Laboratorio 7: Simulación del Sistema de Torniquetes con SimPy

Profesores:
Carlos Andrés Lozano,
Germán Adolfo Montoya,

Profesor de Laboratorio:
Juan Andrés Mendez

15 de mayo de 2025

1 Objetivo

El objetivo de este laboratorio es que los estudiantes comprendan y apliquen los conceptos fundamentales de la simulación de eventos discretos utilizando la biblioteca SimPy de Python, mediante el modelado y optimización de un sistema de entradas y salidas con torniquetes en el edificio Santo Domingo (SD). Al finalizar, los estudiantes serán capaces de:

- Analizar datos históricos para modelar patrones de entrada y salida de personas
- Diseñar e implementar simulaciones para sistemas de flujo de personas
- Evaluar el impacto de diferentes escenarios y configuraciones del sistema
- Proponer y validar estrategias de optimización basadas en resultados de simulación
- Estimar el comportamiento futuro del sistema ante incrementos en la demanda

2 Fundamentación Teórica

2.1 Simulación de Eventos Discretos

La simulación de eventos discretos (DES, por sus siglas en inglés) es una técnica de modelado que permite analizar el comportamiento de sistemas complejos a través del tiempo. A diferencia de la simulación continua, en la cual las variables de estado cambian continuamente, en la simulación de eventos discretos los cambios ocurren en instantes específicos de tiempo, denominados .evento".

Características Principales de la Simulación de Eventos Discretos

- **Eventos**: Ocurrencias instantáneas que pueden cambiar el estado del sistema (llegada de personas, salida de personas).
- **Entidades**: Objetos que fluyen a través del sistema (estudiantes, profesores, personal administrativo).
- **Recursos**: Elementos que proveen servicio a las entidades (torniquetes).
- **Procesos**: Secuencia de eventos que experimentan las entidades (preparación de carnet, lectura, paso por torniquete).
- **Colas**: Lugares donde las entidades esperan cuando los recursos no están disponibles (filas de espera).
- **Reloj de simulación**: Mecanismo para rastrear el tiempo simulado (horas/días de la semana).

2.2 SimPy: Framework para Simulación en Python

SimPy es una biblioteca de simulación de eventos discretos basada en procesos para Python. Proporciona los componentes básicos para modelar procesos activos que pueden:

- Crear instancias de otros procesos
- Comunicarse con otros procesos a través de eventos y recursos compartidos
- Consumir tiempo simulado
- Esperar por la ocurrencia de eventos

3 Definición del Problema

La Universidad de los Andes ha mantenido un número constante de estudiantes durante los últimos 7 años. Sin embargo, debido a la adición de nuevos edificios y diversas tendencias en el país, la Dirección de Planeación de la universidad estima que para el año 2025 la población estudiantil crecerá en un 28 %. Esto ha generado preocupación entre algunos miembros sobre si nuestra infraestructura actual puede manejar tal incremento de estudiantes. En particular, hay inquietudes respecto al sistema de torniquetes.

Para abordar estas inquietudes, la administración decidió acercarse a los estudiantes de ingeniería de sistemas que están cursando **MOS**. Se les proporcionó acceso a un conjunto de datos de un semestre pasado que contiene todas las entradas y salidas del edificio Santo Domingo (SD), con el propósito de realizar una prueba de concepto sobre cómo se podría implementar la simulación de un solo edificio antes de simular toda la universidad.

En el caso del edificio SD, se cuenta con 6 torniquetes ajustados a una única dirección (3 para entrar y 3 para salir). Los datos muestran el flujo semanal (de lunes a sábado) de estudiantes, profesores y personal administrativo que entran y salen del edificio. El objetivo de este estudio es diseñar con gran fidelidad y detalle el comportamiento de la entrada y salida de personas en este edificio.

Para ello, se consideran las siguientes especificaciones: el tiempo para pasar por un torniquete es de 5 segundos, el tiempo necesario para tener el carnet en mano y estar listo para entrar es de unos 10 segundos, y el tiempo de lectura de QR, que es significativamente más alto, es de 20 segundos. Además, hay 2 torniquetes que funcionan con QR (uno de entrada y otro de salida). Es importante notar que estos tiempos tienen un comportamiento aleatorio, es decir, nunca se demoran exactamente 5, 10 o 20 segundos, sino que varían dentro de un rango de ±2 segundos de manera aleatoria.

Actividad	**Tiempo (segundos)**
Pasar por el torniquete	5 ± 2
Preparar el carnet para entrar	10 ± 2
Lectura de QR	20 ± 2

Cuadro 1: Tiempos necesarios para pasar por el torniquete

4 Tareas

4.1 Diseño y Justificación del Modelo

Objetivo: Diseñar y justificar un modelo de simulación para el sistema de torniquetes del edificio Santo Domingo (SD).

