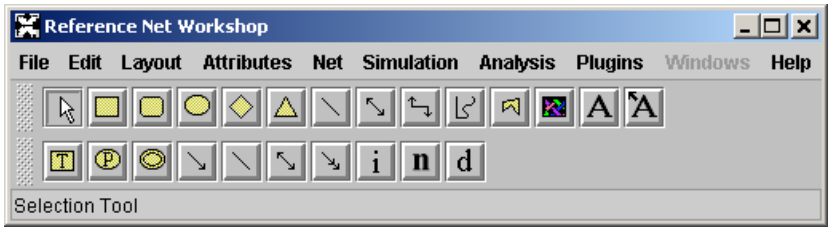
PRACTICA 1: Redes Lugar/Transición

Indicaciones para el manejo de renewz

La herramienta que se va a utilizar para simular y analizar redes de Petri es renewz, que se basa en el programa de simulación de redes renew, al que se le ha añadido un pequeño interfaz para poder realizar análisis de propiedades sencillas.
Al lanzar el programa aparecerá una ventana de comandos (minimizadla) y un entorno como el de la figura.



Para crear una red nueva, **File-> New NetDrawing. CUIDADO, NO File ->New Drawing**, porque aunque el dibujo sea igual, al no estar definido como red no se puede analizar, simular, etc.

Nosotros utilizaremos cuatro de los botones de la pantalla principal (dejando a un lado el menú). De izquierda a derecha:

- El primer botón (el que tiene una T) sirve para poner transiciones.
- El segundo (el que tiene una P) para poner lugares.
- El tercero (el que tiene una **i**) es para poner inscripciones, éste lo utilizaremos para poner el marcado inicial.
- El cuarto (el que tiene una **n**) es el botón de nombre. En las redes autónomas de esta primera práctica sólo se utilizará para dar nombre a lugares u transiciones. Más adelante, a través de las etiquetas que pongamos en el nombre les pasaremos información a nuestras herramientas de análisis sobre la temporización, prioridades, etc.

Hay un modo de editar las redes rápidamente: a partir de una transición o un lugar, si lo seleccionas y arrastras desde el centro del mismo, se añade automáticamente un arco y un lugar o transición (lo que toque). Asimismo, para poner un marcado basta con darle dos veces al botón derecho del ratón sobre el lugar en el que queramos poner la inscripción, y sustituir los [] que aparecen por un número.

Para mejorar la legibilidad del modelo, y poder interpretar los resultados de análisis es necesario asociar etiquetas a lugares y/o transiciones. Para ello, pulsar el botón de nombre, pinchar en el lugar o transición, y escribir "**name=**" y el identificador. Por defecto, lo escribe en negrita. Es útil porque permite detectar fácilmente errores como escribir el nombre usando el botón de inscripción en lugar del de nombre.

Una vez editada la red, se puede simular. Para ello en **Simulation->Formalisms** hay que señalar que es una P/T net. Se puede configurar para ir disparando las transiciones de una en una (**Sequential mode**), o para disparar en un paso todo lo que esté sensibilizado. También se puede ejecutar una secuencia indicando nosotros qué se dispara cada vez (**Simulation Step**).

Además de simular, con esta herramienta podemos analizar. En el menú **Analisis -> Cualitative properties** hay varias herramientas para ayudarnos a estudiar el comportamiento. Para esta práctica podemos usar las siguientes, basadas en el grafo de alcanzabilidad:

- Calcular el grafo de alcanzabilidad (la representación es en forma de lista, indicando para cada marcado las transiciones sensibilizadas y a qué marcado se llega si se disparan), y reducirlo si hay transiciones con distinta prioridad (ocultar los disparos de transiciones prioritarias)
- Calcular el número máximo de marcas en cada lugar
- Calcular el máximo número de sensibilizaciones simulatáneas de cada transición (servidores activos)
- Estudiar la vivacidad de cada transición. 0-viva indica que nunca se dispara, 1-viva que se puede disparar un número finito de veces, 2-viva que hay secuencias que permiten dispararla infinitas veces, pero también otras que la disparan un número finito de veces, 3-viva que siempre se puede volver a disparar.
- Obtener la clasificación de los marcados del grafo de alcanzabilidad en componentes fuertemente conexas.
- Obtener una secuencia que lleva desde un marcado inicial hasta uno final (útil por ejemplo para ayudar a comprender qué hace que se bloquee una red).
- Encontrar los marcados que cumplen una condición, y razonar sobre sus predecesores, sucesores, etc. Es especialmente útil para saber qué número tiene asignado un cierto marcado. Por ejemplo para poder obtener una secuencia de bloqueo es necesario conocer el número del marcado inicial. Para escribir condiciones la notación es del tipo: m[p1]==0 && (m[p2]>2 || m[p3]<=1)

- Pinchando en **Analysis→Cualitative properties** podemos calcular los P- y T-semiflujos del sistema.

