|  |
| --- |
| programación concurrente |
| TRABAJO FINAL |
| ACCIONAMIENTO DE UNA BOMBA DE AGUA EN UNA MINA |
|  |
|  |
|  |

|  |
| --- |
|  |

INDICE

[1 INTRODUCCIÓN 3](#_Toc444023931)

[2 DESARROLLO 3](#_Toc444023932)

[2.1 Programa en Java 4](#_Toc444023933)

[2.1.1 Requerimientos 4](#_Toc444023934)

[2.1.2 Procesador de Petri 4](#_Toc444023935)

[2.1.3 Monitor 4](#_Toc444023936)

[2.1.4 Políticas 4](#_Toc444023937)

[2.2 Red de Petri 4](#_Toc444023938)

[2.2.1 Requerimientos 4](#_Toc444023939)

[2.2.2 Sensores de monitoreo 5](#_Toc444023940)

[2.2.3 Accionamiento de la bomba 6](#_Toc444023941)

[3 TESTING 6](#_Toc444023942)

[4 CONCLUSIONES 10](#_Toc444023943)

# 1 INTRODUCCIÓN

En este trabajo se desarrolla un sistema que controla una bomba de agua que extrae agua de un sumidero de un pozo de una mina.

Para su implementación, se comenzó desarrollando un procesador de redes de petri realizado en lenguaje Java, al cual se le agregarían funcionalidades extras, en base a las necesidades del sistema a desarrollar, partiendo de la posible ejecución de una RdP con transiciones no temporales, sin arcos inhibidores, ni condiciones de prioridad entre transiciones.

Una vez terminado el desarrollo completo de un procesador completo de RdP, junto al testing necesario, se procedió a la elaboración de una red que cumpliera con todas las necesidades del problema planteado, para luego ser simulada y ejecutada sobre nuestro programa.

# 2 DESARROLLO

Como se introdujo en un principio, lo primero en realizarse fue el programa en Java que fuera capaz de ejecutar una RdP, partiendo del archivo .html que devuelve el programa de código abierto "PIPE" (Platform Independent Petri net Editor) en su versión 4.3.0. El mismo, era capaz de ejecutar redes que cumplieran con la ecuación:

Donde:

I : incidencia

m: marcado

δ: disparo

Por supuesto, esto no es suficiente para el problema, debido a la necesidad de utilizar arcos inhibidores, transiciones temporales y además, un sistema de políticas para establecer una prioridad de disparo entre transiciones.

Se tomó, entonces, la decisión de tomar cada una de estas necesidades por separado, tomando cada una de ellas como un requisito del programa en Java, y realizando tests de cada una para seguir adelante con la elaboración de la RdP.

## 2.1 Programa en Java

### 2.1.1 Requerimientos

* RQ1: El programa no debe poder disparar una transición no sensibilizada.
* RQ2: El programa debe poder disparar las transiciones sensibilizadas.
* RQ3: Se necesita un método para conocer las transiciones sensibilizadas.
* RQ4: Se debe poder establecer prioridades de disparo entre transiciones.
* RQ5: Se debe poder disparar una transición sensibilizada por tiempo.
* RQ6: No se debe poder disparar una transición no sensibilizada por tiempo.
* RQ7: Se debe poder disparar una transición sensibilizada por tiempo.
* RQ8: Se debe poder disparar una transición sensibilizada por tiempo.
* RQ9: Se deben poder disparar transiciones de manera automática, en el instante que se sensibilizaron (transiciones automáticas que no requieren actor).

### 2.1.2 Procesador de Petri

El procesador desarrollado contiene dos constructores. Uno es para trabajar con redes temporales, y el otro con redes sin tiempo:

#### 2.1.2.1 Redes no temporales

Este tipo de redes, puede o no, contener transiciones automáticas, las cuales son solicitadas por el constructor al instanciar la red, pero en el caso que no las contenga, el programa simplemente las iniciará como *null*y procederá con la ejecución sin inconvenientes.

