Process Playground User's Manual

Revised 12/26/2011

Contents

Table of Figures

Introduction

Welcome to MoreSteam.com's Process Playground! Process Playground (PP) is a web-based discrete event simulation program that allows you to create virtual models of your processes. Once you have built a model of a process, there are many avenues of exploration available:

- What if the demand level increases or decreases – can my process handle the change?
- What if variation in demand or service time changes – what will happen?
- I need to improve the lead time of the process – what should I change first?

These, and many more questions, can be investigated and answered with a simulation model of your process. But what if you're starting from scratch and designing a brand new process? Creating a simulation model of your new process, or of several alternative new processes, will allow you to perform an infinite variety of virtual pilot runs before you even begin committing resources to the new process. Simulation allows you to test the new process against large ranges of demand and variation, and experiment with various failure modes before they happen. Imagine a Failure Mode and Effects Analysis that is 90% complete before the first live pilot.

With Process Playground, we have developed an easy to learn, easy to use, and easy to install (you don't have to install anything!) web-based simulator. Most traditional simulation programs take a boil-the-ocean approach. They have added a lot of complexity that allows you to model almost every nuance of any process. The result is often a long learning curve before you can build the models you need.

We have taken a simpler approach Process Playground. We analyzed the tools needed to model almost all of the typical processes which might be the subject of a continuous improvement project. We included those tools, and nothing more, into Process Playground. There are no hidden menus or dialog boxes – everything you can do is right there on the screen. Our students usually need little more than a day to learn everything about Process Playground.

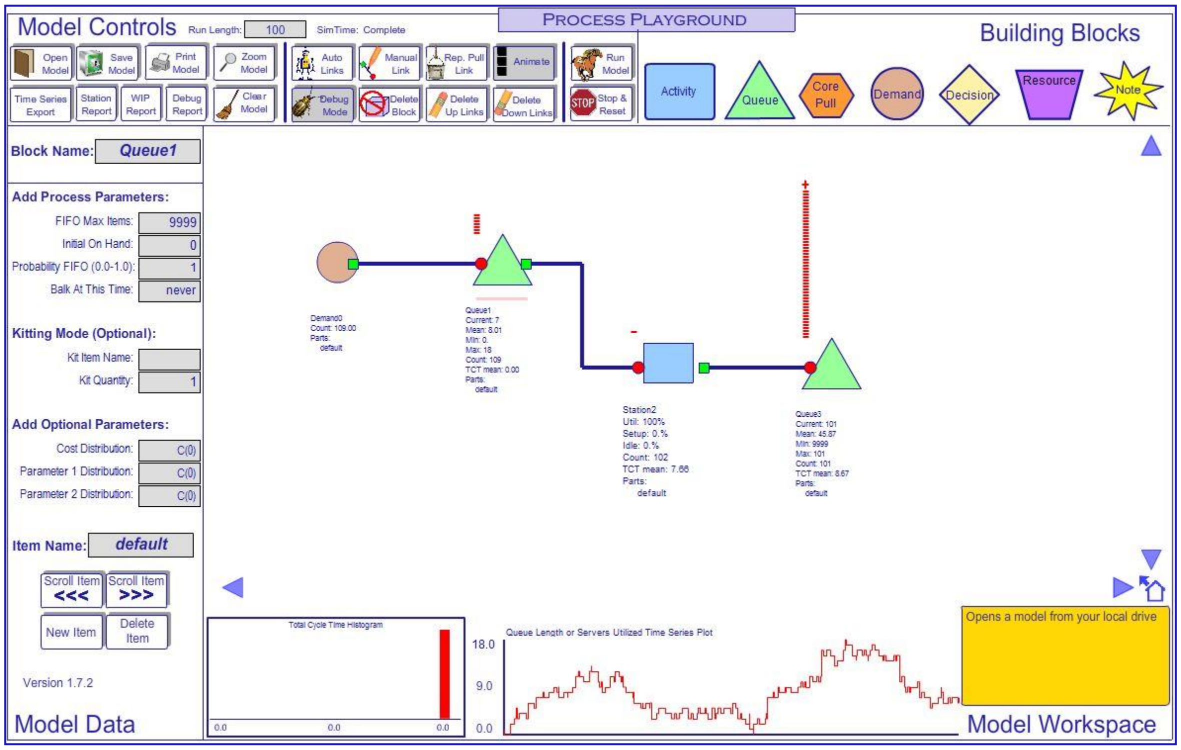
Because Process Playground is web-based, you never need to worry about installation, updates or compatibility. The program runs in any web browser that supports Adobe Flash Player version 10.1 or higher. Your model files are stored locally on your computer, so you have complete control over them. The model files are also very small (typically less than 100 kilobytes) so they are quick and easy to share as well.

This manual will walk you through all of the features and functions of Process Playground, as well as giving you tips on good modeling techniques. There are also training videos that are associated with each of the chapters in this manual, as well as additional features of Process Playground not covered in this manual. The training videos and sample models are also available from the Process Playground launch page.

User Interface Overview

The Process Playground screen is split into 4 main sections. Each section groups together similar functions or actions needed for the program.

- Model Controls: The top left section includes the model controls. This section itself is split into three sub-sections which provide the controls (from left to right) for **managing** model files, **building** models, and finally **running** models in the buttons at the right of the section.
- Building Blocks: The top right section includes the 7 building blocks available to model your processes. These 7 building blocks provide all of the modeling functions available in Process Playground.
- Model Data: The left side of the screen holds data related to the currently selected block. You can tell which block is selected because it will have a red, flashing bar below the icon (Queue 1 in the figure). Each building block will have different types of data displayed in the Model Data section.
- Model Workspace: The last section is the main workspace for Process Playground.



Block Name: Queue1

Add Process Parameters:
FIFO Max Items: 9999
Initial On Hand: 0
Probability FIFO (0.0-1.0): 1
Balk At This Time: never

Kitting Mode (Optional):
Kit Item Name:
Kit Quantity: 1

Add Optional Parameters:
Cost Distribution: C(0)
Parameter 1 Distribution: C(0)
Parameter 2 Distribution: C(0)

Item Name: default

Demand0
Count: 109.00
Parts:
default

Queue1
Current: 7
Mean: 8.01
Min: 0.
Max: 18
Count: 109
TCT mean: 0.00
Parts:
default

Station2
Util: 100%
Setup: 0.%
Idle: 0.%
Count: 102
TCT mean: 7.66
Parts:
default

Queue3
Current: 101
Mean: 45.87
Min: 9999
Max: 101
Count: 101
TCT mean: 8.67
Parts:
default

Figure 1 - Process Playground

This is where you will drag and drop building blocks to create your process model.

In order to help you learn about Process Playground and process modeling at the same time, this guide will follow the tried-and-true method of learning-by-doing. As you progress through this tutorial, you will build your own library of reference models. The reference models will cover every feature and option of Process Playground, allowing you to become completely familiar with Process Playground because you will have recreated every model in this manual.

Please create a directory on your computer to store your models as you go. If you name the models to match the names used in this manual, you will be able to build a reference library to help with future modeling projects.

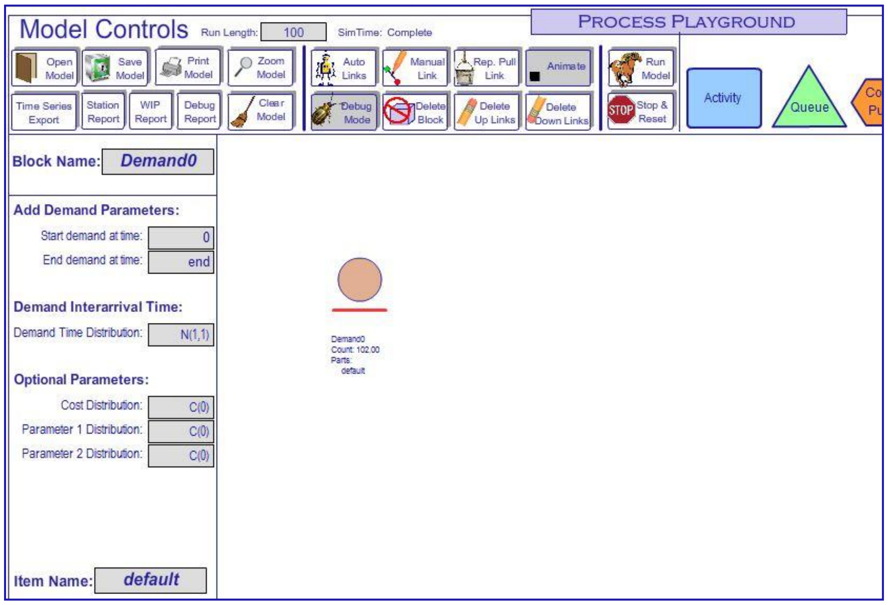
Ready to go? Let's begin with the basics…

Model 1 – The Basic Model

Your first model will be a simple model that consists of a demand block, two queues, and an activity. With this basic model you will learn most of what you need to know to model simple business processes.

Building models with Process Playground is done by dragging the building block icons from the Building Block section to the workspace. Start by clicking on the demand icon and dragging into the workspace. You can drag it anywhere you would like to, but it makes the most sense to put it at the left side of the workspace since the demand block will be the beginning of the process. When you are done, you should have something that looks like the picture to the right.

Notice that the demand block has a red flashing bar under it? That bar is critical because it indicates which block in the model is selected.



Run Length: 100 SimTime: Complete

Block Name: Demand0

Add Demand Parameters:

Start demand at time: 0

End demand at time: end

Demand Interarrival Time:

Demand Time Distribution: N(1,1)

Optional Parameters:

Cost Distribution: C(0)

Parameter 1 Distribution: C(0)

Parameter 2 Distribution: C(0)

Item Name: default

Demand0
Count: 102.00
Parts:
default

Figure 2 - Demand Block of the Basic Model

When a block is selected, the Model Data section changes to display the data associated with the selected block. In this case, you can see that the data displayed is related to the demand block.

Demand Block Data

The demand block data consists of four groups of data:

- The block name: At the top of the Model Data section you will see the default block name that is assigned when you first drag the demand block into your model. In this case, the default name is “Demand0”. You can click or double-click on the block name and change it to a name that makes more sense for your model. Note that your changes to the block name are reflected in the text under the demand icon.
- The distribution: Process Playground supports six different mathematical distributions for use in your model. You can select your distribution type by entering the appropriate character for each of the distribution types in the space provided in the data section. As you make your selection, the appropriate numerical parameter fields will become available for you to enter the data needed by the distribution. The default type is “C”,