Descripción:

- **Definir Componentes y Flujo de Proceso:**
  - **Componentes del Sistema:** Identificar y describir los componentes clave del sistema de torniquetes, incluyendo los torniquetes, los lectores de QR, las tarjetas de identificación y el flujo de personas (estudiantes).
  - **Flujo de Proceso:** Describir el flujo de proceso completo desde la llegada de una persona al torniquete hasta su entrada o salida del edificio. Detallar las etapas del proceso, como la preparación de la tarjeta, el escaneo de QR y el paso a través del torniquete.
- **Tipos de Torniquetes:**
  - **Torniquetes Convencionales:** Describir los torniquetes convencionales y su tiempo de procesamiento (5 segundos ± 2 segundos).
  - **Torniquetes con Lector de QR:** Describir los torniquetes equipados con lectores de QR y su tiempo de procesamiento (20 segundos ± 2 segundos).
  - **Dirección de Flujo:** Detallar la configuración actual de los torniquetes, con 3 torniquetes dedicados a la entrada y 3 a la salida, y la inclusión de 1 torniquete de entrada y 1 de salida con lectores de QR.
- **Justificar Decisiones de Diseño:**
  - **Basado en el Análisis de Datos:** Justificar las decisiones de diseño utilizando los resultados del análisis de datos de la sección anterior. Explicar cómo los patrones de entrada y salida, las horas pico y las tasas de uso influyen en el diseño del modelo.

- **Variabilidad y Aleatoriedad:** Incluir la variabilidad en los tiempos de procesamiento para reflejar condiciones más realistas, tal como se indicó en la definición del problema.

Entrega:

- **Documento de Diseño del Modelo:** Un documento detallado que incluya:
  - Diagramas que muestren los componentes del sistema y el flujo de proceso.
  - Descripciones de los diferentes tipos de torniquetes y sus tiempos de procesamiento.
  - Justificaciones claras y bien fundamentadas para las decisiones de diseño, basadas en el análisis de datos y las proyecciones de incremento en la población estudiantil.

4.2 Creación de la Simulación

Objetivo: Implementar el modelo de simulación utilizando SimPy.
Descripción:

- **Codificar el Proceso Básico del Torniquete:**
  - Desarrollar el código en Python utilizando la biblioteca SimPy para modelar el proceso básico de los torniquetes.
  - Asegurarse de que el código capture todos los componentes y flujos de proceso descritos en el modelo de diseño, incluyendo la llegada y la salida de personas, la preparación del carnet y el paso por el torniquete.
- **Integrar Variabilidad:**
  - Incorporar variabilidad en los tiempos de procesamiento para reflejar condiciones realistas. Implementar distribuciones de tiempo aleatorio para cada actividad (p. ej., pasar por el torniquete, preparar el carnet y leer el QR) utilizando distribuciones normales con los parámetros especificados (media y desviación estándar).
- **Asegurar la Precisión:**
  - Validar el modelo simulando situaciones conocidas y comparando los resultados con los datos reales del edificio Santo Domingo.
  - Revisar y refinar el código para garantizar que la simulación refleje con precisión el comportamiento real del sistema.

Entrega:

- **Modelo de Simulación Funcional:** Un modelo de simulación funcional en SimPy que incluye:
  - Código fuente bien comentado que describe cada parte del proceso de simulación.

  - Scripts de prueba que validen la precisión del modelo en comparación con los datos reales.

- **Documentación del Código:** Un documento que describa el diseño del código, las decisiones tomadas durante la implementación y cómo se integra la variabilidad.

5 Análisis de Escenarios y Optimización

Objetivo: Explorar y optimizar el sistema bajo varios escenarios.
Descripción:

- **Escenario 1: Proyección de Crecimiento:**
  - Evaluar el impacto del proyectado crecimiento del 28 % en la población estudiantil para el año 2025.
  - Determinar si la configuración actual podrá manejar el incremento esperado.
  - Identificar las adaptaciones necesarias en términos de número y tipo de torniquetes.
- **Implementar Estrategias de Optimización:**
  - Identificar estrategias para optimizar el rendimiento del sistema basadas en los resultados de los escenarios anteriores.
  - Proponer e implementar mejoras, como la redistribución de torniquetes, la optimización de horarios de entrada/salida y la mejora de la tecnología utilizada.
  - Simular las estrategias propuestas y evaluar su efectividad en la reducción de los tiempos de espera y la mejora del flujo de personas.

Entrega:

- **Análisis de Escenarios:** Documente los resultados de cada análisis de escenario, incluyendo:
  - Gráficos y tablas que presenten los tiempos de espera y las mejoras logradas.
  - Comparaciones entre los diferentes escenarios simulados y el modelo original.
- **Código de Simulación para Escenarios:** Código fuente de SimPy utilizado para simular los diferentes escenarios, con comentarios detallados que expliquen los cambios realizados y las observaciones derivadas.
- **Recomendaciones de Optimización:** Un conjunto de recomendaciones prácticas y justificadas para optimizar el sistema de torniquetes basado en el análisis de escenarios, acompañado de simulaciones que demuestren su efectividad.
- **Matplotlib y Seaborn**: Para visualización de resultados.
- **Jupyter Notebooks**: Para desarrollo interactivo y presentación de resultados.