Para realizar un disparo, se tiene en cuenta la siguiente ecuación:

#### 2.1.2.2 Redes temporales

### 2.1.3 Monitor

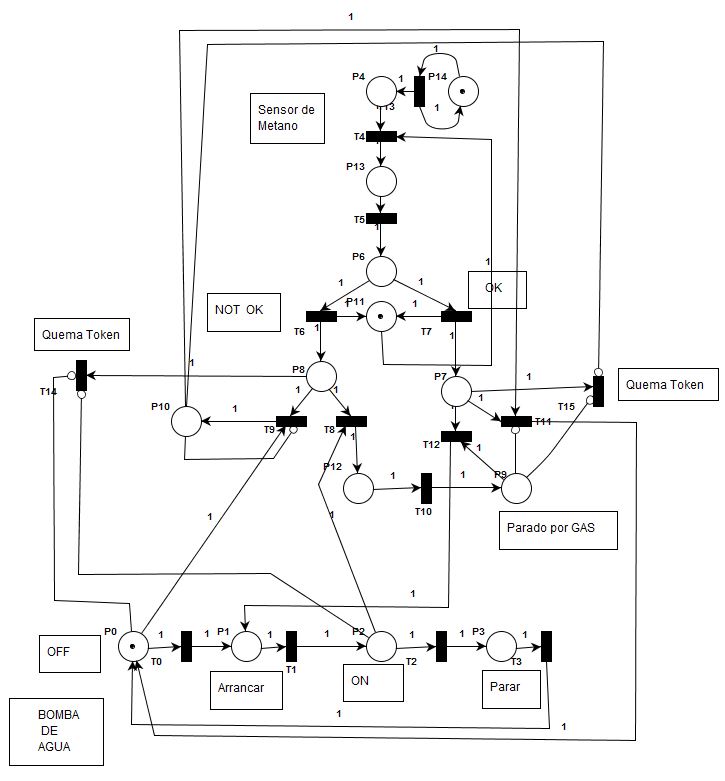
### 2.1.4 Políticas

## 2.2 Red de Petri

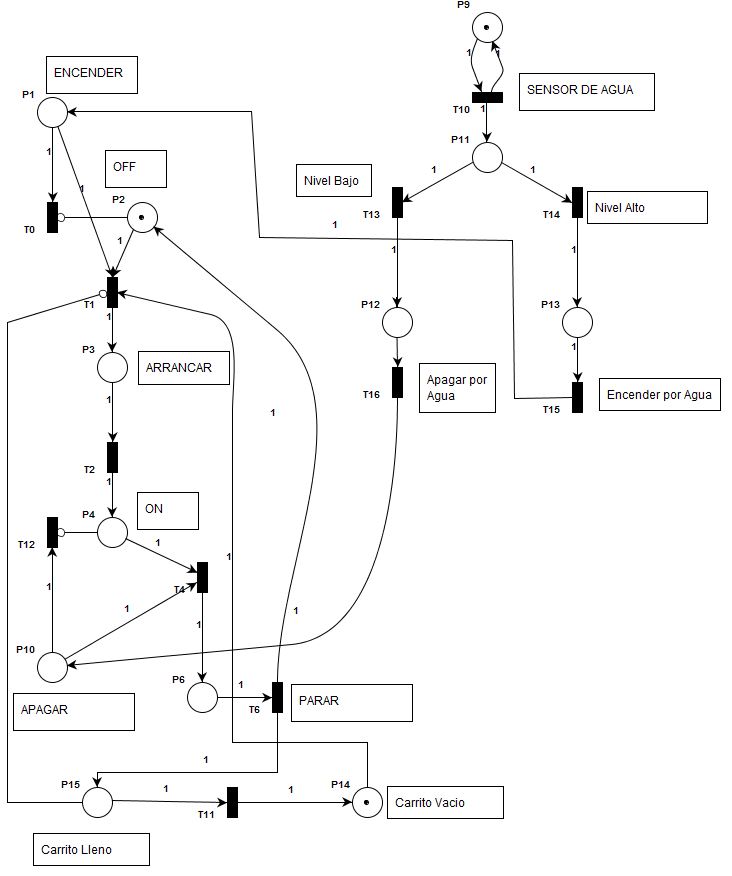
### 2.2.1 Requerimientos

* La bomba se enciende por nivel superior o indicación de un operario.
* La bomba se apaga por nivel inferior o indicación de un operario.
* CO y Aire avisan por alarma
* Sistema se controla por una consola
* Todos los eventos del sistema son guardados para luego ser visualizados