La herramienta de la que disponemos no es capaz de simular con arcos inhibidores, pero sí que puede analizar la red. Para dibujar arcos inhibidores: en **Simulation->Formalisms** hay que señalar **Java Net Compiler** y debajo **Show sequential-only arcs**

Modelo 1

El sistema está compuesto por una máquina y un vehículo guiado automáticamente (AGV). La máquina sólo puede fabricar un producto cada vez, y tiene espacio delante para poder dejar materia prima para otro producto. Los productos fabricados se acumulan a la salida. El AGV suministra materia prima a la máquina. La coge de un almacén (que se supone de capacidad infinita) y espera hasta que hay hueco delante de la máquina para depositarla. Si en ese momento detecta productos en la salida, los retira dejándolos en un almacén (que se supone también de tamaño ilimitado). Si no hay ningún producto acabado vuelve a por materia prima. El objetivo principal del AGV es suministrar materia prima, sólo se lleva los productos que encuentra "para aprovechar el viaje".

a) Modelar el sistema en la situación inicial (almacenes vacíos).
b) Obtener el grafo de alcanzabilidad. ¿Se bloquea el sistema? ¿Son vivas todas las transiciones? ¿Es limitado? Para interpretar los resultados te puede resultar útil representar el grafo de alcanzabilidad en la forma "tradicional".
c) Calcula los P- y T-semiflujos del sistema y estudia su sentido físico.
d) Modificar el comportamiento del AGV, suponiendo ahora que se da más prioridad a la retirada de productos, y que después de llevar materia prima a la máquina el AGV retira todos los productos acabados que haya antes de volver a por más materia prima.¿Es vivo este modelo? ¿Es limitado?
a) Suponed ahora que las dos tareas son igualmente prioritarias y que cada vez que acaba una tarea puede decidir si retira productos o suministra materia prima. ¿Es vivo este modelo? ¿Y limitado?

Modelo 2

Un sistema toma materia prima de dos tipos, les da la forma deseada y ensambla las dos partes. La parte 1 se obtiene a partir de un bloque rectangular, al que se taladra un agujero en la parte superior. La parte 2 es un cilindro que se procesa para que encaje en el agujero de la parte 1. Para el transporte de la materia prima, las partes, y los productos finales se utilizan pallets.

