INTERFAZ DE USUARIO PARA UN SIMULADOR DE REDES DE PETRI COLOREADAS

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per Marcos de Miguel Cruz
i dirigit per Mercedes Narciso
Bellaterra,.....de.....….....de 200...

El sotasignat, **Mercedes Elizabeth Narciso Farias**
Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en **Marcos de Miguel Cruz**

I per tal que consti firma la present.

Signat: .....
Bellaterra, ...de.....de 200.....

CAPÍTULO 1

INTRODUCCION

En este capítulo se presentan las motivaciones para el desarrollo de este proyecto, así como los objetivos del mismo y el contenido de este trabajo.

1.1 Motivaciones

En la actualidad, muchas empresas utilizan la simulación como herramienta de toma de decisiones, cuando desean mejorar la eficiencia operacional y organizacional, debido a la utilidad que presenta para evaluar el rendimiento y comportamiento de los sistemas bajo diferentes configuraciones de operación.

Los actuales paquetes de software comerciales de simulación, integran herramientas de optimización como herramientas de apoyo a la toma de decisiones, para dar una respuesta adecuada en entornos industriales, donde la toma de decisiones usualmente involucran algún riesgo económico o tecnológico.

En este contexto, se ha desarrollado una herramienta computacional basada en el formalismo de Redes de Petri Coloreadas, como técnica para el modelado y análisis de sistemas de eventos discretos [4,5], que permite simular y optimizar el comportamiento de tales sistemas, la herramienta requiere, entre otras, de una interfaz que permita al usuario visualizar e interpretar los resultados obtenidos de la simulación. Este proyecto constituye un paso más en el desarrollo de esta herramienta.

1.2 Objetivos

Este proyecto forma parte de un entorno para el modelado y análisis mediante simulación de sistemas de eventos discretos.

El objetivo principal es desarrollar una interfaz de usuario que permita visualizar el proceso de simulación e interpretar los resultados de la misma, así como presentar los resultados interpretados de la/s solución/es en forma textual o en forma gráfica.

Un segundo objetivo del proyecto consiste en integrar la interfaz en el entorno global de simulación de sistemas de eventos discretos.

1.3 Organización de la memoria

En el segundo capítulo se describen los conceptos necesarios para el desarrollo de este proyecto: Sistemas de eventos discretos, Redes de Petri, y Redes de Petri Coloreadas, como herramientas de modelado de sistemas de eventos discretos.

En el tercer capítulo se describen las fases en las que se ha dividido el desarrollo del proyecto. Se presenta una descripción detallada del análisis de requerimientos, de la fase de diseño del sistema así como la fase de codificación y pruebas.

El cuarto capítulo consiste en un manual de usuario de la interfaz desarrollada.

En el quinto capítulo se muestra un ejemplo de la utilización de la interfaz integrada en el entorno de simulación de eventos discretos.

En el sexto y último capítulo se presentan tanto las conclusiones que se han obtenido sobre el desarrollo de todas las fases del proyecto, como los objetivos alcanzados. Se presentan también algunas líneas futuras de investigación.

CAPÍTULO 2

FUNDAMENTOS TEÓRICOS

En este capítulo se describen los fundamentos teóricos necesarios para el desarrollo del proyecto. Se presenta una breve descripción de los sistemas de eventos discretos y de las Redes de Petri como introducción al formalismo de modelado de Redes de Petri Coloreadas y Redes de Petri Coloreadas temporales.

2.1 Sistema de Eventos Discretos

Los sistemas de eventos discretos son sistemas donde las propiedades de interés (estado del sistema) cambian en una secuencia de instantes de tiempo ante la ocurrencia de un evento, esta secuencia presenta un patrón aleatorio, se puede considerar que el resto del tiempo el estado del sistema permanece constante [5].

Debido a estas características los modelos de simulación de eventos discretos son dinámicos, estocásticos y discretos, donde las variables de estado cambian de valor en instantes no periódicos de tiempo.

Los Sistemas de Eventos Discretos se pueden describir mediante los siguientes elementos:

- **Entidades**: conjunto de componentes del sistema. Se pueden agrupar en dos tipos:
    - **Entidades permanentes o recursos**: son elementos estáticos en cuanto a que su número no varía en el sistema. Suelen utilizarse para describir los medios que permiten ejecutar las actividades del sistema. Los recursos definen quién o qué ejecuta la actividad.

  - **Entidades temporales**: se utilizan para definir los objetos que se procesan en el sistema o sufren cambios durante la ejecución del sistema. Las entidades temporales son los objetos que llegan, se procesan y salen del sistema. En general, se utiliza el término entidad para denominar sólo a las entidades temporales y así distinguirlas de los recursos.

- **Atributos**: se utilizan para caracterizar las entidades, tanto permanentes como temporales. Cada atributo corresponde a una propiedad, así como toda aquella información que fluye en el sistema junto con las entidades temporales. El estado de una entidad viene definido por los atributos asociados a ella, cuyos valores cambian en función de los eventos del sistema. Son imprescindibles para el control del flujo de las entidades en el sistema, así como también para la extracción de entidades de una cola.

- **Actividades**: son el conjunto de tareas que tienen lugar en el sistema, y por lo general suelen estar encapsuladas entre eventos. Casi todas las actividades involucran a más de un tipo de entidad y, por lo tanto, el inicio de una actividad está condicionado a la presencia de todos los tipos de entidades involucrados. La duración, a pesar de no ser un valor constante, de una actividad es un aspecto importante, por lo que ha de ser conocida para poder determinar el instante en el cual finalizará.

- **Evento**: puede definirse como una acción instantánea que puede cambiar el estado del sistema. En los modelos de sistemas de eventos discretos las variables de estado, es decir, toda la información asociada al sistema necesaria para evaluar su comportamiento, tan solo puede cambiar de valor en instantes de tiempo ligados a la ocurrencia de un evento, por tanto, un evento puede cambiar el valor de una variable de estado del sistema modelado. Los valores de las variables de estado quedan determinados por los valores que pueden tomar los atributos de las entidades. El conjunto de valores de las variables de estado en un

instante determinado constituye un estado del sistema. Por lo general, más de una actividad se inicia o finaliza con cada evento.

Se distinguen dos tipos de eventos:

- **Eventos condicionados**: para que estos eventos ocurran es necesario que se cumplan una o más condiciones.
- **Eventos no condicionados**: son eventos cuya ocurrencia está planificada en el tiempo y no depende de ninguna condición.

- **Colas o unidades de almacenamiento**: son estructuras que quedan determinadas a partir de una colección de entidades, en general entidades temporales, ordenadas de una forma lógica, por ejemplo de forma FIFO (primero en llegar, primero en salir). Las entidades que están en una cola sufren un retardo de duración indeterminada.

2.2 Red de Petri

Una Red de Petri [4,5] (en adelante RP) es un formalismo de modelado que permite representar un sistema de eventos discretos. La RP representa formalmente el paralelismo y la sincronización de un sistema, posibilitando efectuar un análisis cuantitativo de este.

Las RPs se representan como un grafo dirigido con los siguientes elementos:

- **Nodos lugar**: Gráficamente se representan mediante círculos y se utilizan para describir tanto las colas de espera del sistema, como las condiciones sobre el estado en que se encuentran las entidades o recursos que componen el sistema.
- **Nodos transición**: Se representan gráficamente mediante rectángulos y se usan para modelar los eventos y/o actividades que aparecen en la dinámica del sistema.

- **Arcos dirigidos**: se usan para conectar un nodo lugar con un nodo transición o un nodo transición con un nodo lugar. Pero nunca dos nodos del mismo tipo. Una flecha es su representación gráfica.

- **Peso asociado al arco**: permite describir tanto las condiciones necesarias para que el evento y/o actividad representada por una transición pueda ocurrir, así como los efectos sobre el estado del sistema después de la ocurrencia de un evento o la ejecución de una actividad.

- **Marcas o Tokens**: se utilizan para modelar entidades dentro de un nodo lugar, o bien el cumplimiento de una condición necesaria para la ocurrencia de un evento o ejecución de una actividad. Se representan gráficamente mediante puntos dentro de los nodos lugar.

- **Marcado o Marcación**: representa cualquier distribución de marcas en los nodos lugar. La distribución inicial de marcas en los nodos lugar se denomina *marcado inicial*. Cada marcado representa un estado del sistema.

En la figura 2.1 se ha representado una RP con cuatro nodos lugares (P_1, P_2, P_3 y P_4), y un nodo transición (T_1), el número junto a cada arco representa el peso asociado a este, por ejemplo, el peso asociado al arco que conecta el nodo lugar P_1 al nodo transición T_1 es de 2, y el peso asociado a los arcos que van del nodo lugar P_2 al nodo transición T_1, del nodo transición T_1 al nodo lugar P_3, y del nodo transición T_1 al nodo lugar P_4 es de 1.

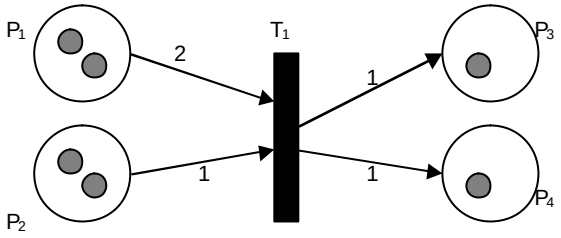

Figura 2.1: Representación de una RP

En la figura 2.1, un nodo lugar, por ejemplo el nodo lugar P_1, es un *nodo lugar de entrada* de un nodo transición, por ejemplo el nodo transición T_1, sí y solo sí existe un arco dirigido desde P_1 a T_1. Un nodo lugar, por ejemplo el nodo lugar P_3, es un nodo lugar de salida de un nodo transición, por ejemplo el nodo transición T_1, sí y solo sí existe un arco dirigido desde T_1 a P_3.

Se dice que una transición T_i esta habilitada si todos los nodos de tipo lugar L_j conectados a su entrada tienen al menos $w(L_j, T_i)$ elementos. Donde $w(L_j, T_i)$ representa el peso del arco que une el nodo lugar L_j con la transición T_i. Una transición estará deshabilitada en caso contrario. Una transición habilitada puede dispararse en cualquier momento. Como resultado de disparar una transición habilitada se eliminan $w(L_j, T_i)$ elementos de cada nodo lugar L_j de entrada hacia la transición T_i, añadiendo, a su vez, $w(T_i, L_k)$ elementos a cada nodo lugar L_k de salida de la transición T_i.

En la figura 2.1 se puede observar que la transición T_1 está habilitada para su disparo al cumplir las condiciones impuestas por los arcos que conectan los nodos lugares de entrada (P_1 y P_2) con la transición, es decir, el nodo lugar P_1 contiene 2 tokens y el arco que lo conecta con la transición T_1 tiene un peso asociado de 2, y el nodo lugar P_2 contiene 2 tokens y el arco que lo conecta con la transición T_1 tiene un peso asociado de 1. Una vez que se ha disparado la transición, se procede a eliminar 2 tokens del nodo lugar P_1, quedando sin ningún token y un token del nodo lugar P_2, quedando con un único token. Y además se añaden en los nodos lugares de salida (P_3 y P_4), a los tokens que

ya contienen, el número de tokens marcado por el peso asociado al arco de salida que los conecta con la transición T_1, es decir, al nodo lugar P_3, que ya tenía un token, se le añade otro token, quedando al final con 2 tokens, y, en este caso, sucede lo mismo con el nodo lugar P_4. El resultado del disparo de la transición se observa en la figura 2.2.

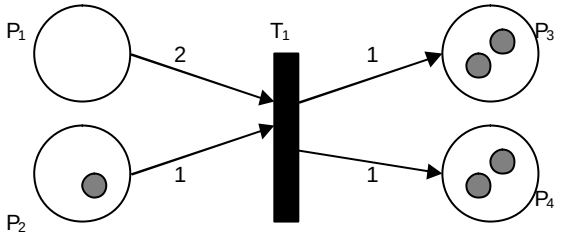

Figura 2.2: RP de la figura 2.1 después de disparar la transición

2.3 Red de Petri Coloreada

Las Redes de Petri Coloreadas [4,5] (en adelante RPC) ofrecen la posibilidad de formalizar tanto los atributos o características de las entidades, como las propiedades que estas deben tener para que una transición esté habilitada. Además de los elementos básicos de una RP, tenemos en una RPC los siguientes elementos de representación:

- **Conjunto color**: cada nodo lugar tan solo puede tener marcas con el mismo tipo de datos, el cual es conocido como conjunto color del lugar. Gráficamente se representa con el nombre del conjunto color a un lado del nodo lugar.

- **Expresiones de inicialización**: representan el número de marcas y el color de cada una de ellas en un nodo lugar. La representación gráfica de la expresión de inicialización consiste dentro de un número en un círculo al lado del nodo que indica el número de marcas que contiene ese nodo lugar. Los colores de las marcas se especifican con una expresión subrayada junto al nodo lugar:

$$\underline{n'(c_1,c_2,\ldots,c_k)}$$

donde:

n: representa el número de marcas con los valores descritos dentro del paréntesis

c_i: representa el valor de un componente del color.

Cuando los valores de los colores de las marcas no son idénticos para todos los elementos del mismo nodo lugar, se utiliza el operador "+" para especificar los valores de los colores de cada marca:

$$n_1'(c_{11},c_{12},\ldots,c_{1k})+n_2'(c_{21},c_{22},\ldots,c_{2k})$$

- **Estado inicial**: se determina evaluando las expresiones de inicialización asociadas a cada nodo lugar, las cuales determinaran el número de marcas en cada nodo lugar, así como los valores de los colores de las marcas.

- **Expresiones de arco**: consisten en la formalización de restricciones entre los colores de las distintas marcas de los nodos lugar conectados a la entrada de una transición, para la cual pueden utilizarse variables que una vez instanciadas a los valores concretos de los colores de las marcas, fuerzan a una selección de aquellas marcas cuyos colores coincidan con los valores de las variables instanciadas.

Las transiciones estarán habilitadas en función del número de marcas en los nodos lugar, así como en función del tipo (colores) de las marcas disponibles en dicho nodo lugar, definiendo nuevos colores para las marcas de salida.

- **Guardas**: son expresiones lógicas que imponen ciertos valores a los colores de las marcas que pueden ser escogidas para habilitar una transición. Gráficamente se representan entre corchetes "[ ]" situados al lado del nodo transición.

- **Marcado o Marcación**: es la especificación del número de marcas en cada nodo lugar, así como los valores de los colores de cada una de las marcas. Representa la información mínima y necesaria para poder predecir cuales son los posibles eventos y/o actividades que pueden producirse. El marcado inicial (estado inicial del sistema) se identifica como M_0.

En la figura 2.3 se ha representado una RPC con dos nodos lugares (*A* y *B*) y dos nodos transición (T_1 y T_2), sobre cada nodo lugar hay dos expresiones subrayadas que corresponden a las expresiones de inicialización, estas indican el número de marcas y los valores de color de cada una de ellas que representa la configuración inicial del sistema. El tipo de marcas que pueden contener cada nodo lugar se indica mediante el nombre de un conjunto color que se escribe junto a cada nodo lugar (el conjunto color *C* para el nodo lugar *A*, y el conjunto color *E* para el nodo lugar *B*). Las expresiones que aparecen junto a cada arco (expresiones de arco) deben ser evaluadas para determinar si la transición está o no habilitada. Y la expresión lógica *[a=b]* junto a la transición T_1 es una guarda que indica que la transición solo podrá dispararse si además de las expresiones de arco, también se satisface que el valor del atributo de color *a* de la marca del nodo lugar *A* es igual al atributo de color *b* del nodo lugar *B*. En el lado derecho de la figura se describen los valores que se pueden representar en los conjuntos color, y se detalla expresión inicial para el sistema modelado.

En el caso de las RPCs, la información sobre el estado del sistema se obtiene analizando la cantidad y los colores de las marcas almacenadas en los distintos nodos lugar. Por ejemplo el estado que representa la RPC de la figura 2.3 es el siguiente:

$$M_0 = [3'(1)+ 3'(2)+ 3'(3)/1'(1,0)+ 1'(2,0)+ 1'(3,0)]$$

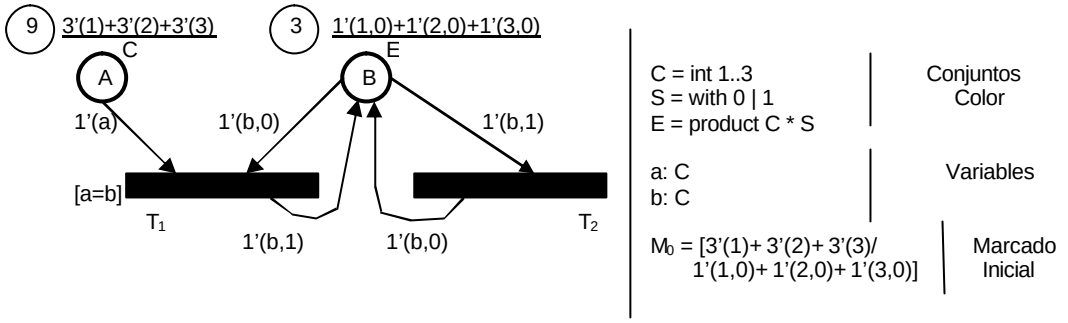

Figura 2.3: Representación de una RPC

Al igual que sucede con las RPs, una transición está habilitada para ser disparada cuando se cumplen todas las condiciones que indican las expresiones de los arcos de los nodos lugares de entrada a la transición. Eliminando de cada nodo lugar de entrada el número de tokens de con los colores que indica la expresión de arco asociada al arco que lo conecta con la transición, y añadiendo en los nodos lugares de salida el número de tokens de cada color que indica la expresión de arco asociada al arco que los conecta con la transición.

En la figura 2.4 se muestra un ejemplo de una RPC con una transición habilitada para ser disparada, y el estado en el que queda la RPC después del disparo de la transición. Se observa como la transición T_1 está habilitada al cumplir los nodos lugares *A* y *B* las condiciones de sus respectivos arcos que los conectan a la transición, es decir contener al menos un token del tipo adecuado en el nodo lugar *A* (*3'(1)*) y otro token, también del tipo necesario en el nodo lugar *B* (*1'(1,0)*), y que además cumplen la guarda de la transición *[a=b]* (*3'(**1**) = 1'(**1**,0)*). Una vez se ha producido el disparo de la transición, se observa que el modelo ha cambiado de estado, pasando de M_0=[3'(1)+3'(2)+3'(3)/ 1'(1,0)+1'(2,0)+1'(3,0)] a M_1=[2'(1)+3'(2)+3'(3)/ 1'(1,1)+1'(2,0)+1'(3,0)]

2.4 Redes de Petri Coloreadas Temporales

Las RPCs pueden ampliarse incluyendo en los modelos el concepto de tiempo. Esto se consigue mediante la introducción de un reloj global discreto, y asociando a cada marca un valor de tiempo. El valor de tiempo describe el instante de tiempo más temprano o próximo en el cual una marca puede ser usada, es decir, eliminada de un nodo lugar como consecuencia del disparo de una transición. El valor de tiempo de las marcas que serán eliminadas debe ser menor o igual que el tiempo actual del reloj global. El reloj global del modelo RPC permanece con el mismo valor de tiempo mientras haya marcas que puedan ser utilizadas para disparar cualquiera de las transiciones. Una vez que

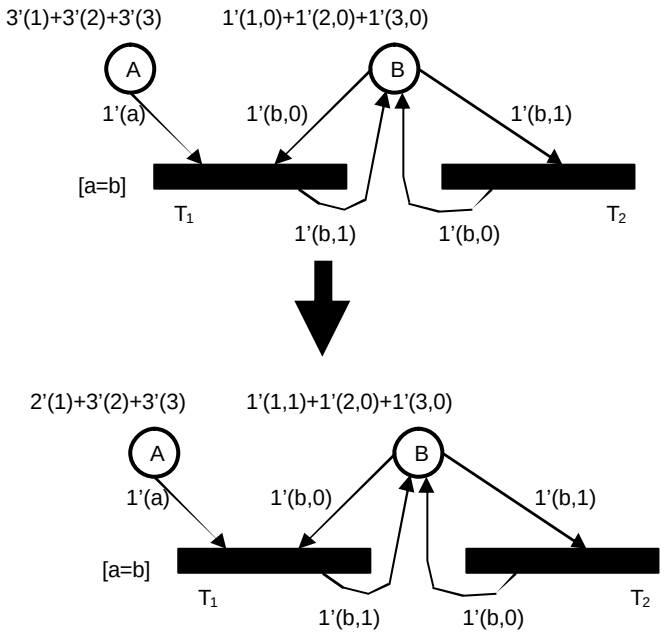

Figura 2.4: Disparo de una transición en una RPC

se han disparado todas las transiciones posibles para el actual valor de tiempo del reloj global, el reloj global se incrementa al tiempo próximo más cercano en el cual al menos una transición esté habilitada. Para modelar que una actividad

o evento correspondiente al disparo de una transición toma *r* unidades de tiempo, a las marcas añadidas a los nodos lugar de salida de la transición se le asocia un valor de tiempo el cual es *r* unidades de tiempo mayor que el valor de tiempo del reloj en el cual se disparó la transición, las marcas no estarán disponibles durante *r* unidades de tiempo y no pueden ser eliminadas mediante el disparo de las transiciones, antes de que el reloj haya sido incrementado en al menos *r* unidades de tiempo.

