Simulación de un proceso industrial mediante el software FlexSim

1. Objetivos

- Conocer las características del simulador FlexSim.
- Tener la capacidad de analizar un proceso y poder simularlo.
- Ser capaz de simular un proceso productivo y obtener los resultados óptimos.

2. Características básicas y descripción de FlexSim

Flexsim (http://www.flexsim.com) es un software para la simulación de eventos discretos, que permite modelar, analizar, visualizar y optimizar cualquier proceso industrial, desde procesos de manufactura hasta cadenas de suministro. Además, Flexsim es un programa que permite construir y ejecutar el modelo desarrollado en una simulación dentro de un entorno 3D desde el comienzo. Actualmente, El software de simulación Flexsim es usado por empresas líderes en la industria para simular sus procesos productivos antes de llevarlo a ejecución real. Actualmente, existe mucha gente implicada en este proyecto y su uso se encuentra muy extendido en EEUU y México. De esta manera posee un extenso grupo o comunidad de desarrolladores (muchos de ellos desinteresados) que han aumentando y mejorado las competencias del software. Existe una web propietaria del software (http://www.flexsim.com/community/forum/downloads.php) que posee multitud de descargas de herramientas adicionales al software, como modelos 3D y librerías, y una gran comunicación entre ellos mediante foros.

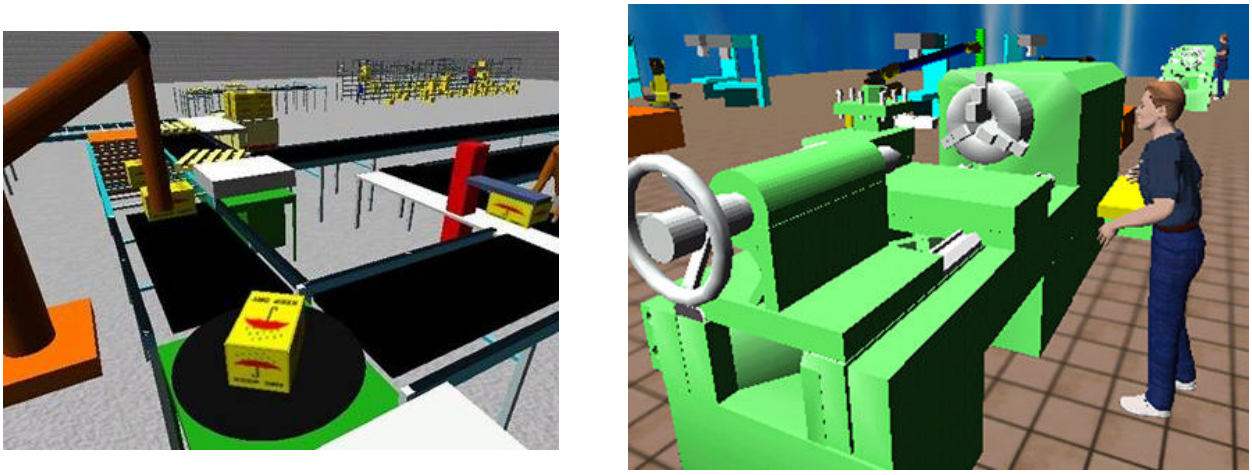


Figura 1. Aspecto gráfico de dos simulaciones desarrolladas con el programa FlexSim

Un modelo desarrollado con el software Flexsim es básicamente un sistema de flujo de entidades (*flowitems*), colas (*queues*), procesos (*processor*) y sistemas de transporte (*transportation*). El proceso consiste en un retraso forzado (*delay*) realizado por una máquina, el transporte consiste en el movimiento de entidades de un recurso a otro, y las

colas son un acumulamiento de entidades tipo FIFO a la entrada de un proceso esperando para su procesamiento.

Básicamente, un modelo en Flexsim consta de los siguientes recursos:

- Recursos constantes o fijos (*fixed resources*). Aquí entrarían las colas (*queues*), las máquinas o procesos (*processor*) y las cintas transportadoras (*conveyors*).
- Recursos compartidos (*shared resoruces*). En este apartado están los operadores.
- Recursos móviles (*mobile resources*). En este apartado entran los sistemas de transporte que permite modelar el software tales como elevadores, trans-paletas, robots industriales, etc.

Como elemento producido o producto en elaboración, está la entidad que fluye por cada uno de los recursos del modelo. A dicha entidad, dentro del Flexsim, se le denomina *flowitem*. Estas entidades pueden guardar información relativa a dicho producto, tales como etiquetas o *labels* (por ejemplo, el código de barras de un producto determinado) y el tipo o *itemtype*, que tipificada a las entidades o productos en elaboración del modelo simulado. Estas dos propiedades pueden utilizarse para definir tiempos de proceso y encaminar las entidades por su flujo de producción correcto.

En los recursos fijos (colas, máquinas, etc.), el usuario deberá configurar el tiempo de proceso que consume una entidad mientras se elabora o procesa en dicho recurso, si es necesario transportarlo mediante un recurso compartido, como puede ser un operador, o si es necesario llamar a un recurso móvil.

Todos estos aspectos comentados se irán mostrando a lo largo de este documento para poder aprender cómo se simula mediante Flexsim.

2.1. Terminología FlexSim.

A continuación, se muestra la definición de una serie de términos necesarios para poder comenzar a modelar un proceso con Flexsim:

- **Modelo de simulación**: un grupo de instancias de objetos que representan un proceso industrial.
- **Objeto**: instancia de una clase (recursos o entidades) con comportamiento propio, con atributos, variables y propiedades visuales definibles por el usuario.
- **Librería**: lista de clases para definir un modelo. Las instancias u objetos son creadas en el modelo Flexsim arrastrando el tipo de clase seleccionado de la librería proporcionada por el software al entorno de simulación.
- **Objetos de Flexsim**: los objetos de Flexsim simulan diferentes tipos de recursos en la simulación. Como se ha comentado con anterioridad, el objeto llamado *queue* o cola, actúa como un buffer o un área de almacenamiento. El *queue* puede representar una fila de personas esperando, una fila de procesos que esperan ser procesados por una computadora, un área de almacenamiento en el piso de una fábrica, o bien una fila de llamadas en espera de un centro telefónico de servicio a clientes. Otro ejemplo comentado de objeto de Flexsim, es el objeto llamado *processor*, que simula un tiempo de demora o de proceso. Este objeto puede representar una máquina de una fábrica, un cajero atendiendo a un cliente en un banco, un empleado del correo acomodando paquetes, el tiempo de curación de un enfermo, etc. A todos los objetos de Flexsim se les puede modificar su apariencia fácilmente cambiando el dibujo 3D que tienen asignado. Los objetos de Flexsim se encuentran la en la librería de objetos.

- **Flowitems**: tal y como se ha comentado, los *flowitems* son los objetos que se mueven a través del modelo. Los *flowitems* pueden representar productos, partes, tarimas, ensambles, papeles, contenedores, llamadas telefónicas o cualquier cosa que se mueva a lo largo del proceso que estás simulando. Se les pueden aplicar procesos a los *flowitems* y también pueden ser cargados y transportados mediante personas o equipos de transporte. En Flexsim, los *flowitems* son creados que el objeto denominado *Source*. Una vez que los *flowitems* han pasado a través del modelo, éstos deben ser enviados a un objeto llamado *Sink*, que se pone al final del proceso.

- **Itemtype**: el *itemtype* es una etiqueta que tienen todos los *flowitems* o productos y puede representar el tipo de producto. Flexsim está preparado para utilizar el itemtype como una referencia para decidir la ruta o el destino al cual deben de enviarse los *flowitems*.

- **Ports**: cada objeto de Flexsim tiene un número ilimitado de puertos llamados *ports* a través de los cuales se comunican con otros objetos. Existen 3 tipos de puertos: puertos de entrada (*input ports*), puertos de salida (*output ports*) y puertos centrales (*central ports*). Los puertos de entrada y de salida se usan para definir el flujo o la ruta de los *flowitems* o productos. Por ejemplo, supongamos una simulación de un separador de correos, que coloca los paquetes en uno de los diferentes transportadores dependiendo del destino del paquete. Para simular esto en Flexsim, se debe conectar los puertos de salida de un objeto del tipo *processor* (que sería el separador de correo) a los puertos de entrada de los diversos objetos del tipo *conveyor*, lo que significa que cuando el *processor* (separador de correos) haya terminado de procesar el *flowitem* (paquete), entonces este será enviado a un *conveyor* específico a través de uno de sus puertos de salida. Los puertos centrales son utilizados para crear referencias de un objeto a otro. Un uso típico de los puertos centrales es para referenciar a los objetos móviles como operadores, montacargas o grúas con los objetos fijos como máquinas, *queues* o colas y *conveyors* o transportadores.

- **Triggers**: un *trigger* es un desencadenador que se activa al ejecutarse eventos claves de objeto. El usuario puede especificar una variedad de eventos que quieren que suceda cuando se activa un desencadenador o *trigger*. Por ejemplo, existen desencadenadores o *triggers* para cuando un *flowitem* entra o sale de un objeto. Cada tipo de objeto tiene un conjunto de desencadenadores que el usuario puede programar cuando lo desee.

2.2. Creación y conexión de puertos.

Los puertos son creados y conectados al hacer *click* con el botón izquierdo del ratón en el primer objeto y arrastrando hasta el segundo objeto mientras se presiona al mismo tiempo alguna de las siguientes letras del teclado (véase Tabla 1). Si se presiona la letra "A" mientras se hace un *click* y se arrastra el puntero del ratón, un puerto de salida será creado desde el primer objeto y un puerto de entrada será creado en el segundo objeto. Estos dos puertos quedarán conectados automáticamente. Si se presiona la letra "S" se creará un puerto central en ambos objetos y se conectarán dichos dos puertos nuevos.

Si se presiona la letra "Q" (que se ubica arriba de la "A" en el teclado), entonces los puertos de entrada y de salida serán borrados. Para borrarlos deberá hacerse las conexiones en el mismo sentido y de la misma forma en que se hicieron con la letra "A", pero ahora con la letra "Q" para eliminarlos. Si se quiere eliminar una conexión central se debe de borrar de la misma forma pero con la letra "W" La tabla siguiente muestra las letras del teclado que se utilizan para crear y romper los dos tipos de conexiones de los puertos.

Puertos (*ports*)	Salida-Entrada	Centrales
Desconectar	Q	W
Conectar	A	S

Tabla 1. Relación de teclas para la conexión y desconexión de puertos

2.3. Vistas del modelo.

Flexsim utiliza un ambiente de modelación tridimensional. La vista del modelo por defecto para construir modelos se llama *ortographic view* o vista ortográfica. También puedes ver el modelo en una forma más realista en la vista en perspectiva llamada *perspective view* (modelo 3D). Generalmente es más fácil construir el modelo con la vista ortográfica y utilizar la vista en perspectiva cuando quieras ver o mostrar el modelo cuando corres la simulación. De todas maneras puedes utilizar cualquiera de estas dos vistas ya sea para construir o para correr el modelo. Puedes abrir cuantas ventanas de vistas desees en Flexsim simultáneamente y cada una puede mostrar partes diferentes del modelo.

