Simulador de Eventos Discretos: línea Tesla

Rodrigo Maranzana

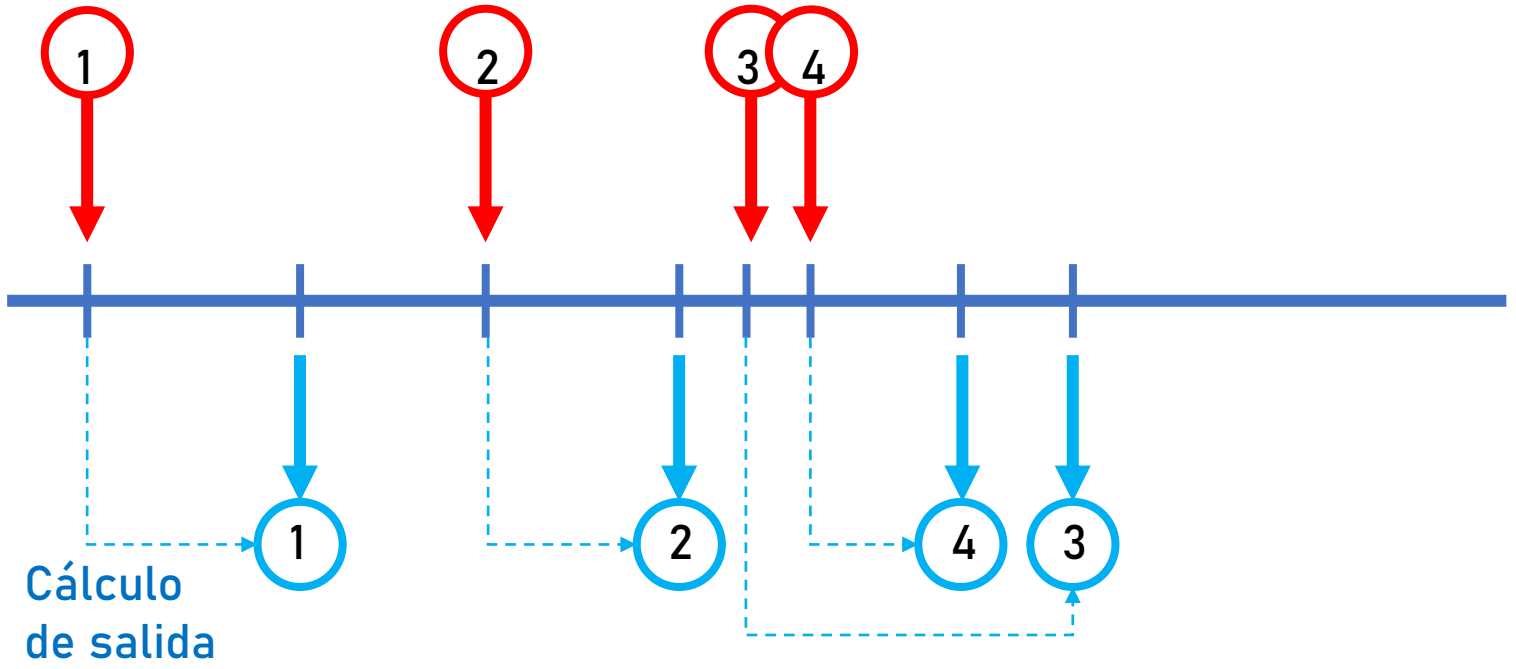
Simulador de eventos discretos, concepto

Mismo objetivo que en la Simulación en general:

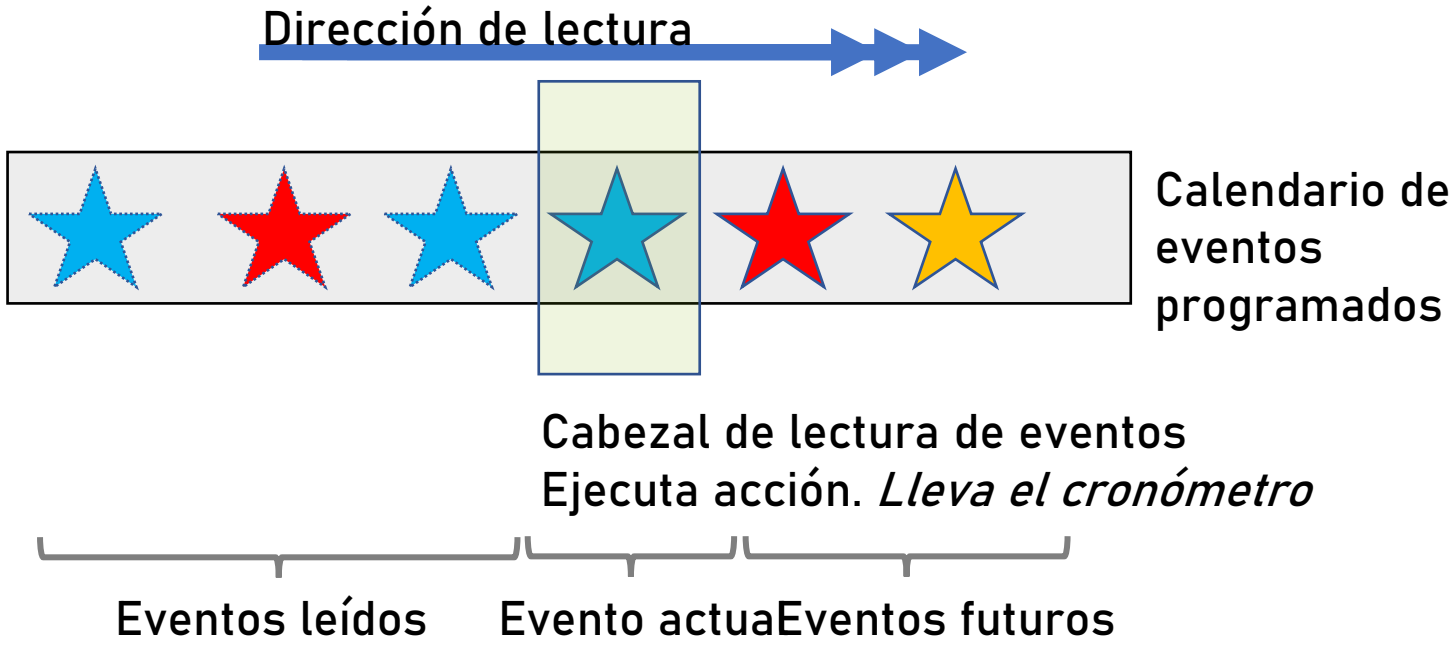
- *Predecir el comportamiento de un sistema dinámico complejo.*

- Los eventos detonan procesos
- Hay procesos que generan nuevos eventos

Simulador de eventos discretos, concepto



Lectura de eventos



Variantes de implementación

Soft conocidos en la industria:

- Arena simulation software
- AnyLogic
- Simul8

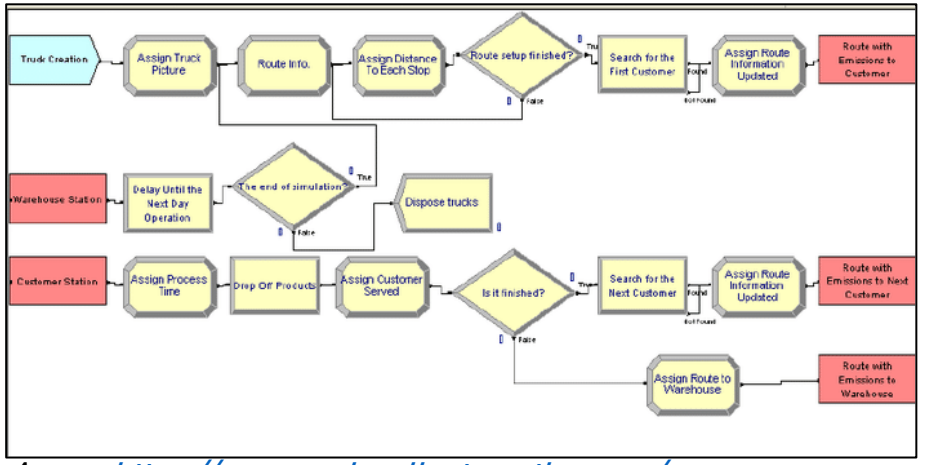
↑ Soporte de compañías conocidas con trayectoria
↑ Interfaz gráfica amigable
↑ Curva de aprendizaje suave
↓ Cajas negras, a medida
↓ Módulos pagos

Orientados a investigación aplicada:

- MATLAB Simulink

↑ Flexibilidad
↑ Ecosistema de MATLAB
↓ Curva de aprendizaje elevada
↓ Pago y costoso

Anylogic, https://www.anylogic.com/features/



Arena, https://www.rockwellautomation.com/es-ar/products/software/arena-simulation.html

Implementación en Python

Implementación plana, sin ayuda de librerías.

