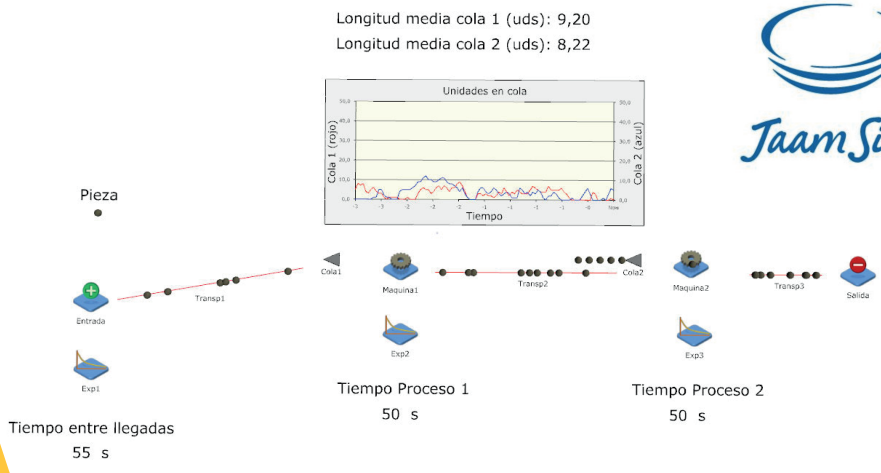
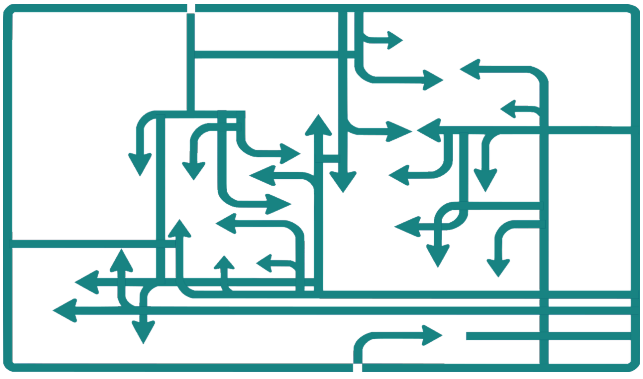
Introducción a la simulación con JaamSim

Joan Lario Femenia
Pascual Cortés Pellicer



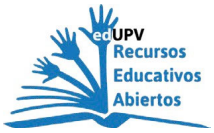


edUPV
Universitat Politècnica de València

Joan Lario Femenia
Pascual Cortés Pellicer

Introducción a la simulación con JaamSim

edUPV
Universitat Politècnica de València



http://tiny.cc/edUPV_rea

Colección *Académica* http://tiny.cc/edUPV_aca

Para referenciar esta publicación utilice la siguiente cita:
Lario Femenia, Joan; Cortés Pellicer, Pascual (2025). *Introducción a la simulación con JaamSim.* edUPV. https://doi.org/10.4995/REA.2025.681901

Autoría
Joan Lario Femenia
Pascual Cortés Pellicer

Editorial
2025, edUPV
Ref: 6819_01_01_01

ISBN: 978-84-1396-001-2
DOI: https://doi.org/10.4995/REA.2025.681901

Autores

JOAN LARIO FEMENIA

Profesor asociado en el Departamento de Organización de Empresas de la Universitat Politècnica de València (UPV) e investigador en el Centro de Investigación Gestión e Ingeniería de Producción (CIGIP) de la UPV. Su trayectoria académica incluye la docencia en el Grado en Ingeniería de Organización Industrial y en distintos programas de máster de la UPV.

Cuenta con más de diez años de experiencia en la industria, habiendo trabajado en multinacionales del sector del automóvil (Johnson Controls) y biomédico (ZimmerBiomet) como ingeniero de procesos e industrialización.

En el ámbito de la investigación, se especializa en el desarrollo de materiales metálicos mediante técnicas pulvimetalúrgicas, con un enfoque particular en aleaciones beta de titanio y tratamientos superficiales para mejorar la biocompatibilidad y la resistencia a la corrosión. Es coautor de diversas publicaciones en revistas científicas indexadas y ponente en congresos internacionales. Además, es revisor en prestigiosas revistas como Corrosion Science, Journal of Materials Science & Technology y Surface and Coating Technology.

Su perfil combina una sólida experiencia industrial con una destacada trayectoria en la docencia e investigación en organización de empresas, logística y gestión de operaciones, contribuyendo al desarrollo de nuevas estrategias de gestión empresarial y mejora continua en la educación universitaria.

PASCUAL CORTÉS PELLICER

Profesor Docente Investigador en el Departamento de Organización de Empresas de la Universitat Politècnica de València (UPV), especializado en logística inversa, sostenibilidad, economía circular y gestión de la cadena de suministro. Su labor docente ha sido clave en la formación de futuros profesionales en estas áreas, impartiendo asignaturas en grado y máster relacionadas con gestión empresarial, dirección de operaciones y logística.

Su producción investigadora es reconocida a nivel académico, con varias publicaciones en revistas internacionales de prestigio y presentaciones en conferencias de renombre. Sus principales líneas de investigación incluyen la logística inversa, la sostenibilidad en la cadena de suministro, la economía circular y la gestión de operaciones.

En 2021, recibió el Premio Cel Universidad a la mejor tesis doctoral en logística, un prestigioso galardón a nivel nacional que destaca la calidad y relevancia de su trabajo en este campo. Su investigación ha abordado temas cruciales como la toma de decisiones en logística inversa, la implementación de prácticas sostenibles en la cadena de suministro y el desarrollo de modelos para mejorar la eficiencia en la recuperación de productos.

Gracias a su experiencia docente y rigurosa labor investigadora, Pascual Cortés Pellicer se ha consolidado como un referente en su área, contribuyendo tanto a la formación académica de nuevos profesionales como al avance del conocimiento en gestión empresarial y dirección de operaciones logísticas.

Resumen

Esta monografía desarrolla un proceso formativo progresivo para la aplicación de la Teoría de Colas en entornos industriales mediante la simulación con JaamSim. A través de sus capítulos, se establecen las bases conceptuales y prácticas que permiten a los ingenieros diseñar, analizar y optimizar sistemas productivos con herramientas de simulación.

El documento se estructura en tres partes fundamentales. En primer lugar, se introduce la Teoría de Colas, proporcionando los conceptos esenciales sobre la gestión de sistemas de espera y su impacto en la eficiencia operativa. Posteriormente, se presenta JaamSim como una herramienta accesible y flexible para la modelización y análisis de estos sistemas, con explicaciones detalladas sobre su interfaz, funcionalidades y ejemplos prácticos de aplicación. Finalmente, se aborda la implementación de la teoría en distintos entornos de fabricación, desde modelos simples con una única máquina hasta configuraciones avanzadas con múltiples estaciones, colas limitadas y estructuras en serie.

A lo largo de la monografía, se plantea una metodología estructurada que permite avanzar desde los fundamentos teóricos hasta la aplicación práctica, aumentando progresivamente el nivel de detalle. Este enfoque facilita la comprensión de las acciones necesarias para crear entornos de simulación de colas en la industria y proporciona ejemplos concretos para la resolución de problemas típicos en el ámbito empresarial.

Además, se enfatiza cómo la simulación con JaamSim permite a ingenieros de procesos, organización o mejora continua definir y optimizar procesos productivos desde cero, sin la necesidad de conocimientos avanzados de programación. El estudio de distintos escenarios productivos posibilita la identificación de cuellos de botella, el balanceo de cargas y la optimización del flujo de producción, contribuyendo a la toma de decisiones estratégicas.

En definitiva, esta monografía constituye una herramienta práctica para ingenieros que buscan aplicar la Teoría de Colas en la simulación de entornos industriales, ofreciendo un marco didáctico para la mejora del rendimiento y la eficiencia en los sistemas de fabricación.

Prefacio

El objetivo de esta monografía es proporcionar un proceso formativo estructurado sobre la Teoría de Colas y su aplicación en entornos de fabricación mediante el uso del software de simulación JaamSim. A medida que los sistemas productivos y logísticos adquieren mayor complejidad, se hace imprescindible contar con herramientas que permitan analizar y optimizar su rendimiento. La simulación es una de las metodologías más efectivas para este propósito, ya que permite evaluar distintas configuraciones de un sistema antes de su implementación real.

La monografía sigue un enfoque progresivo, donde los conceptos y herramientas se presentan de menor a mayor nivel de detalle, facilitando la comprensión y aplicación práctica. En el primer capítulo, se introduce la Teoría de Colas como una metodología clave en la gestión de sistemas de espera y en la mejora de la eficiencia operativa. Se explican los fundamentos matemáticos, las principales distribuciones estadísticas y su aplicabilidad en distintos sectores industriales.

El segundo capítulo profundiza en JaamSim, abordando su estructura, características y ventajas como herramienta de modelado y simulación. Se presentan ejemplos de uso, explicaciones detalladas sobre su interfaz gráfica y una guía práctica para la construcción de modelos sin necesidad de conocimientos avanzados en programación.

En el tercer capítulo, se exploran distintos entornos de fabricación, desde configuraciones básicas con una única máquina hasta sistemas más complejos con múltiples estaciones de trabajo, colas limitadas y procesos en serie. Se analizan escenarios que permiten identificar cuellos de botella, balancear cargas y optimizar los flujos productivos mediante la simulación. Este capítulo no solo ilustra la aplicación práctica de la Teoría de Colas, sino que también plantea soluciones a problemáticas comunes en el ámbito industrial.

Esta monografía está concebida como una herramienta práctica para ingenieros de procesos, organización o mejora continua, proporcionando un marco de referencia para quienes buscan diseñar o redefinir procesos productivos desde cero. Gracias a la flexibilidad de JaamSim, los usuarios pueden simular diferentes escenarios sin incurrir en elevada complejidad de programación, permitiendo evaluar mejoras antes de su implementación real.

Índice

Índice de figuras

Índice de tablas

1
Introducción a la teoría de colas

La teoría de colas es una rama de la investigación operativa que se centra en el análisis matemático y la modelización del comportamiento de las colas o líneas de espera (Pedro et al., 2016). Su objetivo principal es estudiar y comprender los factores que influyen en la formación y el funcionamiento de estas colas para optimizar los sistemas que las generan (Mora, 2011). Esta teoría tiene un amplio espectro de aplicaciones en sectores como el comercio, la industria, las telecomunicaciones, el transporte, la informática, la logística y la gestión empresarial.

En términos generales, una cola surge cuando hay un desequilibrio temporal entre la demanda de un servicio y la capacidad del sistema para atender dicha demanda. Los clientes en este contexto pueden ser personas, máquinas, mercancías, datos o cualquier otro elemento que necesite procesarse o gestionarse. La teoría de colas ofrece herramientas para medir y optimizar variables clave, como el tiempo de ciclo, la trabajo en curso, la producción, la utilización del sistema o el nivel de servicio.

Formación de colas

Las colas se forman cuando la capacidad de un sistema es insuficiente para atender la demanda de manera inmediata. Estas colas pueden encontrarse en contextos como:

- Negocios y comercio: clientes esperando para ser atendidos en un supermercado o una tienda.
- Logística: contenedores en espera para ser embarcados o descargados.
- Ingeniería: mantenimiento de maquinaria en una planta de producción o almacenamiento de piezas en espera de ser procesadas.

- Telecomunicaciones: datos esperando ser transmitidos por una red.
- Transporte: pasajeros en espera de un tren o avión.

Utilidad de la teoría de colas

El análisis de las colas tiene como fin optimizar el diseño y la operación de sistemas donde ocurren estos fenómenos. Esto implica encontrar un equilibrio eficiente entre los costes asociados a la provisión del servicio y los costes asociados a la espera. Por ejemplo:

- **Capacidad excesiva:** si el sistema cuenta con más recursos de los necesarios (personal, máquinas, espacio), se incurre en gastos innecesarios.
- **Capacidad insuficiente:** si el sistema no puede manejar la demanda adecuadamente, aumentan los tiempos de espera, lo que puede generar pérdidas económicas y deterioro en la calidad del servicio.

Los modelos de colas o líneas de espera permiten analizar escenarios para tomar decisiones estratégicas, como determinar el número óptimo de servidores, priorizar ciertos tipos de clientes o calcular la capacidad necesaria para evitar congestiones.

Aplicación de JaamSim en el estudio de colas

El software JaamSim es una herramienta poderosa que permite modelar y simular sistemas de colas, ofreciendo una representación visual y matemática de cómo operan estos sistemas. A través de JaamSim, es posible:

- Simular escenarios reales: permite analizar cómo funciona un sistema bajo distintas configuraciones y niveles de demanda.
- Probar soluciones: los usuarios pueden ajustar parámetros como la capacidad de servicio o la tasa de llegada de clientes para evaluar el impacto de cambios sin necesidad de implementarlos en el mundo real.
- Optimizar recursos: Proporciona datos clave para tomar decisiones que minimicen los tiempos de espera y los costos operativos.

Gracias a herramientas como JaamSim, los estudiantes de Ingeniería de Organización Industrial pueden comprender de forma práctica los principios de la teoría de colas, experimentando con modelos dinámicos y desarrollando habilidades críticas para diseñar sistemas eficientes en el ámbito industrial.

1.1. Características de los sistemas de colas

El estudio de sistemas de colas es una herramienta fundamental en el análisis y optimización de procesos donde la demanda de un servicio no puede ser satisfecha inmediatamente por el sistema. Para describir y modelar un sistema de colas de manera precisa, es necesario analizar seis características clave que definen su funcionamiento. Estas características sirven como base para desarrollar modelos matemáticos y simulaciones en herramientas como JaamSim, utilizadas para evaluar y optimizar sistemas en contextos industriales y organizacionales.

1.1.1. Fuente de llegada de clientes

La fuente de llegada de clientes determina cómo los usuarios (personas, productos, datos, etc.) acceden al sistema. En la mayoría de los casos, las llegadas son estocásticas, es decir, se rigen por una distribución probabilística. El patrón típico para un sistema básico de colas supone una distribución de Poisson para modelar las llegadas de los clientes a intervalos aleatorios, aunque otras distribuciones también pueden aplicarse dependiendo del contexto. Las llegadas pueden ser clasificadas como:

- Estacionarias o deterministas: mantienen una tasa constante a lo largo del tiempo.
- No estacionarias o estocásticas: la tasa de llegadas varía con el tiempo (por ejemplo, durante las horas pico en un supermercado).
- Independientes: los clientes llegan uno a uno.
- En lotes: llegan grupos de clientes al mismo tiempo, lo que requiere definir una distribución específica para los tamaños de los lotes.

