



Introduction to Arena

Dave Goldsman

Georgia Tech

Atlanta, GA, USA

sman@gatech.edu

www.isye.gatech.edu/~sman

(thanks to Seong-Hee Kim and Barry Nelson)



Overview

- We move to the design and analysis of dynamic systems that evolve over time.
- We'll use **Arena**, from Rockwell Software – one of several popular “discrete-event” simulation packages.
- Free Arena download available at <https://www.arenasimulation.com/academic/students>



Some Definitions

- A *system* is a collection of interacting entities (e.g., people, machines).
- A *model* is an abstract representation of a system, describing the system in terms of states, entities, sets, events, etc.
- The *system state* is a set of variables containing enough information to describe the system.
- An *entity* is an object or component explicitly represented in the model. Can be permanent (e.g., machine) or temporary (e.g., customers).
- *Attributes* are properties of entities (e.g., priority of a customer).
- An *event* is a point in time at which the system state changes. E.g., an arriving customer finding the server busy.
- The *simulation clock* is a variable whose value = simulated time.



Modeling Approaches

There are two modeling approaches for the design and analysis of dynamic systems that evolve through time.

- *Event-Scheduling Approach.* Concentrate on the events and how they affect the system state. We help the simulation evolve over time by keeping track of every event. This is a bookkeeping hassle. You might use this approach if you program in C++, Java, or FORTRAN.
- *Process-Interaction Approach.* We use this approach. Concentrate on a generic customer (entity) and the sequence of events and activities it undergoes as it progresses through the system. At any time, the system may have many entities interacting with each other as they compete for resources. You do the generic customer modeling in this approach, but the computer simulation language (e.g., **Arena**) handles the event scheduling and bookkeeping. Saves lots of programming effort.



How Does a Simulation Work?

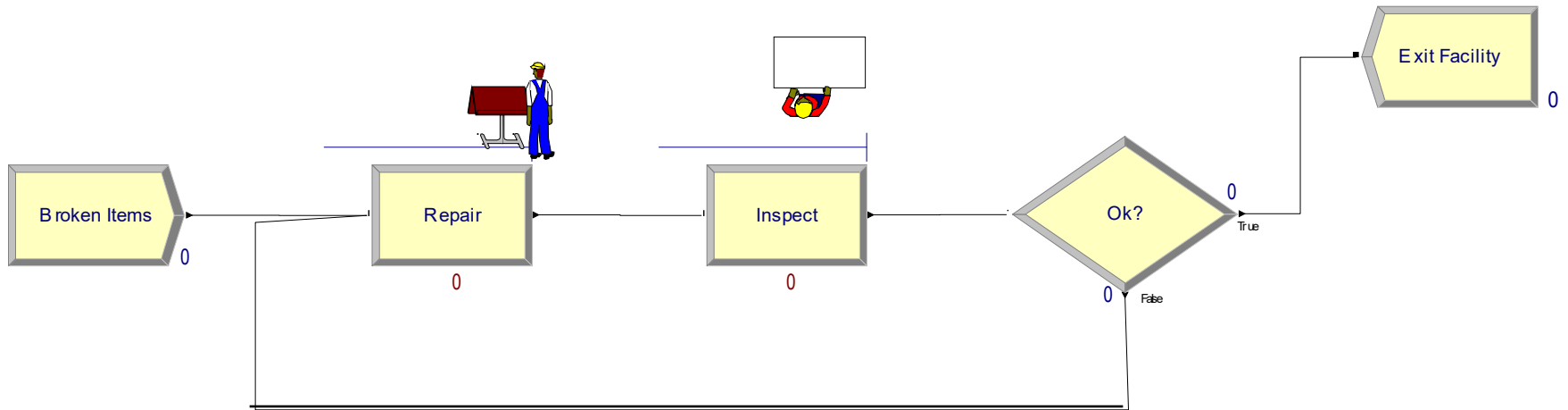
- Every simulation language maintains a **future events list (FEL)**. This is the list of all activities' scheduled times of completion — the list of all the events that we know about.
 - The FEL is a set of events ordered by time.
 - It can be updated any time an event occurs
 - The simulation proceeds by...
 - executing the next time-ordered event,
 - updating the FEL (if necessary), and
 - repeating
 - Arena uses the P-I approach and transparently maintains the FEL (so you don't have to).