Para incorporar el concepto de tiempo en las RPCs es necesario introducir al modelo RPC los siguientes elementos:

- **Reloj global**: representa el tiempo inmediatamente anterior a la ocurrencia de un evento y/o ejecución de una actividad (disparo de una transición).
- **Valor de tiempo (sello de tiempo)**: asociado a cada marca en un marcado, representa el tiempo más próximo en el que la marca puede ser utilizada para habilitar una transición.
- **Tiempo de retardo**: asociado a una transición, indica el tiempo requerido para la ejecución de una actividad asociada a una transición.
- **Tiempo de llegada**: representa la suma del valor del reloj más el tiempo de retardo asociado a la transición que cambia el estado del sistema a un nuevo estado. Representa el tiempo en el que ocurre un cambio en el sistema.

En las RPCs temporales, una transición está habilitada para su disparo cuando un nodo lugar contiene marcas que cumplen las correspondientes expresiones de arco de entrada a la transición, y todos los tiempos asociados a las marcas son menores o iguales que el valor actual del reloj, en caso contrario, el reloj global debe avanzar hasta el mayor valor de sello de tiempo de las marcas para las cuales la transición estará habilitada.

En la tabla 2.1 se puede observar la representación del concepto de tiempo sobre el sistema modelado en la figura 2.3, la representación del valor de tiempo de las marcas es *n[t]*, donde *n* representa el número de marcas con el valor de tiempo especificado entre corchetes, y *t* es el valor de tiempo.

Tiempo transición T_1:	1.00
Tiempo transición T_2:	15.00
Reloj:	0.00
Tiempo de llegada:	0.00

Lugares	A	B
Marcas	3’(1)+3’(2)+3’(3)	1’(1,0)+1’(2,0)+1’(3,0)
Tiempos	3[0.00]+3[0.00]+3[0.00]	1[0.00]+1[0.00]+1[0.00]

Tabla 2.1: Marcado M_0 de la RPC temporal aplicada a la figura 2.3

En la tabla 2.2 muestra la representación del concepto de tiempo sobre el sistema modelado en la figura 2.4 , una vez se ha disparado la transición T_1.

Tiempo transición T_1:	1.00
Tiempo transición T_2:	15.00
Reloj:	0.00
Tiempo de llegada:	1.00

Lugares	A	B
Marcas	**2’(1)**+3’(2)+3’(3)	**1’(1,1)**+1’(2,0)+1’(3,0)
Tiempos	**2[1.00]**+3[0.00]+3[0.00]	**1[1.00]**+1[0.00]+1[0.00]

Tabla 2.2: Marcado M_j de la RPC temporal aplicada a la figura 2.4

2.5 Árbol de alcance

Un espacio de estados es un conjunto de todos los posibles estados y cambios de estado que pueden ocurrir en el sistema que representa.

El árbol de alcance es una herramienta de análisis que proporciona las RPCs para construir el espacio de estados, también llamado grafo de ocurrencia, del sistema descrito por la RPC. El árbol de alcance contiene todas las secuencias posibles de acciones, disparos de transiciones, que pueden ocurrir en el sistema, además de todos los marcados alcanzables.

La idea básica es la construcción de un grafo a partir de un nodo que representa el marcado inicial, y que contenga un nodo por cada marcado alcanzable en el sistema modelado [4], y un arco por cada disparo de una transición. La figura 2.5 muestra parte de un árbol de alcance para el modelo RPC de la figura 2.3, cada nodo representa un marcado (M_0, M_1, M_2, …) y el contenido del marcado se describe mediante la especificación de las marcas presentes en cada nodo lugar, en la primera columna el nodo lugar *A* y en la segunda el nodo lugar *B*. Cada arco representa el disparo de una transición. La especificación de la transición, así como las marcas que la habilitaron se describe mediante el texto asociado al arco. El nodo con el borde más grueso representa el marcado inicial.

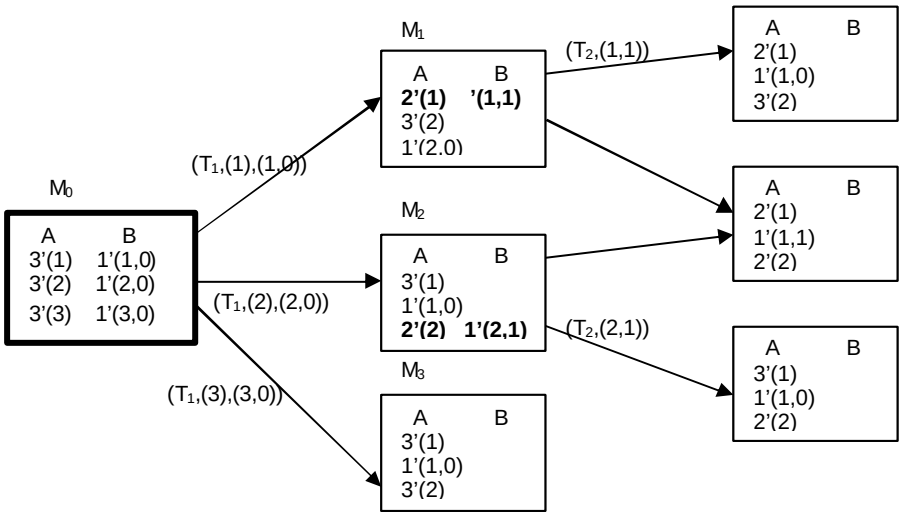

Figura 2.5: Árbol de alcance del modelo representado en la figura 2.3

Para el caso de las RPCs temporales, la información sobre los tiempos asociados a las marcas debe ser incorporada a los nodos del árbol de alcance.

El algoritmo para la generación del árbol de alcance es el siguiente:

- La raíz del árbol es el marcado inicial (estado inicial del sistema).
- Para cada nodo se generan un nodo hijo por cada transición habilitada para ese nodo, cada nodo hijo representa el marcado de la RPC una vez se ha disparado una transición habilitada.
- Si ya existe el nodo hijo en el mismo u otro nivel del árbol se marca como *nodo repetido* y no se generan nuevos nodos a partir de él.
- Un nodo sin transición habilitada se marca como *nodo hoja.*
- Un nodo que no se ajusta a los dos tipos anteriores se marca como *nodo nuevo*, indicando así que se puede generar nodos nuevos a partir de él.

CAPÍTULO 3

DESARROLLO DE LA INTERFAZ

En este capítulo se describen las fases en las que se ha dividido el desarrollo del proyecto. Se presenta una descripción de las herramientas utilizadas, del análisis de requerimientos, de la fase de diseño del sistema y por último de la fase de implementación e integración del proyecto en el entorno global del simulador de RPCs.

3.1 Breve Descripción de UML

Debido a las características del proyecto, el cual forma parte de un entorno implementado utilizando una metodología de desarrollo de software orientado a objetos, usando la herramienta Visual C++, se optó por la utilización de la metodología UML para el desarrollo del mismo.

UML es un método para crear diagramas de análisis, diseño e implementación de software [2], concebido por los autores de los tres métodos más usados de orientación a objetos: Grady Booch, Ivar Jacobson y Jim Rumbaugh. El objetivo principal cuando se empezó a gestar UML era posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado. Para ello era necesario definir una notación y semántica común.

Los diferentes diagramas que dispone UML se pueden clasificar por las fases de desarrollo del software, para la fase de análisis de requerimientos se utilizan los diagramas de clases, para la fase de diseño del sistema se utilizan los diagramas de actividad, los diagramas de secuencia, y los diagramas de colaboración.

3.1.1 Diagramas de Clases

Un diagrama de clases [2] describe las clases principales del sistema con sus relaciones estructurales y de herencia.

Una clase se representa mediante un rectángulo subdividido en tres partes: en la parte superior se muestra el nombre de la clase, en la parte central los atributos y en la parte inferior las operaciones.

Las clases se relacionan mediante asociaciones, representadas por una línea que une dos clases. La dirección de una asociación se puede representar con una flecha que indica la dirección en la que se debe leer la asociación, las asociaciones bidireccionales se representan como una línea sin flecha entre las dos clases. La multiplicidad de la asociación es la restricción que limita el número de instancias de una clase que pueden tener esa asociación con una instancia de la otra clase. Esta se representa con un número fijo, un intervalo de valores (*[1..3]*), un intervalo donde uno de los extremos es un asterisco (intervalo abierto *[1..*]*), una combinación de las opciones anteriores, o con un asterisco que indica que puede tomar cualquier valor en cada extremo de la asociación. Para indicar el papel que juega una clase en una asociación se puede especificar un nombre de rol en el extremo de la asociación junto a la clase que desempeña dicho rol.

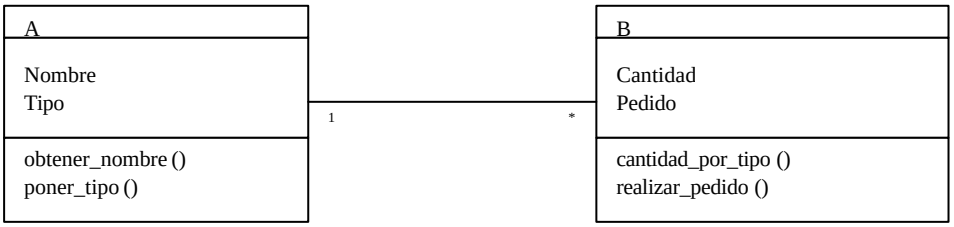

Figura 3.1: Diagrama de Clases

En la figura 3.1 se observa un ejemplo de diagrama de clases que se compone de dos clases (A y B) con sus atributos (Nombre y Tipo para la clase A, y Cantidad y Pedido para la clase B) y operaciones (obtener_nombre () y poner_nombre () para la clase A, y cantidad_por_tipo () y realizar_pedido () para la clase B). Las dos clases se relacionan entre si a través de una asociación con la siguiente multiplicidad, una instancia de la clase A se relaciona con un número indeterminado de instancias de la clase B.

3.1.2 Diagramas de Actividades

Un diagrama de actividades [2] describe grupos secuenciales y concurrentes de actividades. Una actividad o un estado de actividad representa la ejecución de una secuencia en un procedimiento, o el funcionamiento de una actividad en un flujo de trabajo. Las actividades se enlazan por transiciones automáticas, y cuando una actividad termina se desencadena el paso a la siguiente actividad. Un diagrama de actividades es útil para entender el comportamiento de alto nivel de la ejecución de un sistema, sin profundizar en los detalles internos. Los parámetros de entrada y salida de una acción se pueden mostrar usando las relaciones de flujo que conectan la acción y un estado de flujo de objeto.

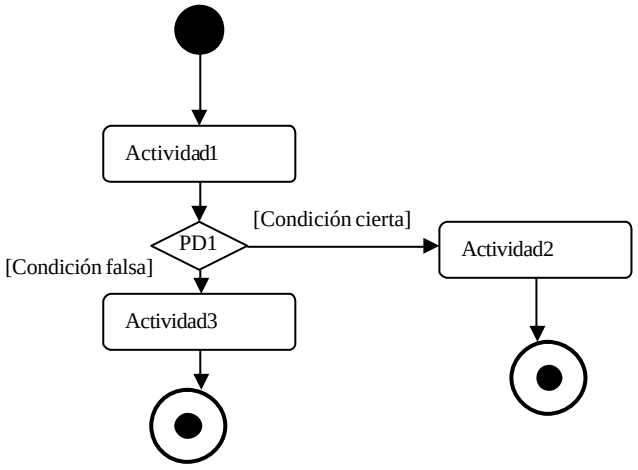

Figura 3.2: Diagrama de Actividad

Un diagrama de actividades puede contener, además de los estados de actividad, estados de acción, similares a los anteriores, pero son atómicos y no permiten transiciones mientras están activos. Se utilizan para operaciones cortas de mantenimiento. También pueden contener bifurcaciones, así como divisiones de control en hilos concurrentes, o actividades que se pueden realizar concurrentemente por diversos objetos o personas, simultáneamente o en cualquier orden.

En la figura 3.2 se puede observar tres estados de actividad (*Actividad1*, *Actividad2* y *Actividad3*) conectados por arcos que representan las transiciones entre ellas, y un Punto de Decisión (*PD1*) donde el hilo ejecución se bifurca en dos opciones. El diagrama se inicia en un estado inicial, representado por un círculo negro, y finaliza en dos posibles estados final, representados mediante dos círculos blancos con un punto negro en su interior.

3.1.3 Diagramas de Secuencia

Un diagrama de secuencia [2] muestra una interacción ordenada según la secuencia temporal de eventos. Los objetos participantes en la interacción y los mensajes que intercambian ordenados según su secuencia en el tiempo. El eje vertical representa el tiempo y en el eje horizontal se colocan los objetos y actores participantes en la interacción, un actor es una entidad externa al sistema que realiza algún tipo de interacción con el mismo. Cada objeto o actor tiene una línea vertical, y los mensajes se representan mediante flechas entre los distintos objetos. El tiempo fluye de arriba abajo.

En la Figura 3.3 se muestra un ejemplo de diagrama de secuencia compuesto por dos objetos (*A* y *B*) y cuatro mensajes (*producto ()*, *obtener_nombre ()*, *nombre ()*, y *poner_tipo ()*) ordenados temporalmente, donde el primero corresponde al mensaje que se encuentra en la parte superior de la figura, *producto ()*, y el último, en la parte inferior, *poner_tipo ()*.

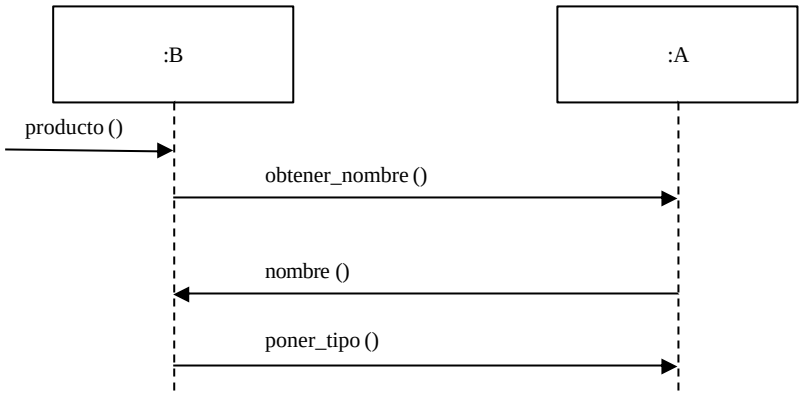

Figura 3.3: Diagrama de secuencia

3.1.4 Diagramas de Colaboración

Un diagrama de colaboración [2] muestra una interacción organizada basándose en los objetos que toman parte en la interacción y los enlaces entre los mismos. A diferencia de los diagramas de secuencia, estos muestran las relaciones entre los roles de los objetos. La secuencia de los mensajes y los flujos de ejecución concurrentes deben determinarse explícitamente mediante números de secuencia.

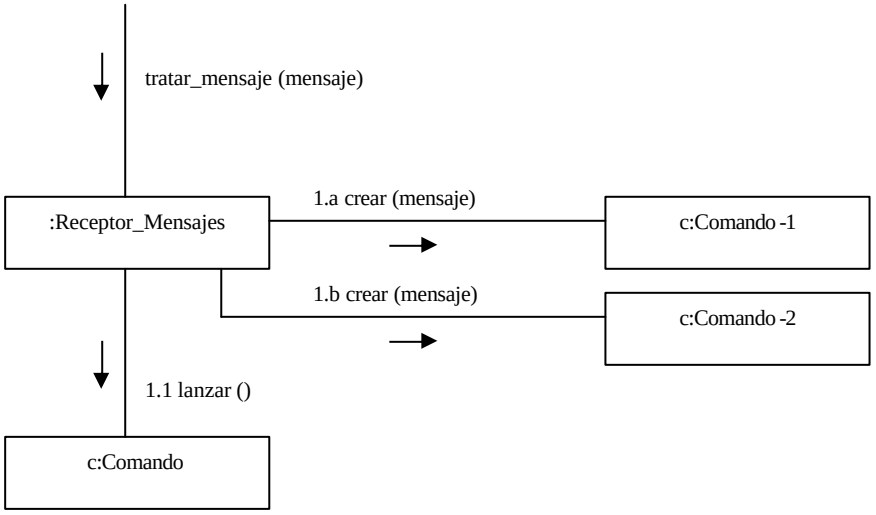

Figura 3.4: Diagrama de colaboración

La representación muestra los objetos con sus correspondientes enlaces, y los mensajes que se intercambian. Los mensajes son flechas que van junto al enlace, con el nombre del mensaje y los parámetros entre paréntesis, si los tiene. Cada mensaje lleva un número de secuencia que marca cual es el mensaje que le precede, excepto el mensaje que inicia el diagrama.

En la figura 3.4 se observa un diagrama de colaboración. El inicio del diagrama lo marca el mensaje sin número (*tratar_mensaje (mensaje)*), que llega hasta el objeto denominado *:Receptor_Mensajes*, desde este parten tres mensajes, dos de ellos anidados, *1.a crear_mensaje (mensaje)* y *1.b crear_mensaje (mensaje)*, que no puede comenzar a ejecutarse hasta que no finalice 1.a. El tercer mensaje, *1.1 lanzar ()*, no tiene parámetros por lo que se deja vacío el espacio entre paréntesis. Cada uno de los mensajes tiene un objeto destino (*c:Comando-1*, c*:Comando-1*, y *c:Comando* respectivamente).

En las siguientes secciones se describe la aplicación de la metodología UML para el desarrollo de este proyecto.

3.2 Análisis de Requerimientos

En la Figura 3.4 se muestra un esquema del entorno de simulación de RPC donde se enmarca el desarrollo de este proyecto (4), el cual consiste en el desarrollo de una interfaz para un simulador de RPC (3).

La interfaz a desarrollar deberá disponer las funcionalidades necesarias tanto para la visualización de la ejecución del simulador de RPCs, como para visualizar los resultados de la simulación de una RPC.

Los datos que debe manejar la interfaz son:

- Una base de datos (en adelante BBDD) que contiene la descripción de la RPC que se utilizará en la simulación (nodos lugares, transiciones, arcos, marcas). La base de datos es generada por la interfaz de captura de datos (1) (figura 3.4).

- El archivo que contiene el Árbol de Alcance generado por el simulador de RPC (8).

- El/Los archivo/s solución/es en caso de que se desea visualizar una simulación finalizada (9).

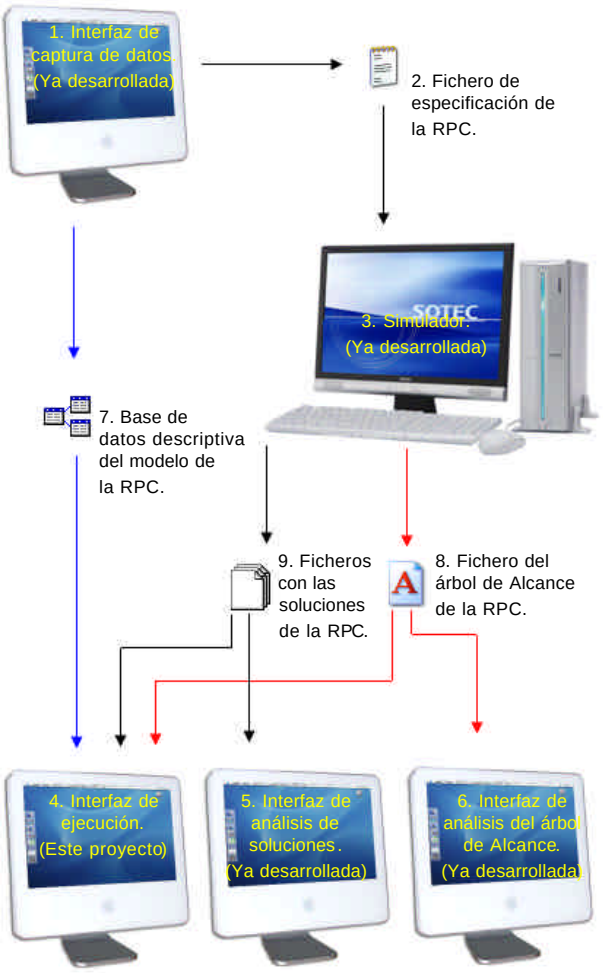

Figura 3.4: Entorno de Simulación de RPC

La interfaz de simulación está pensada para visualizar el proceso de simulación, así como para visualizar los resultados generados por el simulador de RPC. A fin de tener un registro del uso de la interfaz, se ha decidido posibilitar la generación de un archivo de texto (fichero log) con la secuencia de operaciones realizadas por el usuario en la interfaz de simulación ordenadas

por orden temporal, a fin de permitir a cualquier usuario comprobar la secuencia de pasos que se han realizado en ejecuciones anteriores.