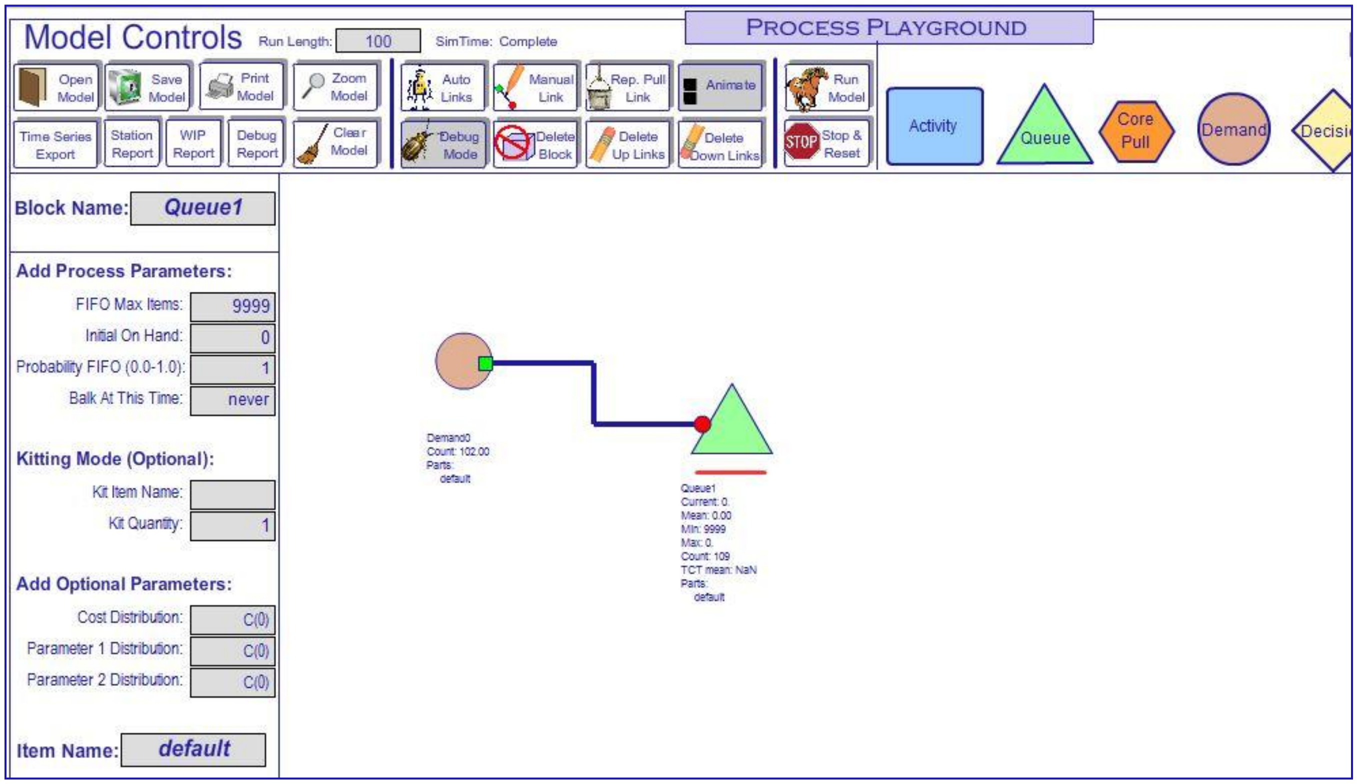
representing a constant. More information on the available distributions can be found in the later section “Random Distributions”.

- Start and End time: With the demand block, you can define when the block will begin to produce demand and when it will stop producing demand. The default values of zero and -1 indicate that the demand will generated for the entire simulation run. If you change the end time of -1 to any positive value, that value will define the time when the demand block will stop generating demand during the simulation run. The start time defines when the demand block will begin generating demand. If you set the start time to a value higher than the end time, the block will generate 1 demand event at the start time you entered and then stop producing any demand.
- Item Name: The default item name is “default”. A powerful feature of Process Playground is the ability to create many items within each block. These items can have the same name or different names, and Process Playground will automatically route the items in the model based on their names. For now, just leave the Item Name as “default”. You will learn more about this powerful feature in later sections.

The next element of your first model will be a queue block. The demand block should always have a queue block following it or you may risk losing the demand in the model. If a demand block is connected directly to an activity block, and that activity block is busy when the demand arrives, the demand will be ignored and lost in the model. Therefore, it is always good practice to have a demand block flow directly into a queue block which can accumulate demand as needed.

Process Playground has a feature called Auto Links which helps speed model building. The on/off button for Auto Links is located in the center section of the Model Controls area of the screen. With Auto Links turned on, Process Playground will automatically connect the next block you drag into the workspace with the block that is currently selected. Remember, you can see which block is selected by looking for the flashing red line below the icon in the workspace.

Since the Demand0 block is the only block in the workspace, it should still be selected and highlighted by the flashing red line. Now drag and drop a queue block somewhere to the right of the demand block. You should have a picture that looks something like the picture below.



Run Length: 100 SimTime: Complete

Block Name: Queue1

Add Process Parameters:
FIFO Max Items: 9999
Initial On Hand: 0
Probability FIFO (0.0-1.0): 1
Balk At This Time: never

Kitting Mode (Optional):
Kit Item Name:
Kit Quantity: 1

Add Optional Parameters:
Cost Distribution: C(0)
Parameter 1 Distribution: C(0)
Parameter 2 Distribution: C(0)

Item Name: default

```
Demand0
Count: 102.00
Parts:
   default
```

```
Queue1
Current: 0.
Mean: 0.00
Min: 9999
Max: 0.
Count: 109
TCT mean: NaN
Parts:
   default
```

Figure 3 - Queue Block of the Basic Model

If Auto Links is turned on, then Process Playground automatically connects the queue block to the existing demand block, and the queue block becomes the selected block when it is dragged into the model workspace. Notice that the flashing red line is now under the queue block. Also notice that the Model Data area has been updated to reflect the data related to the new queue block.

Queue Block Data

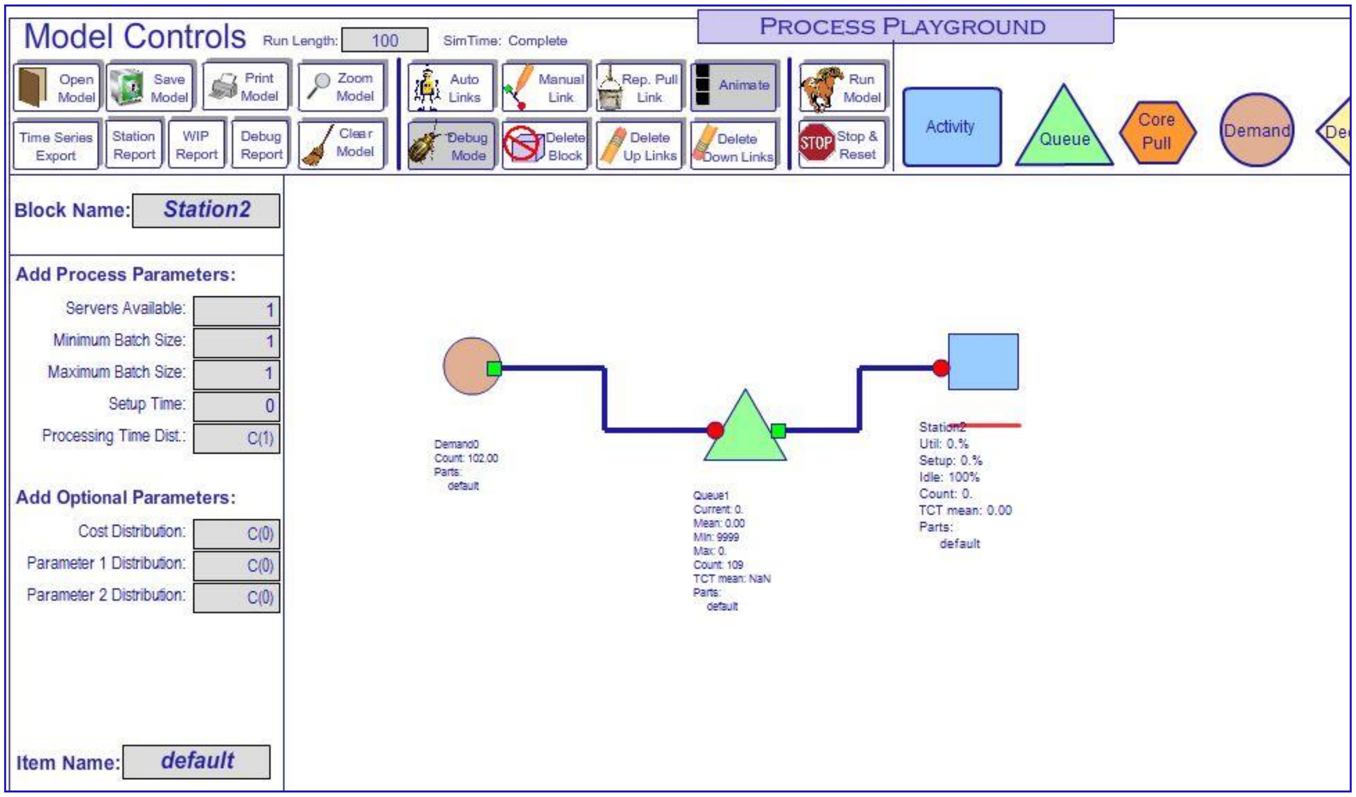
The queue block data consists of several different pieces of data which define how the queue will behave. Some of the options will be covered in later sections, but here are the basics:

- Block Name: Just as with the demand block, you can change the default name to another name that is more representative of your process.
- FIFO Max Items: If your real queue has an upper limit to the number of items it can hold, you can enter that limit here. All queues operate as FIFO queues unless you change another parameter (discussed later in this list).
- Initial On Hand: You can easily start your process with existing Work-in-Process (WIP) by placing initial on hand inventory into the queues in your process.

- The next three data items, all starting with “Repl. Pull”, are used when the queue is converted to a replenishment pull system buffer. This topic will be covered in a later section.
- Current On Hand: You cannot edit this field, but during and after a simulation run, you can review the current on hand value of each unique item in the queue by checking this number.
- Current On Order: This field is also related to the replenishment pull system operation, and will allow you to watch the amount of inventory on order change during a simulation run.
- Kit Quantity: This field is used when the queue is transformed into a kitting queue. As a kitting queue, the queue will pull a mixture of items into the queue, and then output the items as a single assembly or new item. This will be covered in more detail in later sections.
- Kit Item Name: A text entry in this field transforms the queue to a kitting queue. This topic will be covered in its own section later.
- Probability FIFO: This field allows you to disrupt the normal First-In-First-Out operation of the queue. The default value is 1, or 100%, which indicates that the queue will operate as a FIFO queue 100% of the time. If you enter a 0.5, that will force the queue to operate as a FIFO queue 50% of the time, and the other 50% of the time the queue will randomly pull an item from somewhere other than the front of the queue. You can enter values between zero and one in this field. If you enter zero, the queue will never allow the first in line to leave the queue unless that item is the only item in the queue.
- Item Name: Just as with the demand block, the queue block allows multiple items to be tracked in the queue. For now, leave the item name as “default”. More options for using the Item Name field will be covered in later sections.

The next element of your first model will be an activity block. This is the block that actually represents work or a task being performed in your process. Make sure that Auto Links is still turned on and that the queue block is selected (the flashing red line should be below the queue block).

Now drag and drop the rectangular blue activity block into the model. Place it somewhere to the right of your queue block. Once you have created the activity block in your model, you should see the Model Data section change to reflect the data options for the Activity block. Your model should look like the next picture.



Run Length: 100 SimTime: Complete

Block Name: Station2

Add Process Parameters:

Servers Available:	1
Minimum Batch Size:	1
Maximum Batch Size:	1
Setup Time:	0
Processing Time Dist.:	C(1)

Add Optional Parameters:

Cost Distribution:	C(0)
Parameter 1 Distribution:	C(0)
Parameter 2 Distribution:	C(0)

Item Name: default

```
Demand0
Count: 102.00
Parts:
   default
```

```
Queue1
Current: 0.
Mean: 0.00
Min: 9999
Max: 0.
Count: 109
TCT mean: NaN
Parts:
   default
```

```
Station2
Util: 0.%
Setup: 0.%
Idle: 100%
Count: 0.
TCT mean: 0.00
Parts:
   default
```

Figure 4 - Activity Block of the Basic Model

Activity Block Data

The activity block data consists of several different pieces of data which define how the queue will behave. Some of the options will be covered in later sections, but here are the basics:

- Block Name: As with the other blocks, you can rename the activity block as desired.
- Distribution Parameters: The selection of distribution type and the distribution parameters are identical to the selections in the demand block. Once you select a distribution type, the required parameter fields will become available for you to enter your data.
- Servers Available: Each activity can operate as a single server (person or machine) or as multiple servers. For instance, if you have 3 ATM machines at a bank, you can model them as a single activity with 3 servers.
- Setup Time: Each individual item entering the activity will incur a service (delay) time based on the distribution selected. However, each batch (including a batch of 1) will also incur a setup time if you enter a value here.
- Minimum and Maximum Batch Size: the default values for the minimum and maximum batch size are one. If you change these to values greater than one, the activity block will attempt to pull the largest batch possible when it becomes available for work. In order to operate in batch mode, the activity block must have a queue block in front of it

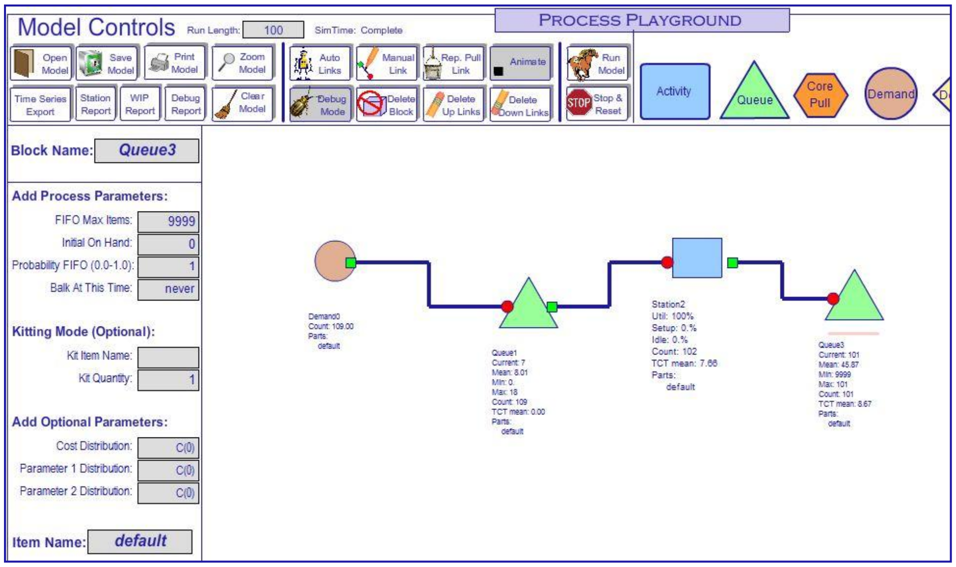
to accumulate items. The accumulated items will then be batched together and sent to the activity for work. The activity block must also have a queue attached behind it (downstream) in the process to accept the batch when it is completed. The downstream queue will break the batch apart and hold the items as individual pieces for the next activity. The next activity may have its own rules for batch sizes.

- Item Name: As with the demand and the queue blocks, the activity block supports multiple items. The default item, named “default”, tells the activity block that it can process any item passed to it. If you change “default” to anything else, the activity block will be limited to processing only those types of parts. For example, if you change “default” to “Part A”, then the activity will only be able to process items named “Part A”. More on this topic in a later section. For now, make sure to leave the item named “default”.

The Exit Queue

All models will need a final queue attached to the last step of the model. This queue will act as an exit to accumulate all of the finished items that have made it through the process. Without an exit queue, the last activity would have nowhere to send completed items and the activity would quickly become blocked, stopping the flow of items to the activity.

Go ahead and add an exit queue to your model. Be sure that the activity block is selected, with the flashing red line below it, before you drag in the exit queue. If you make a mistake and the exit queue is attached to another block in the model, jump to the next section to learn all about managing links in your model. Then come back here to learn how to run the model. Your model should look like the picture to the right.



Run Length: 100
SimTime: Complete

Block Name:	Queue3
Add Process Parameters:	
FIFO Max Items:	9999
Initial On Hand:	0
Probability FIFO (0.0-1.0):	1
Balk At This Time:	never
Kitting Mode (Optional):	
Kit Item Name:	
Kit Quantity:	1
Add Optional Parameters:	
Cost Distribution:	C(0)
Parameter 1 Distribution:	C(0)
Parameter 2 Distribution:	C(0)
Item Name:	default

Demand0
Count: 109.00
Parts:
default

Queue1
Current: 7
Mean: 8.01
Min: 0.
Max: 18
Count: 109
TCT mean: 0.00
Parts:
default

Station2
Util: 100%
Setup: 0.%
Idle: 0.%
Count: 102
TCT mean: 7.66
Parts:
default

Queue3
Current: 101
Mean: 45.87
Min: 9999
Max: 101
Count: 101
TCT mean: 8.67
Parts:
default

Figure 5 - The Exit Queue of the Basic Model

Running the Model

The time has come to run your first model in Process Playground. Running the model is simple, and there are only a few things you need to consider before the first run. The first thing you need to consider is how long to run the model. Are you modeling a restaurant? If so, you may need to run the model for the length of the work day. Or are you just modeling a peak demand period for a call center. If so, then define your run length to match the length of the period you wish to model. The length of the run is determined by the number you enter in the "Run Length" field at the top of the Model Controls section. There is no specification of the time period in Process Playground. That is left for you to interpret for yourself. Just make sure you are consistent. If you decide that each time period should represent 1 minute, then make sure all of your demand and service times in each demand and activity block are entered in minutes. You can define the time period to be anything you want – a second, a minute, an hour, a day – whatever you want. Just be consistent and use that time period throughout your entire model.

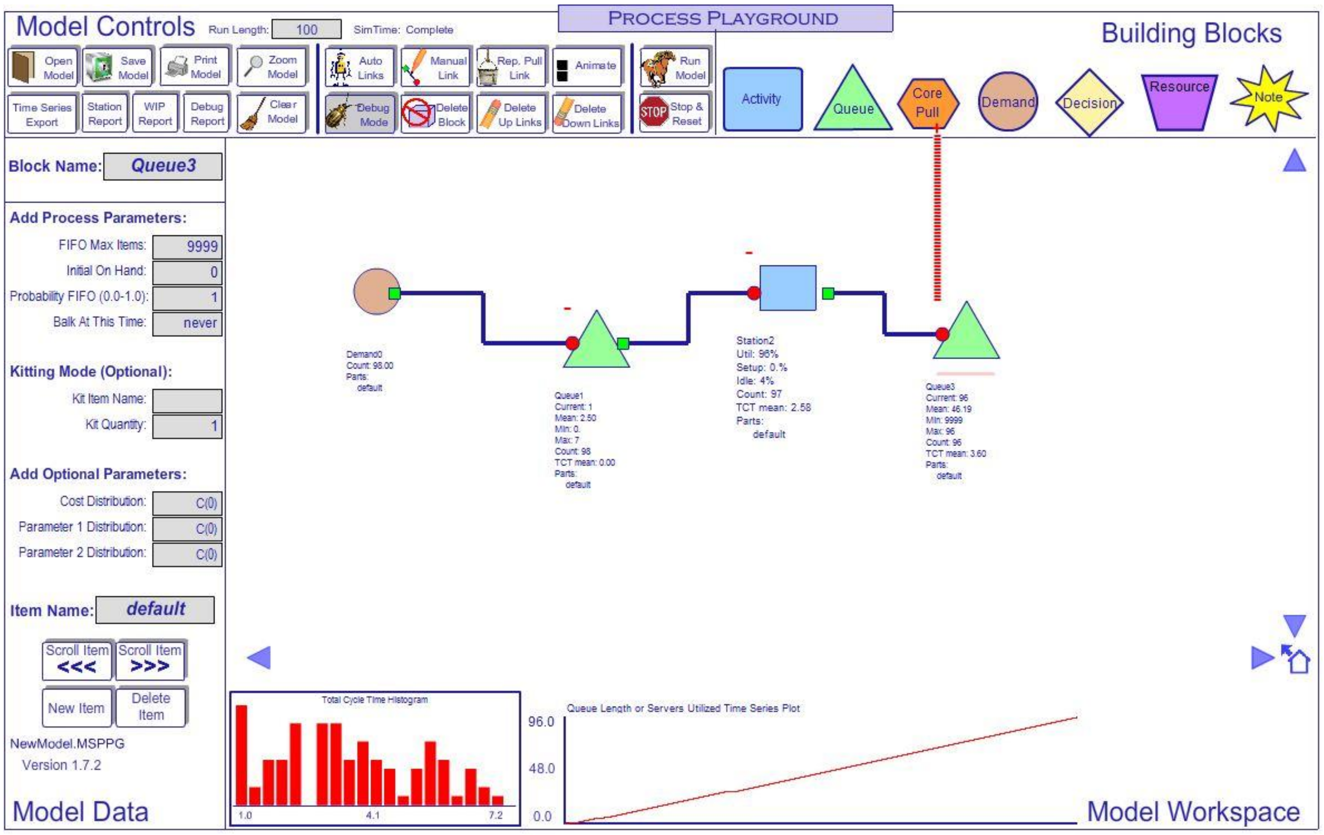
The buttons at the right side of the Model Controls are related to running the model. The functions are pretty straight forward:

- Run Model: This button starts the simulation run. This button also acts as the pause button during the run. The color of the button will tell you the state of the model:
  - White – the simulation is not running and ready to be started.
  - Green – the simulation is running and you should see the simulation time value in the "Sim Time" box just above the buttons growing as the model runs.
  - Red – the simulation is paused. Once you start the simulation, you may pause it by click the start button again. The button will change from green to red. Click it again to restart the run and the button will change back to green.
- Stop & Reset: If you need to stop a simulation run, just click this button to stop the run and reset the model.
- Debug Mode and Single Step: These two buttons allow you to step your way through a model and see each event before it happens. This is a fairly advanced topic that is covered later in this manual.

The default run length is 100 time periods and that will be fine for your first run. Go ahead and click Run Model to start the simulation. The default mode of operation is with animation turned on. The animation button is just to the left of the debug button. If it is white, then animation is turned on. If it is grey, then animation is turned off.

With animation turned on, you should see small black squares moving from block to block in the model. Each black square represents one piece of work-in-process. In this first, basic model the WIP will flow right through the model and pile up in the exit queue. When more than 50 pieces of WIP are present in a station no more black squares will be added and a black plus "+" sign will appear. If you turn off animation you will see that the model runs much more quickly. Animation can be helpful for testing models and presentations, but it takes a lot of computer resources and runs much more slowly.

After you have run the model, your screen should look something like the picture below. Make sure you have clicked on the exit queue to see the same graphs as shown in the picture.



Run Length: 100
SimTime: Complete

Block Name: Queue3

Add Process Parameters:
FIFO Max Items: 9999
Initial On Hand: 0
Probability FIFO (0.0-1.0): 1
Balk At This Time: never

Kitting Mode (Optional):
Kit Item Name:
Kit Quantity: 1

Add Optional Parameters:
Cost Distribution: C(0)
Parameter 1 Distribution: C(0)
Parameter 2 Distribution: C(0)

Item Name: default

NewModel.MSPPG
Version 1.7.2

Demand0
Count: 98.00
Parts:
default

Queue1
Current: 1
Mean: 2.50
Min: 0.
Max: 7
Count: 98
TCT mean: 0.00
Parts:
default

Station2
Util: 96%
Setup: 0.%
Idle: 4%
Count: 97
TCT mean: 2.58
Parts:
default

Queue3
Current: 96
Mean: 46.19
Min: 9999
Max: 96
Count: 96
TCT mean: 3.60
Parts:
default

Figure 6 - The First Run of the Basic Model

Now we will take a closer look at all of the items on the screen. First of all, notice the large pile of black squares above the exit queue, and the black plus sign. This shows that all of the WIP made it through the process and is in the exit queue. We did not have any WIP accumulation in our first queue because we did not have any variation in this model. Demand arrived every 1 time unit and it took exactly 1 time unit to process each item.

The Results of the Simulation Run

Now let's take a look at some of the information available on the screen after a run:

- The Demand Block
  - "Count" shows the total number of demand events generated by the block.
- The Queue Block
  - "Current" indicates the current number of items in the queue.
  - "Mean" indicates the average number of items in the queue.
  - "Min" and "Max" indicate the minimum and maximum number of items in the queue. Note that the exit queue will typically show "9999" for the minimum value as it never has a minimum and only accumulates exits.

    - “Count” shows the number of items that have entered the queue
    - “TCT mean” is the calculation of the average total cycle time (lead time) to get to the queue. Note that the first queue shows TCT mean = 0.00, since the demand events flow directly to the queue with no delay. The exit queue shows TCT mean = 1.0 which reflects the 1 time period of processing in the activity block. Since there is no variation in our base model there was no queue time.
- The Activity Block
    - “Util” is the percentage of available time spent working on items. It stands for utilization.
    - “Idle” is the percentage of available time spent waiting for work.
    - “Setup” is the percentage of available time spent setting up work. Setup time is considered non-value add and does not count as good ‘utilization’ of the activity.
- Total Cycle Time Histogram
    - The graph to the left at the bottom of the screen shows the distribution of Total Cycle Times for each individual item. The Total Cycle Time is the time the item takes to arrive at the station.
- Queue Length or Servers Utilized Time Series Plot
    - This graph shows the length of the queue over the entire run of the simulation. For the exit queue, this graph will just show a line representing the accumulation of exits in the queue. For a real queue, with variation in the model, this graph will look quite a bit more interesting. This is the first place to spot whether or not your process is stable.
- System WIP metrics
    - To the right of the Time Series Plot, you will see some text that is titled “System WIP Metrics”. This data shows the minimum, maximum, and average WIP across the entire model (except the exit queue), as well as a count of total WIP generated by demand blocks (the first number in the “Total WIP” line) and the total WIP in the model including items created by kitting queues (more on that topic later).
- Run Report
    - The Run Report button near the top left of the model will generate a report of the metrics for the queue and activity blocks in the model. The text in this report can be selected and then copied and pasted into Excel or any other application for further analysis. Click the Run Report button again, or move your mouse over your model blocks, to close the report window.

Model 2 – The Basic Model with Variation

The next model will be a simple extension of the basic model with the addition of random variation. The goal of simulation modeling is to see how a process will perform and react to different types and ranges of inputs. All processes have variation. Variation in demand comes from our customers, and variation in processing or service time comes from our process. With Process Playground, you have the ability to choose from several random distributions to help model the true variation in your process. The next section describes the random distributions available. Once you have reviewed the distributions, we will run some experiments in the basic model.

Random Distributions in Process Playground

The different distributions supported include:

- **C: The constant distribution**. Events following this distribution will happen without variation with an inter-arrival time as given in the top data field "Interarrival Time – mean:" So, with a value of 1 for the mean, a new demand event would occur every 1 time period. Note that inter-arrival time refers to the time between arrivals of a unit of demand.
- **U: The uniform distribution**. When the uniform distribution is selected, the minimum and maximum fields will become available, allowing you to enter the minimum value and maximum value for the uniform distribution. This is not an integer uniform distribution, so all real numbers between the min and max will be equally likely to be generated from the random distribution.
- **K: The Erlang distribution**. When you enter "K" as your distribution type, you will be using an Erlang distribution with the mean you enter and the parameter K = 4. The Erlang distribution with K = 4 is created by summing four random variables generated from an exponential distribution with the mean equal to 1/4th of the mean you have entered. The benefit of this distribution is that it closely mimics a log-normal distribution, resulting in a distribution of random variables which are positive, with a single mode slightly less than the mean, and a right skewed tail.
- **E: The Exponential distribution**. The exponential distribution requires only the mean value to be entered, as the variation of the exponential distribution is equal to the mean. The exponential distribution is highly variable and often used in modeling the arrival and service times of processes with very high transaction traffic (such as telephone systems).
    - Poisson arrivals to a process are a very common modeling method, and the exponential distribution can be used to generate Poisson arrivals to a process. For a Poisson process, demand is defined as a random number of arrivals during a fixed period of time. The distribution used to generate the inter-arrival time of those arrivals during the fixed period of time is the exponential distribution. If a

Poisson process has an average of 5 arrivals during a 10 minute time period, then the mean inter-arrival time between each of the arrivals is simply 1/5th of the 10 minute time period, or 2 minutes. So, an exponential distribution with mean = 2 minutes yields the same long term results as a Poisson process with 5 arrivals per 10 minutes. In the Poisson process, the random number of arrivals would all appear at the 10 minute mark, while the exponential process will allow the arrivals to appear, on average, every two minutes.

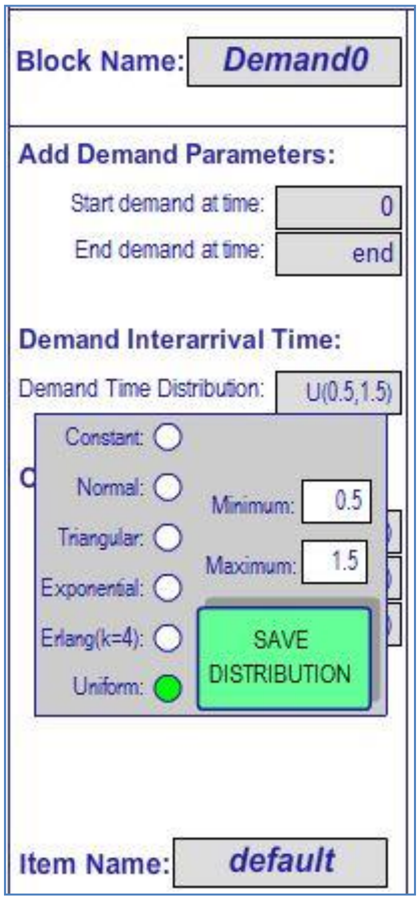
- **N: The Normal Distribution**. The Normal distribution allows you to enter the mean and standard deviation of the traditional Normal distribution. In a simulation, you can never go back in time, however the Normal distribution does have the potential for creating a random value which is negative. If this happens, the negative tail of the Normal distribution will be truncated to the value zero.
- **T: The Triangular Distribution**. The triangular distribution is a popular random distribution when modeling business service processes. The distribution closely matches real-life performance of these processes based on several characteristics of the distribution. In business processes, there is typically a minimum value-added time representing the shortest amount of time required to complete a task. The task can never be completed faster than this minimum time, so the minimum of the triangular distribution matches nicely with this true minimum.
  In business processed there is often a maximum service time, representing the point where an order has become so late that management intervenes and changes the process to get the order completed. This management intervention point is nicely modeled by the maximum parameter of the triangular distribution.
  Finally, many business processes are best able to report the mode, rather than the mean, service time of the process. The mode is directly entered into the triangular distribution, or it can be easily calculated from the minimum, maximum, and mean.

Choosing the correct random distribution for your process model is very important. There are many statistical software packages available which will help identify the appropriate mathematical distribution for data you have. However, remember that the data you have is a sample of the underlying population, and spending large amounts of time to come up with complex array of mathematical equations and distributions to represent that data is rarely a valuable exercise for a business process. Your process model is an abstraction of reality to begin with and probably does not demand super precise input data to generate the insight you need from the model.

Experiments with the Basic Model

Now let's make a few changes to the basic model to see how the model reacts to different forms of variation. Because we will be using randomly generated variation for the rest of this manual, your simulation results will never exactly match the results in this manual. But, they should be close and our average values should be very close.

For the first experiment, we will add some demand variation. We will keep the average demand at 1 time period just as in the basic model. Since the processing time of our activity is still 1 time period, this makes our target average utilization 100%. **What do you think will happen to our process when demand variation is added?** Do you think the process will continue to perform well with no queue forming in front of the activity block?


Block Name: Demand0

Add Demand Parameters:

Start demand at time: 0

End demand at time: end

Demand Interarrival Time:

Demand Time Distribution: U(0.5,1.5)

Constant: ○

Normal: ○

Triangular: ○

Exponential: ○

Erlang(k=4): ○

Uniform: ●

Minimum: 0.5

Maximum: 1.5

Item Name: default

Figure 7 - Demand Variation

To add variation to the demand block, first click on the demand block so that it has focus (the flashing red line below it). The Model Data area at the left side of the screen will display the data for the demand block. Click on the Demand Time Distribution input field. A popup box will appear. Click the radio button next to the Uniform distribution. Notice that the minimum and maximum fields are now available for data entry. Set the minimum value to 0.5 and the maximum to 1.5. This will give us a range of inter-arrival times between 0.5 and 1.5, with a mean of 1.0. After you have entered the minimum and maximum values, click the green Save Distribution button to store the data. Now run the model for the default 100 time periods.

What happened? Does the model appear to be stable based on the time series plot of the queue in front of the activity? Click on that queue and look at the time series plot. Turn off the animation in the model and rerun the model several times. How much does the time series plot for the queue change each time? Chances are high that you will have something that looks fairly similar to the plot below.

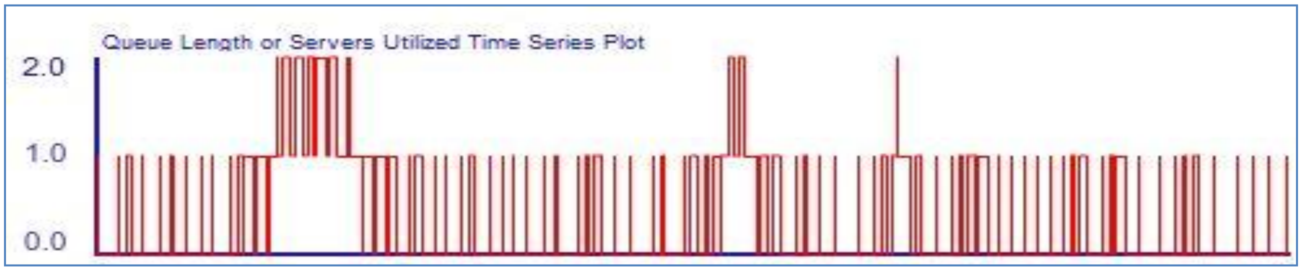


Figure 8 - Queue Length with Uniform Distribution

So it appears that the uniform distribution causes a short queue to form periodically, but the process appears to stay stable and in control for the most part. The uniform distribution is a fairly well behaved, low variation distribution. What if we change the distribution to an exponential distribution? Click on the U(0.5,1.5) shown in the delay time distribution field and change it to an Exponential distribution with mean = 1. Run it several times to see how much the time series plot varies. Did you get a time series plot that looks more like this:

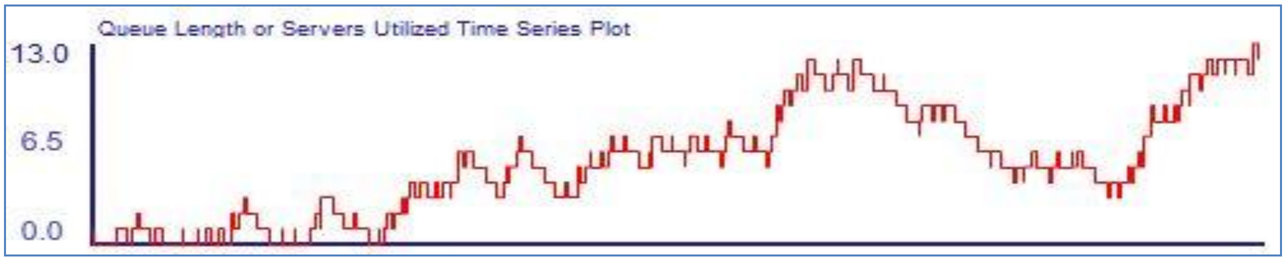


Figure 9 - Queue Length with Exponential Distribution

Obviously the exponential distribution has a much higher amount variation when compared with the uniform distribution. The result is a process that is much less stable. This leads to a very important insight about the relationship of variation and utilization in a process: as the variation present in a process increases, the long term target utilization of the process must decrease in order to maintain a stable process.

In the figure above, the queue length reached a maximum length of 13 items. This is based on a process that has a long term target utilization of 100%. For the run above, the actual utilization was 93%. **How could that be – lower utilization AND longer queue length at the same time?** The answer is in the chart. Early in the run, the queue length is often at zero. During this time, due to the randomness of demand arrivals, there were times when the activity had nothing to work on. As a result, the utilization of the activity went to zero for a period of time. However, the long term average demand rate was still 1.0, so later in the run the demands began to arrive a little more quickly, and the steady processing rate of 1.0 could not keep up. The result is less utilization than targeted, and longer queues – two bad results wrapped into one simulation run.

So what is the solution for the long queues that formed? In the long term, gaining some control over the demand variation is the best solution. Perhaps not all of that variation in demand rate is coming from the ultimate customer, but rather from our own internal processes? In the short term, the only solution is to lower the target utilization rate. This can be accomplished in two ways:

1. Add more resources to the activity block – this is like adding more people or overtime.
2. Reduce the time required to perform the activity – this is like removing some non-value added work or streamlining the work being performed.

Experiment with your model a little. To add another resource to the activity block, click on the activity block to give it focus, then change the number of servers from 1 to 2. Does your time series plot look more in control now? Does it look more like the picture below?

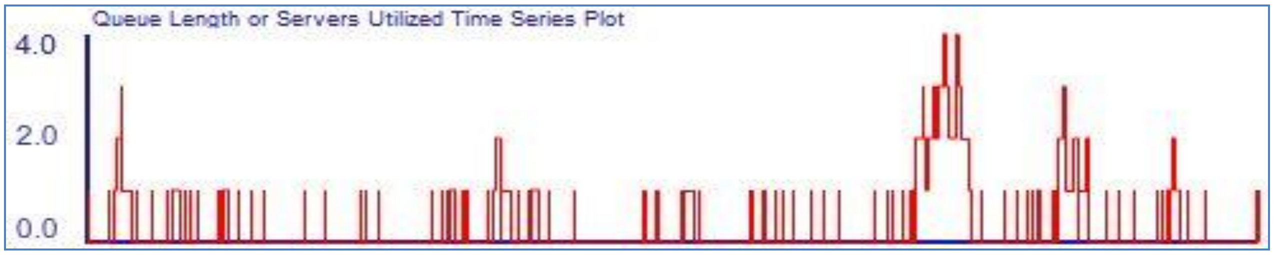


Figure 10 - Queue Length with Two Servers

There is still some variation, but the queue length is definitely much better controlled than the base case with just 1 server. However, the utilization of the two servers has probably fallen to around 50%. The queue problem is under control, but the cost of providing the service has doubled. What about making some improvements in the activity? Try changing the number of servers back to 1 and reducing the mean service time from 1.0 to 0.75, a 25% reduction in service time. Now what does the queue plot look like? Something like the picture below?

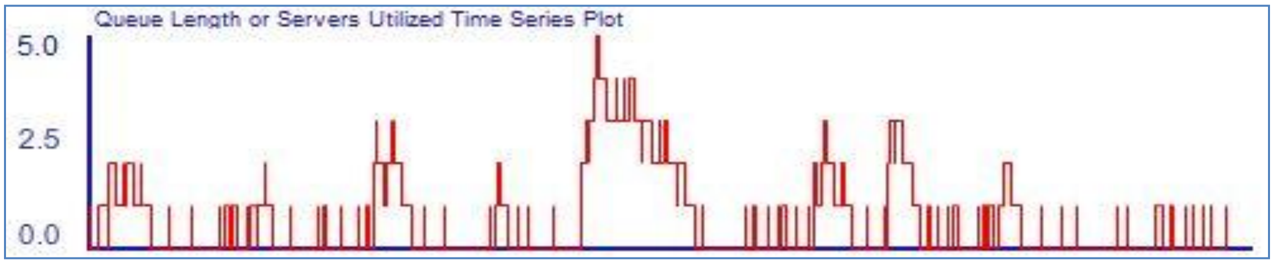


Figure 11 - Queue Length with Process Improvements

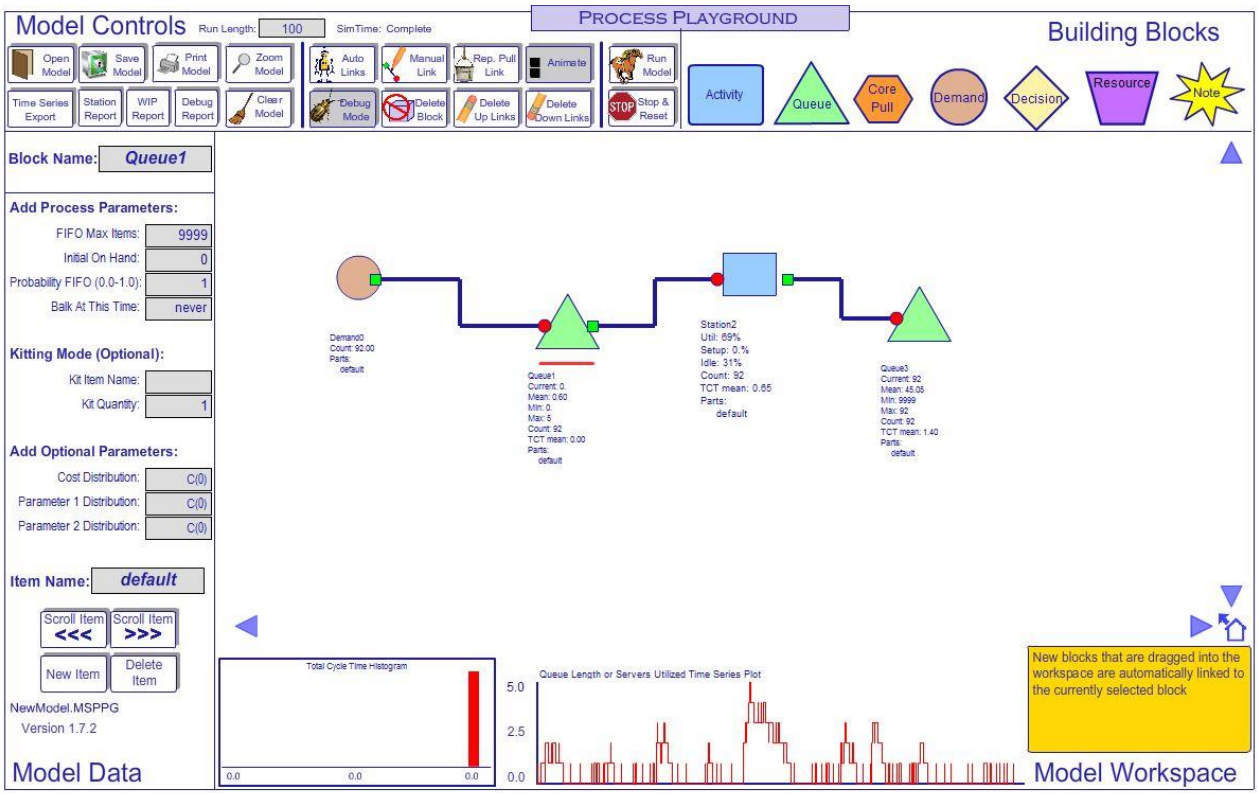
The queue length plot is looking fairly similar to the plot when we put an extra server into the activity. But now we have accomplished the gain by improving the process and removing 25% of the service time, rather than by adding the extra cost of a second server. Utilization should be very close to 75% now.

This exercise is an example of how you can use simulation to evaluate different options for improving the performance of a process, without having to implement all of the options in reality. You can now 'virtually' pilot any number of improvement concepts for your process, fine tuning the results before you run your first, live pilot. For example, how did we come up with the 25% reduction in service time in the example above? In a simple process like this we could use some complicated mathematics, or our years of experience. Or, we could build a table in Excel and run some experiments with Process Playground and quickly gain the insight we need to know how much we need to improve the process.

Model 3 – Adding Items and Controlling Flow by Item Name

The basic model you built had only one type of item, the default item, in the model. In the text under each block in the model you can see a list of parts or items. For the basic model, we left the single item for each block as the 'default' named item. Process Playground has a powerful feature which allows you to easily build complex models, with many items flowing through the process in the model, without needing to add any additional blocks to the model. This is accomplished by adding additional items to the various blocks in the model. Depending on the block, the addition of new items will cause different things to happen.

To start, we will make a few simple changes to the basic model and see what happens. Open up the original basic model (Model 2) from the previous section. This is the basic model with variation. To make sure we start in the same place, make sure your copy of Model 2 has the demand distribution type as "E" for exponential, mean inter-arrival time of 1, and the service time in the activity station set as "C" for constant with mean service time of 0.75. You model should look like the figure below.



Run Length: 100
SimTime: Complete

Block Name: Queue1

Add Process Parameters:
FIFO Max Items: 9999
Initial On Hand: 0
Probability FIFO (0.0-1.0): 1
Balk At This Time: never

Kitting Mode (Optional):
Kit Item Name:
Kit Quantity: 1

Add Optional Parameters:
Cost Distribution: C(0)
Parameter 1 Distribution: C(0)
Parameter 2 Distribution: C(0)

Item Name: default

NewModel.MSPPG
Version 1.7.2

Demand0
Count: 92.00
Parts:
default

Queue1
Current: 0.
Mean: 0.60
Min: 0.
Max: 5
Count: 92
TCT mean: 0.00
Parts:
default

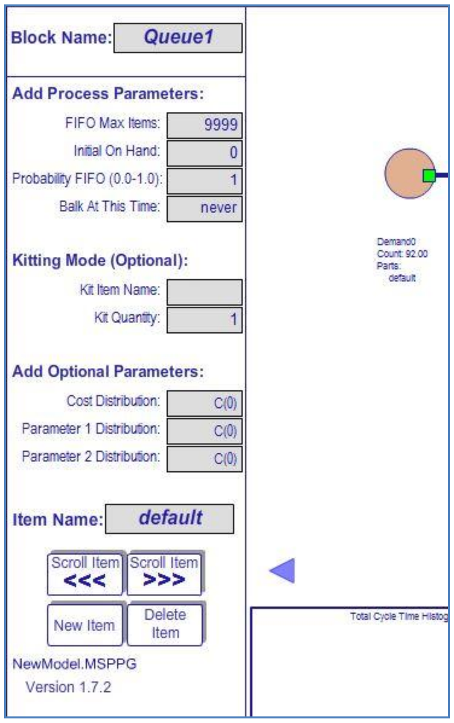
Station2
Util: 69%
Setup: 0.%
Idle: 31%
Count: 92
TCT mean: 0.65
Parts:
default

Queue3
Current: 92
Mean: 45.05
Min: 9999
Max: 92
Count: 92
TCT mean: 1.40
Parts:
default

New blocks that are dragged into the workspace are automatically linked to the currently selected block

Figure 12 - Model 2 Starting Point

Note that each block in the model has a list of parts associated with the block. The first block we will work with is the demand block. Start by selecting the demand block. Now take a look at the lower portion of the Model Data section. Do you see the “Item Name” field and the name “default” for the item name? Your screen should look like the figure at the right.


Block Name: Queue1

Add Process Parameters:

FIFO Max Items: 9999
Initial On Hand: 0
Probability FIFO (0.0-1.0): 1
Balk At This Time: never

Kitting Mode (Optional):

Kit Item Name:
Kit Quantity: 1

Add Optional Parameters:

Cost Distribution: C(0)
Parameter 1 Distribution: C(0)
Parameter 2 Distribution: C(0)

Item Name: default

NewModel.MSPPG
Version 1.7.2

Demand0
Count: 92.00
Parts:
default

Figure 13 - The Default Item

The buttons for managing the items in a block are located below the “default” name. There are buttons for scrolling through the items in the block, adding a new item, or deleting and item. If you click on either of the scroll buttons now, nothing will happen since there is only one item in the block.