Arena World View

- Arena takes the *process-interaction* world view.
- **Entities** flow through a **network** of **modules** that describe their logical behavior.
- We describe the network by developing a process **flowchart**.

Flowchart Approach





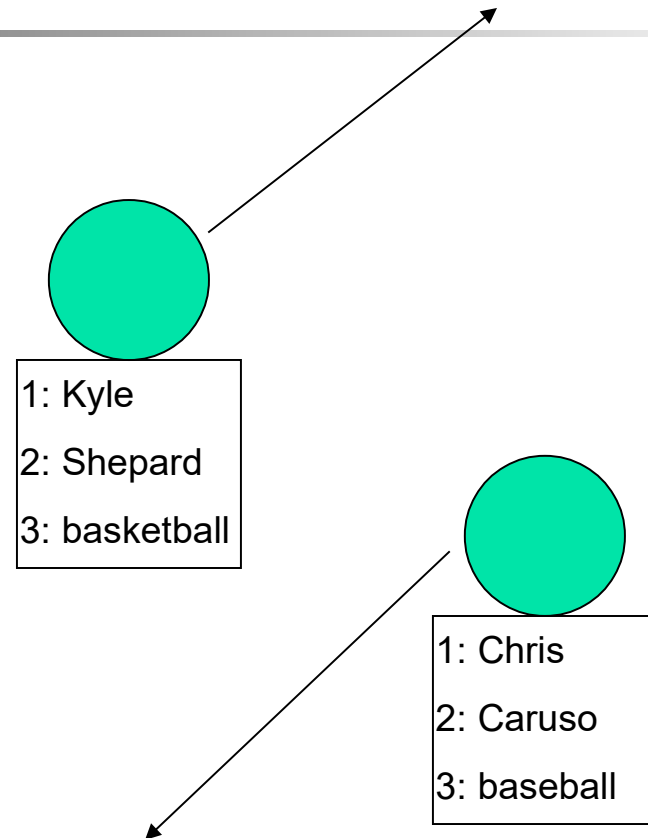
About modules...

- Arena contains a very large number of modules that are organized into *panels*.
- The panels are structured from high level to low level concepts:
 - Basic Process
 - Advanced Process & Advanced Transfer
 - Blocks & Elements (a programming language)
- Our goal is not to learn lots of modules, but rather to understand concepts that allow us to learn new modules as needed.



Entities

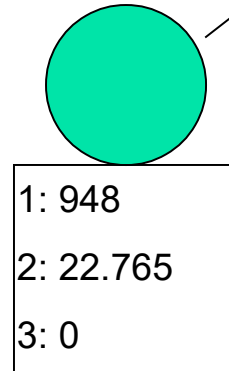
- *Entities* are dynamic elements that pass through the system.
- Entities are distinguished by their *attributes*.
- Ex: people, parts, information, paperwork, etc.





More on Entities

- Entities must be *Created* to get them into the model, and are *Disposed* when they leave.
- Unfortunately, attributes must be *numerical values*.





Queueing

- Entities queue when they need processing.
- In Arena...
 - An entity tries to *Seize* a *Resource*.
 - The time the entity uses the resource is the *Delay*.
 - If the resource is not available, the entity waits in a *Queue*.
 - The entity *Releases* the resource when processing is complete.



Resources

- Resources have...
 - A Name (up to you)
 - A Capacity (number of identical units of this resource; think # of servers).
 - And can have a Schedule (how many available when).
- And Resources can be animated.



More on Resources

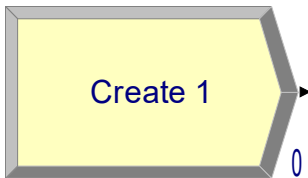
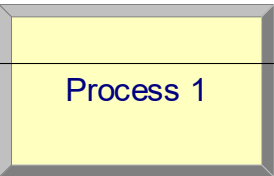
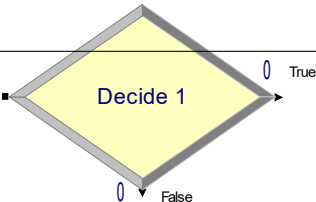
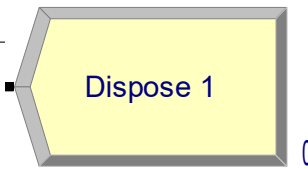
- Resources are automatically defined by some modules (e.g., Process)
- Resources can be defined manually, and the properties of all resources are changed, via the Resources spreadsheet on the Basic Process panel.
- There is also a Schedule spreadsheet for specifying Resource schedules.



