TP 3 - SIMULACIÓN DE UN MODELO MM1 E INVENTARIO

Mateo Simón Arach
Leg. 51394
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
matearach@gmail.com

Francisca Gramaglia
Leg. 51424
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
franciscagramaglia714@gmail.com

Milton Rubén Borsato Gimenez
Leg. 51391
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
borsatomilton@gmail.com

José Ignacio Dayer
Leg. 51508
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
nachodayer@gmail.com

Valentin David Marchese
Leg. 51745
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
marchese52002@gmail.com

Mauricio Fiorini
Leg. 50475
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
mauriciofiorini911@gmail.com

Alan Romero
Leg. 49741
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
alanromero2773@gmail.com

June 18, 2025

ABSTRACT

Este informe presenta un análisis de simulación de dos modelos fundamentales: el modelo de colas M/M/1 y un modelo de inventario. Para el modelo MM1, se evalúan métricas como el promedio de clientes en el sistema y en cola, el tiempo promedio en el sistema y en cola, la utilización del servidor y probabilidades relacionadas con el tamaño de la cola y la denegación de servicio en colas finitas. Se varían las tasas de arribo del 25 porciento al 125 porciento respecto a la tasa de servicio, realizando al menos 10 corridas por experimento para garantizar la fiabilidad. El modelo de inventario analiza el costo de orden, costo de mantenimiento, costo de faltante y costo total, también con un mínimo de 10 corridas por experimento. Los resultados se comparan entre valores teóricos, simulaciones en Python e implementaciones en Anylogic. Además, facilitaremos la variación de parámetros para demostraciones en tiempo real en clase. Este análisis dual busca proporcionar una comprensión profunda del comportamiento del sistema bajo diferentes condiciones, validar la precisión de las simulaciones y poder sacar conclusiones a partir de ellas.

1 Introducción

La competencia y condiciones de mercado actual obliga a las empresas a estar en todos los detalles a la hora de vender un producto o un servicio. En el presente informe desarrollaremos y analizaremos dos modelos fundamentales en la gestión operativa de empresas: la simulación de teoría de colas y la simulación de control de inventarios, teniendo en cuenta distintos escenarios posibles que puedan presentarse. Ambas herramientas permiten representar de forma dinámica y realista el comportamiento de sistemas logísticos y de atención al cliente, con el fin de evaluar su rendimiento

bajo distintas condiciones operativas, analizar indicadores clave de desempeño y proponer mejoras basadas en los resultados obtenidos. Por un lado, la teoría de colas permite modelar situaciones en las que los clientes (o productos, solicitudes, etc.) deben esperar para ser atendidos. Por otro lado, el control de inventario se enfoca en mantener un equilibrio entre el nivel de stock disponible y los costos asociados, tales como los costos de almacenamiento, pedidos y faltantes.

2 Marco Teórico

2.1 Conceptos de interés

Sistema

Un sistema es un conjunto organizado de elementos (físicos o conceptuales) que interactúan entre sí para cumplir un objetivo común o llevar a cabo una función determinada. Está formado por entradas (inputs), procesos, salidas (outputs) y algunos sistemas cuentan con realimentación (feedback), utilizando este último para ajustar el comportamiento del sistema:

Tipos de sistema:

- Sistema Continuo: Un sistema continuo es aquel en el que las variables de entrada, salida y estado cambian de manera continua con el tiempo. Estos sistemas se encuentran comúnmente en fenómenos físicos o naturales
- Sistema Discreto: Un sistema discreto es aquel en el que las variables cambian solo en ciertos instantes definidos de tiempo. El tiempo en este caso no es continuo, sino que avanza en pasos, como pueden ser días, semanas meses. Este tipo de sistema es frecuente en procesos digitales, computacionales o en simulaciones que se evalúan por períodos.

Como represento un sistema: Un sistema puede ser representado a través de modelos físicos o modelos matemáticos. Los primeros son utilizados en sistemas de ingeniería o de administración. Mayormente los modelos son de carácter matemático, representando a un sistema en términos de lógica y relaciones cuantitativas que entonces son manipuladas para ver cómo el sistema reacciona al modelo matemático.

- Solución Analítica: Una solución analítica es una expresión matemática exacta que describe el comportamiento de un sistema a lo largo del tiempo. En otras palabras, es una fórmula cerrada que permite calcular el estado o la salida del sistema para cualquier instante de tiempo. Esta es utilizada ante modelos simples.
- Simulación: La simulación es una técnica que consiste en imitar el comportamiento de un sistema real mediante un modelo matemático o computacional, con el fin de analizar su desempeño bajo distintas condiciones y escenarios posibles. Se utiliza para reproducir el funcionamiento dinámico de un sistema cuando una solución exacta es difícil, costosa o incluso imposible de obtener. Una simulación puede ser:
  - *Determinística o Estocástica*: Una simulación determinista es aquella en la que no existe incertidumbre. Es decir, los resultados están completamente determinados por los datos de entrada y las condiciones iniciales. Por otro lado, una simulación estocástica incluye elementos aleatorios o probabilísticos. Esto significa que algunos valores, como la demanda, el tiempo de llegada o el tiempo de servicio, se generan al azar a partir de una distribución de probabilidad. La principal diferencia entre estas dos es que la Deterministica siempre arrojara el mismo valor para las mismas entradas, en cambio la Estocástica no.
  - *Simulación discreta vs. continua*: La simulación discreta representa sistemas que cambian su estado en momentos específicos. El tiempo se avanza en pasos o eventos definidos, y las variables toman valores separados. Por otro lado, la simulación continua modela sistemas donde las variables cambian constantemente a lo largo del tiempo, sin saltos.
  - *Simulación basada en eventos vs. en tiempo fijo*: Una simulación basada en eventos avanza el reloj del sistema de un evento a otro, es decir, solo "salta" en el tiempo cuando algo relevante ocurre (como la llegada de un cliente, el fin de un servicio, una falla, etc.). Es muy eficiente porque ignora los momentos donde no pasa nada. En cambio, una simulación en tiempo fijo evalúa el sistema en intervalos regulares de tiempo, como cada segundo, cada hora o cada día, aunque no ocurra ningún evento.

Componentes de un Modelo de Simulación:

- *Estado del sistema:* Es el conjunto de variables que describen cómo se encuentra el sistema en un instante determinado. en particular.

- *Reloj de Simulación:* Es una variable que representa el tiempo actual dentro de la simulación.
- *Lista de eventos:* Es una estructura donde se almacenan todos los eventos futuros, junto con el instante en que deben ocurrir.
- *Contadores estadísticos:* Son variables que se usan para recolectar información sobre el desempeño del sistema a lo largo del tiempo simulado.
- *Rutina de inicialización:* Es un bloque de instrucciones que se ejecuta al comienzo de la simulación para preparar el sistema en su estado inicial.
- *Rutina de tiempo:* Determina cuál es el próximo evento que debe ejecutarse y avanza el reloj de simulación hasta ese instante.
- *Rutina de evento:* Conjunto de instrucciones que actualizan el estado del sistema cada vez que ocurre un tipo específico de evento.
- *Librería de rutinas:* Es un conjunto de funciones utilizadas para generar datos aleatorios a partir de distribuciones probabilísticas que modelan el comportamiento del sistema.
- *Generador de reportes:* Es el módulo que calcula y presenta los resultados estadísticos del rendimiento del sistema al finalizar la simulación.
- *Programa principal:* Es el controlador general de la simulación. Coordina la ejecución de eventos, el avance del tiempo, la actualización del sistema y la generación de resultados finales.

Modelo de colas: Un modelo de colas es una representación matemática o computacional de un sistema donde entidades (como personas) llegan para recibir un servicio de uno o más servidores, pero si estos están ocupados, las entidades deben esperar en una cola llamada línea de espera hasta ser atendidas. Este tipo de modelo permite analizar y predecir el comportamiento del sistema bajo distintas condiciones, como tiempos de espera, longitud de la cola, utilización del servidor, entre otros. Son cuatro las características básicas que se deben utilizar para describir correctamente un sistema de colas:

- Patrón de llegada de los clientes
- Patrón de un servicio de los servidores
- Disciplina de cola
- Capacidad del sistema

El tiempo entre llegadas de los clientes y tiempos de servicio son variables aleatorias, esto significa que el sistema se encuentra condicionado por la probabilidad.

Parámetros de colas: Un sistema de colas puede representarse mediante la notación (I, J, s), donde I y J indican distribuciones probabilísticas (una para los tiempos entre llegadas y otra para los tiempos de servicio) y s representa la cantidad de servidores disponibles. Existe también una notación más detallada conocida como la notación extendida de Kendall, que se escribe como I/J/s/k/n/Z, donde cada parámetro significa:

- I: Tipo de distribución estadística para los tiempos entre llegadas.
- J: Tipo de distribución estadística para los tiempos de servicio.
- s: Cantidad de servidores en el sistema.
- k: Capacidad total del sistema (clientes en espera más los atendidos).
- n: Número total de usuarios en el sistema (en algunos contextos, una población finita).
- Z: Política de atención o forma en que se elige qué cliente será atendido (por ejemplo, FIFO).