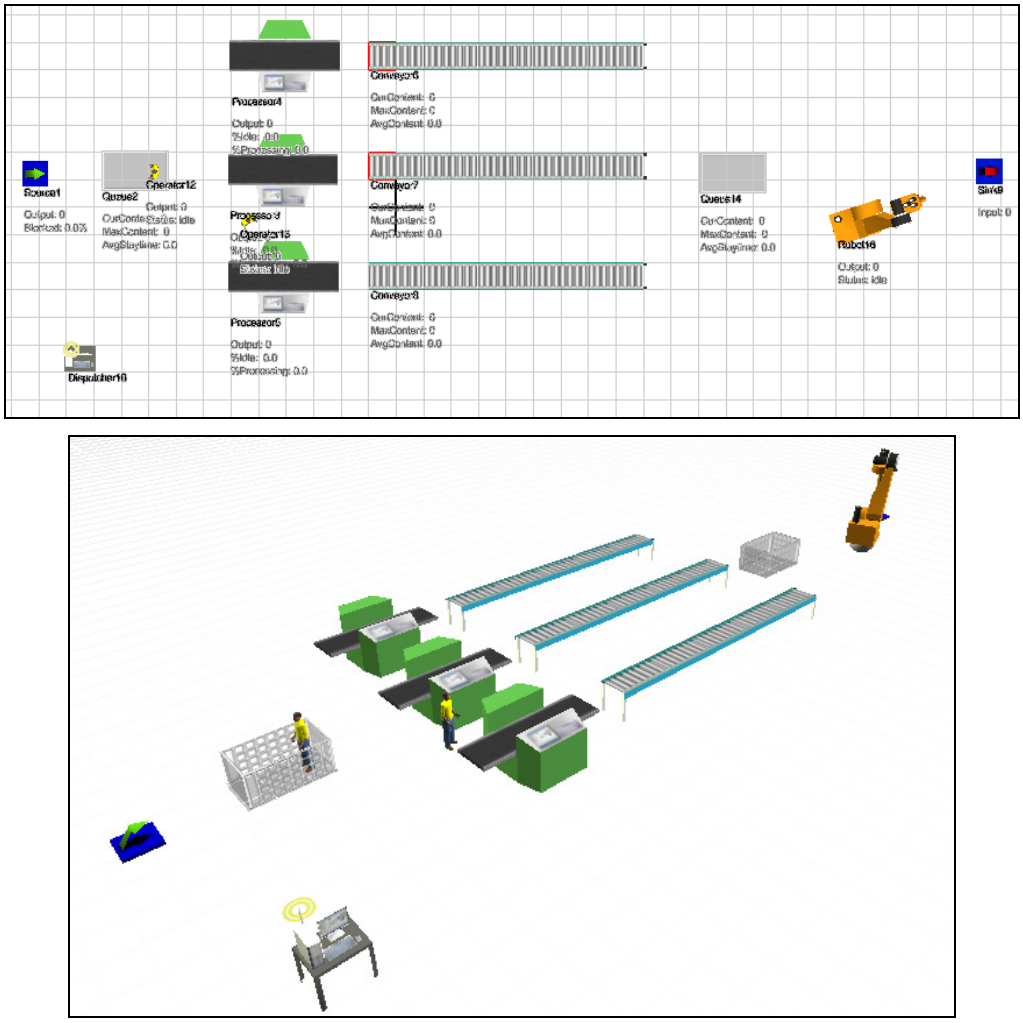


Figura 2. Vistas 2D (*ortographic view*) y 3D (*perspective view*) de un modelo en Flexsim.

2.4. Instalación del software y apertura de un modelo de prueba

Para la realización de la práctica se va a emplear la versión demo. Esta versión tiene limitada el número de objetos que se pueden insertar en un modelo. Para la instalación, hay que bajarse el archivo .zip proporcionado por el profesor y descomprimrlo. Dentro de este archivo, se encuentra una carpeta con el nombre "Flexsim_4.32_Install", que tiene dentro el archivo de intalación "FlexsimInstallation.exe". Una vez instalado, se creará un icono en tu escritorio con la siguiente imagen:

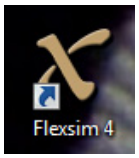


Figura 3. Icono del software Flexsim.

Además del software de instalación, también se ha dejado un modelo de ejemplo. El archivo se denomina "BlowMoulding3.zip". Al descomprimirlo, aparecen muchos archivos, aunque el modelo Flexsim es el archivo "X_Bottling Line.fsm". Para abrirlo, ejecutar el software demo y aparecerá una interfaz como la mostrada en la Figura 4.

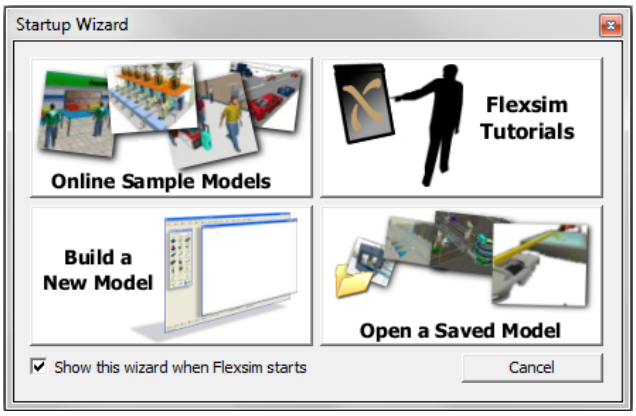


Figura 4. Interfaz de inicio del software Flexsim.

Haciendo *click* en *Open a Saved Model* y abriendo el archivo de ejemplo, se podrá visualizar la simulación de un proceso industrial.

Figura 5. Simulación de un proceso industrial.

2. 3. Objetos utilizados para construir un modelo. Propiedades y parámetros.

Flexsim posee una serie de librerías de objetos para simular procesos industriales. La versión demo que se va a utilizar para esta práctica posee disponibles dos librerías: *Discrete Objects* y *Fluid Objects*. En la práctica se va a emplear sólo la primera, ya que es la que se emplea para la simulación procesos industriales en general. La Figura 6 muestra las clases más utilizadas de la librería de componentes discretos y un árbol de clases donde se muestra la estructura de dicha librería.

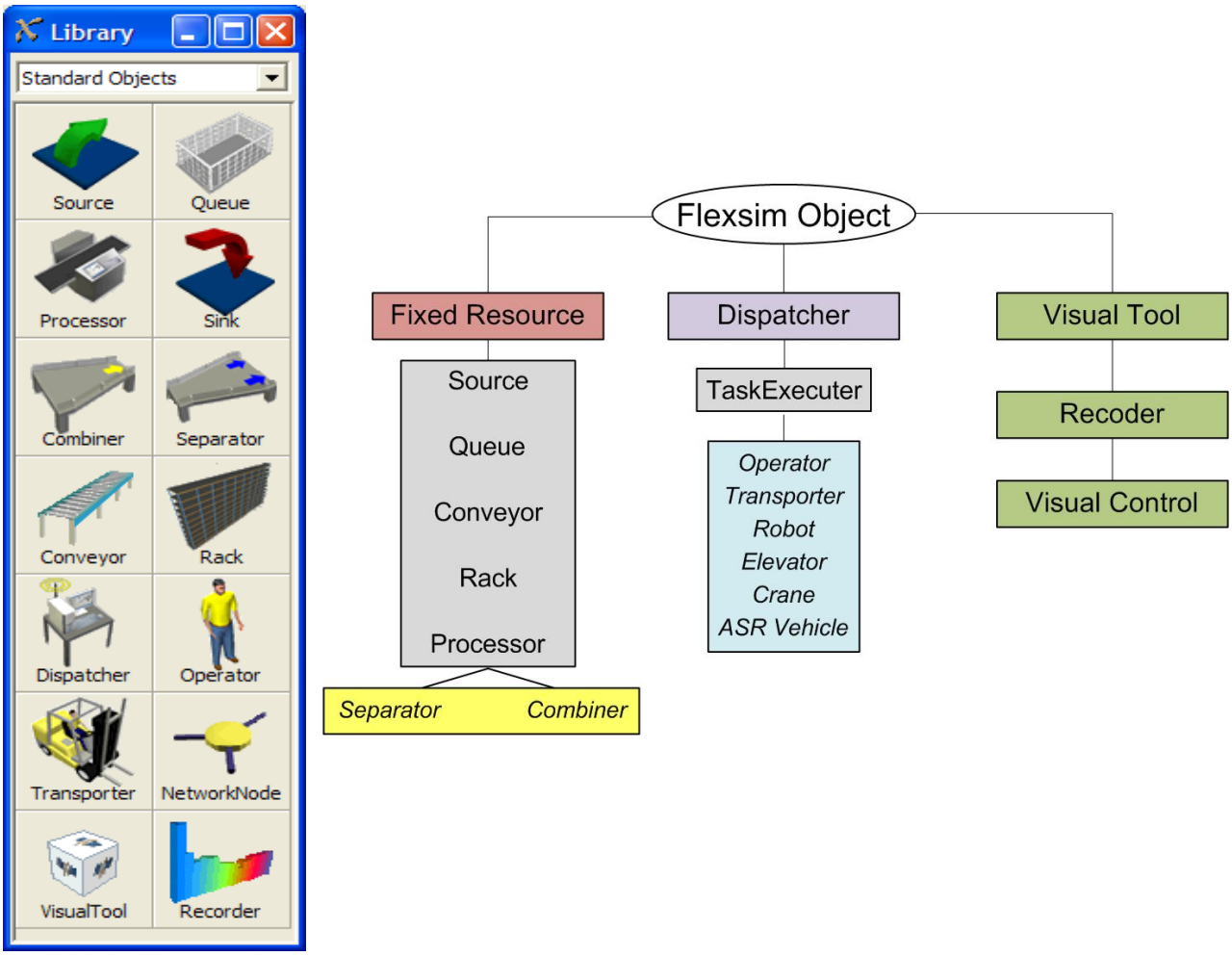


Figura 6. Librería de componentes discretos.

Por otro lado, cada objeto de Flexsim tiene una ventana de propiedades y otra de parámetros. Es importante conocer completamente la diferencia entre las propiedades y los parámetros de un objeto. Para acceder a las propiedades realiza un *click* con el botón derecho del mouse sobre algún objeto que tengas en el *layout* de un modelo y selecciona la opción de propiedades llamada *properties*. La ventana de Propiedades de los Objetos es la misma para todos los objetos de Flexsim. Esta ventana contiene 4 pestañas: *Visual*, *General*, *Labels* y *Statistics*. Cada pestaña contiene información referente al objeto de Flexsim que seleccionaste.