### 2.2.2 Sensores de monitoreo

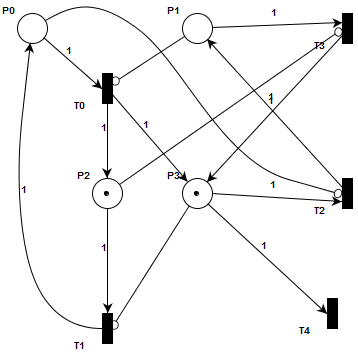


### 2.2.3 Accionamiento de la bomba

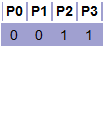
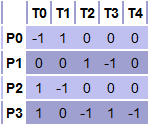
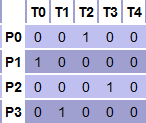


# 3 TESTING

**Red de prueba:**



**Marcado Inicial Incidencia: Inhibición:**

**Requerimientos Funcionales:**

|  |  |
| --- | --- |
| **Descripción del Test** | Disparar transición no sensibilizada |
| **Ejecución** | 1. Se crea la red con la I,m0 y H de la RdP de prueba. 2. Se intenta disparar T3. 3. Se comprueba que el nuevo marcado sea igual al inicial |
| **Resultado esperado** | La transición no se puede disparar. Marcado no modificado. |
| **Pass/Fail** | **PASS** |

|  |  |
| --- | --- |
| **Realizado por** | Ariel Viñas |
| **Descripción del Test** | Disparar transición sensibilizada |
| **Ejecución** | 1. Se crea la red con la I,m0 y H de la RdP de prueba. 2. Se intenta disparar T2. 3. Se comprueba que el nuevo marcado sea igual al inicial |
| **Resultado esperado** | La transición se puede disparar. Nuevo marcado: |
| **Pass/Fail** | **PASS** |
| **Realizado por** | Ariel Viñas |

|  |  |
| --- | --- |
| **Descripción del Test** | Comprobar Política |
| **Ejecución** | 1. Se crea la red con la I,m0 y H de la RdP de prueba. 2. Se aplica la prioridad: T2 - T3 - T0 - T4 - T1 3. Se intenta disparar T2 y T4 4. Se comprueba que T2 sea la elegida para ejecutar |
| **Resultado esperado** | Se ejecutará T2 |
| **Pass/Fail** | **PASS** |
| **Realizado por** | Ariel Viñas |

|  |  |
| --- | --- |
| **Descripción del Test** | Transiciones sensibilizadas |
| **Ejecución** | 1. Se crea la red con la I,m0 y H de la RdP de prueba. 2. Se llama la función *getSensibilizadas().* 3. Se comprueba que las transiciones que devuelve sean T2 y T4 |
| **Resultado esperado** | Las sensibilizadas son T2 y T4 |
| **Pass/Fail** | **PASS** |
| **Realizado por** | Ariel Viñas |

**Matriz Temporal**

|  |  |  |
| --- | --- | --- |
|  | α | β |
| T0 | 150 | 200 |
| T1 | 0 | 0 |
| T2 | 0 | 0 |
| T3 | 0 | 0 |
| T4 | 0 | 0 |

|  |  |
| --- | --- |
| **Descripción del Test** | Disparar Sensibilizada temporal |
| **Ejecución** | 1. Se crea la red con la I,m0, H y el tiempo de la RdP de prueba. 2. En el momento en que la red es creada se inicializan los contadores de las transiciones sensibilizadas 3. Se espera el tiempo necesario para que T0 quede sensibilizada por tiempo 4. Se comprueba que T0 quedó sensibilizada temporalmente. |
| **Resultado esperado** | Se espera un 1 en el subíndice 0 del vector de ejecución temporal. |
| **Pass/Fail** | **PASS** |
| **Realizado por** | Patricio Perpetua |

|  |  |
| --- | --- |
| **Descripción del Test** | Disparar No sensiblizada temporal |
| **Ejecución** | 1. Se crea la red con la I,m0, H y el tiempo de la RdP de prueba. 2. En el momento en que la red es creada se inicializan los contadores de las transiciones sensibilizadas 3. Se comprueba que la transición T0 no esta sensibilizada, porque no pasó el tiempo necesario. |
| **Resultado esperado** | Se espera un 0 en el subíndice 0 del vector de ejecución temporal. |
| **Pass/Fail** | **PASS** |
| **Realizado por** | Patricio Perpetua |

# 4 CONCLUSIONES