

Computer Simulation

Module 5: Arena

Dave Goldman, Ph.D.

Professor

Stewart School of Industrial and Systems Engineering

Batch, Separate, and Record
Modules

Lesson Overview

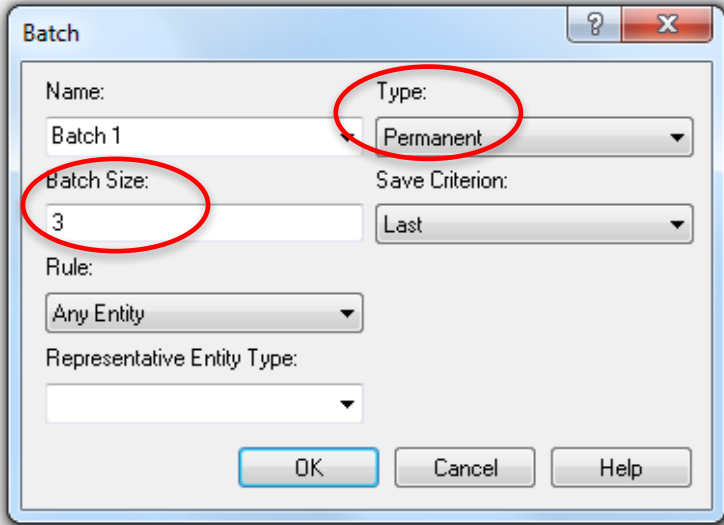
Last Lesson: Looked at a bunch of ways to obtain information during and after the run – e.g., histograms, graphs, output reports.

This Lesson: Discuss the Batch, Separate, and Record modules.

These are the last modules for the Basic Process template... We're making good progress!

Batch Module

- Combine (“batch”) multiple customers into one “super”-customer.
- Batch Size = 3 accumulates 3 guys before sending off the super-customer.
- If info about the individual customers in a batch won’t be needed later on, then choose Type = Permanent.
- But if you want eventually to reconstitute the original members, set Type = Temporary.
- Temporary batches will need to be **split** before being Dispose’d (keep watching).



The screenshot shows a 'Batch' dialog box with the following fields and values:

- Name: Batch 1
- Type: Permanent (circled in red)
- Batch Size: 3 (circled in red)
- Save Criterion: Last
- Rule: Any Entity
- Representative Entity Type: (empty)

Buttons at the bottom: OK, Cancel, Help.

Separate Module



Separate

- Duplicate a single entity, or split multiple entities that had been combined in a Batch module.
- If dealing with a **Permanent** batch, usually use Duplicate Original to get several customers all with the same attributes.
- If dealing with a **Temporary** batch, use Split Existing Batch to reproduce customers with original attributes.

The screenshot shows the 'Separate' dialog box with the following settings: Name: 'Separate 2', Type: 'Duplicate Original' (circled in red), Percent Cost to Duplicates (0-100): '50', and # of Duplicates: '1' (circled in red).

The screenshot shows the 'Separate' dialog box with the following settings: Name: 'Separate 1', Type: 'Split Existing Batch' (circled in red), Member Attributes: 'Retain Original Entity Values' (circled in red).

Record Module



Record

- Collect statistics when an entity passes through the module.
- We'll talk more about this module as we encounter it in future examples.
- **Demo Time!** Now we'll look at a couple of permutations of the Batch and Separate modules.

Summary

This Time: Finished off the Basic Resources template with the Batch, Separate, and Record modules.

Next Time: We'll look at a bunch of Run Setup and other control functionalities.

Computer Simulation

Module 5: Arena

Dave Goldsman, Ph.D.

