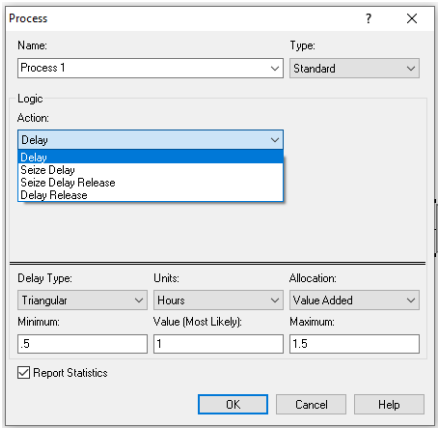
MODULO 4. PROCESOS

Hasta ahora, los bloques DECIDE, CREATE, DISPOSE y ASSIGN tenían una ocurrencia instantánea, sin demoras. Las entidades pasaban por este bloque sin agolparse, sin esperas y sin demoras. Estos bloques, por tanto, son bloques en los que no se consume tiempo, algo irreal en cualquier sistema real. Para poder establecer una demora de tiempo al paso de una entidad por un bloque, se establecen los bloques **PROCESS.**

Las actividades de este bloque pueden involucrar o no involucrar un recurso. El empleo de un recurso generará colas siempre y cuando este recurso esté ocupado despachando a una entidad que le haya solicitado un servicio previamente.

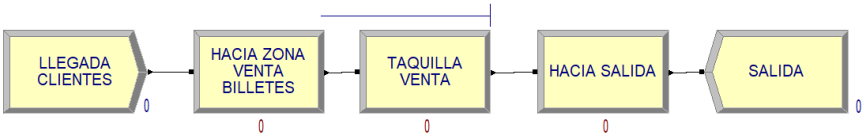
Existen diferentes acciones al emplear un recurso dado: captura del recurso (*Seize*) para dar un servicio, liberación del recurso (*Release*) para que pueda emplearse con otra entidad y generación de demora por la propia acción del recurso (*Delay*) al dar el servicio a una entidad. Etas acciones pueden darse juntas o separadas. Sin embargo, lo más común es encontrar la acción concatenada de captura, demora y liberación del recurso (*Seize, Delay, Release*). El resto de opciones se describirán en detalle más adelante en el transcurso de los problemas. Esto se seleccionará dentro de las propiedades del bloque, dentro de la casilla Action:



- Actividades sin recurso: (por ejemplo, gente andando), entran todas las entidades sin restricción → **NO se generan COLAS**
    - Delay
- Actividades con recursos: (por ejemplo, un cajero automático): las entidades no pueden entrar al bloque hasta que el recurso esté liberado (la entidad haya terminado) → **se generan COLAS**
    - Seize, Delay and Release. En este caso habrá que generar un **RECURSO**

PROBLEMA 4.1. En una estación marítima, los clientes van llegando a la zona de venta de billetes de barco según una distribución de probabilidad que sigue una exponencial de media 5 minutos. Desde la puerta de entrada a la estación a la taquilla de venta de billetes de transporte se pierde un tiempo caminando que varía en cada cliente, pudiendo ajustarse a una distribución triangular de mínimo, moda y máximo de 1, 2 y 3 minutos, respectivamente. Una vez llega la taquilla, es atendido por un operador de venta, únicamente si está disponible, con una tasa de servicio que sigue generalmente una distribución tipo Earlang de parámetro *k*=2 y media exponencial de 2 minutos.

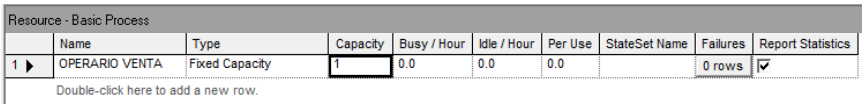
En este problema se verá el uso del bloque PROCESS con y sin asignación de un recurso.



El primer bloque PROCESS simulará una demora: el tiempo consumido por entidad en recorrer el espacio entre la entrada y la taquilla de venta. Este bloque, por tanto, no requiere un recurso, y no se generarán colas (las entidades pueden entrar en grupos, no es necesario ir de una en una al no generarse colas). Para ello, la opción a elegir en la casilla Action es "*Delay*".

El segundo bloque PROCESS simula la atención propia del operador de venta con el cliente. En este caso, existe una demora de tiempo traducida en el tiempo que tarda el operario en despachar al cliente: Por tanto, este bloque requiere de un recurso. La opción a seleccionar en la casilla Action es en este caso "*Seize*, *Delay, Release*", que significa "*Captura, Demora, Liberación*": "*captura*" de un recurso, "*demora*" de tiempo, y "*liberación*" de ese recurso una vez se le haya dado servicio.

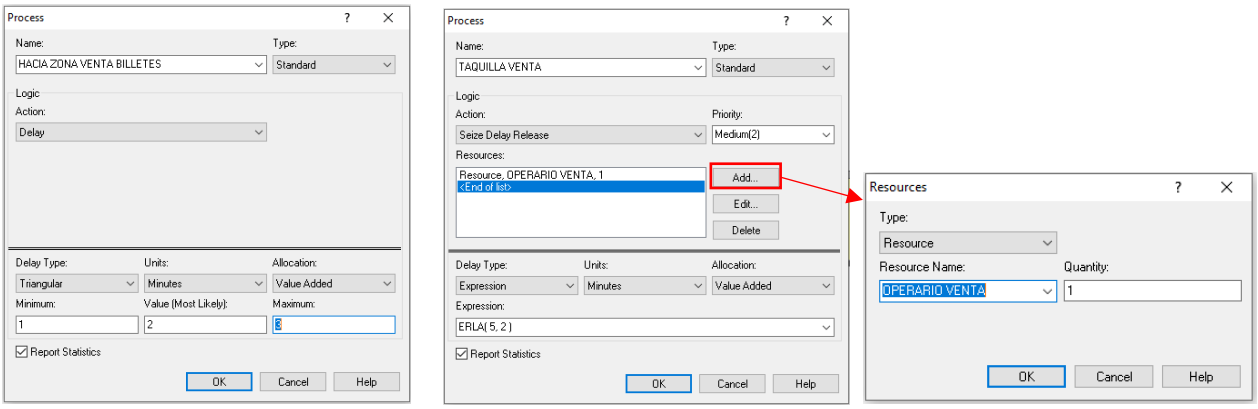
Para añadir un recurso se debe de pulsar el botón Add... . En la ventana de definición de recursos (*Resources*), la ventana Quantity debe tener el valor 1. Esto indica el número de recursos que son necesarios para despachar una entidad. Si se necesitaran dos operarios por cliente, entonces este número debería ser un 2. Es importante recalcar esta casilla no indica la capacidad del servidor (el operario de venta). El aumento de la capacidad del servidor (más operarios de venta) se define desde el bloque de propiedades del recurso que acabamos de generar, en el bloque "*Resource*" Resource del menú izquierdo. Al seleccionar este bloque aparece un menú inferior donde podremos editar o añadir más recursos. Variando el valor de la casilla Capacity podremos definir más o menos servidores (operarios de venta). Se puede comprobar cómo, al aumentar la capacidad, disminuyen las colas.


Resource - Basic Process

	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use	StateSet Name	Failures	Report Statistics
1	OPERARIO VENTA	Fixed Capacity	1	0.0	0.0	0.0		0 rows	☑

Double-click here to add a new row.

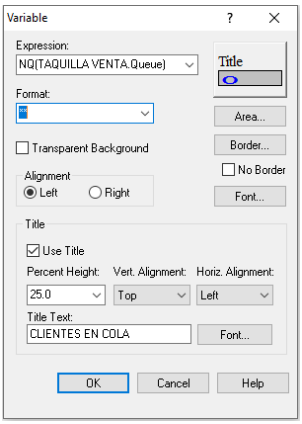
En ambos casos, la función de distribución que se ajusta a estas acciones se puede establecer en el módulo inferior del bloque PROCESS:



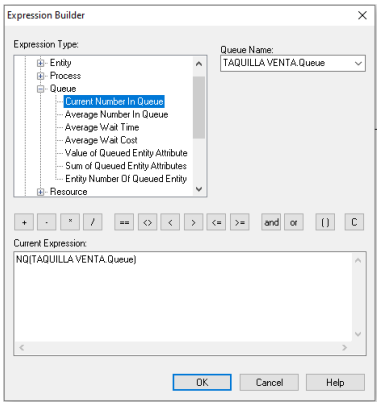
Al arranchar la simulación, se observa cómo en el bloque PROCESS sin recurso no se forman colas, todas las entidades pueden entrar y salir. Sin embargo, se comienzan a generar colas en el bloque PROCESS que tiene un recurso, puesto que tiene que atender a un cliente y simultáneamente le van llegando nuevos clientes, que tienen que esperar hasta que el recurso esté disponible.