Multiple Items in the Demand Block

Go ahead and click on the New Item button now. When you do, you will see the Item Name field change from “default” to “NewPart”. If you click on the scroll buttons now, you’ll see you now have two items in the demand block. Go ahead and click on the text “NewPart” and rename the item “Part B”. Then scroll to the “default” item and rename it “Part A”. Your demand block in the model should now show two parts in its parts list – Part A and Part B.

Now scroll to Part B in the Model Data section. The default distribution type should be “C” for constant. Go ahead and change that to “E” to match Part A. Now both Part A and Part B will be arriving with an exponential distribution. By adding the additional item to the demand block, we have created the same effect as adding a second demand block to the model. Go ahead and run the model now. You should see a very large, out-of-control queue forming in front of the activity block. We doubled the demand rate by adding the second item in the demand block, but did nothing to increase the capacity of the activity block.

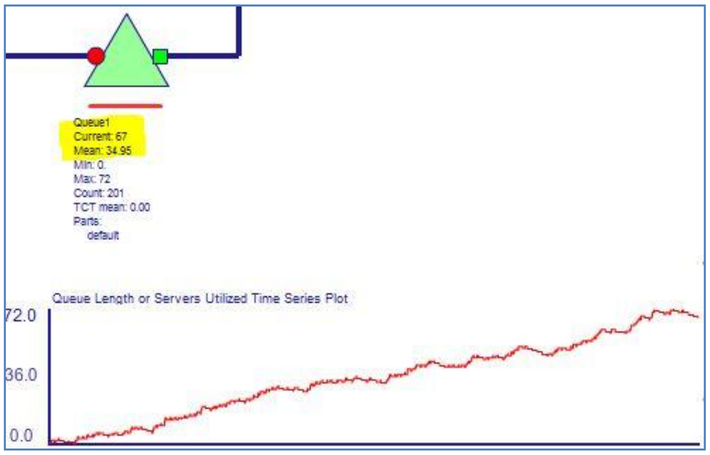
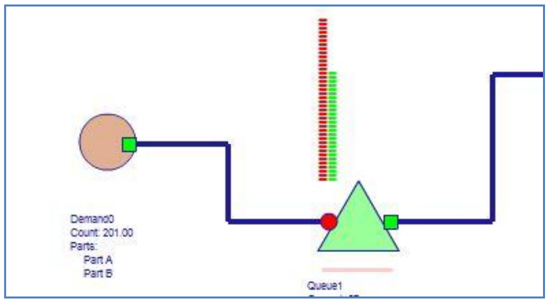

Queue1
Current: 67
Mean: 34.95
Min: 0.
Max: 72
Count: 201
TCT mean: 0.00
Parts:
default

Figure 14 - Items in the Queue Block

Multiple Items in the Queue Block

Now that we have added two items to the demand block and doubled the demand in our model, let’s turn our attention to the next block in the model, the first queue. Once you have run the model, you should be able to see in the text below the block icon that the current inventory in the model is quite high (due to the model have more demand than capacity). However, when you ran the model, you did not see any animation of the inventory levels above the queue block. Why is this happening? If you look at the Item Name field, you will see that the “default” item is still present in the queue block. And, there really are zero “default” items in the queue because the demand block is only generating Part A and Part B items. The text under the icon in the Model Workspace shows the entire inventory in a queue, regardless of its type. In order to see the animation of each type of item, you have to add each of those items to the queue.

Go ahead and double-click on “default” in the Item Name field and rename the item “Part A”. Then click the New Item button and rename the “NewPart” item to “Part B”. Now rerun the model with the animation turned on (the animation button is white, not grey) and watch the animation of the two items. You should now see unique colors and animated stacks of parts in the queue for each item. Once an item has been created with a unique Item Name, that name will remain with it throughout the simulation.


Demand0
Count: 201.00
Parts:
Part A
Part B

Queue1

Figure 15 - Queue Block Current On-Hand

Multiple Items in the Activity Block

The next block in our model is the activity block. So far, the activity block has processed all of the items coming to it, regardless of the name given to the item. <u>Whenever the activity block has an item with the name “default”</u>, the activity block will be able to process any item that comes to it. However, if you change the “default” item to something else, you can adjust the processing times and capabilities based on the item name.

To see how this works, click on the activity block, and in the Model Data section double click on the Item Name “default” and change it to “Part A”. Now rerun the model. What happened? If you look in the first queue you should see a large stack of Part B items, the green animation items in the picture to the right. By changing the “default” item in the activity to “Part A”, we told the activity that it could only work on Part A items. The Part B items had nowhere to go, so they just stopped and remained in the queue.

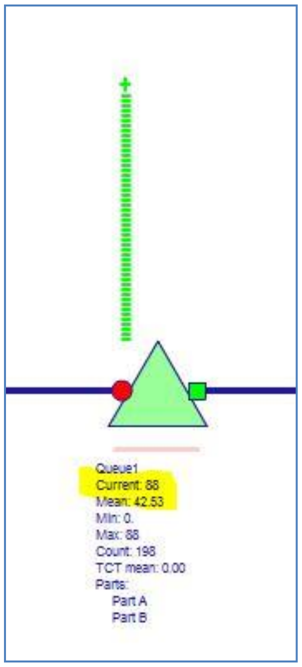
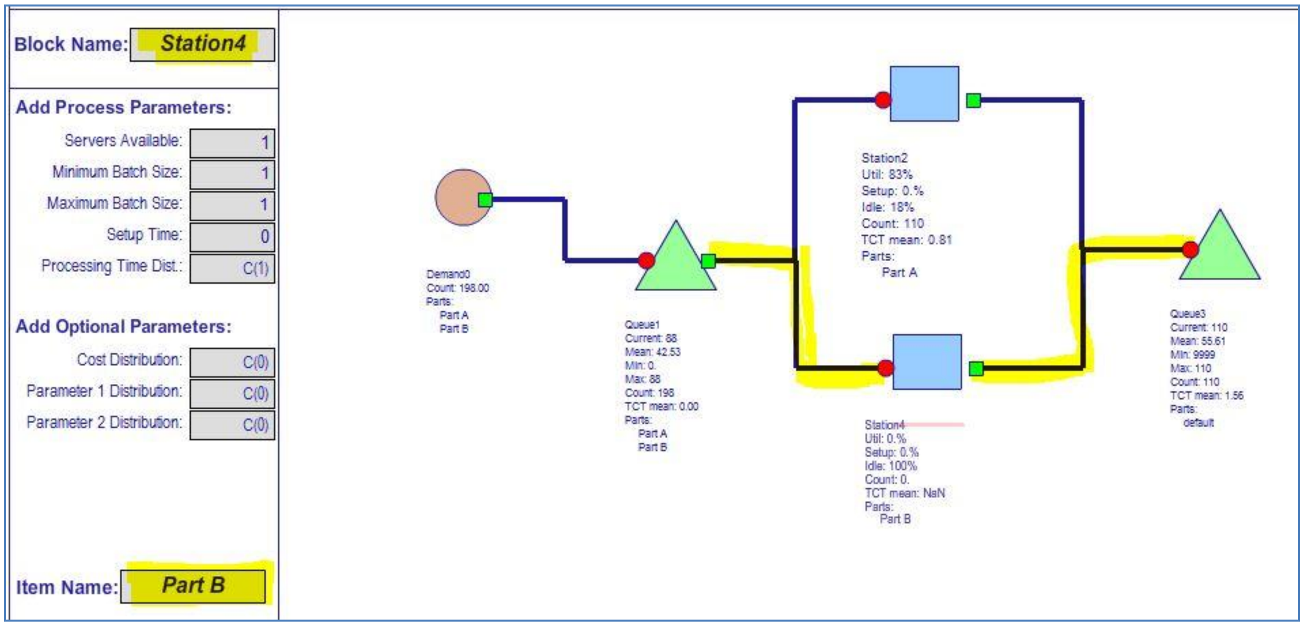

Queue1
Current: 88
Mean: 42.53
Min: 0.
Max: 88
Count: 198
TCT mean: 0.00
Parts:
Part A
Part B

Figure 16 - Part B Stuck In Queue

Process Playground automatically manages the flow of items in the model based on which activities can work on the different types of items. To demonstrate this, we will now add a separate path in the process for the Part B items to flow through. To do this, first click on the queue in front of the activity so that it has focus (the flashing red line beneath it). Then drag and drop a new activity block below the current activity block in the model. It should automatically connect to the queue block as long as you have the AutoLinks button turned on.

Once you have added the new activity block to the model, then use the Manual Link button to add a link between the new activity and the exit queue. To do this, click the Manual Link button first and notice that it will turn green. While it is green, click on the new activity station. Once you have clicked on the activity station, the button will turn red indicating that you need to click on the destination for the link. Now click on the exit queue and the Manual Link button will return to the normal state. When you have completed the manual link between the new activity and the exit queue, the last thing to do is to edit the Item Name in the new activity and change it from “default” to “Part B”. You should end up with a model that looks like the figure below.

Note that the first activity block can only process Part A items, and the second activity block



Block Name: Station4

Add Process Parameters:

Servers Available: 1
Minimum Batch Size: 1
Maximum Batch Size: 1
Setup Time: 0
Processing Time Dist.: C(1)

Add Optional Parameters:

Cost Distribution: C(0)
Parameter 1 Distribution: C(0)
Parameter 2 Distribution: C(0)

Item Name: Part B

```
Demand0
Count: 198.00
Parts:
   Part A
   Part B

Queue1
Current: 88
Mean: 42.53
Min: 0.
Max: 88
Count: 198
TCT mean: 0.00
Parts:
   Part A
   Part B

Station2
Util: 83%
Setup: 0.%
Idle: 18%
Count: 110
TCT mean: 0.81
Parts:
   Part A

Station4
Util: 0.%
Setup: 0.%
Idle: 100%
Count: 0.
TCT mean: NaN
Parts:
   Part B

Queue3
Current: 110
Mean: 55.61
Min: 9999
Max: 110
Count: 110
TCT mean: 1.56
Parts:
   default
```

Figure 17 - Model 3 with Two Items

can only process Part B items. The two activities share a common queue in the model, but the queue actually maintains the first-in-first-out (FIFO) order of all the items in the queue, and will send the appropriate Part A or Part B as required when either of the two activity blocks becomes available.

Go ahead and run the model now. Notice how items are being worked on by both activities now. If you would like to see how many of each item are completed, you can look at the “Count” value in the text below each activity block, or you can create “Part A” and “Part B” in the exit queue and be able to see the individual counts for both items there.

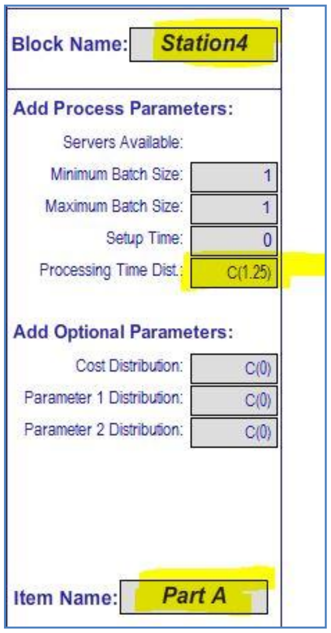
Experiment: Changing Demands and Cross Training

Model 3 now represents a typical process where we have two types of work arriving, Part A and Part B, which are then worked on by separate resources, the two activity blocks. But what happens if demand for Part A increases while demand for Part B decreases. This is a fairly common occurrence in real processes, and one that we can model quite easily with Process Playground.