Además, se debe considerar el fenómeno de la impaciencia, donde los clientes pueden abandonar la cola si perciben que el tiempo de espera es excesivo, afectando las métricas del sistema.

En JaamSim, los usuarios pueden definir patrones de llegada personalizados, modelar llegadas no estacionarias y representar fenómenos como la impaciencia mediante eventos condicionales.

1.1.2. Patrón de servicio de servidores

El patrón de servicio describe cómo los clientes son atendidos una vez ingresan al sistema. Este aspecto incluye múltiples variantes que afectan directamente la eficiencia y el rendimiento del sistema:

- **Variabilidad del tiempo de servicio**: los tiempos de servicio pueden ser constantes (todos los clientes reciben el mismo tiempo de atención) o variables, descritos mediante una función de probabilidad. Ejemplo: en una cadena de montaje, los tiempos de servicio pueden ser constantes, mientras que en una consulta médica varían según las necesidades del paciente.
- **Atención individual o por lotes**: los servidores pueden atender a un cliente a la vez o a un grupo simultáneamente, como en sistemas de transporte donde los pasajeros abordan en conjunto.
- **Dependencia del número de clientes**: en algunos sistemas, los servidores ajustan su ritmo dependiendo del número de clientes en la cola, acelerando o ralentizando el servicio. Este fenómeno, llamado patrón de servicio dependiente, es típico en operaciones humanas donde se prioriza la reducción de tiempos de espera.

- **Estacionariedad del servicio**: al igual que las llegadas, el tiempo de servicio puede ser estacionario (constante) o no estacionario, variando con factores como la hora del día o la carga de trabajo acumulada.

En JaamSim, es posible definir patrones de servicio flexibles y configurar múltiples servidores para atender en paralelo o por lotes, simulando escenarios reales con alta fidelidad.

1.1.3. Disciplina de cola

La disciplina de cola establece las reglas que determinan el orden en el que los clientes son seleccionados para recibir servicio (Pascual, 2011). Estas reglas influyen en la experiencia del cliente y en la eficiencia del sistema. Las disciplinas más comunes son:

- **FIFO (First In, First Out)**: se atiende a los clientes en el orden de llegada. Es la regla más sencilla y ampliamente utilizada en sistemas básicos, como colas de supermercado o ventanillas de banco.
- **LIFO (Last In, First Out)**: se prioriza al cliente que llegó más recientemente, como en un sistema de apilamiento. Este enfoque es menos común en servicios humanos pero frecuente en operaciones con objetos físicos.
- **EDD (Earliest Due Date)**: prioriza los elementos o clientes en función de su fecha límite de entrega más cercana. Este enfoque es útil en sistemas donde cumplir con los plazos establecidos es crucial, ya que minimiza la posibilidad de retrasos en las entregas.
- **SOT (Shortest Operation Time)**: selecciona a los clientes o tareas que requieren el menor tiempo de operación o servicio primero. Esta regla es efectiva para minimizar el tiempo promedio de espera y aumentar la eficiencia del sistema, ya que las tareas más rápidas se completan primero, liberando recursos rápidamente.
- **RSS (Random Selection for Service)**: los clientes se seleccionan de forma aleatoria. Este modelo es utilizado en sistemas donde la prioridad no está claramente definida.
- **Reglas basadas en prioridades**: en algunos sistemas, los clientes se atienden según criterios específicos, como el tipo de servicio requerido, el tiempo estimado de atención o el nivel de urgencia. Ejemplo: en hospitales, los casos más graves se atienden primero.
- **Processor Sharing**: todos los clientes reciben atención simultáneamente, dividiendo la capacidad del sistema entre ellos. Este enfoque es común en sistemas computacionales donde los recursos se comparten entre múltiples procesos.

Con JaamSim, los usuarios pueden configurar diversas disciplinas de cola, experimentar con diferentes reglas y evaluar su impacto en métricas como el tiempo promedio de espera y la utilización del sistema.

1.1.4. Capacidad del sistema

La capacidad del sistema se refiere al número máximo de clientes que pueden permanecer en el sistema, ya sea en espera o recibiendo servicio. Puede clasificarse en:

- **Capacidad infinita**: no hay restricciones en la cantidad de clientes. Este modelo es útil para análisis teóricos, pero poco realista en muchos casos prácticos.
- **Capacidad finita**: existe un límite en el número de clientes. Este límite puede deberse a restricciones físicas (como espacio) o normativas (como el número máximo permitido en un área).

Cola con bloqueo

La capacidad finita también simula de manera efectiva la impaciencia, ya que los clientes que no pueden ingresar al sistema abandonan automáticamente. JaamSim permite definir límites de capacidad y simular el impacto de estas restricciones en la dinámica del sistema.

1.1.5. Número de canales del servicio o número de servidores

El número de canales de servicio define cuántos servidores están disponibles para atender a los clientes. Los principales tipos son:

- **Canal único**: un único servidor atiende a todos los clientes de manera secuencial. Este enfoque es típico en sistemas simples con baja demanda.
- **Canales paralelos**: varios servidores trabajan simultáneamente, recibiendo clientes de una sola cola compartida. Este modelo es eficiente para reducir tiempos de espera y equilibrar cargas de trabajo.
- **Canales independientes**: cada servidor tiene su propia cola, lo que puede generar variaciones significativas en los tiempos de espera dependiendo de la distribución de los clientes.

JaamSim permite modelar estos escenarios y evaluar configuraciones óptimas para distintos niveles de demanda.

1.1.6. Número de etapas de servicio

El número de etapas de servicio describe la secuencia de actividades que un cliente debe completar antes de abandonar el sistema. Los sistemas pueden ser:

- **Unietapa**: el cliente recibe un único servicio antes de salir del sistema.
- **Multietapa**: el cliente pasa por múltiples fases consecutivas. Por ejemplo, en una fábrica, un producto puede requerir varias operaciones antes de estar listo.

En sistemas multietapa, puede haber:

- **Vuelta atrás o reciclaje**: los clientes regresan a etapas anteriores por necesidad de reprocesamiento, común en sistemas de control de calidad.

JaamSim permite modelar tanto sistemas unietapa como multietapa, incluyendo rutas complejas y dinámicas de reciclaje.

Estas seis características son esenciales para comprender y modelar cualquier sistema de colas. Utilizando herramientas como **JaamSim**, los estudiantes de Ingeniería de Organización Industrial pueden experimentar con diferentes configuraciones, realizar simulaciones detalladas y optimizar sistemas reales basados en datos confiables y análisis visuales. Este enfoque prepara a los futuros ingenieros para enfrentar desafíos operativos con soluciones efectivas e innovadoras.

1.2. Notación básica

En 1953, David Kendall propuso una notación estándar para describir y clasificar los sistemas de colas de manera concisa y sistemática. Esta notación, conocida como la notación A / B / m / b / E / F, se utiliza ampliamente en la teoría de colas para identificar las características principales de diferentes modelos. Comprender esta notación es fundamental para modelar y analizar sistemas en herramientas como JaamSim, que permite simular diversos escenarios basados en estos parámetros.

A / B / C / D / E / F

A: ley de llegadas al sistema

B: ley de los tiempos de servicio

C: número de canales o servidores (s) en paralelo, supuestamente iguales

D: ley de los tiempos de servicio o disciplina de Cola, generalmente si es FIFO se omite

E: capacidad de la cola, número máximo de unidades que pueden encontrarse simultáneamente en el interior del sistema

F: tamaño del centro emisor

A: ley de llegadas al sistema

Este componente define cómo los clientes llegan al sistema. En la mayoría de los casos, las llegadas se modelan como un proceso estocástico, en el que los tiempos entre llegadas se distribuyen exponencialmente, y el número de llegadas por unidad de tiempo sigue una distribución de Poisson. Este comportamiento se conoce como un proceso markoviano. Cuando las llegadas son estacionarias e infinitas, se utiliza la notación M (por "Markoviano"). El parámetro principal que describe la tasa de llegadas es λ (lambda), que representa el número promedio de llegadas por unidad de tiempo. En casos prácticos, la tasa puede variar si las llegadas son no estacionarias o si los clientes provienen de una fuente finita.

B: ley de los tiempos de servicio

Este componente describe cómo los servidores procesan a los clientes dentro del sistema. Cuando el servicio se distribuye exponencialmente, se utiliza también la notación M. El parámetro que caracteriza esta ley es μ (mu), que representa el número promedio de unidades que un servidor puede atender por unidad de tiempo. En sistemas más

complejos, pueden usarse otras distribuciones para modelar tiempos de servicio no constantes, ajustándose a las necesidades del sistema.

C: número de canales o servidores

Esta letra especifica cuántos servidores trabajan en paralelo en el sistema, y se representa como un valor entero. Por ejemplo, 1 indica un sistema con un único servidor, mientras que s indica varios servidores trabajando simultáneamente. Los servidores suelen ser idénticos, y el modelo asume que cada uno tiene la misma capacidad de procesamiento.

D: disciplina de cola

La disciplina de cola describe las reglas que determinan el orden en que los clientes son atendidos. La disciplina más común es FIFO (First In, First Out), donde los clientes son atendidos en el orden en que llegan. También pueden utilizarse otras reglas, como LIFO (Last In, First Out), en la que el cliente más reciente es atendido primero, o SIRO (Service In Random Order), en la que los clientes son seleccionados al azar. En algunos casos, se emplean reglas específicas de prioridad, como atender primero a los clientes con mayor urgencia o menor tiempo de servicio estimado.

E: capacidad del sistema

La capacidad del sistema define el número máximo de clientes que pueden estar dentro del sistema en un momento dado, incluyendo aquellos que esperan en la cola y los que están siendo atendidos. Si la capacidad es infinita, no existe un límite para el número de clientes en el sistema. Por otro lado, en sistemas con capacidad finita, este límite se especifica explícitamente en la notación. Una capacidad limitada puede reflejar restricciones físicas (como espacio) o políticas operativas.

F: tamaño del centro emisor

Este componente describe si la fuente de clientes es finita o infinita. En una fuente infinita, las llegadas al sistema son independientes del número de clientes que ya se encuentran dentro de él. Por el contrario, en una fuente finita con k clientes, la tasa de llegadas depende del número de clientes que ya están en el sistema. Esto implica que a medida que más clientes ingresan al sistema, disminuye la probabilidad de nuevas llegadas, lo cual puede ser crucial en aplicaciones donde los recursos son limitados.

La notación de Kendall es fundamental para estructurar y modelar sistemas de colas en herramientas como JaamSim, al facilitar la configuración de parámetros clave que corresponden directamente con las opciones del software. Esta notación permite definir patrones de llegada (A) y tiempos de servicio (B) mediante distribuciones como la exponencial o personalizada, lo que ayuda a analizar la variabilidad en el desempeño del sistema.

El componente C, que indica el número de servidores, se traduce en la capacidad de JaamSim para simular sistemas con diferentes configuraciones de recursos, permitiendo evaluar el impacto de la variación en los servidores sobre tiempos de espera y utilización. Asimismo, la personalización de la disciplina de cola (D), como FIFO o LIFO, permite explorar diversas estrategias de gestión de colas y su efectividad en escenarios

específicos. Finalmente, la notación también incluye la posibilidad de modelar capacidades limitadas (E) y fuentes finitas (F) en JaamSim, lo cual es esencial para analizar restricciones reales como límites físicos o recursos disponibles. Estas herramientas y configuraciones hacen de la notación de Kendall y JaamSim un recurso integral para optimizar sistemas de colas en contextos industriales y organizativos.

1.3. Distribución estadística en teoría de colas

Las distribuciones de probabilidad son herramientas matemáticas esenciales para modelar la variabilidad y aleatoriedad inherentes en los sistemas de colas. Estas distribuciones permiten describir eventos aleatorios como la llegada de clientes, la duración del servicio o los tiempos entre eventos. A continuación, se detallan los fundamentos teóricos de las principales distribuciones y se amplían los ejemplos industriales para contextualizarlas en escenarios reales.

1.3.1. Distribuciones de probabilidad tipo discreto

Las distribuciones discretas se aplican a situaciones donde las variables toman valores de un conjunto finito o numerable, como conteos de eventos en un intervalo de tiempo.

1.3.1.1. Distribución de Bernouilli

La distribución de Bernouilli modela eventos binarios, donde solo hay dos posibles resultados: éxito (1) o fracaso (0), con probabilidades "p" y "q=1−p", respectivamente. Su media es "p" y su varianza es "pq".

Ejemplo Benouilli:

En una planta de ensamblaje de productos electrónicos, cada componente debe pasar por una inspección de calidad. Durante esta inspección, cada componente tiene una probabilidad "p" de pasar (éxito) y q=1−p de fallar (fracaso). La distribución de Bernouilli permite modelar este proceso para analizar la proporción esperada de componentes defectuosos en una muestra aleatoria. Este análisis ayuda a identificar puntos débiles en la cadena de producción.

1.3.1.2. Distribución binomial

La distribución binomial es una extensión de la de Bernouilli y mide la probabilidad de obtener "k" éxitos en "n" ensayos independientes, con una probabilidad "p" de éxito en cada ensayo. Su media es "np", y su varianza es "npq".

Ejemplo binomial:

En un almacén logístico, un supervisor selecciona aleatoriamente 20 paquetes para inspección. Cada paquete tiene una probabilidad del 90% (p=0.9) de cumplir con los estándares de empaque. La distribución Binomial puede usarse para predecir la probabilidad de que al menos 18 paquetes pasen la inspección. Esto permite evaluar la calidad del empaque general antes de su envío a los clientes.

1.3.1.3. Distribución geométrica

La distribución geométrica mide la probabilidad de que el primer éxito ocurra en el n-ésimo intento. Su media es 1/p, y su varianza es q/p^2.

Ejemplo geométrica:

En un sistema de transporte automatizado en una fábrica, los sensores verifican la correcta posición de las cajas en una cinta transportadora. Si la probabilidad de que una caja esté correctamente alineada es p=0.8, la distribución Geométrica puede predecir cuántas verificaciones serán necesarias, en promedio, para encontrar la primera caja mal alineada. Este análisis ayuda a ajustar la configuración del sensor para minimizar errores.