De este bloque PROCESS con recurso se pueden extraer diversos datos y resultados que son de gran interés en cualquier simulación, como el **número de entidades en cola** (NQ), el **número medio de entidades en cola** (DAVG) o el **tiempo medio en cola por entidad** (TAVG). Estos resultados pueden ser obtenidos y observados al instante como una variable accediendo al visor de variables de la banda superior del programa y definiendo en la ventana Expression: la expresión propia de cada una de ellas:

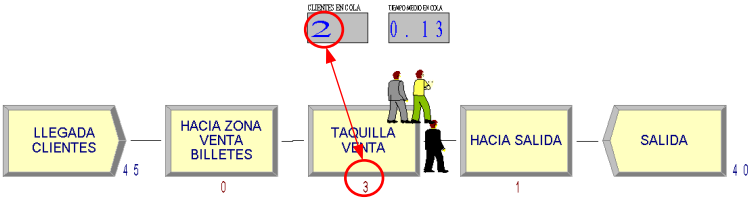
- ✓ Número de entidades en cola: NQ(*nombre_bloque*.Queue)
- ✓ Número medio de entidades en cola: DAVG(*nombre_bloque*.Queue.NumberInQueue)
- ✓ Tiempo medio en cola: TAVG(*nombre_bloque*.Queue.WaitingTime)



Donde *nombre_bloque* debe sustituirse por el nombre del bloque PROCESS donde proceda la cola, para este caso {TAQUILLA VENTA}. Esta expresión puede ser fácilmente construida de igual forma desde el *Expression Builder* de ARENA:



Sin embargo, se observa cómo el número de entidades que aparecen en el *display* inferior del bloque PROCESS no es el mismo que el que aparece en el visor de la variable del número de entidades en cola.



No se trata de un error. El número del visor de variables indica únicamente el número de entidades que están físicamente en la cola, mientras que el propio del bloque de procesos hace referencia a la suma de las entidades en cola más las entidades que están siendo atendidas por los recursos. Éste último valor se denomina en el programa como *Work In Process* (*WIP*), y podemos obtener su valor instantáneo mediante la expresión:

nombre_bloque.WIP

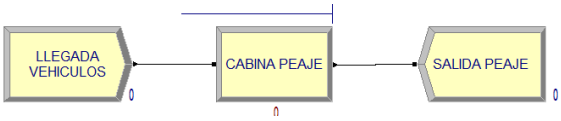
Para el caso concreto, únicamente se dispone de un recurso que puede atender a una entidad, con lo cual siempre tendrá un valor una unidad mayor que el número de clientes en cola. También es posible visualizar esta expresión desde el visor de variables.

PROBLEMA 4.2. En una autopista de peaje, los usuarios (vehículos) llegan a la estación de peaje siguiendo una distribución normal de media 2 minutos y desviación estándar de 1.5 minutos. La estación de peaje dispone de cabinas habilitadas para el pago en cabina de la tasa. Los cajeros tardan un tiempo que sigue una distribución normal de media 3 minutos y desviación estándar de 0.5 minutos en despachar el pago de cada usuario.

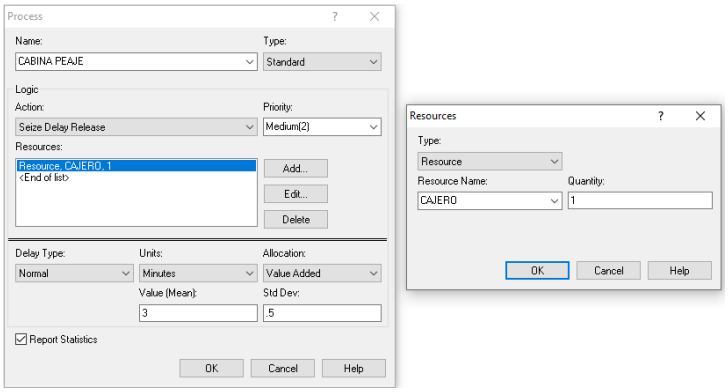
En este problema se intenta focalizar en la importancia de la capacidad de los servidores (número de recursos), de la incidencia de compartir recursos, los tipos de colas (una cola única para todos los servidores, o colas individuales para cada servidor) y las elecciones por condición de cola más corta.

Cola única. Varios servidores.

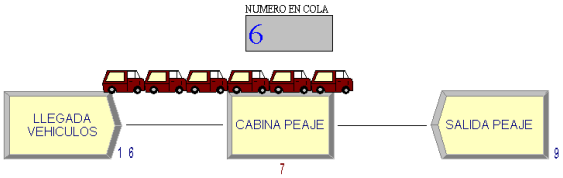
Imagínese que únicamente existe un carril que da acceso a las cabinas de pago, y que solo una de ellas está operativa.



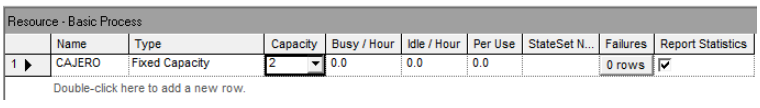
La cabina de pago estará representada por un bloque de proceso y empleará únicamente un recurso (*Seize, Delay Release*). Por tanto, la casilla Quantity: estará definida con un valor de 1



En este caso, se observa como al tener un único recurso destinado a todos los usuarios, se forman colas que se van prolongando hasta el infinito con el paso del tiempo, pues llegan más usuarios de los que el recurso (el servidor) es capaz de despachar.



Por tanto, la solución para evitar este colapso está en aumentar el número de recursos, es decir, abrir otra cabina de pago. Tal como se ha comentado antes, el número de recursos se gestiona desde el menú que genera el bloque de propiedades "*Resource*" Resource, donde cambiaremos la capacidad Capacity a 2 recursos:



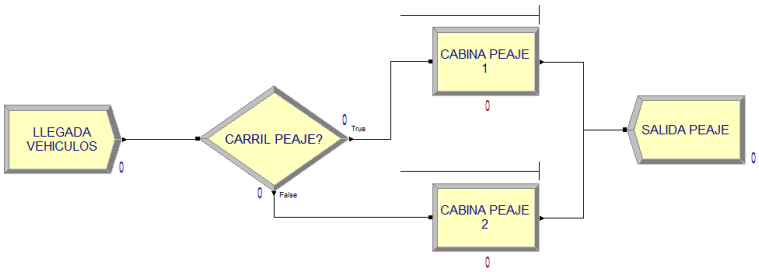
Al correr de nuevo la simulación las colas ya han desaparecido, pues ambos recursos son capaces de absorber el volumen de entidades que les van llegando.

Pero, ¿qué ocurre con la cola? Nótese que la cola sigue siendo única y sin embargo existen dos recursos operando. Ésta es la forma que tiene ARENA de establecer una **cola única para varios servidores** (recursos). En este caso, el primer usuario de la cola irá al primer recurso que quede disponible.

Esta situación es habitual, pero no es la disposición que generalmente siguen las estaciones de peaje. En éstas, cada cabina de peaje tendrá su carril y por tanto su cola independiente de las demás cabinas, que es el siguiente caso:

Colas independientes. Varios servidores (2).

Lo habitual en las estaciones de peaje es, una vez llega el usuario, se elija uno de los carriles que dan acceso a las cabinas de peajes. Si se dispusiera de 2 cabinas de peaje, se tendrían dos colas independientes entre ellas.

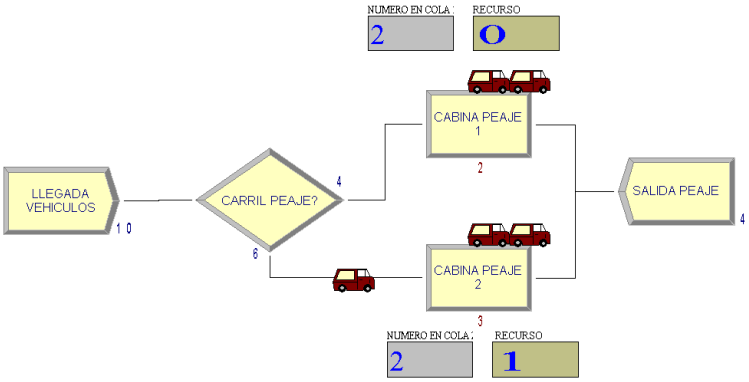


Para ello, se dispone de una bifurcación tipo DECIDE con una condición tipo *2-way by Chance*, con una probabilidad de escoger cola al 50%, seguido de dos bloques PROCESS. El recurso, inicialmente, será compartido entre ambas cabinas, por lo que el recurso generado tendrá que tener un valor de Capacity de 1 (esto es, solo se dispone de un recurso) y en el bloque PROCESS un valor de Quantity de 1 (esto es, solo se necesita de un recurso para dar servicio a una entidad en el bloque).