3.2.1 Visualización de la Ejecución de la Simulación

El objetivo principal de la interfaz de simulación consiste en permitir visualizar la simulación de una RPC en tiempo real, mostrando en todo momento, el estado de la simulación que se está ejecutando. Durante su ejecución, el simulador de RPCs genera un archivo de texto en el cual, escribe todos los marcados que se generan mientras realiza el desplegado del árbol de alcance completo del modelo de RPC que se está analizando. A partir de este archivo, la interfaz de simulación, es capaz de mostrar cada marcado que ha generado el simulador.

El formato del archivo de texto que se genera en la simulación es el siguiente:

```
NODO: 4; NIVEL: 3;
EVALUACION: 8,0////32/3/;

TRANSICION: T1; TIEMPO: 500.00; RELOJ: 180.00; COSTO: 0.00;

TOKENS:
1'(10,1)+1'(7,1)+1'(8,1)+1'(9,0)/1'(5,0)+1'(6,0)/1'(3,0)+1'(4,0)/1'(1,0)+1'(2,0)/1'(20)+1'(30)+
1'(48)/1'(1)+1'(2)+1'(3)/;
TIEMPOS:
1[100.00]+1[180.00]+1[500.00]+1[0.00]/1[0.00]+1[0.00]/1[0.00]+1[0.00]/1[0.00]+1[0.00]/1[
0.00]+1[0.00]+1[0.00]/1[0.00]+1[0.00]+1[500.00]/;
```

donde:

- **NODO**: identificación única de un marcado dentro del árbol de alcance.
- **NIVEL**: indica la profundidad del marcado dentro del árbol de alcance.
- **EVALUACION**: indica los valores de color de los tokens implicados en el disparo de la transición que ha generado la marcación.
- **TRANSICION**: nombre de la transición que se ha disparado.
- **TIEMPO**: indica el tiempo del sistema después del disparo de la transición.

- **RELOJ**: indica el instante de tiempo del sistema inmediatamente antes del disparo de la transición.
- **COSTO**: indica el costo asociado al nuevo marcado (nuevo estado alcanzado).
- **TOKENS**: indica la cantidad y el valor de color de cada token del marcado. Representa el estado del sistema.
- **TIEMPOS**: indica los tiempos asociados a cada token del marcado.

3.2.2 Visualización de los Resultados de la Simulación

Otro objetivo de la interfaz de simulación consiste en permitir visualizar los resultados obtenidos de la simulación de una RPC, posibilitando visualizar cada marcado del resultado de forma independiente, así como visualizar el resultado completo, una descripción de este y una representación gráfica mediante diagramas de Gantt.

El formato del archivo de solución generado por el simulador es el siguiente:

```
Transicion disparada: T1 con tokens: 5,5,12/6/

NODO: 2, NIVEL: 1, TIEMPO LLEGADA: 1.50, RELOJ: 0.00, COSTO: 60.00
(1)1'(1,1,12)+1'(2,2,12)+1'(3,3,12)+1'(4,4,12)+1'(5,6,12)/1'(5)+1'(7)+1'(8)/
(2)1[0.00]+1[0.00]+1[0.00]+1[0.00]+1[1.50]/1[1.50]+1[0.00]+1[0.00]/
```

donde:

- **Transicion disparada**: nombre de la transición que se ha disparado.
- **con tokens**: indica los valores de color de los tokens implicados en el disparo de la transición que ha generado la marcación.
- **NODO**: identificación única del marcado dentro del árbol de alcance.
- **NIVEL**: indica la profundidad del marcado dentro del árbol de alcance.
- **TIEMPO LLEGADA**: indica el tiempo del sistema después del disparo de la transición.
- **RELOJ**: indica el instante de tiempo del sistema inmediatamente antes del disparo de la transición.

- **COSTO**: indica el costo asociado al nuevo marcado (nuevo estado alcanzado).
- **(1)**: indica la cantidad y el valor de color de cada token del marcado. Representa el estado del sistema.
- **(2)**: indica los tiempos asociados a cada token del marcado.

3.2.3 Diseño de la Interfaz

La interfaz principal se compondrá de las siguientes partes, según se ilustra en la figura 3.5:

- **La Zona de dibujo**: Es la parte de la interfaz donde se dibujarán las transiciones y los nodos lugares que se quieran visualizar. Los nodos lugares se visualizarán como un círculo que contiene grupos de pequeños cuadros agrupados horizontalmente, cada grupo de cuadros representará el número de marcas que tienen el mismo valor de color en un conjunto color, visualmente cada valor de color se representará con un color diferente. Los nodos transición se representarán como un rectángulo que contiene grupos de pequeños cuadros que representarán el número de marcas de cada valor de color que se requieren para habilitar la transición y/o un pequeño texto con una operación lógica que representarán las guardas de la transición. Si fuese posible se desea añadir la visualización de la RPC completa en forma de grafo, donde se resaltarían los arcos que representan a las transiciones que se han disparado.

- **La Zona de descripción de los marcados**: Es la zona donde se realizará la descripción de las transiciones disparadas para cada marcado. Estas descripciones se obtienen a partir de una BBDD.

La Lista de nodos lugares y transiciones: Es una lista con todos los nodos lugares y las transiciones que contiene el modelo de RPC, siempre habrá un elemento de esta lista seleccionado que será el que se muestre en la Zona de

Dibujo. Si finalmente se incluye la visión de la RPC completa en forma de grafo, en esta lista se añadiría a principio el elemento “RPC Completo” que sería el elemento seleccionado por defecto (Visualización de toda la RPC).

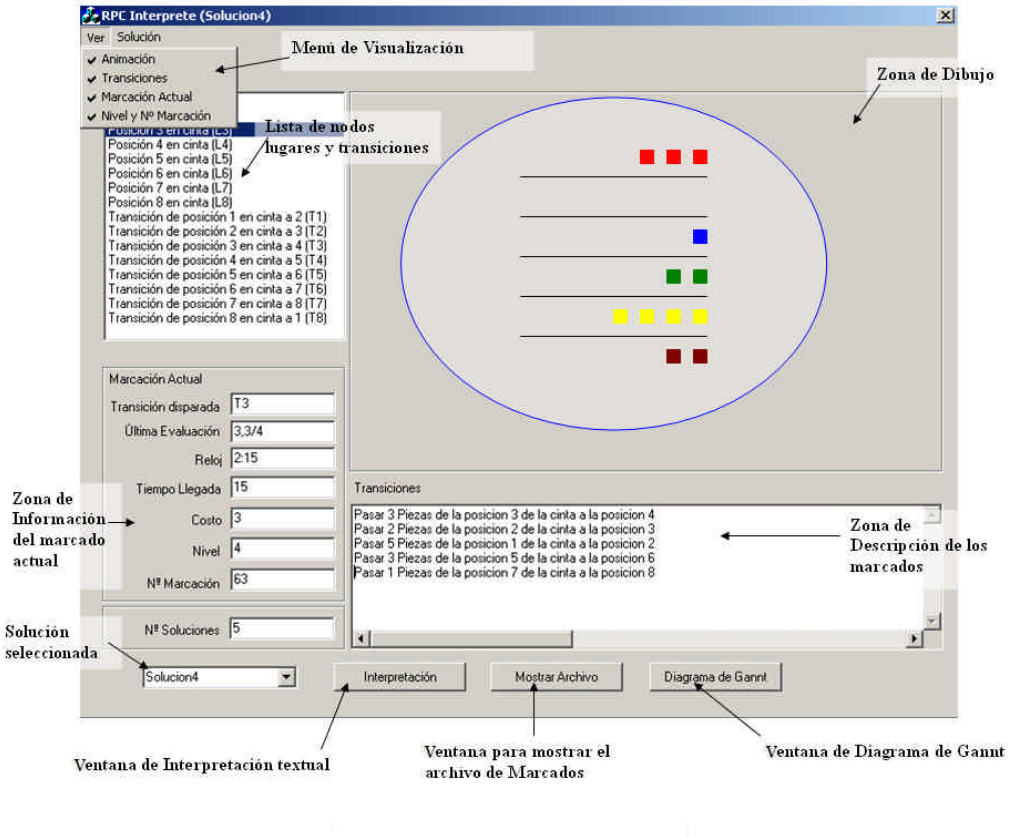

Figura 3.5: Diseño de la interfaz principal

- **La Zona de Información**: En esta zona se mostrará información del marcado actual, así como el nivel en el que encuentra la simulación y el número de nodos que se han generado para llegar al estado actual. También se mostrará, al finalizar la simulación, el número de soluciones encontradas.

- **Menú de visualización (opción ver del menú principal)**: se utilizará para seleccionar las zonas que se desean mantener activadas o desactivadas (Zona de dibujo, Zona de descripción de los marcados, Zona de Información), si se desea obtener un archivo de

texto con las operaciones realizadas por el usuario en la interfaz de simulación, y la solución que se desea visualizar.

- **Lista de Soluciones (Solución seleccionada)**: Elemento de selección de la solución que se desea interpretar.
- **Botón de Interpretación**: Botón para acceder a la interfaz de interpretación textual de la solución.
- **Botón de Diagrama de Gantt**: Botón para acceder a la interfaz que muestra un diagrama de Gantt basado en la solución obtenida.
- **Botón de Archivo de Marcados**: Botón para acceder a la interfaz que muestra el archivo de Marcados.

3.2.4 Diagramas de Clases

Una vez definidos los requerimientos del sistema, se procede a identificar las clases necesarias para el desarrollo de la interfaz de simulación, así como las relaciones existentes entre ellas. En la figura 3.6 se puede observar que se han identificado un total de 6 clases: *Marcación* (marcado), *Transición*, *Lugar*, *Conjunto Color*, *Token* (marca) y *Tiempo* que corresponden a la información sobre los elementos que componen una RPC.

En la figura 3.7 se muestran las clases correspondientes a cada una de las ventanas de las que se compone la interfaz del simulador: interfaz *RPC Interprete*, Interfaz *Interprete/Marcación* y Interfaz *Gantt*.

La herramienta utilizada para generar el diagrama de clases, así como todos los demás diagramas que describen el funcionamiento del sistema en la fase de diseño, ha sido Rational Rose. Dadas las características de esta herramienta para el análisis y diseño orientado a objetos, se pueden generar la mayor parte de los diagramas del lenguaje visual de modelado orientado a objeto UML, método escogido para el desarrollo del sistema.

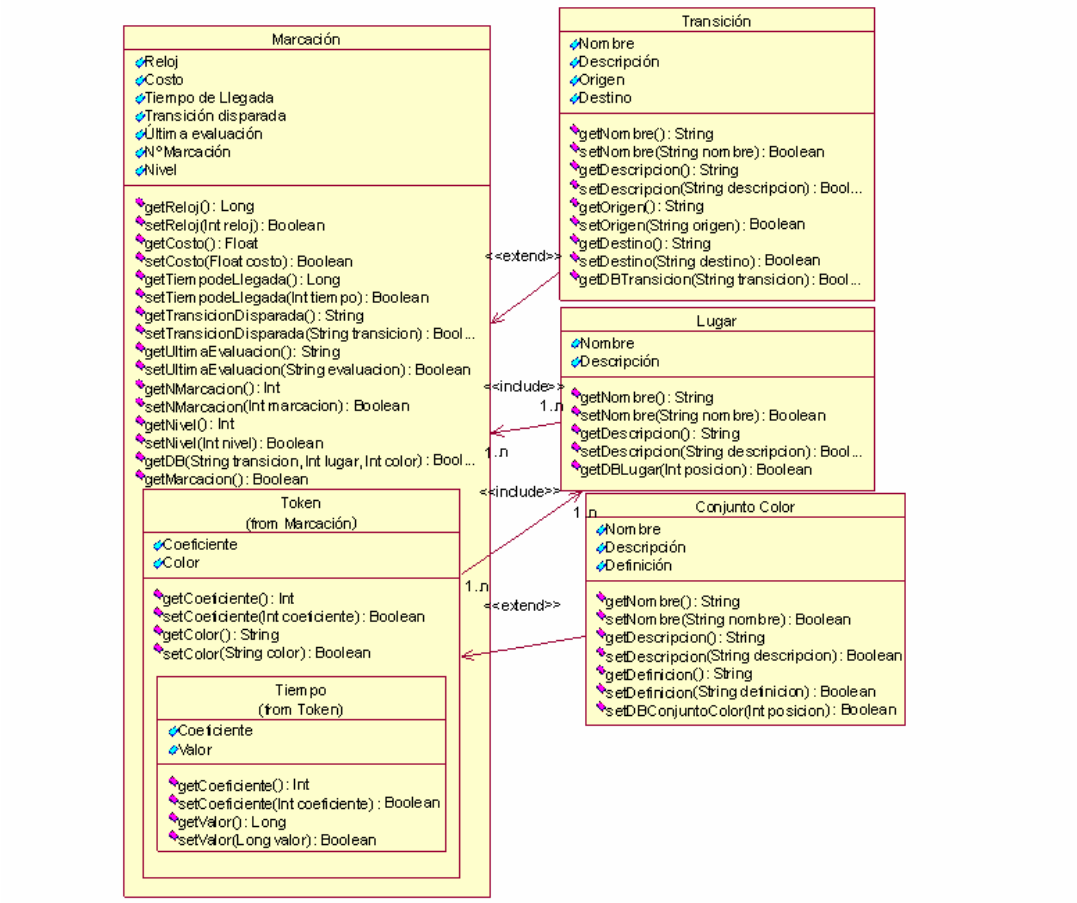

Figura 3.6: Diagrama de clases de los elementos que componen el modelo RPC

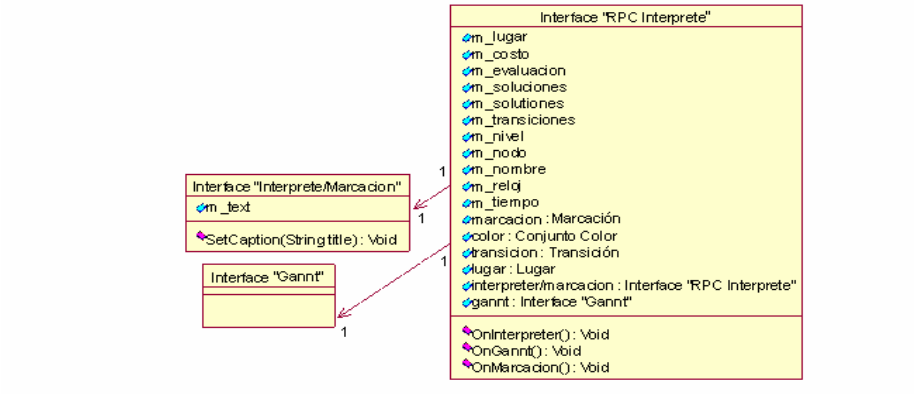

Figura 3.7: Diagrama de clases de la interfaz de simulación

Las relaciones entre las diferentes clases se han representado mediante flechas, de esta manera se puede observar que una clase Marcación se compone de las clases *Transición* y *Lugar*, además de contener la clase *Token*, quien a su vez se compone de la clase *Conjunto Color*, y contiene la clase *Tiempo*. Además de todas estas relaciones se observa también que la clase *Lugar* se compone de la clase *Token*.

En las tablas 3.1, 3.2 y 3.3 se muestran los diccionarios de datos de las clases correspondientes a los elementos que componen el modelo RPC de la interfaz de simulación, en las que se describen los atributos de cada una de las clases.

Nombre	Descripción
Marcación	Entidad principal a interpretar, representa un estado del sistema modelado, es una configuración de tokens
Reloj	Valor de tiempo antes del disparo de la **Transición** que genera la **Marcación** actual
Costo	Valor de costo de la **Marcación** actual
Tiempo de Llegada	Valor de tiempo después del disparo de la **Transición** que genera la **Marcación** actual
Transición Disparada	Nombre de la **Transición** disparada para alcanzar la **Marcación** actual
Última Evaluación	Especificación de los tokens que habilitan en el disparo de la **Transición** que genera la **Marcación** actual
Nº Marcación	Valor que representa el número de la **Marcación** actual si se está simulando el sistema modelado, o el total de marcaciones que se han producido si se interpreta una solución.
Nivel	Valor que representa el nivel de profundidad en el que se encuentra la **Marcación** actual dentro del árbol de cobertura.
Lugar	Componente de la **Marcación** que representa los diversos lugares de la RPC
Nombre	Nombre del **Lugar** para su identificación
Descripción	Descripción semántica del **Lugar** para su interpretación
Token	Componente del **Lugar** que representa las marcas en cada lugar
Coeficiente	Valor que representa el número de elementos que tienen el mismo color en un **Lugar**
Color	Valor de color del token de un **Lugar** (Tipo de **Token**)
Tiempo	Componente del **Token** que representa los tiempos de los **Tokens** que tienen el mismo color
Coeficiente	Valor que representa el número de **Tokens** que, siendo del mismo color, tienen el mismo valor de tiempo en un **Lugar**
Valor	Valor de tiempo de llegada del **Token** a la **Marcación** actual

Tabla 3.1: Diccionario de datos de la clase Marcación

Nombre	Descripción
Transición	Representa la transición disparada para generar la **marcación** actual
Nombre	Identificador de la **Transición**
Descripción	Descripción semántica de la **Transición** para su interpretación
Origen	Lugar de donde provienen los tokens que habilitan el disparo de la **Transición**
Destino	Lugar donde se colocan los tokens generados después del disparo de la **Transición**

Tabla 3.2: Diccionario de datos de la clase Transición

En el anexo A se describen en detalle los atributos y métodos que se han especificado para una de las clases.

Nombre	Descripción
Conjunto Color	Representa el color asociado a cada uno de los lugares del sistema modelado
Nombre	Nombre del **Conjunto Color** para su identificación
Descripción	Descripción del **Conjunto Color** para su interpretación
Definición	Especificación de tipo del **Conjunto Color**

Tabla 3.3: Diccionario de datos de la clase Conjunto Color

3.3 Diseño

3.3.1 Diagramas de actividad

La figura 3.8 muestra el diagrama de actividad de la interfaz principal del simulador.

Se observa que la interfaz se debe diseñar para atender dos funcionalidades diferentes, pero muy similares:

- Debe poder utilizarse para visualizar la ejecución de una simulación de una RPC: está funcionalidad tiene como objetivo principal presentar los datos de la simulación mientras ésta se ejecuta, teniendo en cuenta que la interfaz no debe afectar a la velocidad de ejecución del simulador, la

comunicación entre el simulador y la interfaz se realiza a través de un archivo que se genera en el transcurso de la simulación, al cual accede temporalmente la interfaz para actualizar los datos mostrados.

- También debe poder utilizarse para visualizar una solución de una simulación anteriormente realizada.

La figura 3.9 muestra el diagrama de actividad de la interfaz Interprete/Marcación. Como se puede observar, la finalidad de esta interfaz es la de mostrar los marcados que componen una solución en un formato que facilite su interpretación por parte del usuario.