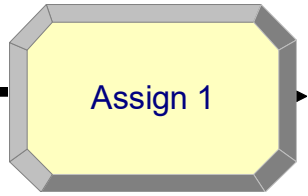

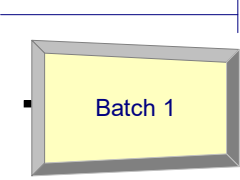
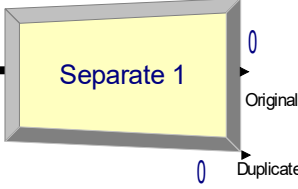
More on Queues

- Queues are created automatically by some modules (e.g., Process), and can be defined manually.
- Properties of a queue, including the ranking rule, are defined via the Queue spreadsheet.
 - First-in or Last-in first out
 - Lowest or Highest attribute value first

Basic Process Modules

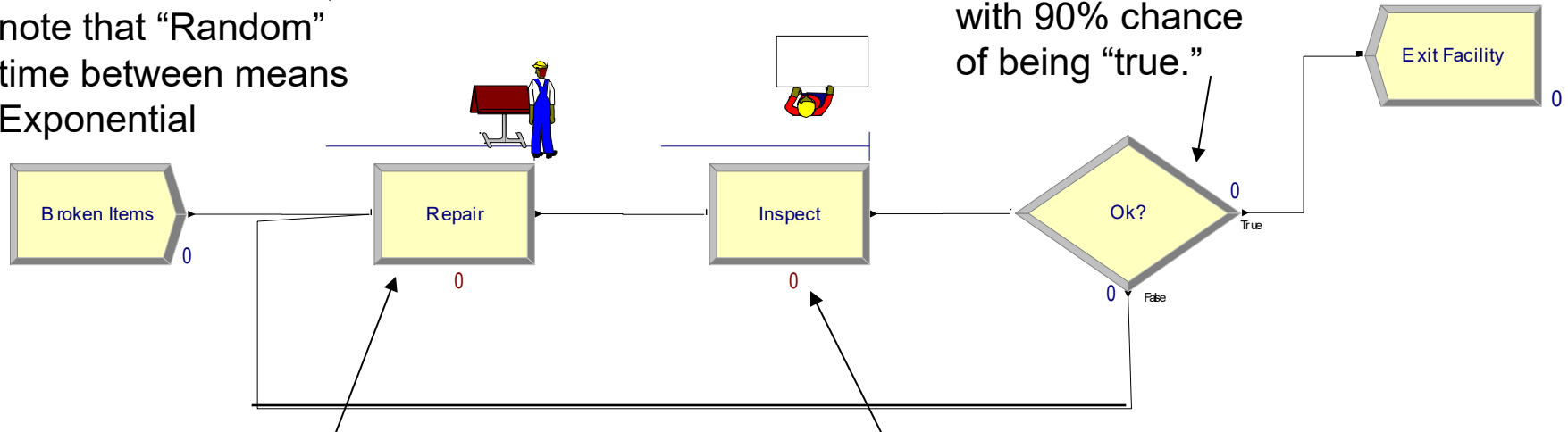
	<p>Push (possibly) batches of entities into the model with a (possibly) random time between.</p>
	<p>Models Queue-Seize-Delay-Release of Resource, or any part of this (like pure Delay).</p>
	<p>Make decisions about where to go next based on conditions or chance.</p>
	<p>Take entities out of the model and (perhaps) record statistics.</p>

Basic Process Modules

	<p>Assign values (especially Attributes) when an entity passes through.</p>
	<p>Record information when entities pass through, typically statistics on entities.</p>
	<p>Combine multiple entities into a single entity.</p>
	<p>Split multiple entities that were combined, or duplicate a single entity.</p>

Example

Create item entities;
note that “Random”
time between means
Exponential



The decision is “2-
way by chance”
with 90% chance
of being “true.”