Click on the demand block in your model, and then scroll to Part A in the item list. Change the mean inter-arrival time for Part A from 1.0 to 0.75 time periods. Now scroll to the Part B item and change the mean inter-arrival time for Part B from 1.0 to 1.5. While demand has shifted from Part B to Part A, the overall demand for both parts together has remained constant. Go ahead and rerun the model – now what do you see? You should see some Part A items in queue waiting to be processed and no Part B items. Also, you should see that the top activity block, the one which can only process Part A items, is 100% utilized, while the bottom activity, which processes only Part B items, is probably between 60-70% utilized.

The shift in demand from Part B to Part A has caused an imbalance in our process, something that we see happening in our real processes every day. A common solution is cross training so that more of our resources can perform more of the tasks needed. However, how much cross training, and which resources? Simulation modeling can help answer that question for us. In the case of our simple model here, we don't have too many choices to make. Let's add some cross training to the bottom activity.

To do this, click on the bottom activity and, in the Model Data section, click on the New Item button to add a new item to this activity. Rename the new item from "NewPart" to "Part A". This creates the ability for the bottom activity block to work on Part A in addition to the existing Part B. With cross-training, it is very common for the cross-trained resources to be a little less productive on their alternative work as opposed to their main work. We can reflect this in our model by changing the processing time for the new Part A item in the bottom activity. Change the mean service time from 1.0 to 1.25 for Part A to reflect this lower productivity.


Block Name: Station4

Add Process Parameters:

Servers Available:	
Minimum Batch Size:	1
Maximum Batch Size:	1
Setup Time:	0
Processing Time Dist.:	C(1.25)

Add Optional Parameters:

Cost Distribution:	C(0)
Parameter 1 Distribution:	C(0)
Parameter 2 Distribution:	C(0)

Item Name: Part A

Figure 18 - Adding Part A

Now rerun the model with the changes to the lower activity in place. What happened this time? Did you get utilization numbers and queue lengths similar to those shown in the next figure?

You probably experienced a bit of a flip in the utilization numbers, with the top activity utilization changing from 100% to the 70% range, while the bottom activity utilization went up. Why did this happen? The bottom activity is now taking both Part A and Part B based on the FIFO order of the queue. This means that at some times, the bottom activity began work on a Part A item leaving only Part B items in the queue. The top activity then had nothing to work on since it was not cross-trained to work on Part A items.

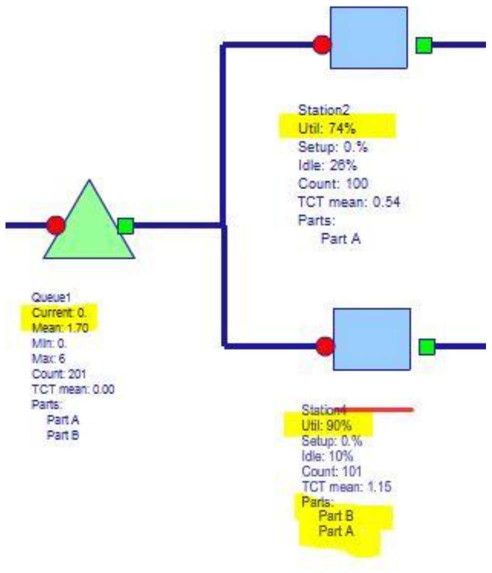


Figure 19 - New Utilization Numbers

So, cross-training for the bottom activity helped, but our experiment has shown that we either need to provide cross-training to the top activity as well, or we need some sort of 'smart' queue management which will not always use FIFO rules to determine how to distribute work. Process Playground does not have the capability to model complex queue management rules like that, so let's go ahead and add Part B to the top activity so that activity will be cross-trained as well. As we did in the bottom activity, let's have Part B in the top activity have a mean service time of 1.25 and Part A in the top activity have a mean service time of 0.75. The utilizations between the stations should now be fairly equal as both stations are able to work on both items. Process Playground automatically manages the flow of items between the stations based on the queue of items waiting and when the activities become available.

Model 4 – The Balking Queue and the Non-FIFO Queue

We will use Model 1 as the basis for Model 4, learning several new options in the queue block as we transform the model. To start, open up your copy of Model 1. Recall that this model should be a simple process which consists of a demand block flowing into a queue, flowing into an activity, and then finally ending in an exit queue. Both the demand and activity blocks should have "C" constant distributions and mean arrival and service times of 1.

Model 1 has no variation and is a perfectly balanced model. If you run the model, you will see in the exit queue that the TCT mean value is 1.00, and looking at the Total Cycle Time Histogram you will also see that all items arrived at the exit in exactly 1 time period.

The first change we will make to the model is to add some initial on-hand inventory into the first queue. Since our process is perfectly balanced, this inventory will simply add a permanent amount of work-in-process in the queue which will be worked on in a first-in-first-out basis. Click on the first queue and enter the value 10 in the "Initial On Hand" field. Now rerun the model. When the run is complete, click on the exit queue so you can see the Total Cycle Time Histogram graph. It should look like the figure to the right. Also notice that the TCT mean value for the exit queue is now above 10. This reflects the 10 items in queue that will require 10 time periods to process, lengthening the total cycle time through the process. Now you know how to introduce work-in-process into a simulation from the start using the Initial On Hand field.

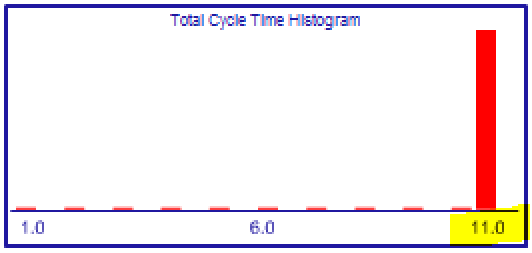


Figure 20 - TCT with Queue of 10

Non-FIFO Queue Behavior

The next feature of the queue block that we will experiment with concerns the behavior of the queue as a FIFO queue. In reality, many queues do not maintain a strict first-in-first-out behavior. There are slips and skips that allow random items in the queue to jump the line and get processed first. Process Playground allows you to model this non-FIFO behavior with a simple input. Click on the first queue again, and now take note of the field called "Probability FIFO". The default value for this field is 1, which corresponds with 100% pure FIFO operation. If you enter 0.5, then 50% of the time the queue will take the next in line following strict FIFO rules, and the other 50% of the time (randomly determined) the queue will randomly select an item from somewhere other than the first position in the line. If you enter 0.0, then the queue will never operate as a FIFO queue and will always randomly select an item from somewhere in the queue.

To see the effects of this, go ahead and click on the first queue in the model, and then enter 0.5 for the Probability FIFO value. Run the model and then click on the exit queue. Remember you can turn off animation using the animation button to make the model run much more quickly. Your results will vary, but you should have a Total Cycle Time Histogram chart which shows a lot of variation in the TCT times for each item. This reflects the 50% non-FIFO, random behavior of the queue. If you rerun the model with the Probability FIFO value set to zero, you should see even more variation in the TCT times.

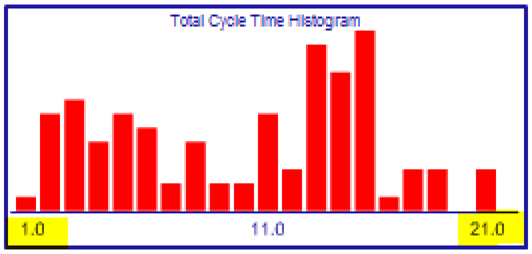


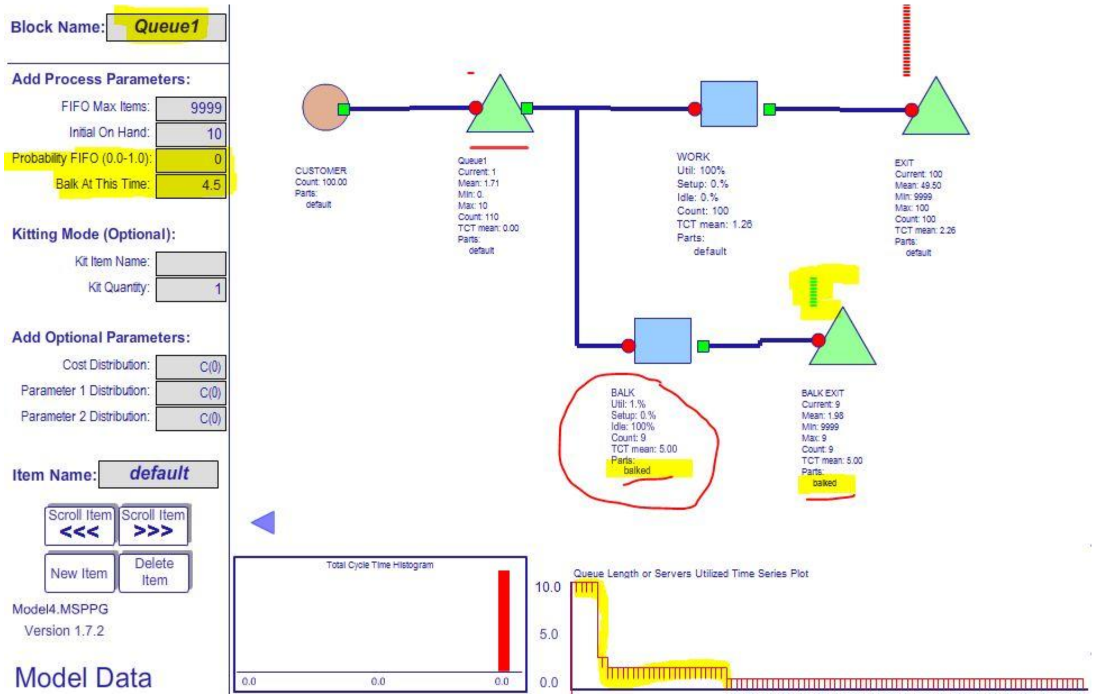
Figure 21 - TCT with Non-FIFO Queue

Balking and Exiting the Queue

Our model is now at a point (with Probability FIFO value set to zero) where the items in the line are experiencing very large variation in the amount of time they wait in queue. If the items were people, it would be reasonable to expect that after some amount of time waiting, the people may choose to leave the line. This behavior is called **balking**, and Process Playground allows you to model balking at a specific time threshold you enter. Another common use of balking in queues is when the items in the queue are perishable. If they pass a certain length of time in the queue, they perish and are disposed of.

Click on the first queue in your model and note the field named “Balk at Time”, directly below the “Probability FIFO” field. If you enter a non-zero time in this field the queue will “balk” any items which spend more than that amount of time in the queue. For our experiment, enter 4.5 as the balking time.

Items that balk from the queue are renamed from the original name and take on the name “balked”. Click on the first queue so it has focus, then drag a new activity into the model below the current activity. Then drag a new queue to connect to the new activity. Finally, rename the default item “default” in the new activity to “balked”. This will tell the first queue that the new activity will only accept items named “balked”, thus providing a new path in the process for any items that have balked. Your model should look like the figure below. Note the highlighted items to make sure you have them right.



