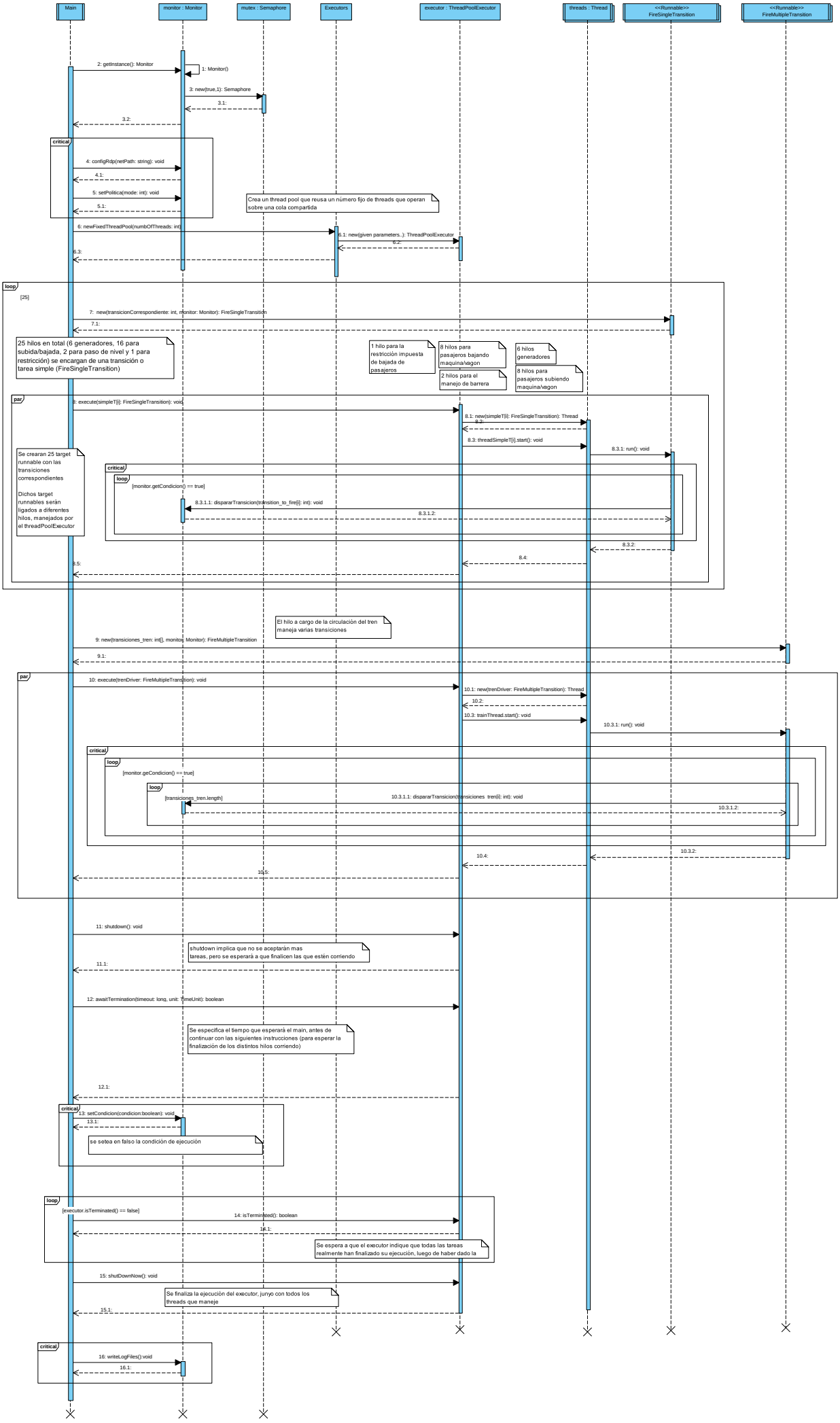
Main
monitor : Monitor
mutex : Semaphore
Executors
executor : ThreadPoolExecutor
threads : Thread
<<Runnable>>
FireSingleTransition
<<Runnable>>
FireMultipleTransition
2: getInstance(): Monitor
1: Monitor()
3: new(true,1): Semaphore
3.1:
3.2:
critical
4: configRdp(netPath: string): void
4.1:
5: setPolitica(mode: int): void
5.1:
Crea un thread pool que reusa un número fijo de threads que operan sobre una cola compartida
6: newFixedThreadPool(numbOfThreads: int)
6.1: new(given parameters..): ThreadPoolExecutor
6.2:
6.3:
loop
[25]
7: new(transicionCorrespondiente: int, monitor: Monitor): FireSingleTransition
7.1:
25 hilos en total (6 generadores, 16 para subida/bajada, 2 para paso de nivel y 1 para restricción) se encargan de una transición o tarea simple (FireSingleTransition)
1 hilo para la restricción impuesta de bajada de pasajeros
8 hilos para pasajeros bajando maquina/vagon
6 hilos generadores
2 hilos para el manejo de barrera
8 hilos para pasajeros subiendo maquina/vagon
par
8: execute(simpleT[i]: FireSingleTransition): void
8.1: new(simpleT[i]: FireSingleTransition): Thread
8.2:
8.3: threadSimpleT[i].start(): void
8.3.1: run(): void
Se crearan 25 target runnable con las transiciones correspondientes
Dichos target runnables serán ligados a diferentes hilos, manejados por el threadPoolExecutor
critical
loop
[monitor.getCondicion() == true]
8.3.1.1: dispararTransicion(transition_to_fire[i]: int): void
8.3.1.2:
8.3.2:
8.4:
8.5:
El hilo a cargo de la circulación del tren maneja varias transiciones
9: new(transiciones_tren: int[], monitor: Monitor): FireMultipleTransition
9.1:
par
10: execute(trenDriver: FireMultipleTransition): void
10.1: new(trenDriver: FireMultipleTransition): Thread
10.2:
10.3: trainThread.start(): void
10.3.1: run(): void
critical
loop
[monitor.getCondicion() == true]
loop
[transiciones_tren.length]
10.3.1.1: dispararTransicion(transiciones_tren[i]: int): void
10.3.1.2:
10.3.2:
10.4:
10.5:
11: shutdown(): void
shutdown implica que no se aceptarán mas tareas, pero se esperará a que finalicen las que estén corriendo
11.1:
12: awaitTermination(timeout: long, unit: TimeUnit): boolean
Se especifica el tiempo que esperará el main, antes de continuar con las siguientes instrucciones (para esperar la finalización de los distintos hilos corriendo)
12.1:
critical
13: setCondicion(condicion:boolean): void
13.1:
se setea en falso la condición de ejecución
loop
[executor.isTerminated() == false]
14: isTerminated(): boolean
14.1:
Se espera a que el executor indique que todas las tareas realmente han finalizado su ejecución, luego de haber dado la
15: shutDownNow(): void
Se finaliza la ejecución del executor, junyo con todos los threads que maneje
15.1:
critical
16: writeLogFiles():void
16.1: