (*.xls)	Model 10-03 Input.xls	Dispose	Hold	1 rows

For Help, press F1

(-593, 3618)

Reading Excel Files

- **Recordsets Editor**
  - Associates a recordset name with a named range
  - Named range must already exist
  - *View* allows you to see a sample of the real data



Arrivals File - Recordsets

Recordsets in file:

Recordset Name	Named Range
ArrivalTimes	ArrivalTimes

Recordset Name: ArrivalTimes

Named Range: ArrivalTimes

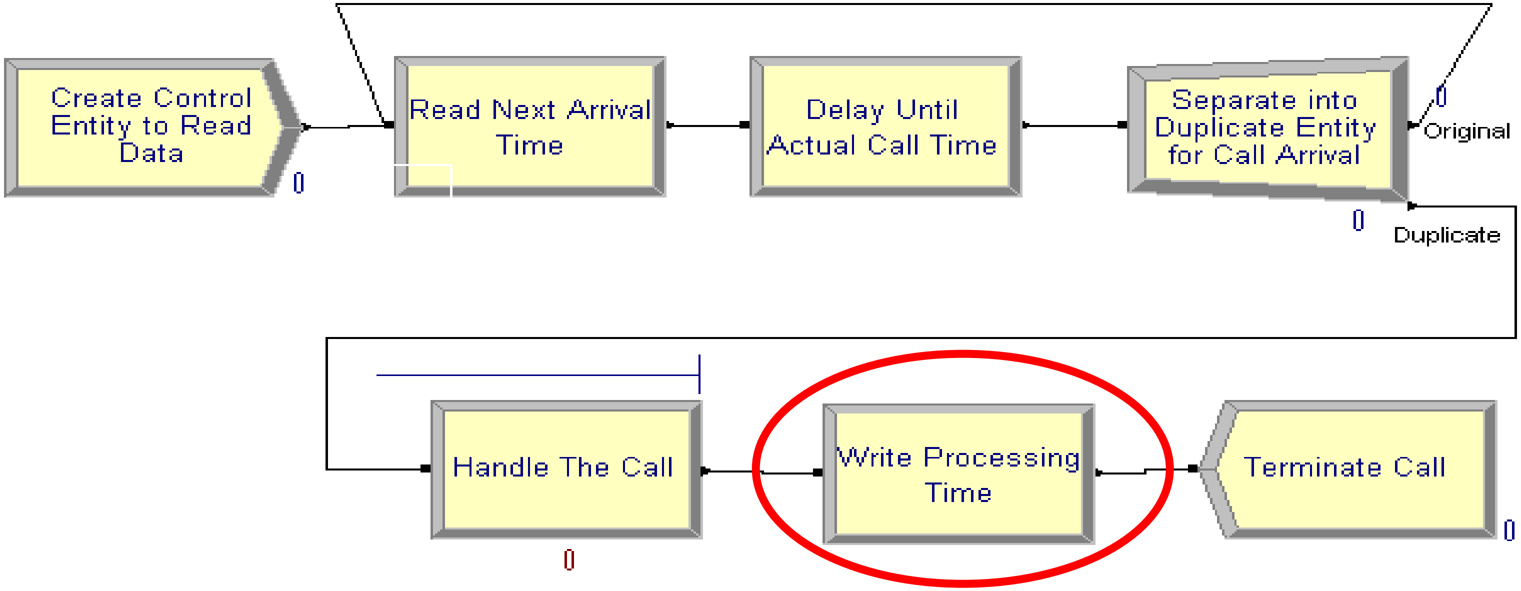
Enter the named range in the Excel workbook that the recordset refers to.

Arrivals File - 'ArrivalTimes'

	F1
1	1.038457
2	2.37412
3	4.7494430000000003
4	9.899661
5	10.525897000000001
6	17.098859999999998
7	17.153127999999999
8	21.618161000000001

Writing Access and Excel Files

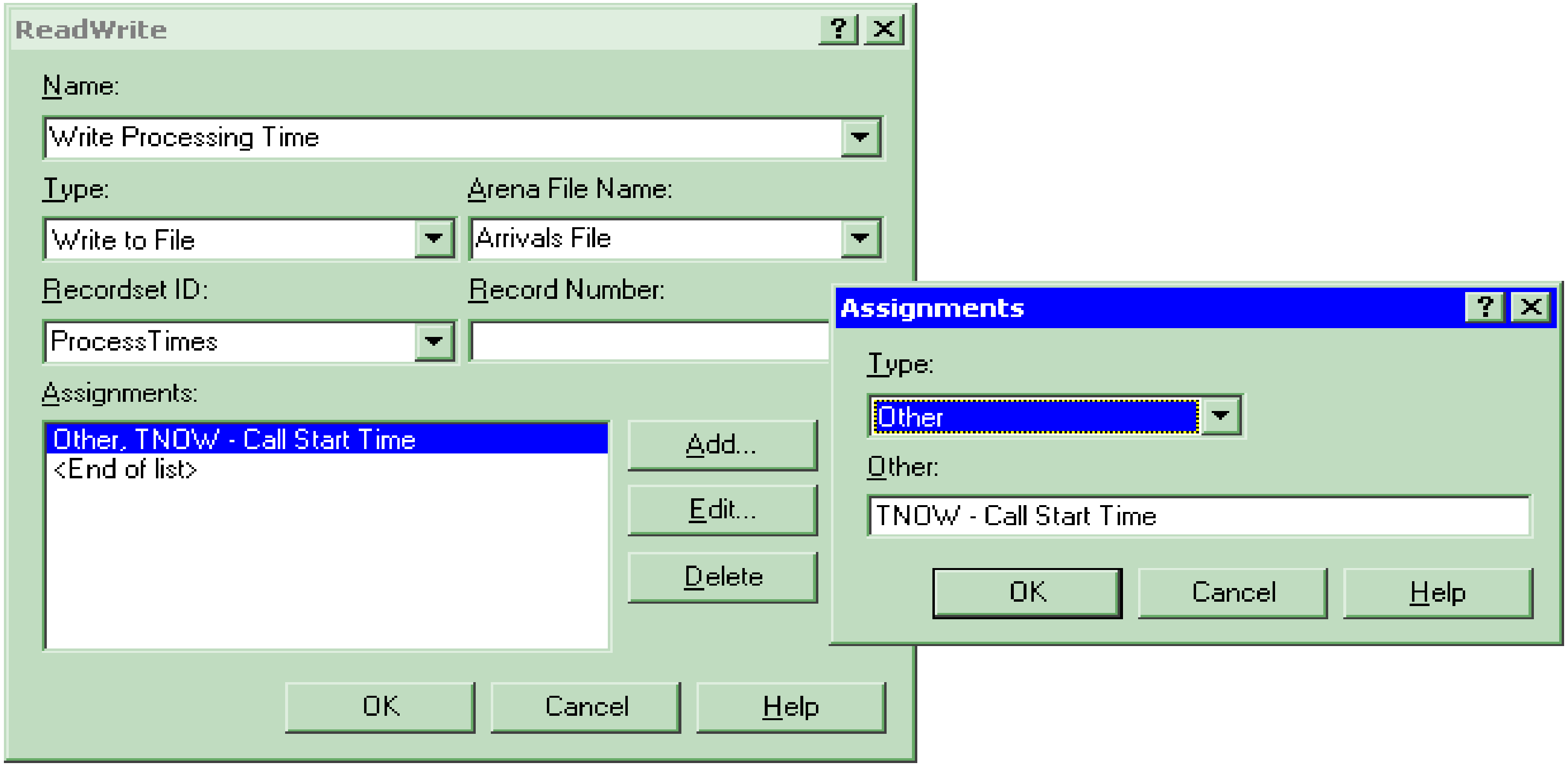
- ***The file:***
  - *The table or named range must already exist.*
  - *An Excel named range should be formatted as numeric.*
- **ReadWrite module:**
  - Add new module to specify which data to write.