Block Name: Queue1

Add Process Parameters:

FIFO Max Items: 9999
Initial On Hand: 10
Probability FIFO (0.0-1.0): 0
Balk At This Time: 4.5

Kitting Mode (Optional):

Kit Item Name:
Kit Quantity: 1

Add Optional Parameters:

Cost Distribution: C(0)
Parameter 1 Distribution: C(0)
Parameter 2 Distribution: C(0)

Item Name: default

Model4.MSPPG
Version 1.7.2

CUSTOMER
Count: 100.00
Parts:
default

Queue1
Current: 1
Mean: 1.71
Min: 0.
Max: 10
Count: 110
TCT mean: 0.00
Parts:
default

WORK
Util: 100%
Setup: 0.%
Idle: 0.%
Count: 100
TCT mean: 1.26
Parts:
default

EXIT
Current: 100
Mean: 49.50
Min: 9999
Max: 100
Count: 100
TCT mean: 2.26
Parts:
default

BALK
Util: 1.%
Setup: 0.%
Idle: 100%
Count: 9
TCT mean: 5.00
Parts:
balked

BALK EXIT
Current: 9
Mean: 1.98
Min: 9999
Max: 9
Count: 9
TCT mean: 5.00
Parts:
balked

Figure 22 - Balking Queue

Once you have made all the required changes, go ahead and run the new model. You should see a graph similar to the one shown in the figure when you click on the first queue. Notice how the queue length started out at 10, but then after the 4.5 time periods had passed, the queue “balked” nine items from the queue and sent them to the bottom activity. Once the initial on-hand inventory had balked from the queue, the process stabilized and eventually returned to just one piece in queue. Due to the randomness of the non-FIFO operation, your chart may look a little different, but it should be close to the one in the figure.

Model 5 – The Kitting Queue

A very common function in any process is the collection of several items together into a single assembled package. This could be the individual pieces of paperwork required for a loan application assembled into a single folder, or individual pieces of a machine assembled into the final product, the machine. In Process Playground, we accomplish this collection of items by building a kit of parts. The kit of parts then takes on a new name as a new item in the model.

To see how this works, we will build a simple process of brewing coffee. Start with a clean sheet in Process Playground and drag a demand block to the top left corner of the workspace, then drag a queue into the workspace. It should automatically connect to the demand block if you have AutoLinks turned on. Next, while the new queue still has focus, drag another queue into the model to connect to the first queue. Rename the second queue “KIT QUEUE”. You should have a model that looks like this:

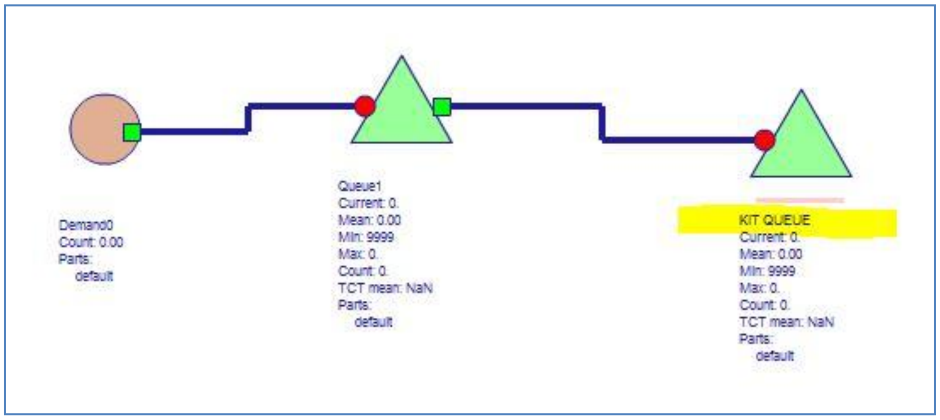


Figure 23 - Kitting Queue

The next step is to turn off the AutoLinks function by clicking the AutoLinks button at the top of Process Playground. It will turn grey to indicate that the function is off. Now drag a new queue into the workspace and place it below the first queue, Queue1. Now you will need to use the Manual Links button to create manual link between the new queue you just dragged into the model and the KIT QUEUE. Once you have completed these steps, turn AutoLinks back on and then click the KIT QUEUE block so that it has focus. Then drag an activity block into the model. It should automatically link to the KIT QUEUE block. Finally drag another queue block into the model to the right of the activity block – this will be the exit queue. When you are finished, your model should look like this:

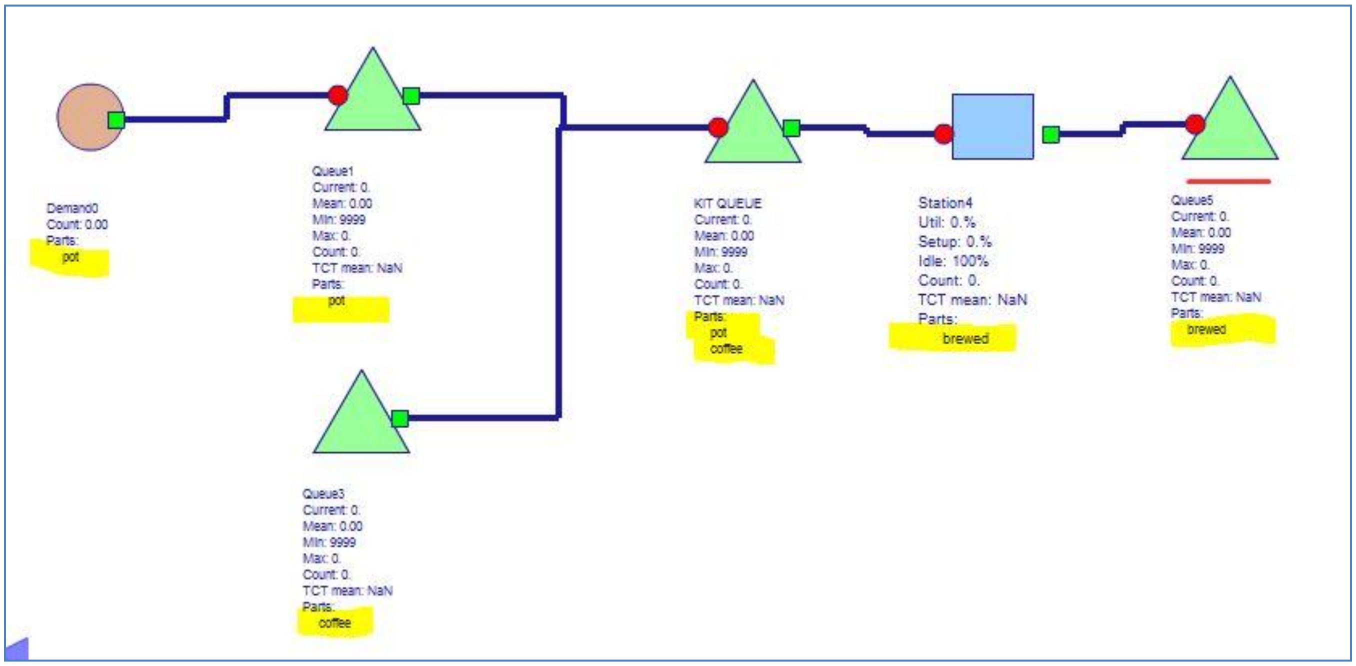
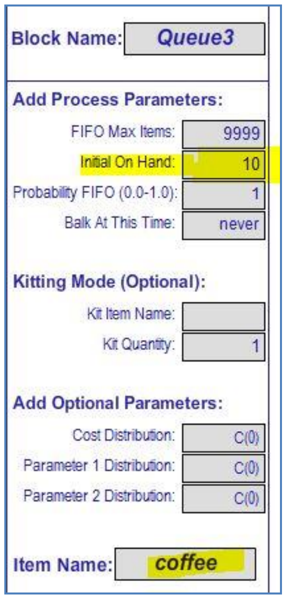


Figure 24 - Coffee Brewing

Now you will have to edit all of the item names in the model. The demand block and Queue1 will have the item name "pot". The bottom queue, Queue3, will have the item name "coffee". The KIT QUEUE will have two items, "pot" and "coffee", and the activity and exit queue will have the item name "brewed". Set the run length of the simulation to 10 time periods at the top of the screen, and set the Initial On Hand field in the bottom queue to 10 as well.



Block Name: Queue3

Add Process Parameters:

FIFO Max Items:	9999
Initial On Hand:	10
Probability FIFO (0.0-1.0):	1
Balk At This Time:	never

Kitting Mode (Optional):

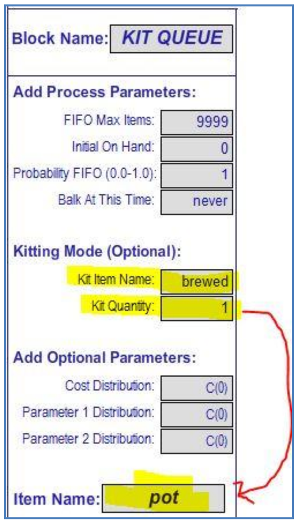
Kit Item Name:	
Kit Quantity:	1

Add Optional Parameters:

Cost Distribution:	C(0)
Parameter 1 Distribution:	C(0)
Parameter 2 Distribution:	C(0)

Item Name: coffee

Figure 26 - Queue 3



Block Name: KIT QUEUE

Add Process Parameters:

FIFO Max Items:	9999
Initial On Hand:	0
Probability FIFO (0.0-1.0):	1
Balk At This Time:	never

Kitting Mode (Optional):

Kit Item Name:	brewed
Kit Quantity:	1

Add Optional Parameters:

Cost Distribution:	C(0)
Parameter 1 Distribution:	C(0)
Parameter 2 Distribution:	C(0)

Item Name: pot

Figure 25 - Kitting Queue

Now you will need to setup the kitting function in the KIT QUEUE. Select the KIT QUEUE and scroll through the two items you have built in this queue – "coffee" and "pot". Depending on the order you entered the two items, one of them will display the field "Kit Item Name" and the other one will not (notice that the Probability FIFO and Balk At Time fields are also set only once for the queue, not for each item). In the Kit Item Name field enter the name "brewed". This will be the name of the new item that is created from the joining of 1 pot with 1 unit of coffee. For each item you will see the field "Kit Quantity". This field is used to tell the kitting queue how many of each item are required to make a kit. For our model, we will leave both of these values at one, however we could easily change the "coffee" item to 2 if we required two scoops of coffee to brew a pot.

The figure to the left shows the kitting queue Model Data. When you run the model with animation turned on, you will see the KIT QUEUE pulling in one unit of coffee and then waiting for the demand block to generate a demand event called “pot”. When the “pot” item joins the “coffee” item in the KIT QUEUE, the two items are combined into a “brewed” item which is sent to the brew activity.

For this lesson, and the following lessons, you may want to refer to the training videos that are also provided on the Process Playground launch page. These videos will take you through all of the functions of Process Playground while you watch the action on the screen.

Additional Training Videos

Model 6 – Replenishment Pull Systems

Replenishment pull systems are a common Lean tool used in many projects. Process Playground has been designed to make it easy to create replenishment pull models. This video will take you through the entire process of building, testing, and analyzing the performance of a replenishment pull system. This video will also introduce the use of the Demand block as a “pull” demand generator rather than the “push” demand generator we have seen so far.

Model 7 – Complex Demand Patterns

Process Playground has the ability to create complex demand patterns using the start and stop time settings for individual items in the demand block. Using this feature, almost any pattern of demand variation can be recreated in your models.

Model 8 – Batch Processing

Process Playground supports batching and un-batching of items using the activity block. Setup time can be applied to the batch, while individual processing time is calculated for each item in the block. This video will show you how to create and analyze batch processing models.

Model 9 – The Core Process Pull Block

Core Process pull is a fundamental Lean concept that is critical to almost every concept. The Core Process pull block allows you to quickly build models which incorporate this type of pull system.

Model 10 – The Resource Block

The Resource block allows you to create a pool of resources to be shared amongst a group of activities. The resource pool might represent people in a transactional model or tooling in a manufacturing model. The resource block also supports the start and stop time features of the demand block, allowing you to create complicated models of employee shifts.

Model 11 – The Decision Block

The Decision block is a simple block that offers a random redirection of items in the model based on the probabilities you enter. The decision block is most often used to create random rework and scrap loops in models.

Model 12 – The Note Block

The final building block available in Process Playground is the Note block. This block is not functional, but allows you to add notations directly in the model and connect those notations to the functional blocks of the model.