1.3.1.4. Distribución de Poisson

La distribución de Poisson modela la probabilidad de que ocurran "k" eventos en un intervalo fijo de tiempo, asumiendo que los eventos son independientes y ocurren con una tasa promedio constante (λ). Su media y varianza son iguales a λ. La mayor parte de los modelos de colas estocásticas asumen que el tiempo entre diferentes llegadas de clientes siguen una distribución exponencial. O lo que es lo mismo que el ritmo de llegada sigue una distribución de Poisson.

Ejemplo Poisson:

En un centro de atención telefónica, los agentes reciben llamadas entrantes de clientes. Si el promedio es de 30 llamadas por hora (λ=30), la distribución de Poisson puede predecir la probabilidad de recibir exactamente 25 llamadas en una hora. Esto permite dimensionar el número de agentes necesarios para evitar tiempos de espera excesivos.

1.3.2. Distribuciones de probabilidad tipo continuo

Las distribuciones continuas se aplican cuando las variables pueden tomar cualquier valor dentro de un rango, como tiempos o longitudes.

1.3.2.1. Distribución uniforme continua

La continua uniforme toma valores equiprobables en un determinado rango [a,b]. La media de esa función es (a+b)/2 y la varianza es (b-a)^2/12.

Ejemplo uniforme continua:

En una empresa de transporte de mercancías, los camiones que entregan productos suelen llegar dentro de un intervalo de 20 a 40 minutos. Si no hay un patrón preferido dentro de este rango, la distribución uniforme continua puede modelar los tiempos de llegada para planificar la asignación de personal de descarga.

1.3.2.2. Distribución exponencial

La exponencial (o negativa exponencial) es la complementaria de la distribución de Poisson. Su media es 1/λ y la varianza es 1/λ^2. Se utiliza en teoría de colas para expresar el tiempo que transcurre entre dos ocurrencias consecutivas de eventos independientes.

1.3.2.3. Erlang

La Erlang [k,β] es una distribución que es la suma de k exponenciales de media β/k. La media de dicha distribución es β y la varianza es β^2/k. De hecho, la distribución Erlang es una parte de una clase más amplia que son las distribuciones gamma. Cada función gamma es definida por dos parámetros α y β. La media es βα y la varianza es αβ^2.

1.3.2.4. Weibull

La distribución Weibull es la que habitualmente se utiliza para describir el tiempo que transcurre entre dos averías consecutivas de la misma máquina, mientras que la distribución logNormal se utiliza para describir el tiempo que se utiliza para la reparación de las máquinas.

1.3.2.5. log-Normal

La distribución log-Normal se utiliza para describir el tiempo que se utiliza para la reparación de máquinas. Una variable X se dice que tiene una distribución log-Normal si los logaritmos neperianos de sus valores están normalmente distribuidos.

La elección de la distribución adecuada depende de las características del proceso que se desea modelar. En ambientes industriales, el análisis estadístico de datos históricos, combinado con técnicas de validación de hipótesis, permite identificar la distribución que mejor se ajusta a la realidad observada. Esto mejora la precisión de los modelos de simulación y facilita la optimización de los recursos.

En JaamSim, estas distribuciones pueden configurarse para modelar llegadas, tiempos de servicio y otros eventos clave. Por ejemplo, la distribución exponencial puede representar tiempos entre llegadas de clientes, mientras que la Weibull puede modelar fallos en equipos. Este nivel de personalización permite realizar simulaciones precisas para resolver problemas operativos y mejorar la eficiencia en una amplia variedad de industrias.

2
Introducción a JaamSim

JaamSim (Java Animation Modelling and Simulación) es un programa de simulación de eventos discretos que cuenta con una interfaz gráfica de usuario intuitiva basada en arrastrar y soltar, gráficos en tres dimensiones y una amplia variedad de objetos integrados para la construcción de modelos (Abdelmegid et al., 2022; Hashash et al., 2020; King et al., n.d.). Este software, diseñado bajo un enfoque orientado a objetos, se caracteriza por su rapidez y capacidad de escalabilidad para adaptarse a aplicaciones de gran envergadura. Es compatible con los sistemas operativos Windows, Linux y macOS.

Este software es de código abierto y se distribuye de manera gratuita bajo la licencia Apache 2.0. La última versión, junto con los manuales, está disponible para descarga en su sitio oficial: www.jaamsim.com. El código fuente se encuentra alojado en GitHub: www.github.com/jaamsim/jaamsim. Además, en la sección de Videos del sitio web se pueden acceder a tutoriales y presentaciones relacionadas con su uso. JaamSim incluye todas las funciones esenciales para crear modelos de simulación, tales como:

- Herramientas para iniciar y gestionar simulaciones
- Interfaz gráfica fácil de usar basada en arrastrar y soltar
- Visualización en 3D interactiva
- Procesamiento de datos de entrada y salida
- Editores y herramientas especializadas para desarrollar modelos

Adicionalmente, el programa ofrece una variedad de objetos integrados para la creación de modelos, como:

- Elementos para procesos de flujo (servidores, colas, etc.)
- Componentes para modelar procesos continuos (como integradores y controladores PID)
- Objetos de texto para etiquetas y documentación
- Gráficos para representar los resultados de la simulación
- Distribuciones probabilísticas para generar valores aleatorios
- Elementos gráficos destinados a mapas de fondo y logotipos

2.1. Ventana principal

Al iniciar JaamSim, la interfaz principal se organiza en seis áreas distintivas que facilitan la construcción y análisis de modelos de simulación. En primer lugar, el **panel de control** (**Control Panel)** se sitúa en la parte superior y ofrece herramientas para gestionar la ejecución de la simulación. Este panel se divide en una barra de menú, donde se encuentran opciones como cargar o guardar modelos, y una barra de herramientas con botones para iniciar, pausar o detener la simulación, así como para ajustar la velocidad de ejecución.

A continuación, la **ventana de vista (View Window)** presenta una representación gráfica del modelo en desarrollo. Es posible definir múltiples vistas que muestren diferentes secciones del modelo, permitiendo una interacción directa para modificar y ajustar los elementos según sea necesario. Esta ventana es esencial para visualizar en tiempo real la disposición y comportamiento de los componentes del sistema simulado.

El **constructor de modelos (Model Builder)** proporciona una paleta de objetos que pueden incorporarse al modelo mediante la técnica de arrastrar y soltar hacia la ventana de vista. Una vez añadidos, las propiedades de estos objetos pueden ser editadas en el **editor de entradas**, donde se ajustan parámetros clave y se asignan características específicas a cada elemento, asegurando que el modelo refleje con precisión el sistema que se desea simular.

Por último, el **selector de objetos** lista todos los componentes que forman parte del modelo actual, incluyendo tanto los creados por el usuario como los predeterminados del software. Estos objetos se organizan jerárquicamente según su categoría, facilitando su localización y selección. El visor de salidas complementa esta funcionalidad mostrando, en tiempo real, los parámetros de salida predefinidos para el objeto seleccionado, actualizándose continuamente a medida que avanza la simulación, lo que permite un análisis detallado del rendimiento y comportamiento del modelo.

2.1.1. Control Panel

Proporciona una serie de funciones de control de ejecución. El panel está dividido en dos partes, una parte para la barra de menú y otra para la barra de herramientas (Fig. 1). Desde la barra de menú accedemos a los principales comandos del archivo para guardar o cargar

archivos (File), abrir o cerrar cada una de las ventanas que conforman la interfaz gráfica (Tools), ver las vistas que tenemos definidas actualmente o crear nuevas vistas (View), mostrar u ocultar los ejes y la cuadrícula de coordenadas (Options) y acceder a la ayuda del programa (Help).

El lado izquierdo de la barra de herramientas contiene controles para la manipulación de la ejecución de la simulación y la vista 3D de la ventana de visualización activa. El lado derecho de la misma, muestra el estado de la ejecución de la simulación, así como el tiempo de simulación transcurrido.

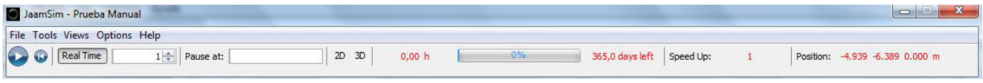


Figura 1. Panel de control de JaamSim

2.1.2. View Window

Muestra una representación gráfica del modelo de simulación. Se pueden definir diferentes ventanas de vista que representen las diferentes partes de un modelo. Cada ventana de vista puede ser una vista de un objeto y se puede modificar de forma interactiva. La posición y el formato de los objetos que muestra esta ventana por defecto (ejes, cuadrícula, título, reloj y logotipo de Ausenco) se pueden modificar a través del menú ***Input Editor*** o eliminar usando el ***Object Selector.*** El usuario puede acercar o alejar la cámara usando la rueda del ratón y mover la cámara haciendo clic y arrastrando el cursor del ratón por la ventana de la vista. Para poder modificar la posición, el tamaño o la orientación de un objeto, se debe mantener pulsada la tecla Control y arrastrar los cursores correspondientes, teniendo seleccionado el objeto a manipular. Al hacer clic con el botón derecho sobre la ventana de vista, se despliega un menú que nos permite una serie de acciones sobre el objeto seleccionado (duplicar, borrar, abrir diferentes vistas, cambiar los gráficos, añadir texto o centrar la vista).

2.1.3. Model Builder

Ofrece una selección de objetos que se pueden agregar al modelo arrastrando y soltando de la paleta de selección a la ventana de vista. Una vez que un objeto es creado desde el model builder al view window, sus características pueden ser modificadas desde el Input Editor. JaamSim incluye una serie de paletas integradas con los siguientes tipos de objetos:

- Controles de simulación
- Unidades
- Modelos de muestra
- Objetos gráficos
- Distribuciones de probabilidad
- Objetos básicos
- Flujos de proceso

- Objetos de cálculo
- Objetos fluidos

Las paletas de controles de simulación, unidades y modelos de muestra contienen objetos que no tienen una representación gráfica y no se pueden arrastrar y soltar por el usuario. Aparecen en el Object Selector, pero no en el Model Builder. El resto de paletas contienen los objetos estándar que se necesitan para muchos modelos de simulación.

2.1.4. Object Selector

Enumera los objetos que se han creado para el presente modelo, así como los que incluye automáticamente el software (Fig. 2). Los objetos se agrupan en función de su paleta, agrupados en formato de árbol según la estructura del Model Builder. Un mismo objeto específico se puede seleccionar bien haciendo clic sobre el nodo del presente menú o directamente sobre la ventana de vista. Dentro del presente menú, los objetos creados se pueden borrar (tecla Supr) o renombrar (F2 o haciendo triple clic sobre el nombre del objeto).

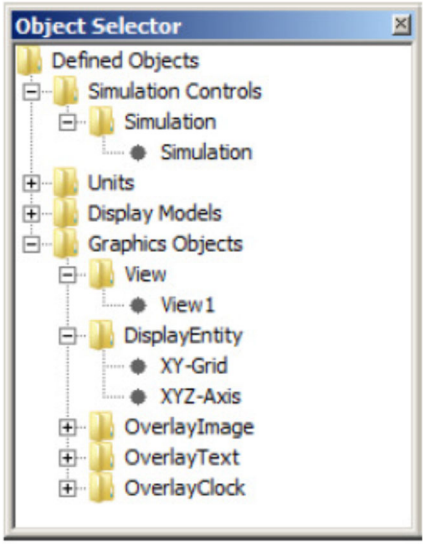


Figura 2. Object selector de JaamSim

2.1.5. Input Editor

Permite la edición de aspectos clave para los objetos del modelo, modificando inputs para los objetos ya definidos o asignando nuevas características para los definidos recientemente. Al pasar el cursor sobre alguna característica, se muestra un pequeño texto de ayuda sobre la misma y un ejemplo de entrada (Fig. 3).

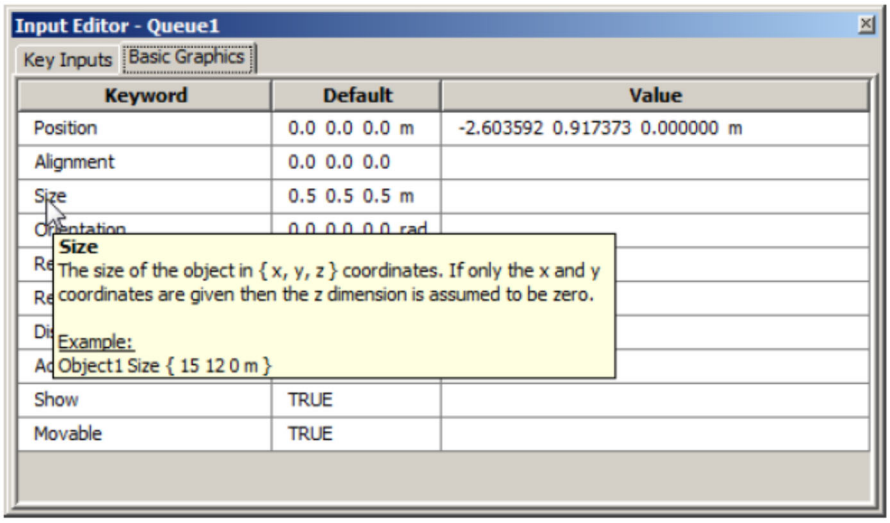


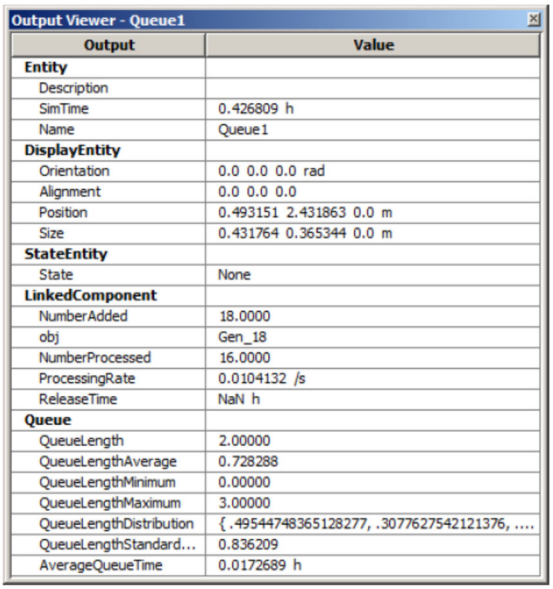
Figura 3. Input editor de JaamSim

Las características son modificadas haciendo clic en la columna del valor e introduciendo el nuevo parámetro, con las unidades correspondientes de la paleta de unidades. Los números se introducen sin espacios ni comas y las características Booleanas deben tomar el valor TRUE o FALSE, según corresponda. Al realizar una entrada en la columna valor, se sobrescribirá el valor predeterminado.

Tipos de entrada:

- **Números sin unidades:** especificado con o sin punto decimal, por ejemplo, 5, 5.0 y 5. son entradas equivalentes.
- **Los números con unidades:** una unidad de pre-definida debe ser incluida, por ejemplo, 1.000 mm, 1,0 m, y 0,001 kilómetro son equivalentes.
- **Formatos de hora:** los tiempos se pueden introducir como un número y la unidad (por ejemplo, 5,2 s), en un formato año / mes / día (por ejemplo, 02/28/2015), o en formato de hora / minuto/ segundo (por ejemplo, 01: 35: 20.5).
- **Expresiones:** especificado mediante salidas de objetos y operadores matemáticos, por ejemplo, '1 + 2 * [Entity1]. OutputB'.
- **Booleanos:** indicado como TRUE o FALSE.
- **Colores:** especificado por una palabra clave color o por un valor RGB, por ejemplo, pink y 255 192 203 son registros equivalentes.
- **Comillas:** el texto debe ir entre comillas simples si contiene espacios, por ejemplo, 'a b c'.
- **Objetos:** especificado por un nombre de objeto. Por ejemplo, View1 indica la ventana de vista, View1, creado automáticamente por JaamSim.

- **Salidas de objetos:** especificado por un nombre de objeto y un nombre de salida, por ejemplo, Queue1 QueueLength es la salida llamada "QueueLength" para el objeto de Queue1.
- Usaremos las llaves para alinear diferentes entradas en una lista. Por ejemplo, una lista de dos puntos se introduce como {0,0 1,0 0,0 m} {1,0 1,0 0,0 m}. Cuando una entrada sólo está compuesta por un único valor, no hace falta introducirlo mediante las llaves.


Output Viewer - Queue1

Output	Value
Entity	
Description	
SimTime	0.426809 h
Name	Queue1
DisplayEntity	
Orientation	0.0 0.0 0.0 rad
Alignment	0.0 0.0 0.0
Position	0.493151 2.431863 0.0 m
Size	0.431764 0.365344 0.0 m
StateEntity	
State	None
LinkedComponent	
NumberAdded	18.0000
obj	Gen_18
NumberProcessed	16.0000
ProcessingRate	0.0104132 /s
ReleaseTime	NaN h
Queue	
QueueLength	2.00000
QueueLengthAverage	0.728288
QueueLengthMinimum	0.00000
QueueLengthMaximum	3.00000
QueueLengthDistribution	{ .4954474836512827 7, .3077627542121376, ...
QueueLengthStandard...	0.836209
AverageQueueTime	0.0172689 h

Figura 4. Output viewer de JaamSim

2.1.6. Output Viewer

Muestra los parámetros de salida pre-programados que están disponibles para el objeto seleccionado. Los valores de salida se actualizan continuamente a medida que progresa la simulación (Fig. 4). Además, los atributos definidos por el usuario también aparecerán en el Visor de salida. El sistema de salida proporcionado por el software, es la base para todos los gráficos e informes del modelo.

2.2. Paleta controles de simulación

El objeto GraphicSimulation (llamado Simulation) es usado para definir parámetros básicos del modelo, como la duración de la simulación. El objeto Simulation es creado automáticamente cuando el modelo es creado, por lo que no necesita ser seleccionado por el usuario, por lo que no aparecerá en el Model Builder. En este sentido, se pueden introducir parámetros del tipo duración del periodo de inicialización n de la simulación, elaborar un informe de resumen o modificar el valor de tiempo más pequeño para el mantenimiento.

Unit Type	Default Unit	Supported Units
DimensionlessUnit		
TimeUnit	seconds (s)	min, h, d, w, y, ms, us, ns
DistanceUnit	meters (m)	m, km, nmi, mi, ft, in, mm
SpeedUnit	meters per second (m/s)	m/s, km/h, knots, mph
AccelerationUnit	meters per squared second (m/s²)	ft/s²
MassUnit	kilograms (kg)	tonnes, kt, Mt
MassFlowUnit	kilograms per second (kg/s)	(any mass unit)/(h, d, y)
VolumeUnit	cubic meters (m³)	km³, bbl, mbbl, mmbbl
VolumeFlowUnit	cubic meters per second (m³/s)	(any volume unit)/(h, d, y)
AngleUnit	radians (rad)	degrees
AngularSpeedUnit	radians per second (rad/s)	rad/h, deg/s, deg/h
EnergyUnit	joules (J)	kWh
EnergyDensityUnit	joules per cubic meter (J/m³)	kWh/m³
SpecificEnergyUnit	joules per kilogram (J/kg)	kWh/t
PowerUnit	watts (W)	kW, MW
CostUnit	dollars ($)	
CostRateUnit	dollars per second ($/s)	$/h, $/d
LinearDensityUnit	kg/m	t/m, kt/m
LinearDensityVolumeUnit	m³/m	
DensityUnit	kg/m³	
PressureUnit	Pa	kPa, psi
ViscosityUnit	Pa-s	P, cP
AreaUnit	m²	cm², mm², in²
RateUnit	/s	/h, /d, /w, /y

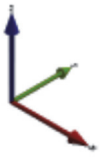
Figura 5. Paleta unidades de JaamSim

2.2.1. Paleta unidades

A continuación, se muestra unas tablas con las unidades que utilizan el programa por defecto y aquellas que pueden adoptar los diferentes tipos de unidades (Fig. 5). Al mismo tiempo, un usuario puede definir nuevas unidades con las que mostrar los valores de interés.

2.2.2. Paletas "Model Builder"

2.2.2.1. Paleta Objetos Gráficos (Graphics Objects)



Region. El objeto de región se utiliza para definir un sistema de coordenadas local. Cuando se especifica una región para un objeto, sus entradas con respecto a la posición "n" y orientación son en relación a su región o sistema local. Una vez introducidas dichas características, lo mejor es establecer para Show y Movable a FALSE para que se vuelva invisible y no se pueda mover accidentalmente. A través de sus características se puede editar la posición (position), orientación (orientation), la visibilidad (Show) y la movilidad (Movable).

DisplayEntity. Nos sirve para introducir objetos en forma de 3D o como imagen 2D en el modelo. El aspecto gráfico de un DisplayEntity y sus subclases se determina en su DisplayModel, ya que en general el DisplayEntity determina lo que se muestra y el DisplayModel determina la forma en que se muestra. La apariencia predeterminada es un cubo gris, cuyo aspecto se puede cambiar haciendo clic sobre el botón derecho en el DisplayEntity sobre la ventana de vista o bien en el "Object Selector" seleccionando "Change Graphics". El usuario puede seleccionar entre los modelos disponibles o crear uno nuevo importando un archivo en un formato compatible. Todos los objetos visualizados en el modelo de JaamSIm tienen un conjunto de características que les permite definir su aspecto: Posición (Position), Alineamiento (Alignment), Tamaño del objeto (Size), orientación (Orientation), sistema local de coordenadas (Region), objeto con el cual se puede vincular (RelativeEntity), representación gráfica del modelo (DisplayModel), la visibilidad (Show) y la movilidad (Movable).

Text. El objeto de texto se utiliza para definir texto estático o dinámico que se mostrará en la ventana de vista, para el etiquetado de diversas partes del modelo y para supervisar el estado del modelo. Se puede determinar inicialmente por su "Format Keyword", pero si lo que queremos es introducir un texto dinámico, lo podemos hacer a través de un código de formato Java en la "Format Keyword" y especificando la variable que se mostrará a través de las características "OutputName". Un objeto de texto para una entidad específica se puede generar de forma automática haciendo clic en el objeto en la ventana vista y seleccionando "Add Label".

Overlay Objects. Se trata de introducir objetos superpuestos que no se muevan cuando la vista se desplaza o hace un zoom, existiendo tres tipos: imagen, texto y reloj. Los objetos superpuestos que por defecto nos presenta en programa son el título, el reloj y el logotipo de la empresa.

BillboardText. Un objeto de texto Billboard es similar a un objeto de texto normal, excepto que el texto siempre se orienta hacia la ventana de vista y su altura se da en pixeles el lugar de metros.

Arrow. Es un objeto de flecha que se puede utilizar para señalar objetos en una ventana de vista.

Graph. Es una representación visual en tiempo real de uno o más valores de salida, mostrando su valor actual en el contexto de su historia reciente. La Figura 6 muestra un ejemplo de un gráfico, con las características relacionadas etiquetadas directamente en él. El formato se puede especificar en el objeto "GraphModel" de la "DisplayModel Keyword".

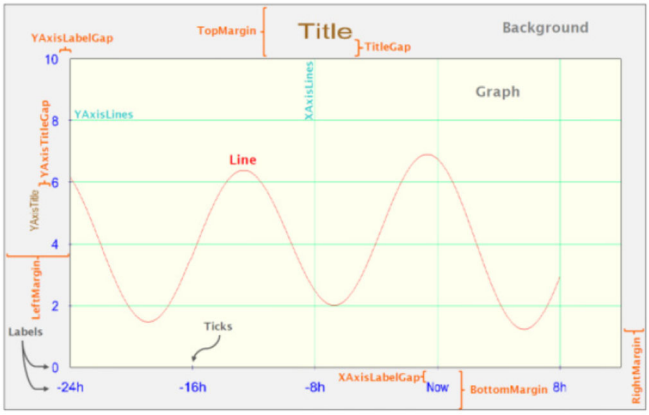


Figura 6. Gráfico de JaamSim

View. Cuando se selecciona el menú View de la barra de menú, se muestra una lista de los objetos View. Cada objeto View se compone de una cámara imaginaria situada en el mundo y una ventana en la que se mostrará la imagen. Al hacer clic en uno de estos objetos se mostrará la vista en una ventana nueva.

VideoRecorder. JaamSim incluye la funcionalidad de capturar imágenes fijas o vídeos de modelos de simulación a través del objeto Videorecorder. La longitud y la velocidad de la reproducción del vídeo se rigen por tres palabras clave:

- ***CaptureStartTime*** es el tiempo simulado en el que comienza la captura de vídeo;
- ***CaptureFrames*** es el número de fotogramas de capturar;
- ***CaptureInterval*** es el tiempo simulado transcurrido entre fotogramas.

La frecuencia de la salida vídeo está fijado a 30 fotogramas por segundo. La grabadora de vídeo se compone de las vistas seleccionadas (de captureViews) sobre una superficie con dimensiones especificadas en CaptureArea. El ejemplo que se muestra a continuación ilustra la captura de un vídeo largo de 20 segundos (en la resolución 1080p) que abarca 600 horas de tiempo simulado:

- Define Videorecorder {VideoRecorder1}
- VideoRecorder1 VideoCapture {TRUE}
- VideoRecorder1 CaptureInterval {1 h}
- VideoRecorder1 CaptureStartTime {0 h}
- VideoRecorder1 CaptureFrames {600}
- VideoRecorder1 CaptureArea {1920 1080}
- VideoRecorder1 SaveVideo {TRUE}
- VideoRecorder1 CaptureViews {Overview}
- VideoRecorder1 VideoBackgroundColor {skyblue}

El archivo producido puede ser visto con un reproductor de vídeo que soporta una amplia gama de códigos como VLC (www.videolan.org). El valor por defecto de Windows Media Player no se puede reproducir el vídeo.

2.2.3. Paleta visualización de modelos (Display Models)

La apariencia gráfica de un DisplayEntity y sus subclases se determina por su DisplayModel. Los dos conceptos trabajan juntos para generar la vista de un objeto (en general, la DisplayEntity determina lo que se muestra, mientras que su DisplayModel determina la forma en que se muestra). JaamSim comienza con un ejemplo predefinido de cada tipo de modelo de visualización. Una selección de características de visualización se puede encontrar en la paleta "DisplayModels" del Selector de objetos.

Lista de objetos:

- ***ColladaModel.*** Se utiliza para mostrar gráficos 3D en un modelo de simulación. El formato de archivo de Collada (.dae) es un formato de archivo de intercambio utilizado para gráficos 3D.
- ***ImageModel.*** Se utiliza para mostrar gráficos personalizados en 2D, como una imagen o un mapa. Tanto DisplayEntity como OverlayImage pueden aceptar un objeto ImageModel como entrada. Los tipos de archivo de imagen soportados actualmente por JaamSim incluyen BMP, GIF, JPG, PCX y PNG, o un archivo ZIP que contiene cualquiera de estos tipos de archivo.
- ***TextModel.*** Especifica el aspecto general de los objetos de texto.
- ***ScreenPointsModel.*** Se utiliza para determinar los gráficos para las entidades que aparecen como una línea simple o múltiple (EntityComveyor y EntityDelay).
- ***ArrowModel.*** Utilizado para determinar los gráficos para las flechas.
- ***GraphModel.*** Sirve para determinar el estilo de presentación y las proporciones de varios estilos de gráficos.
- ***DisplayModelCompat.*** JaamSim incluye objetos que sirven de representaciones visuales como predeterminados para objetos, comúnmente modelados en el módulo Simulador del Transporte Logístico (transporte marítimo, ferroviario y equipos terminales).

Todos estos diversos objetos de visualización de modelos tienen las mismas características clave básicas que controlan la representación y la ampliación opcional en diferentes rangos de dibujo.

2.2.4. Paleta distribuciones probabilísticas (Probability Distributions)

El siguiente conjunto de distribuciones de probabilidad estándar está disponible:

- UniformDistribution. Genera muestras con una probabilidad constante entre un valor máximo y mínimo.
- TriangularDistriction. Genera muestras de una distribución triangular entre un valor máximo y mínimo.

- NormalDistribution
- ExponentialDistribution
- NonStatExponentialDist. Genera muestras de una distribución de probabilidad exponencial negativa variable en el tiempo. Se utiliza el algoritmo de proceso no estacionario de Poisson.
- ErlangDistribution
- GammaDistribution
- BetaDistribution
- WeibullDistribution
- LogNormalDistribution
- LogLogisticsDistribution

La distribución definida por el usuario puede ser o bien un valor que varía continuamente (ContinuousDistribution) o un conjunto de valores discretos (DiscreteDistribution).