Create Control Entity to Read Data
0
Read Next Arrival Time
Delay Until Actual Call Time
Separate into Duplicate Entity for Call Arrival
0
Original
0
Duplicate
Handle The Call
0
Write Processing Time
Terminate Call
0

Writing Access and Excel Files

- **ReadWrite module:**
  - Use *Type* of *Write To File*
  - *Use Recordset ID as before.*



ReadWrite

Name: Write Processing Time

Type: Write to File

Arena File Name: Arrivals File

Recordset ID: ProcessTimes

Record Number:

Assignments:
Other, TNOW - Call Start Time
<End of list>

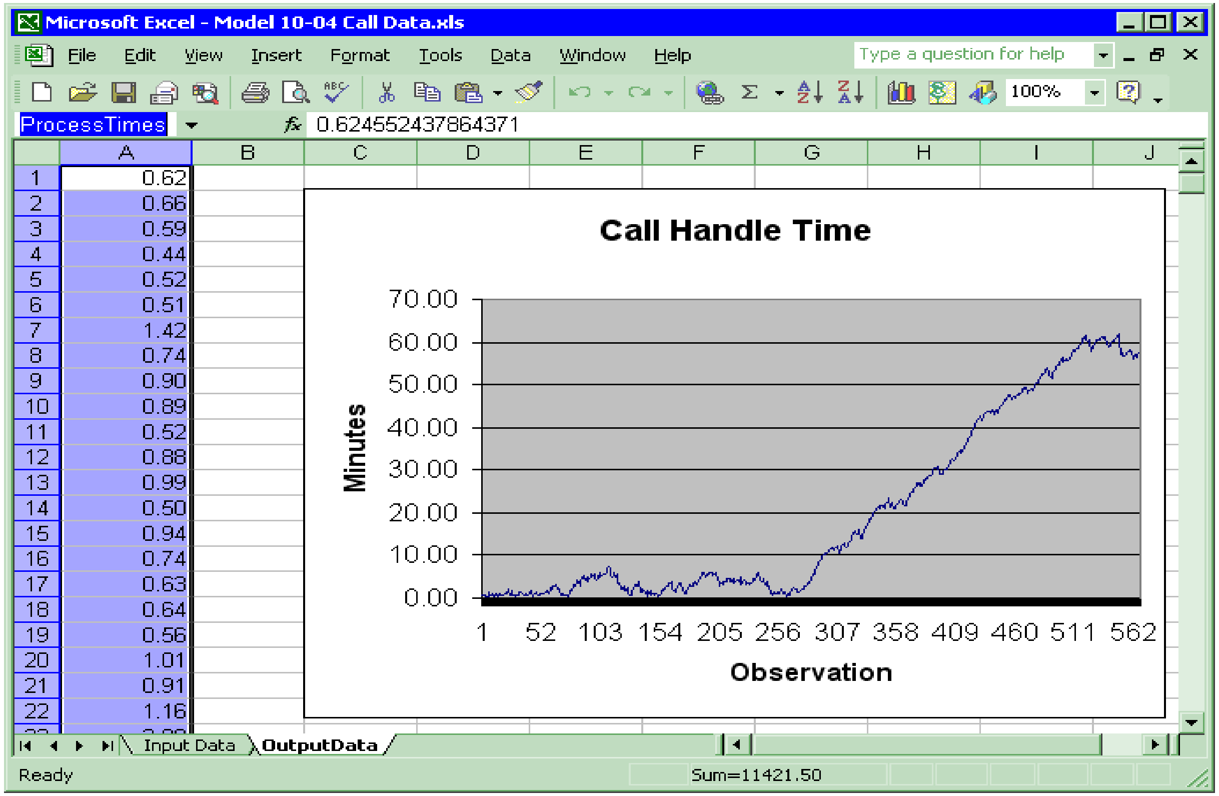
Assignments

Type: Other

Other: TNOW - Call Start Time

Writing Access and Excel Files

- **Spreadsheet options:**
  - You may predefine a plot on the named range and the plot will be built dynamically as data is added to the file.



ProcessTimes

0.624552437864371

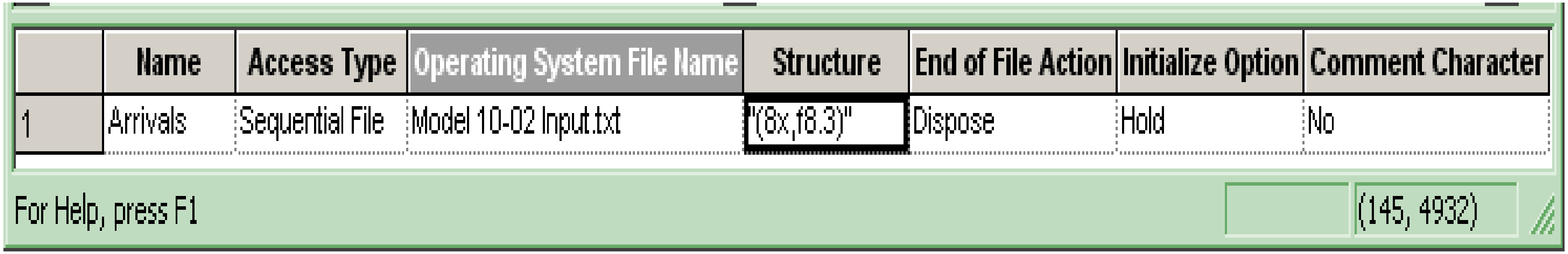
	A
1	0.62
2	0.66
3	0.59
4	0.44
5	0.52
6	0.51
7	1.42
8	0.74
9	0.90
10	0.89
11	0.52
12	0.88
13	0.99
14	0.50
15	0.94
16	0.74
17	0.63
18	0.64
19	0.56
20	1.01
21	0.91
22	1.16

Sum=11421.50

Advanced Reading and Writing

- **Formatting can be used to handle text files with fields not delimited by spaces:**

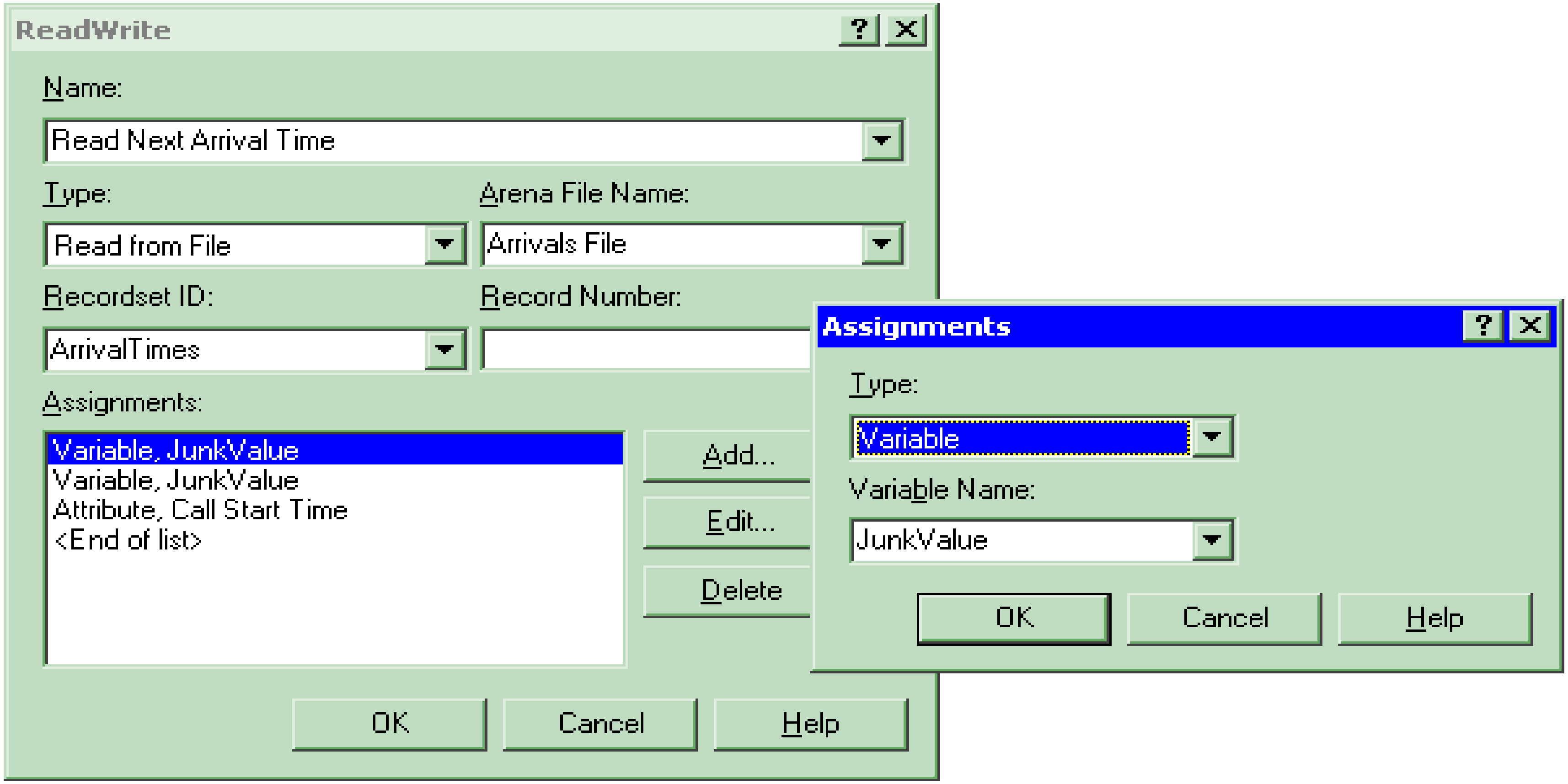
```
Part 1  1.038
Part 27 2.374
Part 5194.749
Part 67 9.899
Part 72 10.52
Part 16217.09
```