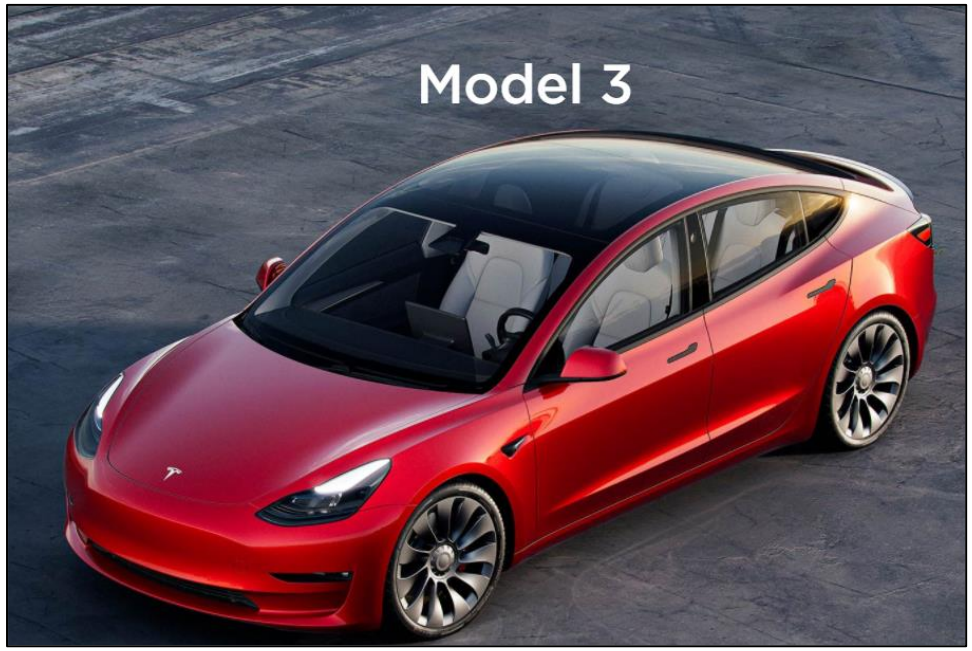
- Solo a modo didáctico, entender los conceptos.
- Nula escalabilidad.
- Baja performance.

https://simpy.readthedocs.io/en/latest/

Librería Simpy, simulador de eventos discretos

↑ Librería open-source, aceptada y mantenida por la comunidad
↑ Flexibilidad
↑ Gratuito, escalable
↓ Curva de aprendizaje elevada
↓ Sin interfaz gráfica
↓ Baja simplificación, requiere pensar lógicas adaptadas al problema particular

Caso Tesla



https://www.tesla.com/model3

- Auto eléctrico de tamaño mediano.
- Cuatro puertas
- Comienzo de fabricación: mediados 2017
- Modelo más accesible que sus predecesores.
- Auto eléctrico más vendido 2018-2020

“Elon Musk sobre el Tesla modelo 3: Estamos encaminándonos a, por lo menos, 6 meses de infierno de fabricación” – Business Insider, Julio 2017

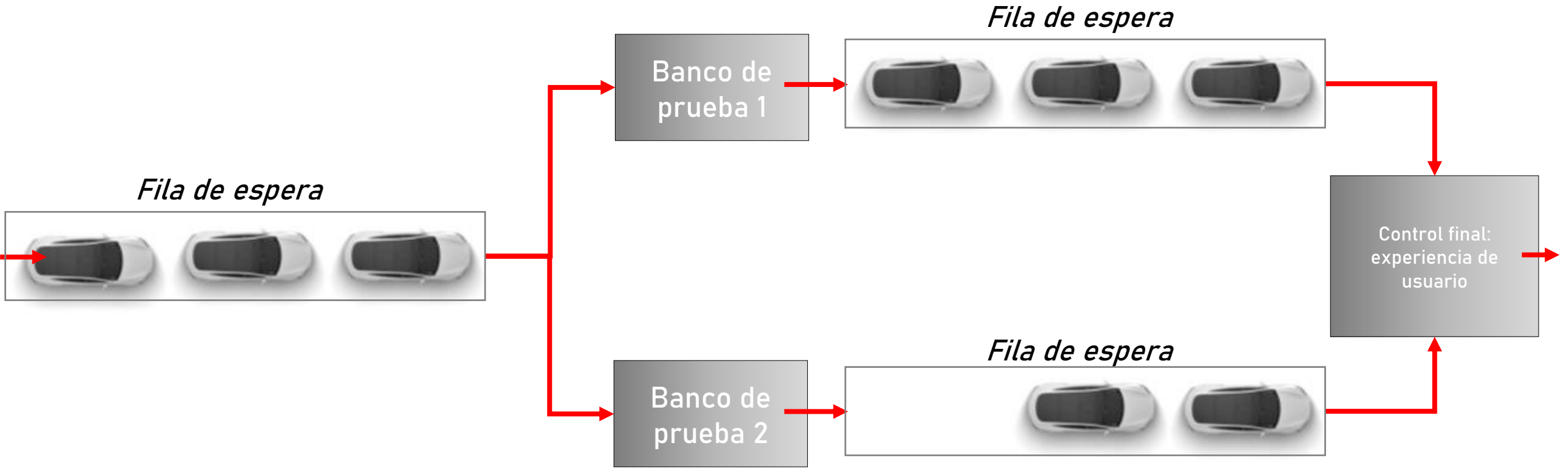
“El infierno de fabricación del Tesla Modelo 3 pone a prueba el esquema de –arreglar mientras se avanza – de Elon Musk ” – Los Angeles Times, Octubre 2017

Los pre-compradores de Tesla Modelo 3, tendremos notificaciones o información oficial?

Lo sabrás tan pronto lo hagamos. Estamos sumergidos en el ***infierno de fabricación***

Caso Tesla: configuración del sistema

Queremos modelizar la punta de línea de montaje:



Caso Tesla

- Simular proceso con “simulador de eventos discretos” en Python
- Codificar productos para futuros indicadores de tiempos.

- Tiempo de simulación: 4 horas.
- Objetivo:

  medir la cantidad de productos por hora.

  cantidad en filas por hora.

Caso Tesla: agentes del simulador

Filas de espera:

Atributo variable: cantidad en fila.

Atributo fijo: distribución y parámetros, política FIFO, entrega de la fila menor.

Estaciones de trabajo:

Atributo variable: libre/ocupada

Atributo fijo: distribución y parámetros.

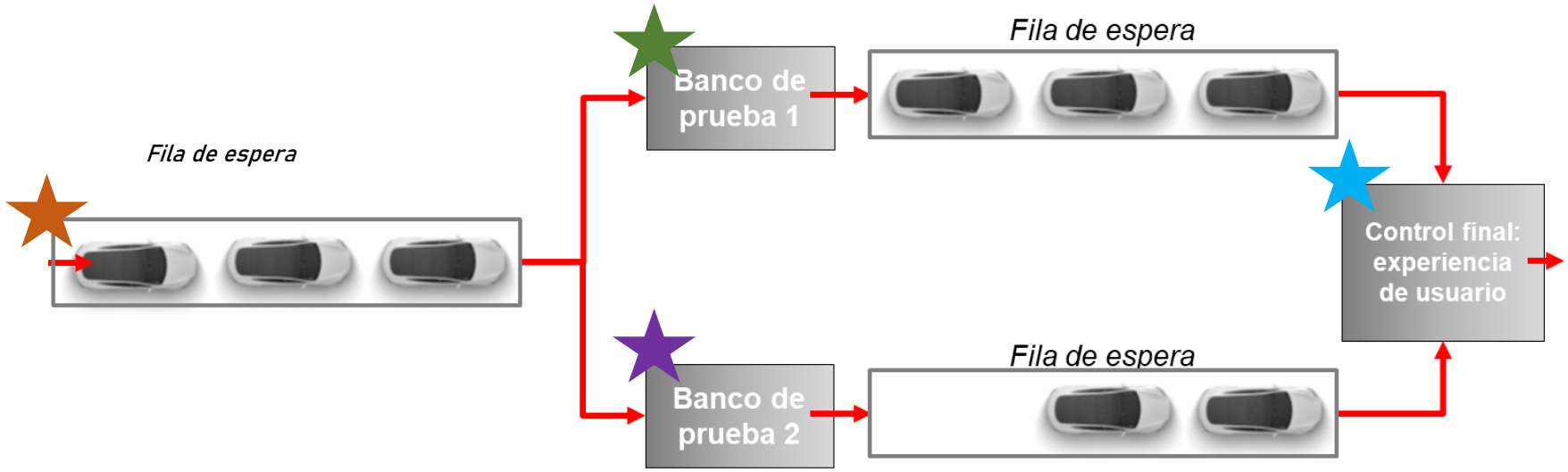
Vehículos:

Atributo variable: En máquina “X” / En fila de espera “X”

Atributo fijo: distribución de llegadas, productos iguales codificados.

