Single Queue Multi Server Model
Start with Single Server Queue

- Basic graphical layout for a single-server

- "Executive" block (Discrete Event.lix)
- "Generator" block (Discrete Event.lix)
- "Queue FIFO" block (Discrete Event.lix)
- "Exit" block (Discrete Event.lix)
- "Activity Delay" block (Discrete Event.lix)
- "Input Random" block (Generic.lix)
Add Activity Delay Blocks to Model Servers Individually

- Each Activity Delay block (*Discrete Event.lix*) pulls an item (if there is one available) from the Queue FIFO block (*Discrete Event.lix*) whenever it is idle.
- Text labels have been used to minimize connector clutter
- Configuration and results boxes have been cloned onto the model plane
Alternative: Use an Activity, Multiple Block

- If there is no need to model servers individually, an Activity Multiple block (Discrete Event.lix) can be used.

  - The number of servers can be set in the dialogue, or it can be set dynamically via the C input connector.
  - When the Activity, Multiple block pulls an item from the Queue FIFO block, it is set to delay by the amount on the D input.
    - Subsequently arriving items exit first if their delay value is sufficiently less.
Implementing Parallel Models

• Since the first example does not endow the servers with any special characteristics, the two examples are functionally equivalent

• To directly verify that the two models produce the same behavior each must get the same set of items to work on
  – the input item streams must have the same interarrival times

• Ways to do this:
  – Run the two models one after the other (serial) and have each Generator block produce the same set of items
  – Alternatively, run the two models in parallel, duplicating each item as it is generated, simultaneously sending item clones to each model
Serial Implementation

• Set the "block seed" in each Generator block's dialogue to be the same, causing the random number generator to start identically for each
  – Items for each run have the same interarrival times

• Set the "block seed" in each Input Random block's dialogue to be the same, causing the random number generator to start identically for each
  – Delay times used for corresponding items are identical
Parallel Implementation

• Implement both model paradigms within the same **Extend** model
• Use the Unbatch block (*Discrete Event.lix*) to produce multiple copies of each input item
  – Feed cloned items to each paradigm
• Set the "block seed" in the Input Random block for each paradigm dialogue to be the same so corresponding clones get the same delay time
Models in Parallel

- Each paradigm produces the same overall outcomes
There is a Batch Block

• Usually paired with an Unbatch Block
• Item attributes can be preserved
  – Dialogue specification