	Name	Access Type	Operating System File Name	Structure	End of File Action	Initialize Option	Comment Character
1	Arrivals	Sequential File	Model 10-02 Input.txt	"(8x,f8.3)"	Dispose	Hold	No

Advanced Reading and Writing

- **Skip columns in Access & Excel by using dummy variables:**



ReadWrite

Name: Read Next Arrival Time

Type: Read from File

Arena File Name: Arrivals File

Recordset ID: ArrivalTimes

Record Number:

Assignments:

Variable, JunkValue
Variable, JunkValue
Attribute, Call Start Time
<End of list>

Assignments

Type: Variable

Variable Name: JunkValue

Advanced Reading and Writing

- **Advanced data access is available using *Access Type* of *Active Data Objects (ADO)* and a *Connection String:***
  - ***Excel With Headings Using ADO***

```
Provider=Microsoft.JET.OLEDB.4.0;
Data Source=C:\Documents\Book1.xls;
Extended Properties=”””Excel 8.0; HDR=Yes;”””
```

  - ***SQL Commands Using ADO***

```
Driver={SQL Server};
Server=RSI-Joe; Database=BizBikes;
Uid=BizWareUser;  pwd=MyPassword
```

Use two double quotes for each embedded double quote

What We'll Do ...

- Reading and Writing Data Files (ReadWrite)
- ActiveX™ and Visual Basic® for Applications (VBA)
- Creating Modules with Arena Professional Edition

ActiveX Automation

- **Program applications to “automate” tasks**
  - Act on themselves (e.g., macros in Excel)
  - Act on other applications (e.g., Arena creating Excel file)
- **External programming languages**
  - C++, Visual Basic®, Java, etc.
- **Visual Basic for Applications (VBA) programming embedded in application**
  - Microsoft Office®, Visio®, AutoCAD®, Arena®, …
- **Both types work together (e.g., Arena VBA controlling Excel)**

Application Object Model

- **Objects**: **application *components* that can be controlled**
- **Properties**: ***characteristics* of objects**
- **Methods**: ***actions* performed on or by objects**

Arena Objects	Properties	Methods
Application	Visible	Show
Model	Name, State	Close, Go
View	Background Color	Zoom In
…		

Name three arena objects?

Visual Basic for Applications (VBA)

- Included with Arena
- Full Visual Basic programming environment
- Code stored with Arena model (*.doe*) file
- UserForms (dialogs) for custom interfaces
- Code-debugging tools
- Comprehensive online help
- Visual Basic Editor window: “child” of Arena (Tools/Show Visual Basic Editor)

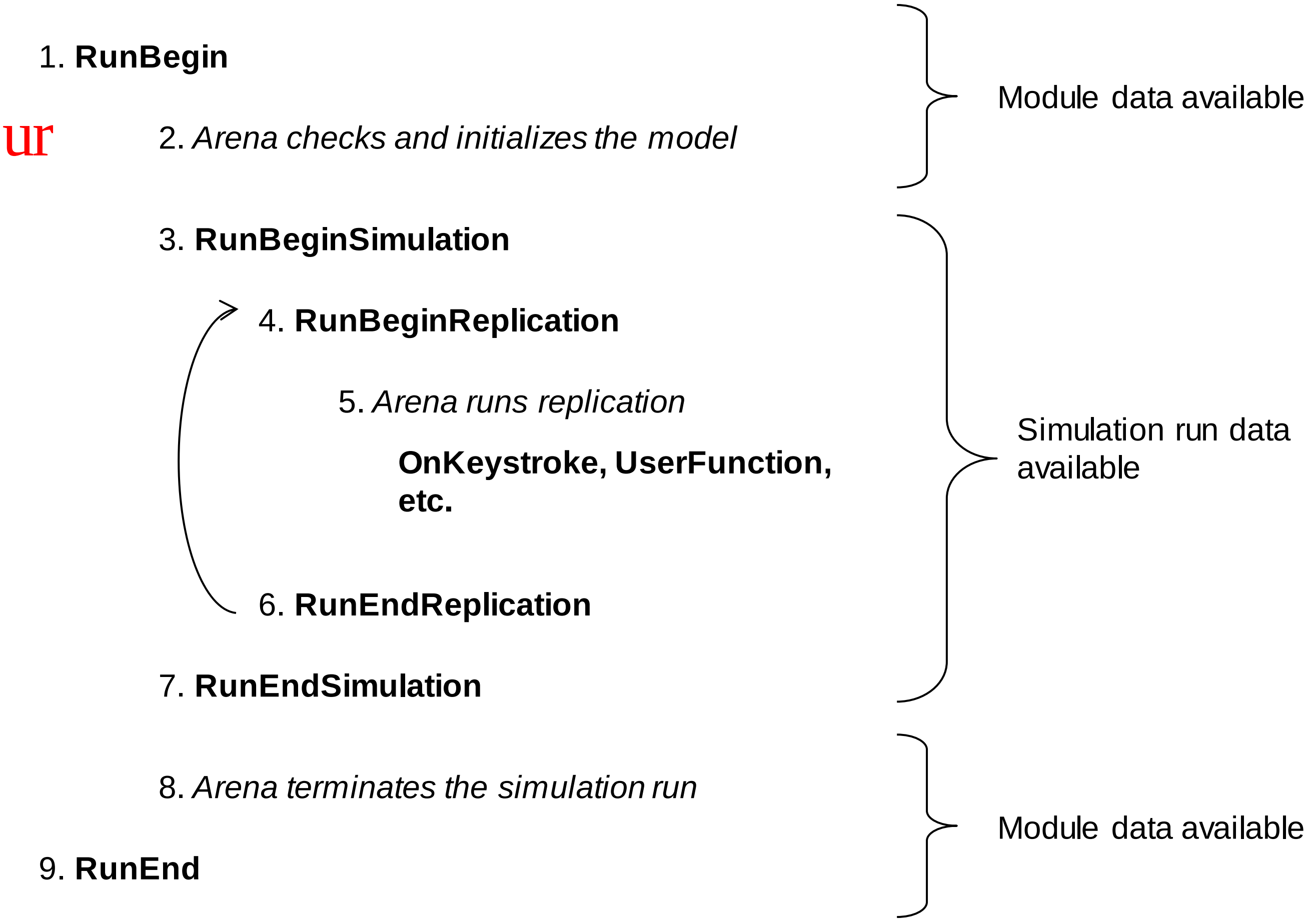
Built-in Arena VBA Events

- **ThisDocument: accesses objects, events in Arena's object model**
- **Built-in VBA events: locations where VBA code can be activated**
  - Pre-run events (e.g., DocumentOpen)
  - Arena-initiated run events (e.g., RunBegin, RunEndReplication)
  - Model/user-initiated run events (e.g., UserFunction, VBA_Block_Fire)
- **Type code in Visual Basic Editor to populate an event**

Simulation Run VBA Events

- **Arena/VBA sequence of events when model runs:**

Name four arena evens?