Podemos utilizar las siguientes distribuciones de probabilidad para las variables I y J:

- M :Distribución Exponencial.
- D :Distribución Degenerada
- E_k : Distribución Erlang de parámetro k o k-Erlang.
- H_k : Distribución hiper-exponencial en k estados de tiempo de servicio

2.2 Modelo de colas M/M/1

El modelo de colas M/M/1 es uno de los modelos más básicos y ampliamente utilizados en teoría de colas. Representa un sistema de atención con:

- Llegadas **aleatorias** que siguen una distribución **exponencial** (es decir, un proceso de Poisson).
- Tiempos de servicio **aleatorios**, también distribuidos exponencialmente.
- Un único servidor que atiende a los clientes en orden de llegada (disciplina FIFO: *First-In, First-Out*).

Notación M/M/1:

- El primer "M" significa que los tiempos entre llegadas son *Markovianos* (exponencial).
- El segundo "M" indica que los tiempos de servicio también son *Markovianos*.
- El "1" representa que hay un solo servidor.

Parámetros principales:

- λ: tasa promedio de llegada de clientes al sistema (clientes por unidad de tiempo).
- μ: tasa promedio de servicio del servidor (clientes atendidos por unidad de tiempo).

Variables de desempeño típicas:

- $\rho = \frac{\lambda}{\mu}$: factor de utilización del sistema. Debe cumplirse $\rho < 1$ para estabilidad.
- $L(t)$: número promedio de clientes en el sistema en el instante t.
- L_q: número promedio de clientes en la cola.
- W: tiempo promedio que un cliente pasa en el sistema.
- W_q: tiempo promedio que un cliente pasa esperando en la cola.
- P(n): Probabilidad de encontrar n clientes en el sistema.
- P cola(n): Probabilidad de encontrar n clientes en cola.
- P(K): Probabilidad de denegación de servicio en un sistema M/M/1/K (K = Cap Máxima).

Fórmulas clave:

$$
\begin{aligned}
L &= \frac{\rho}{1-\rho} \\
L_q &= \frac{\rho^2}{1-\rho} \\
W &= \frac{1}{\mu-\lambda} \\
W_q &= \frac{\lambda}{\mu(\mu-\lambda)} = W - \frac{1}{\mu} \\
P(n) &= (1-\rho)\rho^n \\
P_{cola}(n) &= (1-\rho)\rho^{n+1} \\
P(K) &= \frac{(1-\rho)\rho^K}{1-\rho^{K+1}}
\end{aligned}
$$

2.3 Modelo de Inventario

Este tipo de modelo se fundamenta en la gestión del stock dentro de un sistema. Su principal objetivo es almacenar unidades de un artículo para responder a las variaciones en la demanda. El stock representa la cantidad disponible de un producto en un momento dado, y va cambiando en el tiempo según los eventos que se presenten, como el consumo por parte de la demanda o la reposición a través de pedidos programados con base en una política de inventario establecida. Los modelos de simulación de inventario se utilizan para comparar distintas políticas de control de stock y evaluar su desempeño, considerando tanto los costos implicados como la eficacia en la satisfacción de la demanda. Las políticas

de inventario determinan los niveles mínimos y máximos que se deben mantener y se busca alcanzar un nivel crítico que permita minimizar tanto los costos asociados a mantener stock como los costos de escasez o de ordenar nuevos productos. Los tipos de costos que se presentan en este modelo son:

- Costos de pedido: Representa un gasto fijo por cada solicitud de reposición, y puede depender tanto del número de unidades pedidas como de la frecuencia de los pedidos.
- Costo de mantenimiento: Se refiere a los gastos relacionados con almacenar productos en stock, dependiendo del nivel de inventario y de los parámetros definidos por la política de control.
- Costo por falta de stock: Se produce cuando el inventario es insuficiente para cubrir la demanda y se generan costos por no poder satisfacerla, ya sea por pérdida de ventas o retrasos en la entrega.

En este trabajo se considera un modelo de revisión continua tipo (s, Q), donde:

- s: punto de re-orden. Cuando el inventario disponible cae por debajo de s, se realiza un pedido.
- Q: cantidad fija que se ordena cada vez.

Supuestos típicos del modelo:

- La demanda sigue una distribución probabilística (por ejemplo, Poisson o normal) y es independiente en cada unidad de tiempo.
- El tiempo de entrega (lead time) puede ser determinístico o aleatorio.
- Se permiten faltantes, los cuales incurren en un costo.
- Los pedidos son recibidos instantáneamente o con un lead time definido.

Parámetros:

- D: demanda media por unidad de tiempo.
- s: nivel de inventario que activa un pedido.
- Q: cantidad de unidades solicitadas por pedido.
- L: lead time (tiempo entre que se hace el pedido y se recibe).
- C_o: costo por realizar un pedido (setup cost).
- C_h: costo de mantener una unidad en inventario por unidad de tiempo.
- C_s: costo por unidad faltante o backorder.

Variables de desempeño:

- Costo total: suma de los tres tipos de costos principales.
- Número de pedidos por unidad de tiempo: D/Q.
- Inventario promedio: depende de Q y la política de control.
- Nivel de servicio: proporción de demanda que se satisface sin incurrir en faltantes.

Fórmulas generales:

Costo total anual esperado:

$$CT = \underbrace{\frac{D}{Q}C_o}_{\text{Costo de orden}} + \underbrace{\frac{Q}{2}C_h}_{\text{Costo de mantenimiento}} + \underbrace{E[\text{Faltantes}] \cdot C_s}_{\text{Costo de faltantes}}$$

Importancia del modelo: Este tipo de modelos es fundamental para evaluar cómo distintas políticas de reposición afectan el desempeño económico de un sistema. En este trabajo, se realizarán simulaciones variando los parámetros del modelo y se analizarán los resultados bajo diferentes configuraciones, justificando la elección de parámetros y evaluando los costos totales, con especial atención al impacto de la variabilidad de la demanda.

3 Descripción del trabajo

El presente trabajo consiste en el análisis e implementación de dos modelos clásicos de simulaciones de eventos discretos: el modelo de colas M/M/1 y un modelo de control de inventarios discreto. Ambos modelos se simulan con el objetivo de evaluar el comportamiento de un sistema de atención al cliente y la gestión de existencias dentro de una empresa. Explicitaremos las fórmulas matemáticas utilizadas para cada modelo. De ambas analizaremos las medidas de rendimiento finales variando sus parámetros, en el MM1 algunos de ellos son el promedio de clientes en el sistema y en cola, probabilidad de n clientes en cola y utilización del servidor. Para el de inventario serán los costos de mantenimiento, de orden, de faltante de stock y la sumatoria de todos estos. Para cada modelo se generaran como mínimo 10 corridas, incluyendo gráficos y conclusiones, comparando los resultados desde su valor obtenido, el valor obtenido a través del programa en Python y de su implementación en AnyLogic.

4 Resultados

4.1 Resultados modelo M/M/1

En esta sección, analizaremos el desarrollo y ejecución de un modelo de colas M/M/1 en código Python donde se desarrolló una simulación por eventos discretos del sistema de colas M/M/1. El objetivo de la simulación es calcular medidas de rendimiento del sistema a lo largo de múltiples corridas independientes, para así obtener estimaciones estables.

Estructura del código
El simulador desarrollado sigue una estructura de simulación basada en eventos discretos, en la cual el tiempo avanza hacia el siguiente evento relevante: una llegada o una salida (fin de atención).

1. **Parámetros de entrada:**
   - -c: Cantidad de corridas del sistema. Define cuántas veces se ejecuta la simulación con los mismos parámetros, permitiendo obtener promedios y disminuir la variabilidad de los resultados.
   - -n: Cantidad de usuarios a simular. Es el número total de clientes que pasarán por el sistema en cada corrida. Se utiliza como criterio de corte para finalizar la simulación.
   - -mu (μ): Tasa de servicio del servidor (clientes por unidad de tiempo). Representa la capacidad del sistema para atender clientes. Debe ser mayor que la tasa de arribo (λ) para que el sistema sea estable; de lo contrario, la cola crecería indefinidamente.
   - -l (λ): Tasa de arribo de clientes al sistema (clientes por unidad de tiempo). Define la frecuencia con la que los usuarios llegan al sistema. Se modela como un proceso de Poisson, por lo que el tiempo entre llegadas sigue una distribución exponencial con parámetro λ.
2. **Generación de variables aleatorias:** El código genera los tiempos de arribo y de retiro utilizando el método de la inversa, que transforma una variable uniforme en una exponencial. La fórmula usada es:

$$T = -\frac{1}{\lambda} \cdot \ln U$$

   donde:
   - T es el valor generado (tiempo entre eventos).
   - λ es la tasa correspondiente (λ para llegadas o μ para servicios).
   - U es una variable aleatoria uniforme en el intervalo (0, 1), generada con la función random.uniform(0, 1).

   El proceso se repite para cada cliente, utilizando variables aleatorias con distribución exponencial para representar la aleatoriedad de los tiempos entre llegadas y de servicio. Para los arribos de clientes, se modelan como una suma acumulada de tiempos de arribo. Para los retiros de clientes, el tiempo depende de dos factores: tiempo de arribo y tiempo que deberá esperar si el servidor está ocupado. Por eso, el tiempo de servicio se suma al tiempo de arribo más la demora, si la hay.
3. **Cálculo de momentos clave por cliente:** Para cada cliente:
   - Se acumula el tiempo de llegada.
   - Se define el momento de inicio del servicio como el máximo entre su tiempo de llegada y el fin del servicio anterior.
   - Se calcula el fin de servicio como inicio + duración del servicio.