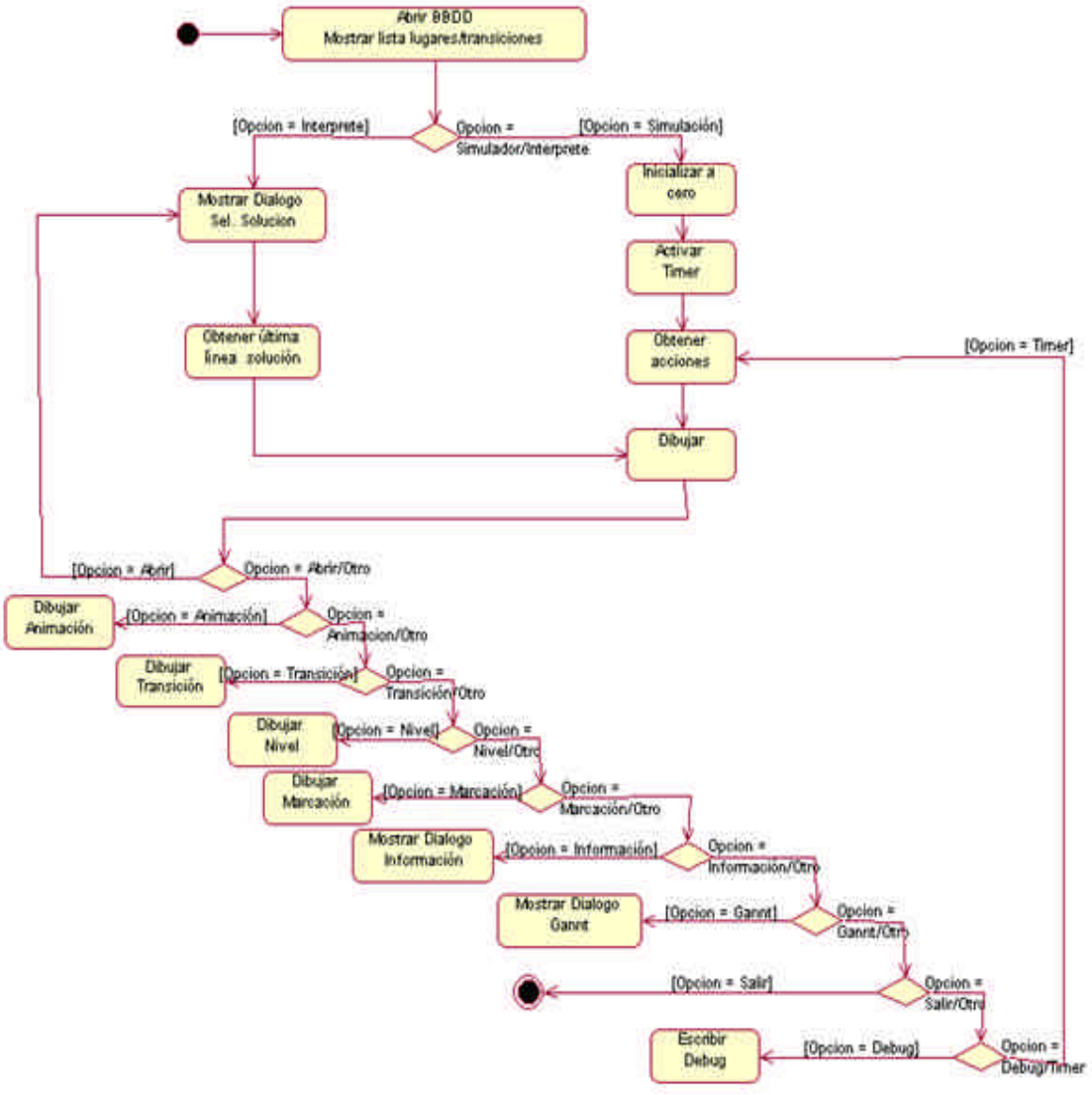

Figura 3.8: Diagrama de actividad de la interfaz principal

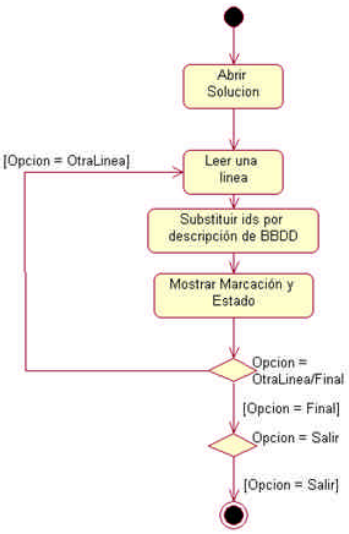

Figura 3.9: Diagrama de actividad de la interfaz Interprete/Marcación

Finalmente, en la figura 3.10 se muestra el diagrama de actividad de la interfaz que muestra el diagrama de Gantt que representa un resultado de la simulación, la cual generará un diagrama de Gantt a partir de dos atributos del RPC seleccionados por el usuario sobre una solución.

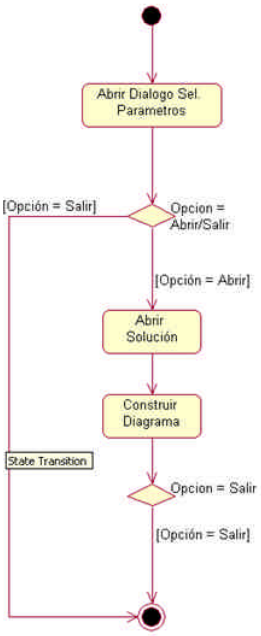

Figura 3.10: Diagrama de actividad de la interfaz Gantt

3.3.2 Diagramas de Secuencia

La figura 3.11 muestra el diagrama de secuencia del menú de visualización (menú Ver) que permitirá al usuario activar o desactivar las diferentes zonas de dibujo de la interfaz. También contiene la secuencia de eventos que permiten activar o desactivar el volcado del log a archivo.

La figura 3.12 muestra el diagrama de secuencia con los eventos ocurren cuando el usuario visualiza la interfaz del interprete de la solución.

La figura 3.13 muestra el diagrama de secuencia de los eventos que ocurren cuando el usuario selecciona una solución de una RPC que ha sido simulado. Desde que el usuario solicita visualizar una solución, hasta el momento en que el gestor encargado de la lectura de archivos finaliza la carga de este.

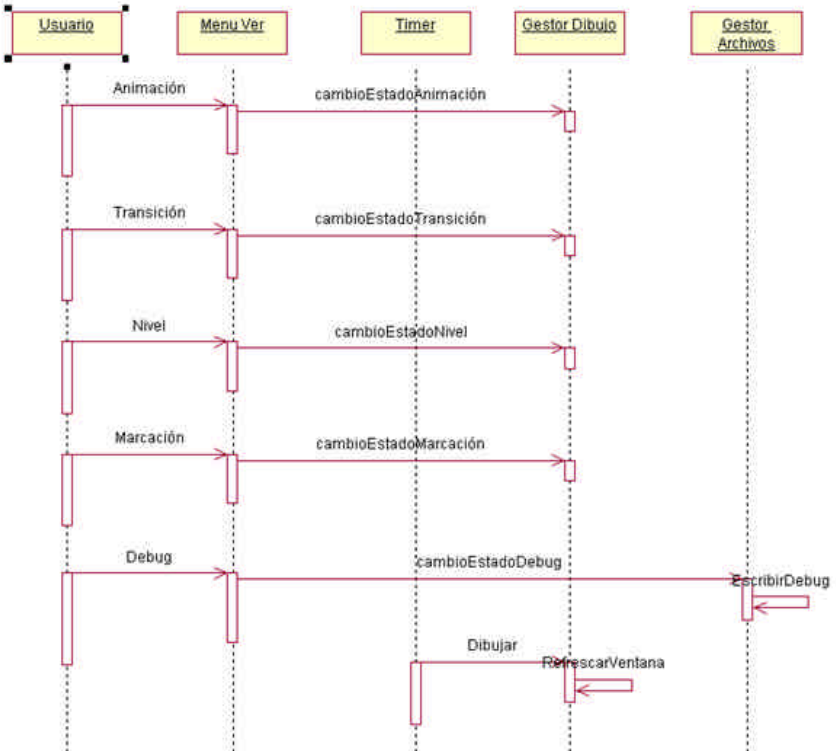

Figura 3.11: Diagrama de secuencia Menú Ver

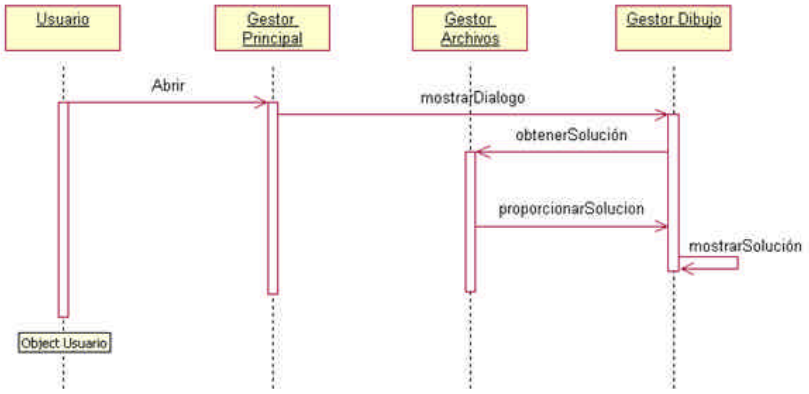

Figura 3.12: Diagrama de secuencia Interprete Solución

La figura 3.14 muestra el diagrama de secuencia con los eventos que ocurren cuando el usuario solicita la interfaz del diagrama de Gantt de la solución.

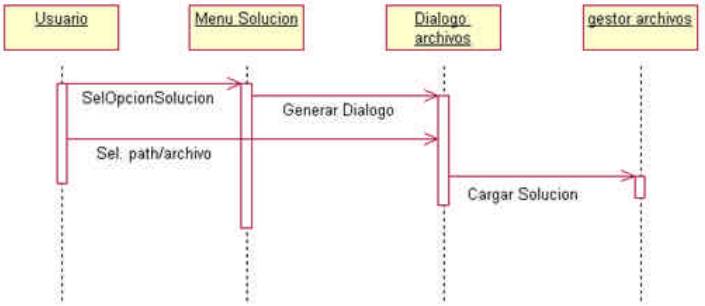

Figura 3.13: Diagrama de secuencia Selección Solución

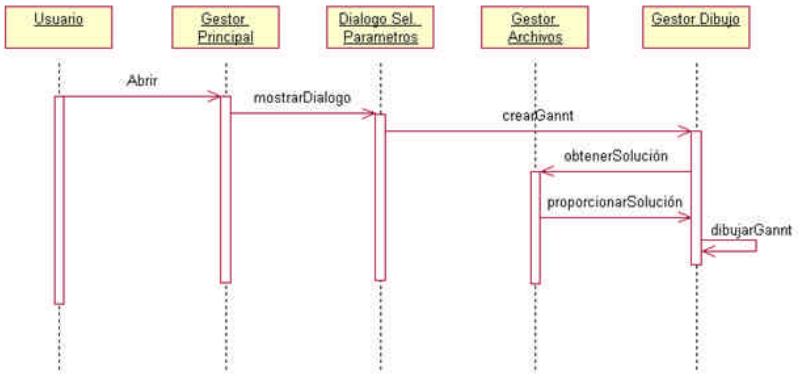

Figura 3.14: Diagrama de secuencia Gantt Solución

3.3.3 Diagramas de Colaboración

La figura 3.15 corresponde al diagrama de colaboración que se obtiene a partir del diagrama de secuencia de Menu Ver (Figura 3.11).

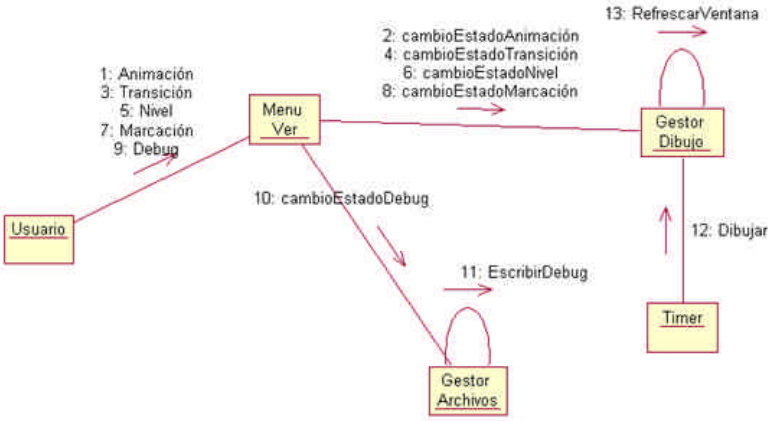

Figura 3.15: Diagrama de colaboración Menú

La figura 3.16 corresponde al diagrama de colaboración que se obtiene a partir del diagrama de secuencia de Interprete Solución (Figura 3.12).

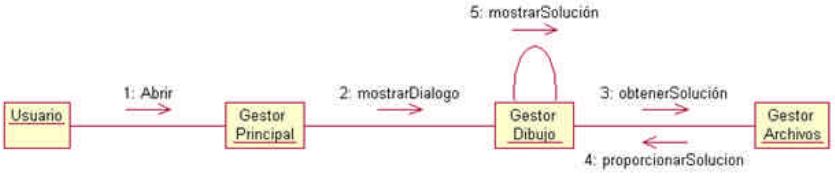

Figura 3.16: Diagrama de colaboración Interprete Solución

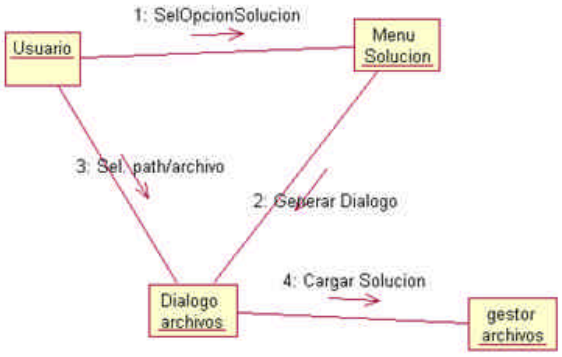

Figura 3.17: Diagrama de colaboración Selección Solución

La figura 3.17 corresponde al diagrama de colaboración que se obtiene a partir del diagrama de secuencia de Selección de Solución (Figura 3.13).

La figura 3.18 corresponde al diagrama de colaboración que se obtiene a partir del diagrama de secuencia de Gantt Solución (Figura 3.14).

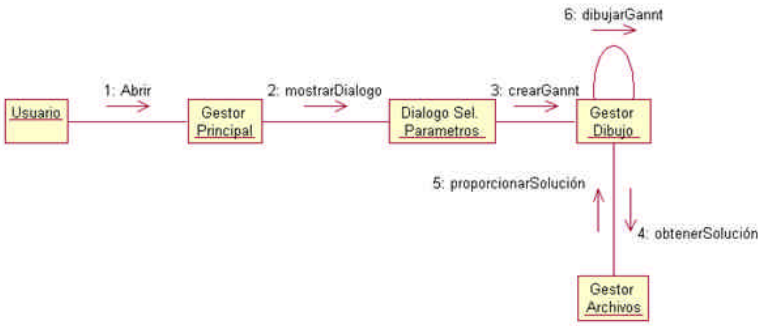

Figura 3.18: Diagrama de colaboración Gantt Solución

3.4 Implementación

La interfaz del simulador se implementada utilizando el lenguaje Microsoft Visual C++, el cual es el lenguaje en que ha sido implementado el simulador de RPCs.

Para afectar lo menos posible el rendimiento del simulador se implementa esta interfaz como un módulo aparte. La comunicación entre el simulador y la interfaz es unidireccional y se realiza a través de un archivo de texto que genera el simulador (figura 3.4), la interfaz del simulador se encarga de leer y tratar el contenido del archivo de la forma adecuada. Para su visualización, la interfaz del simulador ofrece la posibilidad de ajustar las zonas de la interfaz que se desea estén activas (figura 3.5), de esta manera se busca afectar mínimamente el rendimiento del simulador (velocidad de ejecución). En cuanto a la interpretación de los resultados, el simulador almacena el resultado de la simulación en un archivo, por lo que la interfaz solo se encarga de visualizar los datos del archivo. Para interpretar los resultados del simulador se utiliza una base de datos que contiene la descripción del modelo RPC a simular. La base

de datos ya ha sido implementada mediante el desarrollo de un proyecto complementario a este (ver figura 3.4).

Para implementar la zona de dibujo, se ha recurrido a elementos simple de dibujo, se ha hecho de esta forma para no afectar la velocidad de ejecución del simulador, el dibujo consiste en círculos y rectángulos sin relleno para los elementos y líneas para separar los elementos del conjunto color.

Para los elementos del diagrama se utilizan rectángulos que contienen los parámetros seleccionados para visualizar. Se han incluido en el panel botones con funcionalidad de scroll para poder visualizar diagramas grandes.

CAPÍTULO 4

MANUAL DE USUARIO

En este capítulo se explica el funcionamiento de la interfaz desarrollada. Se mostrarán las diferentes opciones de la interfaz, así como una descripción detallada y genérica de su uso.

4.1 Interfaz Principal

En la figura 4.1 se observa la interfaz que permite mostrar la ejecución de una simulación de una RPC. Esta interfaz se divide en seis partes:

- **En la lista de nodos lugares y transiciones** se muestra una lista de nodos lugares y nodos transiciones que componen el modelo de la RPC que se está simulando. En la lista se muestra la descripción de cada uno de los nodos.

- **En la zona de dibujo** se muestran una representación gráfica del nodo que se ha seleccionado en la lista anterior. Si se ha seleccionado un nodo lugar, se representa como un óvalo, y en su interior se dibujan pequeños cuadros de colores para representar los valores de color del nodo lugar (ver figura 4.1), estos se agrupan horizontalmente por color, y verticalmente por el conjunto color al cual pertenecen, el número que se representa encima de cada línea vertical es la cantidad de tokens disponibles, y el número representado al lado de cada cuadro es su identificación. Si se ha seleccionado un nodo transición, se representa como un rectángulo (ver figura 4.2), y en su interior se representan los guardas, los cuales pueden ser representados como funciones lógicas o como los tokens que se requieren para el disparo de la transición.

- **En la zona de información** se muestra la información del proceso de simulación, que se compone de la transición disparada, las últimas marcas evaluadas, el reloj de simulación, el tiempo de llegada al estado más reciente generado, el costo, el nivel y el número de identificación del marcado.

- **En la zona de descripción de los marcados** se muestra una lista de las transiciones que se han disparado.

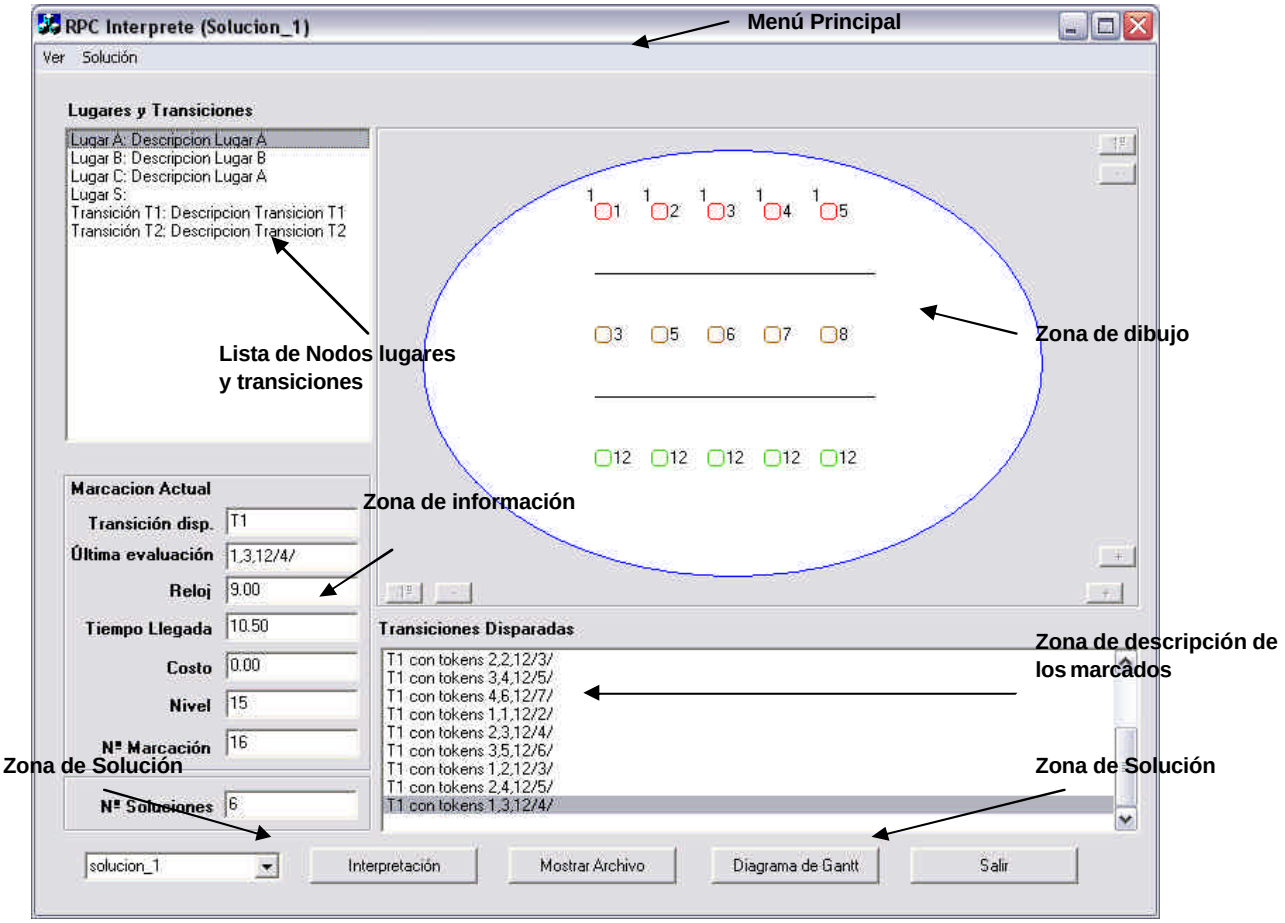

Figura 4.1: Interfaz principal con nodo Lugar seleccionado

- **En el menú principal** se muestra el submenú de visualización (opción *Ver*) que permite seleccionar que elementos se han de visualizar durante la simulación del modelo.