1. **RunBegin**

2. *Arena checks and initializes the model*

Module data available

3. **RunBeginSimulation**

4. **RunBeginReplication**

5. *Arena runs replication*

OnKeystroke, UserFunction, etc.

6. **RunEndReplication**

7. **RunEndSimulation**

Simulation run data available

8. *Arena terminates the simulation run*

9. **RunEnd**

Module data available

Arena’s Object Model

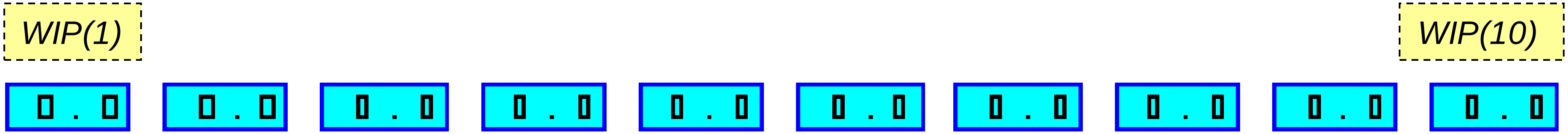
- **Model-window objects: items placed in model window, such as:**
  - Modules
  - Connections
  - Lines
- **SIMAN object: simulation run data, such as:**
  - Variable values
  - Queue lengths
  - Simulation time
- **Structural objects: access general functions**
  - Application
  - Panels

Sample: Create Ten Status Variables

```
Dim oModel As Arena.Model
Dim i As Integer
Dim nX As Long

' Add the status variables to this Arena model
Set oModel = ThisDocument.Model

nX = 0                   ' Start at x position 0
For i = 1 To 10
    ' Add a status variable to the model window
    oModel.StatusVariables.Create nX, 0, _
       nX + 400, 150, "WIP(" & i & ")", "**.*", False, _
       RGB(0, 0, 255), RGB(0, 255, 255), RGB(0, 0, 0), "Arial"
    ' Move over 500 world units for next position
    nX = nX + 500
Next i
```

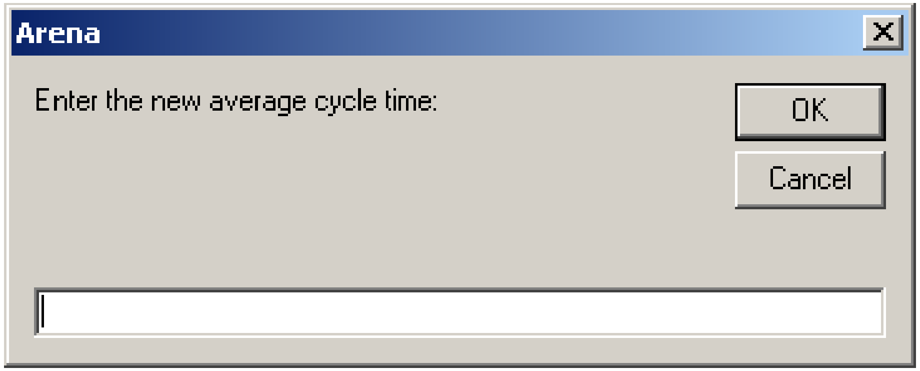


Sample: Assign Variable Value During Run

```
Dim oSIMAN As Arena.SIMAN
Dim nVarIndex As Long
Dim sNewValue As String

' Prompt for a new value
sNewValue = InputBox("Enter the new average cycle time:")

' Assign their answer to the Mean Cycle Time variable
Set oSIMAN = ThisDocument.Model.SIMAN
nVarIndex = oSIMAN.SymbolNumber("Mean Cycle Time")
oSIMAN.VariableArrayValue(nVarIndex) = sNewValue
```



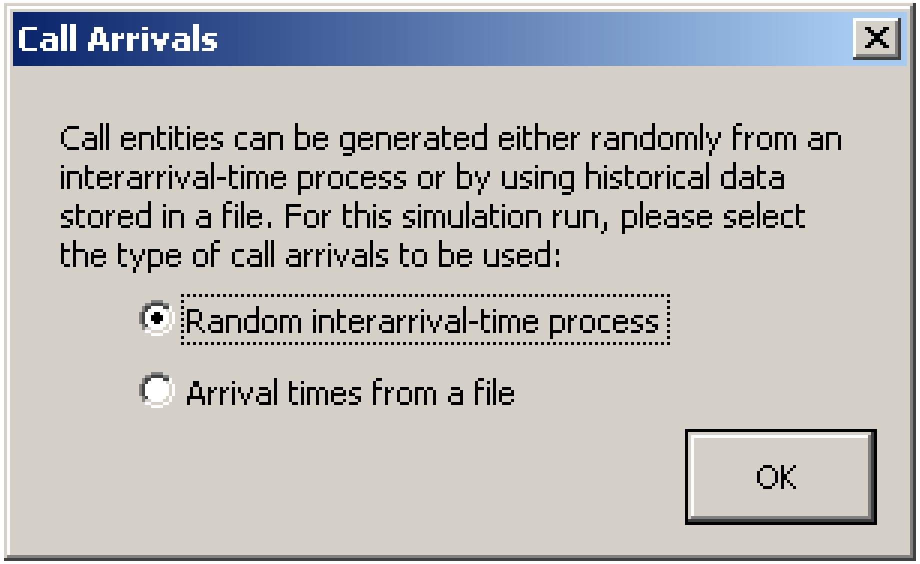
Enter the new average cycle time:

Arena Macro Recorder

- A Macro is a VBA function to perform a task.
- Macro recording automatically creates the VBA code to reproduce the actions you take while performing the steps in the task.
- Use the Record Macro toolbar to start/stop and pause/resume recording.
- Useful for automating repetitive tasks.
- Ideal for learning VBA commands and prototyping functions.

Model 10-05: Presenting Arrival Choices to the User

- **Prompt at beginning of run**
  - Generate entities via random process … or …
  - Generate based on arrival times stored in a file



Call entities can be generated either randomly from an interarrival-time process or by using historical data stored in a file. For this simulation run, please select the type of call arrivals to be used:

● Random interarrival-time process

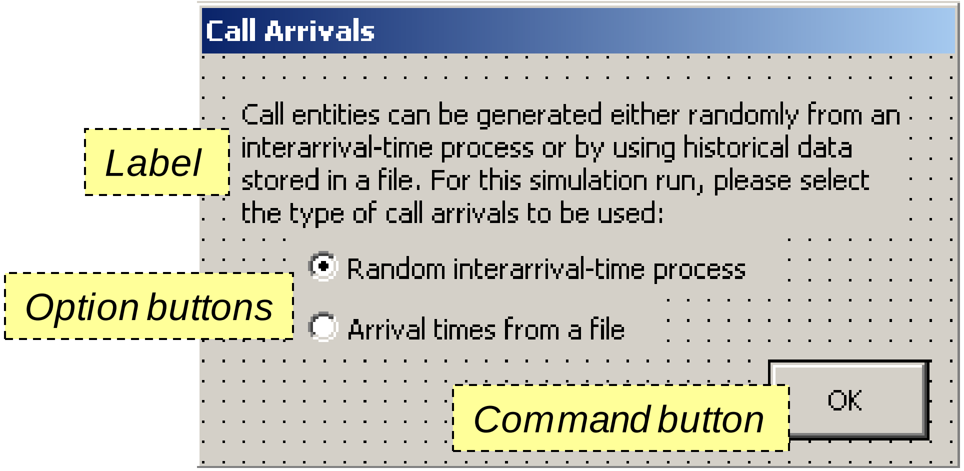
○ Arrival times from a file

Our Approach (cont’d.)

- **Change Max Arrivals field in Create module to turn “on” or “off” its generation of entities**
- **Random interarrival-time process**
  - ***Create Call*** module: ***Infinite***
  - ***Create Control Entity to Read Data*** module: ***0***
- **Arrival times from a file**
  - ***Create Call*** module: ***0***
  - ***Create Control Entity to Read Data*** module: ***1***
- **Give unique “tag” to each Create module (so VBA code can find them)**

VBA UserForm

- **Insert/UserForm menu in Visual Basic Editor**
- **Drop controls from Control Toolbox (labels, option buttons, command button)**



Call Arrivals

Call entities can be generated either randomly from an interarrival-time process or by using historical data stored in a file. For this simulation run, please select the type of call arrivals to be used:

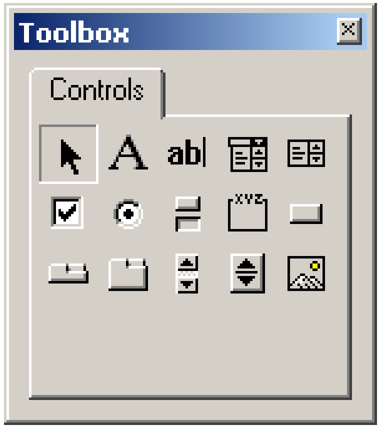
● Random interarrival-time process

○ Arrival times from a file

Label

Option buttons

Command button



Show the UserForm

- **At beginning of run (ModelLogic_RunBegin), show the form:**

```
Option Explicit
Private Sub ModelLogic_RunBegin()
    ' Display the UserForm to ask for the type of arrivals
    frmArrivalTypeSelection.Show

    Exit Sub
End Sub
```

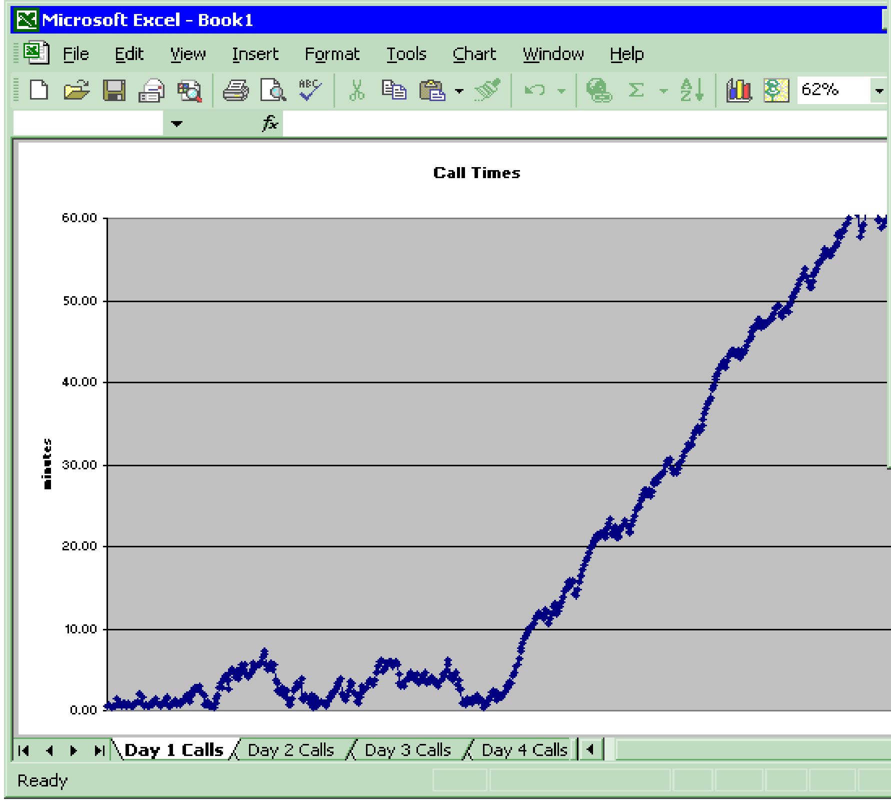
- **Program control passes to the form until it’s closed**
- **Arena run “suspended” while form is in control**

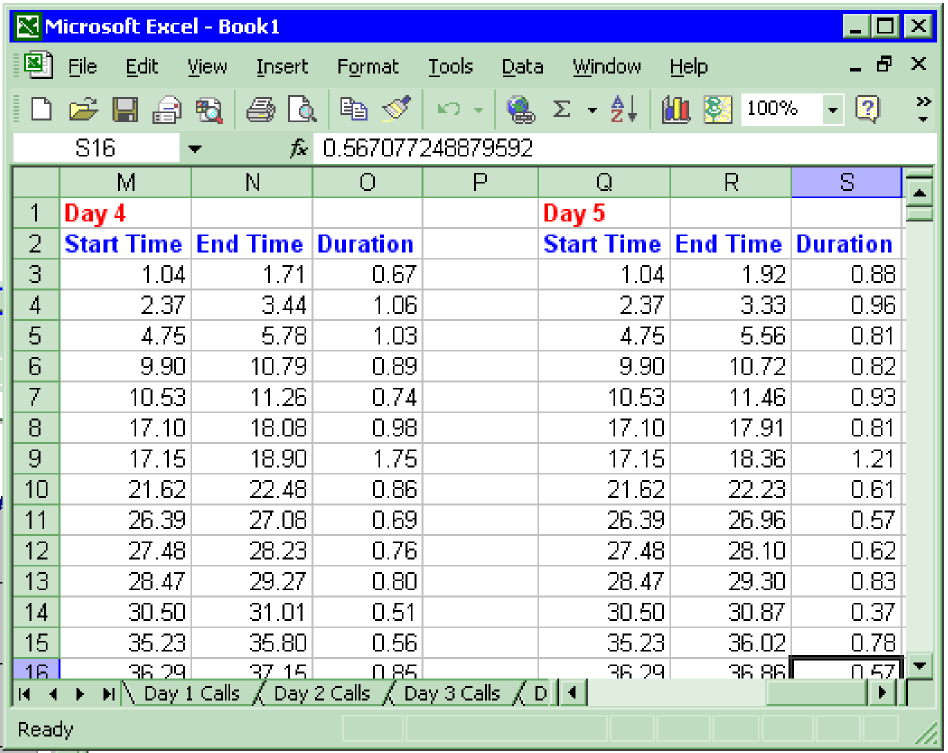
Change Module Data On OK

- **When user clicks OK button on form, modify the Create module data**
  - Find the Create modules
  - Set the Max Arrivals fields
  - Play a sound
  - Close the UserForm
- **When form is closed, simulation run commences with the new data values in the Create modules**

Model 10-06: Record Call Data in Microsoft Excel

- **Our goal:**
  - Raw call data tables
  - Daily call duration charts





S16 0.567077248879592

	M	N	O	P	Q	R	S
1	Day 4				Day 5		
2	Start Time	End Time	Duration		Start Time	End Time	Duration
3	1.04	1.71	0.67		1.04	1.92	0.88
4	2.37	3.44	1.06		2.37	3.33	0.96
5	4.75	5.78	1.03		4.75	5.56	0.81
6	9.90	10.79	0.89		9.90	10.72	0.82
7	10.53	11.26	0.74		10.53	11.46	0.93
8	17.10	18.08	0.98		17.10	17.91	0.81
9	17.15	18.90	1.75		17.15	18.36	1.21
10	21.62	22.48	0.86		21.62	22.23	0.61
11	26.39	27.08	0.69		26.39	26.96	0.57
12	27.48	28.23	0.76		27.48	28.10	0.62
13	28.47	29.27	0.80		28.47	29.30	0.83
14	30.50	31.01	0.51		30.50	30.87	0.37
15	35.23	35.80	0.56		35.23	36.02	0.78
16	36.29	37.15	0.85		36.29	36.86	0.57