Cada distribución utiliza las siguientes características clave estándar, además de las que son específicas de cada distribución individual: Tipo de unidad para el valor devuelto por la distribución (UnitType), generados de números aleatorios (RandomSeed), valor mínimo (MinValue) y máximo (MaxValue).

También existen ciertas salidas estándar que tienen en común todas las distribuciones: valor medio calculado directamente de las entradas (CalculatedMean), la desviación estándar calculada también directamente de las entradas (CalculatedStandardDeviation), número de muestras (NumberOfSamples), el mínimo de las muestras devueltas por la distribución (SampleMin), el máximo (SampleMax), la media (SampleMean) y la desviación estándar (SampleStandardDeviation). Cada distribución incluye la palabra clave RandomSeed, que permite una secuencia pseudoaleatoria diferente de valores a ser generada para cada distribución. A menudo, se ejecuta un modelo de simulación varias veces (llamados "repeticiones") utilizando diferentes valores aleatorios para determinar la precisión estadística de los parámetros de salida. Una forma de lograr múltiples repeticiones es cambiar el RandomSeed para cada distribución, sin embargo, esto puede ser un ejercicio tedioso cuando se utilizan un gran número de distribuciones en un modelo. La palabra clave GlobalSubstreamSeed de Simulación proporciona una manera más fácil de lograr el mismo resultado.

2.2.5. Paleta objetos básicos (Basic Objects)

La paleta básica de objetos contiene un número de objetos de gran alcance que se pueden utilizar con cualquier tipo de modelo de simulación:

- **TimeSeries.** El objeto TimeSeries simula un valor numérico que varía con el tiempo. Los datos para el objeto consisten en una serie de plantillas y valores de tiempo según lo especificado por la palabra clave Value. Las plantillas de tiempo pueden ser espaciadas irregularmente y es posible repetir la serie de tiempo una y otra vez durante la simulación usando la palabra clave CycleTime. El valor devuelto por un objeto TimeSeries tiene un tipo de unidad específica especificado por la palabra clave UnitType.

Ejemplo:

Define TimeSeries { TS }

TS UnitType { SpeedUnit }

TS Value { { 2014-01-01T00:00:00 0.00 km/h } { 2014-01-03T00:00:00 0.76 km/h } { 2014-01-11T00:00:00 0.24 km/h } }

TS CycleTime { 14 d }

En este ejemplo, si la simulación se inició el 01/01/2014, llevaría a cabo el 3 de enero 0.76 km / h y 11 de enero 0,24 Km / h. El 15/01/2014, el primer valor de la lista se leería de nuevo y el valor de la TimeSeries sería 0 km / h. Los TimeSeries asumirían nuevos valores el 17 de enero y 25, y este ciclo de 14 días se repetiría hasta el final de la ejecución de la simulación.

Una entrada equivalente para la palabra clave de valor es:

TS Valor {{0 d 0,00 kmh} {2 d 0,76 kmh} {10 d 0,24 kmh}}

- **ExpressionThreshold**. Varía su estado entre abierto y cerrado en función del valor devuelto por una expresión, evaluada por la palabra clave OpenCondition. Es reevaluada con cada avance de tiempo del modelo y cuando la apertura es demandada por otra expresión.
- **EntitlementSelector**. El objeto EntitlementSelector es similar al objeto
- **DiscreteDistribution** en que devuelve un índice en el rango de 1 - N, excepto que sea incitado a devolver una selección aleatoria basada en probabilidades, hechas mediante un algoritmo basado en proporciones. El algoritmo elige el índice que minimiza la diferencia entre el número real devuelto para cada índice y el número esperado basado en la palabra clave proportions. La ProportionList se utiliza para especificar las proporciones relativas de las opciones N (1 - N).

2.2.6. Paleta objetos básicos (Basic Objects)

La paleta de flujo del proceso contiene todos los objetos necesarios para crear modelos tipo de flujo de proceso. Estos modelos se caracterizan por un ente pasivo que se pasa de un objeto a otro siguiendo un diagrama de flujo del proceso. Estos tipos de modelos se utilizan a menudo para simular un proceso de fabricación en donde las entidades representan partes que se mueven entre las estaciones de procesamiento. En esta paleta se pueden encontrar los siguientes elementos:

- **SimEntity**. El objeto SimEntity es la entidad básica que se pasa a través de un modelo de tipo de flujo de proceso. La característica principal de SimEntity es su salida llamado State que indica lo que la actividad de la SimEntity está realizando en la actualidad. Su estado se puede configurar cada vez que entra un nuevo objeto utilizando la palabra clave StateAssignment para ese objeto. El usuario es libre de elegir los nombres de estado apropiados para su uso. Tenga en cuenta, sin embargo, que los nombres de estado no pueden incluir espacios en blanco ni caracteres especiales. Un SimEntity rastrea el tiempo que permanece en cada estado durante la ejecución de la simulación. Esta información se puede registrar para cada SimEntity por el objeto EntityLogger.

- **EntityGenerator**. El objeto *EntityGenerator* crea una serie de entidades que se pasan al siguiente objeto en un proceso. La palabra clave *PrototypeEntity* identifica la entidad que desea copiar. Esta entidad puede ser cualquier tipo de objeto, sin importar su complejidad. La velocidad a la que se generan las entidades se determina por las palabras clave *InterArrivalTime* y *FirstArrivalTime*. Estas entradas tienen unidades de tiempo y puede ser un valor constante, un TimeSeries, una distribución de probabilidad, o una expresión.

- **EntitySink**. El objeto *EntitySink* destruye toda entidad entrante y nos muestra información de salida como el número de entidades recibidas, el número de entidades que han sido procesadas, la velocidad a la que han sido procesadas y el tiempo que ha pasado desde que llegó la última unidad.

- **Server**. El objeto *Server* procesa una entidad entrante y luego la pasa al siguiente objeto. Las entidades que están esperando para ser procesados quedan en manos de un objeto de cola *(Queue)* identificada por el parámetro *WaitQueue*. Todas las entidades que reciba el *Server* primero pasan a través de este objeto de cola, incluso si el servidor está inactivo. Las entidades también pueden ser enviadas directamente a la cola especificada por la palabra clave *WaitQueue*. Cada vez que se añade una entidad a una cola, todos los servidores que especifican esta Cola como su *WaitQueue* serán notificados. El primer servidor que está disponible a continuación recibirá la entidad para su procesamiento. La velocidad a la que se procesan las entidades está determinada por el parámetro *ServiceTime*. Esta entrada tiene unidades de tiempo y acepta un valor constante, un *TimeSeries*, una distribución de probabilidad, o una expresión. El servidor puede ser detenido en diversas circunstancias utilizando el parámetro *OperatingThresholdList*, que especifica una lista de objetos de límite como *SignalThreshold, TimeSeriesThreshold o*

ExpressionThreshold. Todos los objetos de límite especificados deben estar abiertos para que el servidor funcione. Sin embargo, si un límite cierra mientras el servidor está procesando una entidad, se completará el trabajo de esa entidad antes de iniciar otra operación.

- **Queue**. El objeto Queue define una ubicación para las entidades de simulación que esperan mientras se procesan otras entidades de simulación. A diferencia de muchos otros objetos en esta paleta, una entidad recibida por una *Queue* no se pasa automáticamente al siguiente objeto. Se debe esperar en la cola hasta que se elimina por algún otro objeto. Las colas se utilizan de esta manera por el *Server*, *Seize, EntityGate, Assemble, Combine, Pack, UnPack, AddTo* y objetos *RemoveFrom*. Cada vez que se añade una entidad a una cola, todos los objetos que utilizan esta cola son notificados de que una nueva entidad está disponible. El primer objeto que está listo para empezar a trabajar eliminará la entidad de la cola y la procesará. Las entidades de cola pueden ser secuenciadas por un valor de prioridad opcional especificado por el parámetro *Priority,* el cual debe ser entero. En la mayoría de los casos, el valor de prioridad se especifica mediante una expresión que se evalúa cuando la entidad llega a la cola. Las entidades con los mismos valores de prioridad pueden ser secuenciados ya sea primero en entrar primero en salir (FIFO), por defecto, o como último en entrar-primero en salir (LIFO). Una entidad de cola puede estar provista de un número de identificación opcional usando el *keyword* Match.

- **EntityConveyor**. El objeto *EntityConveyor* mueve una entidad de entrada a lo largo de un camino a una velocidad fija, y luego pasa al siguiente objeto. El tiempo de viaje para la *EntityConveyor* está determinada por la palabra clave *TravelTime,* siendo este un valor constante.

- **EntityDelay.** El objeto EntityDelay retrasa una entidad entrante por una duración variable antes de pasar al siguiente objeto. El retraso se representa como el movimiento a lo largo de una línea que es similar en apariencia al objeto EntityConveyor. Se diferencia, sin embargo, en que las entidades que se mueven a lo largo de la línea pueden pasar de uno al otro lado con diferentes tiempos de retardo. La duración del retardo de cada entidad está determinada por la palabra clave Duration. Esta entrada tiene unidades de tiempo y acepta un valor constante, un TimeSeries, una distribución de probabilidad, o una expresión. Este objeto nos permite aportar cierta variabilidad, e incluso, simular mejor el comportamiento humano con respecto al conveyor.

- **Resource.** Proporciona un conjunto de unidades que pueden ser aprovechados y lanzados por los objetos Seize y Release, respectivamente. La capacidad se modela como un valor entero adimensional.

- **Assign.** El objeto Assign realiza una o más asignaciones de atributos cada vez que recibe una entidad entrante. Este objeto es el único lugar donde el valor de un atributo se puede modificar. Los atributos pueden ser asignados por el objeto Assign, por la entidad recibida o por cualquier otra entidad en el modelo. Una vez que se han realizado las tareas, la entidad recibida pasa al siguiente objeto sin demora. La asignación de atributos a realizar es especificada por la palabra clave AttributeAssignmentList. Los ejemplos de las entradas a esta palabra clave son:

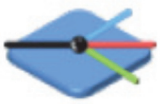
  {'This.A = this.A + 1'}

  {'This.obj.B = this.obj.B + 1'}

  {'[Entity1] .C = [Entity1] .C + 1'}

  En este ejemplo, las entradas A, B y C son los atributos (adimensional) del objeto Assign, la entidad recibida, y Entity1, respectivamente. El lado derecho de cada ecuación de asignación es una expresión para evaluar. El lado izquierdo identifica el atributo cuyo valor va a ser modificado, mediante el uso de la misma sintaxis que una expresión. Tenga en cuenta que solamente se puede asignar atributos; no es posible asignar un nuevo valor a una salida. El tipo de unidad para la expresión debe coincidir con el tipo de unidad para el atributo que se define como parte de la entrada a la palabra clave AttributeDefinitionList.

- **Branch.** El objeto Branch dirige una entidad entrante a un destino que se elige de una lista de alternativas. El valor de la palabra clave Choice determina el destino que se elija: 1 = primera rama, 2 = segunda rama, etc. La entrada de Choice puede ser un valor constante, una DiscreteDistribution, un TimeSeries, o una expresión. El uso de una expresión permite la elección que se hará en base a los atributos de la entidad entrante.



- **Duplicate.** El objeto duplicate envía copias de la entidad recibido a uno o más objetos.

- **Combine.** El objeto Combine toma una entidad de múltiples colas y pasa al siguiente objeto como una sola entidad. Cada una de las entidades deben tener el mismo valor Match en su parámetro de entrada de cada Queue. Las entidades se combinan cuando cada cola tiene al menos una entidad con el mismo valor que las otras colas. Cuando se encuentra una coincidencia, la entidad en la primera cola se pasa al objeto especificado por la palabra clave NextComponent, mientras que las entidades de las otras colas son destruidas.

- **SetGraphics.** El objeto SetGraphics se utiliza para cambiar la apariencia gráfica de una entidad especificada.

- **EntityGate.** El objeto EntityGate permite a las entidades entrantes a ser bloqueadas temporalmente. Cuando el EntityGate está abierta y la cola está vacía, las entidades entrantes pasan a través sin demora. Cuando está cerrado, las entidades entrantes se dirigen a una cola donde se llevan a cabo hasta el momento en abre la EntityGate. Cuando el EntityGate se abre, las entidades en cola son liberadas una a la vez, separadas por un retardo determinado por la palabra clave ReleaseDelay. Este retraso no se aplica a las entidades que llegan cuando el EntityGate está abierta. Sin embargo, si las entradas en cola todavía están siendo liberadas, una entidad entrante se coloca al final de la cola, aunque la EntityGate está abierta. El estado del EntityGate, ya sea abierto o cerrado, está determinado por la entrada del parámetro OperatingThresholdList.

- **EntitySignal.** El objeto EntitySignal establece el estado de un objeto SignalThreshold cuando recibe una entidad entrante. El SignalThreshold y su nuevo estado se especifican mediante las palabras clave TargetSignalThreshold y NewState, respectivamente.

- **Assemble.** El objeto Assemble combina una serie de sub-componentes en una parte montada. Los sub-componentes en espera de procesamiento se recogen en una serie de objetos de cola, identificados por el parámetro WaitQueueList, uno para cada tipo de sub-componente. Los subcomponentes entrantes deben ser enviados directamente a la cola adecuada, no al objeto de Ensamble. Si es necesario, un objeto Branch se puede utilizar para dirigir entidades entrantes a las distintas colas. El proceso de montaje comienza cuando hay al menos una entidad sub-componente en cada una de las colas. Cuando esto ocurre, el primer sub-componente en cada cola se retira y se destruye, y una nueva pieza montada se crea copiando el objeto especificado por el parámetro PrototypeEntity. El tiempo necesario para completar el proceso de montaje está dada por la palabra clave ServiceTime.

- **EntityContainer.** El objeto EntityContainer se utiliza para almacenar una colección de entidades para el procesamiento posterior como una unidad. EntityContainer es una subclase de SimEntity y en consecuencia incluye la variable de salida del Estado. Se puede transmitir a cualquier objeto que pueda aceptar un SimEntity. El objeto Pack se utiliza para colocar las entidades en EntityContainers generados. El objeto Unpack se utiliza para eliminar las entidades y para destruir la EntityContainer.