6 Guía para Abordar el Problema

A continuación, se presenta una serie de pasos sugeridos para abordar este problema de simulación:

6.1 Fase 1: Modelado de Llegadas y Salidas

1. **Extracción de Tasas de Llegada**:
   - Debido a la naturaleza claramente no homogénea de los datos (picos recurrentes en ciertas horas del día), no es adecuado utilizar una única distribución global para modelar todas las llegadas y las salidas.
   - Se recomienda utilizar un enfoque de tasa variable por intervalo (modelo no homogéneo), donde cada bloque de tiempo (ej. 15 minutos) tenga su propia tasa de llegada $\lambda(t)$.
   - Estas tasas pueden luego usarse para generar llegadas y salidas usando un proceso de Poisson no homogéneo (NHPP) o bien interpolar una curva de densidad empírica.
2. **Modelado de Comportamiento por Tipo de Usuario**:
   - Analizar si existen diferencias en los patrones de llegada entre estudiantes, profesores y administrativos
   - Determinar la proporción de usuarios que utilizan los torniquetes QR vs. torniquetes normales

6.2 Fase 2: Implementación de la Simulación en SimPy

1. **Definición de Clases y Recursos**:
   - Crear clase para los torniquetes (como recursos SimPy)
   - Definir diferentes tipos de torniquetes (entrada normal, salida normal, entrada QR, salida QR)
   - Implementar generadores de tiempos aleatorios para procesos
2. **Implementación de Procesos**:
   - Definir proceso de llegada de personas según tasas extraídas de los datos
   - Implementar proceso de preparación de carnet y paso por torniquete
   - Codificar lógica para selección de torniquete según tipo y disponibilidad
3. **Recopilación de Estadísticas**:
   - Implementar mecanismos para registrar tiempos de espera
   - Registrar utilización de torniquetes
   - Calcular longitudes de cola a través del tiempo

6.3 Fase 3: Validación y Verificación

1. **Comparación con Datos Reales**:
   - Ejecutar la simulación con parámetros derivados de los datos históricos
   - Comparar patrones de flujo simulado vs. real
   - Ajustar parámetros del modelo según sea necesario
2. **Análisis de Sensibilidad**:
   - Evaluar el efecto de pequeñas variaciones en parámetros clave
   - Identificar variables con mayor impacto en el rendimiento

6.4 Fase 4: Experimentación de Escenarios

1. **Diseño de Experimentos**:
   - Definir variables de respuesta (tiempos de espera, longitud de colas)
   - Establecer factores a modificar (número de torniquetes, velocidad de procesamiento)
   - Crear una matriz de experimentos
2. **Ejecución de Escenarios**:
   - Implementar el escenario propuesto en las tareas
   - Ejecutar múltiples réplicas para obtener estimaciones confiables
   - Documentar resultados con visualizaciones claras

6.5 Fase 5: Análisis y Recomendaciones

1. **Análisis Comparativo**:
   - Comparar métricas clave entre escenarios
   - Evaluar trade-offs entre diferentes configuraciones
   - Identificar la configuración óptima según objetivos definidos
2. **Elaboración de Recomendaciones**:
   - Formular recomendaciones claras y específicas
   - Sustentar cada recomendación con datos de la simulación
   - Proponer un plan de implementación gradual

7 Entregables y Criterios de Calificación

7.1 Entregables

1. **Código Fuente**:
   - Archivo Python (*.py) con la implementación completa de la simulación
   - Scripts para el análisis de datos y pre-procesamiento
   - Notebook Jupyter con visualizaciones y análisis de resultados
   - README con instrucciones claras para ejecutar el código
2. **Informe Técnico (PDF)**:
   - Descripción detallada del diseño del modelo
   - Metodología de validación y resultados
   - Análisis de escenarios experimentales
   - Conclusiones y recomendaciones fundamentadas
   - Limitaciones del modelo y trabajo futuro

Nota: Se puede entregar todo en unico archivo siempre y cuando sea un Jupyter Notebook.

7.2 Criterios de Calificación

Criterio	**Puntuación**
Diseño e implementación del modelo de simulación	45 %
Validación del modelo y análisis estadístico	15 %
Evaluación de escenarios y estrategias propuestas	20 %
Calidad del código y documentación	10 %
Visualizaciones y presentación de resultados	5 %
Conclusiones y recomendaciones fundamentadas	5 %

Cuadro 2: Criterios de calificación del laboratorio

8 Referencias

1. Banks, J., Carson, J. S., Nelson, B. L., & Nicol, D. M. (2014). Discrete-Event System Simulation (5th ed.). Pearson.
2. Law, A. M. (2015). Simulation Modeling and Analysis (5th ed.). McGraw-Hill.
3. Team SimPy. (2022). SimPy Documentation. Recuperado de https://simpy.readthedocs.io/
4. McKinney, W. (2017). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython (2nd ed.). O'Reilly Media.

5. Robinson, S. (2014). Simulation: The Practice of Model Development and Use (2nd ed.). Palgrave Macmillan.

6. Rossetti, M. D. (2015). Simulation Modeling and Arena (2nd ed.). Wiley.

7. Kelton, W. D., Sadowski, R. P., & Zupick, N. B. (2014). Simulation with Arena (6th ed.). McGraw-Hill.