Eventos sampleables posibles

Clave: simplificar eventos "sampleables" al máximo



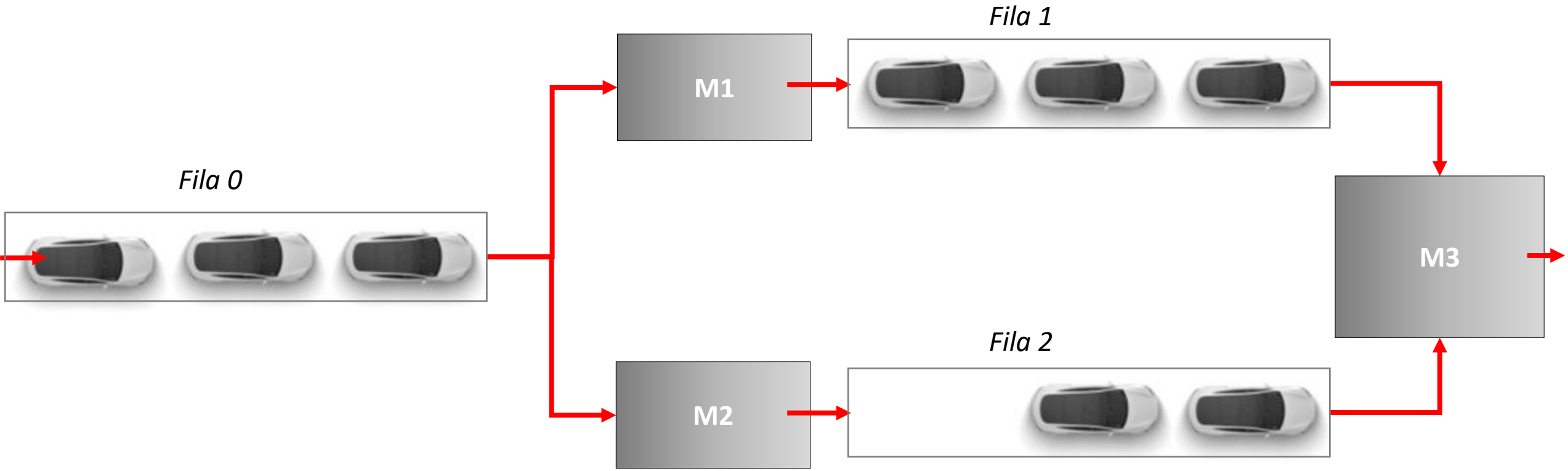
- Llegada del vehículo al sistema
- Salida del vehículo del banco de prueba 1
- Salida del vehículo del banco de prueba 2
- Salida del vehículo del control final

Composición de los eventos

- Tiempo de acción
- Tipo de evento
- Nº de vehículo asociado

- 25:56 min
- Salida de banco de prueba 1
- ID 2509

Codificación de máquinas



Datos para eventos sampleables

- El tiempo entre arribos es una variable aleatoria que sigue una distribución exponencial.

- Llegada del vehículo al sistema

- El tiempo de producción en cada máquina es una variable aleatoria que sigue una distribución normal con media y desvío estándar.

- Salida del vehículo de M1
- Salida del vehículo de M2
- Salida del vehículo de M3

Parámetros en Python

```python
import numpy as np
from numpy.random import exponential, normal
import matplotlib.pyplot as plt
import pandas as pd
```

```python
# Parámetros de llegadas:
media_llegada = 33 # u/hr
beta = 1/media_llegada # hr/u

# Parámetros de procesamiento en máquina 1:
mu_m1 = 1/25 # hr/u
sigma_m1 = mu_m1 * 0.1 # hr/u

# Parámetros de procesamiento en máquina 2:
mu_m2 = 1/10 # hr/u
sigma_m2 = mu_m2 * 0.1 # hr/u

# Parámetros de procesamiento en máquina 3:
mu_m3 = 1/33 # hr/u
sigma_m3 = mu_m3 * 0.15 # hr/u

codigos_productos = range(0, 20000)
gen_codigo_productos = iter(codigos_productos)
```

Parámetros para sampleo de llegadas

Parámetros para sampleo de salidas M1, M2 y M3

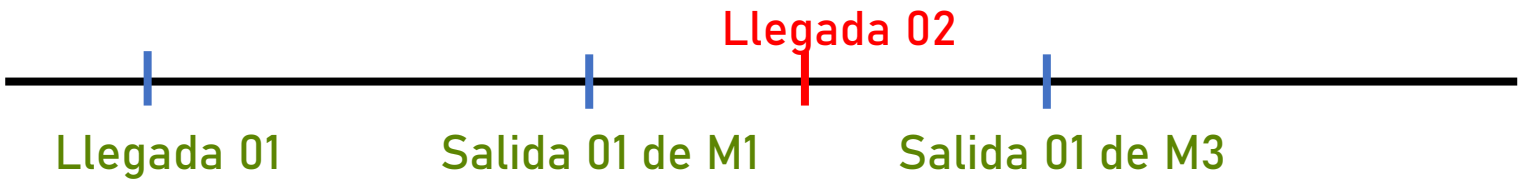
Proceso de sampleo

Llegada del vehículo al sistema

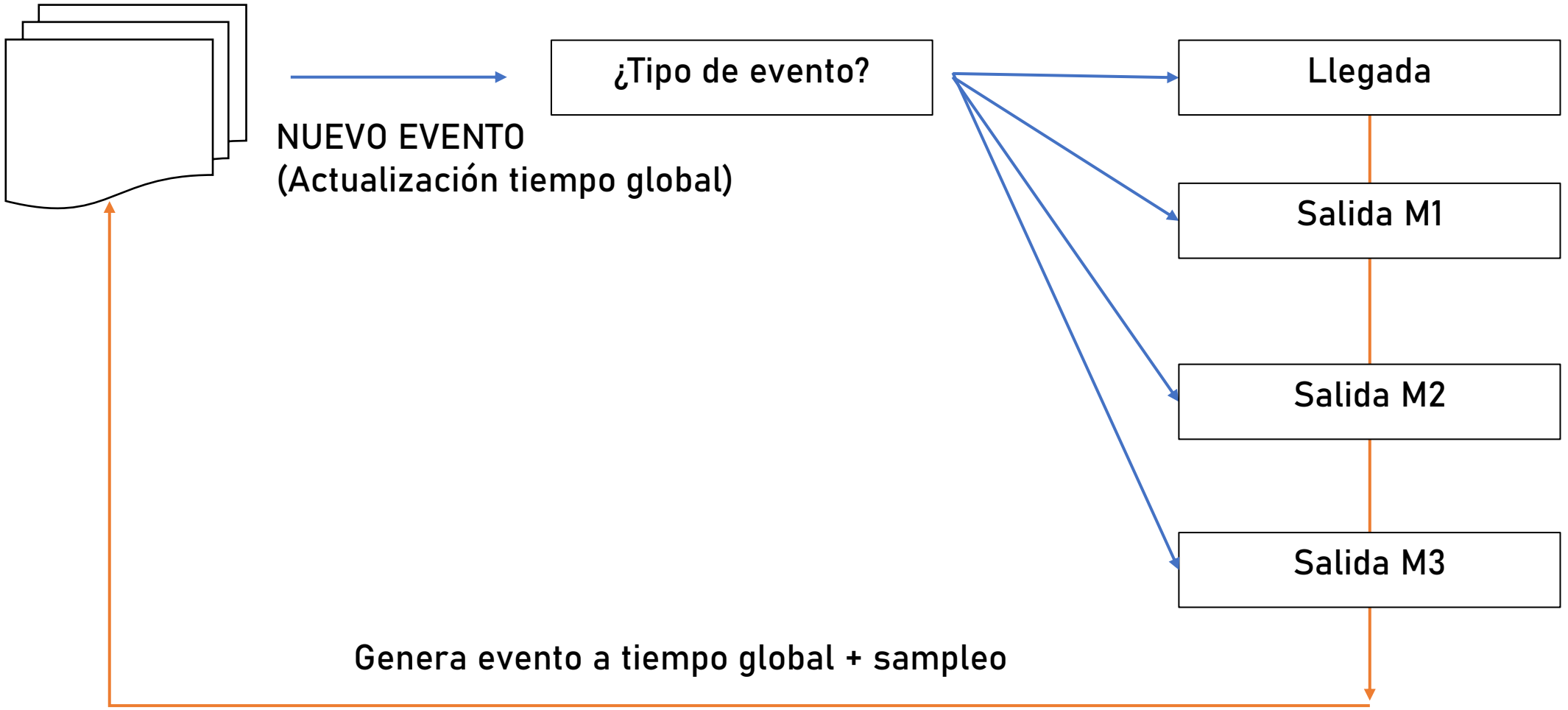
t_entre_llegadas = samplear_exponencial(tasa de llegadas)
t_de_llegada = t_global + t_entre_llegadas