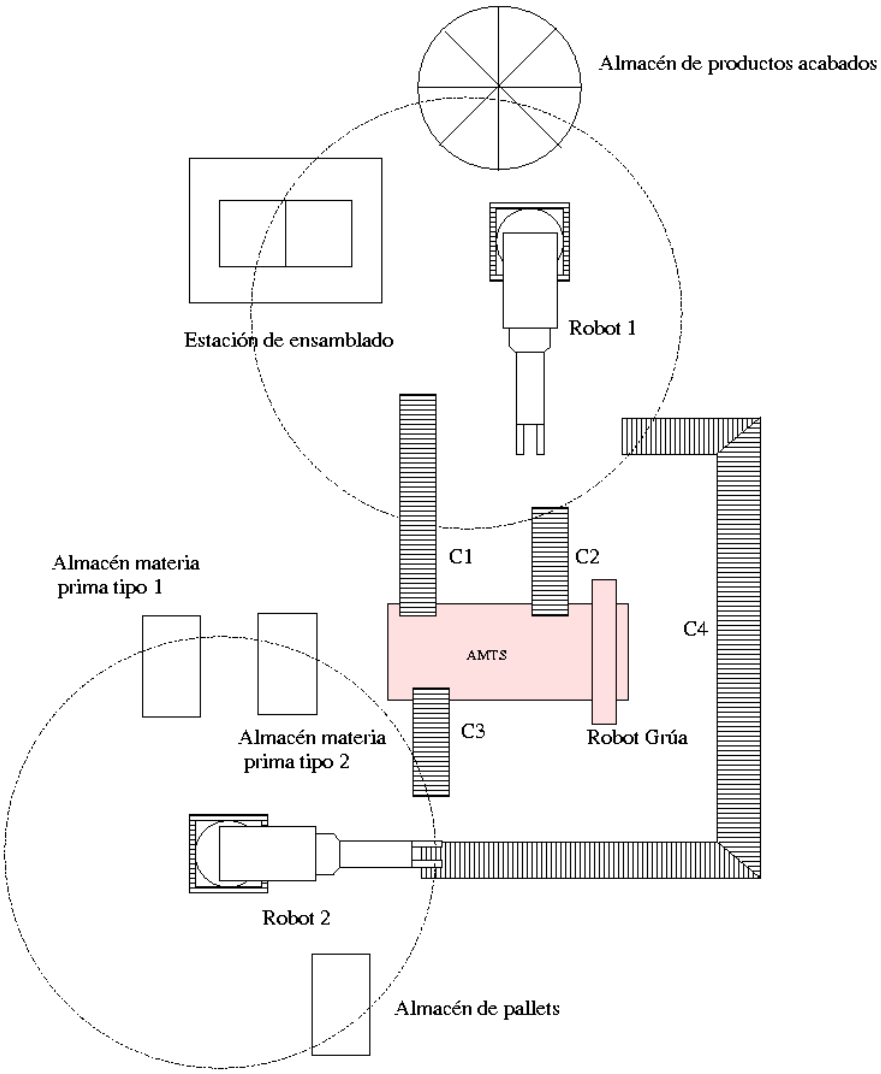
Distribución de la planta

Los principales componentes del sistema, como se ilustra en la figura, son:

- Una estación de ensamblado
- Un almacén de pallets
- Un almacén de materia prima para la parte 1
- Un almacén de materia prima para la parte 2
- Un sistema de transferencia de material automatizado, que incluye cuatro cintas transportadoras con sensores de presencia al final (Ci), un robot tipo grúa y dos robots, R1 y R2.

- El robot grúa transporta pallets con materia prima desde la cinta transportadora 3 a las cintas 1 y 2. Tiene capacidad para almacenar 3 pallets.
- El robot 1 transfiere la materia prima de las cintas 1 y 2 a la estación de procesado y ensamblado, y retira los productos acabados. También lleva los pallets vacíos de las cintas 1 y 2 a la cinta 4.

- El robot 2 mueve los pallets y la materia prima desde los almacenes correspondientes hasta la cinta transportadora 3, y retira los pallets vacíos de la cinta 4.
- Las cintas transportadoras 1 y 2 llevan pallets con materia prima del robot grúa al robot 1.
- La cinta 3 lleva pallets con materia prima desde la zona de almacenamiento de materia prima hasta el robot grúa.
- La cinta 4 lleva los pallets vacíos desde las cintas 1 y 2 hasta el almacén de pallets.



Procesado y ensamblado

El producto final se obtiene ensamblando una parte de tipo 1 y una parte de tipo 2.
El procedimiento que se sigue para fabricar una parte de tipo 1 es el siguiente: el robot R2 coge un pallet del almacén de pallets y lo pone en la cinta transportadora C3. Después coge materia prima de tipo 1 y la deposita en el pallet. C3 transporta el pallet hasta el robot grúa, donde se puede quedar almacenado hasta que se procese en la taladradora. Para esto, se carga el pallet en C1 que lo lleva hasta la célula de ensamblado, de donde es descargada la materia prima por R1.
Las partes de tipo 2 siguen un proceso análogo. El robot R2 coge un pallet del almacén y lo pone en la cinta transportadora C3. Seguidamente coge materia prima de tipo 2 y la deposita en el pallet. C3 transporta el pallet hasta el robot grúa, donde se puede quedar almacenado temporalmente. El transporte hasta la célula de ensamblado se realiza utilizando C2. El robot R1 descarga la materia prima del pallet, y transporta la pieza a la máquina de ensamblado.
Cuando en la máquina de ensamblado se dispone de una pieza de cada tipo, se procede a su ensamblado. R1 se encarga de retirar los productos acabados.
Una vez se descarga la materia prima de un pallet, R1 lo carga en C4, que se encarga de transportarlo al almacén, de donde es descargado por R2.