- **En la zona de Soluciones** existen funciones específicas para trabajar con archivos de soluciones, accesibles desde el submenú de selección de solución, así como los botones que nos permiten acceder a las ventanas de interpretación del resultado.

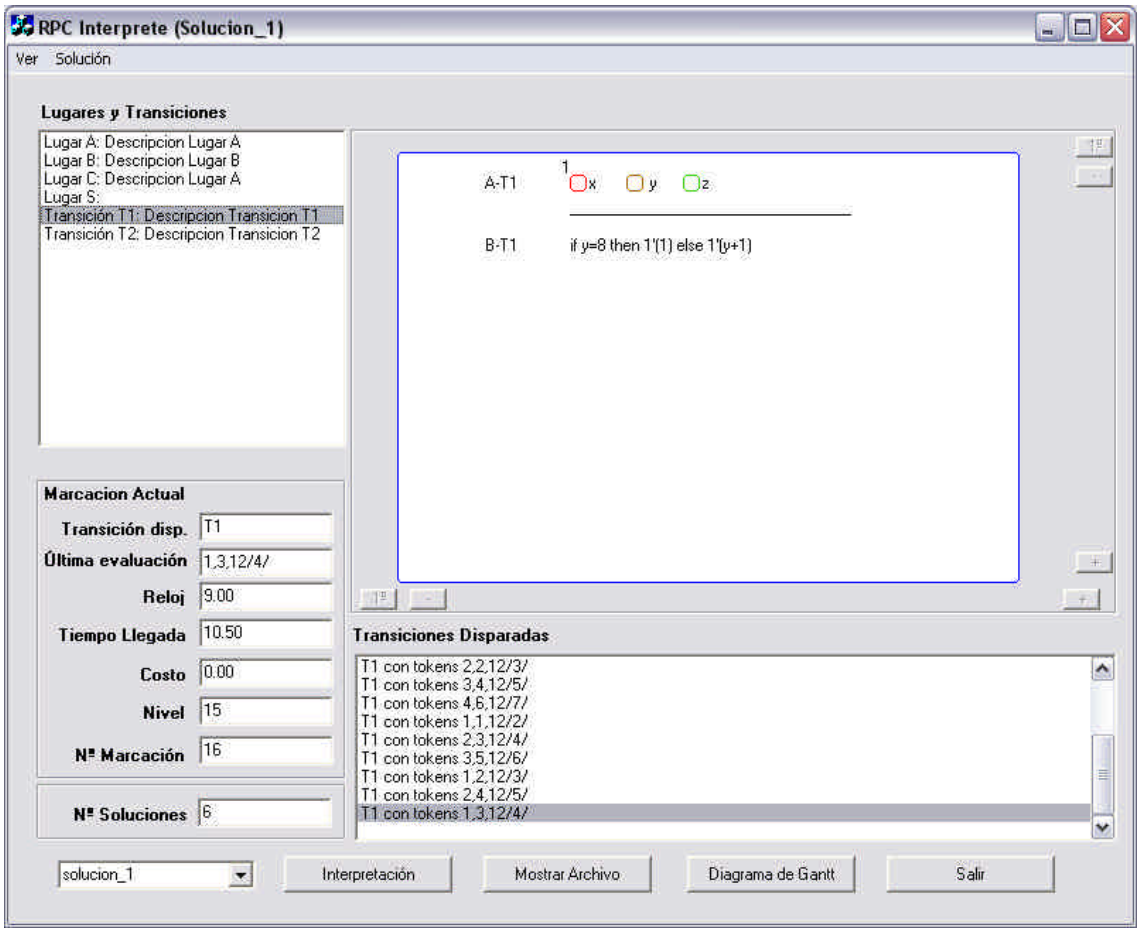

Figura 4.2: Interfaz principal con nodo Transición seleccionado

4.2 Visualización de resultados

La visualización de los resultados puede iniciarse de dos maneras:

1. Si la herramienta se inicia porque se está simulando una RPC, aparecerá la interfaz principal (figura 4.1), utilizando la base de datos del simulador de RPC, y la información que este genera.

2. Cuando un usuario inicializa la herramienta para visualizar los resultados de una simulación anterior, aparece la interfaz principal (figura 4.1), desde el menú principal, con el submenú *Solución* se abre la ventana de dialogo de la figura 4.3, para seleccionar la base de datos de la RPC que contiene la descripción del modelo de la RPC de la simulación. Una vez se ha seleccionado la base de datos, aparece la ventana que se muestra en la figura 4.4, para seleccionar el archivo con la solución que se desea estudiar. Los archivos solución se identifican como "solucion_x.txt" donde x es un número entero que indica el número de la solución encontrada.

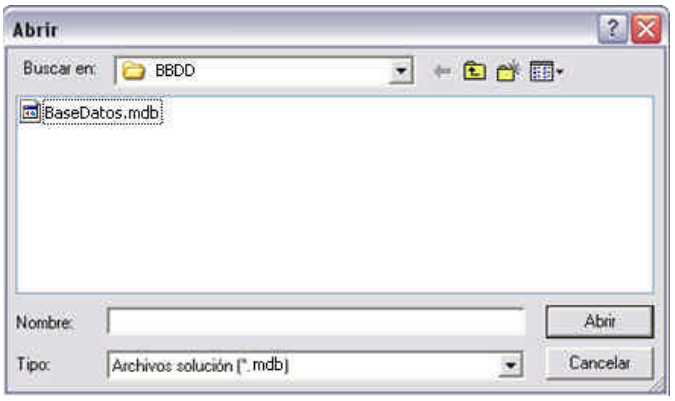

Figura 4.3: Selección de la base de datos

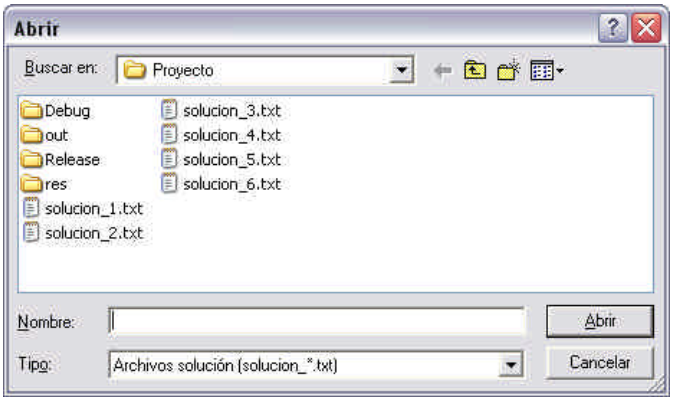

Figura 4.4: Selección de la solución

4.3 Menú Ver

En la figura 4.5 se muestran las opciones disponibles en el menú *Ver*, compuesto por *Animación*, *Transiciones Disparadas*, *Marcación Actual*, *Nivel y Nº Marcación*.

Las funcionalidades son las siguientes:

- *Animación*: activa y desactiva la representación gráfica de los lugares y de las transiciones. En la zona superior derecha (zona de dibujo).
- *Transiciones Disparadas*: activa y desactiva la lista de transiciones que han sido disparadas. En la zona inferior derecha (zona de descripción de los marcados).
- *Marcación Actual*: activa y desactiva los cuadros de texto *transición disparada*, *última evaluación*, *reloj*, *tiempo de llegada*, y *costo*. En la zona inferior izquierda (zona de información).
- *Nivel y Nº Marcación:* activa y desactiva los cuadros de texto *nivel* y *Nº marcación*. En la zona inferior izquierda (zona de información).

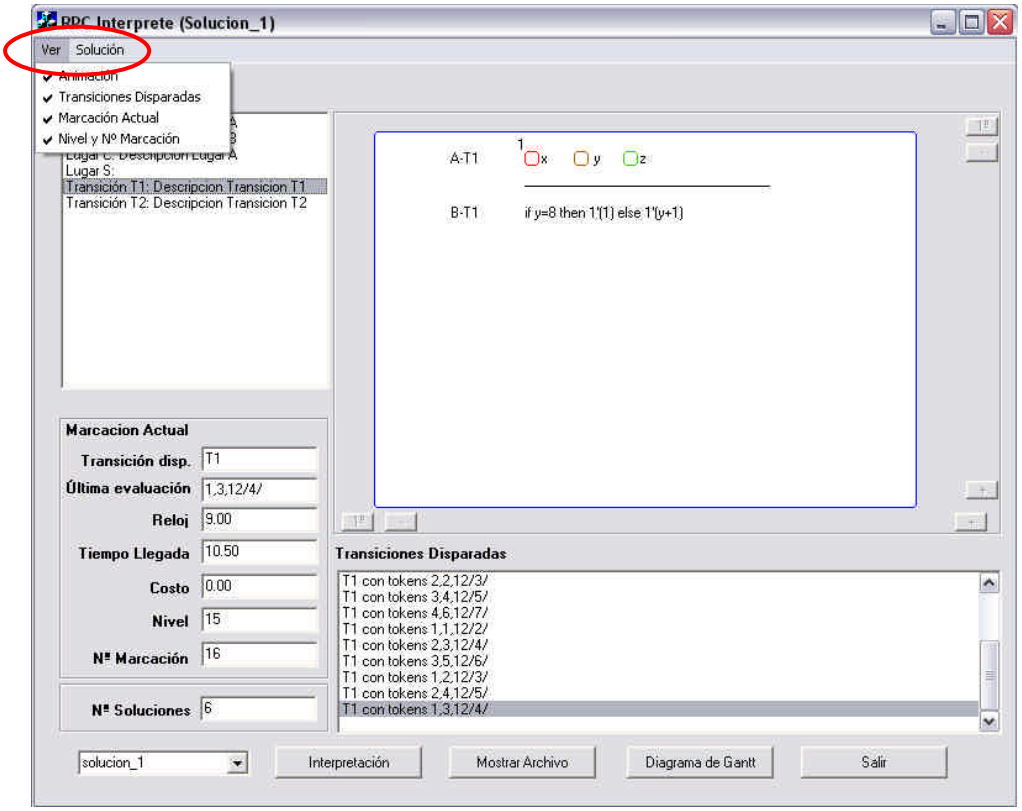

Figura 4.5: Interfaz principal, menú *Ver*

4.4 Menú Solución

A través de la opción *Solución* del menú principal que se muestra en la figura 4.5, se accede a los diálogos de las figuras 4.3 y 4.4, dando la opción de esta manera de seleccionar una base de datos de otra RPC, para poder estudiar las soluciones previas que se obtuvieron del simulador de RPC.

4.5 Seleccionar Solución

En la figura 4.6 se muestra una lista (esquina inferior izquierda) que permite seleccionar entre las diferentes soluciones obtenidas por el simulador para la misma RPC que se estaba estudiando.

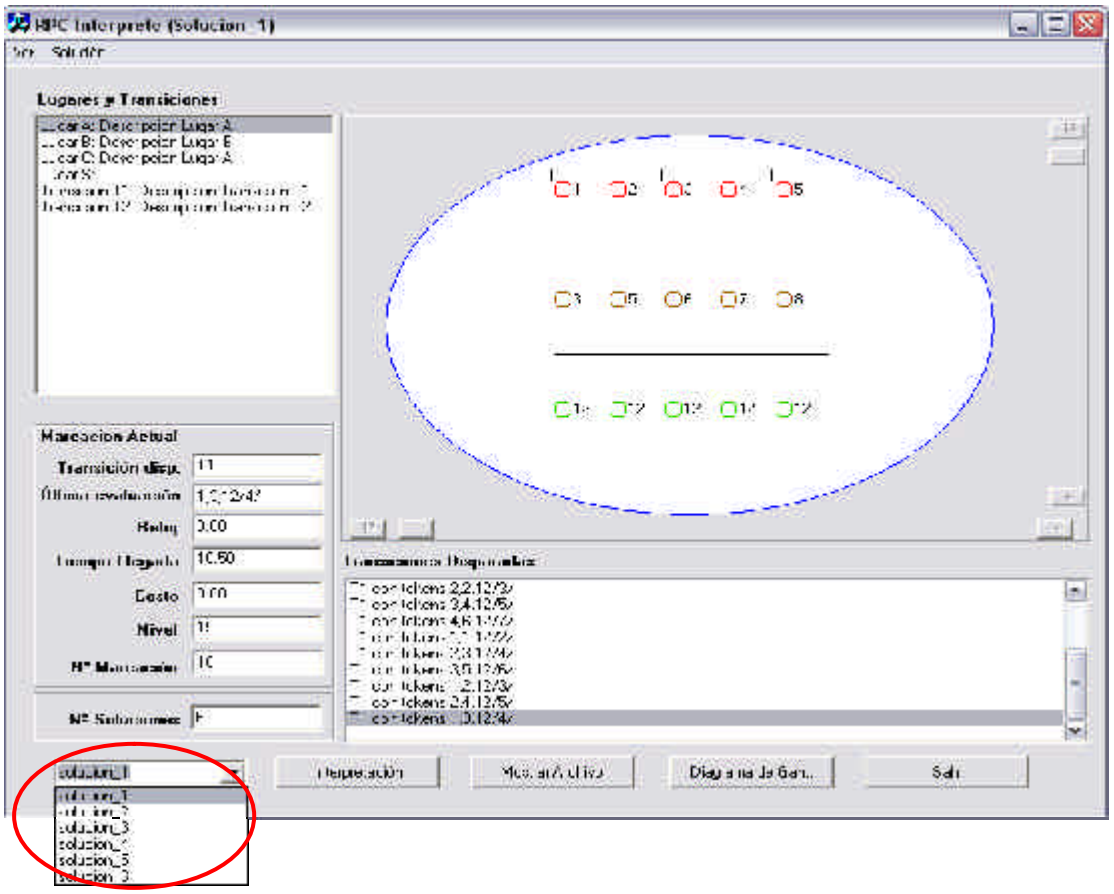

Figura 4.6: Interfaz principal, selección de solución

4.6 Mostrar Archivo

Con una solución seleccionada al pulsar el botón “Mostrar Archivo” de la ventana de la figura 4.6, se despliega una ventana, figura 4.7, en la que se puede visualizar el archivo de solución que genera el simulador.

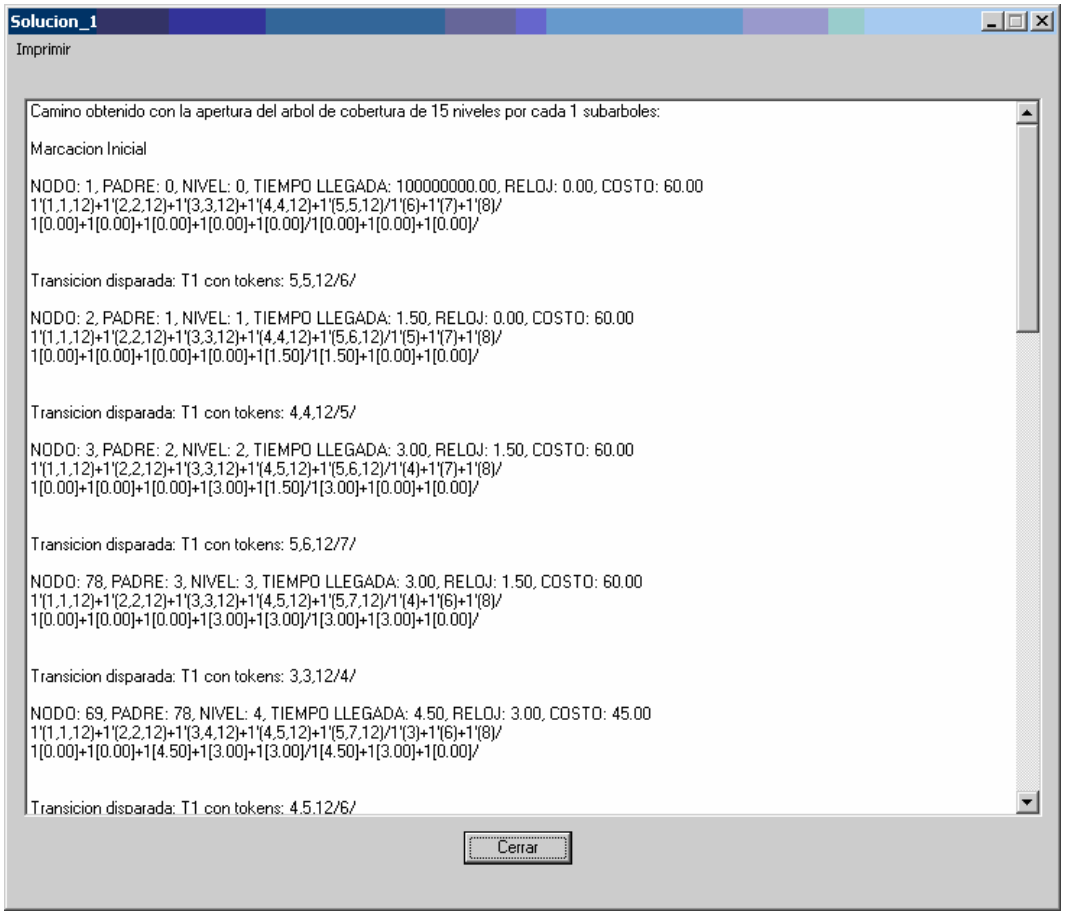

Figura 4.7: Interfaz del archivo solución

4.7 Interpretación

Cuando se ha seleccionado una solución, y si se pulsa el botón “Interpretación” de la ventana de la figura 4.6, se accede a una ventana, figura 4.8, donde se muestra la interpretación del archivo de solución con las descripciones de los nodos lugares, nodos transiciones, conjuntos color y marcas que describen cada marcado. En la figura se han desplegado descripciones genéricas, sin

embargo, para un modelo RPC, las descripciones corresponderán a la semántica que describe cada uno de los elementos del modelo.

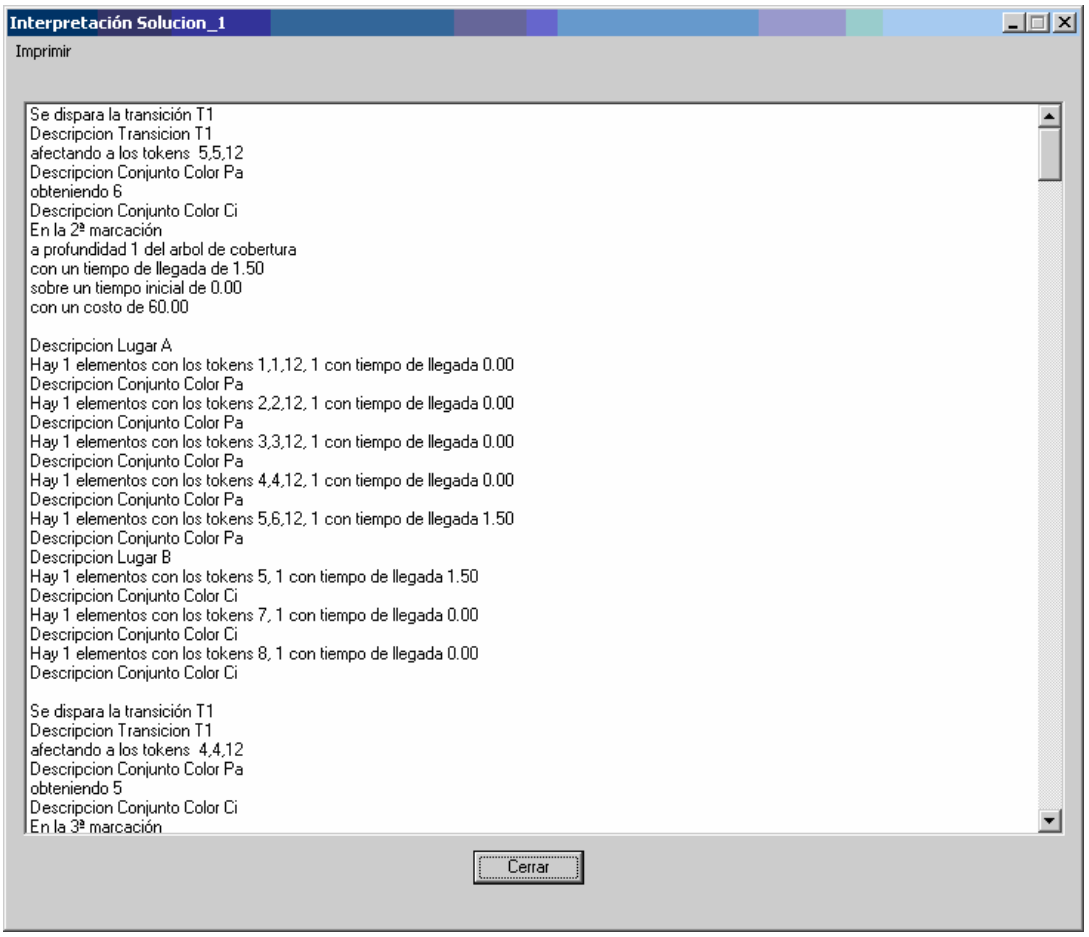

Figura 4.8: Interfaz con la interpretación de la solución

4.8 Diagrama de Gantt

Con una solución seleccionada, si se pulsa el botón “Diagrama de Gantt”, se accede a una ventana de selección, figura 4.9.