Salida del vehículo de M1
Salida del vehículo de M2
Salida del vehículo de M3

t_proceso = samplear_normal(media, desvío)
t_de_salida = t_global + t_proceso



Gestión de eventos sampleables

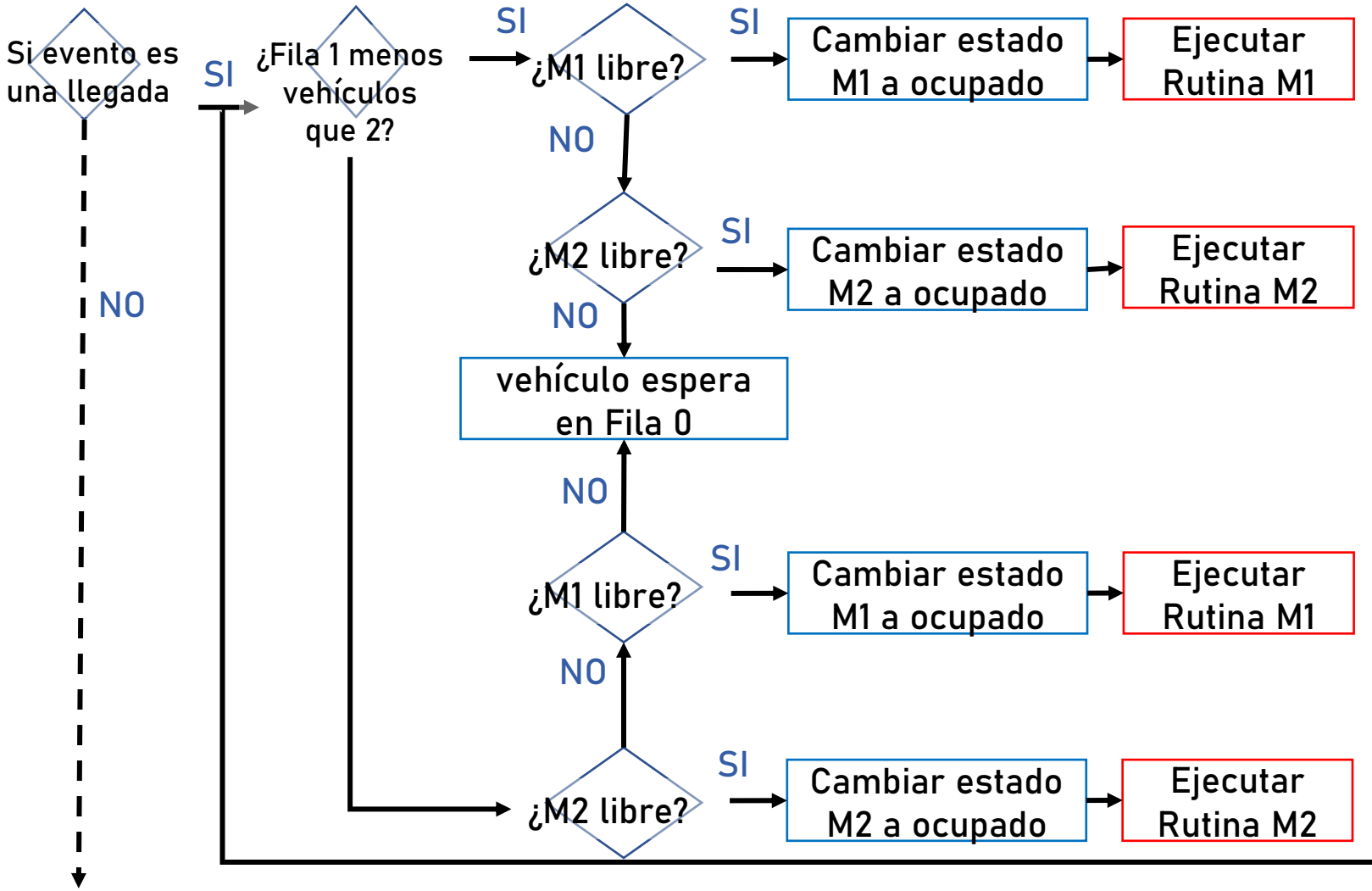


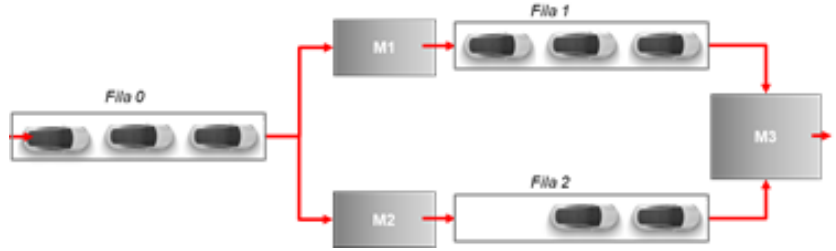
Gestión de eventos sampleables

```
while True:
    # Sacar evento de la fila de eventos:
    nuevo_evento = recuperar_de_fila_simulador()
    t_global = nuevo_evento[0]
    if t_global > t_corte:
        break
    tipo_evento = nuevo_evento[1]
    n_producto = nuevo_evento[2]
```

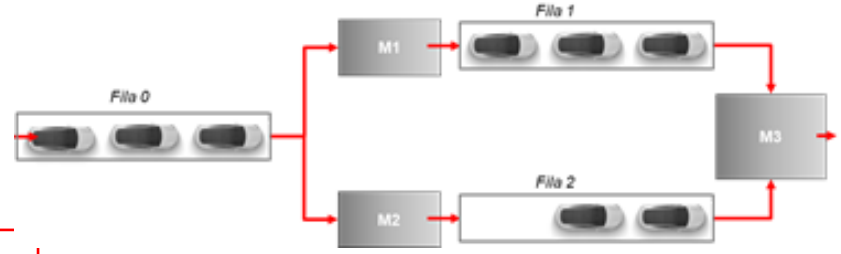
¡Recordar que estamos de ahora en más en un while!

Lógica al leer “llegada”





Rutinas M1, M2, M3



Ejecutar Rutina M1

Samplear tiempo de procesamiento M1

Crear evento: salida del vehículo de M1

Ejecutar Rutina M2

Samplear tiempo de procesamiento M2

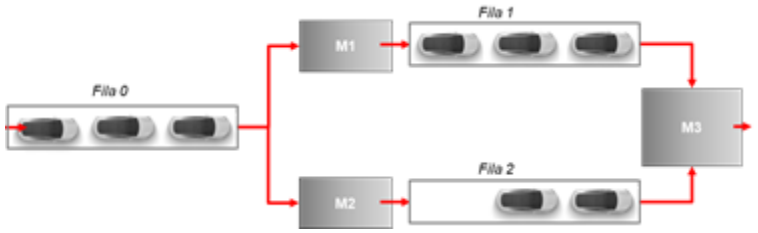
Crear evento: salida del vehículo de M2

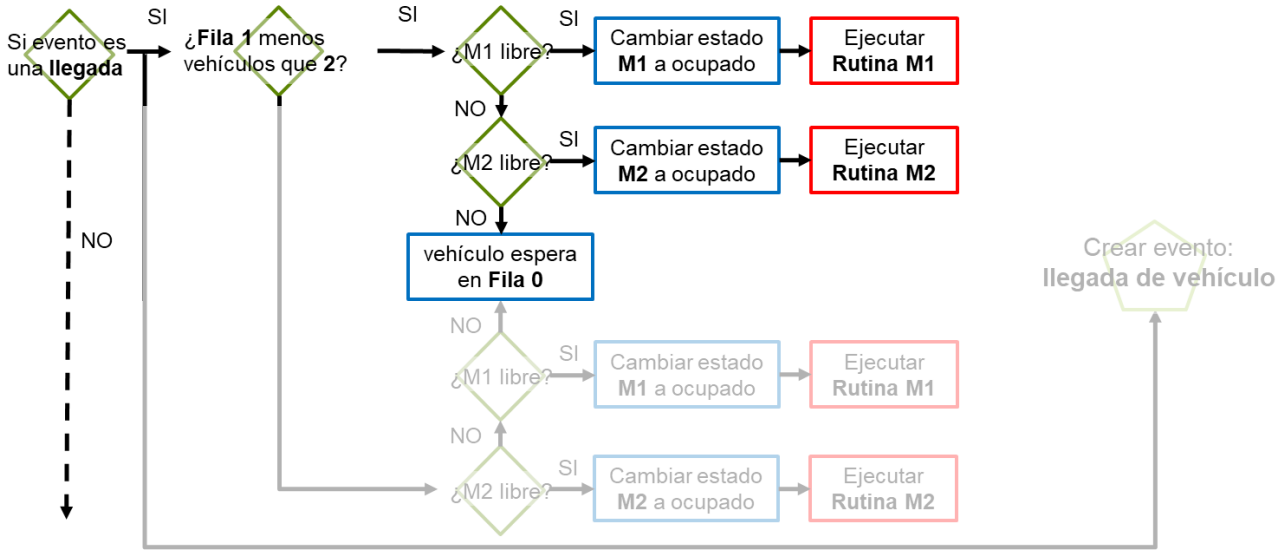
Ejecutar Rutina M3

Samplear tiempo de procesamiento M3

Crear evento: salida del vehículo de M3

Lógica al leer “llegada”