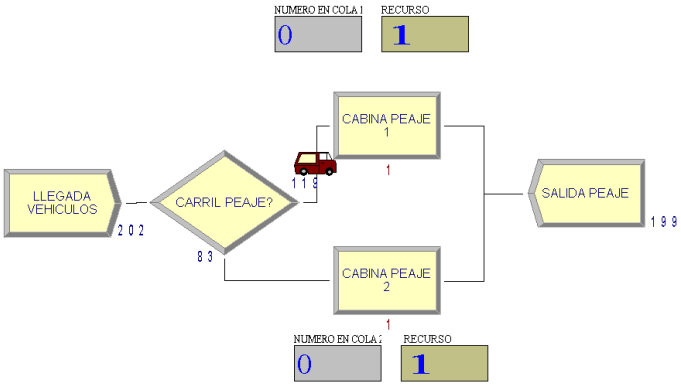
Lo que se observa con esta configuración es la formación de colas en ambas cabinas, pese a que anteriormente se ha demostrado que con dos servidores (recursos) era suficiente para no generar congestión. Como conclusión, aunque existan dos procesos para atender a las entidades, únicamente hay disponible un recurso que se va compartiendo según necesidades. Por tanto, el recurso no puede estar en ambos procesos simultáneamente. En definitiva, esta configuración tiene el mismo resultado que si únicamente existiera una cola y un solo servidor.

Podemos comprobar dónde está operando el recurso visualizándolo con el visor de variables, estableciendo una expresión (que llamaremos RECURSO) que indique el número de entidades que están siendo atendidas, restando al número total del proceso las entidades encoladas:

nombre_bloque.WIP - NQ(*nombre_bloque*.Queue)



Para solucionar este problema se procede como en el Problema 4.1, aumentando la capacidad del Recurso a 2, obteniéndose una simulación sin apenas colas y con los dos recursos constantemente ocupados:



Sin embargo, lo habitual en estos casos no es que el cliente escoja la cola por probabilidad o al azar, sino que se escoja dependiendo de qué cola es la más corta (a priori será la que menor demora conlleve). Este caso se verá a continuación:

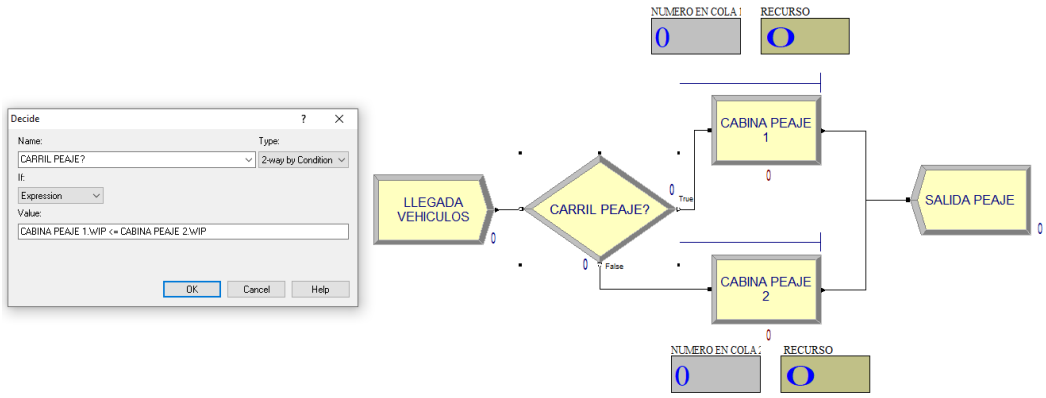
Colas independientes. Varios servidores (2). Cola más corta.

La reacción lógica de un usuario al llegar a una zona de servidores es escoger la cola más corta. En este caso, tendremos que establecer un tipo de condición en el DECIDE basado en *2-way by Condition*, y establecer la condición: si la cola en la cabina de peaje 1 es menor o igual que en la cabina de peaje 2, escoge el camino "*True*" y se dirige a la cola de la cabina de peaje 1.

NQ(CABINA PEAJE 1.Queue) <= NQ(CABINA PEAJE 2.Queue)

No obstante, pese a que a priori parece la expresión correcta, esta es una condición errónea. Puede darse la circunstancia que el número de entidades en cola en ambos procesos sea 0, y por tanto la entidad seleccionaría el camino 1. Sin embargo, ha de tenerse en cuenta el número de entidades que están siendo servidas por el recurso, y que en el caso anterior no se ha contabilizado. Por tanto, puede que el camino 1 sea el incorrecto al haber una entidad en el servidor 1 y automáticamente se ponga en cola, y sin embargo no existan entidades en el servidor 2 (para el cual directamente pasaría a darse servicio). Para tener en cuenta esta circunstancia, la condición debe darse con la expresión WIP que se ha visto en el problema anterior:

CABINA PEAJE 1.WIP <= CABINA PEAJE 2.WIP



Colas independientes. Varios servidores (3). Cola más corta.

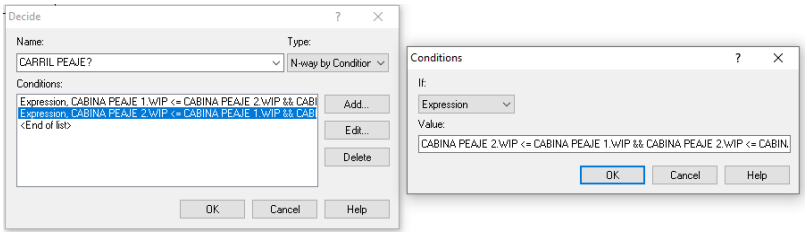
Se trata de añadir una cabina de peaje (bloque de procesos) y establecer la misma condición que el caso anterior, pero esta vez deberá ser una condición anidada al existir más de dos opciones posibles.

Se generan recursos con una capacidad Capacity de 3 (un recurso por proceso). Se establece en el bloque DECIDE un tipo de condición *N-way by Condition*. La condición deberá ser anidada y doble. Para este caso:

Condición 1: CABINA PEAJE 1.WIP <= CABINA PEAJE 2.WIP **&&** CABINA PEAJE 1.WIP <= CABINA PEAJE 3.WIP

Condición 2: CABINA PEAJE 2.WIP <= CABINA PEAJE 1.WIP **&&** CABINA PEAJE 2.WIP <= CABINA PEAJE 3.WIP

Se debe de cumplir una doble condición (anidada con el símbolo "*and*", **&&**) para que la entidad elija la primera ruta. Si la condición anidada número 1 no se cumple, se comprobará la condición anidada 2. Si no se cumple ninguna de las dos condiciones anidadas, la entidad tomará la tercera vía ("*Else*"):

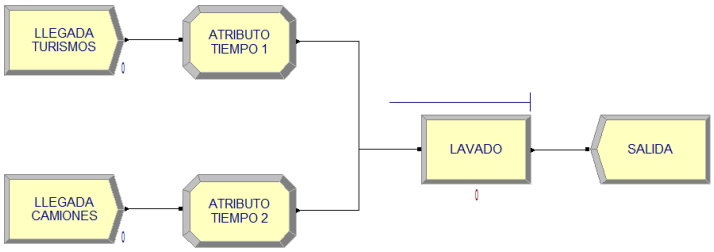


Otro tipo de sentencias que permiten indicar condiciones son las siguientes, y están disponibles como botones directos desde el "*Expression Builder*".

- ✓ *And*: **&&**
- ✓ *Or*: **||**
- ✓ *Igual*: **==**
- ✓ *Mayor o menor*: **>** o **<**

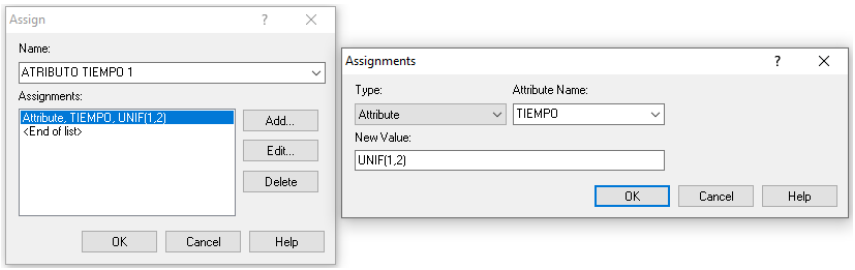
PROBLEMA 4.3: A una máquina de lavado de vehículos llegan tanto turismos como camiones. Ambos vehículos tienen unas llegadas que se pueden asimilar a una Exponencial de media 5 minutos para los turismos, y de media 10 minutos para los camiones. El tiempo de lavado es menor en un turismo, que sigue una distribución de probabilidad uniforme de mínimo y máximo 1 y 2 minutos respectivamente (UNIF(1,2)), que para un camión, que sigue una distribución de probabilidad uniforme de mínimo y máximo 5 y 10 minutos (UNIF(5,10)).