Esto permite construir eventos del tipo:

- Inicio de servicio
- Fin de servicio
- Tiempo en cola = entrada servicio - llegada
- Tiempo en sistema = retiro - llegada

4. **Cálculo de métricas por corrida:** Se registran para cada corrida los siguientes valores:
   - Promedio de tiempo en cola,
   - Promedio de tiempo en sistema,
   - Promedio de clientes en cola y en el sistema,
   - Utilización del servidor,
   - Cantidad máxima de clientes en cola,
   - Frecuencia de cada tamaño de cola.
5. **Promedios finales:** Después de todas las corridas (c veces), se calcula el promedio de cada métrica para obtener resultados representativos del rendimiento del sistema.

4.1.1 Resultados del modelo de colas M/M/1 en Python

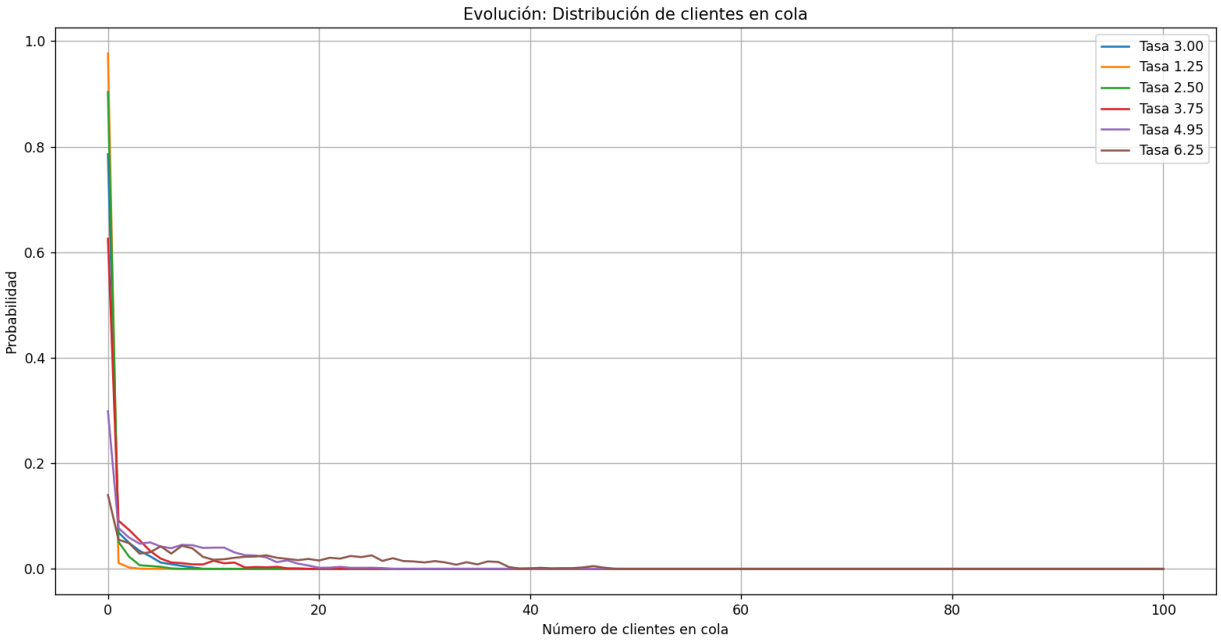
Se simulo el comportamiento de un sistema M/M/1 utilizando los siguientes parámetros: 10 corridas, 100 usuarios a simular, $\mu = 5$, $\lambda = 3$. Este escenario representa un sistema sobrecargado ($\rho = \frac{3}{5}$), lo cual implica que el sistema es inestable: la cola tiende a crecer indefinidamente si no se impone un límite en su capacidad. A continuación se muestran los resultados obtenidos a partir de la simulación, representados en distintas gráficas:

Métricas por tasa de arribo

	3.00	1.25	2.50	3.75	4.95	6.25
Clientes promedio en sistema	1.45447	0.32297	0.85602	2.77046	7.03438	15.07784
Clientes promedio en cola	0.86949	0.07775	0.37802	2.05687	6.1509	14.11763
Tiempo promedio en sistema	0.49626	0.26406	0.34161	0.76123	1.58661	3.31423
Tiempo promedio en cola	0.17694	0.01299	0.06371	0.41611	1.2133	2.91474
Utilización servidor	0.58498	0.24523	0.478	0.71359	0.88347	0.96021

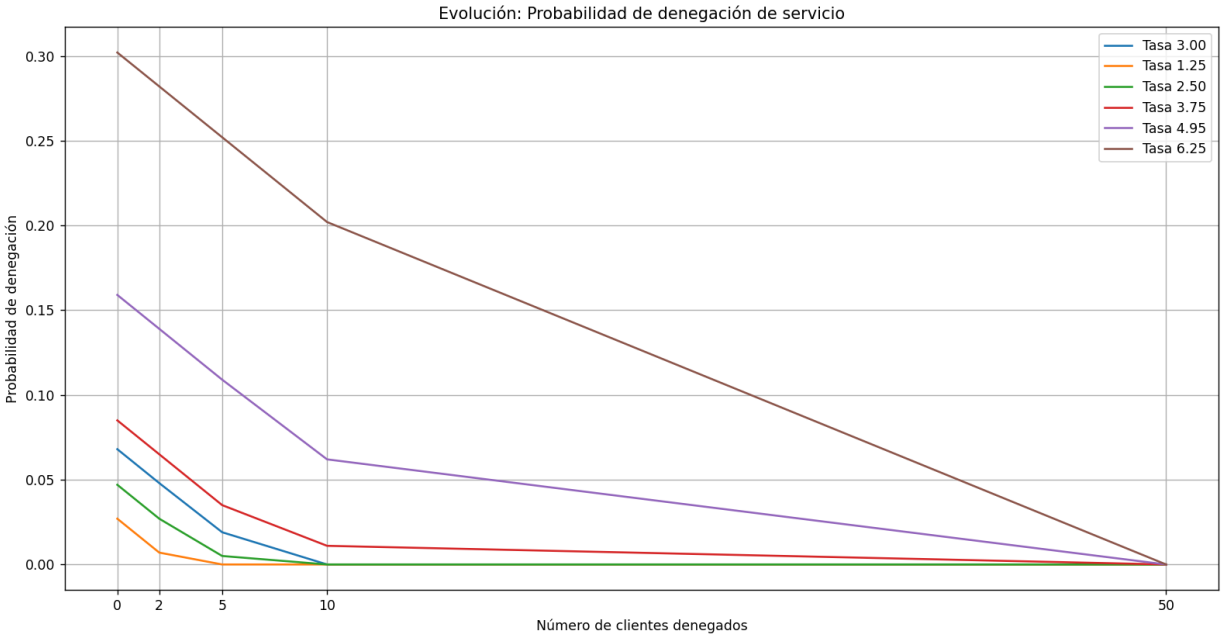
La tabla muestra cómo evolucionan las principales métricas a medida que aumenta la tasa de arribo λ:

- **Clientes en el sistema y en cola:** Crecen progresivamente con λ. Para valores bajos de λ, el sistema funciona sin congestión. A partir de $\lambda = 3{,}75$, las colas y el número de clientes en sistema comienzan a aumentar rápidamente. En $\lambda = 6{,}25$, se evidencia una situación de saturación.
- **Tiempos promedio:** Tanto el tiempo total en el sistema (W) como el tiempo en la cola (W_q) aumentan fuertemente al acercarse a μ, y se disparan cuando $\lambda > \mu$. Esto evidencia que la espera se vuelve crítica en entornos sobrecargados.
- **Utilización del servidor:** Aumenta con λ, acercándose al 100%. Este indicador refleja la proporción de tiempo que el servidor está ocupado, y resulta clave para evaluar la eficiencia del sistema.



Este gráfico muestra la probabilidad de encontrar exactamente n clientes en la cola:

- Para tasas bajas ($\lambda = 1{,}25$ o $2{,}5$), la probabilidad está concentrada en valores bajos de n (0 o 1), lo que indica un sistema ágil y sin acumulación.
- A medida que λ se acerca a μ, la distribución se vuelve más dispersa y aparecen colas más largas.
- En $\lambda = 4{,}95$ y $6{,}25$, hay probabilidades considerables de encontrar colas de 10, 20 o incluso más clientes, lo que representa un sistema cercano al colapso.