Restricciones del sistema

- Las cintas transportadoras sólo pueden transportar una parte (un pallet) cada vez, excepto C4, que puede llevar todos los pallets vacíos que haya al mismo tiempo.
- Se dispone de un total de 4 pallets, que inicialmente están en el almacén.
- El robot grúa puede llegar a almacenar 3 pallets.
- Cada máquina y cada robot sólo puede tener una pieza cada vez, excepto la célula de ensamblado que puede tener una de cada tipo.
- Se puede considerar que se dispone de materia prima de ambos tipos en cantidad ilimitada, y que el almacén donde se depositan los productos acabados tiene capacidad infinita.

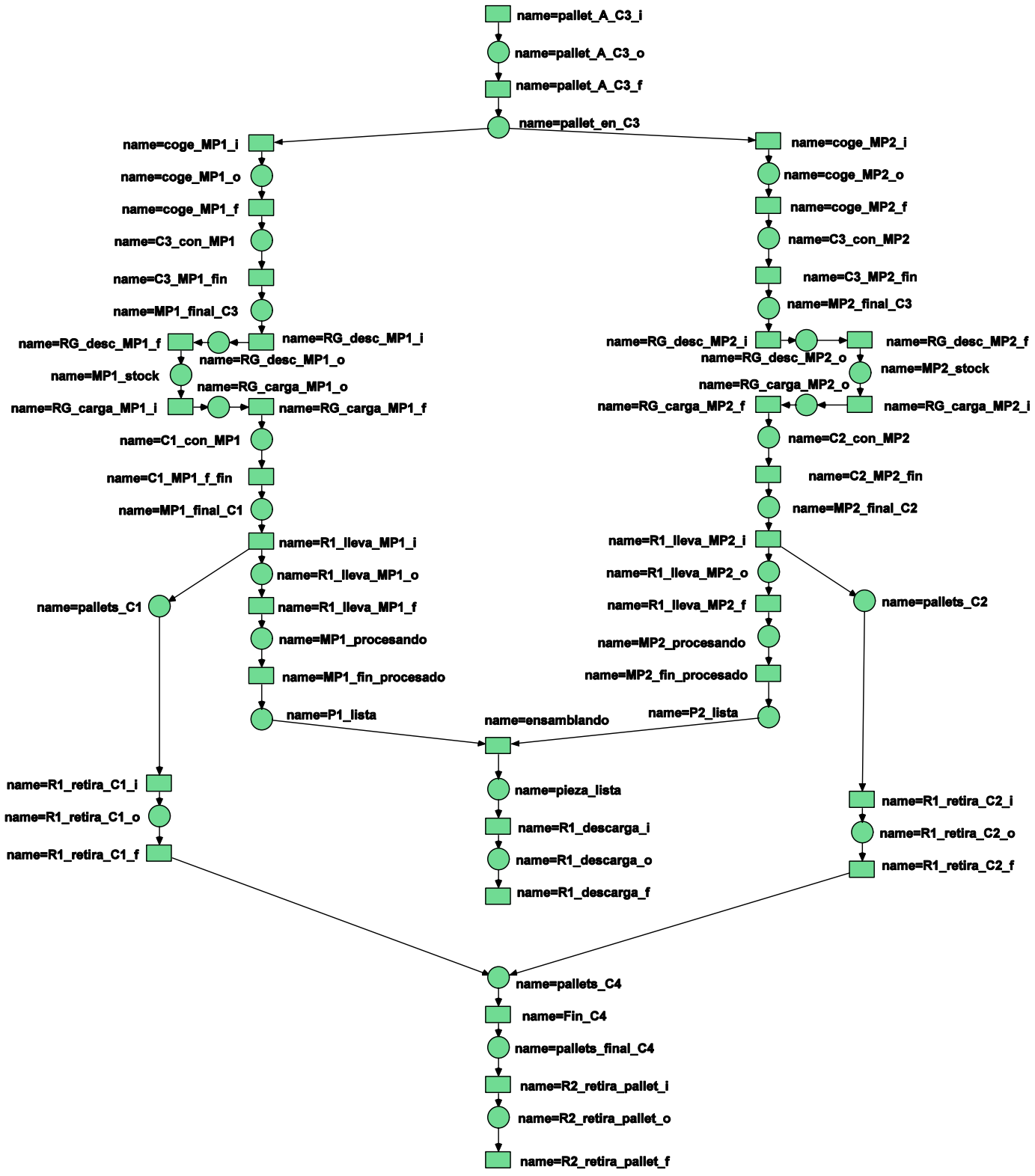
Modelado y validación

1) Dado que se trata de un sistema no trivial, se propone realizar el modelo de forma incremental:
   a) Obtener un modelo para representar el procesado de cada tipo de pieza y su ensamblado
   b) Añadir al modelo los elementos correspondientes al movimiento de los pallets
   c) Añadir las restricciones asociadas a los recursos necesarios para realizar cada actividad (robots, pallets, almacenes, cintas transportadoras,…)