Con este problema se pone de manifiesto que no todas las entidades que atraviesan un proceso deben servirse con la misma tasa de servicio (tiempo de servicio). En ocasiones dependerá del tipo de entidad. En este caso, es evidente, por las dimensiones, que la entidad camión tardará más tiempo ser despachado que el turismo.

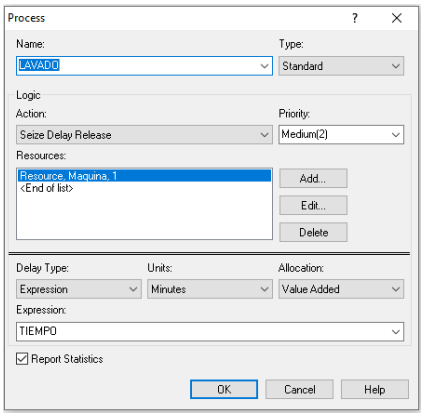


Para poder establecer este criterio, se emplearán las técnicas Atributo Genérico e Índice(Expresión) ya desarrolladas con anterioridad.

A. Con ATRIBUTO GENÉRICO: al usar esta técnica se generará un atributo específico para cada entidad, denominado TIEMPO, y que tendrá como valor la expresión propia de la distribución de probabilidad de tiempos de servicio (lavado) específico de cada entidad. De este modo, se evita que este tiempo sea asignado desde el bloque de procesos de forma general e igual para todas las entidades.



Para establecer la distribución de probabilidad que trae como atributo cada entidad en el bloque PROCESS, en lugar de definir una probabilidad, en la casilla Expression de este bloque indicaremos el nombre del atributo (TIEMPO). De este modo, cuando la entidad entre en el bloque, el bloque llamará a su atributo TIEMPO y establecerá el valor que tenga definido en él. Así, cada entidad tendrá su propia distribución de probabilidad en los tiempos de servicio.



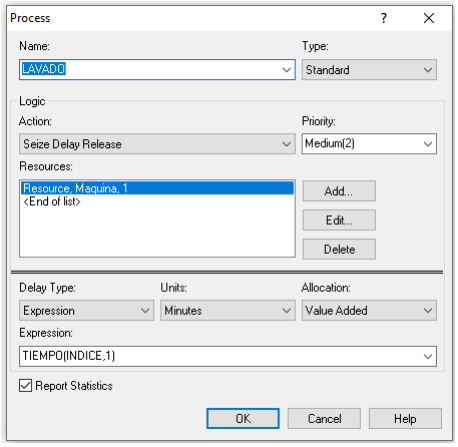
B. Con INDICE(EXPRESION): esta técnica empleaba una variable del sistema a modo de matriz (con *i* filas y *j* columnas) donde se introducen los valores deseados globales del sistema, y un índice a modo de atributo, con un valor determinado para cada entidad, que sirve para buscar dentro de la matriz formada por la variable. Sin embargo, las variables únicamente aceptan valores numéricos en su interior. No permiten introducir, como es el caso, expresiones y otros elementos. Para solventar este problema se recurre al bloque "*Expression*" Expression dentro del módulo Advanced process del menú izquierdo Advanced Process.

   En este elemento, denominado también TIEMPO, introduciremos, a modo de matriz 2×1, las expresiones propias de la distribución de probabilidad de tiempos de servicio de los vehículos (fila 1) y de los camiones (fila 2), de la siguiente forma:

$$TIEMPO = \begin{bmatrix} UNIF(1,2) \\ UNIF(5,10) \end{bmatrix}$$

   En segundo lugar, se asignarán los atributos INDICE para ambas entidades, desde el bloque ASSIGN, con valores de 1 para los turismos y 2 para los camiones. En el bloque PROCESS estableceremos en la casilla Expression:

TIEMPOS(INDICE,1).

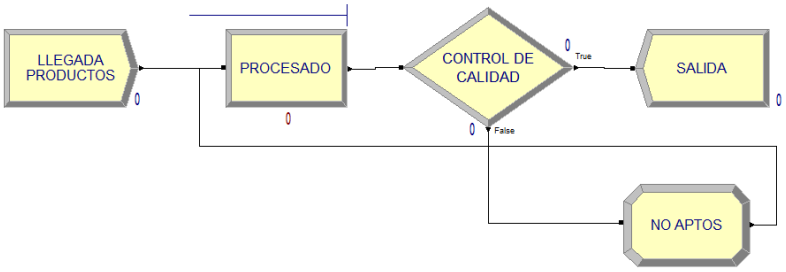


Se ha de resaltar que, generando un elemento matriz 1x2 la expresión debería ser *TIEMPOS(1,INDICE).*

PROBLEMA 4.4: **En una fábrica, diversos productos (bola verde) llegan a un subproceso de fabricación para ser procesados con una tasa de llegada constante cada 5 minutos. El tiempo de procesamiento es variable y sigue una distribución normal de media 4 minutos y desviación estándar de 0.5 minutos. Una vez son procesados, en la siguiente etapa deben superar un control de calidad que indicará si el producto es apto o no. Según estadísticas de la fábrica, el 40% de los productos no es apto (bola roja), para lo cual debe ser redirigido al subproceso de fabricación. En este subproceso tienen preferencia con respecto a los productos que aún no han sido procesados, quedando por delante en cola.**

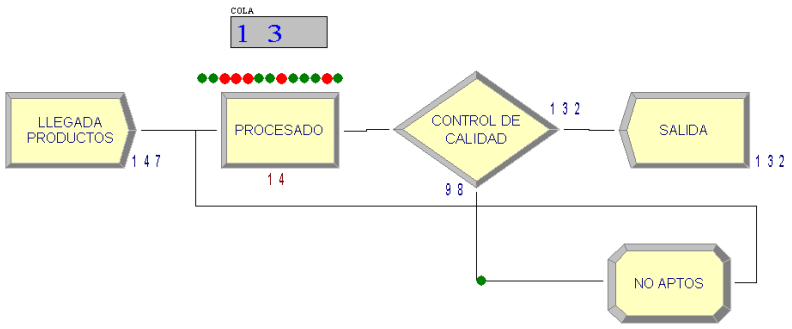
Generar, además, gráficos del número de entidades en cola y su tiempo medio.

En este ejemplo se definirá el procedimiento para establecer prioridades en las colas. El esquema básico del problema es el siguiente:

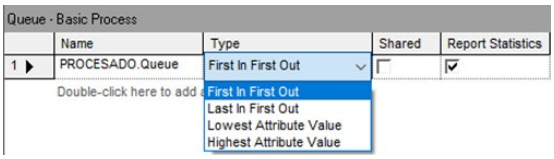


El bloque de proceso PROCESS será definido como un *Seize, Delay, Release* que consumirá el único recurso del que dispone el subproceso de fabricación. Además, en el bloque de ruteo DECIDE se establecerá la condición como un *2-way by Chance*.

Al poner en marcha la simulación se advierte como en la cola formada en el subproceso llegan las entidades que han sido marcadas como no aptas (bolas rojas). Sin embargo, estas bolas rojas no obtienen prioridad con respecto a las verdes (aun sin procesar).



La prioridad se define dentro del bloque de propiedades del menú izquierdo, en *Basic Process,* denominado "*Queue*" Queue. Al seleccionar este bloque aparecerá un menú inferior donde quedan definidas todas las colas generadas en la simulación (una única cola en este caso). En la opción Type es posible establecer la disciplina de la cola desde el desplegable.



- ✓ *First In First Out*: primera entidad que llega a la cola, primera entidad que sale de la cola (por defecto)
- ✓ *Last In First Out*: última entidad que llega a la cola, primera entidad que sale de la cola.

- ✓ *Lowest Attribute Value*: entidad que tenga un valor menor de un atributo determinado, primera entidad en salir.
- ✓ *Highest Attribute Value*: entidad que tenga un valor mayor de un atributo determinado, primera entidad en salir.

La opción es por tanto Highest Attribute Value, seleccionando como atributo PRIORIDAD.

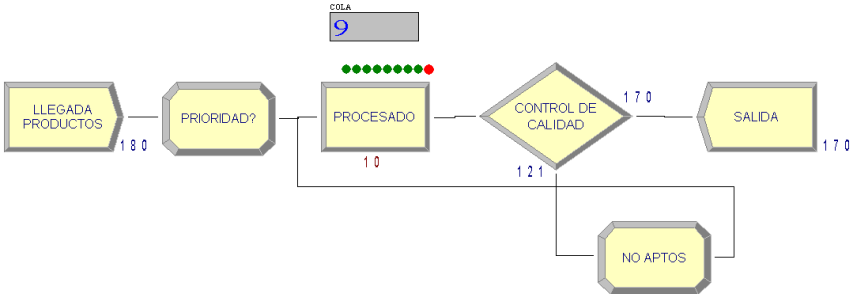
✓

Queue - Basic Process

	Name	Type	Attribute Name	Shared	Report Statistics
1 ▶	PROCESADO.Queue	Highest Attribute Value	PRIORIDAD	☐	☑
	Double-click here to add a new row.				

La prioridad se asigna para este caso mediante un atributo que se denomina PRIORIDAD. Este atributo es definido con el valor 0 para las entidades que aun estén sin procesar (bolas verdes), y cambiará a un valor 1 para las entidades evaluadas como no aptas (bolas rojas) Por tanto, un nuevo bloque ASSIGN asignará este atributo una vez se genere la entidad (posterior al bloque CREATE, PRIORIDAD == 0) y en el bloque existente ASSIGN posterior al rechazo de la entidad (posterior al DECIDE ruta "*False*", PRIORIDAD == 1), modificará este valor del atributo.

Toda entidad que pase por la cola del subproceso y tenga un atributo de valor mayor (bolas rojas con atributo PRIORIDAD ==1), tendrá prioridad sobre cualquier entidad que ya esté en la cola (bolas verdes con atributo PRIORIDAD == 0), colocándose por delante de éstas.

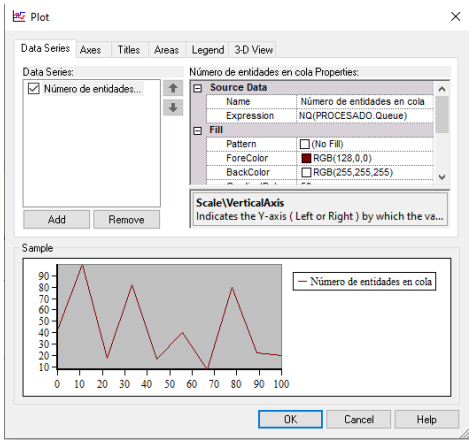


Definición de gráficas

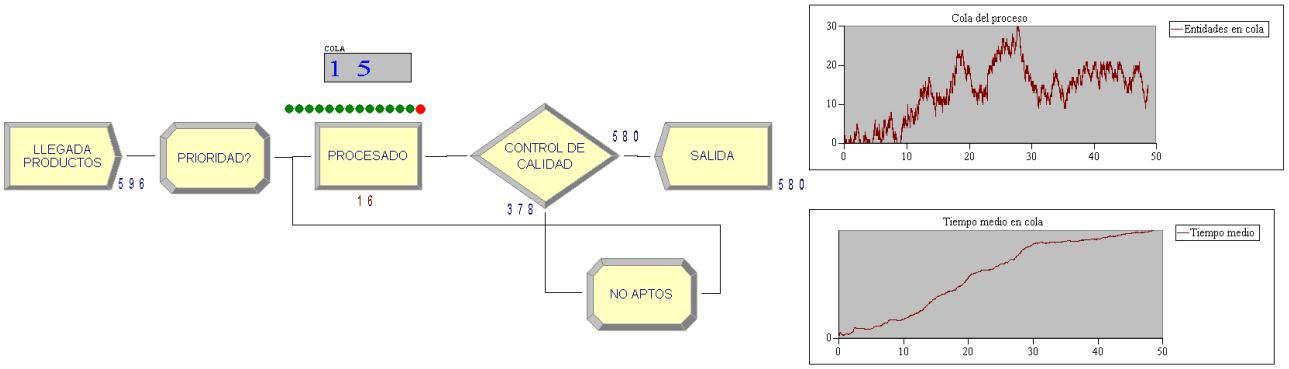
El objetivo de este ejemplo es también el familiarizarse con el uso de gráficas a un nivel básico. Se pretende establecer un gráfico que enfrente tiempo (de simulación) con respecto a otros factores, para este ejemplo variables clásicas como el número de entidades en cola y el tiempo medio en cola.

Las opciones de gráficas se encuentran en el icono de acceso directo "*Plot*" situado en la zona superior. Inmediatamente nos da acceso a la ventana de opciones, con varias pestañas. Se pasa a continuación a ver las opciones más importantes.

En la pestaña _Data Series_ se introducen los datos de entrada a la gráfica pulsando el botón Add y posteriormente, entre otros aspectos importantes, modifica el nombre de la serie (variable que se quiere conocer) y se introduce la expresión de la variable que se desee enfrentar al tiempo. Para las dos gráficas seleccionadas: NQ(PROCESADO.Queue) y TAVG(PROCESADO. Queue.WaitingTime).



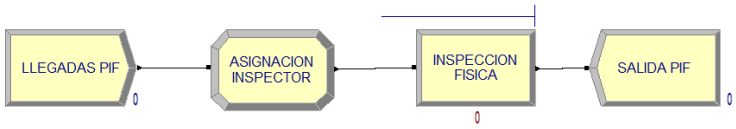
En la pestaña _Axes_ podremos cambiar la escala de los ejes, los intervalos en los que aparezcan las marcas de estos ejes o activar el autoscroll de la gráfica. En la pestaña _Title_ se define el nombre de la gráfica (*Heater*). Estas son las opciones más importantes y las principales para poder establecer gráficos.



PROBLEMA 4.5. A un Puesto de Inspección Fronterizo (PIF) de un puerto llegan contenedores para su inspección física siguiendo una distribución de probabilidad exponencial de media 2 minutos de tiempo. Para atenderlos, el PIF dispone de 4 inspectores. Cada inspector tiene su propio tiempo de servicio. Este tiempo sigue una normal de medias 3, 3.5, 4 y 4.5 minutos respectivamente para los inspectores 1, 2, 3 y 4 y una desviación estándar de 0.2 minutos para todos. Además, al llegar el contenedor, se crea una alerta automática que decide el inspector que debe de atender la mercancía. Según las estadísticas aportadas por el gestor del PIF, las probabilidades de que el contenedor sea atendido por estos inspectores es del 35%, 30%, 25% y 20%, respectivamente para los inspectores 1, 2, 3 y 4.

En este problema es necesario recurrir a un elemento denominado SET, en este caso de recursos. Estos elementos son capaces de agrupar otros elementos.

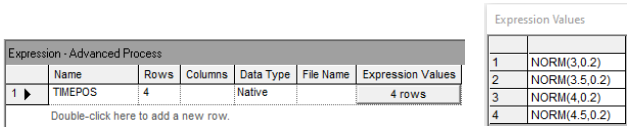
El problema define la existencia de 4 recursos para poder dar un servicio (inspección) a la entidad que llega (contenedor). Cada inspector o recurso tiene su propio tiempo de inspección. Además, al llegar el contenedor, hay una selección basada en probabilidad de ser atendido por uno u otro recurso. Por tanto, el problema podría solucionarse con un bloque DECIDE y cuatro bloques PROCESS, cada uno con su recurso propio (inspectores). Sin embargo, existen soluciones más eficaces y menos costosas empleando, en un solo bloque PROCESS, un SET de recursos, que aglutine a los 4 recursos de una vez.



