

# SIMULADOR DE TRANSICIONES

FABIAN LEONARDO CORREA ROJAS  
DIEGO ALEJANDRO RODRIGUEZ LOPEZ  
JOSE LUIS ORTEGA CASTILLO

DOCUMENTACIÓN CASOS DE PRUEBA Y VALIDACIÓN

UNIVERSIDAD PEDAGÓGICA Y TECNOLÓGICA DE COLOMBIA  
ESCUELA DE INGENIERÍA  
INGENIERÍA DE SISTEMAS  
TUNJA  
2025

## TABLA DE CONTENIDO

pág

1.	INTRODUCCIÓN.....	3
2.	PRUEBAS .....	4
2.1	ALCANCE.....	4
2.2	ESTRATEGIA.....	4
2.3	ESTRUCTURA.....	5
2.4	ESCENARIOS .....	5
2.5	PRUEBAS DE MÉTRICAS Y EXPORTACIÓN DE CSV .....	6
2.6	EJECUCIÓN DE PRUEBAS.....	6
3.	CONCLUSIONES.....	8

## 1. INTRODUCCIÓN

En este documento se presentan las pruebas realizadas sobre el simulador de procesos, ejecutadas en distintos escenarios planteados. Se incluyen los resultados esperados y los observados, así como reportes detallados que permiten identificar y analizar los pasos y transiciones realizadas durante cada prueba.

## 2. PRUEBAS

### 2.1 ALCANCE

El alcance de las pruebas unitarias del proyecto abarca la verificación del funcionamiento de la máquina de estados de los procesos, asegurando que las transiciones válidas entre estados se realicen de manera correcta y que los intentos inválidos sean detectados y gestionados sin afectar la estabilidad del sistema. De igual forma, se incluyen las pruebas sobre las llamadas al sistema relacionadas con operaciones de I/O, la validación de las métricas generadas durante la ejecución de cada proceso y la correcta exportación de dicha información en reportes CSV. De esta manera, el conjunto de pruebas garantiza la cobertura de los componentes críticos del núcleo de simulación, dejando como complementario el análisis de escenarios más complejos como concurrencia de procesos o simulaciones de gran escala.

### 2.2 ESTRATEGIA

La estrategia de pruebas unitarias para el proyecto, se centra en validar de manera aislada el comportamiento de los módulos principales mediante escenarios representativos que permitan verificar tanto el flujo normal de ejecución como situaciones de error. Se emplea Jest como framework de pruebas, con validaciones orientadas a garantizar que las transiciones de estado respondan únicamente a eventos válidos, que las llamadas al sistema se registren correctamente y que las métricas generadas se reflejen de forma consistente en los reportes CSV. Los criterios de aceptación se basan en la correcta respuesta de las funciones bajo prueba, el rechazo controlado de transiciones inválidas y la generación confiable de reportes, asegurando que el entorno de simulación mantenga integridad y robustez en cada escenario validado.

## 2.3 ESTRUCTURA

- Archivo principal: processModel.test.js
- Agrupación por escenarios mediante bloques describe.
- Tests individuales con test() para cada comportamiento específico.
- Validaciones realizadas con expect de Jest.

## 2.4 ESCENARIOS

Escenario	Descripción	Entrada / Acciones	Resultados Esperados	Observaciones
1	Proceso sin I/O	Admit → Assign CPU → Finish	Todas las transiciones retornan status: true; CSV generado con métricas correctas	Primer flujo básico sin interrupciones
2	Proceso con 1 operación de I/O	Admit → Assign CPU → I/O Request → I/O Completed → Assign CPU → Finish	Transiciones válidas retornan status: true; métricas incluyen tiempo en WAITING; CSV generado correctamente	Valida llamadas al sistema y manejo de I/O
3	Intento de transición inválida	Intentar transición NEW → TERMINATED directamente	Retorna status: false; estado actual permanece NEW; CSV generado	Valida manejo de errores y FSM robusta

## 2.5 PRUEBAS DE MÉTRICAS Y EXPORTACIÓN DE CSV

- Cada escenario genera un CSV con los siguientes campos:
  - PID, Priority, ProgramCounter, Registers
  - State, Time(ms), Transitions, Syscalls
- Los archivos se guardan en la carpeta ./reports con nombres seguros y únicos por timestamp.
- Validación de consistencia de métricas con las acciones realizadas.

## 2.6 EJECUCIÓN DE PRUEBAS

La ejecución puede ser realizada mediante el comando: `npm test`. Dando por resultado en consola lo mostrado en la siguiente imagen.

```
\State-Transition-Simulator> npm
test
> state_transitions_simulator@0.0.0 test
> node --experimental-vm-modules node_modules/jest/bin/jest.js

console.log
  CSV generated: report_Process_without_IO_df98da_2025-09-07T23-14-20-163Z.csv

    at Object.log (src/core/transitions.test.js:42:13)

console.log
  CSV generated: report_Process_with_1_IO_e07d29_2025-09-07T23-14-20-209Z.csv

    at Object.log (src/core/transitions.test.js:57:13)

console.log
  CSV generated: report_Invalid_transition_attempt_191e2c_2025-09-07T23-14-20-223Z.csv

    at Object.log (src/core/transitions.test.js:68:13)

(node:8220) ExperimentalWarning: VM Modules is an experimental feature and might change at any time
(Use `node --trace-warnings ...` to show where the warning was created)
PASS src/core/transitions.test.js
  Process FSM Scenarios with Detailed Metrics
    ✓ Scenario 1: Process without I/O (86 ms)
    ✓ Scenario 2: Process with 1 I/O operation (12 ms)
    ✓ Scenario 3: Invalid transition attempt (18 ms)

Test Suites: 1 passed, 1 total
Tests:      3 passed, 3 total
Snapshots:  0 total
```

Figura 1. Resultado de la ejecución de pruebas unitarias con Jest.

```
PID,Priority,ProgramCounter,Registers
8070ce,Medium,6,"AX:25|BX:47|CX:60|DX:4"

State,Time(ms),Transitions,Syscalls
New,0,1,0
Ready,1,2,0
Running,0,2,0
Waiting,0,1,0
Terminated,1,1,0
```

Figura 2. CSV resultante de las pruebas.

### 3. CONCLUSIONES

Las pruebas unitarias realizadas en el proyecto, demuestran que la máquina de estados implementada para los procesos se comporta de acuerdo con lo esperado en los escenarios definidos, validando correctamente las transiciones válidas, rechazando las inválidas y generando métricas consistentes en los reportes CSV. Los resultados evidencian que el sistema es estable frente a los flujos básicos de ejecución y al manejo de operaciones de I/O, y que los archivos exportados contienen información detallada y coherente con las acciones ejecutadas durante la simulación.