Professor

Stewart School of Industrial and Systems Engineering

Run Setup and Control

Lesson Overview

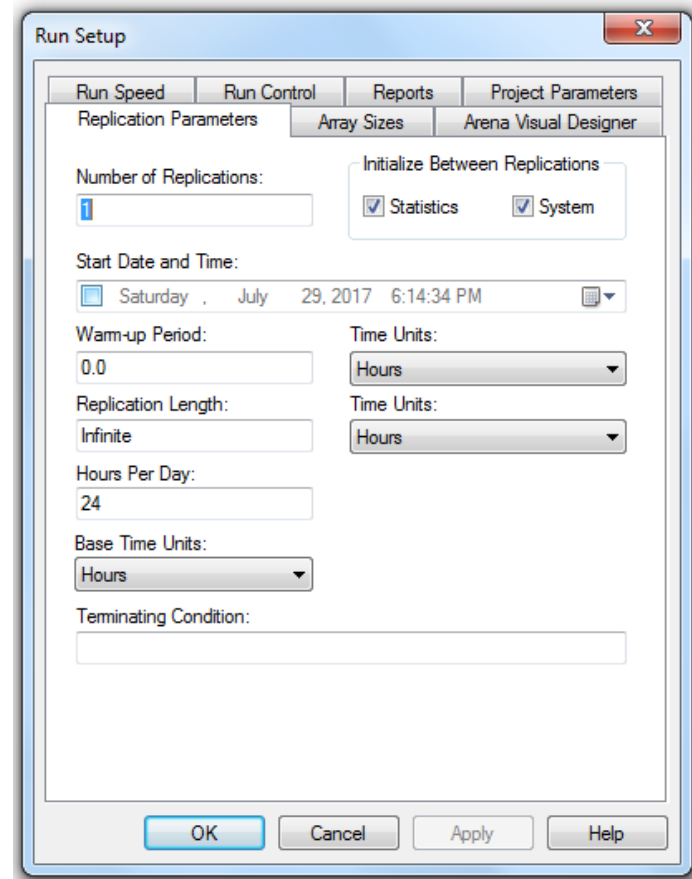
Last Lesson: Finished off the Basic Resources template with the Batch, Separate, and Record modules.

This Lesson: We'll be looking at a bunch of easy little trivia points to help you get your simulation runs going efficiently.

Run Setup

Run > Setup gives you **lots** of stuff.

- Replication Parameters tab
 - Number of Reps: # of indep runs.
 - Initialize Between Reps: Do you start the stats collection and/or system state from scratch for each replication?
 - Warm-up Period: How long to run before you start keeping data.
 - Rep Length: How long each run is.
 - Terminating Condition: Any special ways to stop the simulation (besides rep length condition from this tab or max arrivals from Create module)?



Run Setup (cont'd)

- Run Speed tab: Often need to speed up or slow down the simulation by more than what you can do on the main screen.
- Reports: Arena has a variety of reports that it gives upon completion of a run.
 - Contains info on customer waits, lengths of queues, server utilizations, user-defined variables, etc.
 - Category Overview summarizes all reps.
 - Category by Replication gives tedious info about each rep.
 - SIMAN Report gives concise text file.

ModelOut - Notepad

File Edit Format View Help

ARENA Simulation Results
ISyE - License: STUDENT

Summary for Replication 1 of 10

Project: Unnamed Project
Analyst: ISyE
Run execution Model revision

Replication ended at time : 48.0 Hours
Base Time Units: Hours

TALLY VARIABLES

Identifier	Average	Half width	Minimum	Maximum
Entity 1.VATime	1.0614	(Insuf)	.75651	1.0614
Entity 1.NVATime	.00000	(Insuf)	.00000	.00000
Entity 1.WaitTime	3.1369	(Insuf)	.00000	7.00000
Entity 1.TransTime	.00000	(Insuf)	.00000	.00000
Entity 1.OtherTime	.00000	(Insuf)	.00000	.00000
Entity 1.TotalTime	4.1983	(Insuf)	.92386	7.00000
Process 1.Queue.WaitingTime	3.1427	(Insuf)	.00000	7.00000

DISCRETE-CHANGE VARIABLES

Identifier	Average	Half width	Minimum	Maximum
Entity 1.WIP	3.9353	(Insuf)	.00000	8.00000
barber.NumberBusy	.98894	(Insuf)	.00000	1.00000
barber.NumberScheduled	1.0000	(Insuf)	1.0000	1.0000
barber.Utilization	.98894	(Insuf)	.00000	1.00000
Process 1.Queue.NumberInQueue	2.9463	(Insuf)	.00000	7.00000

OUTPUTS

Identifier	Value
Entity 1.NumberIn	45.000
Entity 1.NumberOut	44.000
barber.NumberSeized	45.000
barber.ScheduledUtilization	.98894

SIMAN report

Run Control

Run > Setup > Run Control gives a variety of ways that you can run the simulation.

Example: Batch mode, which turns off all graphics and result in extremely fast runs.

Demo Time! Let's...

- Look at some reports.
- Stop the simulation the first time the queue size hits 4.

Summary

This Time: Learned how to do various setup and run control tasks. For instance, we heard about independent reps, run termination from the setup menu, and output reports.

Next Time: A small case study for a two-channel manufacturing system, where we put a lot of the previous material together.

Computer Simulation

Module 5: Arena

Dave Goldman, Ph.D.

Professor

Stewart School of Industrial and Systems Engineering

Simple Two-Channel
Manufacturing Example

Lesson Overview

Last Lesson: Discussed various various run setup and control tasks. E.g., how to implement indep reps, run speed, reports.

This Lesson: Demo a two-channel manufacturing system, where we finally put a lot of the previous material together.

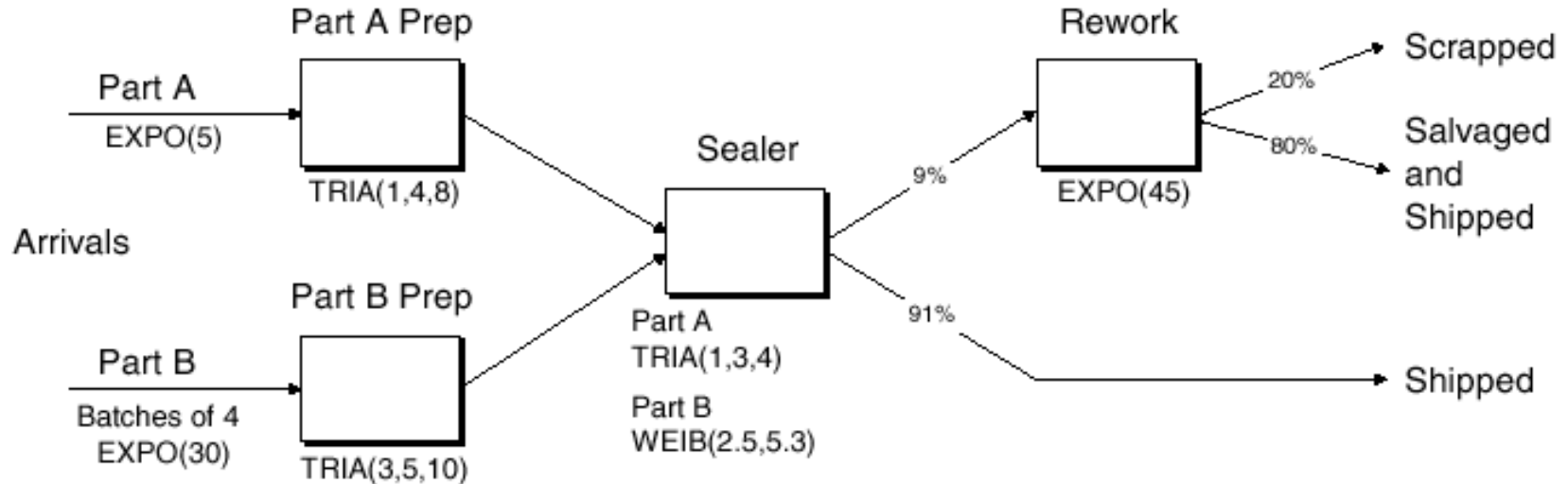
Watch how we use attributes!

The Story

- This is Model 4-1 “An Electronic Assembly and Test System” from the KSZ (2015) text, *Simulation with Arena*.
- Two different arrival streams
 - Type A parts show up 1-at-a-time; Type B’s 4-at-a-time.
 - Type A’s show up a little more often than Type B’s.
- A’s feed into a Prep A server; B’s go to Prep B. Different service times.
- Then the parts get processed by the **same** Sealer server, but again with **different service time distributions**.
- All parts undergo an inspection. If they pass, they exit.
- If they don’t pass, they go to a Rework server, and then another inspection. Whether or not they pass, they exit the system.

The Story (cont'd)

I stole this flowchart from KSZ...



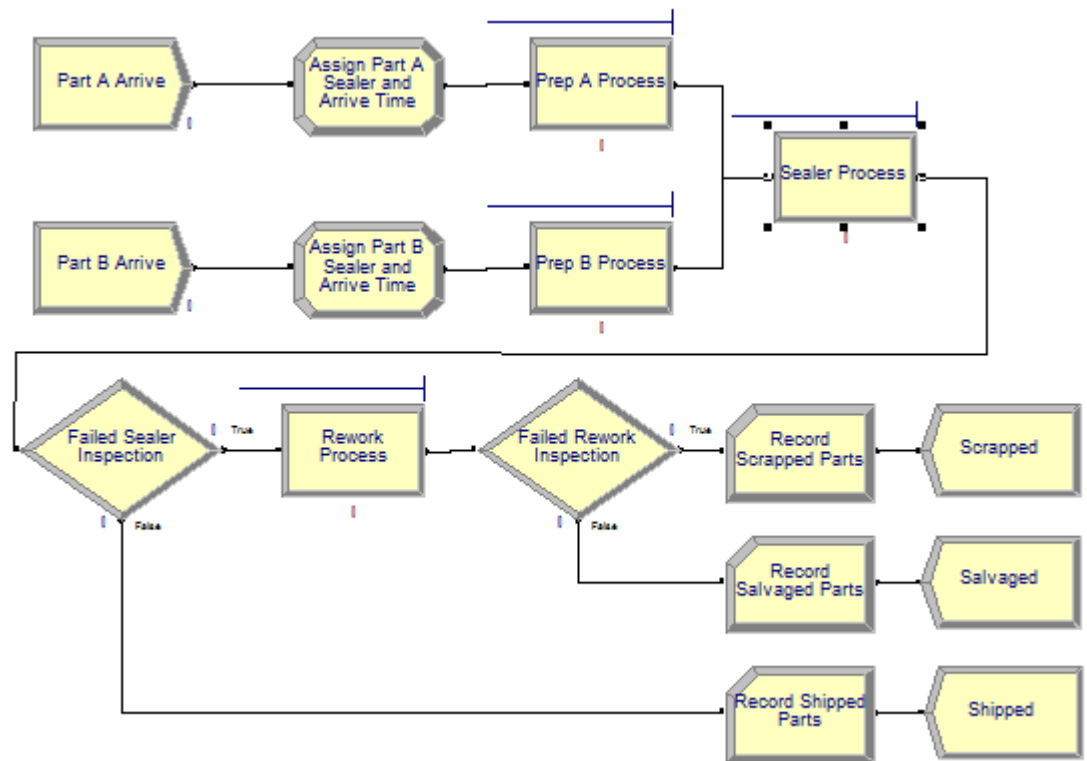
The Story (cont'd)

- How to handle different A and B service times at Sealer?
 - Trick 1: **Pre-assign** the service times as an **attribute** (Sealer Time) in an Assign module immediately after each customer arrives. Use that attribute regardless of being a Type A or B part.
 - Trick 2: While we're at it, use the Assign to store each customer's arrival time as an **attribute**. Use the Arena variable TNOW to do so.
- Record departure times just before parts get Dispose'd, This will allow us to get average cycle times (depart – arrival times) for any of the 3 types of parts (pass on first try, pass on second, fail both).

Let's Get Logical

- Dave's Alternative Trick: Is there another way to model the process time at the Sealer without having to assign a Sealer Time attribute for A's and B's?
- Yes! It involves a little work, but it's nice to know.
 - Note that the entity types (Part A and Part B) are assigned in the respective Create modules.
 - Instead of assigning the Sealer Time as an attribute in the Assign module, we can just wait until the Sealer Process module and use the following **logical** expression, where $(x==y) = 1$ if $x=y$, and 0 otherwise. (See why?)
 $((\text{Entity.Type} == \text{Part A}) * \text{TRIA}(1,3,4)) + ((\text{Entity.Type} == \text{Part B}) * \text{WEIB}(2.5,5.3))$

Demo Time!



Summary

This Time: Our first “real” system – a small manufacturing system.
Learned a couple of nice tricks involving attributes and logic.

Next Time: Fake Customers!

Computer Simulation

Module 5: Arena

Dave Goldsman, Ph.D.

Professor

Stewart School of Industrial and Systems Engineering

Fake Customers

Lesson Overview

Last Lesson: Demo'd a small electronic assembly and test system.

This Lesson: We'll talk about “fake” customers!

Idea: These aren't real customers at all... their purpose is just to do some task that Arena needs.

The Next Few Lessons

We'll now cover material that will lead us to the simulation of a call center:

- Fake Customers ← now
- The Advanced Process Template
- Resource Failures + Maintenance
- The Blocks Template
- The Joy of Sets
- Description of Call Center
- Call Center Demo

Fake Customers

- You can use “fake” customers to accomplish various tasks during a simulation.
- Not actual customers that you care about in terms of waiting times or use of resources.
- **Demo Time** will explain all!
 - Calculate normal probabilities
 - Keep track of which time period you’re in (part of Call Center example coming up)
 - Breakdown demon

Summary

This Time: Talked about fake customers. We'll be using these several times in upcoming applications.