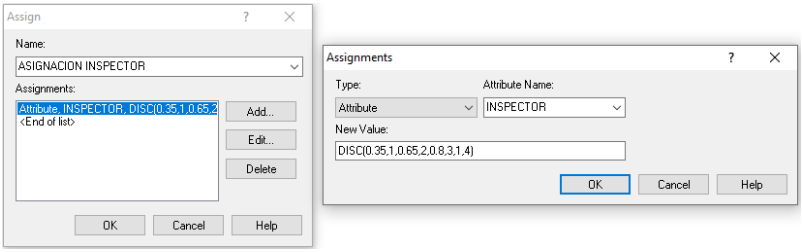
Como ya se ha visto en el PROBLEMA XXXX, para poder establecer un tiempo de servicio independiente para un recurso, es posible recurrir a las técnicas TAG o TIE. Para este caso, empleando TIE se genera una variable que pueda contender expresiones. Por tanto, dentro de las opciones del menú izquierdo, dentro de *Advanced Process*, seleccionamos el bloque de propiedades "*Expression*" Expression y generamos la variable de expresiones como una matriz 4×1 que contenga las distribuciones de probabilidad de cada recurso (inspector) y que se denominará TIEMPOS:

$$TIEMPOS_{4\times1} = \begin{bmatrix} Norm(3.0, 0.2) \\ Norm(3.5, 0.2) \\ Norm(4.0, 0.2) \\ Norm(4.5, 0.2) \end{bmatrix}$$

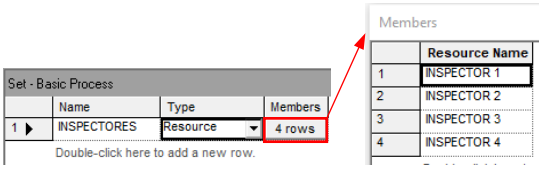
De este modo:



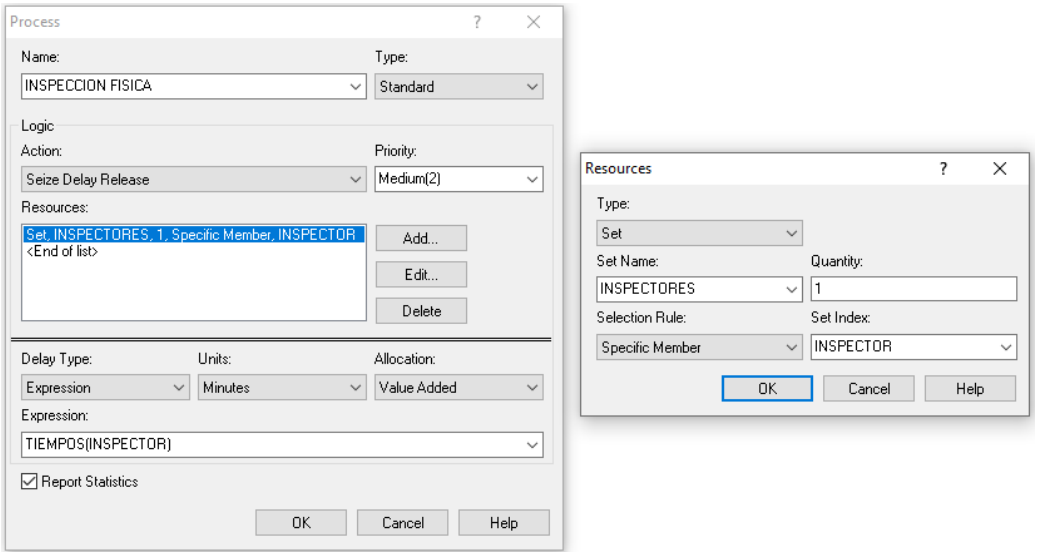
Para poder establecer qué recurso dará servicio a una entidad según probabilidad, se asigna un atributo (INSPECTOR) a cada entidad con valor 1, 2, 3 o 4, según una Discreta que sigue la siguiente expresión:



Las opciones para la generación de un SET se encuentran en el menú izquierdo, *Basic Process*, bloque de propiedades "*Set*" . El menú de la barra inferior que se genera permite crear un SET, indicando el tipo de SET, en este caso de Recursos, y por último el botón de número de miembros permite introducir tantos miembros como sean necesarios:



Una vez generado el SET, se procede a indicar en el bloque PROCESS el recurso a emplear, que para el caso será un SET. Esto se indica en las opciones del recurso, en la casilla Type donde, en lugar de seleccionar "*Resource*", se seleccionará "*Set*", seleccionando en la casilla Set Name el SET ya generado.



La casilla Selection Rule indica la regla de selección de recurso que seguirá la entidad. En este menú desplegable se observan diferentes opciones:

- ✓ *Cyclical* (cíclico): empezará por el 1, y después en orden lógico de selección: 2, 3 y 4, para comenzar nuevamente en el 1.
- ✓ *Random* (aleatorio): elección aleatoria del recurso del set.
- ✓ *Preferred Order* (orden preferente): según un orden predefinido.
- ✓ *Specific Number* (número específico): número del recurso según una etiqueta previa (atributo, variable).
- ✓ *Largest Remaining Capacity* (mayor capacidad restante): recurso que tenga mayor capacidad.
- ✓ *Smaller Number Busy* (menor número ocupado): recurso que esté menos ocupado

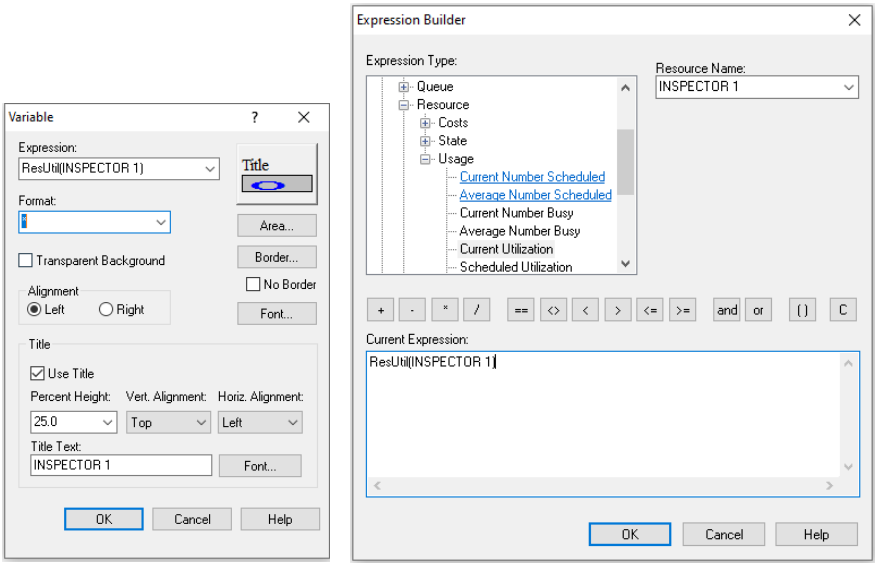
El problema exige una selección basada en probabilidad, por lo que cada entidad lleva el número del recurso que le atenderá a modo de atributo (INSPECTOR), como se ha visto anteriormente. De este modo, la opción a escoger es "*Specific Number*", indicando como índice del Set, Set Index, el atributo INSPECTOR. De esta

forma, cada entidad indicará que el recurso que debe servirle es el que indica el valor de su atributo INSPECTOR.

Lo mismo ocurre para el tiempo de servicio, que es diferente según el recurso seleccionado. Para establecer este criterio se emplea, como se ha definido anteriormente, la técnica TIE empleando tanto la variable expresión TIEMPOS como el atributo INSPECTOR. Se define la expresión TIEMPOS(INSPECTOR) en la casilla Expression.

Si se desea saber qué recurso está siendo usado, es posible recurrir al visor de variables y definir la expresión que indique si un recurso particular está siendo usado o no (búsqueda a través del *Expression Builder*):

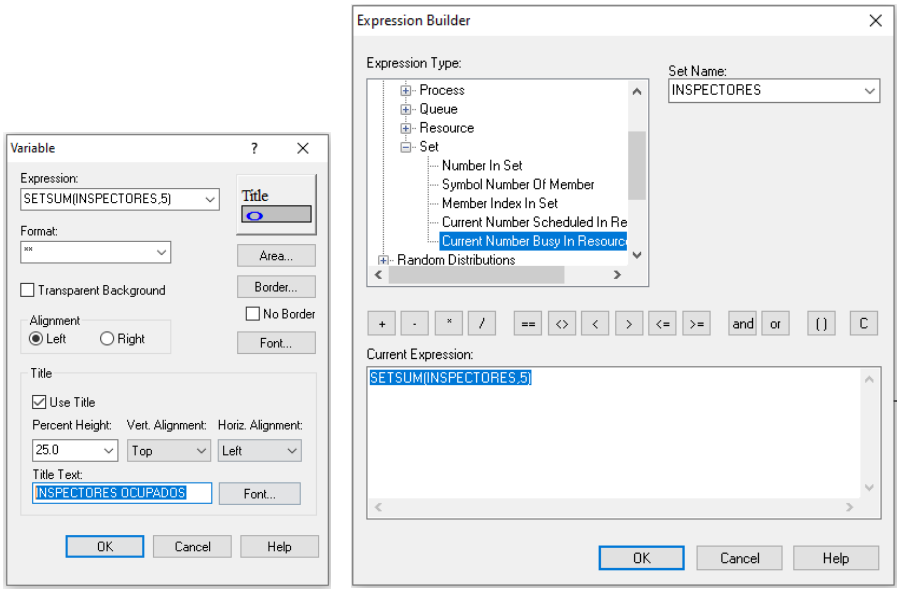
ResUtil(*nombre_recurso*)



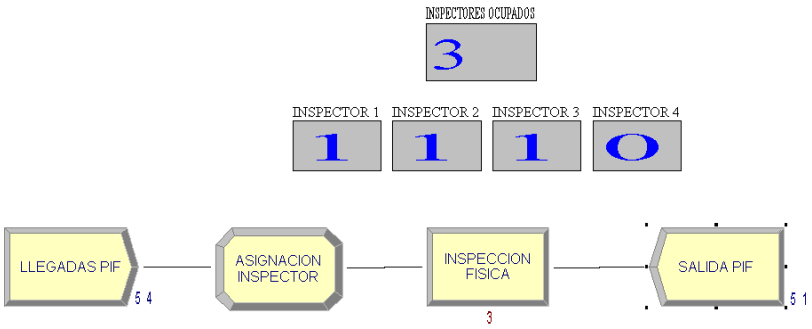
Mediante esta expresión, el visor muestra un valor de 0 si el recurso no está ocupado, y un valor de 1 si lo está.

