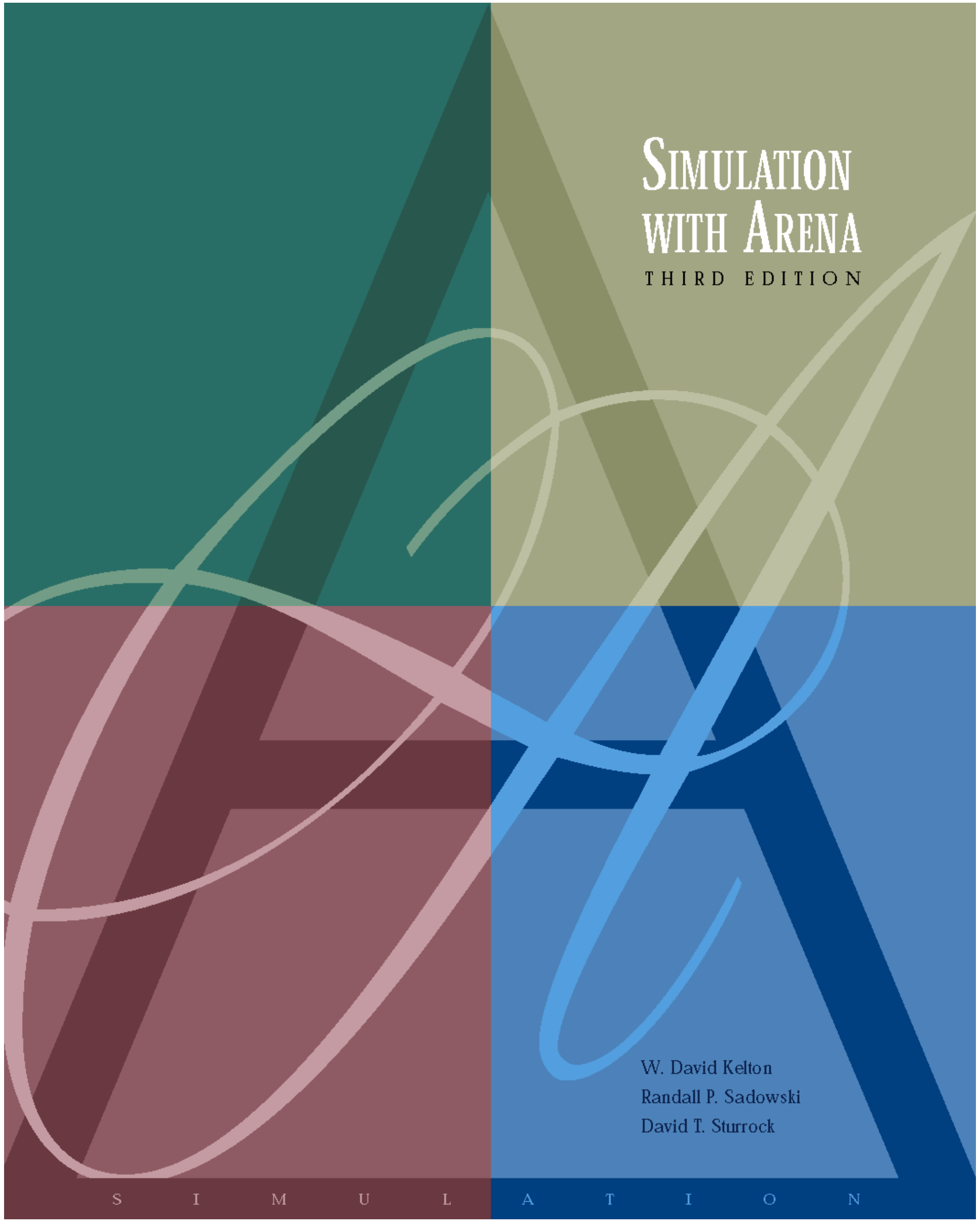


Modeling Basic Operations and Inputs

Chapter 4

Last revision June 7, 2003

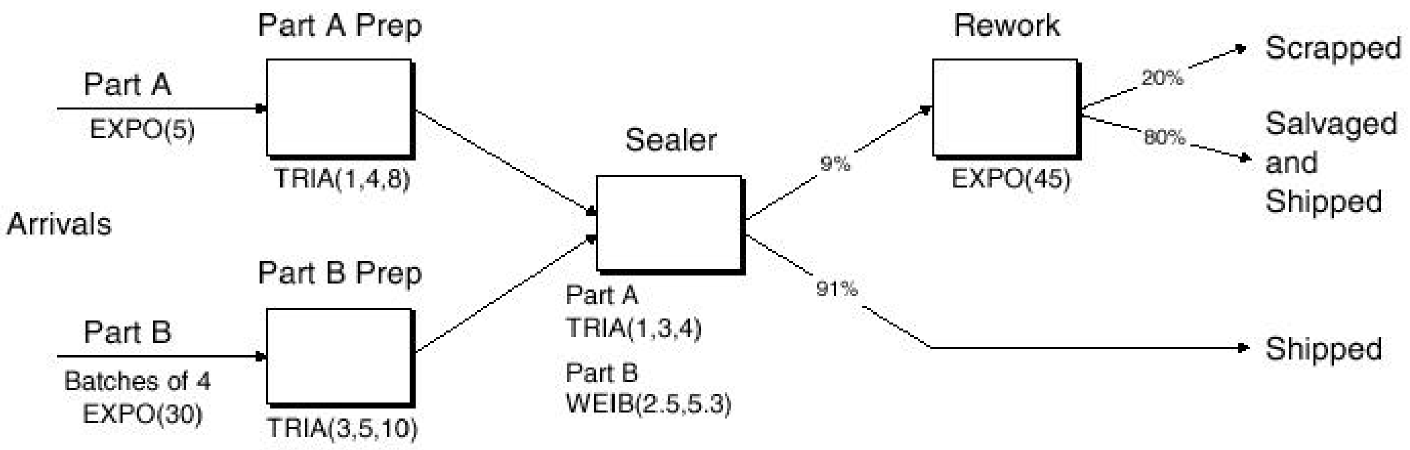
What We’ll Do ...

- **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)



- **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`**

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

- **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

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

Running the Model

- **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***

Change Run Conditions

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

Schedules

- **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”

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)**

Frequencies

- **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)

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

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

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

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 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

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 (?)
- ***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?

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)
- **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)

No Data?

- **Happens more often than you'd like**
- **No good solution; some (bad) options:**
  - Interview "experts"
    - Min, Max: Uniform
    - Avg., % error or absolute error: Uniform
    - Min, Mode, Max: Triangular

      Mode can be different from Mean — allows asymmetry
  - Interarrivals — independent, stationary
    - Exponential— still need some value for mean
  - Number of "random" events in an interval: Poisson
  - Sum of independent "pieces": normal
  - Product of independent "pieces": lognormal

Cautions on Using Normal Distributions

- **Probably most familiar distribution – normal "bell curve" used widely in statistical inference**
- **But it has infinite tails in both directions … in particular, has an infinite left tail so can always (theoretically) generate negative values**
  - Many simulation input quantities (e.g., time durations) must be positive to make sense – Arena truncates negatives to 0
- **If mean m is big relative to standard deviation s, then P(negative) value is small … one in a million**
- **But in simulation, *one in a million can happen***
- **Moral – avoid normal distribution as input model**

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
- **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**

Multivariate and Correlated Input Data

- **Usually we assume that all generated random observations across a simulation are independent (though from possibly different distributions)**
- **Sometimes this isn't true:**
  - A "difficult" part requires long processing in both the Prep and Sealer operations
  - This is positive correlation
- **Ignoring such relations can invalidate model**
- **See textbook for ideas, references**