Using ActiveX Automation in VBA

- **Reference the Excel Object Library**
  - Tools/References menu in Visual Basic Editor
  - Check the Microsoft Excel Object Library
  - Establishes link between Arena VBA and Excel
- **Object variables from application’s object model**
  - *Excel.Application*, *Excel.Workbook*
  - *Arena.SIMAN*
- **Starting Excel**
  - *CreateObject*: starts application, returning “handle” to the program (stored in *oExcelApp* variable)
  - *oExcelApp.Workbooks.Add*: similar to “File/New” in Excel

Retrieving Simulation Data

- **ThisDocument**
  - Built-in variable accessing the Arena model
  - Use only within Arena’s VBA

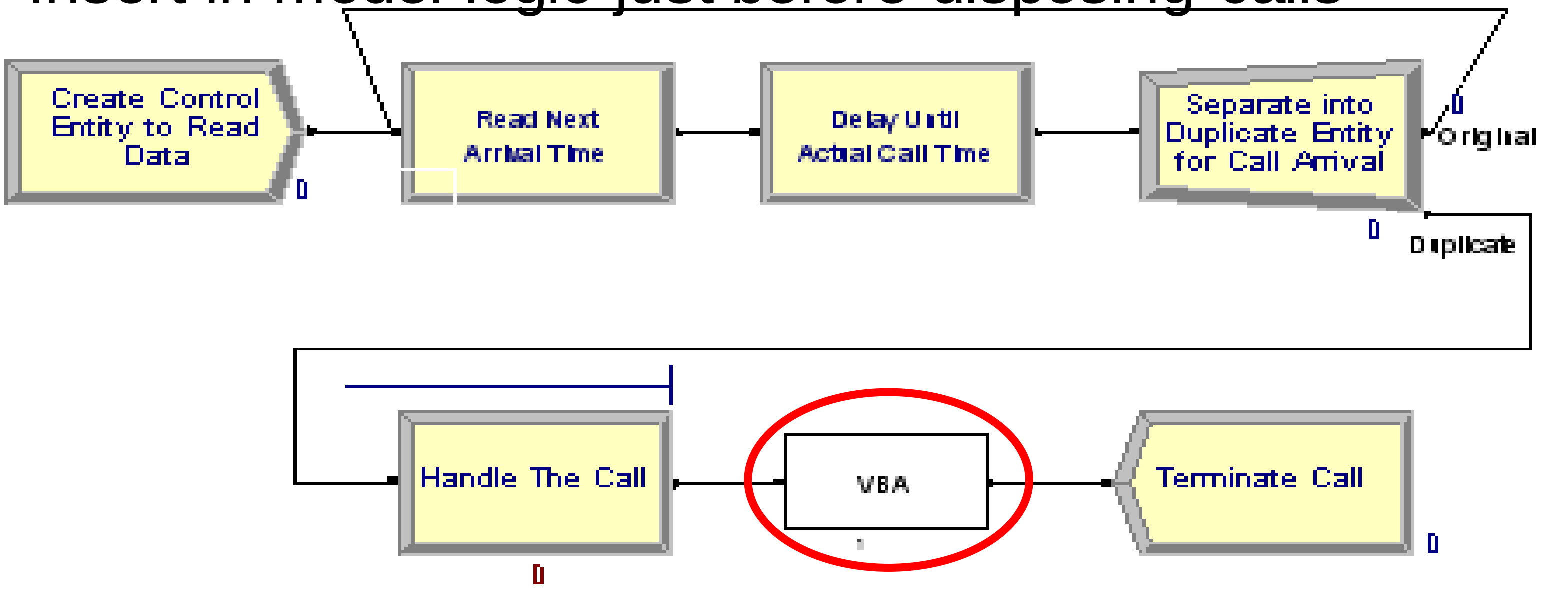
- **ThisDocument.Model.SIMAN**
  - Used to access simulation run data
  - Browse (**F2**) in VBA window for full list of variables
  - Active only when simulation run data is available -- i.e., built-in events:
    - after (and including) ModelLogic_RunBeginSimulation
    - before (and including) ModelLogic_RunEndSimulation

Our Approach

- **VBA ModelLogic_RunBeginSimulation**
  - Called ***once*** at the beginning of the simulation run
    - Start Excel with a new spreadsheet (“Workbook”)
    - Format header rows for data worksheet
- **VBA ModelLogic_RunBeginReplication**
  - Called at the beginning of ***each replication***
    - Write headers for the three columns and the Day
    - Format the data columns

Our Approach (cont'd.)

- **VBA Module (Blocks panel)**
  - Insert in model logic just before disposing calls



Create Control Entity to Read Data
Read Next Arrival Time
Delay Until Actual Call Time
Separate into Duplicate Entity for Call Arrival
Original
Duplicate
Handle The Call
VBA
Terminate Call
0

- **VBA Code**
  - VBA modules numbered as they're placed, with corresponding VBA_Block_<n>_Fire events in VBA

See the recorded the lecture for the project to better understand the VBA and excel integration

Chapter 7
Random-Number Generation

Banks, Carson, Nelson & Nicol

Discrete-Event System Simulation

Properties of Random Numbers

- **Two important statistical properties:**
  - Uniformity
  - Independence.
- **Random Number, R_i, must be independently drawn from a uniform distribution with pdf:**

$$f(x) = \begin{cases} 1, & 0 \le x \le 1 \\ 0, & \text{otherwise} \end{cases}$$

$$E(R) = \int_0^1 x\,dx = \left.\frac{x^2}{2}\right|_0^1 = \frac{1}{2}$$

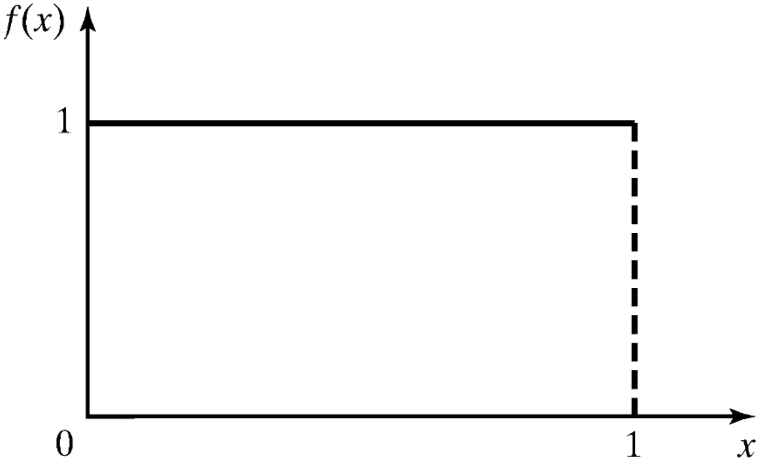


Figure: pdf for random numbers

Generation of Pseudo-Random Numbers

- **“Pseudo”, because generating numbers using a known method removes the potential for true randomness.**
- **Goal: To produce a sequence of numbers in [*0,1*] that simulates, or imitates, the ideal properties of random numbers (RN).**
- **Important considerations in RN routines:**
  - Fast
  - Portable to different computers
  - Have sufficiently long cycle
  - Replicable
  - Closely approximate the ideal statistical properties of uniformity and independence.