Action is “Seize-Delay-
Release” to represent
a queue.

The delay can be given
by an expression, in this
case $\text{Expo}(0.125)$,
exponential with mean
0.125.



Basic Animation

- Entity movement (via module connections) and queues are automatically animated.
- The entity movement does **not** correspond to the passage of simulated time.
- Later we will learn how to animate transportation delays.



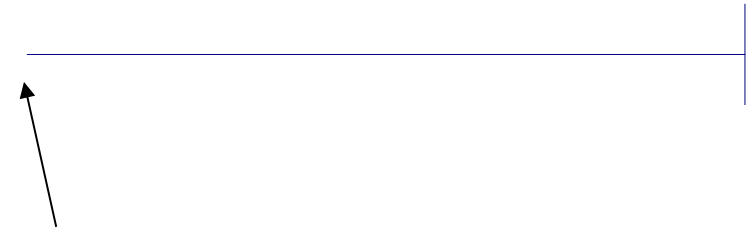
Entity Animation

- The Entity spreadsheet allows you to change the entity picture for each entity type.
- The Entity Type is a name, usually given when the entity is created.
 - Create: Entity Type: Items
- An Assign module can be used to change the entity Type or Picture as it moves through the model.



Queue Animation

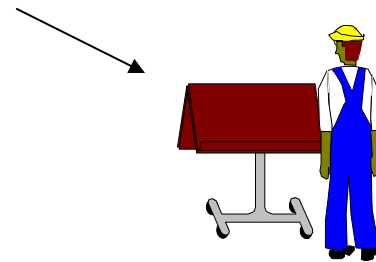
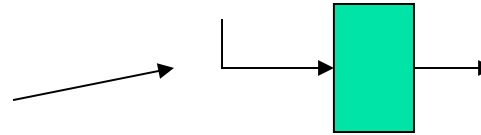
- The default queue is the sideways T.
- The queue symbol can be dragged anywhere, or reoriented.
- Often need to make the queue picture longer (which has no effect on queue capacity).



To lengthen the queue symbol, select it, grab the end, and pull.

Resource Animation

- Clicking the resource button lets you add a resource picture.
- You select pictures for the Busy, Idle, Inactive and Failed states.
- The Identifier must be the name of a resource already in the model (e.g., defined by a Process)