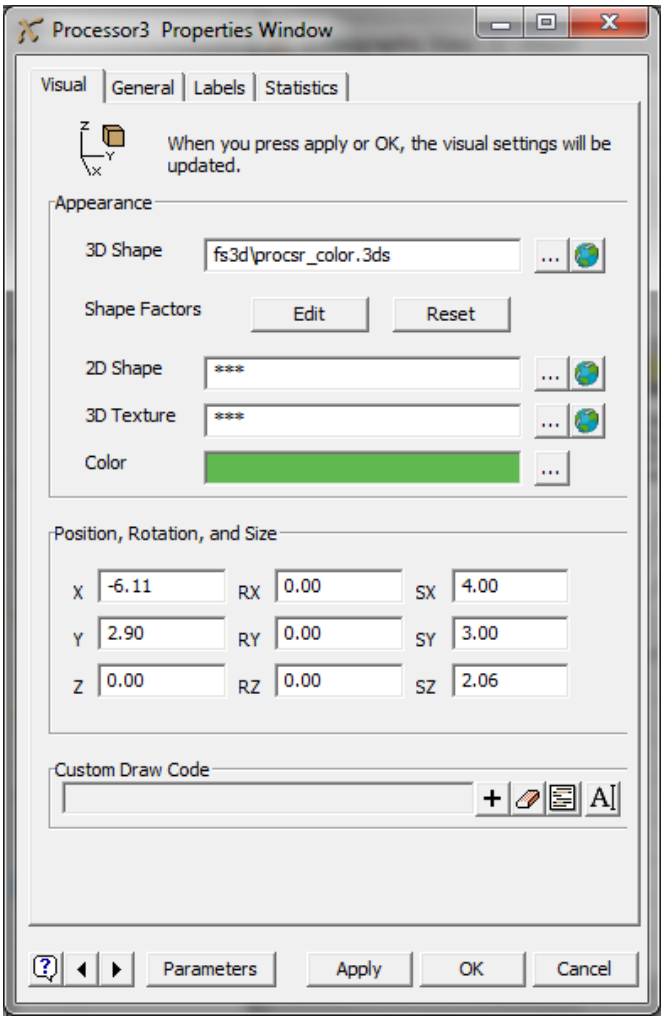


Figura 7. Propiedades de un objeto tipo Processor.

- <u>Propiedades visuales (*Visual*)</u>: esta pestaña permite modificar aspectos visuales tales como la forma 3D (se puede cambiar la forma 3D de cualquier objeto de Flexsim), texturas 3D, color, posición, tamaño y rotación. Es posible cambiar los valores de la posición, tamaño, rotación ya sea desde esta pestaña o bien utilizando el mouse en el objeto cuando este se encuentra en el *layout*. En el apartado *3D Shape* es posible introducir archivos gráficos tipo VRML y 3DS, previamente creados con un software de diseño 3D.

- <u>Propiedades generales (*General*)</u>: esta pestaña muestra información del objeto seleccionado como el nombre (se puede cambiar el nombre si lo deseas), tipo de objeto, ubicación, conexiones de los puertos, opciones y la descripción del usuario.

- <u>Propiedades de las etiquetas (*Labels*)</u>: esta pestaña muestra las etiquetas definidas por el usuario que se le han añadido al objeto seleccionado. Las etiquetas son un mecanismo mediante el cual puedes almacenar información. Existen dos partes de una etiqueta o *label*, que son su nombre y su valor. El nombre puede ser el que quieras (letras y números) pero no se debe utilizar acentos, la letra ñ, puntos o guiones medios (guiones bajos si son permitidos). Es válido poner espacios en blanco entre las palabras. En cuanto al valor del *label*, este puede ser numérico o alfanumérico (texto y números). Para añadir un *label* que deba contener únicamente datos numéricos, presiona el botón para añadir

etiquetas numéricas llamado *Add Number Label* que se ubica en parte inferior de la ventana. En caso de que quieras un *label* que tenga letras solas o bien letras con números presionado el botón de *label* de texto llamado *Add String Label*. Ahora puedes modificar tanto los nombres como los valores de las etiquetas desde la tabla. Las etiquetas pueden ser actualizadas, creadas o borradas dinámicamente mientras corres el modelo. Esta pestaña te mostrará los *labels* y sus valores actualizados en tiempo real cuando corras el modelo. Esta información es muy útil para la lógica del modelo. Los *labels* de esta pestaña son exclusivos para ese objeto pero también los *flowitems* o productos pueden tener sus propios *labels*.

- Propiedades estadísticas (*Statistics*): esta pestaña muestra información predefinida del objeto. Esta información se actualiza dinámicamente y se muestra mientras el modelo está en ejecución. Cuando se selecciona esta pestaña, se muestran cuatro subpestañas: *General*, *State*, *Content*, *Staytime*.

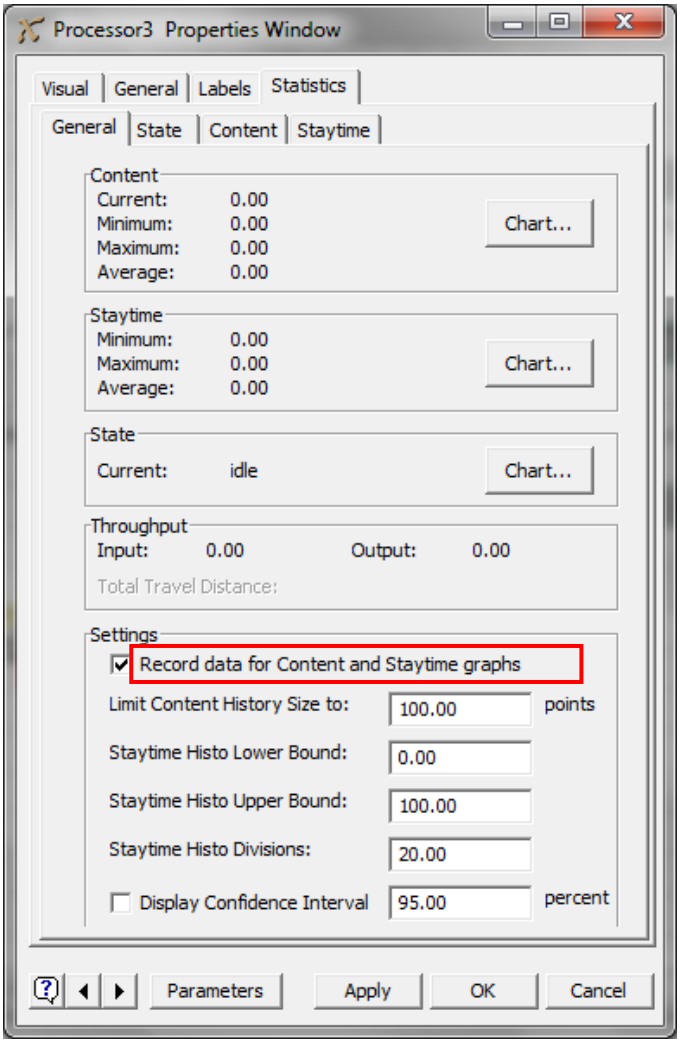


Figura 8. Propiedades de estadísticas (*Statistics*).

Dentro de esta opción, para que el objeto guarde y recopile los datos estadísticos hay que activar la opción *Record data for Content and Staytime graphs*. Para activar la recopilación de datos, también se puede realizar desde la interfaz general, en la pestaña *Statistics→Object Graph Data→Selected Objects On*. Para visualizar si está activada la recogida de datos estdísticos, hay que desactivar la opción *Hide Green Indicator Boxes*. Dentro del aparatado de estadítica, se tienen las siguientes opciones:

- ✓ Propiedades estadísticas generales (*General*): muestra información estadística acerca del contenido o inventario del objeto, el *staytime* o tiempo de permanencia de los productos en el objeto, el estado en el que se encuentran y el *throughput* del objeto indicando la cantidad de productos que entraron y que salieron de éste. La sección de la parte inferior llamada *settings* te permite definir la cantidad de datos que quieres mostrar en las gráficas de contenido y del *staytime*.
- ✓ Propiedades estadísticas de los estados (*State*): se muestra una gráfica muestra con el porcentaje del tiempo y también el tiempo en el que el objeto ha estado en cualquiera de los 21 posibles estados, como ocio, procesando, vacío, en mantenimiento, en tiempos de preparación, bloqueado, etc.
- ✓ Propiedades estadísticas de contenido (*Content*): se muestra una gráfica de contenido o inventario donde se grafica el contenido del objeto en el tiempo.
- ✓ Propiedades estadísticas del tiempo de permanencia (*Staytime*): la gráfica *Staytime* presenta un histograma que muestra la cantidad de tiempo de permanencia de los productos en el objeto.

Con respecto a los parámetros del objeto, esta interfaz sí puede variar puede variar un poco dependiendo del objeto que se seleccione. Debido a que cada objeto tiene una función específica dentro del modelo, los parámetros están personalizados para permitir al usuario que use el objeto de la forma más flexible posible. Ciertas pestañas serán iguales para todos los objetos mientras que otras serán específicas del objeto. Para acceder a los parámetros del objeto, bien se puede hacer un doble *click* con el botón izquierdo del ratón sobre el objeto, o bien con el botón derecho, accediendo a *Parameters* del menú flotante.

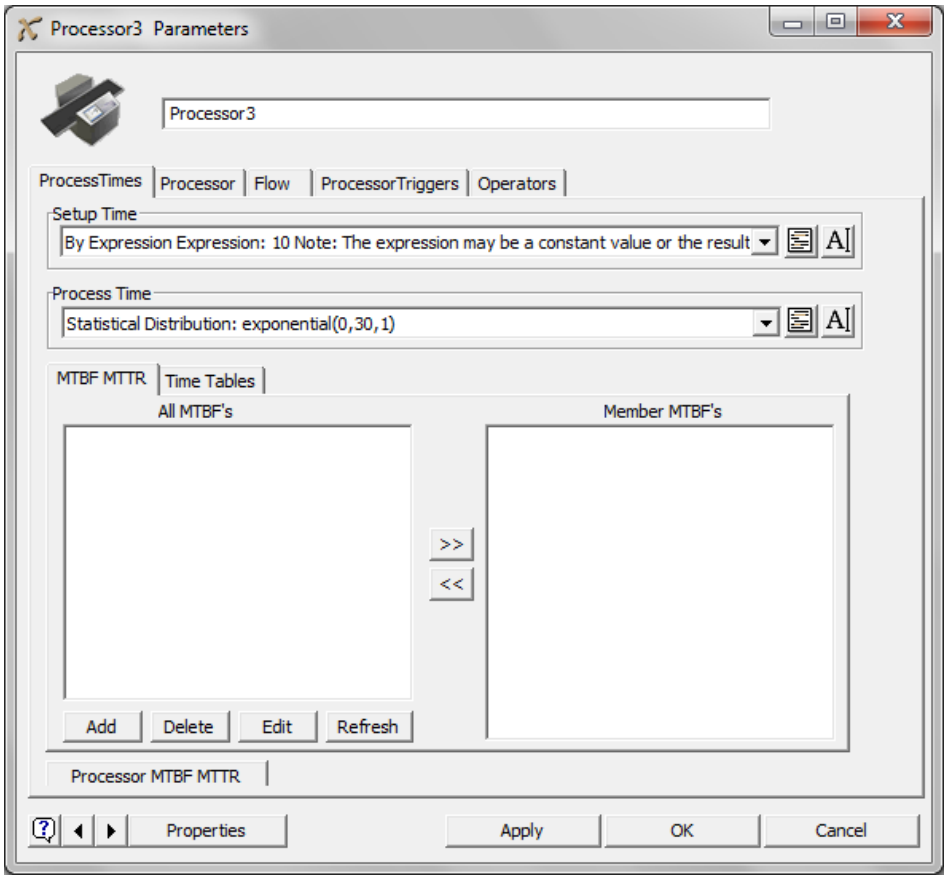


Figura 9. Parámetros de un objeto tipo *Processor*.