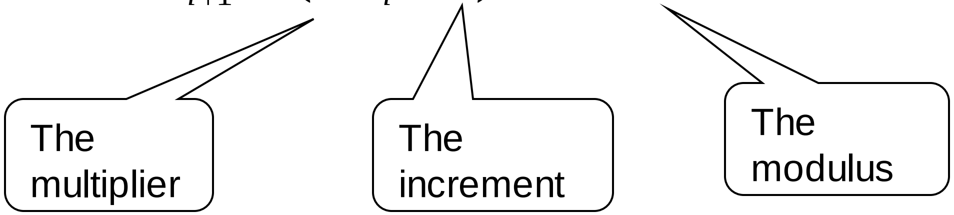
Techniques for Generating Random Numbers

- Linear Congruential Method (LCM).
- Combined Linear Congruential Generators (CLCG).
- Random-Number Streams.

Linear Congruential Method

- To produce a sequence of integers, *X_1, X_2, …* between *0* and *m-1* by following a recursive relationship:

$$X_{i+1} = (aX_i + c) \bmod m, \quad i = 0,1,2,...$$



The multiplier

The increment

The modulus

- The selection of the values for *a*, *c*, *m*, and *X_0* drastically affects the statistical properties and the cycle length.
- The random integers are being generated [*0,m-1*], and to convert the integers to random numbers:

$$R_i = \frac{X_i}{m}, \quad i = 1,2,...$$

Example

[LCM]

- **Use $X_0 = 27$, $a = 17$, $c = 43$, and $m = 100$.**
- **The X_i and R_i values are:**

$X_1 = (17{*}27+43) \bmod 100 = 502 \bmod 100 = 2, \qquad R_1 = 0.02;$

$X_2 = (17{*}2+43) \bmod 100 = 77, \qquad R_2 = 0.77;$

$X_3 = (17{*}77+43) \bmod 100 = 52, \qquad R_3 = 0.52;$

…

Random-Variate Generation

Need for Random-Variates

- We, usually, model uncertainty and unpredictability with statistical distributions
- Thereby, in order to run the simulation models involving uncertainty, we need to get samples from these statistical distributions
- Here, we assume that the distributions (type and parameters) are already specified
- ARENA comes with ready functions to sample from specified distributions.
- It is still useful to know how it is done
- We will generate the variates always using random numbers, whose generation is discussed previously

Inverse Transform Technique -- Exponential Distribution

- **To generate samples from exponential distribution we use the inverse transform technique**

Step 1. Compute the cdf of the desired random variable X, F(x).

Step 2. Find the inverse of F(x) function

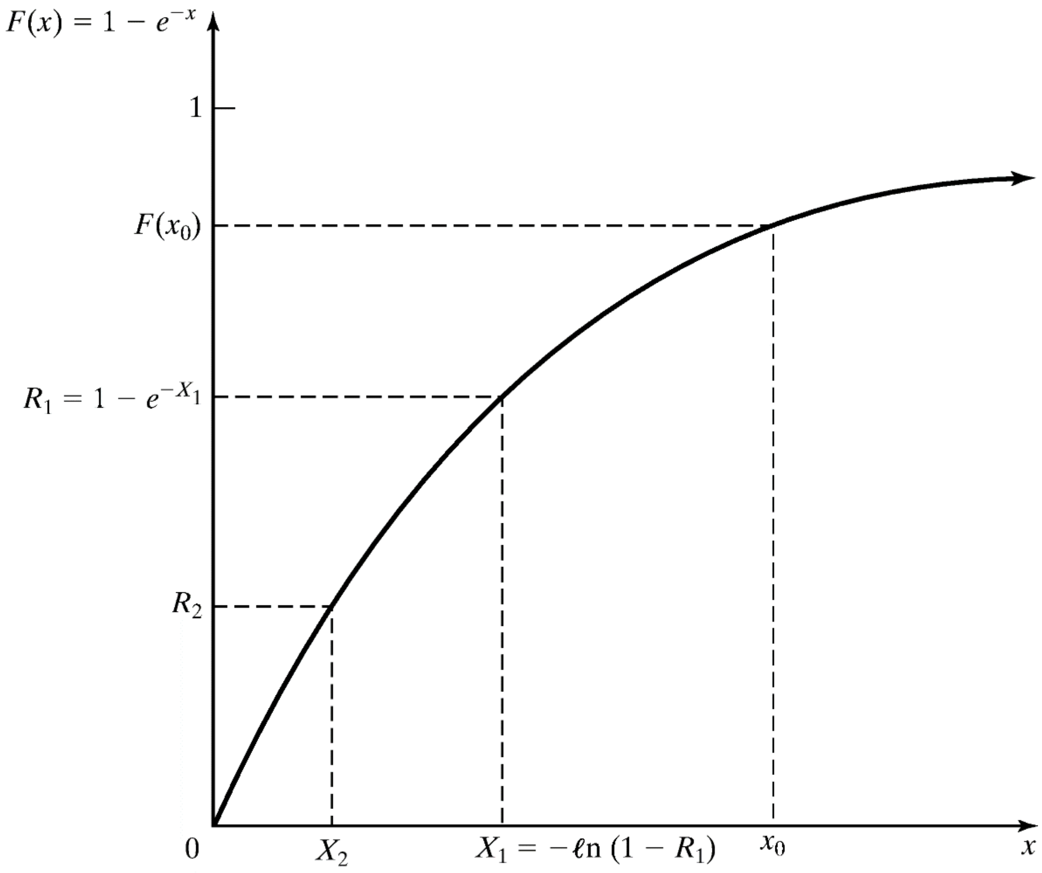
$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & x \ge 0 \\ 0, & x < 0 \end{cases} \qquad F(x) = \begin{cases} 1 - e^{-\lambda x}, & x \ge 0 \\ 0, & x < 0 \end{cases}$$

Step 4. Generate uniform random variables R_1, R_2, R_3, … and compute the desired random variates by

$$y = F(x) \Rightarrow y = 1 - e^{\lambda x} \Rightarrow x = F^{-1}(y) = -\frac{1}{\lambda}\ln(1-y)$$

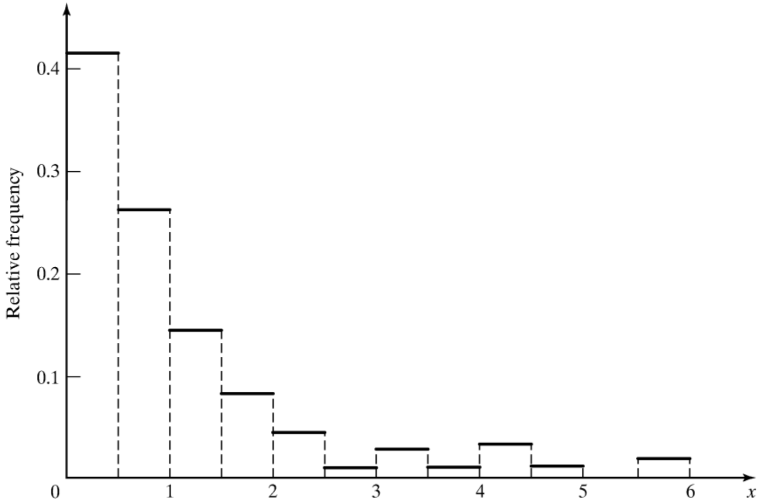
$$X_i = F^{-1}(R_i) = -\frac{1}{\lambda}\ln(1-R_i)$$

Inverse Transform Technique -- Exponential Distribution



Inverse Transform Technique -- Exponential Distribution

- **Example: Generate *200* variates X_i with distribution *exp(*λ*= 1)***



  - Check: Does the random variable X_1 have the desired distribution?

Proof?