- **Pack.** El objeto Pack se utiliza para generar EntityContainers y llenarlos con entidades entrantes.  EntityContainers se crean automáticamente en la demanda del objeto Pack. El número de entidades a envasar en cada EntityContainer está determinado por la palabra clave NumberOfEntities. Una vez que el número especificado de entidades están disponibles en la cola, se embalan de una en una en el EntityContainer en el mismo orden que la cola. El tiempo para embalar cada entidad se especifica mediante la palabra clave ServiceTime.
- **Unpack.** El objeto Desembale se utiliza para eliminar las entidades de EntityContainers entrantes.  Las entidades se descomprimen una por una de la EntityContainer en mismo orden en que fueron embalados. El tiempo para desembalar cada entidad se especifica mediante la palabra clave ServiceTime. El EntityContainer recibido se destruye una vez desembalado.
- **AddTo.** El objeto AddTo se utiliza para agregar un número determinado de entidades a un EntityContainer recibido.  El número de entidades que se añaden a cada EntityContainer está determinada por la palabra clave NumberOfEntities. Una vez que el número especificado de entidades están disponibles en la cola y hay una EntityContainer espera en su cola, se añaden una por una a la EntityContainer en mismo orden que la cola. El momento de añadir cada entidad se especifica mediante la palabra clave ServiceTime.
- **RemoveFrom.** El objeto RemoveFrom se utiliza para eliminar un número dado de entidades de una EntityContainer entrante.  Entidades se eliminan una por una desde el EntityContainer en mismo orden en que estaban llenas. El tiempo para desempacar cada entidad se especifica mediante la palabra clave ServiceTime. EntityContainers se pasan a otro objeto una vez que el número especificado de entidades han sido desempaquetado.

2.3. Ejemplo Básico

Este ejemplo básico lo dividiremos en cuatro pasos:

1. Crear objetos de modelo
2. Unión entre los diferentes objetos
3. Cambio de modelos gráficos
4. Añadir distribuciones probabilísticas

2.3.1. Crear modelo de objetos

Para crear un objeto, desplegamos el menú de Process Flow situado en Model Builder y luego arrastramos y soltamos en la ventana de visualización un SimEntity (Fig.7) con un nombre predeterminado (SimEntity1), con forma de esfera y destacado en color verde, seleccionando el objeto:

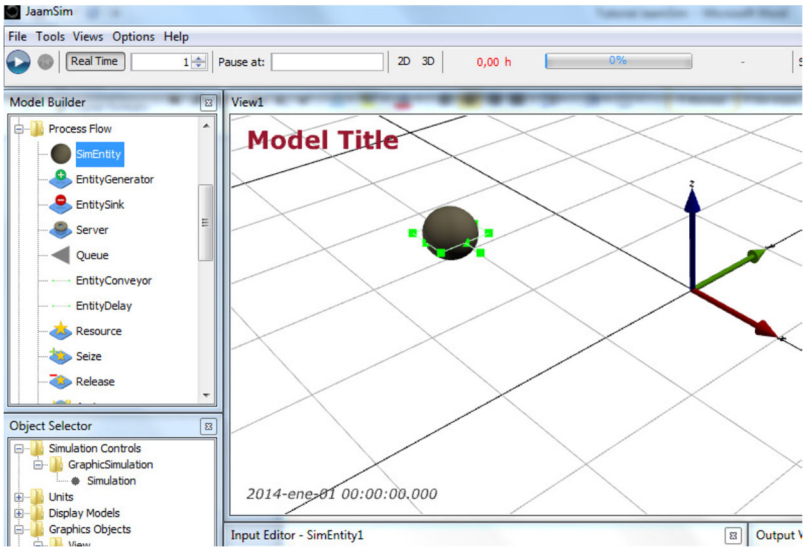


Figura 7. Ejemplo básico de JaamSim

Este objeto nos servirá como prototipo para las entidades que se manejan en el modelo, por lo que primero que haremos será cambiarle el nombre. Para ello, en el selector de objetos seleccionamos SimEntity1 y pulsamos F2 o triple clic para cambiar el nombre del objeto. Para proseguir con el ejemplo básico vamos a introducir una serie de objetos, introduciéndolo en el visor con el siguiente orden:

EntityGenerator1 – EntityConveyor – Server – EntityConveyor – EntitySink – Queue

Y los renombramos según el listado siguiente.

Object	**Name**
EntityGenerator1	Gen
EntityConveyor1	GenToServ
Server1	Serv
EntityConveyor2	ServToSink
EntitySink1	Sink
Queue1	ServQueue

De forma que el modelo quede de la siguiente manera:

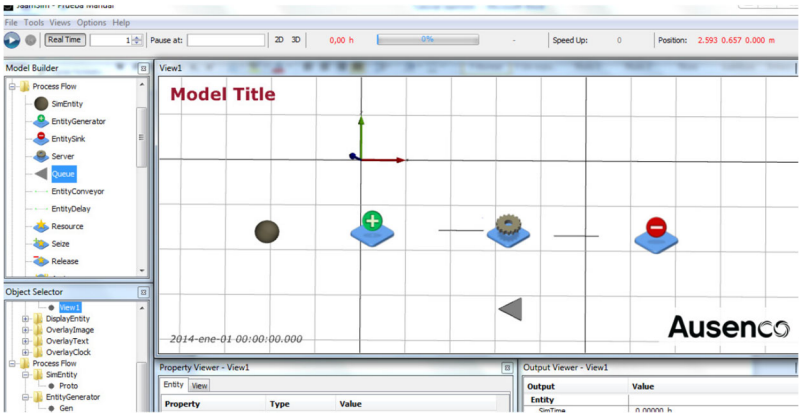


Figura 8. Ejemplo básico entidades de JaamSim

El modo de visualización puede modificarse desde el panel de control en los botones de 2D – 3D así como con la ayuda del ratón. El cambio de posición de los objetos en el visor se verá más adelante. Tal vez sea un buen momento para guardar el trabajo hecho hasta el momento, a través del menú archivo del panel de control. Se recomienda guardar regularmente el trabajo realizado.

2.3.2. Unión entre los diferentes objetos

Ahora que todos los objetos han sido colocados en el modelo, necesitamos conectarlos para interactuar entre sí para generar (gen), transportar (GenToServ, GenTosink), procesar (Serv), consumir (Sink) o almacenar (Queue).

Para poder interconectarlos, seleccionamos un objeto y dentro del menú Input Editor nos dirigimos al parámetro clave Next Component desplegando y eligiendo en el apartado de value el objeto con el que lo queremos conectar.

Object	**Keyword**	**Value**
Gen	NextComponent	GenToServ
GenToServ	NextComponent	Serv
Serv	NextComponent	ServToSink
ServToSink	NextComponent	Sink

Ahora, le añadiremos el tiempo en el que la entidad Proto será introducida en el sistema a través de Gen, así como el tiempo en el que tardará en recorrer cada paso. Los objetos tienen unos valores por defecto que tendremos que modificar introduciendo los siguientes valores en las keywords correspondientes:

Object	Keyword	Value
Gen	InterArrivalTime	2 s
GenToServ	TravelTime	1 s
Serv	ServiceTime	1 s
ServToSink	TravelTime	1.5 s

El objeto de proceso (Serv) requiere de un almacenamiento *Queue* para que esperen las piezas (*entities)* que van a ser procesadas. Por lo tanto, debemos asociar el siguiente *KeyWord* al procesador *(Serv*):

Tab	Keyword	Value
KeyInputs	WaitQueue	ServQueue

Por último, configuramos el generador de piezas (*Gen)* para que produzca copias de *Proto*, a través de la siguiente instrucción de *KeyWord:*

Tab	Keyword	Value
Key Inputs	Prototype	Entity Proto

Ahora ya sólo resta salvar el modelo y darle al Play para ver la simulación. El factor de *Real Time* controla lo rápido que transcurre el tiempo simulado en el modelo. El valor predeterminado está situado en 1, que significa que cada segundo simulado corresponde a un segundo viendo el modelo. Dicho valor se puede cambiar en las flechas del factor o introduciendo directamente el parámetro. Para seguir trabajando en el modelo se puede, o bien pausar el modelo, permitiéndonos reanudar posteriormente la simulación donde la dejamos, o bien detenerlo para resetear la simulación.

2.3.3. Cambio de modelos gráficos

A continuación, se realizarán una serie de ajustes gráficos que nos mejorará la apariencia del modelo. Si bien en este paso no afectará a la funcionalidad del modelo, los gráficos pueden resultarnos muy valiosos a la hora de confirmar el funcionamiento correcto de modelos más grandes.

Los objetos se pueden mover y cambiar el tamaño, seleccionándolos y presionando la tecla Control y arrastrando el ratón con el botón izquierdo presionado. Los podemos mover si seleccionamos directamente el objeto o redimensionarlos si presionamos sobre el borde del objeto seleccionado.

Si queremos eliminar los ejes XYZ que nos salen por defecto en la pantalla de simulación, los podemos desactivar desde las opciones del panel de control. Por último, si queremos cambiar el nombre del modelo, lo podemos hacer desde las KeyWords del Título en OverlayText de Graphics Objects del Object Selector.

2.3.4. Añadir distribuciones probabilísticas

Hasta ahora, el modelo generado presenta una secuencia uniforme. Para tratar de darle más dinamismo, se le puede añadir una distribución probabilística que determine cada cuando entra la entidad Proto al sistema.

Desde ModelBuilder expandimos el menú de Probability Distribution, seleccionamos ExponentialDistribution por ejemplo y la situamos en el modelo. En este momento, ya podemos cambiarle el nombre y definir los parámetros de la distribución exponencial.

Model	**Builder Palette**	**Object Name**
Probability Distributions	ExponentialDistribution	GenIATDist

Tab	**Keyword**	**Value**
Key Inputs	UnitType	TimeUnit
Key Inputs	MinValue	0 s
Key Inputs	MaxValue	10 s
Key Inputs	Mean	2 s

Ahora, se debe asignar al generador *Gen* dicha distribución exponencial como tiempo entre llegadas:

Tab	**Keyword**	**Value**
Key Inputs	InterArrivalTime	GenIATDist

Salvamos el modelo en este momento y simulamos. Observar como ahora, la entidad *Proto* es generada de manera aleatoria y hay veces en que las entidades se esperan en el almacén para ser procesadas.

La creación del presente ejemplo básico se puede observar a través del siguiente video tutorial: https://www.youtube.com/watch?v=KBibGXcnBMg

3
Entornos de fabricación

En el estudio de los sistemas de producción y su eficiencia operativa, la simulación juega un papel fundamental al permitir el análisis de distintos escenarios sin la necesidad de implementar cambios en el mundo real (Kloock-Schreiber et al., 2020; Xanthopoulos & Koulouriotis, 2021). En este contexto, el software JaamSim se presenta como una herramienta versátil para modelar y evaluar diversos entornos de fabricación con diferentes configuraciones y condiciones operativas (Vieira et al., 2019).

Este apartado se centra en la exploración de modelos de fabricación monomáquina y sistemas con múltiples máquinas, analizando la influencia de factores deterministas y estocásticos en su desempeño. Para ello, se presentan distintas configuraciones de modelos de colas, desde el más básico con una sola máquina y tiempos de llegada y servicio deterministas (D/D/1/∞) hasta sistemas más complejos con colas limitadas, máquinas en paralelo y estructuras en serie.

Los modelos considerados permiten analizar aspectos fundamentales de la gestión de la producción, tales como la sincronización entre la llegada y el procesamiento de piezas, la formación de cuellos de botella, el balanceo de carga en los recursos y la eficiencia del sistema. Además, se exploran variantes en las que se incorporan distribuciones probabilísticas para reflejar la incertidumbre en los tiempos de llegada y servicio, simulando escenarios más cercanos a los entornos de manufactura reales.

A lo largo del documento, cada modelo es analizado en detalle, incluyendo su configuración en JaamSim, la interpretación de los resultados obtenidos y la identificación de oportunidades de optimización. De esta manera, se proporciona un marco didáctico para que los estudiantes de Ingeniería de Organización Industrial desarrollen habilidades en el modelado y análisis de sistemas productivos, facilitando la comprensión de dinámicas operativas clave y su impacto en la eficiencia de la producción.

3.1. D/D/1/∞ entorno fabricación monomáquina con tiempos de suministro y servicio determinista

En este apartado exploraremos el uso del software JaamSim para modelar y analizar un entorno de fabricación simple titulado "Una máquina determinista". Este ejercicio tiene como objetivo familiarizar a los estudiantes de Ingeniería de Organización Industrial con los conceptos de modelado y simulación en entornos productivos mediante el uso de herramientas digitales avanzadas como JaamSim.

El modelo "Una máquina determinista (D/D/1/∞)" representa un sistema de producción básico en el que las piezas se procesan de manera secuencial a través de una única máquina bajo condiciones completamente deterministas (tanto en el suministro/entrada, cómo en la máquina/server). Este modelo sirve como base para analizar la sincronización entre entradas y salidas, el balanceo de tiempos, y la eficiencia en sistemas simples de fabricación (Fig. 9). Este ejemplo proporciona un marco simple para comprender el flujo de producción en sistemas básicos. Implementarlo en el software JaamSim permite a los estudiantes observar el impacto de las decisiones de diseño y los parámetros en el desempeño del sistema, sentando las bases para análisis más complejos en sistemas de manufactura reales.

El modelo de "Una máquina determinista" puede ser modificado o ampliado para estudiar diversos escenarios, como introducir un tiempo entre llegadas estocástico para considerar la variabilidad en las llegadas, configurar tiempos de procesamiento aleatorios para reflejar la variabilidad en el procesamiento, limitar el tamaño de la cola o incluir múltiples máquinas para simular restricciones de capacidad, y conectar varias máquinas en una línea de producción para analizar sistemas en serie. Estos aspectos se evaluarán en mayor detalle en los siguientes ejemplos.