En esta ventana se seleccionan la coordenada Y para el diagrama de Gantt a partir de los conjuntos color representados en el modelo de RPC que se está visualizando. Una vez seleccionada la coordenada Y, hay que seleccionar los datos que se desean mostrar en el diagrama, las opciones que se muestran son los nodos transición del modelo de RPC, individualmente o todos a la vez, y, si existe algún caso, los conjuntos color de los que se componen del

conjunto color seleccionado para la coordenada Y. Además se puede modificar el intervalo de tiempo que se muestra en cada página en las que se puede dividir el diagrama. En la figura 4.10 se muestra la ventana del diagrama de Gantt.

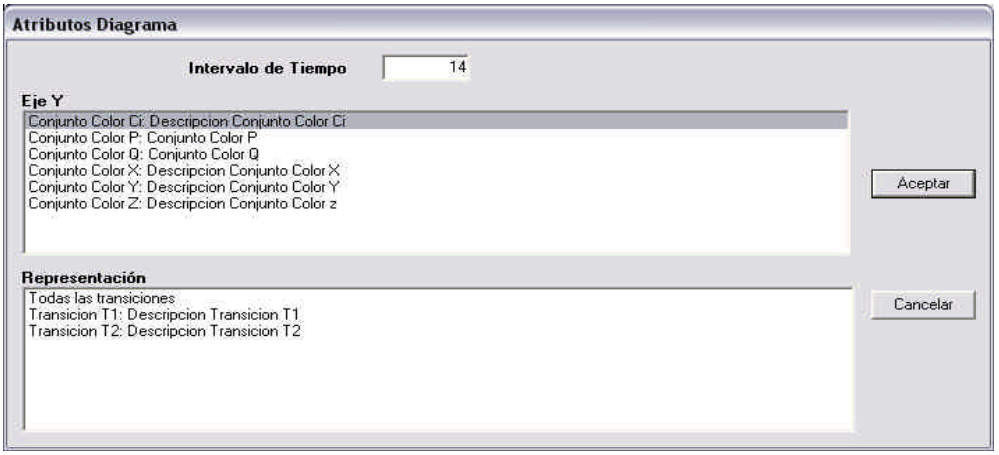

Figura 4.9: Interfaz de selección de atributos

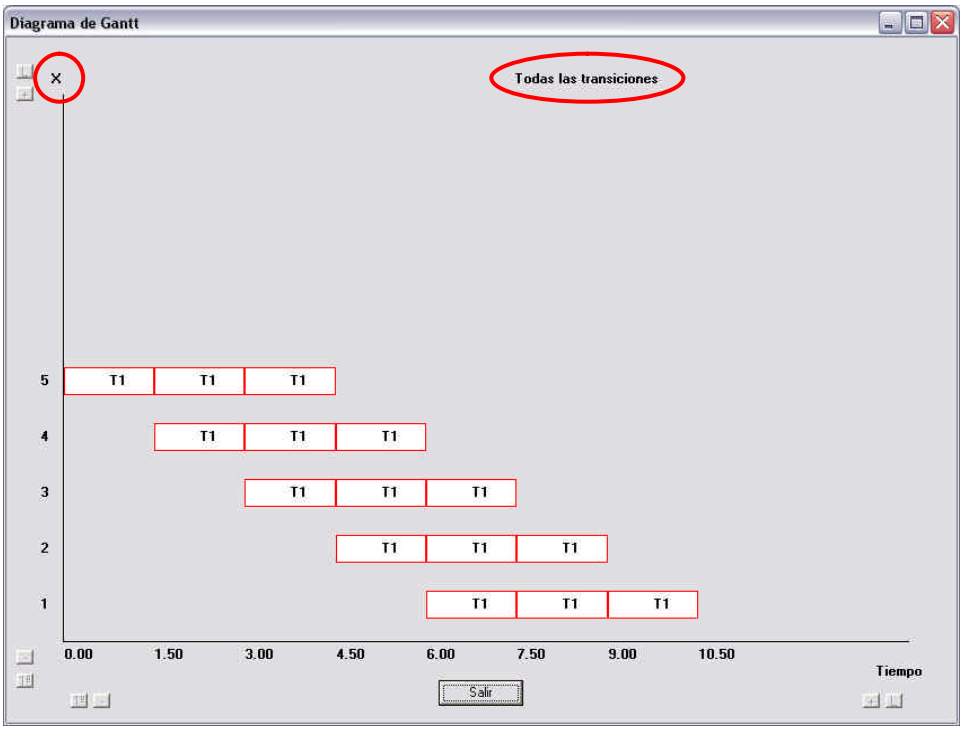

Figura 4.10: Interfaz con el diagrama de Gantt de la solución

En el diagrama de Gantt se puede ver la evolución en el tiempo de uno de los conjuntos color del modelo respecto a todas las transiciones, una transición en concreto o uno de los conjuntos color que se componen por el conjunto color seleccionado, si existiese el caso.

En la figura 4.10 se muestra el diagrama de Gantt de todas las transiciones ocurridas en la solución seleccionada respecto al conjunto color *X*. Se puede observar como inicialmente, en el instante 0, se produce el disparo de la transición T_1 que afecta al token *5* con información representada por el Conjunto Color *X*. El tiempo de llegada es en el instante *1.5*, momento en que se dispara la transición T_1 nuevamente, esta vez sobre el token *4* y simultáneamente sobre el token *5*.

CAPÍTULO 5

EJEMPLO DE APLICACIÓN

En este capítulo se muestra un ejemplo de la ejecución de la interfaz de simulación integrada en el entorno de simulación de RPC.

5.1 Descripción del Problema

Para realizar el ejemplo de la ejecución de la interfaz de simulación, se ha seleccionado un problema de planificación de la producción para un sistema de manufactura multi-etapas[1]. Este tipo de problema consiste en la organización de la producción en sistemas de fabricación de familias de productos a partir de ciertos productos intermedios comunes, a través de varias etapas de procesamiento y ensamblaje.

En la figura 5.1 se muestra un grafo que describe los trabajos asociados a la fabricación de los productos finales, denominados *D* y *E*.

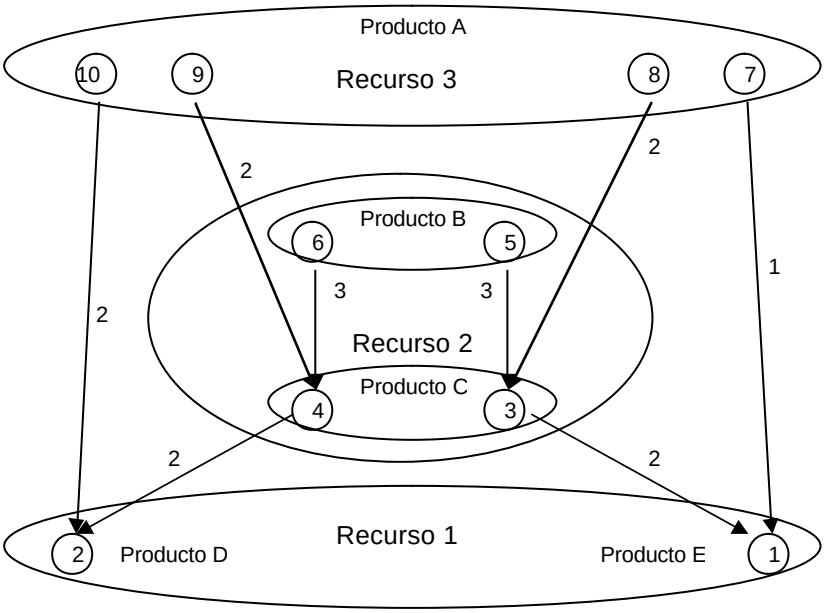

Figura 5.1: Grafo de una estructura de producción multi-etapas

[1] Las figuras y tablas de este ejemplo han sido tomadas de [5].

En el grafo se observa que existen dos trabajos, la fabricación del producto final *D*, el cual se compone de las tareas *2*, *4*, *6*, *9*, y *10*, y la fabricación del producto final *E*, que se compone de las tareas *1*, *3*, *5*, *7*, y *8*. Los arcos representan las restricciones de precedencia de las tareas, y los nodos representan las restricciones de los recursos compartidos: El recurso *1* se utiliza para fabricar los productos finales *D* y *E*, el recurso *2* se utiliza para fabricar el producto primario *B* y el intermedio *C*, y el recurso *3* se utiliza para fabricar el producto primario *A*. Los pesos de los arcos representan la cantidad de productos primarios e intermedios que se necesitan para fabricar una unidad de producto final *D* y una unidad de producto final *E* respectivamente.

El orden de precedencia en el que se deben realizar las tareas es el siguiente:

- Para fabricar el producto *D*: se realiza la tarea *2*, la cual necesita 2 unidades de producto intermedio *C* (tarea *4*) y 2 unidades de producto primario *A* (tarea *10*). El producto intermedio *C* se consigue con 3 unidades de producto primario *B* (tarea *6*) y 2 unidades de producto primario *A* (tarea *9*).

- Para fabricar el producto *E*: se realiza la tarea *1*, la cual necesita 2 unidades de producto intermedio *C* (tarea *3*) y 1 unidad de producto primario *A* (tarea *7*). El producto intermedio *C* se consigue con 3 unidades de producto primario *B* (tarea *5*) y 2 unidades de producto primario *A* (tarea *8*).

Tarea	Recurso	Producto	Tiempo
10	3	A (Primario)	10
9	3	A (Primario)	10
8	3	A (Primario)	10
7	3	A (Primario)	10
6	2	B (Primario)	5
5	2	B (Primario)	5
4	2	C (Intermedio)	50
3	2	C (Intermedio)	80
2	1	D (Final)	100
1	1	E (Final)	80

Tabla 5.1: Tiempos de procesamiento

Los tiempos de procesamiento para la fabricación de una unidad de cada tipo de producto se muestran en la tabla 5.1

5.2 Modelado del Problema

Las tablas 5.2, 5.3 y 5.4 describen el modelado del problema para poder representarlo posteriormente en la interfaz de simulación del entorno de simulación de RPCs.

Transición	Descripción
T1	Fabricación en lote de producto tipo A
T2	Fabricación en lote de producto tipo B
T3	Fabricación en lote de producto tipo C
T4	Fabricación en lote de producto final tipos D y E

Tabla 5.2: Descripción de los nodos transición del problema

Color	Definición	Descripción
MP	Int 8,10,20,30,32,48	Cantidad de producto primarios (materia prima) necesaria para fabricar 5 unidades de producto final D y 8 unidades de producto final E.
TA	Int 7..10	Identificador de la tarea (7,8,9 ó 10) realizada para fabricar el producto A.
FA	Int 0..1	Indica si el producto A ha sido fabricado mediante la tarea cuyo valor corresponde al identificador de tarea TA (0: no ha sido fabricado por la tarea TA, 1: ha sido fabricado por la tarea TA).
TB	Int 5,6	Identificador de la tarea (5 ó 6) realizada para fabricar el producto B.
FB	Int 0..1	Indica si el producto B ha sido fabricado mediante la tarea TB (0: no ha sido fabricado por la tarea TB, 1: ha sido fabricado por la tarea TB).
TC	Int 3,4	Identificador de la tarea (3 ó 4) realizada para fabricar el producto C.
FC	Int 0..1	Indica si el producto C ha sido fabricado mediante la tarea TC (0: no ha sido fabricado por la tarea TC, 1: ha sido fabricado por la tarea TC).
TF	Int 1,2	Identificador de la tarea (1 ó 2) realizada para fabricar el producto final.
FF	Int 0..1	Indica si el producto final ha sido fabricado mediante la tarea TF (0: no ha sido fabricado por la tarea TF, 1: ha sido fabricado por la tarea TF).
Pa	Product TA*FA	Información sobre producto A.
Pb	Product TB*FB	Información sobre producto B.
Pc	Product TC*FC	Información sobre producto C.
Pf	Product TF*FF	Información sobre producto final.
Ma	Int 1..3	Identificador de recurso (máquinas).

Tabla 5.3: Descripción de los conjuntos color del problema

Nodo Lugar	Color	Descripción
P	Mp	Respresenta la información asociada a la cantidad de materia prima requerida para fabricar el producto final.
A	Pa	Representa la cantidad asociada al producto A: identificación de la tarea mediante la cual se fabrica (7..10) y el producto fabricado (1) o no fabricado (0).
B	Pb	Representa la cantidad asociada al producto B: identificación de la tarea mediante la cual se fabrica (5,6) y el producto fabricado (1) o no fabricado (0).
C	Pc	Representa la cantidad asociada al producto C: identificación de la tarea mediante la cual se fabrica (3,4) y el producto fabricado (1) o no fabricado (0).
F	Pf	Representa la cantidad asociada al producto final: identificación de la tarea mediante la cual se fabrica (1,2) y el producto fabricado (1) o no fabricado (0).
M	Ma	Representa la información asociada a la máquina o recurso de procesamiento: recurso 1 (1), recurso 2 (2), recurso 3 (3).

Tabla 5.4: Descripción de los nodos lugar del problema

Las figuras 5.2, 5.3, 5.4 y 5.5 muestran los subsistemas que componen el modelo del sistema de producción, cada uno asociado a la fabricación de un tipo de producto (ver tabla 5.2).

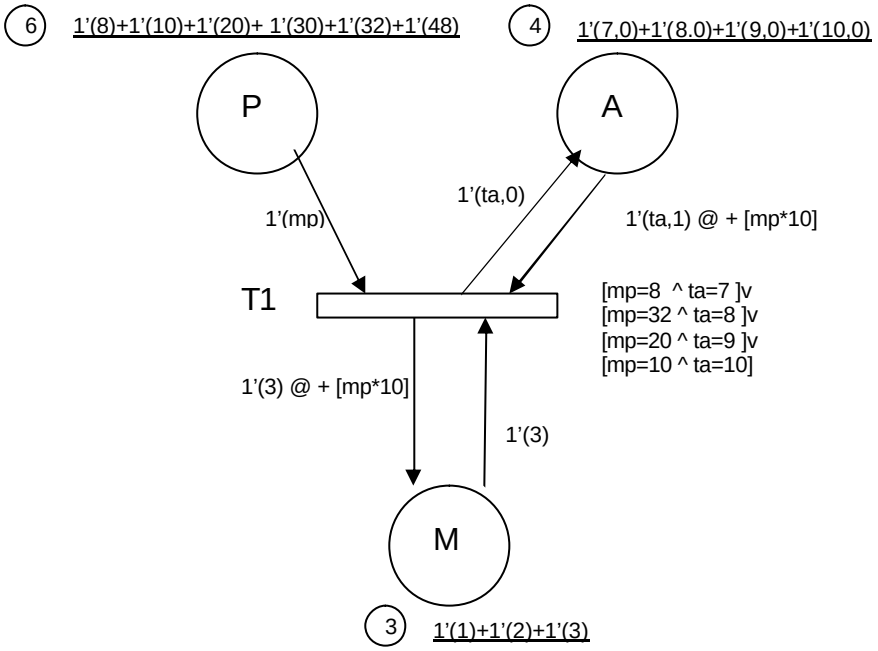

Figura 5.2: Subsistema de fabricación del producto A

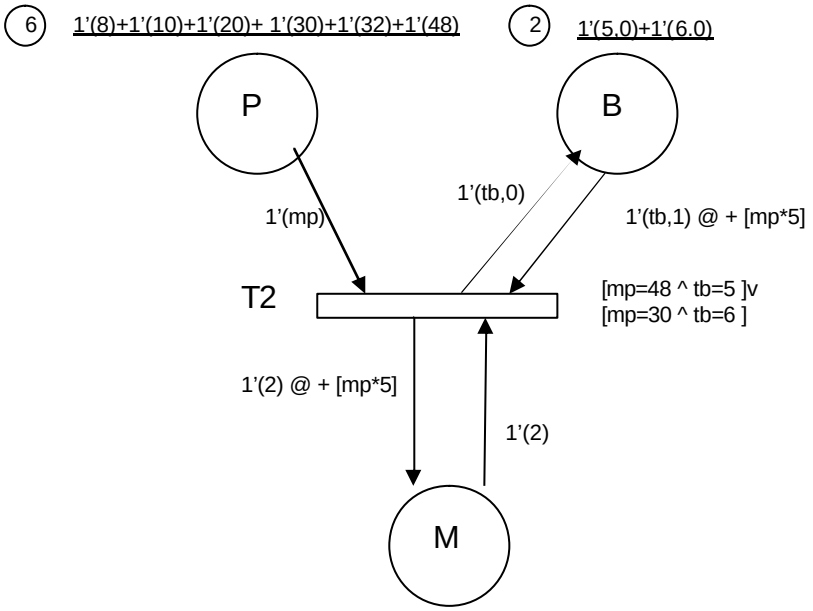

Figura 5.3: Subsistema de fabricación del producto B

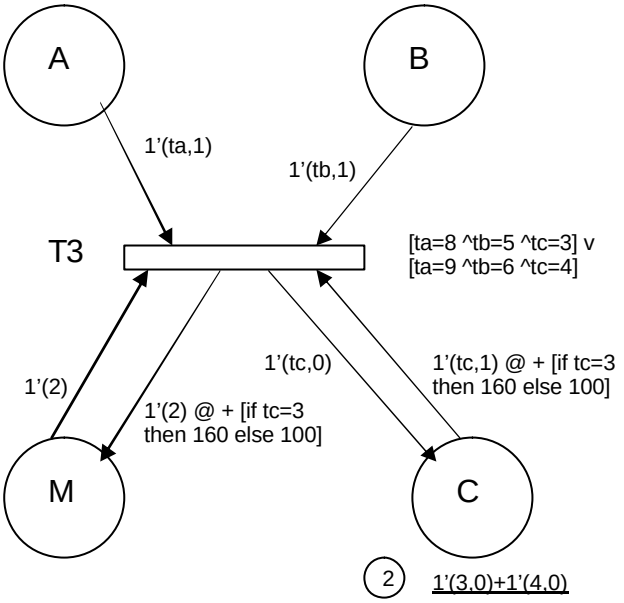

Figura 5.4: Subsistema de fabricación del producto C

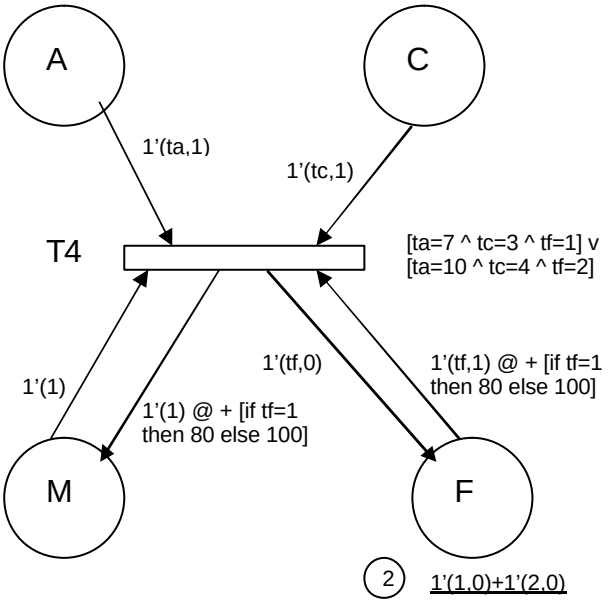

Figura 5.5: Subsistema de fabricación del producto final

5.3 Simulación del Problema

Para iniciar la simulación del problema se necesita crear la BBDD del modelo de la RPC, para esto se utiliza el proyecto complementario denominado *Interfaz de Captura de Datos* (proyecto 1 de la figura 3.4). Una vez se dispone de la BBDD del modelo, se inicia el simulador y la interfaz de simulación desde la que se puede cargar la BBDD.