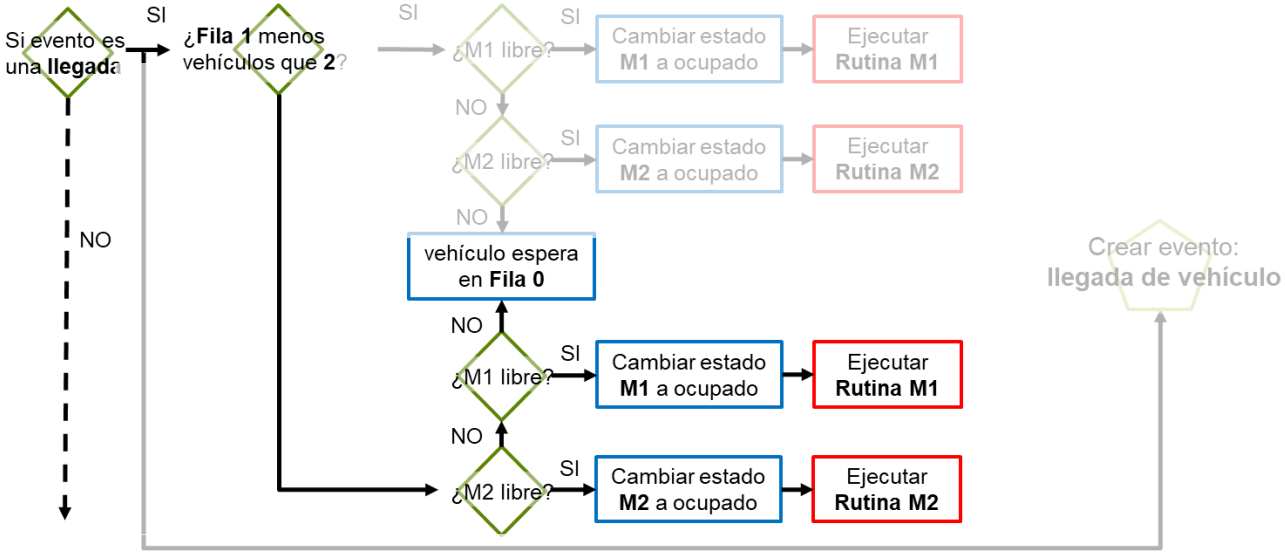


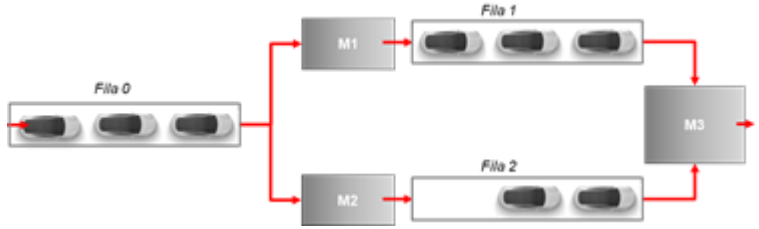



```
################ EVENTO: llegada de un vehículo ################
 if tipo_evento == 'nueva_unidad':
     if len(fila1) < len(fila2):
         if maquina_libre['m1']:
             # Se elige máquina 1:
             maquina_libre['m1'] = False
             t_salida = generar_salida_m1(t_global)
             ingresar_a_fila_simulador(t_salida, 'm1_out', n_producto)

         elif maquina_libre['m2']:
             # Se elige máquina 2:
             maquina_libre['m2'] = False
             t_salida = generar_salida_m2(t_global)
             ingresar_a_fila_simulador(t_salida, 'm2_out', n_producto)
         else:
             almacenar_producto_en_fila0(n_producto)
```

¡Recordar que estamos de ahora en más en un while!

Lógica al leer “llegada”

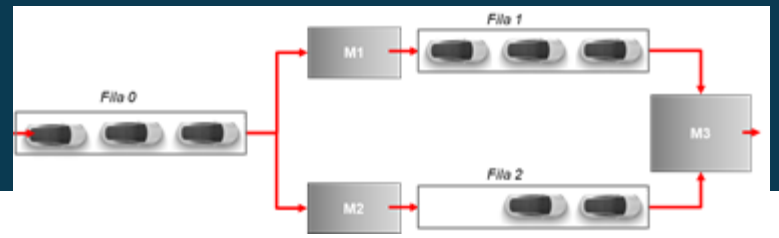


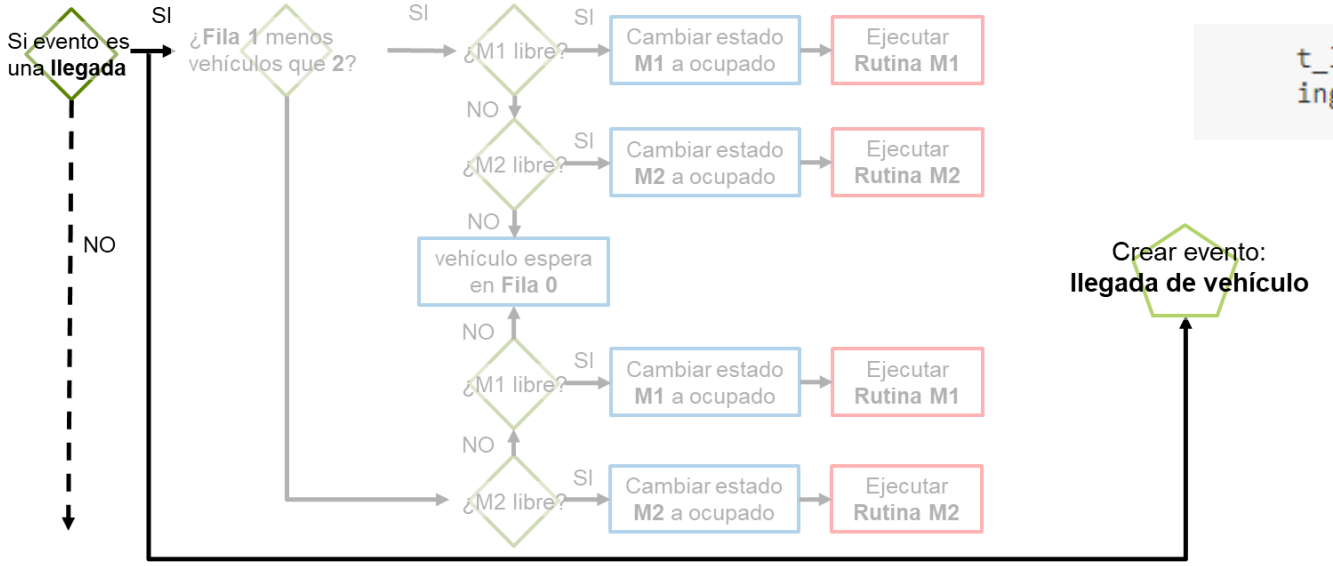



```
        else:
            if maquina_libre['m2']:
                # Se elige máquina 2:
                maquina_libre['m2'] = False
                t_salida = generar_salida_m2(t_global)
                ingresar_a_fila_simulador(t_salida, 'm2_out', n_producto)
            elif maquina_libre['m1']:
                # Se elige máquina 1:
                maquina_libre['m1'] = False
                t_salida = generar_salida_m1(t_global)
                ingresar_a_fila_simulador(t_salida, 'm1_out', n_producto)
            else:
                almacenar_producto_en_fila0(n_producto)
```

¡Recordar que estamos de ahora en más en un while!

Lógica al leer “llegada”