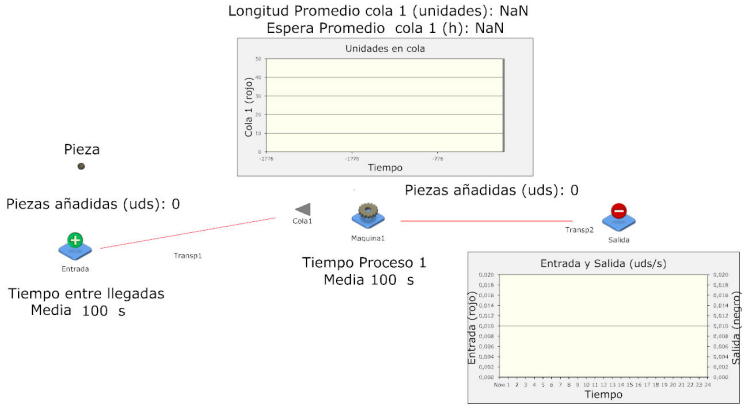


Figura 9. Modelo D/D/1/∞ en JaamSim

3.1.1. Descripción general del modelo D/D/1/∞

El modelo representado en la Fig. 9 corresponde a un sistema de fabricación básico que consta de un flujo de piezas desde una entrada hasta una salida, pasando por una máquina de procesamiento y elementos de transporte. El diseño tiene las siguientes características:

1. **Entrada (Fuente de piezas)**

   El sistema inicia con una fuente de piezas que genera elementos de manera determinista. Los intervalos entre llegadas son constantes, con una media de 100 segundos. Este comportamiento predecible garantiza un flujo uniforme de piezas hacia el resto del sistema.

2. **Transporte 1 (Transp1)**

   Una vez generadas, las piezas son transportadas desde la entrada al sistema hacia una cola específica utilizando un elemento de transporte denominado "Transp1". Este transporte conecta la etapa inicial con el área de almacenamiento temporal para la máquina 1 y permite un flujo continuo de piezas.

3. **Cola 1 (Buffer de almacenamiento)**

   Al llegar a la cola, las piezas se almacenan temporalmente mientras esperan su turno para ser procesadas en la máquina 1. Esta cola actúa como un buffer que equilibra las fluctuaciones entre la generación de piezas y la capacidad de procesamiento de la máquina.

4. **Máquina de procesamiento (Máquina 1)**

   Las piezas almacenadas en la cola son procesadas en la Máquina 1 (Server 1), que opera de forma determinista. Cada pieza requiere exactamente 100 segundos para completar su procesamiento. Esta etapa es clave para transformar las piezas según los objetivos del sistema.

5. **Transporte 2 (Transp2)**

   Tras ser procesadas, las piezas son trasladadas hacia la salida mediante un segundo elemento de transporte denominado "Transp2". Este componente asegura que las piezas lleguen al punto final del sistema sin interrupciones (Sink).

6. **Salida (Punto Final)**

   Finalmente, las piezas llegan a la salida, donde se contabilizan como unidades que han completado el proceso. Este punto marca el final del flujo del sistema y proporciona métricas clave para evaluar su desempeño.

3.1.2. Asunciones modelo D/D/1/∞

1. Los tiempos de suministro de piezas o entrada al sistema son constantes.
2. Tiempos de servicio de la máquina 1 son constantes (determinista).
3. Factor de utilización $\rho<1$.
4. El sistema está en un estado estable.

5. Los clientes son atendidos según el principio de primero en entrar, primero en salir (FIFO) en la cola disponible en la máquina 1.
6. La cola tiene una capacidad infinita.
7. Tiempo entre llegadas = 100 segundos.
8. Tiempo de servicio de la máquina 1 = 100 segundos.

3.1.3. Configuración en JaamSim del modelo D/D/1/∞

Para implementar el modelo en JaamSim, son esenciales los siguientes pasos:

1. **Crear una fuente de piezas**
   - Añadir un objeto *"EntityGenerator"* para generar las piezas
   - Configurar el tiempo entre llegadas ("*InterArrivalTime*") como un valor constante (Ej: 100 segundos)
2. **Definir el buffer (Cola)**
   - Añadir un objeto *"Queue"* para representar la cola
   - Establecer la capacidad de la cola según los requisitos (puede ser infinita)
3. **Añadir una máquina de procesamiento**
   - Usar el objeto "Server" para modelar la máquina 1
   - Configurar el tiempo de procesamiento ("*ServiceTime*") como un valor constante (Ej: 100 segundos)
4. **Configurar los elementos de transporte**
   - Conectar la fuente de piezas con la cola y la máquina usando objetos de tipo Link o *"EntityConveyor"*
   - Establecer el tiempo de transporte si es necesario ("*Traveltime*")
5. **Definir la salida**
   - Añadir un objeto *"EntitySink"* para representar la salida final
   - Configurar el contador de piezas procesadas
6. **Ajustar los parámetros de simulación**
   - Definir el tiempo total de simulación en el RunTime
7. **Agregar reportes**

Para analizar los resultados, se añade "Graphics Objects" en forma de texto para proporcionar mayor información:

- Piezas añadidas (unidades):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 1. Las tablas recopilan los valores de los parámetros necesarios para configurar cada uno de los elementos del JaamSim.

Tabla 1. Configurar un campo para la visualización "Piezas añadidas"

Object	Keyword	Value
Text_Piezas_añadidas	Format	'Piezas añadidas (uds): %.0f'
Text_Piezas_añadidas	UnitType	DimensionlessUnit
Text_Piezas_añadidas	Datasource	[Transp1].NumberAdded

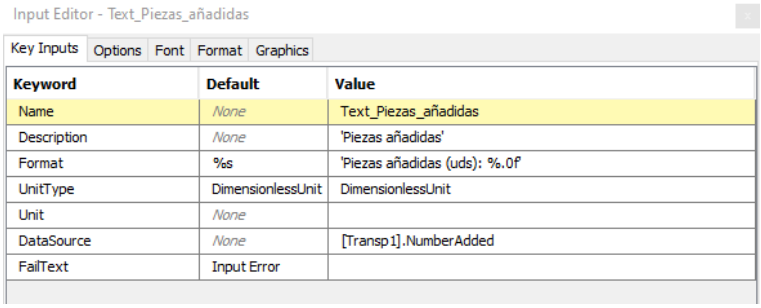


Figura 10. JaamSim Input Editor para "Piezas añadidas"

- Piezas añadidas salida (uds):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 2.

Tabla 2. Configurar un campo para la visualización "Piezas añadidas salida"

Object	Keyword	Value
Text_Añadidas_Salida	Format	Piezas añadidas salida (uds): %.0f
Text_Añadidas_Salida	UnitType	DimensionlessUnit
Text_Añadidas_Salida	DataSource	[Transp2].NumberAdded

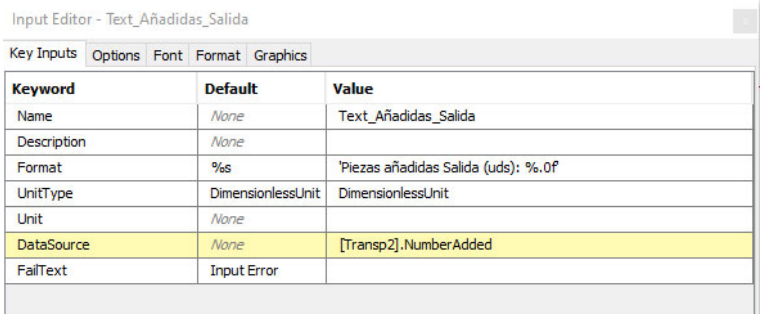


Figura 11. JaamSim Input Editor para "Piezas añadidas salida"

Tabla 3. Configuración general modelo D/D/1/∞

Object	Keyword	Value
Entrada	KeyInputs/NextComponent	Transp1
Entrada	KeyInputs/InterArrivalTime	100 s
Entrada	KeyInputs/PrototypeEntity	Pieza
Máquina1	KeyInputs/NextComponent	Transp2
Máquina1	KeyInputs/WaitQueue	Cola1
Máquina1	KeyInputs/ServiceTime	100 s
Transp1	KeyInputs/NextComponent	Máquina1
Transp1	KeyInputs/TravelTime	500 s
Transp2	KeyInputs/NextComponent	Salida
Transp2	KeyInputs/TravelTime	500 s

La tabla de configuración general del modelo (Tabla 3) para cada uno de los talleres definidos en esta monografía reúne todos los parámetros y sus valores para configura el JaamSim de cada taller. En este caso en concreto se recopila la información para un taller monomáquina determinista y constante para las llegadas y el servicio, y con cola de tamaño infinito.

3.1.4. Análisis del modelo D/D/1/∞

El modelo simula un sistema determinista ideal donde el tiempo de llegada de las piezas y el tiempo de procesamiento están perfectamente balanceados. Esto resulta en un flujo continuo sin acumulaciones ni tiempos muertos, lo que maximiza la eficiencia del sistema. Las características principales que destacan son:

- **Sin acumulación en la cola:** esto indica que no hay tiempo de inactividad en la máquina ni retrasos en el procesamiento.
- **Tasa de salida constante:** las piezas se procesan a la misma tasa a la que ingresan, lo cual es ideal para sistemas de producción en serie.

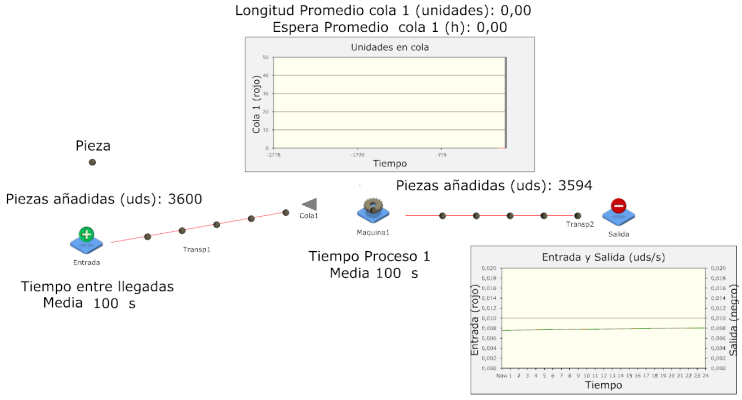


Figura 12. Resultados modelo D/D/1/∞ en JaamSim

Este ejercicio en JaamSim muestra un modelo básico pero efectivo para comprender los principios de los sistemas de producción deterministas. Permite a los estudiantes experimentar con parámetros clave como tiempos de llegada y procesamiento, identificando cómo estos afectan el desempeño del sistema. La simulación también puede extenderse introduciendo variabilidad en los tiempos o añadiendo nuevos elementos, como máquinas adicionales o cambios en las tasas de entrada.

Con este conocimiento, los estudiantes estarán mejor preparados para modelar y analizar sistemas de producción más complejos, aplicando estas herramientas para optimizar procesos reales en entornos industriales.

3.2. M/M/1/∞ entorno fabricación monomáquina con tiempos de suministro y servicio aleatorios

En los sistemas de simulación estocásticos, la incertidumbre y la variabilidad de los procesos se representan mediante distribuciones probabilísticas. El modelo de una monomáquina aleatoria (M/M/1/∞), que forma parte de estos sistemas, es una simplificación de un entorno de producción donde los tiempos de llegada y procesamiento no son determinísticos, sino aleatorios (Fig. 13). Este modelo permite explorar el impacto de la variabilidad en el desempeño del sistema, lo cual es fundamental en escenarios reales, como plantas de fabricación o cadenas de suministro.

El modelo de una máquina aleatoria se utiliza para responder preguntas críticas sobre el desempeño del sistema, como el impacto de la variabilidad, analizando cómo la aleatoriedad en los tiempos de llegada y procesamiento afecta la capacidad del sistema para manejar el flujo de piezas; la identificación de cuellos de botella, evaluando si la acumulación en la cola indica un cuello de botella en la máquina o una incapacidad para manejar altos volúmenes; la evaluación de métricas de desempeño, como la tasa de entrada y salida para determinar si el sistema mantiene un flujo balanceado, y el tiempo promedio de espera y la longitud de cola para medir la eficiencia operativa; y la optimización del sistema, considerando si es posible reducir los tiempos de espera o aumentar la tasa de salida ajustando parámetros como el tiempo medio de procesamiento.

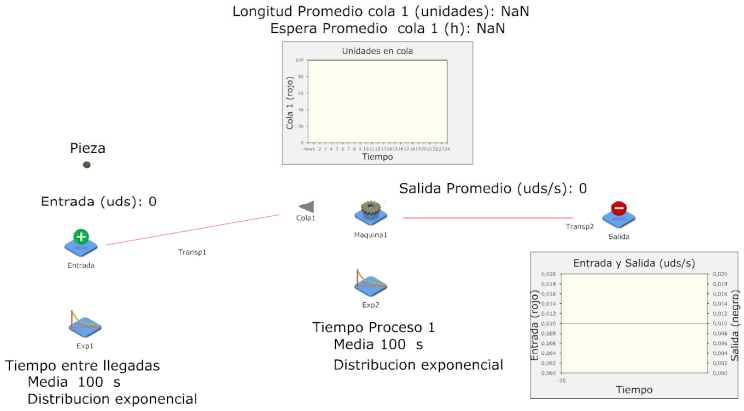


Figura 13. Modelo M/M/1/∞ en JaamSim

3.2.1. Descripción general del modelo M/M/1/∞

A continuación, se describe el modelo con mayor detalle, incluyendo sus componentes, las distribuciones utilizadas y el propósito del diseño estocástico.

1. **Entrada de elementos**

   Los elementos ingresan al sistema desde una fuente externa o una etapa previa en el proceso. La llegada de estos elementos sigue una distribución aleatoria (“Exp1”), con una distribución exponencial de media 100 segundos, que refleja intervalos variables entre cada ingreso. Este comportamiento aleatorio es común en sistemas donde las entradas son independientes y permite modelar la incertidumbre inherente en los tiempos entre llegadas.

2. **Transporte inicial**

   Una vez que los elementos ingresan al sistema, son transportados hacia un área de espera o cola (“Transp1”). Este transporte puede representarse como un flujo continuo o discreto, dependiendo del contexto, y podría involucrar mecanismos como cintas transportadoras, vehículos o sistemas automatizados (“EntityConveyor”), con un “TravelTime” de 500 segundos.

3. **Almacenamiento temporal (cola)**

   Los elementos que esperan para ser procesados se acumulan en una cola “Cola1”, cuya longitud y tiempo de espera promedio son indicadores clave del rendimiento del sistema. Estas colas tienden a fluctuar debido a la variabilidad en las llegadas y los tiempos de procesamiento, pero eventualmente pueden estabilizarse si el sistema alcanza un estado de equilibrio. La gestión eficiente de la cola es crucial para evitar acumulaciones excesivas, que pueden indicar puntos de congestión.