Al inicio, la interfaz de simulación solo dispone de la BBDD del modelo RPC, y por tanto solo se puede visualizar la lista de nodos lugar y nodos transición que componen el modelo, junto con su descripción (en la figura 5.6 se puede ver la interfaz de simulación con el modelo de RPC utilizado para el ejemplo de ejecución). A medida que el simulador genera y almacena en un archivo el árbol de alcance del modelo, la interfaz de simulación lo comprueba periódicamente dicho archivo para mostrar al usuario el estado actual del proceso de simulación, hasta que esta finalice.

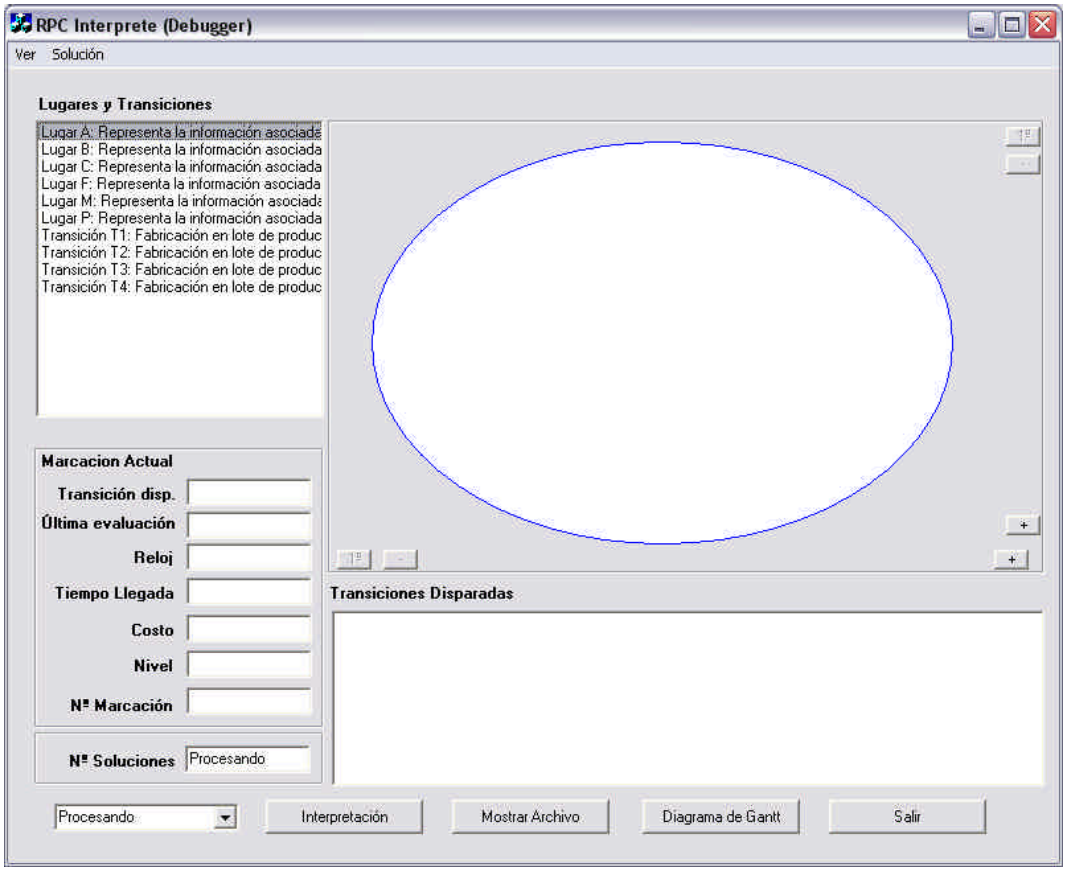

Figura 5.6: Interfaz de simulación del problema

En la figura 5.7 se puede observar que en el instante de tiempo 740 (*Reloj*) se ha disparado la transición *T2*, afectando a los lugares *B*, *M* y *P* con los tokens (6,0), (30) y (2) respectivamente. El tiempo de llegada al nuevo estado (*Tiempo Llegada*) es 890 y el costo (*Costo*) de alcanzar el estado es 0. También podemos observar que la información representada corresponde a la 7ª marcación generada (*Nº Marcación*) y que está a profundidad 6 del árbol de alcance (*Nivel*).

Cuando se dispone de las soluciones que se producen al finalizar el proceso de simulación, se puede utilizar la interfaz de simulación para interpretarlas. En el problema que se está presentando, el simulador genera dos posibles soluciones (solucion_1 en la figura 5.8 y solucion_2 en la figura 5.9).

En la figura 5.10, se muestra la interfaz a la que se accede al pulsar el botón *Mostrar Archivo,* en esta interfaz se puede ver el archivo de la primera solución del problema que genera el simulador de RPC.

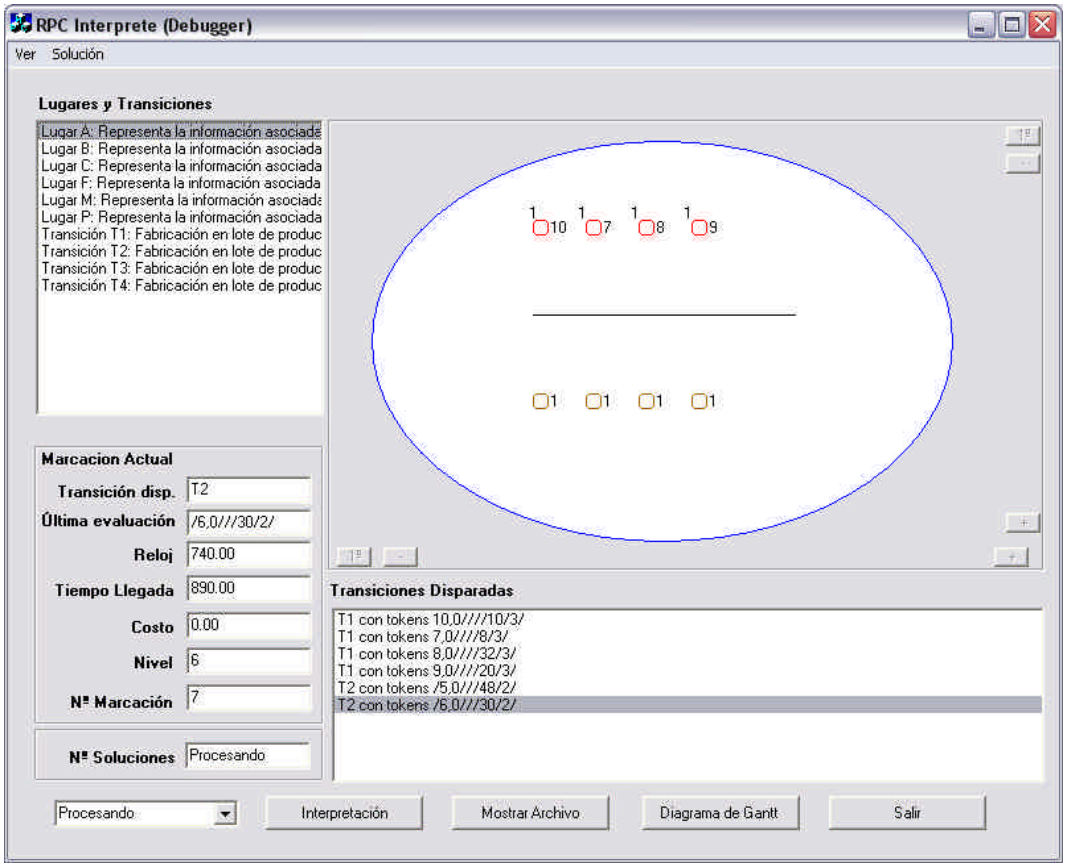

Figura 5.7: Simulación del problema

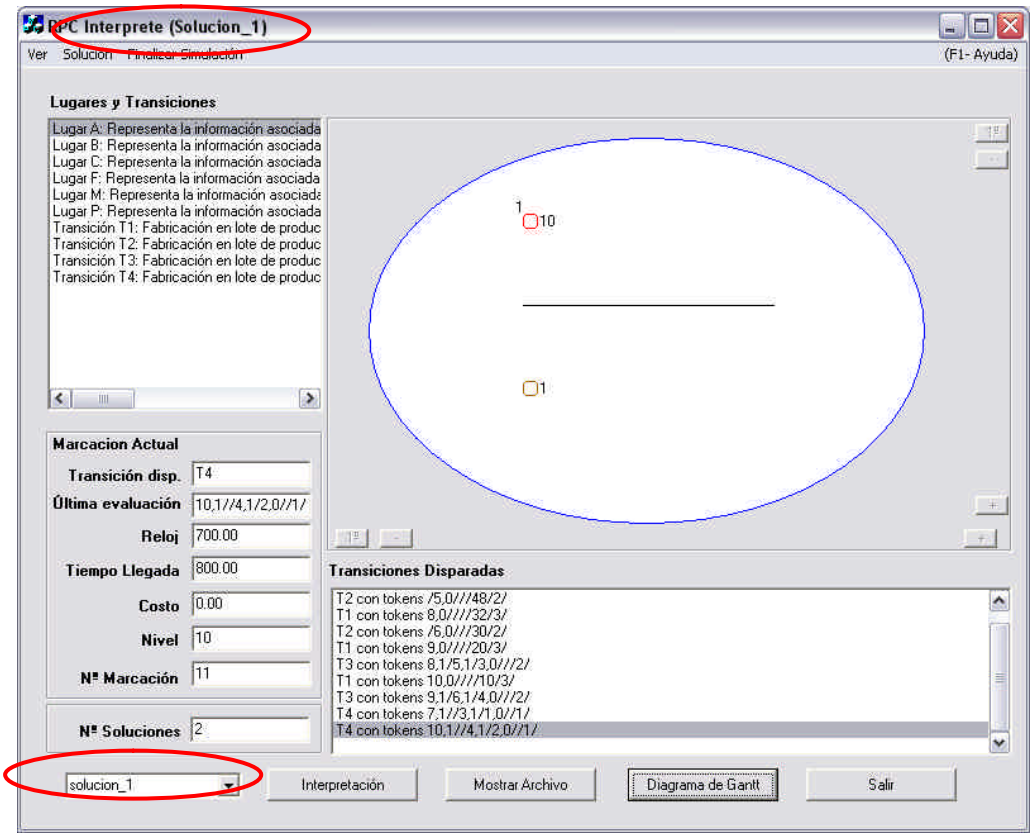

Figura 5.8: Primera solución del problema

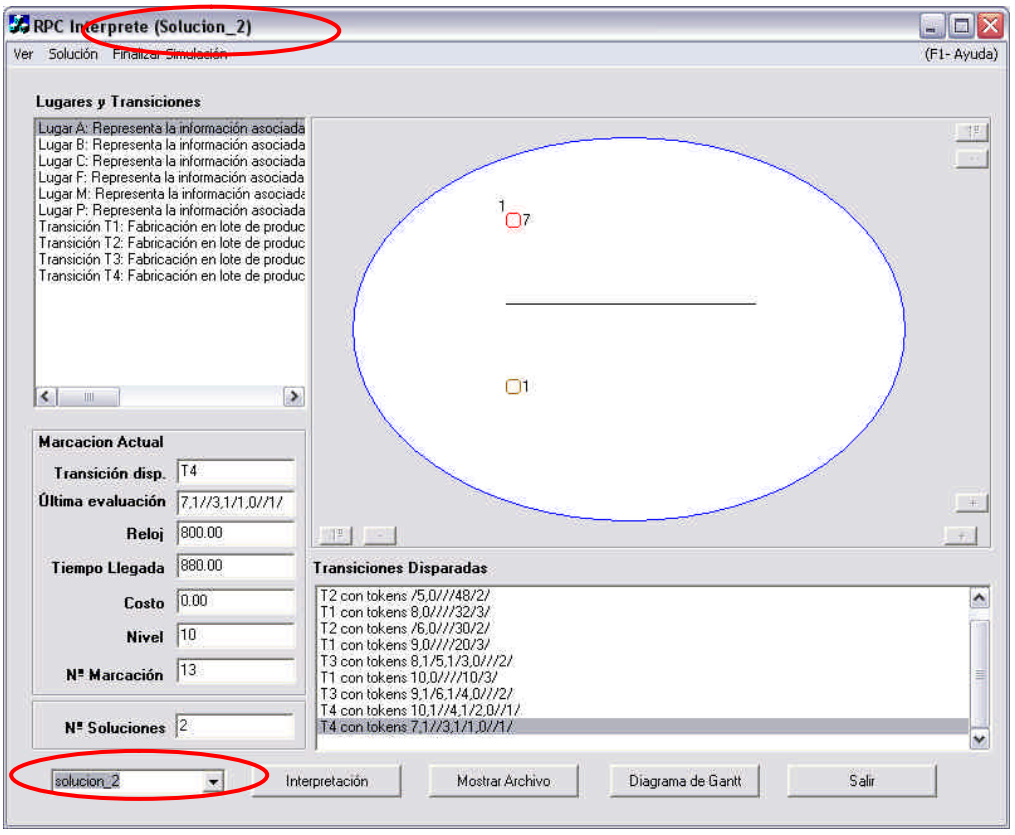

Figura 5.9: Segunda solución del problema

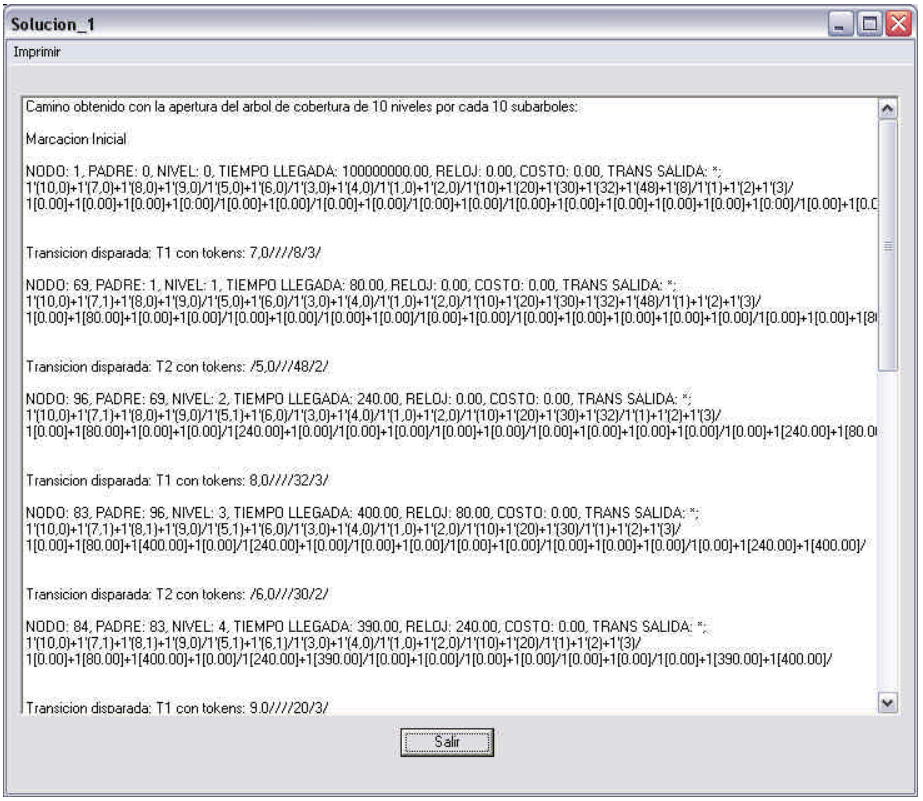

Figura 5.10: Mostrar primera solución

En la figura 5.11, se muestra la interfaz a la que se accede a partir del botón *Interpretación*, en esta interfaz se muestra un informe del archivo de la primera solución, utilizando la información de la BBDD del modelo.

Y por último pulsando el botón *Diagrama de Gantt*, se puede crear un diagrama representativo de la evolución de la solución del problema en tiempo. En la figura 5.12 se muestra la relación temporal existente entre el Conjunto Color *TA* (*Identificador de la tarea (7, 8, 9 ó 10) realizada para fabricar el producto A*) y el disparo de las transiciones. En esta figura se observa como la tarea *7* está implicada en el disparo de la transición *T1* (fabricación del producto *A*) en el instante de tiempo *0*, finalizado en el instante *80*, y en el disparo de la transición T4 (fabricación del producto *E*) en el instante de tiempo *600*, finalizando en el instante *680*.

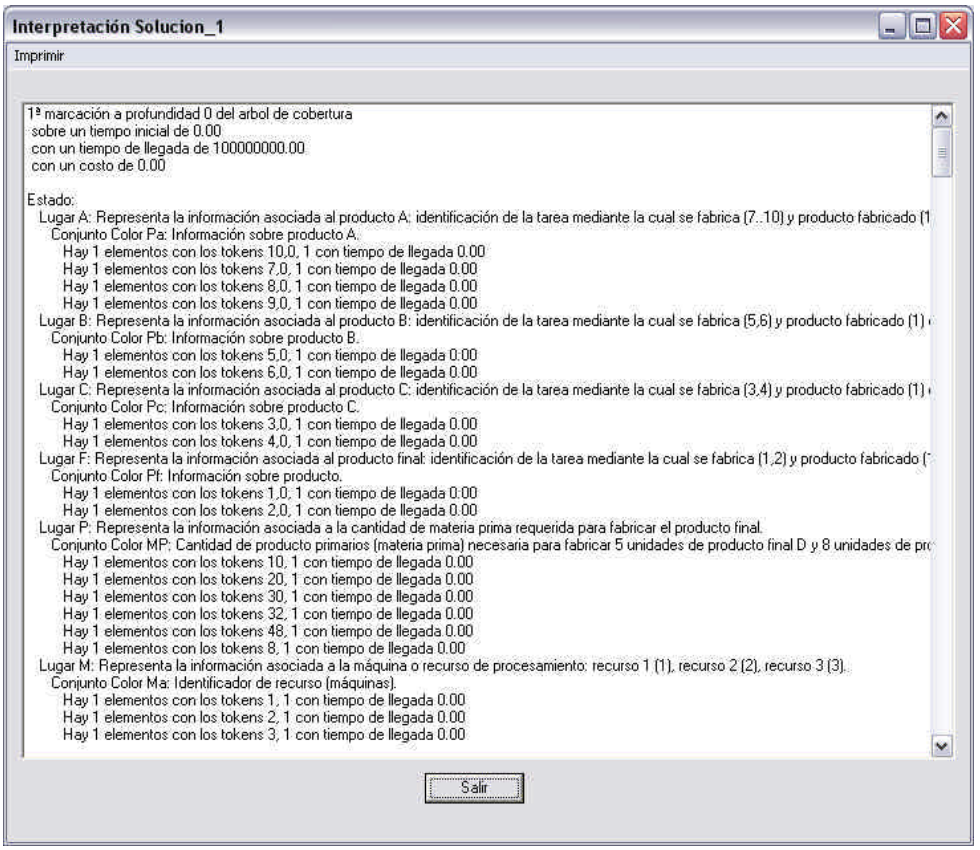

Figura 5.11: Interpretar primera solución

Figura 5.12: Diagrama de Gantt de *TA* respecto a las transiciones

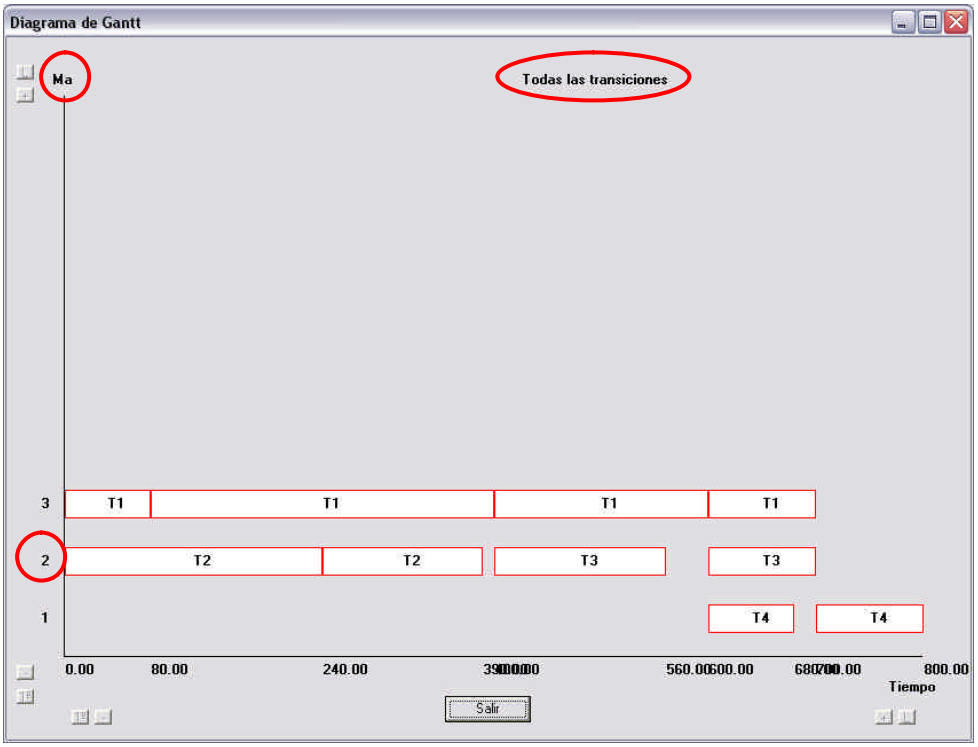

Figura 5.13: Diagrama de Gantt de *Ma* respecto a las transiciones

En la figura 5.13 se representa la relación temporal existente entre el Conjunto Color *Ma* (*Identificador de recurso*) y el disparo de las transiciones. La figura muestra como el recuso *2* está implicado en el disparo de las transiciones *T2* (fabricación del producto *B*) y *T3* (fabricación del producto *C*). La transición *T2* se dispara en los instantes de tiempo *0* y *240*, finalizando en *240* y *390*, respectivamente, y la transición *T3* se dispara en los instantes *400* y *600*, finalizando en *500* y *700*, respectivamente.