2) Calcula los P- y T-semiflujos del sistema y estudia su sentido físico.

3) Si el grafo de alcanzabilidad del modelo tiene “demasiados” estados, el programa alcanza el límite de memoria disponible, y produce un mensaje de error. Utiliza las reglas de reducción para obtener un sistema más simple. Para reducciones de lugar-transición-lugar, o transición-lugar-transición basta con seleccionar la subred, y dar a **Net -> Coarsen subnet**, y luego eliminar etiquetas si es necesario.

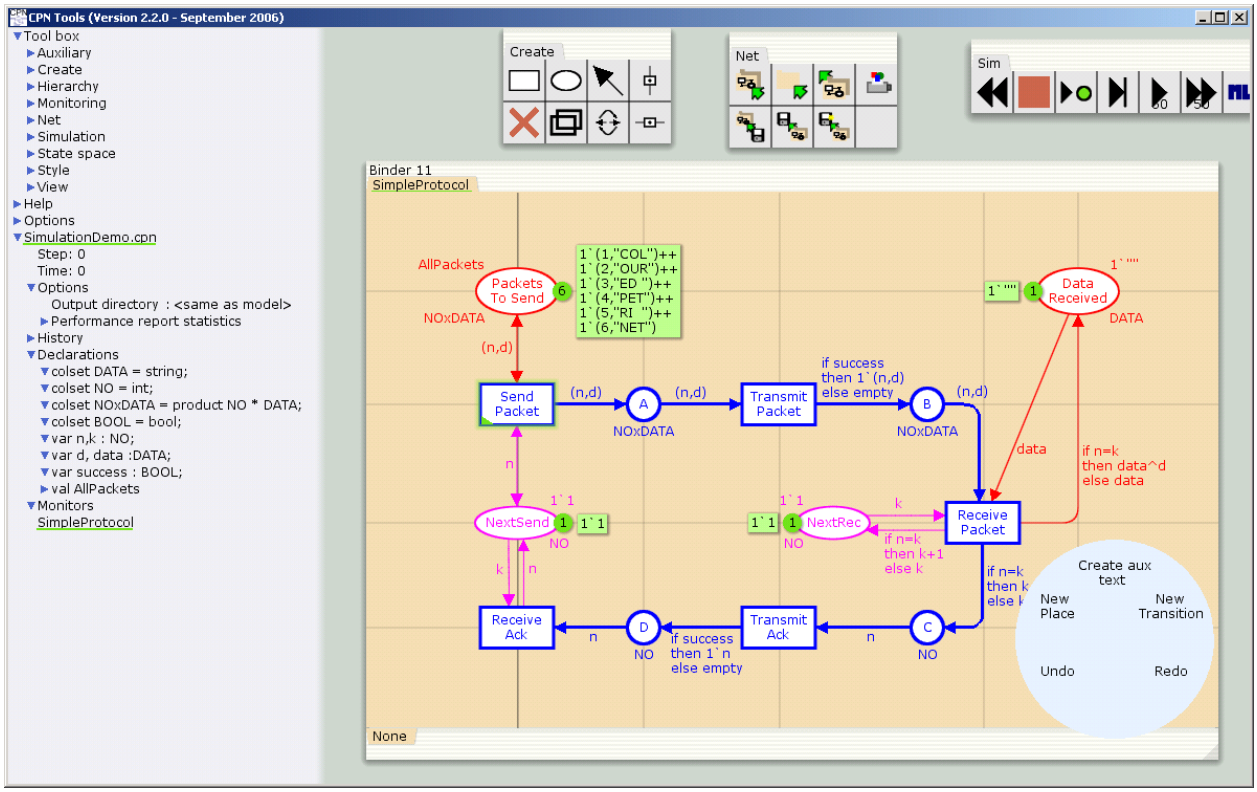
4) ¿Se puede llegar a bloquear el sistema? ¿Es vivo? En caso de que no lo sea, estudia cómo se pueden añadir mecanismos de control que eviten los problemas y sean lo menos restrictivos posible. ¿Funcionaría el mismo mecanismo si en lugar de haber 4 pallets, sólo se dispusiese de 3? Si no es así indica cómo lo modificarías.