El gráfico muestra la probabilidad de que un cliente sea rechazado cuando el sistema tiene una capacidad de cola limitada ($K = 0, 2, 5, 10, 50$):

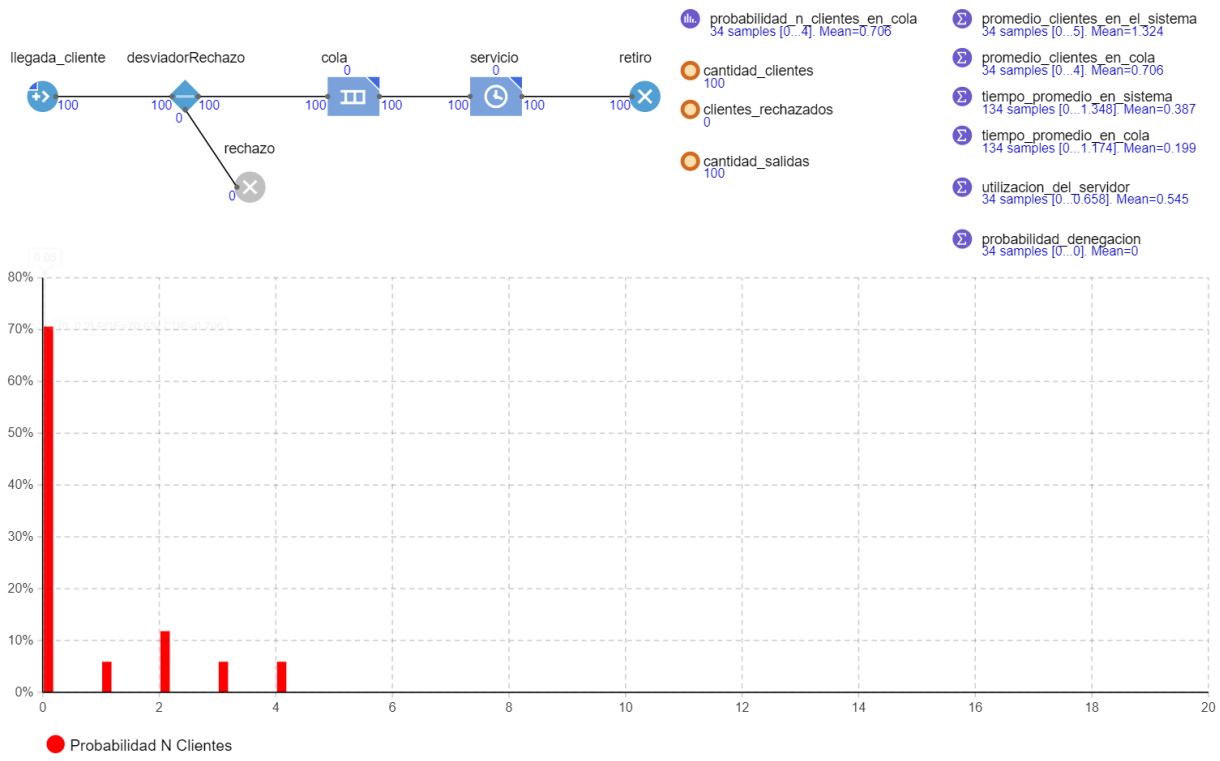
Conclusión del análisis gráfico

Los gráficos presentados permiten visualizar cómo el sistema M/M/1 reacciona ante diferentes niveles de carga. Para valores de λ menores a μ, el sistema opera de manera estable y eficiente. Sin embargo, cuando λ se aproxima a μ o lo supera, comienzan a evidenciarse problemas de saturación: colas largas, tiempos de espera elevados y altas tasas de rechazo.

- Para $\lambda \leq 2{,}5$, la probabilidad de rechazo es cercana a cero incluso con colas pequeñas. El sistema puede atender la demanda sin saturarse.
- Con $\lambda = 4{,}95$, la probabilidad de rechazo ya no es despreciable para $K \leq 10$.
- En $\lambda = 6{,}25$, se observa una probabilidad de denegación mayor al 30% cuando $K = 0$, lo que indica que un tercio de los clientes no recibe servicio.
- A medida que se incrementa la capacidad de cola, la probabilidad de rechazo disminuye, pero no desaparece completamente cuando $\lambda > \mu$.

4.1.2 Resultados del modelo de colas en AnyLogic

En esta sección, analizaremos el comportamiento del modelo de colas M/M/1 implementado en AnyLogic, utilizando los mismos parámetros que en la simulación realizada con Python: simulando el paso de 100 clientes por el sistema, $\lambda = 3$ y $\mu = 5$. En la siguiente imagen se muestra la estructura del modelo y la distribución de probabilidad del número de clientes en cola. Además, se visualizan las métricas obtenidas durante la ejecución de la simulación.



Los resultados reflejan que el promedio de clientes en el sistema fue de aproximadamente 1.32, mientras que el promedio de clientes en cola se situó cerca de 0.36. A su vez, el tiempo promedio que un cliente permanece en el sistema fue cercano a 0.387 unidades de tiempo, y el tiempo promedio en cola fue de alrededor de 0.199. La utilización del servidor fue de 0.545, lo que indica que el servidor estuvo ocupado el 54.5% del tiempo. Este valor muestra que el sistema funciona de manera eficiente, sin sobrecargas ni tiempos ociosos excesivos. Además, se observa que no hubo denegación de servicio, es decir, no se rechazaron clientes durante la simulación. En base a estas métricas, puede concluirse que el sistema es estable y responde adecuadamente para esta configuración de demanda.

Ahora vamos a mostrar cómo se comporta el sistema con una cola con capacidad de 2:

Podemos observar que con una cola con capacidad de 2 clientes la utilización del servidor sigue siendo buena sin saturarse, pero esta vez sí se han rechazado clientes, debido a que ya había 2 clientes en cola.
Si lo llevamos a un entorno realista, la segunda evaluación se asemejaría más.

4.1.3 Comparación de resultados

Con el fin de validar la implementación del modelo M/M/1 y comprender cómo se comporta el sistema bajo distintas herramientas de análisis, se compararon los resultados obtenidos mediante tres fuentes:

- Fórmulas analíticas de teoría de colas.
- Simulación en Python con generación manual de eventos.
- Simulación visual con AnyLogic.

Se utilizó el siguiente conjunto de parámetros comunes:

- $\lambda = 3$
- $\mu = 5$
- 100 clientes simulados
- 10 corridas

Comparación teórica vs Simulación Python vs Simulacion en AnyLogic

λ	ρ	L teórico	L simulado (py)	L simulado (alp)	W teórico	W simulado (py)	W simulado (alp)
3.00	0.60	1.50	1.21	1.324	0.50	0.40	0.387
1.25	0.25	0.33	0.34	0.382	0.27	0.26	0.269
2.50	0.50	1.00	0.90	0.825	0.40	0.37	0.358
3.75	0.75	3.00	2.47	2.111	0.80	0.66	0.6
4.95	0.99	99.00	7.11	4.348	20.00	1.58	0.959
6.25	1.25	–	14.04	7.842	–	2.80	1.423

Table 1: Comparación de cantidad promedio de clientes en el sitema (L) y tiempo promedio en el sistema (W)

Análisis de la simulación en Python
Para tasas de arribo con $\rho < 1$, los resultados simulados en Python se aproximan correctamente a los valores teóricos, con una diferencia razonable atribuida a la aleatoriedad y a que se simularon solo 100 clientes por corrida. A medida que $\rho \to 1$, como en $\lambda = 4.95$, las métricas teóricas crecen muy rápidamente (tienden a infinito). En cambio, la simulación muestra valores más moderados, producto de trabajar con una muestra finita y sin calentamiento (warm-up). Para $\rho > 1$ (por ejemplo, $\lambda = 6.25$), el sistema se vuelve inestable, por lo que los valores teóricos dejan de tener sentido (incluso aparecen negativos por inestabilidad numérica). Sin embargo, la simulación sigue funcionando y refleja cómo el sistema colapsa progresivamente (crecen cola y tiempos).

Análisis de la simulación en AnyLogic
Se realizaron dos simulaciones del modelo M/M/1 en AnyLogic bajo los mismos parámetros de entrada: una tasa de arribo $\lambda = 3$ clientes por unidad de tiempo, una tasa de servicio $\mu = 5$, y una población de 100 clientes. En la primera simulación, se modelo una cola sin límite de capacidad, es decir, los clientes no eran rechazados al llegar, sin importar la cantidad de personas ya en espera. Los valores de las métricas reflejan un comportamiento esperable para un sistema estable (donde $\lambda < \mu$). El servidor está moderadamente ocupado, la espera es baja y no se produce congestión. En la segunda simulación, se incorporó una restricción en la capacidad de la cola, fijando un máximo de 2 clientes en espera. Una vez alcanzado ese umbral, los nuevos clientes que intentan ingresar al sistema son rechazados. Los resultados mostraron un cambio importante ya que se observaron rechazos de clientes y el promedio de clientes en el sistema y en cola disminuyó levemente. Este escenario es más representativo de un entorno real, donde no siempre es posible permitir que los clientes esperen indefinidamente. Sin embargo, una capacidad de cola tan baja puede generar una pérdida considerable de clientes.

4.2 Modelo de inventario

En esta sección, analizaremos el desarrollo y ejecución de un modelo de inventario en código Python en el cual aplicamos distintas políticas para luego evaluar los resultados obtenidos a través de la simulación.

Estructura del código
El simulador desarrollado muestra cómo evoluciona el inventario y los costos asociados (Q, R) considerando una demanda aleatoria y tiempos de reposición aleatorios **Parámetros de la simulación**
Para todos los escenarios posibles se tiene que el tiempo de arribo de clientes es un proceso de Poisson que sigue una distribución exponencial. Para todos los escenarios utilizaremos un nivel inicial de inventario de 30 unidades y las distintas políticas implicarán modificar los siguientes parámetros:

- Valor máximo de inventario
- Punto de compra
- Cantidad de arribos de clientes.