Finalmente, en este apartado se van a comentar cosas relativas al manejo básico del entorno y de los objetos contenidos en el modelo, tales como el posicionamiento y rotación de los objetos y su dimensionamiento en el entorno.

- Movimiento de traslación de la vista: *click* botón izquierdo y arrastrar.
- Rotación de la vista: *click* botón derecho y arrastrar.
- Zoom de la vista: *click* en rueda central (*scroll wheel*) y girar.
- Traslación XY de objetos: seleccionar un objeto y arrastrando mientras se hace *click* en el botón izquierdo.
- Traslación Z de objetos: seleccionar un objeto y haciendo *click* en la rueda central y girando para trasladarlo en el eje Z.
- Rotación XYZ de objetos: seleccionar un objeto, haciendo *click* en botón derecho y arrastrando sobre alguno de los ejes XYZ.
- Dimensionamiento XYZ de objetos: seleccionar un objeto, y en la caja amarilla que surge al seleccionarlo, hay unos ejes en las direcciones XYZ. Seleccionando uno de esos ejes XYZ, haciendo *click* con botón izquierdo y arrastrando se cambia el tamaño del objeto en el eje seleccionado.

3. Modelo de prueba: simulación de un proceso industrial básico

Es conveniente que realicéis este ejercicio de prueba que consiste en un modelo simple con 3 diferentes tipos de productos, cada uno con su ruta específica. Los objetos que se usarán en este modelo serán el *Source*, *Queue*, *Processor*, *Conveyor* y el *Sink*. También se mostrará una introducción a las estadísticas básicas y los parámetros de cada objeto serán explicados. Además, se añadirán operadores (*Operators*) y manipulación al proceso (*Robot*). También se explicará cómo cambiar la forma 3D de los procesadores o máquinas a partir de un modelo bajado de la web de Flexsim.

3.1. Construyendo el modelo.

Abre el software Flexsim mediante en el icono de acceso que se creó automáticamente en el escritorio. Una vez que se haya abierto el programa, aparecerá una pantalla comentando que es un software demo. Aquí, hay que hacer *click* a Aceptar. Posteriormente surgirá una interfaz que presenta 4 opciones: *Online Sample Models*, *Flexsim Tutorials*, *Build a New Model* y *Open a Saved Model*. Se toma la opción de *Build a New Model* y el programa quedará como lo muestra en la Figura 10. En la parte izquierda de la interfaz, se puede ver se la biblioteca de objetos discretos (*Discrete Objetcs*) y el *layout* del proceso con la vista ortográfica del modelo (*Orthographic Model View*).

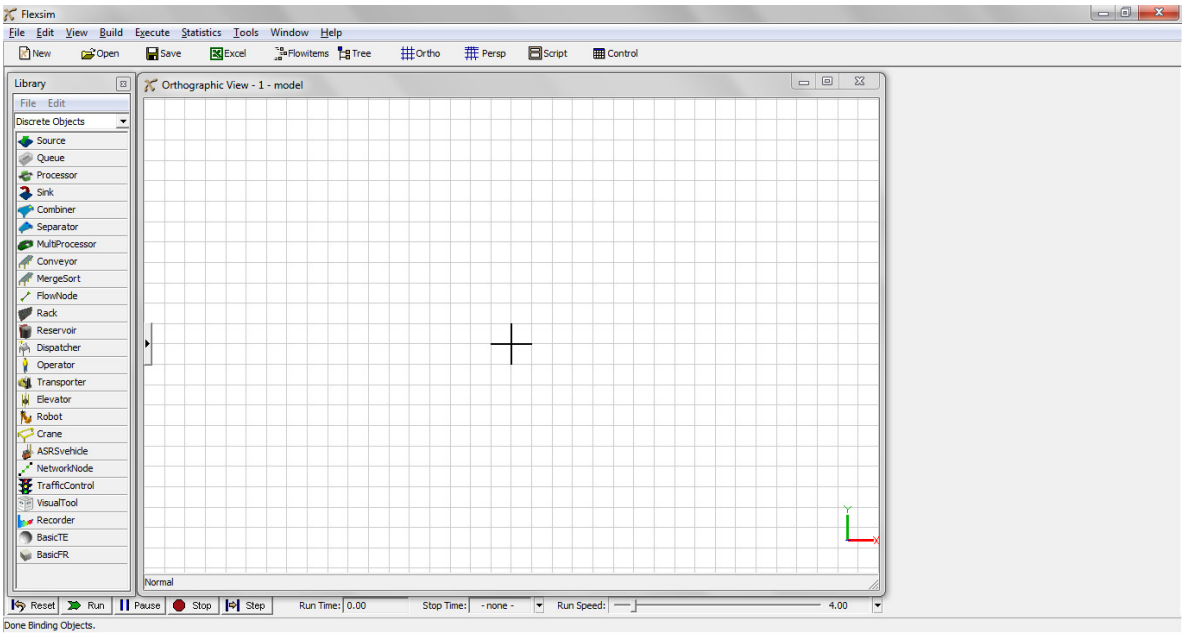


Figura 10. Creación de un nuevo modelo en Flexsim.

A continuación arrastra y suelta los componentes de la biblioteca de objetos en la vista ortográfica hasta obtener el modelo que se presenta en la Figura 11. En ella, hay un objeto tipo *Source*, un *Queue*, tres objetos tipo *Processor*, tres objetos tipo *Conveyor* y un objeto tipo *Sink*. Es conveniente desactivar la opción *Snap to Grid* de las propiedades del layout para poder posicionar los objetos más adecuadamente.

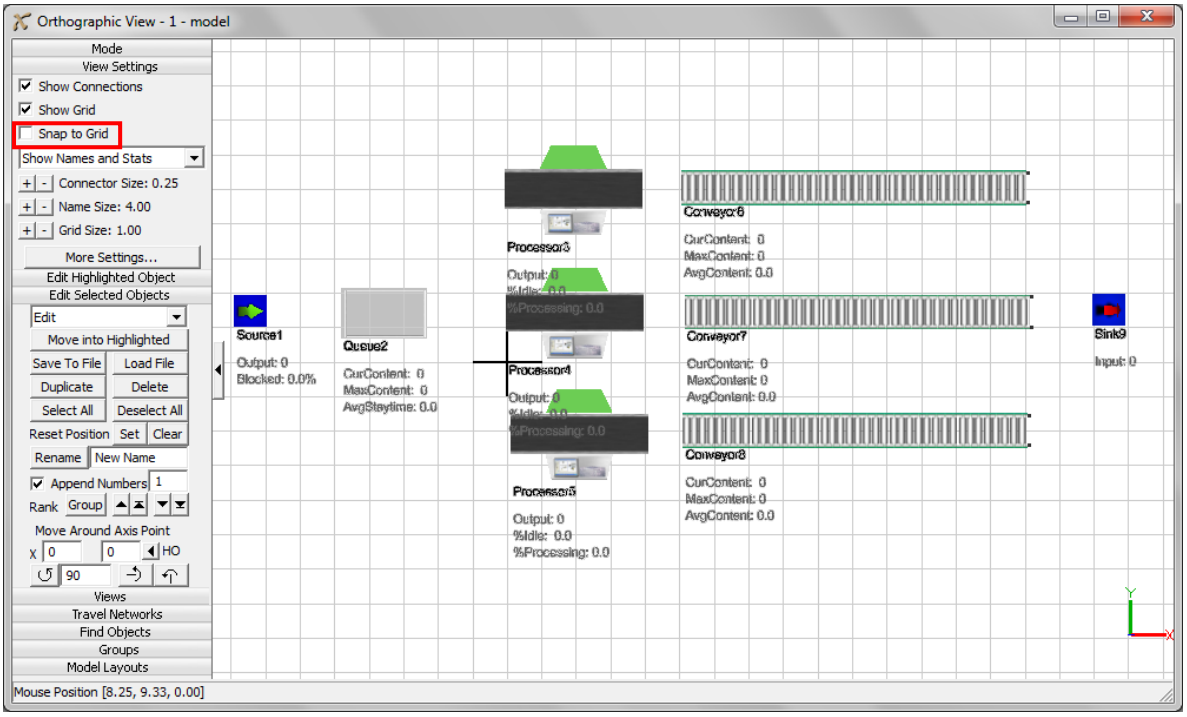


Figura 11. Inserción de los primeros objetos del modelo en el *layout*.

3.2. Conectando los puertos.

El siguiente paso es conectar los puertos para definir la ruta de los *flowitems* (puertos de entrada y salida). Eso se realiza manteniendo presionada la letra "A" del teclado, sin soltarla y mediante un *click* con el botón izquierdo sobre el objeto origen y arrastrando el

ratón hasta el objeto destino. Al terminar deberá haber una línea de conexión negra entre los objetos. Para este modelo, conectas los puertos del *Source* con el *Queue*, el *Queue* con cada uno de los *Processors* (aquí es conveniente que sigas un orden, comenzando por arriba por ejemplo), cada uno de los *Processors* con su *Conveyor* correspondiente y todos los *Conveyors* con el *Sink*. La Figura 12 muestra cómo debería quedar el layout después de conectar los puertos de entrada-salida.

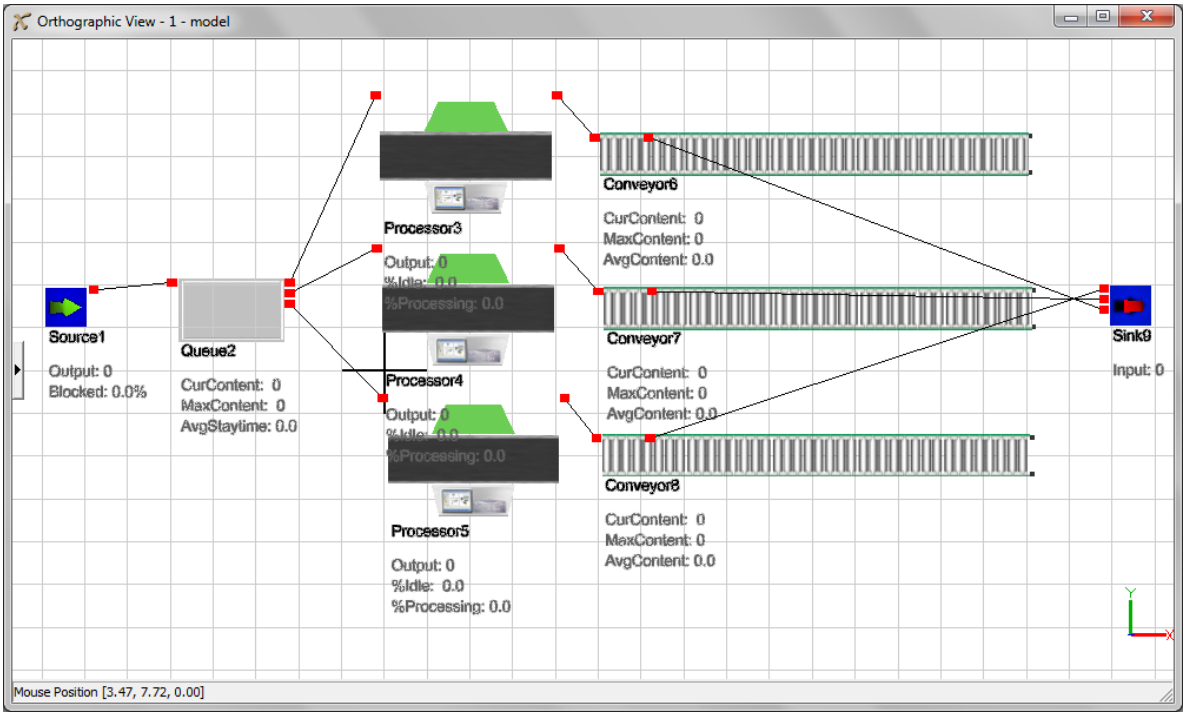


Figura 12. Inserción de los primeros objetos del modelo en el *layout*.