Del mismo modo, el número total de recursos que se están empleando simultáneamente en un SET durante la simulación puede obtenerse mediante la expresión:

SetSum(*nombre_set*,5)



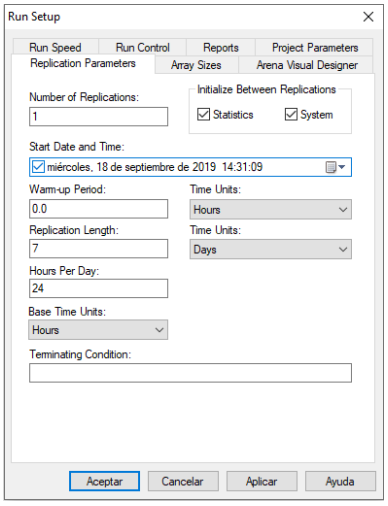
En la siguiente figura se muestra cómo, en un instante determinado de la simulación, se están empleando simultáneamente 3 de los 4 inspectores (recursos), y que estos inspectores son los correspondientes con los números 1, 2 y 3:



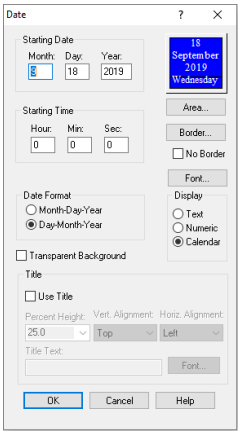
PROBLEMA 4.6. A una terminal de contenedores llegan buques portacontenedores siguiendo una distribución exponencial de media 3 horas. Esta terminal dispone de únicamente dos atraques, independientemente del tamaño del buque. Estos buques van cargados de contenedores, que descargan por completo en la terminal una vez atracados. Un estudio estadístico de la terminal determina que el número de contenedores que descargan estos buques sigue una normal con media 100 contenedores y desviación típica de 15 contenedores, mientras que el tiempo que emplea una grúa de muelle en descargar un contenedor se estima sigue una distribución uniforme de mínimo 2 y máximo 3 minutos. Se quiere comprobar el número de contenedores que habitualmente son descargados en la terminal durante una semana.

Este problema se debe abordar con la creación de elementos denominados ACUMULADORES. Estos elementos se comportan de la misma forma que los contadores, añadiendo además de valores, expresiones. Estos acumuladores son variables propias del sistema, ya que van sumando cantidades independientemente del tipo de entidad que vaya entrando al sistema.

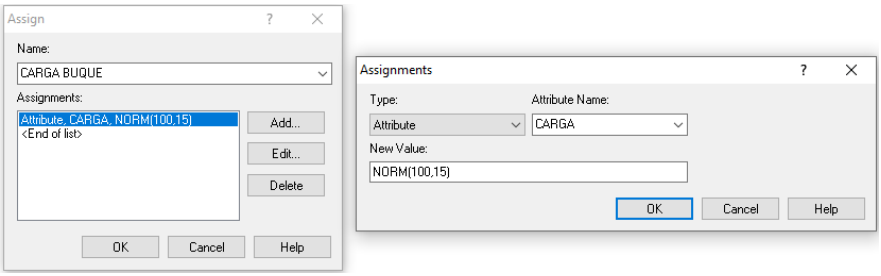
Primero se ha de establecer el tiempo de simulación en las opciones de la pestaña superior Run/Setup. En la pestaña _Replication Parameters_ se establece el tiempo de simulación en la ventana Replication Length:, con un valor de 7 días.



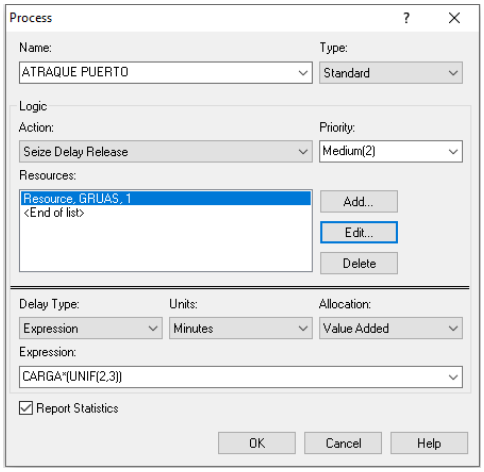
Para una mejor visulazación del horario, se dispone de un calendario que se activa desde el incono de la barra superior "*Date*" . En las opciones de la ventana emergente es posible la modificación de diversos parámetros del calendario.

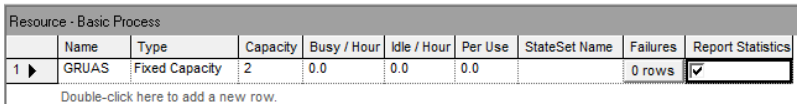


El prolema se aborda asignando en el inicio un atributo CARGA a cada entidad (buque) que indique el número de contenedores que tiene que descargar en muelle. Este atributo tendrá el valor de la distrubución de probabilidad definida para el número de contendores por buque.



Una vez llegan los buques, esperan fondeados hasta que queda libre uno de los dos atraques (recursos). En el bloque PROCESS se genera el recurso GRUAS. Este recurso estará presente en cada uno de los atraques, por lo que deben existir al menos 2, que se definen como la capacidad de los recursos en la casilla Capacity, del menú que se genera desde el bloque de bloque de propiedades "*Resource*" Resource del menú izquierdo *Basic Process*, siendo en este caso un valor de 2 atraques.



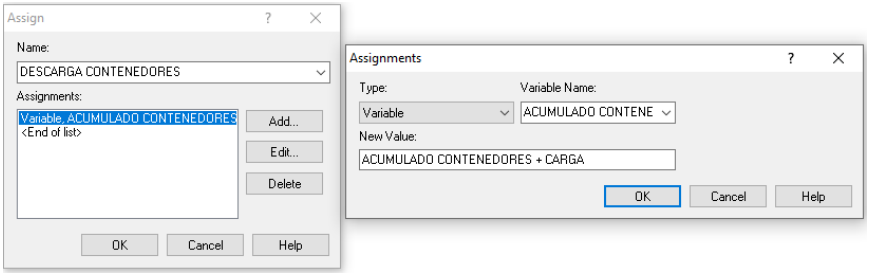


Una vez definidos los dos atraques, se procede a implementa el tiempo que cada recurso dedica a despachar (descargar) la entidad (buque). Para este caso, estableceremos una expresión que considera el tiempo que se pierde en la descarga de cada contenedor (UNIF(2,3)) y el número total de contenedores que carga cada buque, siendo esta última una expresión contenida dentro del atributo CARGA. Por tanto, la expresión final será la resultante de multiplicar el tiempo de descarga por contenedor por el número de contenedores:

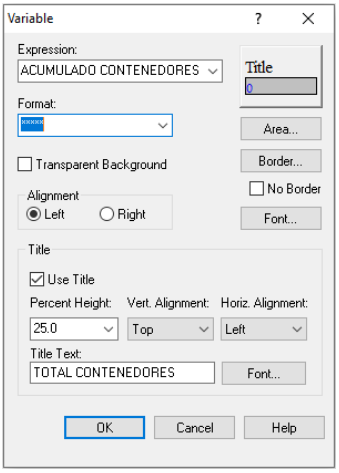
"CARGA×(UNIF(2,3)"

Para contabilizar el número de contenedores acumulado que se está descargando en la terminal se establece la variable ACUMULADO CONTENEDORES, definida desde un bloque ASSIGNE posterior al bloque PROCESS. La variable debe acumular el valor que traiga consigo el atributo CARGA en cada entidad. Para ello, se establece la expresión del valor de la variable, en la casilla New Value, como:

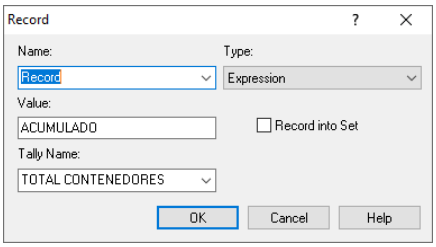
"ACUMULADO CONTENEDORES + CARGA"



Es posible visualizar la variable durante la simulación mediante el visor de variables:



Es posible generar estadísticas de conteos, expresiones, tiempo entre intervalos y otro tipo de datos de interés con el bloque denominado RECORD. Para establecer una estadística de la variable acumuladora, se selecciona "*Expression*" en la casilla Type, estableciendo como valor el nombre de la variable a reportar.