En base a esto, podremos visualizar el número de elementos en el inventario y además determinar los siguientes costos:

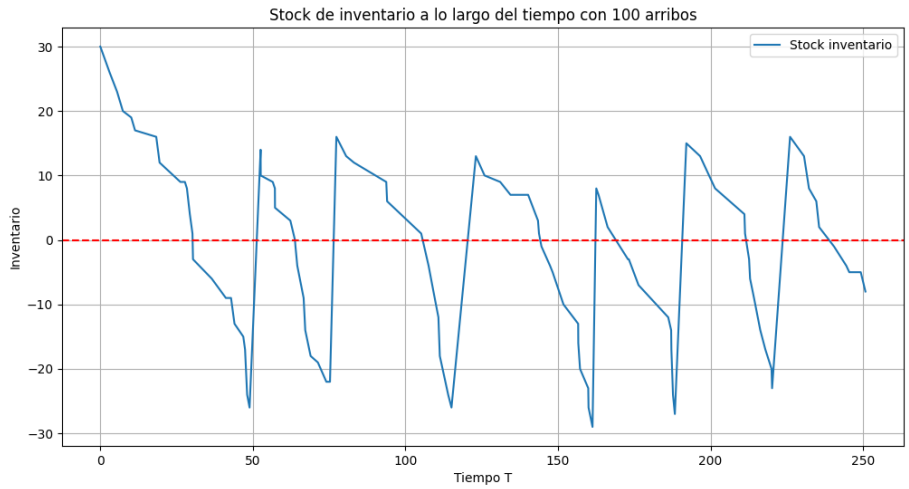
- Costo de orden: Costo fijo al realizar una compra.
- Costo de mantenimiento: Costo de mantenimiento unitario (3 u.m) multiplicado por la cantidad de unidades disponibles por unidad de tiempo.
- Costo por escasez: Costo de escasez unitario (5 u.m) multiplicado por la cantidad de unidades demandadas que no estén disponibles por unidad de tiempo.

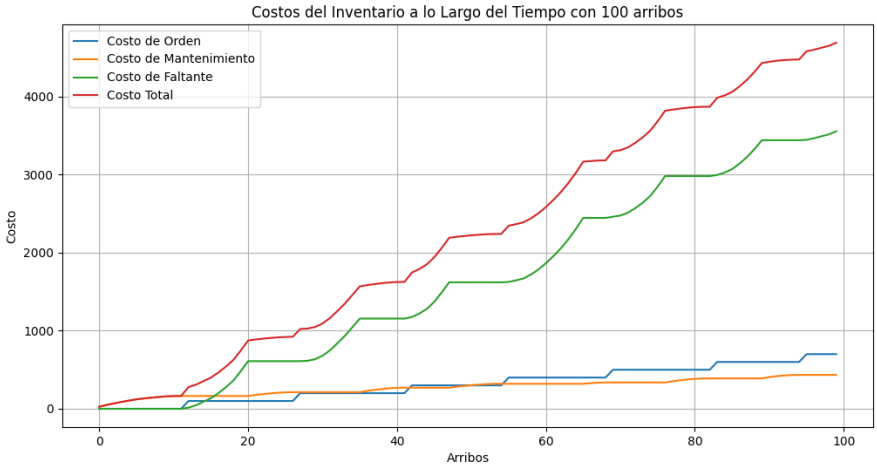
Notar que hacemos énfasis en que el costo por unidad no disponible es superior al costo de mantenimiento unitario. Esto debido a que suponemos que el faltante puede desembocar en ventas perdidas que afecten monetariamente más que el costo de mantenimiento para esa unidad.

4.2.1 Resultados modelo de inventario en Python

Aplicamos la política de inventario con punto de compra $z = 0$ y nivel máximo $Z = 30$.
La tasa de demanda es $\lambda = 3$ y la tasa de arribos es $\sigma = 0.45$.

Como puede apreciarse, hubo varios períodos en el cual el inventario no fue suficiente para suplir la demanda. Esto podría mejorarse aplicando una política de inventario que anticipe mejor estas situaciones provocando que se realicen órdenes de pedidos más frecuencias al aumentar el valor de la cota inferior z

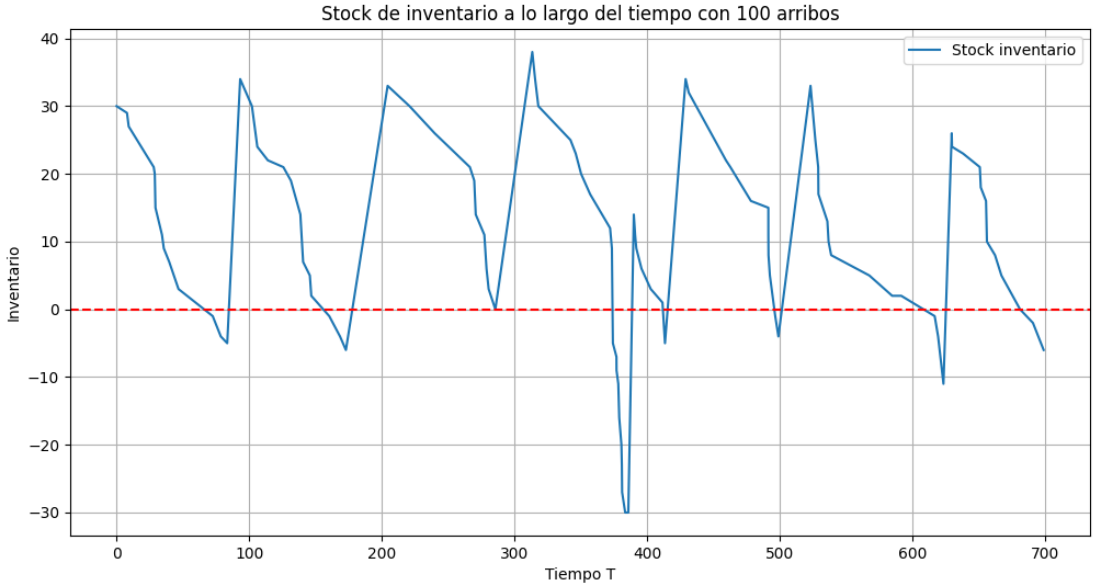




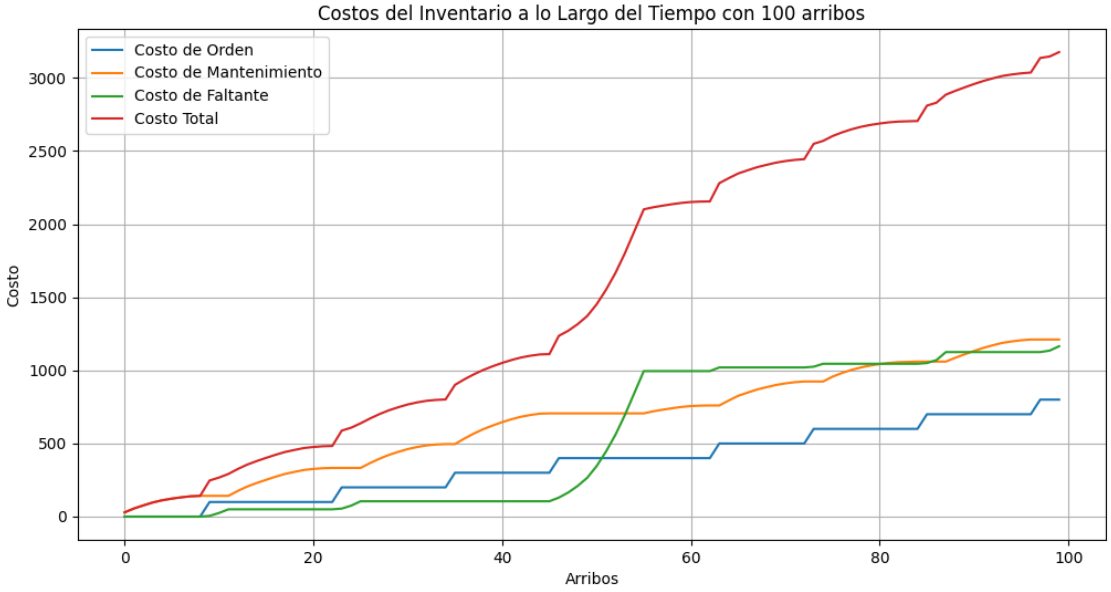
Debido a la cantidad de veces que el nivel de inventario no fue suficiente, podemos ver que la mayor parte del costo total obtenido es a razón del costo por faltante. Los resultados exactos de los costos fueron:

- Costo de orden total: $31.700.00
- Costo de mantenimiento total: $27.721.00
- Costo de faltantes total: $164.105.00
- Costo total: $223.526.00

Analicemos ahora una simulación pero con una tasa de demanda $\lambda = 1.5$ y la tasa de arribos es $\sigma = 0.25$.



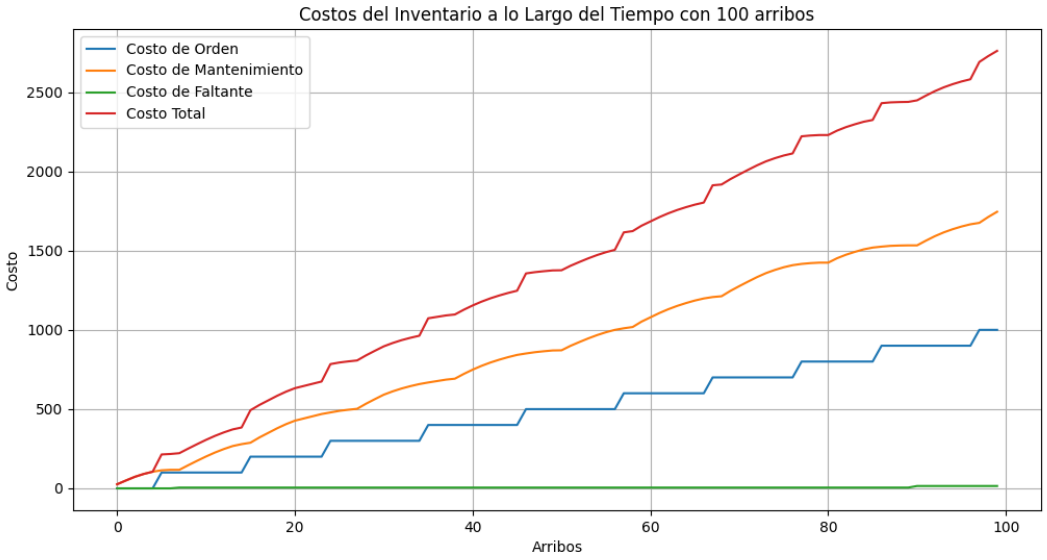
Como podemos observar, al disminuir la llegada de clientes y la cantidad de artículos, hay muchas menos oportunidades en las que el inventario se vacía.