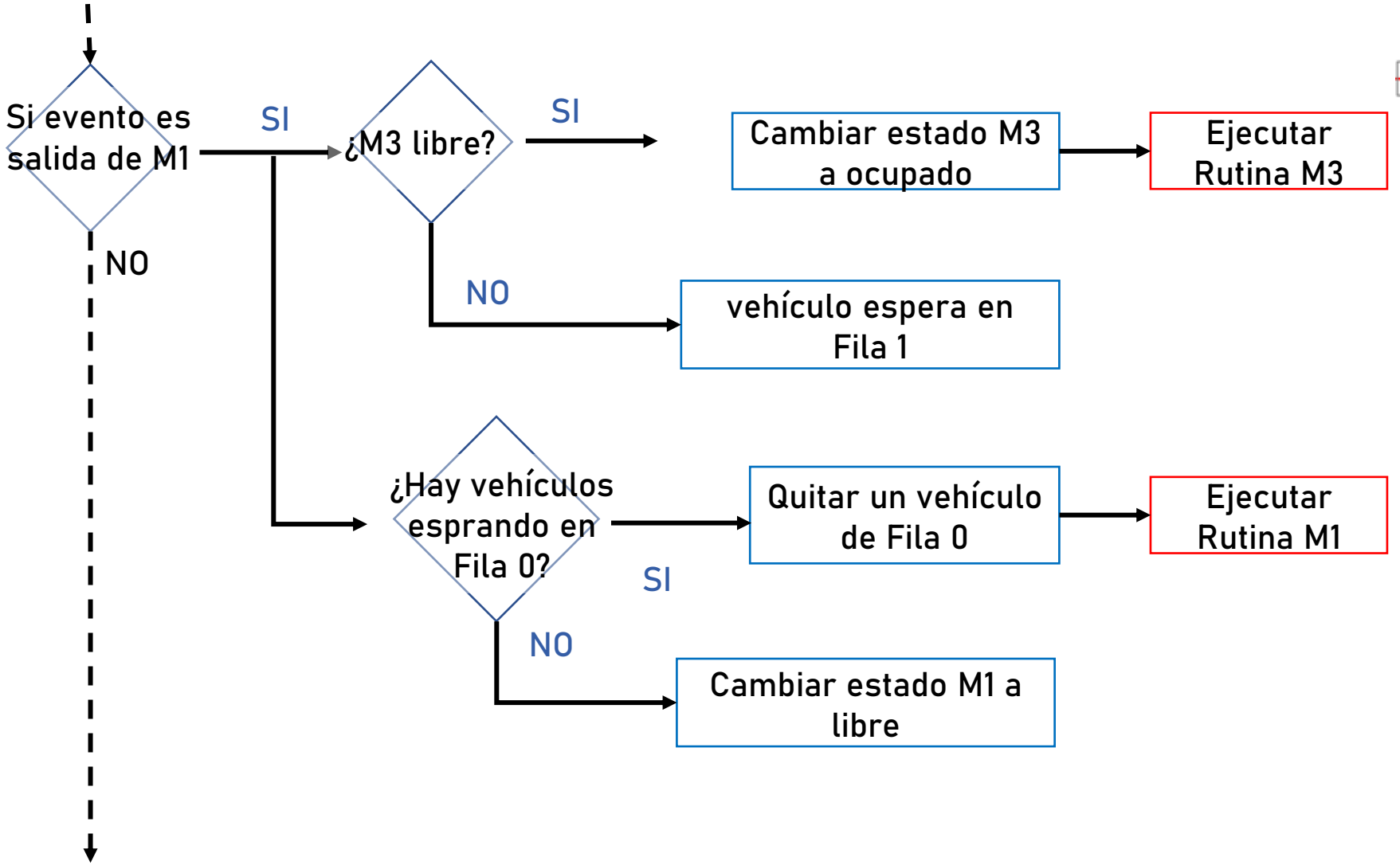


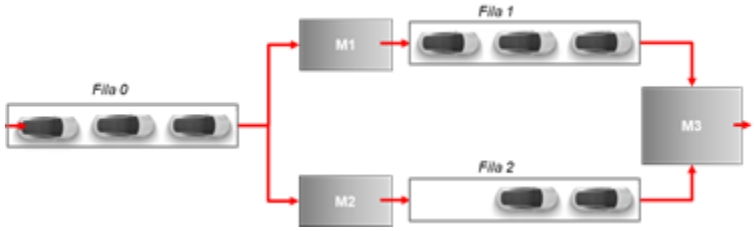



```
t_llegada, n_nuevo_producto = generar_llegada(t_global, gen_codigo_productos)
ingresar_a_fila_simulador(t_llegada, 'nueva_unidad', n_nuevo_producto)
```

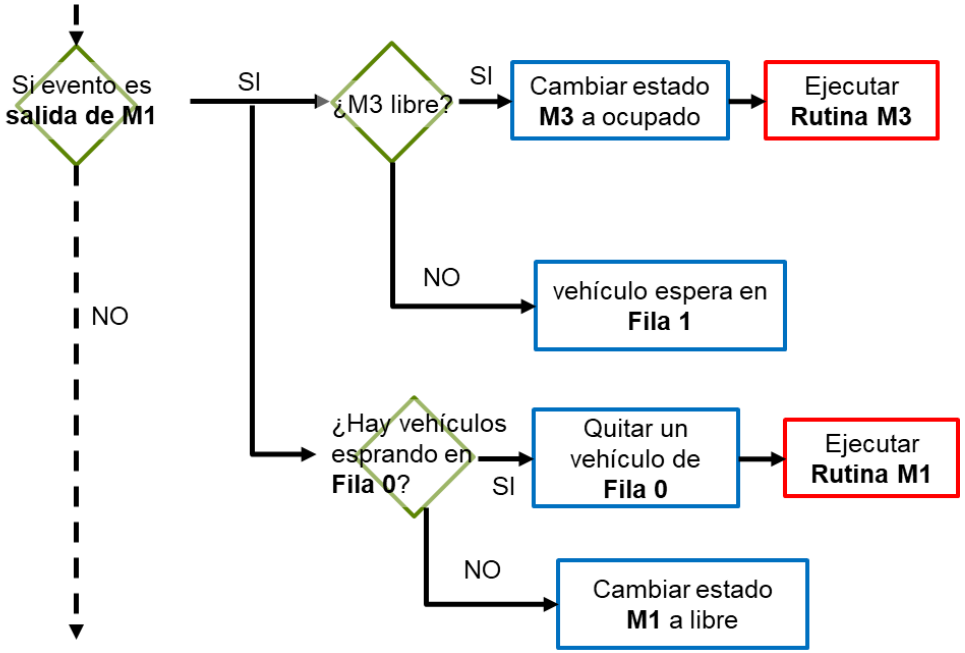
¡Recordar que estamos de ahora en más en un while!

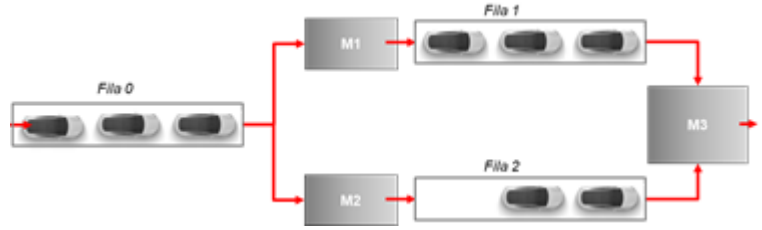
Lógica al leer “salida de M1”





Lógica al leer “salida de M1”

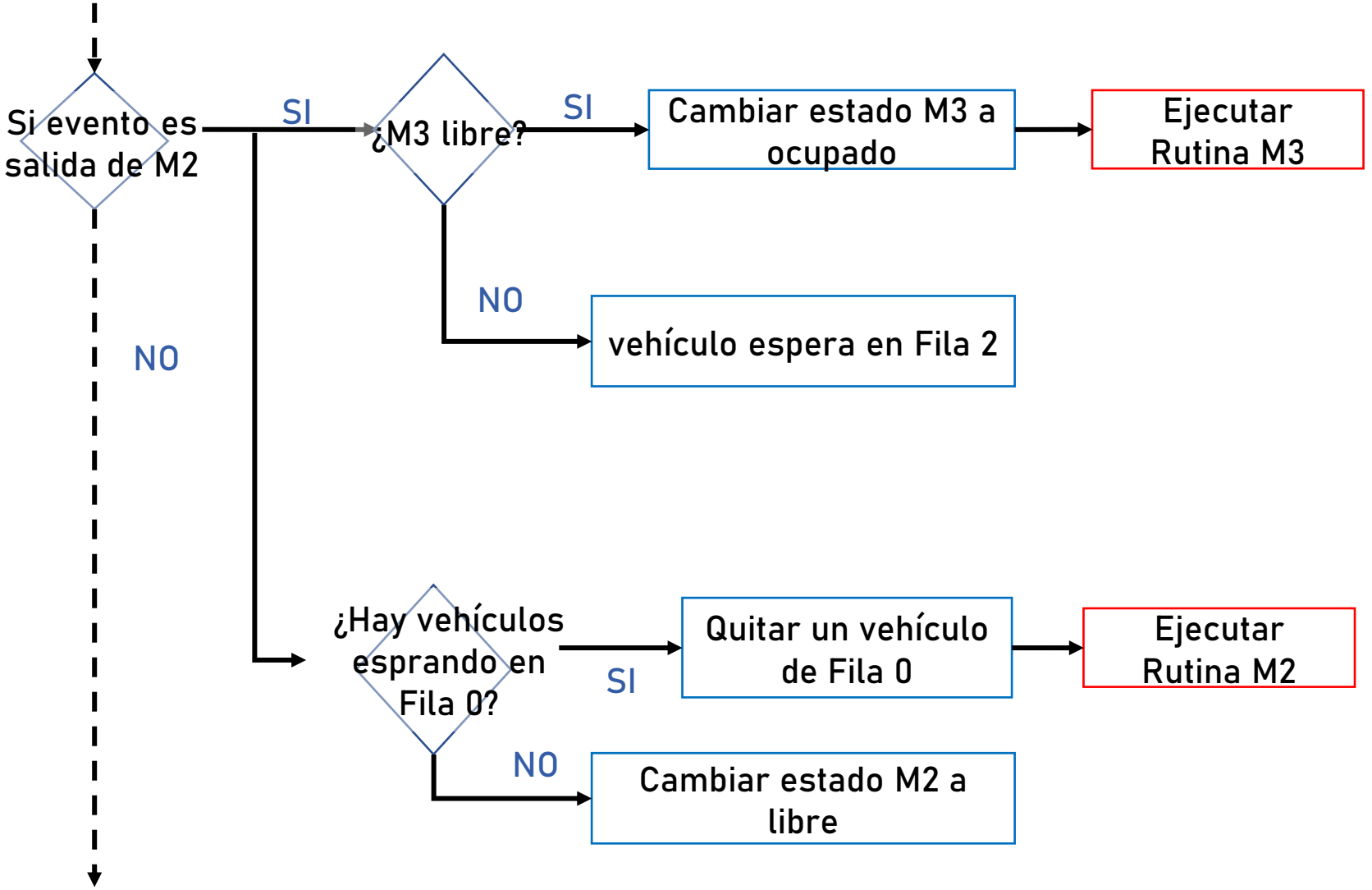


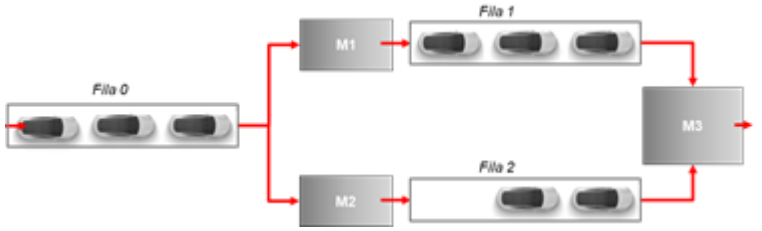



```python
################ EVENTO: salida de un vehículo de máquina 1 #################
if tipo_evento == 'm1_out':
    # definir destino de salida:
    if maquina_libre['m3']:
        t_salida = generar_salida_m3(t_global)
        ingresar_a_fila_simulador(t_salida, 'm3_out', n_producto)
        maquina_libre['m3'] = False
    else:
        almacenar_producto_en_fila1(n_producto)

    # ingresar nuevo producto:
    if len(fila0) > 0:
        n_producto = obtener_producto_de_fila0()
        t_salida = generar_salida_m1(t_global)
        ingresar_a_fila_simulador(t_salida, 'm1_out', n_producto)
    else:
        maquina_libre['m1'] = True
```