Delays

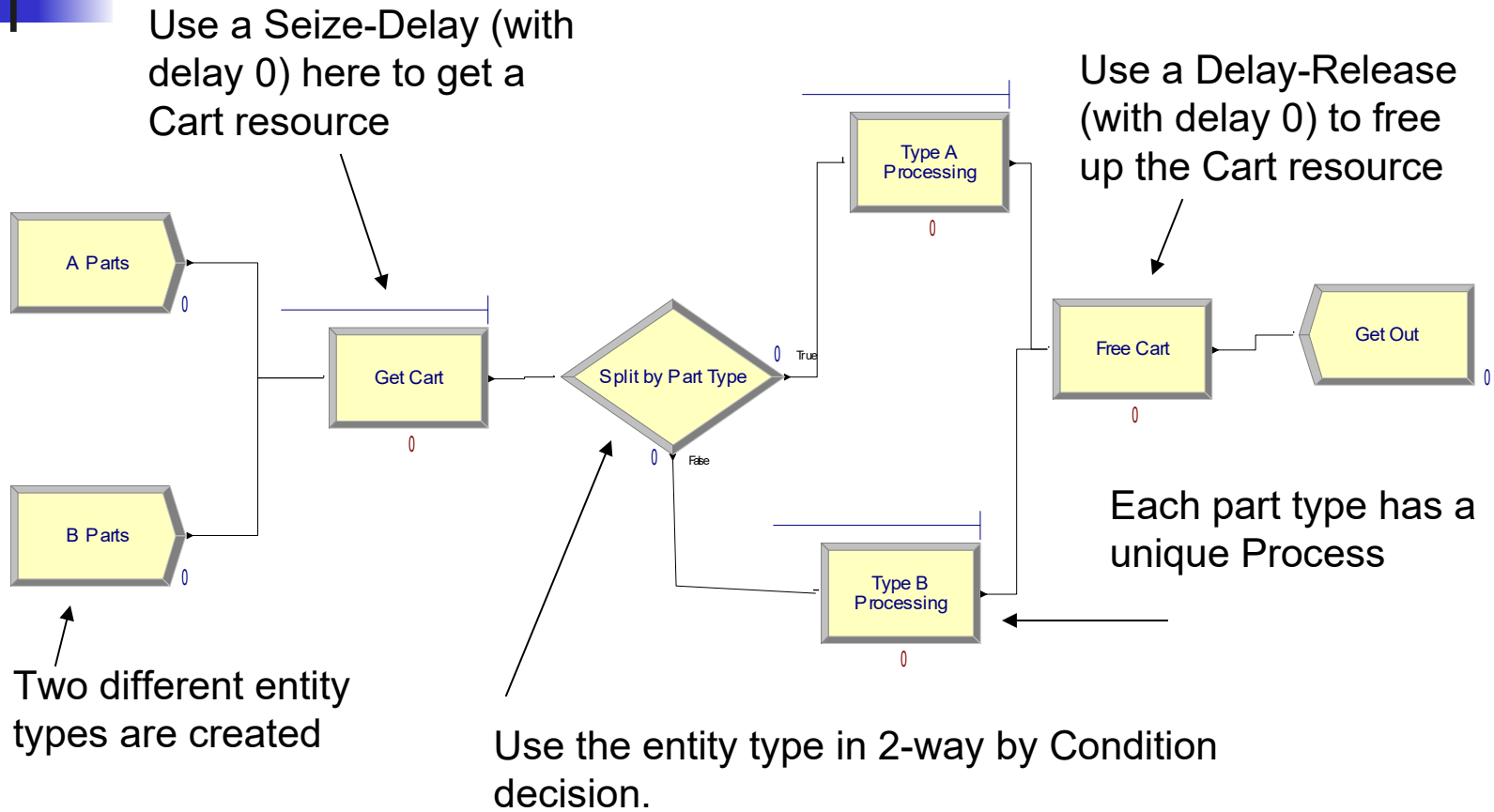
- Arena gives a default distribution for time between creations (“Random” = Expo) and delay (“Triangular”).
- If we want to put in a different distribution, we select “Expression” and enter the appropriate Arena function, such as WEIB, POIS, etc.
- We often get the expressions from the Input Analyzer.



Seize-Delay-Release

- Seize-Delay-Release need not be done in a single Process.
- One Process may be used to Queue and Seize the resource, a number of other modules may represent the processing, and yet another Process may finally Release the resource.

Example





Internal Variables

- Arena keeps a number of internal variables continually updated.
- These variables are useful for making choices in a Decide module, displaying in animated plots, or for recording statistics.
- The basic syntax is Name.Quantity

