

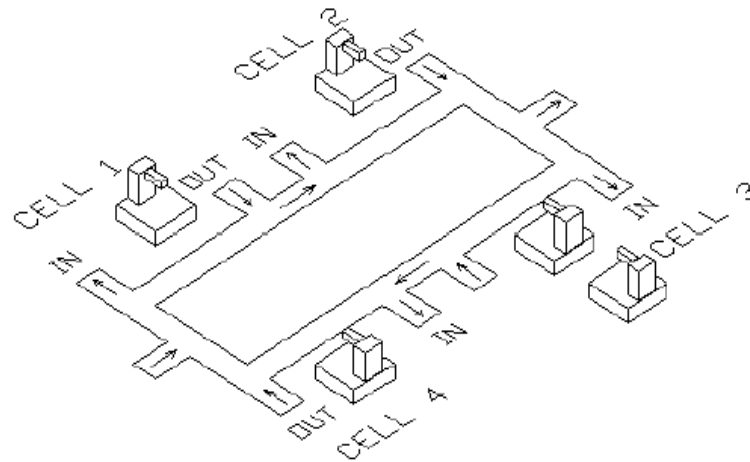


# Sequencing Lab

---

A Small Manufacturing System

## Model 7-1 Chapter 7



- 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?)

# Part Pathways

- 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		

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



# New Modelling 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



# Modelling approach

---

- 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 Module cont.

---

- 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





# Variables 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, and 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 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

---

- Run/Setup Dialog
  - Replication Parameters Tab
    - Replication Length = 32 Hours
    - 24 Hours/Day
    - Base Time Units = Minutes



# Part Arrivals

---

- Create module for arrival of one part
  - One-at-a-time, Time Between Arrivals is exponential with meant 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)**



# Entities in the 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



# 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)



## 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





## 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**