name=pallet_A_C3_i
name=pallet_A_C3_o
name=pallet_A_C3_f
name=pallet_en_C3
name=coge_MP1_i
name=coge_MP1_o
name=coge_MP1_f
name=C3_con_MP1
name=C3_MP1_fin
name=MP1_final_C3
name=RG_desc_MP1_f
name=RG_desc_MP1_i
name=RG_desc_MP1_o
name=MP1_stock
name=RG_carga_MP1_o
name=RG_carga_MP1_i
name=RG_carga_MP1_f
name=C1_con_MP1
name=C1_MP1_f_fin
name=MP1_final_C1
name=R1_lleva_MP1_i
name=R1_lleva_MP1_o
name=R1_lleva_MP1_f
name=MP1_procesando
name=MP1_fin_procesado
name=P1_lista
name=pallets_C1
name=R1_retira_C1_i
name=R1_retira_C1_o
name=R1_retira_C1_f
name=coge_MP2_i
name=coge_MP2_o
name=coge_MP2_f
name=C3_con_MP2
name=C3_MP2_fin
name=MP2_final_C3
name=RG_desc_MP2_i
name=RG_desc_MP2_f
name=RG_desc_MP2_o
name=MP2_stock
name=RG_carga_MP2_o
name=RG_carga_MP2_f
name=RG_carga_MP2_i
name=C2_con_MP2
name=C2_MP2_fin
name=MP2_final_C2
name=R1_lleva_MP2_i
name=R1_lleva_MP2_o
name=R1_lleva_MP2_f
name=MP2_procesando
name=MP2_fin_procesado
name=P2_lista
name=pallets_C2
name=R1_retira_C2_i
name=R1_retira_C2_o
name=R1_retira_C2_f
name=ensamblando
name=pieza_lista
name=R1_descarga_i
name=R1_descarga_o
name=R1_descarga_f
name=pallets_C4
name=Fin_C4
name=pallets_final_C4
name=R2_retira_pallet_i
name=R2_retira_pallet_o
name=R2_retira_pallet_f

PRACTICA 2: Redes Coloreadas

Para el modelado y análisis de redes coloreadas vamos a utilizar la herramienta **CPN Tools.** Podéis encontrar más detalles en http://wiki.daimi.au.dk/cpntools-help/_home.wiki

En la parte izquierda de la ventana aparece el índice de las herramientas disponibles, e información sobre los modelos abiertos. Para utilizar las herramientas tenemos que colocar la paleta correspondiente en la parte derecha de la pantalla, lo que se consigue pinchando en el índice y arrastrando. Por ejemplo, en la imagen aparecen las paletas `Create, Net` y `Simulation.`



Además al pinchar con el botón derecho sobre algún objeto, aparece un menú circular que muestra distintas acciones posibles relacionadas con este objeto.

Al cargar una red, los elementos aparecen de color naranja mientras el programa chequea que la sintaxis es correcta. Si todo es correcto, este "halo naranja desaparece o se transforma en verde (depende del elemento), si hay algún error, se mantiene el color naranja o se transforma en rojo (depende del tipo de error).

Como ejemplo vamos a utilizar un sencillo protocolo de comunicación. El emisor tiene que mandar un conjunto de datos al receptor. La comunicación ocurre en una red insegura y los paquetes pueden perderse, o pueden llegar en distinto orden. Para poder asegurar que llegan en el orden correcto, se transmite asociado a cada dato un número que identifica su posición. Además, para asegurar que llegan todos los datos, el mismo dato se retransmite hasta que se recibe la confirmación de llegada. La confirmación consiste en que el receptor envíe al emisor el número asociado al dato que solicita (el siguiente del que ha recibido).

Un modelo de este sistema podéis encontrarlo en *CPNTools\Samples\demo\SimulationDemo.cpn*

Observa que cada lugar tiene asociado:

- un nombre (para facilitar la lectura del modelo),
- un tipo de dato asociado (los tipos de datos están definidos en la pestaña `declarations`),
- el marcado inicial denotado en forma de multiconjunto. El marcado actual, puede haber cambiado respecto al inicial en la simulación, aparece dentro de una caja (aparece también en un círculo el número total de tokens en el lugar). El marcado actual puede estar oculto, si en algún lugar no se ve el marcado, pulsar sobre él con el botón derecho y señalar `Show Marking.`

Para ir del nombre al tipo y de éste al marcado inicial se puede utilizar la tecla TAB.

En la pestaña `declarations` se pueden también definir constantes de tipo multiconjunto que se pueden usar para facilitar la escritura de los marcados iniciales, como en este caso `AllPackets`.

En los arcos aparecen variables y expresiones escritas en CPN ML. Las variables también tienen que estar definidas en la pestaña `declarations`, asignándoles un tipo de dato. También se pueden asociar guardas a las transiciones, lo que permite evitar las expresiones condicionales en los arcos. Por ejemplo, comparad el modelo anterior con *CPNTools\Samples\demo\SimulationDemo2.cpn.* Como para los lugares, la tecla TAB pasa del nombre a la guarda.

Para simular, pinchar sobre `ToolBox/Simulation` y arrastrar sobre la ventana de la derecha. Aparece una pequeña paleta de simulación. Los botones de izda. a dcha. son: volver al marcado inicial, detener una simulación, disparar una transición indicando los colores manualmente en el caso de que haya varias posibles sensibilizaciones activas, disparar una transición con asignación aleatoria de colores, ejecutar una secuencia de disparos (con asignación aleatoria) mostrando los marcados intermedios, ejecutar una secuencia de disparos (con asignación aleatoria) sin indicar todos marcados intermedios, evaluar una expresión CPN ML (no lo necesitaremos).

Aunque las simulaciones son útiles, no permiten garantizar al 100% que el comportamiento es el deseado. Para esto necesitamos calcular el grafo de alcanzabilidad del sistema. Sin embargo, sólo podemos calcular grafos que sean limitados, y el de nuestro ejemplo no lo es. Para poder probar esta herramienta, vamos a transformarlo limitando el número de paquetes que pueden estar en los lugares intermedios A y D (paquetes y acknowledgements que están siendo enviados por la red), dejando que entre los dos tengan como mucho tres datos. Para ello añadimos un lugar `Limit` de tipo UNIT, definido como `colset UNIT = unit;`
Unit es un tipo básico de dato que sólo contiene el valor (). Corresponde por tanto a un token no coloreado. En las expresiones de los arcos se indica ().

Pinchar sobre `ToolBox/State space` y arrastrar sobre la ventana de la derecha. La herramienta permite (de izda. a dcha. y de arriba a abajo): inicializar la herramienta, generar el espacio de estados, generar un grafo de componentes fuertemente conexas, guardar un resumen de la información obtenida en un fichero de texto, mostrar un nodo del grafo, mostrar los sucesores de un nodo, mostrar los predecesores de un nodo.
Empezar pinchando en inicializar, calcular espacio de estados, calcular componentes fuertemente conexas y guardar el resumen en un fichero de texto. NOTA: el cálculo del grafo le costará alrededor de 5 minutos.
Observar la información del fichero.
También se puede dibujar el grafo de alcanzabilidad (o una parte). Abrir una página nueva, dibujar el nodo 1 utilizando la opción de mostrar un nodo del grafo. A continuación, podemos ir ampliando el grafo con la herramienta para mostrar los sucesores de un nodo.

Modificar el modelo para representar las siguientes mejoras del protocolo.

1. El modelo no especifica un límite sobre el número de veces que un paquete se puede retransmitir. Modifica el modelo de forma que cada dato se transmita como máximo 3 veces.
2. Cuando se recibe un acknowledgment, se actualiza el contador de `NextSend` de acuerdo con el número del acknowledgement. Esto implica que el contador de `NextSend` puede disminuir cuando se recibe un acknowledgement “Viejo”. Modifica el modelo para que el contador de `NextSend` nunca disminuya.

Modelado de una gasolinera

1. Considera una gasolinera con cinco surtidores y un operario. Cuando llega un cliente elige un surtidor y va a la caja donde paga al operario (puede tener que hacer cola). Entonces el operario activa el surtidor correspondiente y el cliente puede echar gasolina. Cuando acaba se marcha.