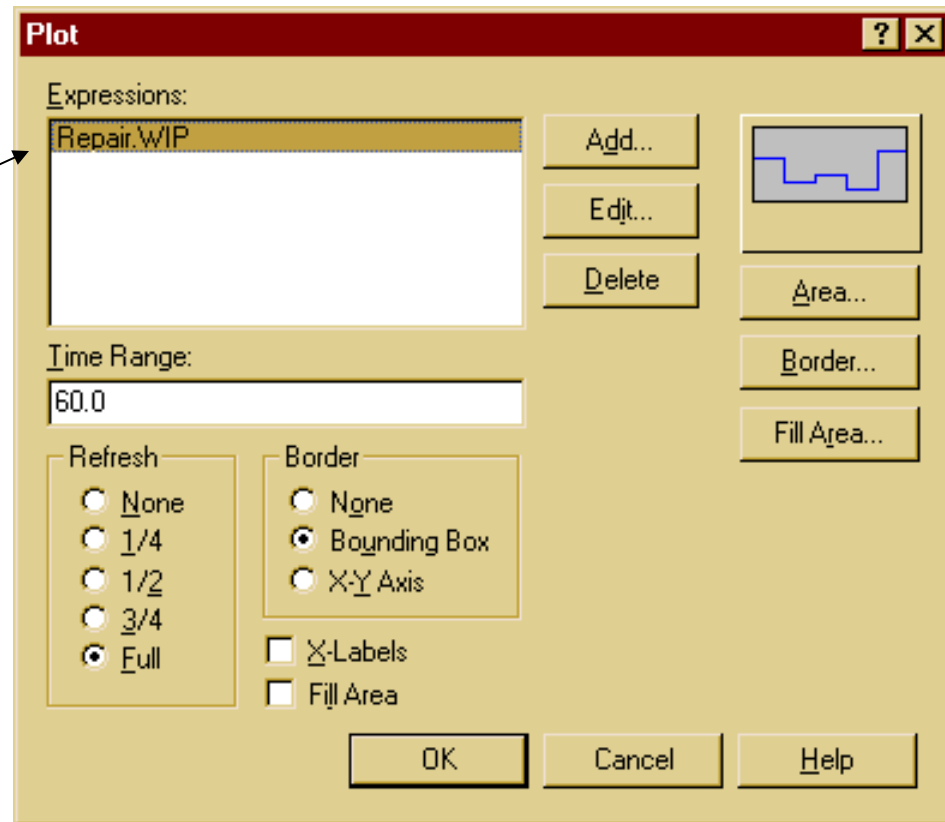


Basic Process Variables

- Create
Name.NumberOut
- Process
Name.NumberIn
Name.NumberOut
Name.WIP
Name.WaitTime
- Decide
Name.NumberOut True
Name.NumberOut False
- Assign
Name.NumberOut
- Batch
Name.NumberOut
- Separate
Name.NumberOut Orig
Name.NumberOut Dup
- Record
Name.NumberOut
- Dispose
Name.NumberOut

Example

We can use the internal variable **Repair.WIP** to create a dynamic plot of the number of parts at the Repair Process