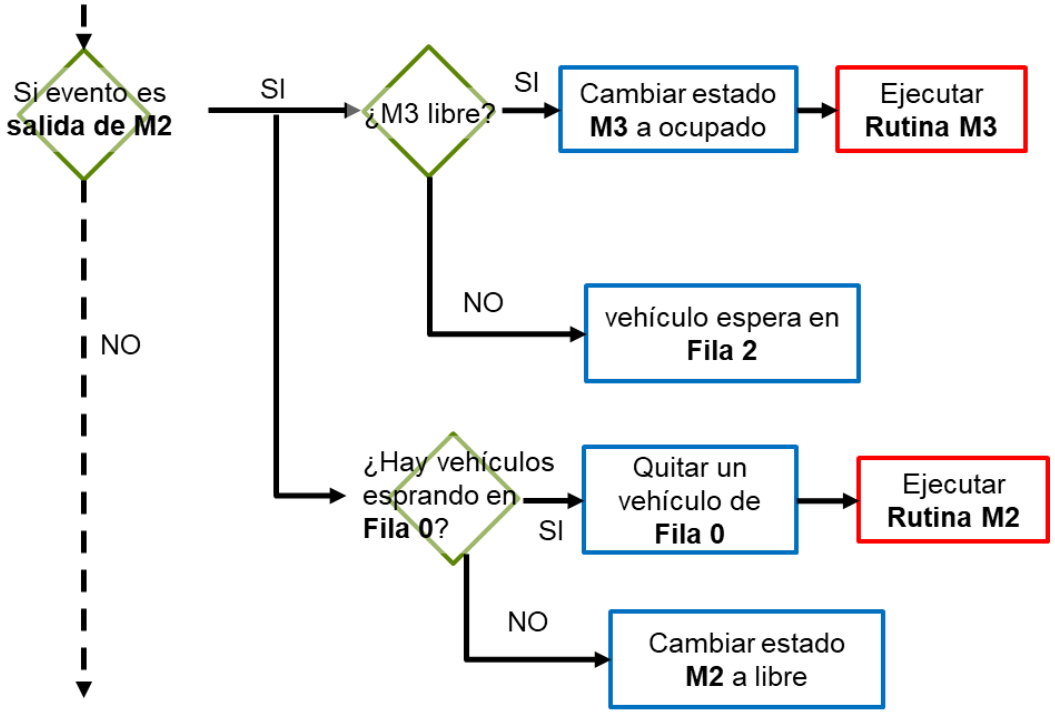
¡Recordar que estamos de ahora en más en un while!

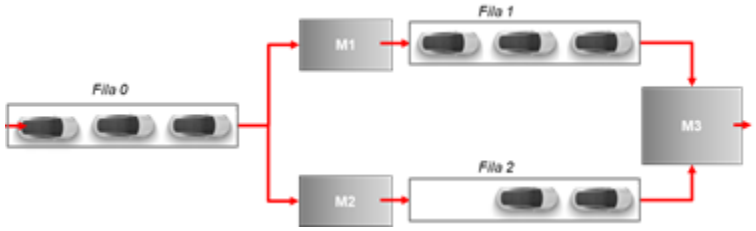
Lógica al leer “salida de M2”





Lógica al leer “salida de M2”

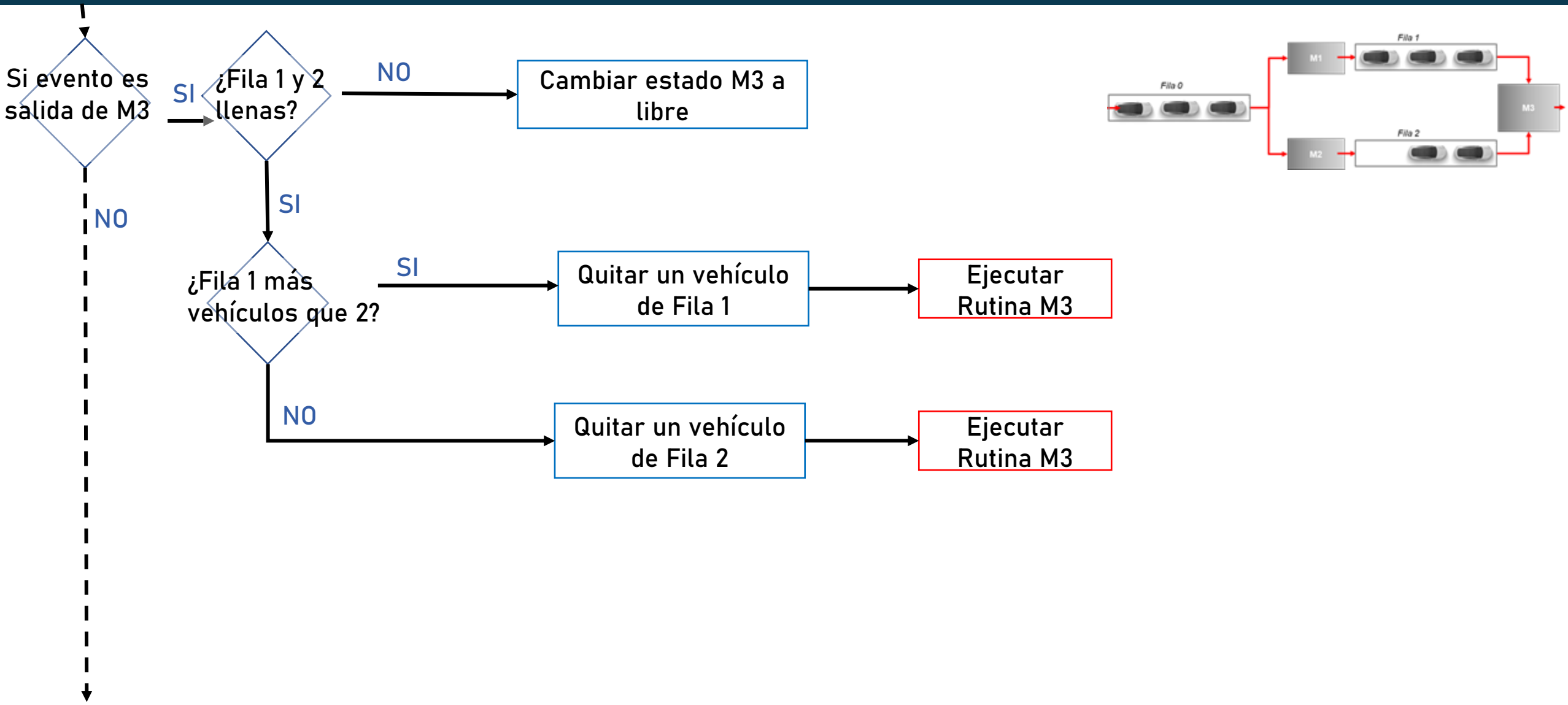




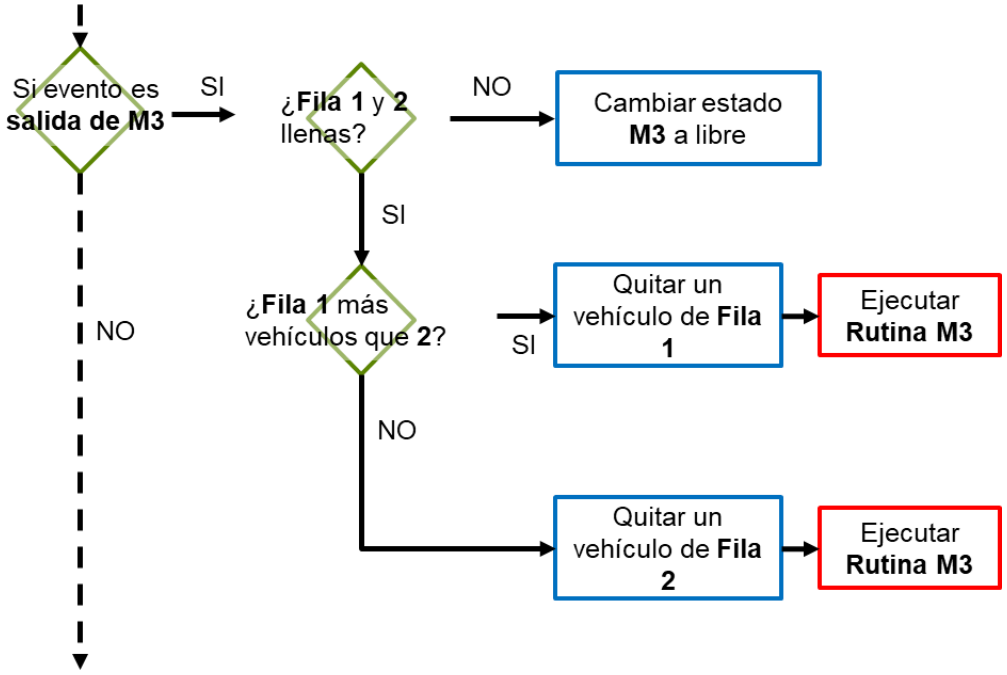

```
################ EVENTO: salida de un vehículo de máquina 2 ################
if tipo_evento == 'm2_out':
    # definir destino de salida:
    if maquina_libre['m3']:
        t_salida = generar_salida_m3(t_global)
        ingresar_a_fila_simulador(t_salida, 'm3_out', n_producto)
        maquina_libre['m3'] = False
    else:
        almacenar_producto_en_fila2(n_producto)

    # ingresar nuevo producto:
    if len(fila0) > 0:
        n_producto = obtener_producto_de_fila0()
        t_salida = generar_salida_m2(t_global)
        ingresar_a_fila_simulador(t_salida, 'm2_out', n_producto)
    else:
        maquina_libre['m2'] = True
```