3.3. Configurando la simulación.

Cada objeto tiene su propia interfaz gráfica del usuario, que se utiliza para añadir los datos y la lógica al objeto. Haciendo un doble *click* sobre un objeto se accede a los parámetros del objeto. Para este modelo, se desea que tres diferentes tipos de productos se introduzcan en la simulación. Para hacer esto, a cada producto o *flowitem* se le asignará un *itemtype* o tipo de producto que contendrá un valor numérico entero comprendido entre uno y tres utilizando una distribución uniforme que escoja ese valor. Esto será realizado utilizando el *Trigger* de salida (*OnExit*) del *Source*. Para ello, se accede a los parámetros del *Source* (Figura 13).

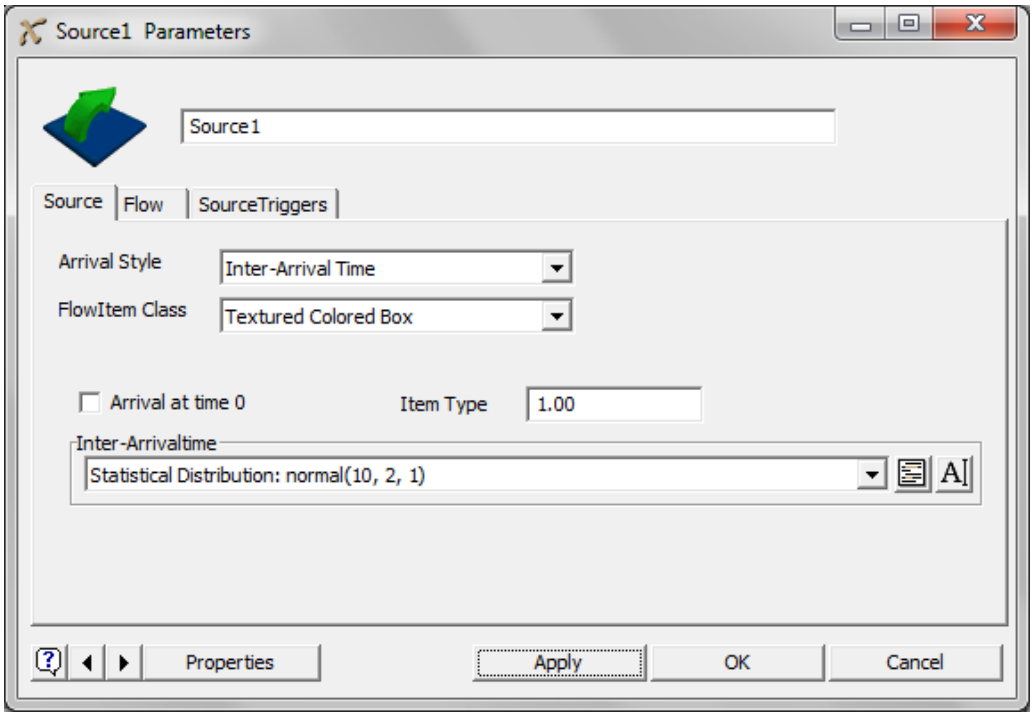


Figura 13. Interfaz de parámetros del Source.

Para este modelo a simular, será necesario cambiar el tiempo entre *flowitems* generados por el objeto o *Inter-Arrivaltime*, y también el tipo de producto o *itemtype* para que se generen 3 tipos diferentes de productos. Para cambiar el tiempo entre flowintems generados, seleccionamos una distribución *normal(10,2)* – 10 de media, 2 de desviación típica. Para ello, selecciona el desplegable que existe en esta opción y seleccionamos, dentro de *Statistical Distribution* la opción de *normal*, y se cambian los valores a los deseados.

Dentro de esta opción, también se puede seleccionar el tipo de generación de flowitems (tiempo, secuancia, horario) y el tipo de flowitems generado (*Flowitem Class*), donde se pueden seleccioanar varias formas que ofrece el software. Si se desea añadir un nuevo *flowitem*, se puede realizar accediendo a la opción del menú principal *Tools→Flowitem Bin*.

Lo siguiente a realizar es asignar el valor del número de *itemtype* o tipo de producto a los productos que se vayan introduciendo en el sistema. El valor es distribuido uniformemente entre un valor de 1 hasta 3 (3 tipos diferentes de productos). La forma e hacerlo sería cambiar el *itemtype* el el *trigger OnExit* del *Source*. Para ello, en la pestaña *SourceTriggers* de la interfaz de parámetros y despliega la lista de opciones del *Trigger OnExit* (de salida). Selecciona la opción de cambiar el itemtype de los *flowitem* y su color, que se llama "*Set Itemtype and Color*". Por defecto, asigna al itemtype la salida de una función tipo *duniform(1,3)*, que es distribución que devuelve un número entero comprendido entre el rango que se le indica.

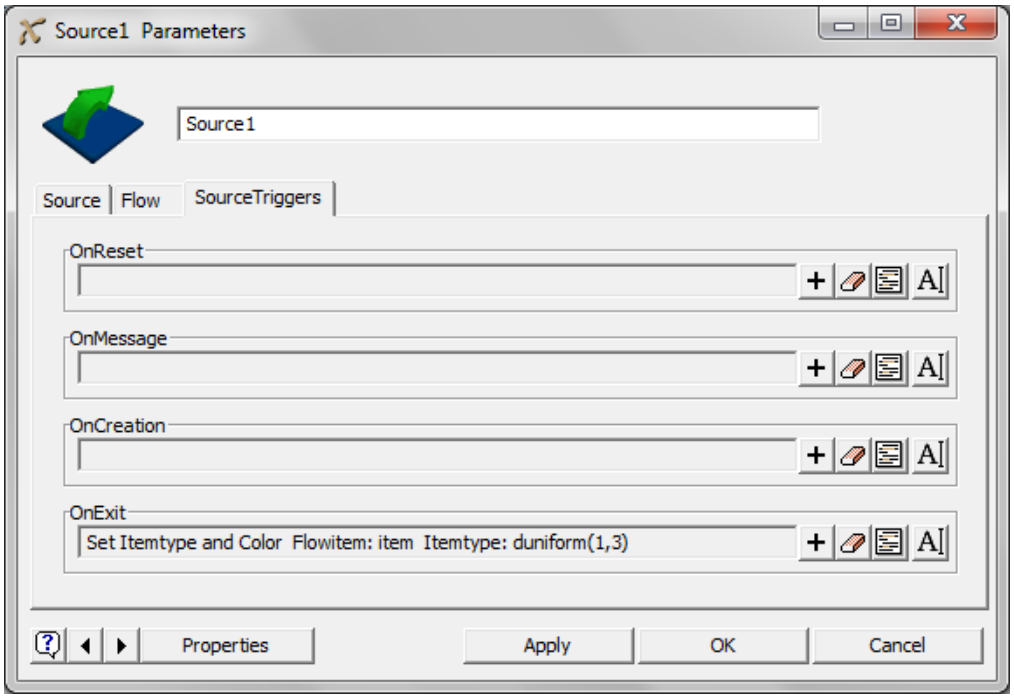


Figura 14. Configuración del trigger OnExit.

El siguiente paso será detallar el *Queue*. Debido a que el *Queue* es un lugar donde se pueden acumular los *flowitems* hasta que puedan ser procesados por el *Processor*, existen dos cosas que se necesitan hacer. Primero, debemos definir la capacidad del *Queue* para que pueda contener hasta 25 *flowitems*. Segundo, se debe definir la regla del flujo para que los *flowitems* cuyo tipo de producto o *itemtype* sea 1 vayan al *Processor* 1, si su *itemtype* es 2 vayan al *Processor* 2 y si es 3 al *Processor* 3. Para ello, tan sólo será necesario cambiar el contenido máximo (*Maximum Content*) a 25 unidades (Figura 15).

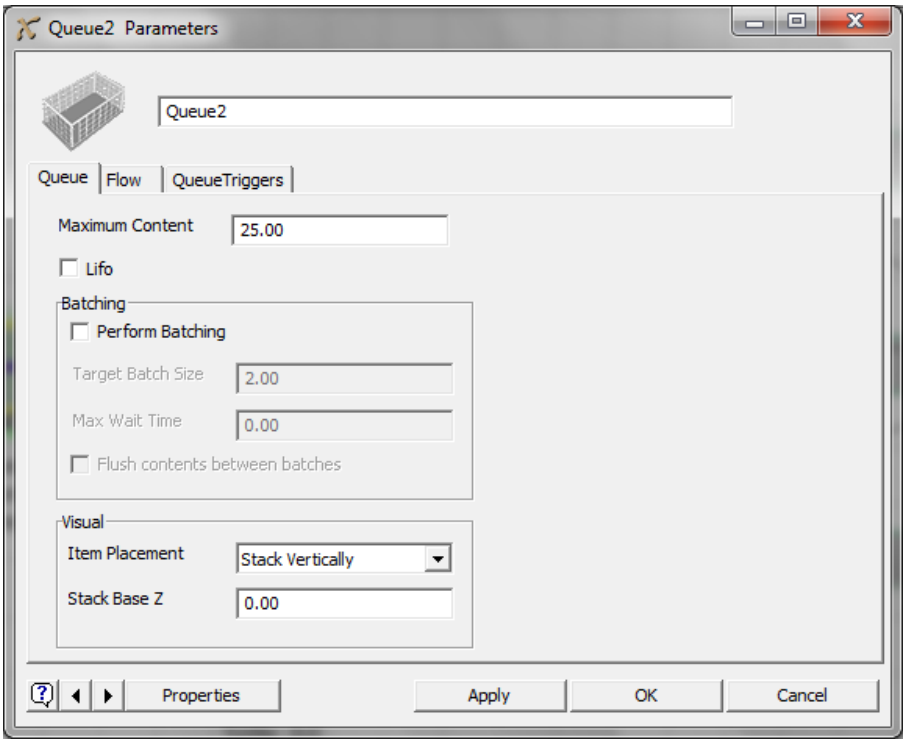


Figura 15. Configuración del Queue.