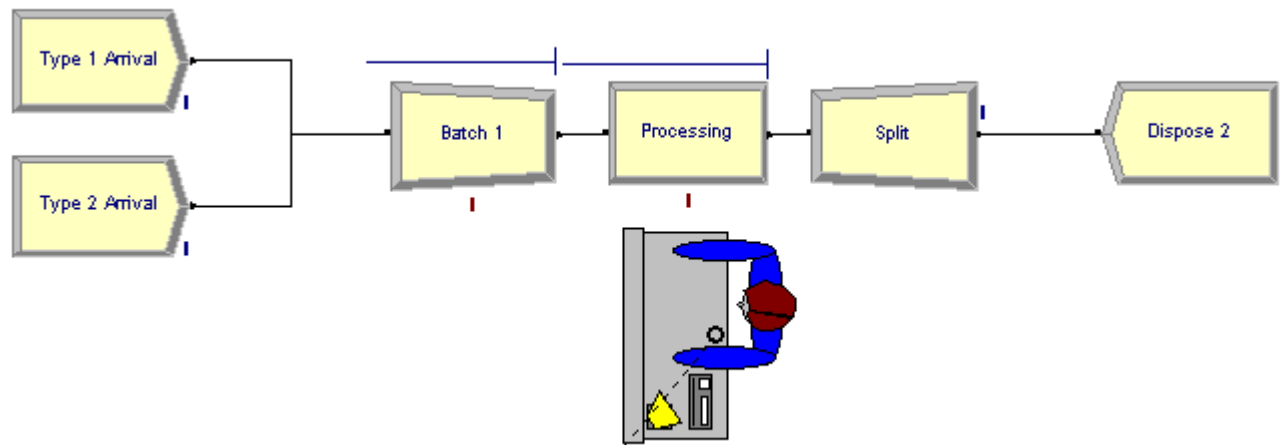


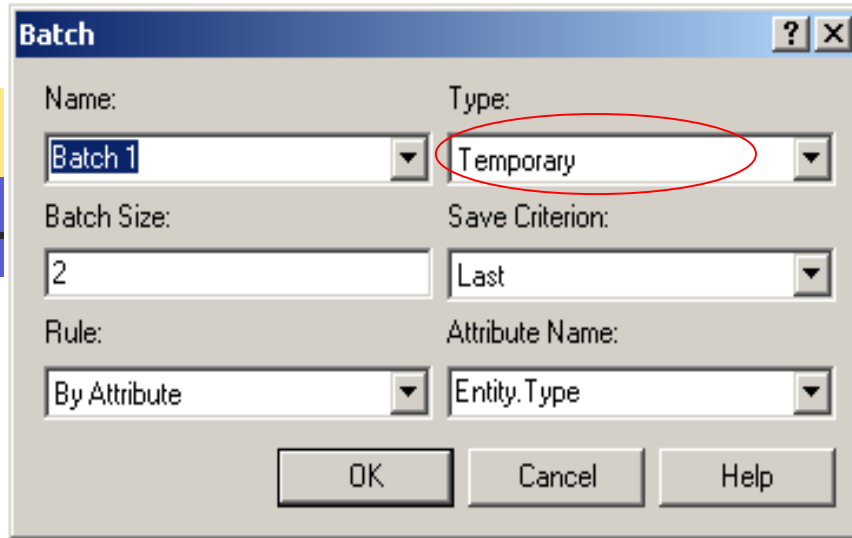

Simulated Time

- The simulation keeps its own internal clock that jumps forward from event time to event time.
- The time on the simulation clock is accessible through the Arena variable `TNOW`.
- `TNOW` is useful for marking entities or making time-based decisions.

Batch and Split

- Entities are processed in a batch of size two and then the system split entities that were combined.



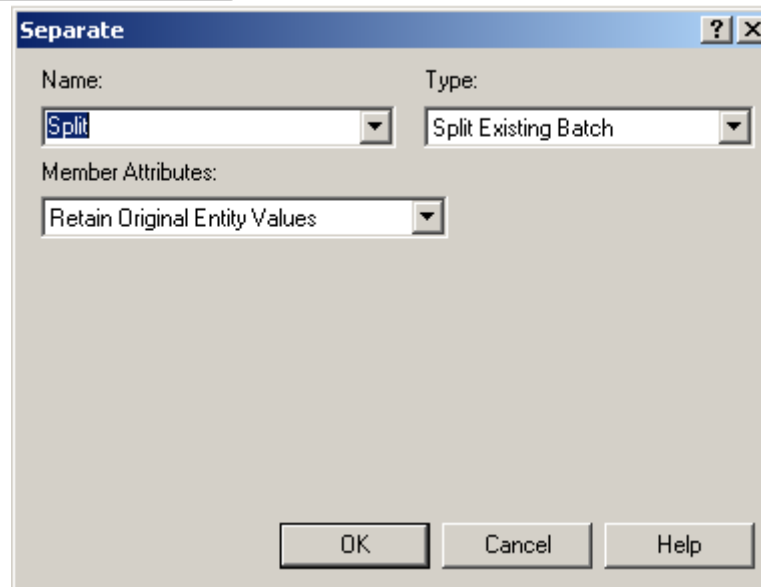


The 'Batch' dialog box contains the following fields and controls:

- Name:** A dropdown menu with 'Batch 1' selected.
- Type:** A dropdown menu with 'Temporary' selected, which is circled in red.
- Batch Size:** A text input field containing the number '2'.
- Save Criterion:** A dropdown menu with 'Last' selected.
- Rule:** A dropdown menu with 'By Attribute' selected.
- Attribute Name:** A dropdown menu with 'Entity.Type' selected.
- Buttons:** 'OK', 'Cancel', and 'Help' buttons at the bottom.

If one chooses "Permanent" as Type, batched entities will never be split.

Separate module can be used to generate a duplicate of an entity. An example will be shown when we discuss Advanced Input Modeling.



The 'Separate' dialog box contains the following fields and controls:

- Name:** A dropdown menu with 'Split' selected.
- Type:** A dropdown menu with 'Split Existing Batch' selected.
- Member Attributes:** A dropdown menu with 'Retain Original Entity Values' selected.
- Buttons:** 'OK', 'Cancel', and 'Help' buttons at the bottom.



Demos

- Single-Server Queue
- Multiple Arrival Streams
- Call Center
- Manufacturing Cell
- Immunization Clinic
- Pandemic Influenza Model