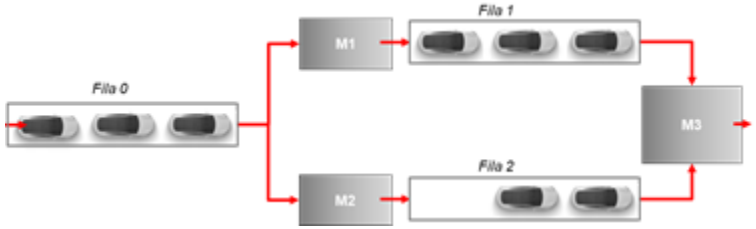
¡Recordar que estamos de ahora en más en un while!

Lógica al leer “salida de M3”



Lógica al leer “salida de M3”






```
################ EVENTO: salida de un vehículo de máquina 3 ################
if tipo_evento == 'm3_out':
    # ingresar nuevo producto:
    if len(fila1) > len(fila2):
        n_producto = obtener_producto_de_fila1()
        t_salida = generar_salida_m3(t_global)
        ingresar_a_fila_simulador(t_salida, 'm3_out', n_producto)
    elif len(fila2) > 0:
        n_producto = obtener_producto_de_fila2()
        t_salida = generar_salida_m3(t_global)
        ingresar_a_fila_simulador(t_salida, 'm3_out', n_producto)
    else:
        maquina_libre['m3'] = True
```

¡Recordar que estamos de ahora en más en un while!

Resultados

El log de eventos guarda en orden el historial de eventos generados en el calendario.

######### Log de eventos ###############

	tiempo	evento	producto
0	0.015994	nueva_unidad	0
1	0.064310	nueva_unidad	1
2	0.101958	m1_out	1
3	0.116578	m2_out	0
4	0.133281	m3_out	1
5	0.133383	nueva_unidad	2
6	0.149710	nueva_unidad	3
7	0.160706	nueva_unidad	4
8	0.162255	m3_out	0
9	0.186807	m1_out	3
10	0.201683	nueva_unidad	5
11	0.216906	m3_out	3
12	0.217765	m2_out	2
13	0.226138	m1_out	4
14	0.228058	nueva_unidad	6
15	0.248311	m3_out	2
16	0.260331	nueva_unidad	7
17	0.272992	m1_out	6
18	0.281791	m3_out	4

producto: 0

	tiempo	evento
0	0.015994	nueva_unidad
3	0.116578	m2_out
8	0.162255	m3_out

producto: 1

	tiempo	evento
1	0.064310	nueva_unidad
2	0.101958	m1_out
4	0.133281	m3_out

producto: 2

	tiempo	evento
5	0.133383	nueva_unidad
12	0.217765	m2_out
15	0.248311	m3_out

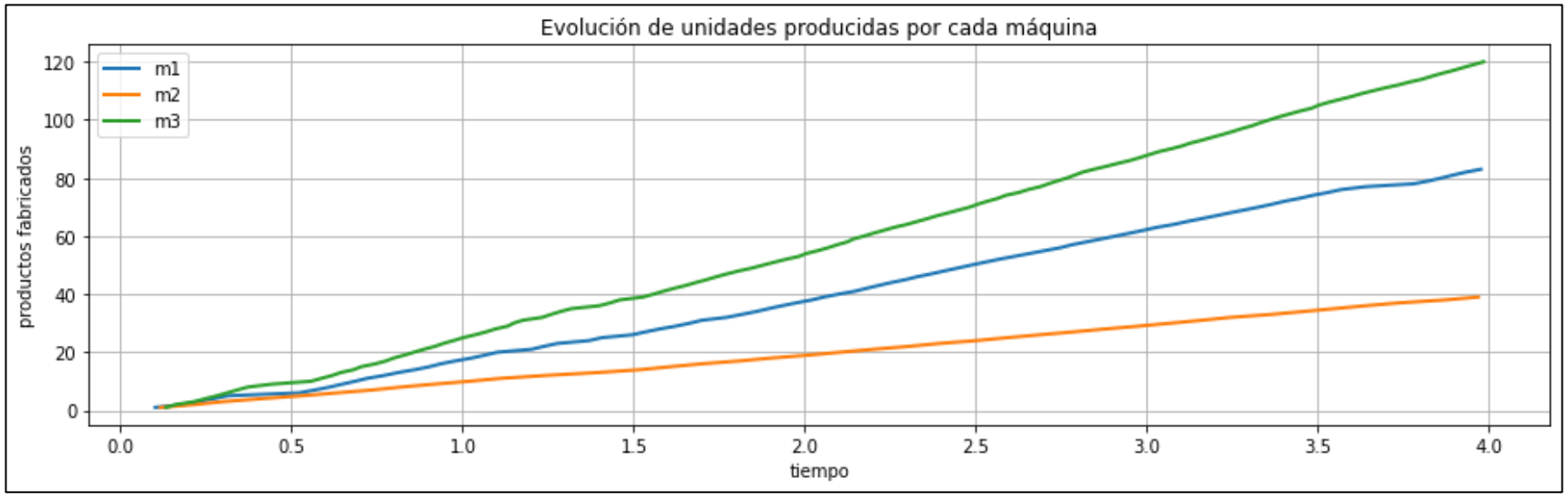
producto: 3

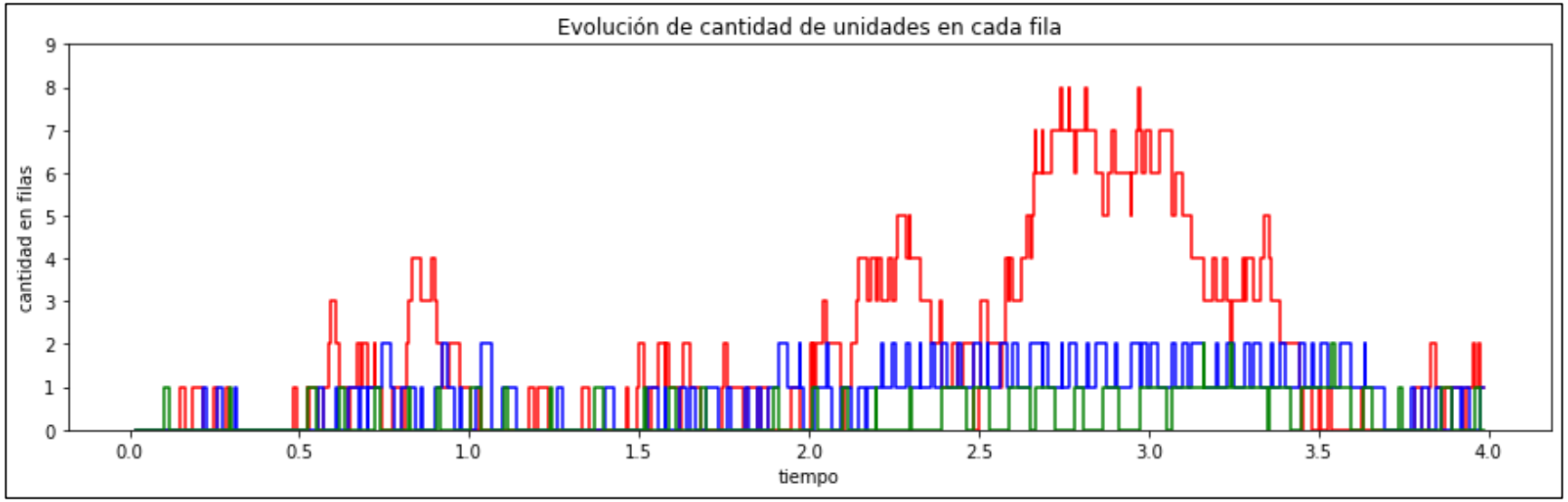
	tiempo	evento
6	0.149710	nueva_unidad
9	0.186807	m1_out
11	0.216906	m3_out

producto: 4

	tiempo	evento
7	0.160706	nueva_unidad
13	0.226138	m1_out
18	0.281791	m3_out

Resultados: evolución de unidades producidas





Lo siento, pasamos del infierno de fabricación al infierno de la logística; pero este problema es más manejable. Estamos teniendo un rápido progreso. Debería estar resuelto cuanto antes.