Para el flujo de salida del *Queue*, hay que dirigirse a la pestaña de flujo (*Flow*) de la ventana de parámetros para definir las opciones del flujo. Se desea enviar el *itemtype* 1 al *Processor* 1, el *itemtype* 2 al *Processor* 2, y el *itemtype* 3 al *Processor* 3. Dado que los puertos de sallida del *Queue* fueron conectados en orden a los *Processors*, hay que indicar que los *flowitems* salgan por el puerto que sea igual a su *itemtype*. Para este fin, hay seleccionar de la lista desplegable de *Output*, la opción de *By Expression*, y en en la parte de *Output Port*, la función de *getitemtype(item)*, que coge el valor del tipo del flowitems que va a salir (Figura 16).

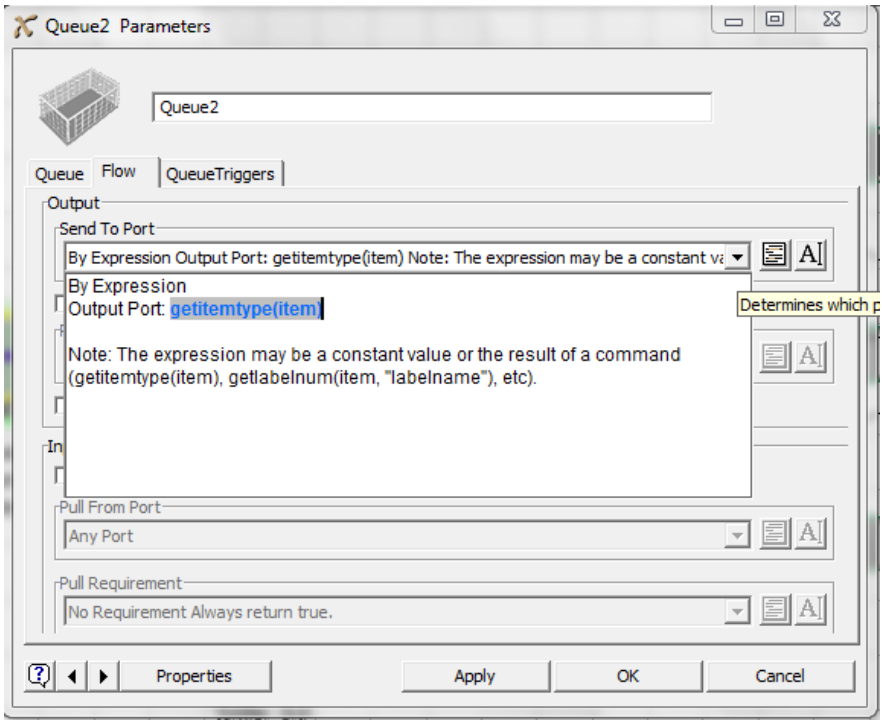


Figura 16. Configuración del flujo de salida del Queue.

A continuación, se va terminar de configurar el *Processor* o máquina. En primer lugar, se va a cambiar la forma 3D del objeto. Para ello, se accede a sus propiedades y en la opción de *Appearance*, en *3D Shape*, se accede al icono de la web (véase Figura 17) y se baja un archivo de la parte de *Processors* (*3D Shapes for Processors*). La ruta del archivo bajado (.wrl, .3ds, etc) se tiene que insertar en la opción de *3D Shape*. De esta manera, cambiará el aspecto de la máquina.

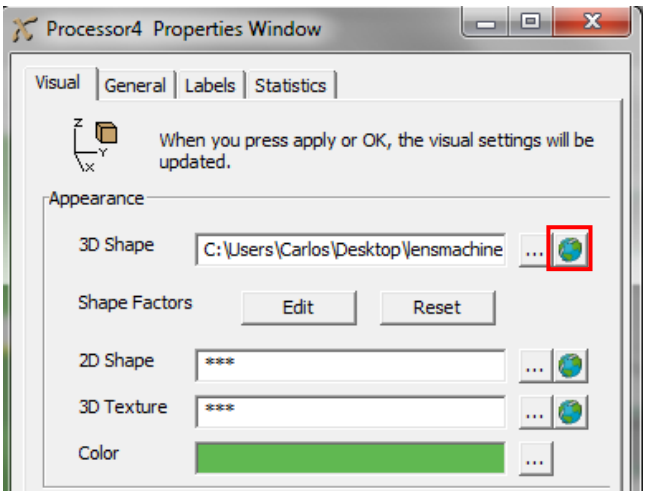


Figura 17. Cambio de la forma 3D del *Processor*

Finalmente, se va a configurar el tiempo de proceso del *Processor* o máquina. Este objeto tiene dos tipos de tiempo, el *Setup Time*, o tiempo de preparación, y el *Process Time* o tiempo de proceso. La configuración que se va a establecer es un tiempo constante de 10 seg. para el *Setup Time* (mediante la opción *By Expression*), y para el *Process Time*, una distribución estadística tipo exponencial, de media 30, tal y como muestra la Figura 18.

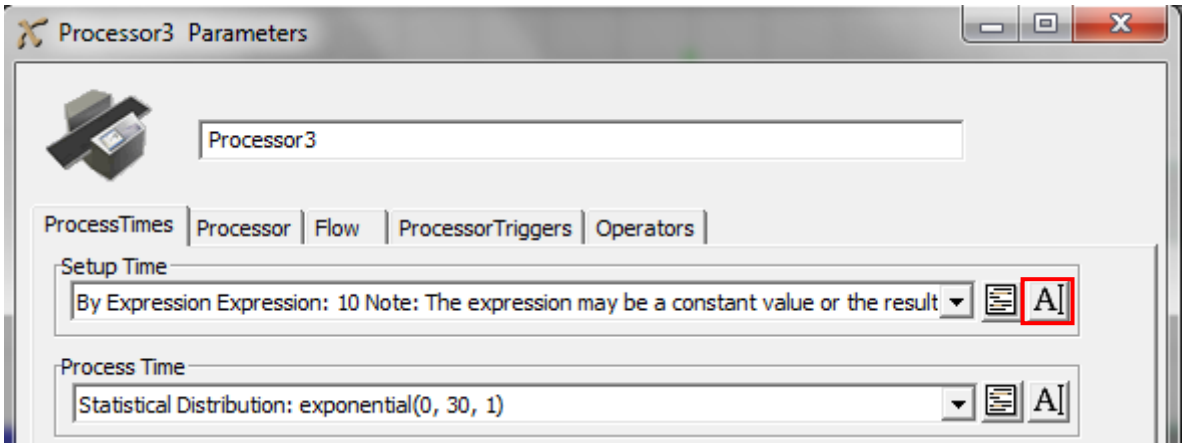


Figura 18. Configuración del *Processor*.

Después de haber configurado los objetos comentado, es posible ejecutar la simulación y visualizar cómo se simula el modelo elaborado. Para controlar la simulación del modelo, hay una serie de controles en la parte inferior de la interfaz (*Reset*, *Run*, *Pause*, *Stop*, *Step*) Para ver mejor la simulación, se puede acceder a la viste en perspectiva o 3D (Figura 19).

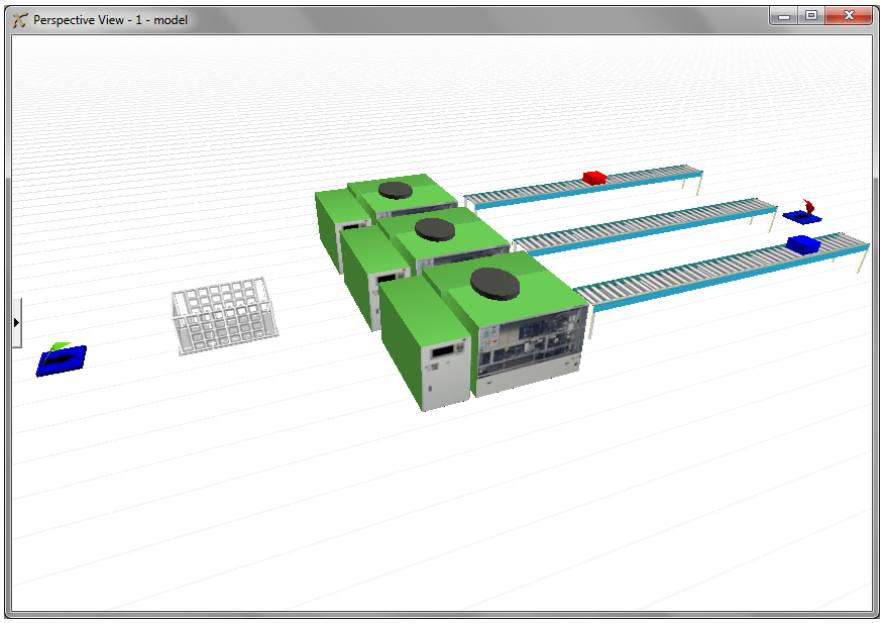


Figura 19. Imagen en 3D del modelo Flexsim configurado en ejecución.

Nota: hasta el momento, se ha configurado los parámetros de los objetos mediante las opciones que proporciona el software. Es posible también programar rutinas para la configuración. Para editar el código de una parametrización concreta de un objeto, hay que hacer *click* en el botón A que hay en cada opción (véase Figura 18). En el anexo de esta práctica, se explica brevemente algunas de las funciones y nomenclatura del lenguaje (que realmente en código C++).

3.4. Añadiendo operarios y un sistema robótico para transportar objetos.

Se va añadir un operador para para que recoja el *flowitem* del *Queue* y lo lleve a una de las máquinas. Para ello, se inserta un objeto *Operator* y se conecta mediante puertos centrales con el *Queue*. Los puertos centrales se utilizan para el transporte de *flowitems* mediante recursos compartidos. El paso siguiente es modificar los parámetros en el flujo (*Flow*) del *Queue* para que utilice al operador para hacer el transporte de los productos. Esto lo se realizará en la ventana de Parámetros, en la pestaña llamada *Flow*. Activa la opción de usar transporte llamada *Use Transport* que se encuentra debajo de la lista *Send to Port*. En el momento en que se active la opción *Use Transport* aparecerá una nueva lista llamada *Request Transport From*. Esta lista permite que se defina que operador, montacargas o transporte utilizarás para mover el producto basándose en el número de puerto central. En este caso el puerto central número 1 está conectado al *Operator*.

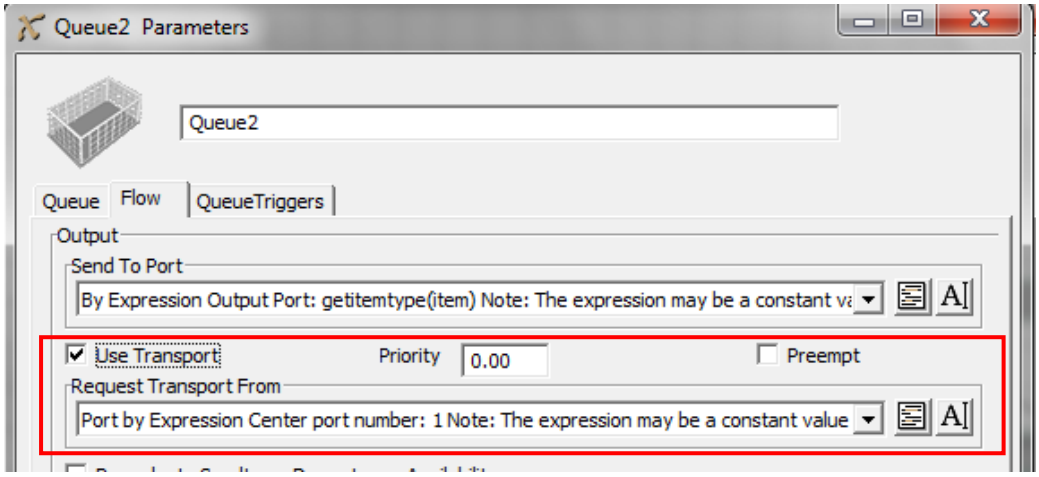


Figura 20. Configuración del Queue para usar el operador.