Una vez finalice la simulación, se comprueba dentro del informe generado que la variable acumuladora está presente:

User Specified

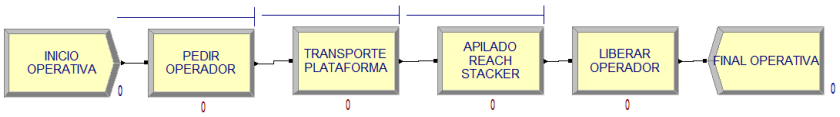
Tally

Expression	Average	Half Width	Minimum Value	Maximum Value
TOTAL CONTENEDORES	2600.41	(Insufficient)	94.5957	5119.85

Es importante mencionar que este bloque permite registrar datos y estadísticas hasta la posición en la que se encuentra dentro de la cadena de simulación.

PROBLEMA 4.7: **En una terminal de contenedores, la operativa de descarga de un buque se inicia descargando directamente los contenedores sobre una plataforma (propulsada por una cabeza tractora) en el cantil del muelle (uniforme de mínima 2 y máxima 3 minutos). Este contenedor tiene que ser transportado hasta el patio de almacenamiento por la plataforma (distribución triangular, de valores mínimo 0.5, más probable 1 y máximo 1.5 minutos). Una vez en el patio, una grúa *reach stacker* se encarga de descargar el contenedor de la plataforma y apilarlo en una de las pilas (el tiempo que se pierde en la acción sigue una distribución normal de media 1.5 y desviación estándar de 0.5 minutos). Únicamente se dispone de un operador de grúa en la terminal, que se encarga de operar ambas grúas, pero nunca simultáneamente. El operador no podrá transportar un nuevo contenedor hasta que termine la operativa completa.**

En problemas anteriores se ha visto cómo un recurso se capturaba, existía un retraso y por último se liberaba dentro del mismo proceso o actividad (*Seize, Delay, Realease*). Este problema plantea el uso de un solo recurso para dos procesos, en el que no se podrá liberar hasta que se completen ambos procesos o actividades. Es decir, el recurso del primer proceso quedará liberado cuando finalice el segundo proceso.

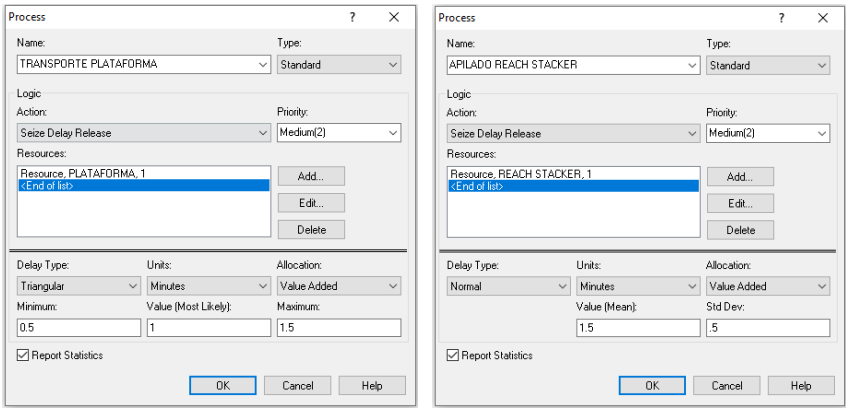


Como recursos, este problema plante el uso de tres: la plataforma de transporte horizontal, la grúa *reach stacke*r de apilamiento y el operario que las manipula.

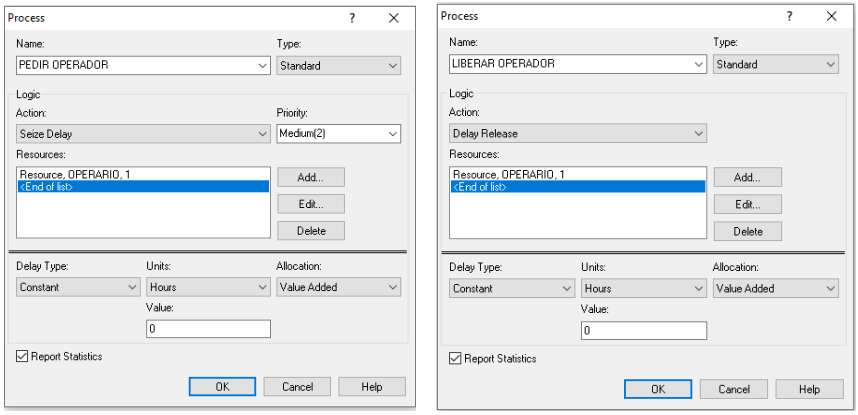
Resource - Basic Process

	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use	StateSet Name	Failures	Report Statistics
1	OPERARIO	Fixed Capacity	1	0.0	0.0	0.0		0 rows	☑
2	PLATAFORMA	Fixed Capacity	1	0.0	0.0	0.0		0 rows	☑
3	REACH STACKER	Fixed Capacity	1	0.0	0.0	0.0		0 rows	☑

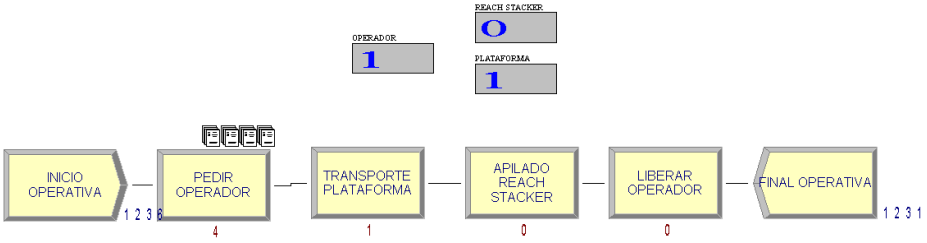
Primero, se definen los procesos que requieren un recurso que es liberado al finalizar el propio proceso. Para ello, se generan dos bloques PROCESS, en los que se define como recurso tanto la plataforma que transporta el contenedor, como la grúa *reach stacker* que apilar el contenedor en el patio, respectivamente:



A continuación, anterior y posterior a estos dos procesos, se definirán sendos bloques de procesos, uno para capturar al recurso OPERARIO, y otro para liberarlo. En este caso, la opción de la casilla Action no es "*Seize, Delay, Realse*", pues ello involucra no solo la captura, sino también la liberación del recurso en la misma acción. Para ello, en el bloque PROCESS previo a los dos procesos anteriores, se establecerá una acción "*Seize, Delay*" que únicamente tendrá la labor de capturar el recurso OPERARIO, sin liberarlo. En el bloque posterior, para poder liberar ese recurso, se selecciona la acción "*Delay, Release*". En ambos procesos, la captura y la liberación del recurso, no debieran existir demoras y, sin embargo, la propia acción las incluye. Para definir un "*Seize*" y un "*Release*" puros, sin demoras, se selecciona como tipo de demora, Delay Type, la opción "*Constant Value*", con un valor de 0.



De este modo, se captura el recurso OPERARIO y no es liberado hasta que completa sus tareas de transporte y apilado con cada uno de los recursos asignados para ello (maquinaria). Es posible observar cuándo el recurso está ocupado mediante el visor de variables (uno para cada recurso), indicando el número instantáneo de recursos ocupados mediante la expresión NR(nombre_recurso). Si se proyecta el valor 1, el recurso estará ocupado, mientras que un valor 0 indica que está disponible.



Como parece lógico, el operador se mantiene la mayor parte del tiempo ocupado (valor 1), mientras que una de las dos máquinas (la que no esté manipulada por el operario) estará libre (valor 0).