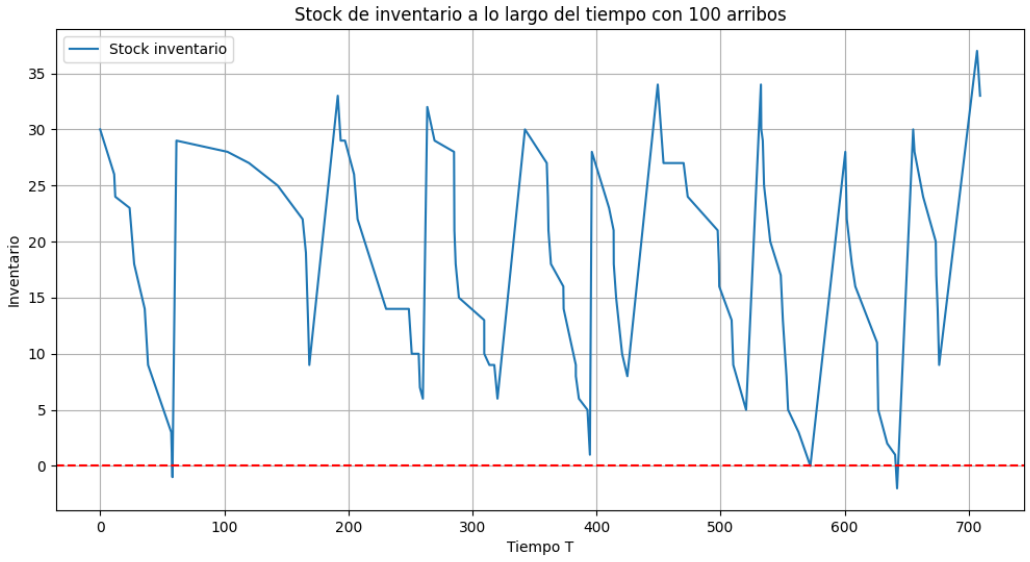


Debido a que el costo por faltante es superior al costo de mantenimiento, notamos una importante baja en los costos de esta nueva simulación. Estos dos casos nos permiten concluir en que no solo importa en qué momento uno hace la reposición del inventario, sino que también la densidad en la que se vacía este. Con esto quiero decir que un modelo no es siempre malo o bueno, sino que a veces depende del contexto en el que está ubicado. Con esto queremos decir que se debe adaptar la estrategia del negocio al momento actual que está viviendo el mercado, para así poder modificar las métricas ajustándolas a lo demandado.

Probando con distintas políticas de inventario
En todas las siguientes simulaciones utilizamos una tasa de demanda $\lambda = 3$ y una tasa de arribos $\sigma = 0.15$.

- Punto de compra:z = 10 y Nivel máximo de inventario: Z = 40





- Punto de compra:z = 15 y Nivel máximo de inventario: Z = 40

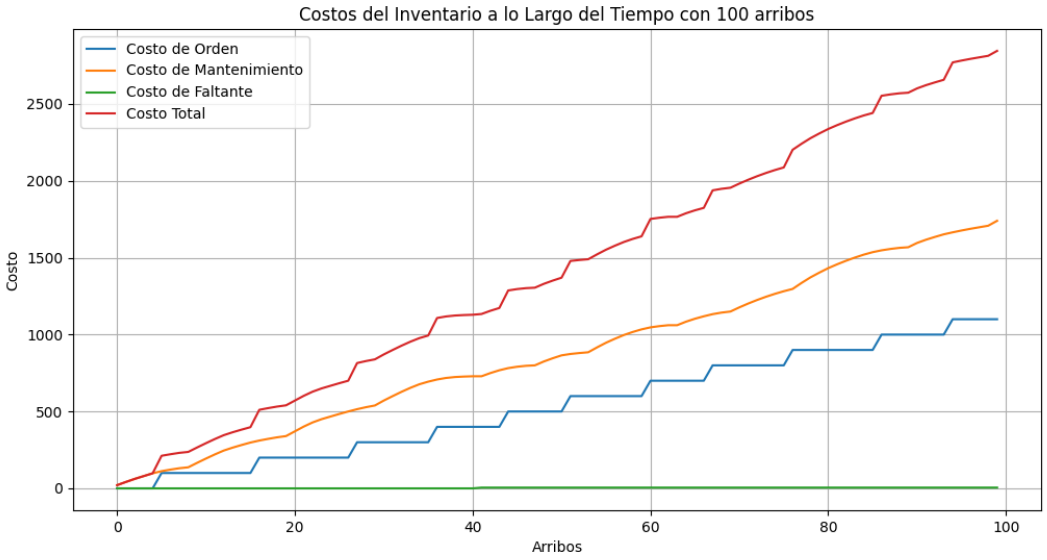
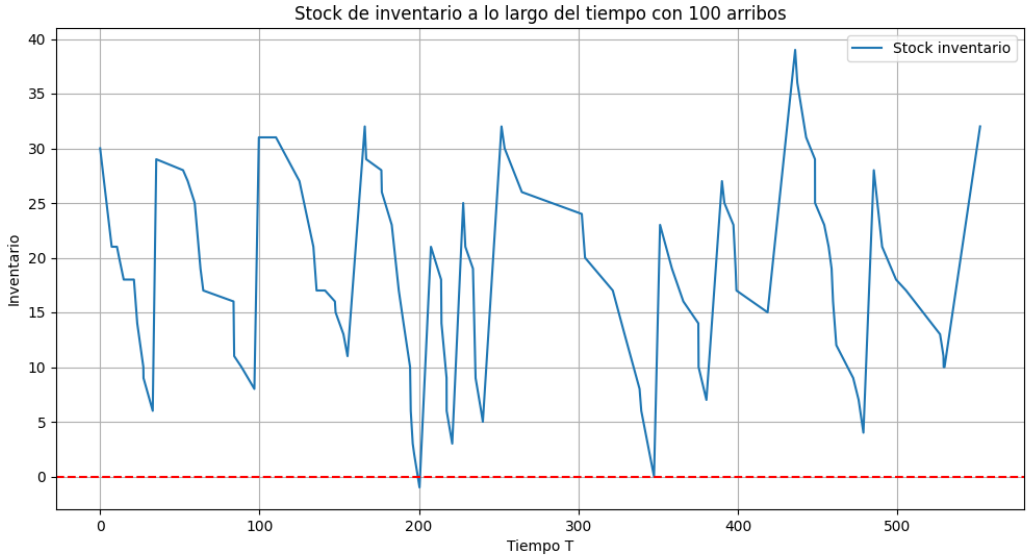
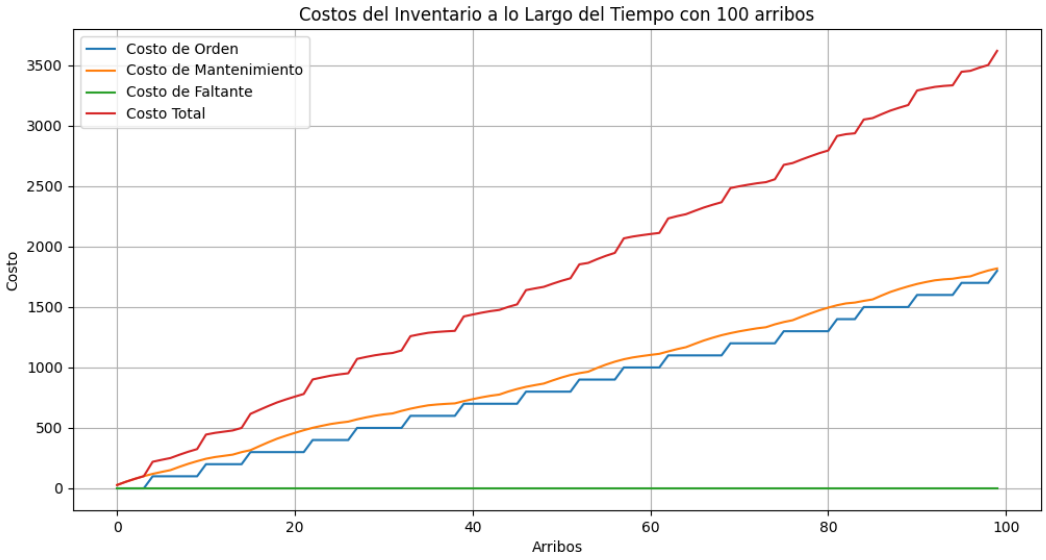
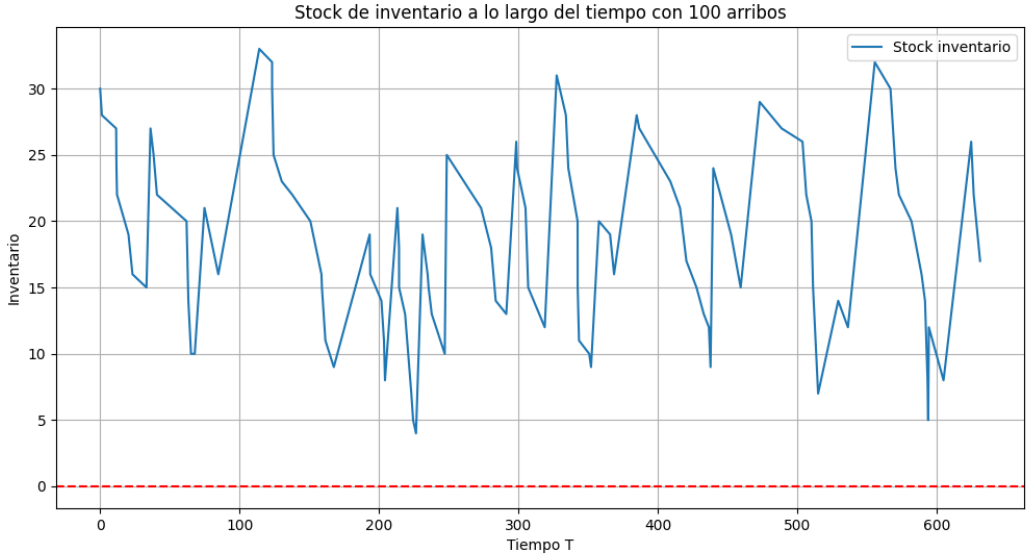