A continuación, también se desea que las máquinas utilicen al operador durante el *Setup*. Para ello, se debe crear una conexión entre los puertos centrales de cada estación y el puerto central del operador. A continuación, abre la pestaña llamada *Operators* del *Processor*. Activa la opción para que use el operador para el *Setup* simplemente activando Use Operator(s) for Setup. Cuando se active esta opción aparecerá un campo llamado *Number of Operators* y una lista para definir a quien llamar. El número de operadores que se necesita para el *Setup* de la estación es de 1 y en la lista *Pick Operator* se debe llamar al puerto central 1 (es decir, al operador que ya se conectó por el puerto central 1).

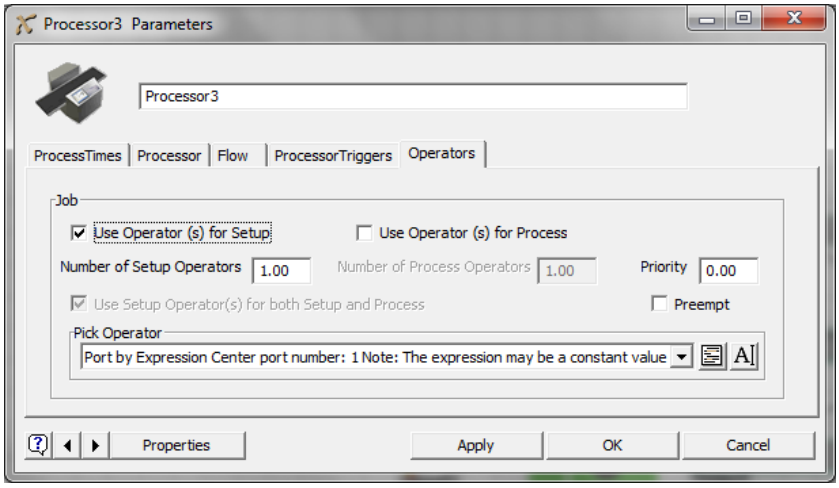


Figura 21. Configuración del Processor para utilizar el Operator en el Setup.

Finalmente, se va insertar un sistema robótico que manipule los *flowitems* y los lleve al *Sink* para finalizar el proceso . Previamente, es conveniente añadir un *Queue* al final de los *Conveyors* para que el sistema robótico pueda coger los elementos uno a uno. El layout nuevo tendrá que incorporar el *Queue* al final de las cintas y el objeto *Robot* entre el *Queue* y el *Sink*. Será necesario desconectar las anteriores conexiones realizadas y realizar las nuevas, es decir, puertas de salida de las cintas con los de entrada del *Queue*, el central del *Queue* con el objeto *Robot*, y la salida-entrada entre el *Queue* y el *Sink*. También habrá que definir el uso de transporte para recoger los objetos del *Queue*. La Figura 22 muestra el aspecto final de la simulación en ejecución de este ejemplo propuesto.

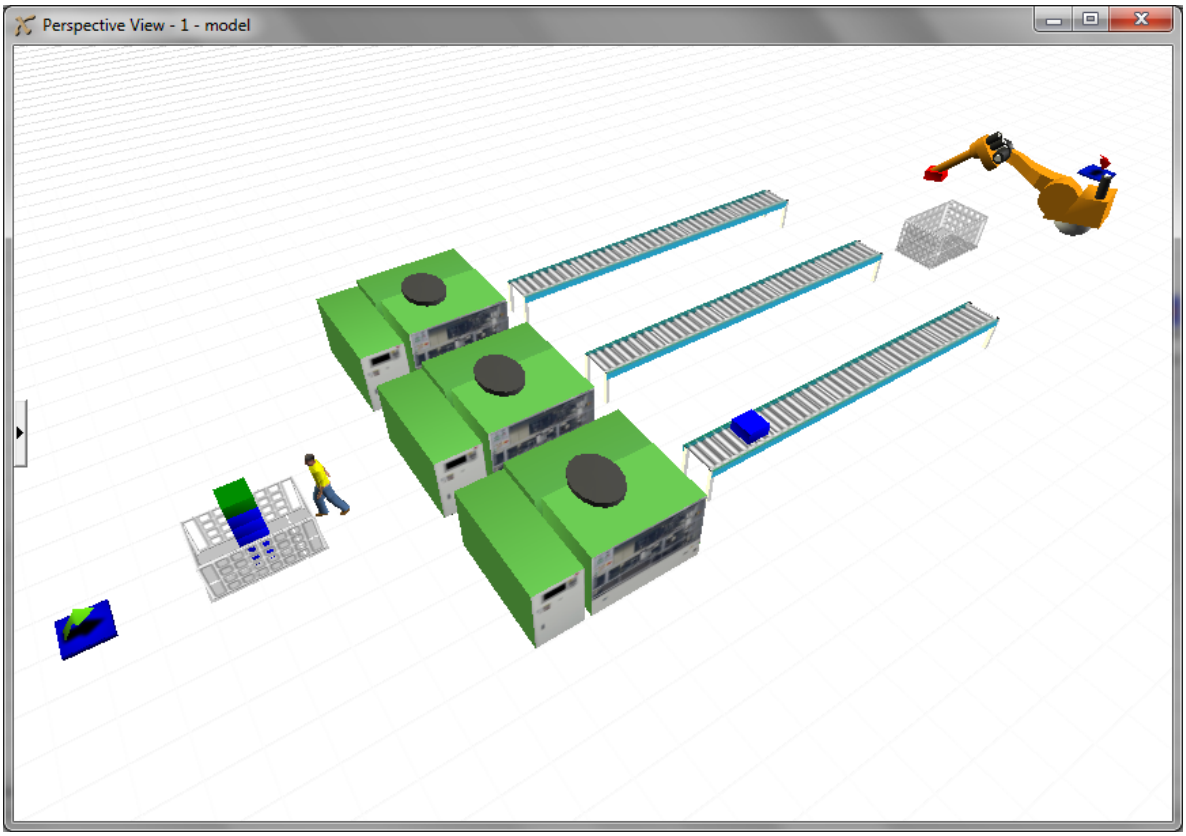


Figura 22. Simulación del ejemplo propuesto.

4. Ejercicio de la práctica

El ejercicio de la práctica consistirá en realizar una simulación de un sistema de paletizado. Para ello se utilizará una librería más completa sobre sistemas robóticos, ya que el robot que se ha mostrado anteriormente es de la librería de *Discrete Objects*. La configuración que permite este robot es bastante básica y no es posible configurar ningún movimiento programado. Para poder manejar un sistema robótico y realizar operaciones más reales de carga y descarga de elementos, se va a emplear esta librería adicional denominada *Articulated Robot (AR)*, que simula un sistema robótico de 6 GDL tipo FANUC.

4.1. Instalación de la librería *Articulated Robot*.

En este apartado, se va a describir cómo instalar la librería *AR* como un plug-in adicional en el software. En primer lugar, hay que bajársela de la web de Flexsim, en concreto del enlace http://www.flexsim.com/community/forum/downloads.php?do=cat&id=26. Básicamente, para instalar la librería es necesario descomprimir el archivo bajado y copiar la

información extraída en la carpeta *libraries* de la ruta de dónde esté instalado el software en el disco duro. Una vez copiada a esta carpeta, se debe importar al sistema desde el visor de librerías mediante *File→Load Libraries*. Al cargar la librería *AR*, deberá aparecer un icono como el que se muestra en la Figura 23.

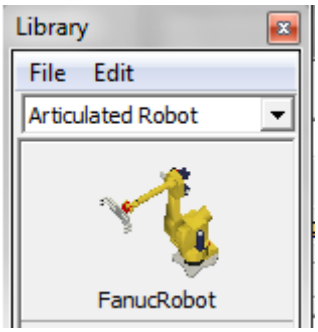


Figura 23. Icono de la librería *Articulated Robot*

Dentro del paquete bajado, se encuentra un pequeño manual para saber cómo funciona este *driver* del robot. Será tarea de la práctica conocer el funcionamiento del robot FANUC mediante dicho manual.

4.2. Tareas de la práctica-descripción del proceso.

El proceso a simular consistirá en un sistema de paletizado de dos productos distintos. Los *items* que debe generar la fuente deben ser del tipo *Texture Colored Box*, cada uno de diferente color y ambos con un tamaño de [2.26; 0.90; 1.11]. Para cambiar las propiedades de los *flowitems*, hay que dirigirse a *Tools→Flowitem Bin* y editar sus propiedades. Dichos productos, a través de una cinta transportadora (a una velocidad de 1 u/s), se deben dirigir hacia una celda robotizada mediante el robot FANUC de 6GDL. Antes de ser manipulados, la cinta transportadora se subdividirá en dos cintas con una velocidad más lenta (0.5 u/s), que separa los productos por su color. El ítem es manipulado por el robot y puesto es un palet con una capacidad de 8 productos. Se deben almacenar en dos columnas de 4 items cada uno. Dicho palet, deberá ser generado por otra fuente, que estará conectado a un *Combiner* de capacidad 8 y en almacenaje en modo *Pack* (véase las propiedades y configuración de este objeto en Flexsim). El *Combiner* será mejor que se haga invisible y que directamente pongáis el palet en su posición. Una vez terminado el palet, deberá ser recogido por un objeto tipo *Transporter* (monta cargas), que lo llevará a su empaquetado una máquina (*Processor*), que para su visualización tendréis que emplear el archivo *palletwrapperbase.wrl* proporcionado por el profesor. Finalmente, la salida de esta máquina se debe conectar al objeto *Sink* (finalización del proceso).

Las configuraciones principales de la práctica son las siguientes:

- Fuentes. Las fuentes se configurarán con un valor constante de 0, que significa la producción continuada de elementos.
- Máquina de empaquetado. No configurar con tiempo de *Setup*, e introducir un tiempo de proceso constante de 20 seg/item.
- Robot. Mediante el driver del objeto robot se deberá configurar las trayectorias para carga y descarga de los productos en el palet de forma que realicen el paletizado correctamente (2 columnas de 4 elementos).

Notas:

- Dimensionad el *layout* (incluidos los objetos) de la manera más conveniente que veáis.

- En la siguiente imagen se muestra una distribución simple del *layout* planteado en la práctica.
- Si es necesario, emplear la programación para la configuración de los objetos (véase Anexo A de la práctica).