CAPÍTULO 6

CONCLUSIONES Y LÍNEAS FUTURAS DE INVESTIGACIÓN

Este proyecto ha consistido en el desarrollo de una interfaz para un simulador de modelos de RPCs. La interfaz presenta dos funcionalidades principales:

- Visualización de la ejecución de la simulación.
- Visualización de los resultados de la simulación.

Los objetivos planteados han sido alcanzados. La interfaz desarrollada facilita el entendimiento por parte del usuario de la información mostrada.

Un aspecto importante de la interfaz es el diagrama de Gantt. Este se ha adecuado para que el usuario pueda seleccionar los elementos que quiere que se representen, así como añadir un sistema de scroll para que en casos de simular un modelo complejo, el diagrama de Gantt se adapte a un tamaño adecuado, permitiendo al usuario definir el intervalo de tiempo que desea visualizar.

El mayor problema a solucionar durante el desarrollo de la interfaz consistió en que ésta no afectara la velocidad de ejecución del simulador de Redes de Petri Coloreadas. La interfaz permite mostrar el estado de la simulación en un momento determinado, para llevar a cabo este proceso, el simulador y la interfaz se comunican a través de mensajes, estos mensajes son unidireccionales, del simulador a la interfaz, con este procedimiento se consigue que la comunicación entre los dos programas afecte lo menos posible la velocidad de ejecución del simulador. Posteriormente la interfaz se encarga

de gestionar los datos que recibe para mostrar al usuario la información que solicita.

A continuación se indican las posibles líneas de continuación y mejoras aplicables a la interfaz desarrollada, dichas mejoras no han sido implementadas dado que no estaban contempladas en bs requerimientos de desarrollo:

- Con objeto de permitir al usuario mayor control sobre el simulador, sería deseable disponer de comunicaciones bidireccionales entre la interfaz de simulación y el simulador, consiguiendo de esta forma que el usuario pueda detener o pausar la simulación en puntos de interés, o incluso realizar pequeñas modificaciones sobre el modelo para comprobar como afectan al resultado de la simulación.

ANEXO A

DESCRIPCIÓN DE CLASES

En este anexo se describen todos los atributos y los métodos que se han desarrollado para cada una de las clases de las que se compone el proyecto.

A.1 Clase Marcación

Clase Marcación
Descripción: Clase que contiene toda la información procedente de la entidad Marcación, así como todos los métodos necesarios para la obtención de dicha información.

Atributo Reloj
Descripción: Atributo de la clase Marcación que contiene el valor de tiempo anterior al disparo de la transición

Atributo Costo
Descripción: Atributo de la clase Marcación que contiene el valor de costo de la Marcación.

Atributo Tiempo_de_Llegada
Descripción: Atributo de la clase Marcación que contiene el valor de tiempo después del disparo de la transición

Atributo Transicion_Disparada
Descripción: Atributo de la clase Marcación que contiene el identificador de la transición disparada

Atributo Ultima_Evaluacion
Descripción: Atributo de la clase Marcación que contiene la especificación de los tokens implicados en la transición

Atributo N_Marcacion
Descripción: Atributo de la clase Marcación que contiene el número de la marcación actual si se esta simulando, o el número total de marcaciones

Atributo Nivel
Descripción: Atributo de la clase Marcación que representa el nivel de profundidad en el que se encuentra la marcación actual dentro del árbol de cobertura

Método getReloj
Descripción: Método de la clase Marcación para obtener el valor del atributo Reloj de la misma clase Entradas: Salidas: valor de tiempo en milésimas de segundo

Método setReloj
Descripción: Método de la clase Marcación para asignar un valor al atributo Reloj de la misma clase Entradas: valor de tiempo en milésimas de segundo Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getCosto
Descripción: Método de la clase Marcación para obtener el valor del atributo Costo de la misma clase Entradas: Salidas: valor entero del costo de la marcación

Método setCosto
Descripción: Método de la clase Marcación para asignar un valor al atributo Costo de la misma clase Entradas: valor entero del costo de la marcación Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getTiempodeLlegada
Descripción: Método de la clase Marcación para obtener el valor del atributo Tiempo_de_Llegada de la misma clase Entradas: Salidas: valor de tiempo en milésimas de segundo

Método setTiempodeLlegada
Descripción: Método de la clase Marcación para asignar un valor al atributo Tiempo_de_Llegada de la misma clase Entradas: valor de tiempo en milésimas de segundo Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getTransicionDisparada
Descripción: Método de la clase Marcación para obtener el valor del atributo Transicion_Disparada de la misma clase Entradas: Salidas: cadena de caracteres con el identificador de la transición

Método setTransicionDisparada
Descripción: Método de la clase Marcación para asignar un valor al atributo Transicion_Disparada de la misma clase Entradas: cadena de caracteres con el identificador de la transición Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getUltimaEvaluacion
Descripción: Método de la clase Marcación para obtener el valor del atributo Ultima_Evaluacion de la misma clase Entradas: Salidas: cadena de caracteres con la especificación de tokens de la transición

Método setUltimaEvaluacion
Descripción: Método de la clase Marcación para asignar un valor al atributo Ultima_Evaluacion de la misma clase Entradas: cadena de caracteres con la especificación de tokens de la transición Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getNMarcacion
Descripción: Método de la clase Marcación para obtener el valor del atributo N_Marcacion de la misma clase Entradas: Salidas: Valor numérico

Método setNMarcacion
Descripción: Método de la clase Marcación para asignar un valor al atributo N_Marcacion de la misma clase Entradas: Valor numérico Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getNivel
Descripción: Método de la clase Marcación para obtener el valor del atributo Nivel de la misma clase Entradas: Salidas: Valor numérico

Método setNivel
Descripción: Método de la clase Marcación para asignar un valor al atributo Nivel de la misma clase Entradas: Valor numérico Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDB
Descripción: Método de la clase Marcación para obtener de la BBDD toda la información necesaria, este método agrupa los métodos que acceden a la BBDD Entradas: identificador de la transición, posición del lugar y posición del color Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getMarcacion
Descripción: Método de la clase Marcación para obtener toda la información necesaria de la última Marcación que se ha generado Entradas: Salidas: Verdadero si finaliza la operación correctamente o Falso si no

A.2 Clase Token

Clase Token
Descripción: Clase que contiene toda la información procedente de la entidad Marcación referente al token, así como todos los métodos necesarios para la obtención de dicha información.

Atributo Coeficiente
Descripción: Atributo de la clase Token que contiene el número de elementos que tienen el mismo color en un Lugar

Atributo Color
Descripción: Atributo de la clase Token que contiene el valor de color

Método getCoeficiente
Descripción: Método de la clase Token para obtener el valor del atributo Coeficiente de la misma clase Entradas: Salidas: Valor numérico

Método setCoeficiente
Descripción: Método de la clase Token para asignar un valor al atributo Coeficiente de la misma clase Entradas: Valor numérico Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getColor
Descripción: Método de la clase Token para obtener el valor del atributo Color de la misma clase Entradas: Salidas: cadena de caracteres que representa un valor o conjunto de valores

Método setColor
Descripción: Método de la clase Token para asignar un valor al atributo Color de la misma clase Entradas: cadena de caracteres que representa un valor o conjunto de valores Salidas: Verdadero si finaliza la operación correctamente o Falso si no

A.3 Clase Tiempo

Clase Tiempo
Descripción: Clase que contiene toda la información procedente de la entidad Marcación referente al tiempo de un token, así como todos los métodos necesarios para la obtención de dicha información.

Atributo Coeficiente
Descripción: Atributo de la clase Tiempo que contiene el número de elementos de un lugar que tienen el mismo valor de tiempo siendo del mismo color

Atributo Valor
Descripción: Atributo de la clase Tiempo que contiene el valor de tiempo

Método getCoeficiente
Descripción: Método de la clase Tiempo para obtener el valor del atributo Coeficiente de la misma clase Entradas: Salidas: Valor numérico

Método setCoeficiente
Descripción: Método de la clase Tiempo para asignar un valor al atributo Coeficiente de la misma clase Entradas: Valor numérico Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getValor
Descripción: Método de la clase Tiempo para obtener el valor del atributo Valor de la misma clase Entradas: Salidas: Cadena de caracteres que representa un valor o conjunto de valores de tiempo en milésimas de segundo

Método SetValor
Descripción: Método de la clase Tiempo para asignar un valor al atributo Valor de la misma clase Entradas: Cadena de caracteres que representa un valor o conjunto de valores de tiempo en milésimas de segundo Salidas: Verdadero si finaliza la operación correctamente o Falso si no

A.4 Clase Transición

Clase Transición
Descripción: Clase que contiene toda la información procedente de la BBDD referente a una transición, así como todos los métodos necesarios para la obtención de dicha información.

Atributo Nombre
Descripción: Atributo de la clase Transición que contiene el identificador de la transición

Atributo Descripción
Descripción: Atributo de la clase Transición que contiene una descripción semántica de la transición

Atributo Origen
Descripción: Atributo de la clase Transición que contiene el identificador del lugar de donde provienen los tokens que habilitan la transición

Atributo Destino
Descripción: Atributo de la clase Transición que contiene el identificador del lugar donde se colocan los tokens generados después del disparo de la transición

Método getNombre
Descripción: Método de la clase Transición para obtener el valor del atributo Nombre de la misma clase Entradas: Salidas: cadena de caracteres que representa un identificador

Método setNombre
Descripción: Método de la clase Transición para asignar un valor al atributo Nombre de la misma clase Entradas: cadena de caracteres que representa un identificador Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDescripcion
Descripción: Método de la clase Transición para obtener el valor del atributo Descripción de la misma clase Entradas: Salidas: cadena de caracteres (larga)

Método setDescripcion
Descripción: Método de la clase Transición para asignar un valor al atributo Descripción de la misma clase Entradas: cadena de caracteres (larga) Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getOrigen
Descripción: Método de la clase Transición para obtener el valor del atributo Origen de la misma clase Entradas: Salidas: cadena de caracteres representa un identificador

Método setOrigen
Descripción: Método de la clase Transición para asignar un valor al atributo Origen de la misma clase Entradas: cadena de caracteres representa un identificador Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDestino
Descripción: Método de la clase Transición para obtener el valor del atributo Destino de la misma clase Entradas: Salidas: cadena de caracteres representa un identificador

Método setDestino
Descripción: Método de la clase Transición para asignar un valor al atributo Destino de la misma clase Entradas: cadena de caracteres representa un identificador Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDBTransicion
Descripción: Método de la clase Transición para obtener toda la información de una transición de la BBDD, y asignarla a los atributos Entradas: cadena de caracteres, representa el identificador de la transición Salidas: Verdadero si finaliza la operación correctamente o Falso si no

A.5 Clase Lugar

Clase Lugar
Descripción: Clase que contiene toda la información procedente de la BBDD referente a un lugar, así como todos los métodos necesarios para la obtención de dicha información.

Atributo Nombre
Descripción: Atributo de la clase Lugar que contiene el identificador del lugar

Atributo Descripción
Descripción: Atributo de la clase Lugar que contiene una descripción semántica del lugar

Método getNombre
Descripción: Método de la clase Lugar para obtener el valor del atributo Nombre de la misma clase Entradas: Salidas: cadena de caracteres representa un identificador

Método setNombre
Descripción: Método de la clase Lugar para obtener el valor del atributo Nombre de la misma clase Entradas: cadena de caracteres representa un identificador Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDescripcion
Descripción: Método de la clase Lugar para obtener el valor del atributo Descripción de la misma clase Entradas: Salidas: cadena de caracteres (larga)

Método setDescripcion
Descripción: Método de la clase Lugar para obtener el valor del atributo Descripción de la misma clase Entradas: cadena de caracteres (larga) Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDBLugar
Descripción: Método de la clase Lugar para obtener toda la información de un lugar de la BBDD, y asignarla a los atributos Entradas: valor, representa la posición del lugar en la marcación Salidas: Verdadero si finaliza la operación correctamente o Falso si no

A.6 Clase Conjunto_Color

Clase Conjunto_Color
Descripción: Clase que contiene toda la información procedente de la BBDD referente a un conjunto color, así como todos los métodos necesarios para la obtención de dicha información.

Atributo Nombre
Descripción: Atributo de la clase Conjunto_Color que contiene el identificador del conjunto color

Atributo Descripción
Descripción: Atributo de la clase Conjunto_Color que contiene una descripción semántica del conjunto color

Atributo Definición
Descripción: Atributo de la clase Conjunto_Color que contiene especificación del tipo del conjunto color

Método getNombre
Descripción: Método de la clase Conjunto_Color para obtener el valor del atributo Nombre de la misma clase Entradas: Salidas: cadena de caracteres representa un identificador

Método setNombre
Descripción: Método de la clase Conjunto_Color para obtener el valor del atributo Nombre de la misma clase Entradas: cadena de caracteres representa un identificador Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDescripcion
Descripción: Método de la clase Conjunto_Color para obtener el valor del atributo Descripción de la misma clase Entradas: Salidas: cadena de caracteres (larga)

Método setDescripcion
Descripción: Método de la clase Conjunto_Color para obtener el valor del atributo Descripción de la misma clase Entradas: cadena de caracteres (larga) Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDefinicion
Descripción: Método de la clase Conjunto_Color para obtener el valor del atributo Descripción de la misma clase Entradas: Salidas: cadena de caracteres

Método setDefinicion
Descripción: Método de la clase Conjunto_Color para obtener el valor del atributo Descripción de la misma clase Entradas: cadena de caracteres Salidas: Verdadero si finaliza la operación correctamente o Falso si no

Método getDBConjuntoColor
Descripción: Método de la clase Conjunto_Color para obtener toda la información de un conjunto color de la BBDD, y asignarla a los atributos Entradas: valor, representa la posición del conjunto color en el lugar Salidas: Verdadero si finaliza la operación correctamente o Falso si no

A.7 Clase RPC_Interpreter

Clase RPC_Interpreter
Descripción: Clase Interfaz, muestra toda la información que se ha recopilado en las clase anteriores

Atributo m_lugar
Descripción: Atributo de la clase RPC_Interpreter que contiene la lista de lugares seleccionables

Atributo m_costo
Descripción: Atributo de la clase RPC_Interpreter que contiene el valor de costo

Atributo m_evaluacion
Descripción: Atributo de la clase RPC_Interpreter que contiene que contiene la especificación de la última evaluación

Atributo m_soluciones
Descripción: Atributo de la clase RPC_Interpreter que contiene el número de soluciones encontradas

Atributo m_solutiones
Descripción: Atributo de la clase RPC_Interpreter que contiene una lista de soluciones seleccionable

Atributo m_transiciones
Descripción: Atributo de la clase RPC_Interpreter que contiene una lista de transiciones disparadas

Atributo m_nivel
Descripción: Atributo de la clase RPC_Interpreter que contiene el nivel en el que se encuentra el árbol de cobertura

Atributo m_nodo
Descripción: Atributo de la clase RPC_Interpreter que contiene el número de marcación actual

Atributo m_nombre
Descripción: Atributo de la clase RPC_Interpreter que contiene el identificador de la transición actual

Atributo m_reloj
Descripción: Atributo de la clase RPC_Interpreter que contiene el valor de reloj

Atributo m_tiempo
Descripción: Atributo de la clase RPC_Interpreter que contiene el valor de tiempo de llegada

Atributo marcación
Descripción: Atributo de la clase RPC_Interpreter que contiene la clase Marcación

Atributo color
Descripción: Atributo de la clase RPC_Interpreter que contiene la clase Conjunto_Color

Atributo transición
Descripción: Atributo de la clase RPC_Interpreter que contiene la clase Transición

Atributo lugar
Descripción: Atributo de la clase RPC_Interpreter que contiene la clase Lugar

Atributo interpreter/marcación
Descripción: Atributo de la clase RPC_Interpreter que contiene la clase Interpreter/Marcación

Atributo Gantt
Descripción: Atributo de la clase RPC_Interpreter que contiene la clase Gantt

Método OnInterpreter
Descripción: Método de la clase RPC_Interpreter para llamar a la clase Interpreter/Marcación Entradas: Salidas:

Método OnGantt
Descripción: Método de la clase RPC_Interpreter para llamar a la clase Gantt Entradas: Salidas:

Método OnMarcacion
Descripción: Método de la clase RPC_Interpreter para llamar a la clase Interpreter/Marcación Entradas: Salidas:

A.8 Clase Interprete/Marcación

Clase Interprete/Marcación
Descripción: Clase Interfaz, muestra toda la información que se ha recopilado en las clase anteriores en formato texto

Atributo m_text
Descripción: Atributo de la clase Interpreter/Marcación que contiene el texto a mostrar

Método SetCaption
Descripción: Método de la clase Interpreter/Marcación para asignar un título a la Interfaz Entradas: Salidas:

A.9 Clase Gantt

Clase Gantt
Descripción: Clase Interfaz, muestra un diagrama de Gantt

BIBLIOGRAFIA

1. Joseph Casanovas – Antoni Guasch - Miquel Àngel Piera – Jaume Figueras (Junio, 2003). *"Modelado y simulación. Aplicación a procesos logísticos de fabricación y servicios"* Edicions UPC.

2. Xavier Ferré Grau, María Isabel Sánchez Segura: *"Desarrollo orientado a objetos con UML*" Facultad de Informática. UPM. http://groups.google.com/group/lmp2006/files (umlTotal.pdf)

3. Kate Gregory (1999): "*Microsoft Visual C++ 6"* Prentice Hall

4. Kurt Jensen (1997): "*Coloured Petri Nets: Basic concepts, Analysis Methods and Practical Use*" Springer Verlag.

5. NARCISO Farias, Mercedes Elizabeth (Marzo, 2007). "*Metodología para la resolución de problemas de optimización mediante la exploración del espacio de estados generado a partir de modelos de Redes de Petri Coloreadas.*" Tesis de Doctorado. Universitat Autònoma de Barcelona.

6. Microsoft. "*Microsoft Developer Network"* (Consulta Junio, 2007) http://msdn2.microsoft.com

RESUM

Aquest projecte presenta el desenvolupament d'una interficie d'usuari que permet visualitzar i interpretar, en forma de texte o en forma gràfica, els resultats de la simulació de models de Xarxes de Petri Acolorides (XPAs). L'estudi de les solucions obtingudes mitjançant la simulació de models de XPAs, permetrá analitzar y evaluar tan el comportament com el rendiment del sistema modelat. La interficie ha estat programada en Visual C++.

RESUMEN

Este proyecto presenta el desarrollo de una interfaz de usuario que permite visualizar e interpretar, en forma textual o en forma gráfica, los resultados de la simulación de modelos de Redes de Petri Coloreadas (RPCs). El estudio de las soluciones obtenidas mediante la simulación de modelos de RPCs, permitirá analizar y evaluar tanto el comportamiento como el rendimiento del sistema modelado. La interfaz ha sido programada en Visual C++.

ABSTRACT

This Project presents the development of a user interface that permits to visualize and to interpret, in textual form and graphical form, the results of the simulation of Coloured Petri Network (CPNs) models. The study of the solutions obtained by means of the simulation of the CPNs models, will permit to analyse and evaluate as much the behaviour as the performance of the modelled system. The interface has been programmed with a Visual C++ tool.