Figure 1: Costos del Inventario



- Punto de compra:z = 20 y Nivel máximo de inventario: Z = 30





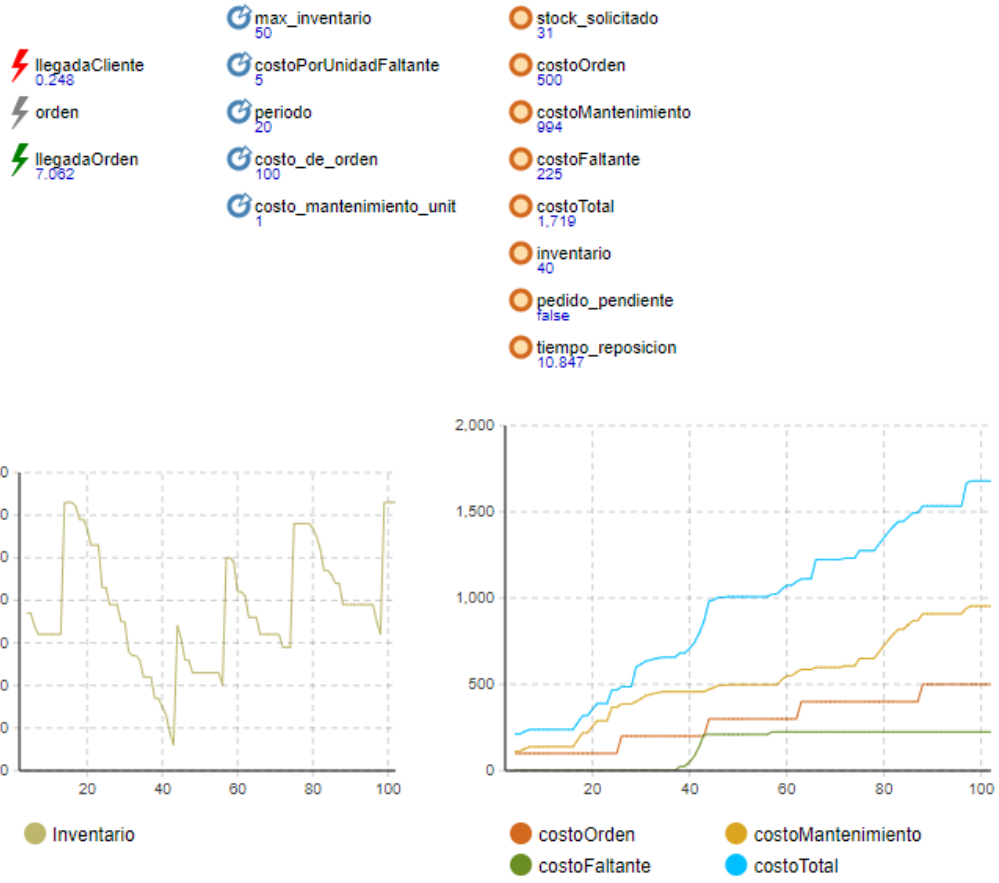
Resultados finales:

- Costo de orden total: $84000.00
- Costo de mantenimiento total: $93604.00
- Costo de faltantes total: $0.00
- Costo total: $177604.00

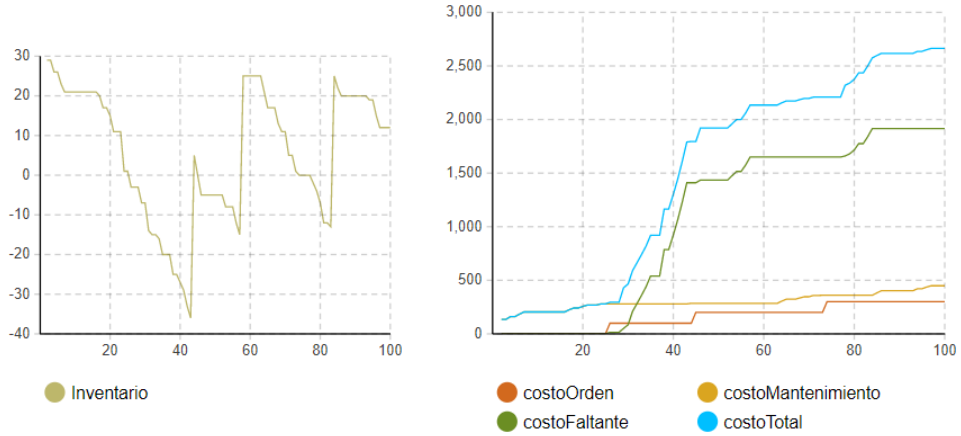
Como podemos observar, estableciendo un punto de compra en z = 20 y un nivel máximo de inventario de Z = 30 es suficiente no tener costos de escasez, concluyendo que se deban efectuar órdenes a proveedores más seguidas para así evitar niveles de inventario negativos.

4.2.2 Resultados modelo de inventario en AnyLogic

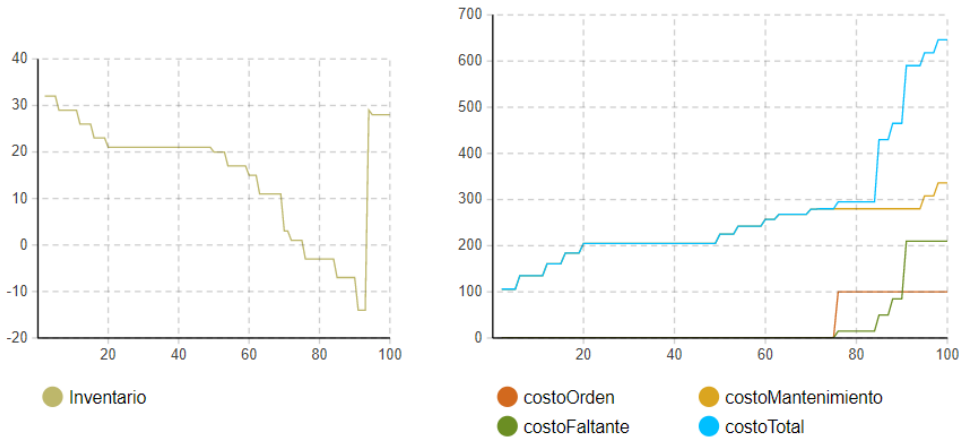
En esta sección, analizaremos el comportamiento del modelo de inventario, pero ahora simulado con Anylogic, aplicando las mismas políticas de parámetros de la simulación anteriormente aplicadas en Python.
Una simulación de modelo de inventario en AnyLogic se ve así:



Allí podemos visualizar el gráfico correspondiente al nivel de inventario a través del tiempo. También se incluyeron relojes de eventos que representan a cada uno de los posibles eventos de interrupción que modificarán el nivel de inventario (la llegada de clientes y la llegada de la orden). Por otro lado,se incluyeron gráficas de tiempo para visualizar el crecimiento de cada uno de los costos referidos a las órdenes, al mantenimiento de mercadería y a la penalización por stock insuficiente.
Salida de primer política de inventario z = 0 y Z = 40

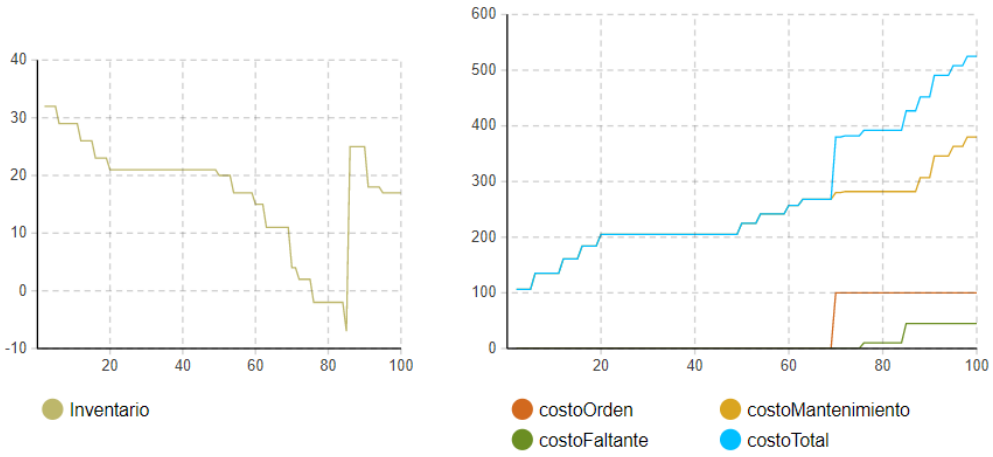


Simulación efectuada con una tasa de llegada de clientes de $\sigma = 0.45$.
Notemos que la mayor parte del costo total es a razón de las múltiples veces en las que quedamos con inventario negativo, siendo el costo por faltante el costo a corregir.
Probemos ahora disminuyendo la tasa de llegada de clientes a $\sigma = 0.15$:



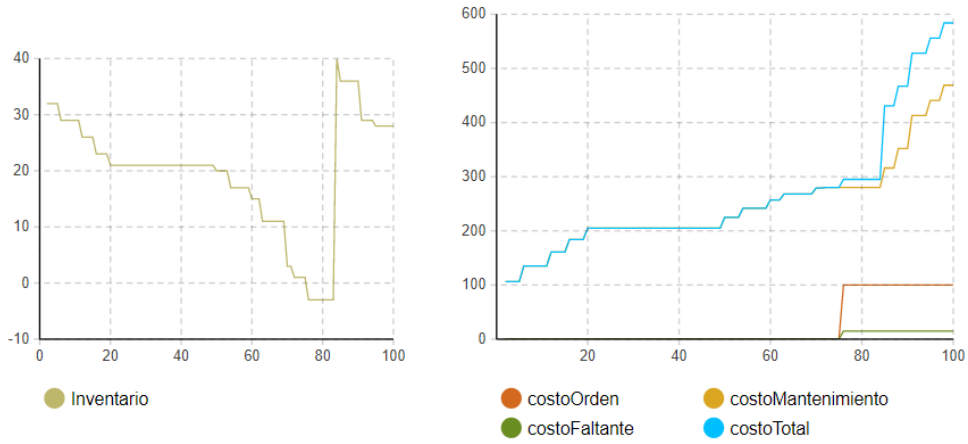
Un par de cuestiones a resaltar luego de estos datos; con una menor tasa de llegada de clientes los costos disminuyen notablemente, debido a que al realizarse menos ventas por ende disminuye la cantidad de compras que deben realizarse, disminuyendo el costo de orden. Por otro lado, al sufrir menos veces un nivel de inventario negativo, el costo por unidad faltante sera mucho menor. Estos dos factores desencadenan en un costo total inferior cuando se aplica una tasa de llegada de clientes de $\sigma = 0.15$ en comparación a una tasa de $\sigma = 0.45$Punto de compra:z = 10 y Nivel máximo de inventario: Z = 40

Punto de compra: z = 10 y Nivel máximo de inventario: Z = 40



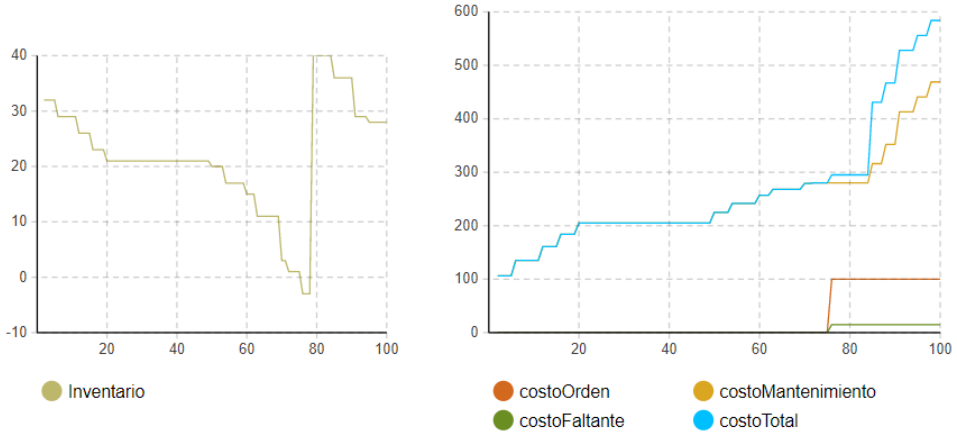
Como es observable, a partir de que tomamos una política más exigente elevando la cota inferior (z) a 20, ocurre lo mismo que se observaba con la implementación en Python, el nivel de inventario se mantiene prácticamente siempre por encima del 0 y así se reduce notablemente el costo por escasez, y esto tiene sentido, ya que una vez que el nivel de inventario sea menor a 20 inmediatamente se van a hacer los pedidos de reposición correspondiente. Como contraparte, esto puede llegar a aumentar los costos por ordenes, debido a que con esta política vamos a realizar mayor cantidad de ordenes, y también aumentara el costo por mantenimiento, debido a que, en promedio, poseeremos mayor cantidad de unidades en stock por unidad de tiempo. Se deben encontrar las politicas exactas para lograr la mayor eficacia posible para el modelo de inventario.

Veamos que sucede si variamos la demora del proveedor
Todas las simulaciones ejecutadas hata el momento, se realizaron utilizando un tiempo de demora de reposicion resultado de multiplicar el periodo (p = 20) con un numero X aleatorio que sigue una distribucion uniforme entre (0.5, 1).
Disminuyamos la variable aleatoria X / $X \sim \mathcal{U}(0.10,\ 0.5)$.



Es interesante ver como a veces las variables externas a nuestro negocio son las que terminan modificando una ecuacion. En este caso, vemos como al disminuir el tiempo de demora del proveedor en la entrega del producto, utilizando un punto de compra de z = 0, prácticamente no ocurren momentos en el que el nivel de inventario es negativo. El problema de esto, es que al ser una variable externa, por ahi puede exceder de nuestro alcance. No obstante, si tenemos la posibilidad de lograr esa aceleración en el proceso de reposición, deberíamos considerar llevarlo a cabo.

Disminuyamos la variable aleatoria X aun mas / $X \sim \mathcal{U}(0.05,\ 0.15)$.



5 Conclusión

5.1 Conclusión Modelo de Colas

Luego de haber realizado las correspondientes simulaciones, el análisis demuestra que la simulación desarrollada en Python reproduce fielmente el comportamiento del modelo M/M/1 dentro del régimen de estabilidad ($\rho < 1$). Además, permite explorar escenarios fuera del dominio teórico, lo cual es especialmente útil para estudiar saturación y rechazo en colas finitas. El análisis comparativo de los dos escenarios en AnyLogic demuestra cómo la capacidad de cola influye directamente en el desempeño del sistema. Con cola infinita, el modelo se comporta de forma óptima: no hay pérdidas, el tiempo de espera es bajo y el servidor trabaja de manera eficiente. Es un sistema idealizado. Con cola finita de tamaño 2, el sistema introduce rechazos que afectan a la accesibilidad del servicio. El sistema comienza a mostrar signos de congestión. Por lo que, el valor óptimo de la capacidad de cola dependerá del equilibrio deseado entre tiempo de espera y tasa de rechazo. Ambas simulaciones son útiles para analizar distintos escenarios operativos: el primero demuestra la capacidad teórica del sistema, mientras que el segundo refleja una limitación práctica que puede aparecer en situaciones reales. Estas pruebas permiten tomar decisiones informadas sobre cómo dimensionar recursos y establecer límites en un sistema de colas, según el equilibrio deseado entre eficiencia y accesibilidad.

5.2 Conclusión Modelo de Inventario

Luego de haber realizado las correspondientes simulaciones de varios modelos de control de inventario de los cuales las empresas se basan para tomar decisiones estrategicas en sus negocios hemos notado que tanto el modelo de inventario aplicado en Python como el aplicado en AnyLogic responden similarmente a las variables ingresadas, variando levemente razones internas de cada herramienta. Ambos modelos dejan en evidencia que al aumentar la frecuencia de que los clientes acuden a realizar compras, representado por la tasa de demandas, es necesario aumentar los minimos de inventario, ya que quedo demostrado que los costos que mas pesan son los Costos de Orden y Costos de Escasez. Mas alla de esto, no es cuestion de aumentar los parametros sin medida, debido a que lo que se gana por un lado, se pierde por el otro. Ademas, debemos tener en cuenta que todos los recursos utilizados representan recursos de la vida real. Por ejemplo, si se decide tener un monto muy elevado de elementos en stock para no perder ninguna venta y por ende no tener Costos de Escasez, esto hara que se tenga un Costo de Mantenimiento mucho superior al correspondiente.
Analizando otros aspectos observaddos, algo que influye notablemente es la demora entre que se solicita la reposicion del inventario hasta que llega finalmente el pedido. Este es un aspecto que en el caso de ser posible de mejorar, repercutira positivamente disminuyendo los tiempos con nivel de inventario negativo y por ende la disminucion del Costo de Escasez.
Finalmente, la combinación de parámetros optima seria con una politica de inventario (z = 10 y Z = 40) teniendo una tasa de arribo de $\sigma = 0{,}15$

References

[1] Teoría de Colas
https://es.wikipedia.org/wiki/Teor%C3%ADa_de_colas

[2] Modelos de Inventario - MECALUX
https://www.mecalux.com.ar/blog/modelos-de-inventario

[3] Simulation modeling and analysis - Law Kelton 1986

[4] Simulation-Modeling-and-Analysis-Averill-M.-Law-Edisi-5-2014