Next Time: Now that we're experts in the Basic Process template, we'll move on to the Advanced Process template for more good stuff!

Computer Simulation

Module 5: Arena

Dave Goldman, Ph.D.

Professor

Stewart School of Industrial and Systems Engineering

The Advanced Process
Template

Lesson Overview

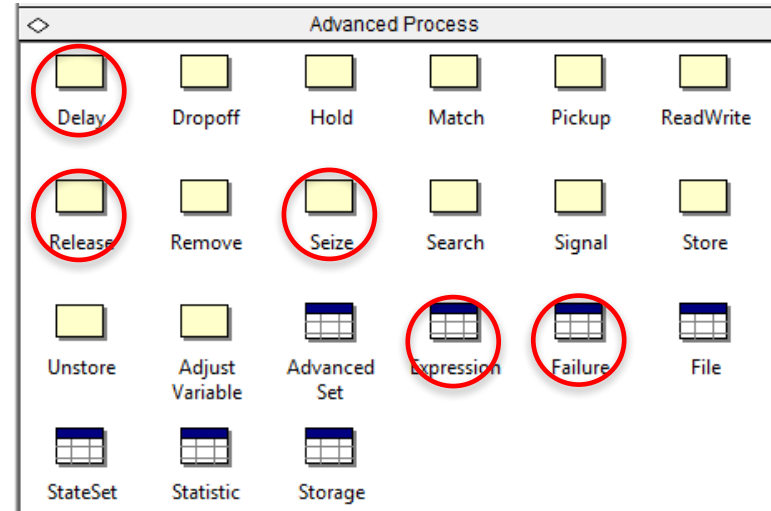
Last Lesson: Introduced “fake” customers, which can be used for a variety of purposes, including arrivals of breakdown demons.

This Lesson: We'll add to our arsenal of modules by introducing the Advanced Process template!

Idea: Lots of great new stuff, though I don't expect you to memorize everything.

Advanced Process Template

- File > Template Panel > Attach (and hopefully you find it in the Rockwell Software \ Arena \ Template directory).
- Lots of new modules and spreadsheets.
- I don't expect you to memorize them all.
- Concentrate for now on a few of these (comment on others later):
 - Seize, Delay, Release modules
 - Expression, Failure spreadsheets.



Seize, Delay, Release

- Why do we have separate Seize, Delay, and Release modules? Aren't they right there in the Process module?
- Yup, but sometimes you need to do things that are too complicated for the Process module version of Seize–Delay–Release.
- Now you can handle things like...
 - Seize–Assign–Delay–Release
 - Non-symmetric multiple Seize's and Release's
 - Complicated Seize's and Release's that might depend on sets of servers (as in the upcoming Call Center example)
- **Demo Time!**
 - Will talk about some of these Seize, Delay, Release issues.
 - Will also sneak in some wisdom about the Expression spreadsheet.

Summary

This Time: Learned about the Advanced Process template, which gives us a new bag of tricks to play with!

Next Time: How to deal with resource failures and maintenance.

Computer Simulation

Module 5: Arena

Dave Goldman, Ph.D.

Professor

Stewart School of Industrial and Systems Engineering

Resource Failures and
Maintenance

Lesson Overview

Last Lesson: Learned about the Advanced Process treasure chest.

This Lesson: We'll study model breakdowns and maintenance in a way that's more direct than via fake customers.

Idea: It's just a certain type of schedule!

Failures

- You can cause resource failures by scheduling “breakdown demons” (fake customers with high priority). But this isn’t elegant.
- Better way is to use the Resource and Failure spreadsheets in conjunction with each other.

Resource - Basic Process						
	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use
1 ▶	Drill Press ▼	Fixed Capacity	1	0.0	0.0	0.0

Failures

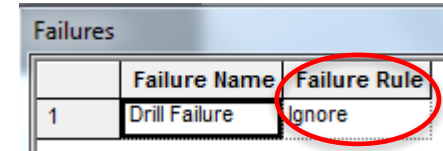
	Failure Name	Failure Rule
1	Drill Failure	Ignore

StateSet Name	Failures
	1 rows

Failure - Advanced Process					
	Name	Type	Count	Down Time	Down Time Units
1 ▶	Drill Failure	Count	10	Expo(30)	Minutes

Failures (cont'd)

- Go to Resource spreadsheet in the Basic Process template.
- Click on Failures column.
- Add a Failure Name.
- Choose Failure Rule (will discuss in a minute).
- Go to the Failure spreadsheet in the Advanced Process template, where you'll see your new failure name.
- Choose type of failure:
 - Count (failure after a certain # of arrivals)
 - Time (after a certain amount of time)
- Choose downtime (for repair): Can be any expression.