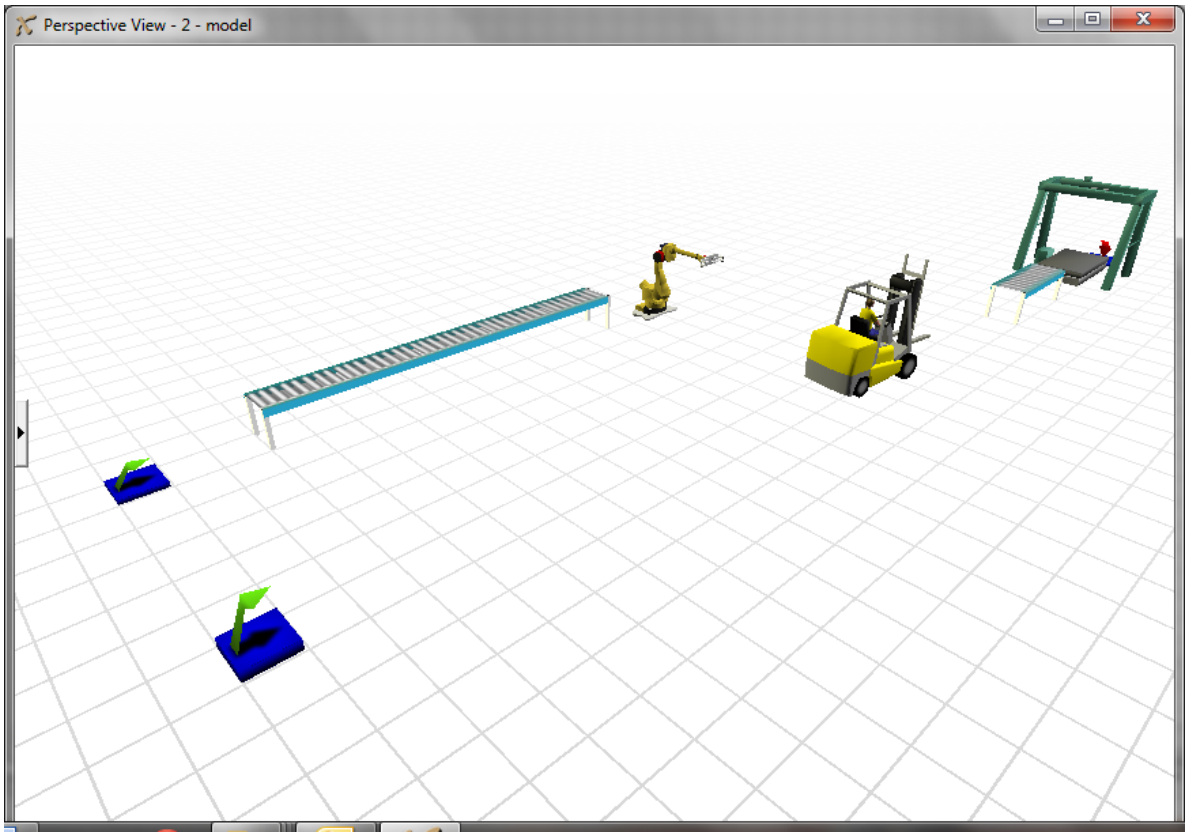
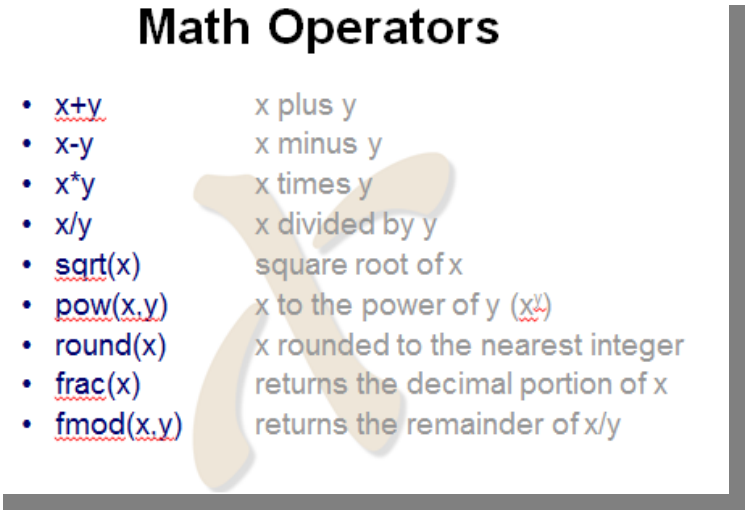


Figura 24. Layout simple del ejercicio de la práctica.

ANEXOS

A. Programación en el entorno Flexsim.

A continuación se muestran una serie de imágenes (en inglés) que describen la programación en Flexsim. Esta serie de figuras han sido extraídas de un manual de Flexsim bajado de la web principal:



Math Operators

- x+y — x plus y
- x-y — x minus y
- x*y — x times y
- x/y — x divided by y
- sqrt(x) — square root of x
- pow(x,y) — x to the power of y (x^y)
- round(x) — x rounded to the nearest integer
- frac(x) — returns the decimal portion of x
- fmod(x,y) — returns the remainder of x/y

Comparing

• x > y	x greater than y
• x < y	x less than y
• x >= y	x greater than or equal to y
• x <= y	x less than or equal to y
• x == y	x equal to y
• x != y	x not equal to y
• comparetext(string1,string2)	string1 matches string2

Relating

• &&	logical AND
• ||	logical OR
• !	logical NOT
• min(x,y)	returns minimum of x and y
• max(x,y)	returns maximum of x and y

Updating

• x = y	set x to y
• x += y	set x to x plus y
• x -= y	set x to x minus y
• x *= y	set x to x times y
• x /= y	set x to x divided by y
• x ++	add 1 to x
• x --	subtract 1 from x

Variable Types

- int — integer
- double — double precision
- string — text string
- fsnode* — pointer to a Flexsim tree node

Declaring and Setting Variables

- int index = 1;
- double weight = 175.8;
- string category = "groceries";
- fsnode* forklift = centerobject(current,1);

Basic Object Referencing

current	the current object
item	the involved flowitem
rank(*node,ranknum*)	rank(current,3)
inobject(*object,portnum*)	inobject(current,1)
outobject(*object,portnum*)	outobject(current,1)
centerobject(*object,portnum*)	centerobject(current,1)
model()	model()

Basic Object Attributes

- getname(*object*)
- setname(*object, "name"*)
- getitemtype(*object*)
- setitemtype(*object, num*)
- setcolor(*object, red, green, blue*)
- colorred(*object*) – blue,green,random...

Object Spatial Attributes

- xloc(*object*) yloc(*object*) zloc(*object*)
- setloc(*object, xnum, ynum, znum*)
- xsize(*object*) ysize(*object*) zsize(*object*)
- setsize(*object, xnum, ynum, znum*)
- xrot(*object*) yrot(*object*) zrot(*object*)
- setrot(*object, xdeg, ydeg, zdeg*)

Basic Object Statistics

- content(*object*)
- getinput(*object*)
- getoutput(*object*)

- int inventory = content(current);
- int produced = getoutput(current);

Object Labels

- syntax
  - getlabelnum(*object, "labelname"*)
  - setlabelnum(*object, "labelname", value*)
  - getlabelstr(*object, "labelname"*)
  - setlabelstr(*object, "labelname", string*)
  - label(*object, "labelname"*)
- examples
  - getlabelnum(item, "serialnumber")
  - setlabelnum(item, "serialnumber", 5)
  - getlabelstr(current, "category")
  - setlabelstr(current, "category", "groceries")
  - inc(label(item, "stepnum"), 1) only works with number data nodes

Global Tables

- gettablenum(*"tablename", rownum, colnum*)
- settablenum(*"tablename", rownum, colnum, value*)
- gettablestr(*"tablename", rownum, colnum*)
- settablestr(*"tablename", rownum, colnum, string*)

Logical 'if' Statement

```
if (test expression)
{
   code block
}
else
{
   code block
}
```

```
if (content(item) == 2)
{
   colorred(item);
}
else
{
   colorblack(item);
}
```

Logical 'while' Statement

```
while (test expression)
{
   code block
}
```

```
while (content(current) > 0)
{
   destroyobject(last(current));
}
```

Logical 'for' Statement

```
for (start expression;
     test expression;
     count expression)
{
   code block
}
```

```
for (int index = 1;
     index <= content(current);
     index++)
{
   colorblue(rank(current,index));
}
```

Logical 'switch' Statement

```
switch ( switchvariable )
{
   case casenum:
   {
    code block
    break;
   }
   default:
   {
    code block
    break;
   }
}
```

```
int type = getitemtype(item);

switch (type)
{
   case 1:
   {
      coloryellow(item);
      break;
   }
   case 5:
   {
      colorred(item);
      break;
   }
   default:
   {
      colorgreen(item);
      break;
   }
}
```

Basic Object Control

- closeinput(*object*), openinput(*object*)
- stopinput(*object*), resumeinput(*object*)
- closeoutput(*object*), openoutput(*object*)
- stopoutput(*object*), resumeoutput(*object*)
- sendmessage(*toobject, fromobject, parameter1, parameter2, parameter3*)
- senddelayedmessage(*toobject, delaytime, fromobject, parameter1, parameter2, parameter3*)
- stopobject(*object*), resumeobject(*object*)
- moveobject(*object, containerobject*)

TaskExecuter Control

- requestoperators(*taskexecuter, station, involvedobj, nrofops, priority, preempting*)
- freeoperators(*taskexecuter, involvedobj*)

- createemptytasksequence(*taskexecuter, priority, preempting*)
- inserttask(*tasksequence, tasktype, involved1, involved2, parameter1, parameter2, parameter3*)
- dispatchtasksequence(*tasksequence*)

Sample code for tasksequences

```
fsnode *forklift = centerobject(current,1);
fsnode *origin = centerobject(current,2);
fsnode *destination = centerobject(current,3);
fsnode *item = first(origin);

fsnode *ts = createemptytasksequence(trans,0,0);

inserttask(ts, TASKTYPE_TRAVEL, origin, NULL);
inserttask(ts, TASKTYPE_FRLOAD, item, pickup);
inserttask(ts, TASKTYPE_TRAVEL, destination, NULL);
inserttask(ts, TASKTYPE_FRUNLOAD, item, destination);

dispatchtasksequence(ts);
```

Task Types

TASKTYPE_LOAD: flowitem, pickup
TASKTYPE_FRLOAD: flowitem, pickup, outputport
TASKTYPE_UNLOAD: flowitem, dropoff
TASKTYPE_FRUNLOAD: flowitem, dropoff, inputport
TASKTYPE_UTILIZE: involved, station, state
TASKTYPE_STOPREQUESTFINISH: stoppedobject, NULL
TASKTYPE_TRAVEL: destination, NULL
TASKTYPE_TRAVELTOLOC: NULL, NULL, xloc,yloc,zloc, endspeed
TASKTYPE_TRAVELRELATIVE: NULL, NULL, xloc,yloc,zloc, endspeed
TASKTYPE_BREAK: NULL, NULL
TASKTYPE_DELAY: NULL, NULL, delaytime, state
TASKTYPE_SENDMESSAGE: receiver, NULL, param1,param2,param3
TASKTYPE_MOVEOBJECT: flowitem, container, port
TASKTYPE_DESTROYOBJECT: object, NULL

Brown parameters are required, Gray parameters are optional
** use profiletasksequence(ts) to print to console full description of ts