- **Can you prove that the numbers you have generated are indeed samples from an exponential distribution?**

Other Distributions

- **Uniform Distribution [UN(a,b)] (X = a + (b-a)R)**
  - Does it really work?
- **Weibull Distribution**

$$f(x) = \begin{cases} \dfrac{\beta}{\alpha^{\beta}} x^{\beta-1} e^{-\left(x/\alpha\right)^{\beta}}, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

  - Derive the transformation
- **Triangular ….**
- **The moral is if we can find a closed-form inverse of the cdf for a distribution we can use this method to get samples from that distribution**

Continuous Functions without a Closed-Form Inverse

- Some distributions do not have a closed form expression for their cdf or its inverse (normal, gamma, beta, …)
- What can be done then?
- Approximate the inverse cdf
- For the standard normal distribution:

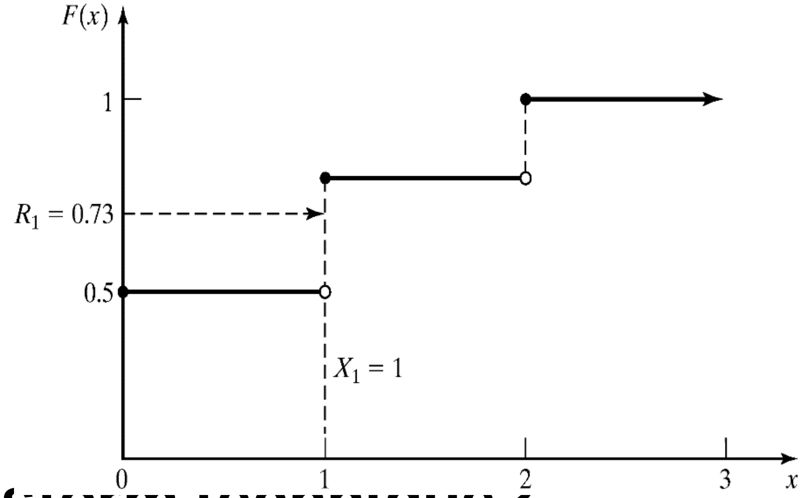
$$X = F^{-1}(R) \approx \frac{R^{0.135} - (1-R)^{0.135}}{0.1975}$$

- This approximation gives at least one-decimal place accuracy in the range [0.0012499, 0.9986501]

Discrete Distributions

- **An Empirical Discrete Distribution**
  - p(0) = P(X=0) = 0.50
  - p(1) = P(X=1) = 0.30
  - p(2) = P(X=2) = 0.20

$$F(x) = \begin{cases} 0, & x < 0 \\ 0.5, & 0 \le x < 1 \\ 0.8, & 1 \le x < 2 \\ 1.0, & 2 \le x \end{cases}$$



- **Can we apply the inverse transform technique?**

Discrete Distributions

- Let $x_0 = -\infty$, and $x_1, x_2, \ldots, x_n$, be the ordered probability mass points for the random variable X
- Let R be a random number

$$if \quad F(x_{i-1}) < R \leq F(x_i) \;\Rightarrow\; X = x_i$$

Discrete Distributions

- **A Discrete Uniform Distribution**

$$p(x) = \frac{1}{k}, \quad x = 1, 2, \ldots, k$$

$$F(x) = \begin{cases} 0, & x < 1 \\ \dfrac{i}{k}, & i \le x < i + 1, \ \ for \ \ i \in [1, k - 1] \\ 1, & k \le x \end{cases}$$

$$if \quad \frac{i-1}{k} < R \le \frac{i}{k} \quad \Rightarrow \quad X = i$$

$$or \quad X = roundup(kR)$$

Discrete Distributions

- **The Geometric Distribution**

$$p(x) = p(1-p)^{x}\ ,\quad x = 0, 1, 2,\ \ldots$$

$$F(x) = 1-(1-p)^{x+1}\ ,\quad x = 0, 1, 2, \ldots$$

- **Some algebraic manipulation and…**

$$X\ =\ roundup\left(\frac{\ln(1-R)}{\ln(1-p)}-1\right)$$

Acceptance-Rejection Technique

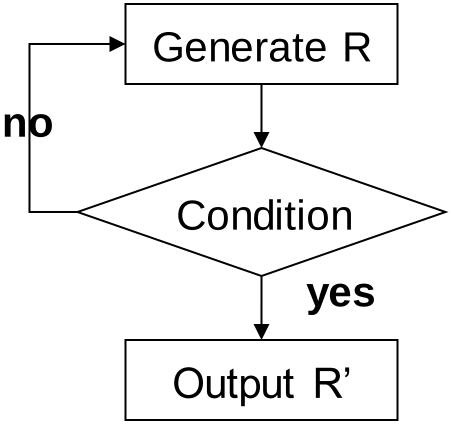
- **Useful particularly when inverse cdf does not exist in closed form, a.k.a. thinning**
- **Illustration: To generate random variates, *X* ~ U(*1/4, 1*)**

Procedures:

Step 1. Generate R ~ U[0,1]

Step 2a. If R >= ¼, accept X=R.

Step 2b. If R < ¼, reject R, return to Step 1



- ***R* does not have the desired distribution, but *R* conditioned (*R'*) on the event {*R* ≥ ¼} does.**
- **Efficiency: Depends heavily on the ability to**

Acceptance-Rejection Technique

- **Poisson Distribution**

$$p(n) = P(N = n) = \frac{e^{-\alpha}\alpha^{n}}{n!}, \quad n = 0, 1, 2, \ldots$$

  - N can be interpreted as number of arrivals from a Poisson arrival process during one unit of time
  - Then time between the arrivals in the process are exponentially distributed with rate α

$$N = n \iff \sum_{i=1}^{n} A_i \leq 1 < \sum_{i=1}^{n+1} A_i$$

Acceptance-Rejection Technique

$$\sum_{i=1}^{n} A_i \le 1 < \sum_{i=1}^{n+1} A_i \Leftrightarrow \sum_{i=1}^{n} -\frac{1}{\alpha}\ln R_i \le 1 < \sum_{i=1}^{n+1} -\frac{1}{\alpha}\ln R_i$$

$$\Leftrightarrow \prod_{i=1}^{n} R_i \ge e^{-\alpha} > \prod_{i=1}^{n+1} R_i$$

- **Step 1. Set n = 0, and P = 1**
- **Step 2. Generate a random number R_{n+1} and let P = P. R_{n+1}**
- **Step 3. If P < $e^{-\alpha}$, then accept N = n. Otherwise, reject current n, increase n by one, and return to step 2**
- **How many random numbers will be used on the average to generate one Poisson variate?**

Direct Transformations

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-t^2/2}\, dx$$

- **Consider two normal variables Z$_1$ and Z$_2$**

$$B^2 = Z_1^2 + Z_2^2 \sim Chi - square \text{ with two degrees of freedom}$$

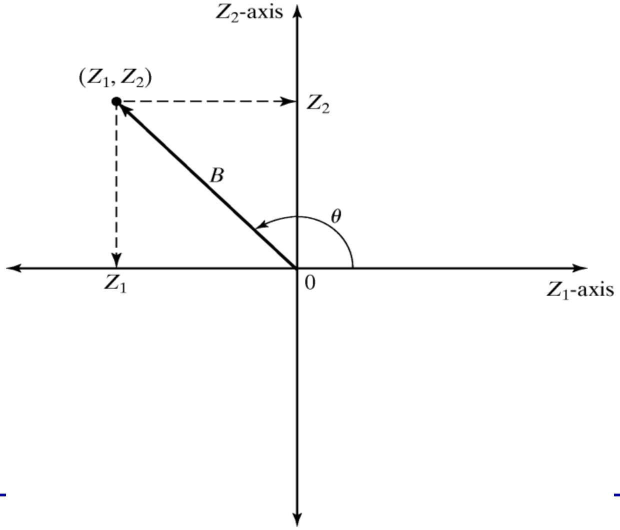
$$\text{(Exponential with parameter 2)}$$

$$\theta = \tan^{-1}\left(\frac{Z_1}{Z_2}\right) \sim Uniform[0, 2\pi]$$

$$B = \sqrt{-2\ln R}$$

$$Z_1 = \sqrt{-2\ln R_1}\cos\left(2\pi R_2\right)$$

$$Z_2 = \sqrt{-2\ln R_1}\sin\left(2\pi R_2\right)$$


Z_2-axis
(Z_1, Z_2)
Z_2
B
θ
Z_1
0
Z_1-axis