A screenshot of a table titled "Failures". The table has two columns: "Failure Name" and "Failure Rule". The first row contains the values "1" and "Drill Failure". The "Failure Rule" column is circled in red.

	Failure Name	Failure Rule
1	Drill Failure	Ignore

Remarks

- Can schedule multiple failures by using multiple rows of the Failures column in the Resource spreadsheet, e.g., type I failure, type II failure, scheduled maintenance.
- Types of Failure Rules:
 - Ignore (complete service of current customer, but **reduce** repair time). E.g., if repair time = 1 hour and cust still needs 10 min, then repair time gets reduced to 50 min and finishes at 60 min mark.
 - Wait (complete service of current cust and **delay** repair). E.g., if repair time = 1 hr and cust needs 10 min, then repair finishes at 70 min mark.
 - Preempt (stop service of current cust, but **complete service after the repair**). Repair stops at 60 min mark, cust finishes at 70 min mark.
- **Demo Time!**

Summary

This Time: How ironic! We
succeeded in learning how to
implement failures in Arena!

Next Time: A quick intro to the Blocks
template, which gives us a `kr_aAaZy`
number of new things to look at...
but we'll be very gentle about this.

Computer Simulation

Module 5: Arena

Dave Goldsman, Ph.D.

Professor

Stewart School of Industrial and Systems Engineering

The Blocks Template

Lesson Overview

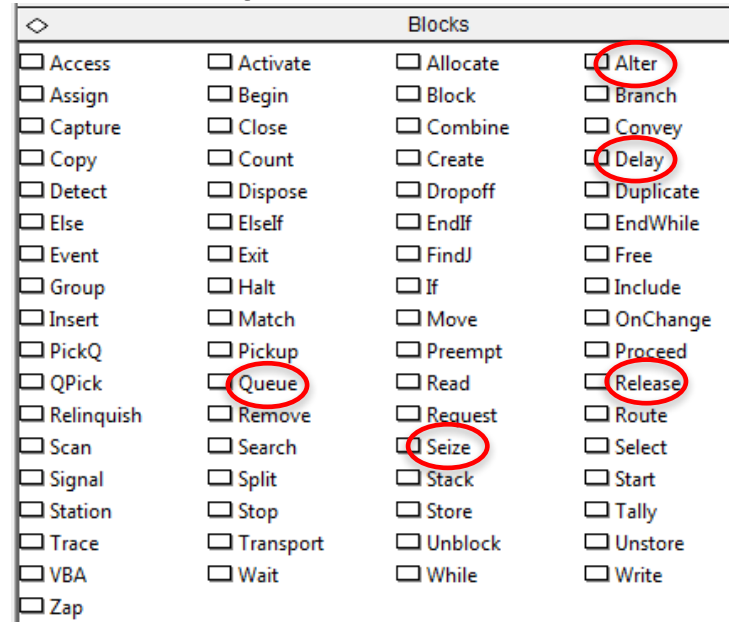
Last Lesson: Discussed the implementation of resource failures.

This Lesson: We'll add to our bag of tricks by introducing the Blocks template.

These are lots or blocks in this template, but don't be intimidated.

Blocks Template

- File > Template Panel > Attach (and hopefully you find it in the Rockwell Software \ Arena \ Template directory).
- Huge number of blocks!
- Do not be scared!
- They're sort of a self-contained language (related to SIMAN).
- Very specialized and low-level.
- Very useful, but we'll only need a few for now:
 - Seize, Delay, Release, Queue, Alter



Eh??

- Yikes! Why are we seeing stuff like Seize, Delay, Release **yet again!?**
At least two reasons...
 - Arena has been built in many layers over the years, and these are what's left from SIMAN – the “original” incarnation of Arena.
 - But in any case, and for whatever reason, certain primitive blocks such as the Queue block can't even connect to a Seize module from the Advanced Process template or a Process module from the Basic Process template. So you're stuck!
- **Demo Time!**
 - A Queue-Seize-Delay-Release example using primitive blocks (we'll see this again in the Call Center example coming up).
 - Using the Alter block to change the number of resources.

Summary

This Time: The Blocks template!
Later on, we'll find a number of interesting uses for some of these guys.

Next Time: The joy of sets! How to use certain vectorized sets to make modeling easier.