4. **Procesamiento de elementos**

   El procesamiento de los elementos se realiza en una etapa específica "Máquina1", donde se transforman según los requisitos del sistema. Este proceso suele tener un tiempo asociado que puede variar de manera aleatoria "Exp2", con una distribución exponencial de media 100 segundos, reflejando diferencias en las características de los elementos o en el entorno de procesamiento. La variabilidad en los tiempos de procesamiento puede generar interrupciones en el flujo continuo, dando lugar a acumulaciones temporales o períodos de inactividad.

5. **Transporte final**

   Después del procesamiento, los elementos se trasladan hacia la siguiente etapa o hacia la salida del sistema. Este transporte final se realiza mediante un elemento "EntityConveyor", cuyo tiempo de transporte "TimeTravel" es de 500 segundos.

6. **Salida del sistema**

   Los elementos procesados finalmente salen del sistema. El rendimiento del sistema se mide a través de métricas como la tasa de salida, que permite evaluar la relación entre los elementos ingresados y procesados.

3.2.2. Asunciones modelo M/M/1/∞

1. La distribución de Poisson mide el número de llegadas por unidad de tiempo y los tiempos de servicio de la máquina. La distribución del tiempo entre llegadas de un proceso de Poisson es la distribución exponencial negativa.
2. Tiempos de servicio de la Máquina 1 son aleatorios.
3. Factor de utilización $\rho<1$.
4. El sistema está en un estado estable.
5. Los clientes son atendidos según el principio de primero en entrar, primero en salir (FIFO) en la cola disponible en la Máquina 1.
6. La cola tiene una capacidad infinita.
7. Tiempo entre llegadas media = 100 segundos, con una distribución exponencial.
8. Tiempo de servicio de la máquina 1 media = 100 segundos, con una distribución exponencial.

3.2.3. Configuración en JaamSim del modelo M/M/1/∞

Para implementar el modelo descrito en JaamSim, es necesario seguir una serie de pasos que configuran cada uno de los componentes principales del sistema. A continuación, se detalla el proceso de configuración, desde la creación del entorno hasta la parametrización de las distribuciones aleatorias.

1. **Entrada de piezas**
   - Arrastra un objeto EntityGenerator desde el menú de objetos básicos.

- Configura las propiedades:
  - Inter-Arrival Time: introduce 100 s y selecciona la distribución exponencial para modelar los tiempos entre llegadas aleatorios.
  - EntityType: define un tipo de entidad, por ejemplo, "Pieza".
  - Number of Entities: configura el número total de piezas, como 878 unidades, para limitar la simulación.

2. Transporte inicial

- Arrastra un objeto Link para conectar el EntityGenerator a la siguiente etapa del flujo (la cola).
- Configura el tiempo de transporte, si corresponde:
  - Si el transporte es instantáneo, déjalo en cero.
  - Si no, puedes definir un tiempo constante o una distribución aleatoria para el transporte.

3. Cola (Buffer)

- Añade un objeto Queue.
- Conecta la salida del transporte inicial al buffer.
- Configura las propiedades:
  - **Queue Type**: selecciona "FIFO" (Primero en Entrar, Primero en Salir) como regla de prioridad.
  - **Maximum Capacity**: deja la capacidad infinita (por defecto) o establece un límite para analizar el impacto de restricciones en la cola.
  - **Statistics**: activa las métricas para monitorear la longitud promedio de la cola y el tiempo promedio de espera.

4. Máquina

- Añade un objeto Server desde el menú de objetos básicos.
- Conecta la salida de la cola al servidor.
- Configura las propiedades:
  - **Service Time**: introduce 100 s como el tiempo medio de procesamiento y selecciona la **Distribución Exponencial**.
  - **Capacity**: establece la capacidad en Máquina 1 para simular que la máquina procesa una pieza a la vez.
  - **InputQueue**: asegúrate de enlazar el buffer anterior con la cola de entrada del servidor.

5. Transporte final

- Añade un objeto Link para conectar la máquina al punto de salida.
- Configura este enlace de manera similar al transporte inicial, si corresponde.

6. Salida

- Añade un objeto EntitySink para registrar las piezas que salen del sistema.
- Conecta la salida del transporte final al EntitySink.
- Configura las propiedades:
  - **Statistics**: activa la recolección de datos para medir la tasa de salida.

7. Añadir distribuciones estadísticas a las entidades

- Añade un objeto "ExponentialDistribution", disponible en /ModelBuilder/Probability Distributions/….
- Configurar la distribución estadística según Tabla 4.

Tabla 4. Configurar un campo para la visualización "Distribución Exponencial Entidad Entrada"

Object	Keyword	Value
Key Inputs	Name	Exp1
Key Inputs	UnitType	TimeUnit
Key Inputs	RandomSeed	1
Key Inputs	Mean	100 s

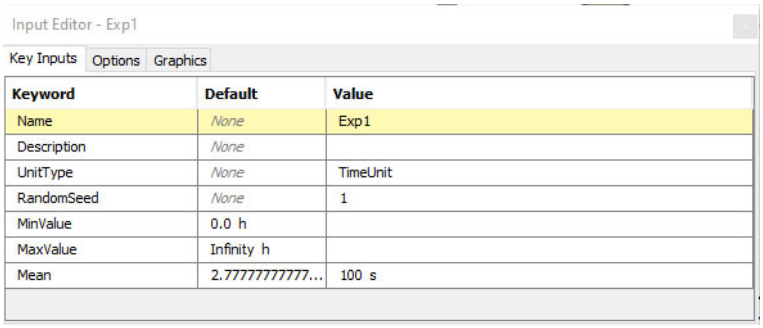


Figura 14. JaamSim Input Editor para "Distribución Exponencial Entidad Entrada"

- Conecta la distribución estadística con una entidad de "Process Flow":
  - **Statistics**: activa la recolección de datos para medir la tasa de salida.

8. Conectar los componentes

Use el modo de edición para conectar visualmente los componentes en JaamSim:

- Conecte Source a Entity Generator (Bernoulli).
- Desde el Bernoulli, dibuje conexiones hacia la QueueConecte cada Queue a su respectivo Server.
- Finalmente, conecte los Servers a la Sink.

9. Agregar reportes

Para analizar los resultados, añade "Graphics Objects" en forma de texto para proporcionar mayor información:

- Longitud promedio cola 1 (unidades):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 5.

Tabla 5. Configurar un campo para la visualización "Longitud promedio cola1"

Object	Keyword	Value
Text 1	Format	Longitud promedio cola 1 (unidades): %.2f
Text 1	Datasource	[Cola1].QueueLengthAverage

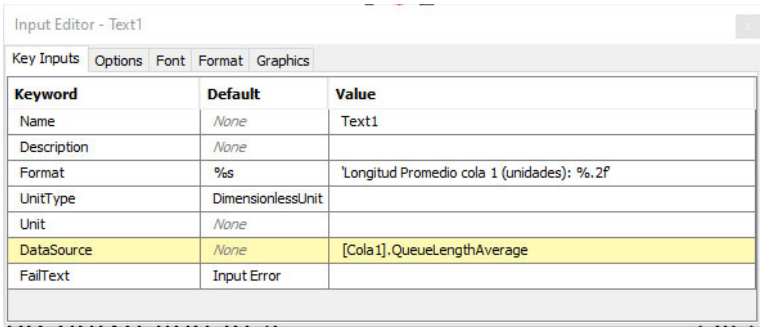


Figura 15. JaamSim Input Editor para Longitud promedio cola1

- Espera promedio cola 1 (h):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 6.

Tabla 6. Configurar un campo para la visualización "Espera promedio cola 1"

Object	Keyword	Value
Text 2	Format	Espera promedio cola 1 (h): %.2f
Text 2	UnitType	TimeUnit
Text 2	Unit	h
Text 2	DataSource	[Cola1].AverageQueueTime

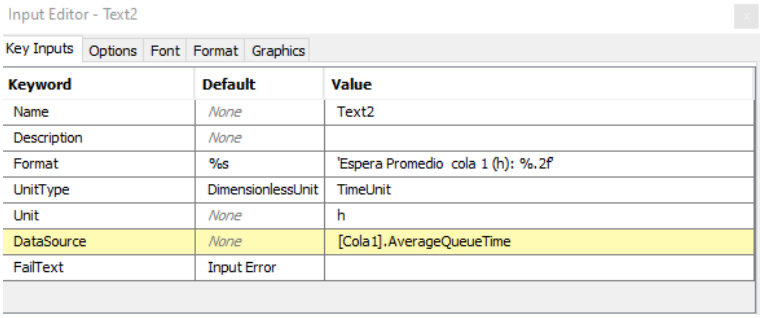


Figura 16. JaamSim Input Editor para espera promedio cola 1

Tabla 7. Configuración general modelo M/M/1/∞

Object	Keyword	Value
Entrada	KeyInputs/NextComponent	Transp1
Entrada	KeyInputs/InterArrivalTime	Exp1
Entrada	KeyInputs/PrototypeEntity	Pieza
Máquina1	KeyInputs/NextComponent	Transp2
Máquina1	KeyInputs/WaitQueue	Cola1
Máquina1	KeyInputs/ServiceTime	Exp2
Transp1	KeyInputs/NextComponent	Máquina1
Transp1	KeyInputs/TravelTime	500 s
Transp2	KeyInputs/NextComponent	Salida
Transp2	KeyInputs/TravelTime	500 s
Exp1	KeyInputs/UnitType	TimeUnit
Exp1	KeyInputs/RandomSeed	1
Exp1	KeyInputs/Mean	100 s
Exp2	KeyInputs/UnitType	TimeUnit
Exp2	KeyInputs/RandomSeed	2
Exp2	KeyInputs/Mean	100 s

Nota: RandomSeed, es la ley de distribución estadística actúa de forma independiente, por lo tanto, cada valor generado (tiempo de procesado, tiempo de llegada, velocidad de conveyor, etc.) podrá tomar valores distintos. De no asignar distintas RandomSeed a cada distribución estadística designada a cada entidad (maq, input, conveyor) los valores proporcionados serán iguales y dependiente entre sí. Provocando que los resultados obtenidos puedan no coincidan con los concretos mostrados en la monografía.

3.2.4. Análisis del modelo M/M/1/∞

Esta configuración en JaamSim permite modelar un sistema de fabricación estocástico sencillo. Los pasos descritos garantizan que los estudiantes comprendan cómo configurar y analizar un modelo de simulación. El enfoque en la variabilidad y el flujo de trabajo ofrece una base sólida para explorar sistemas más complejos en contextos industriales reales.

El modelo de una máquina aleatoria en JaamSim es una herramienta poderosa para analizar sistemas estocásticos y comprender la influencia de la incertidumbre en el desempeño. A través de este ejemplo, los estudiantes pueden visualizar cómo los eventos aleatorios impactan el flujo de trabajo y cómo se pueden utilizar técnicas de simulación para evaluar y optimizar sistemas reales.

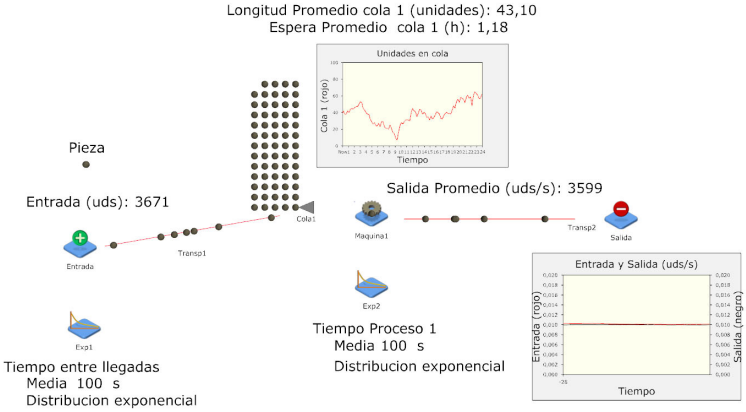


Figura 17. Resultados modelo M/M/1/∞ en JaamSim

3.3. M/M/2/∞ entorno fabricación con dos máquinas y una cola por máquina con tiempos de suministro y servicio aleatorios

Este modelo está diseñado para simular un sistema de fabricación donde los productos (o piezas) pasan por dos líneas de procesamiento paralelas, cada una con su propia cola y máquina. Se analizarán los componentes principales del modelo, las estadísticas generadas y su interpretación (Fig. 18).

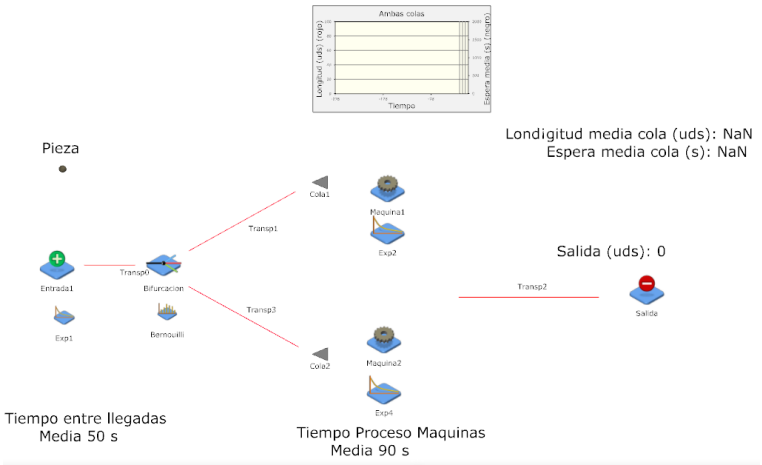


Figura 18. Modelo M/M/2/∞ con dos máquinas y una cola por máquina en JaamSim

El flujo de operación del sistema consiste en que las piezas ingresan al sistema siguiendo una distribución exponencial con una media de 50 segundos, tras lo cual se bifurcan ("Branch") aleatoriamente hacia la Cola1 o la Cola2. Las piezas esperan en sus respectivas colas hasta que la máquina asignada esté disponible, donde son procesadas en un tiempo promedio de 90 segundos. Finalmente, las piezas procesadas salen del sistema y se contabilizan en la salida. Este modelo en JaamSim es una representación eficiente y educativa de un sistema de fabricación simple con bifurcación aleatoria. La simulación permite entender los principios de operación y evaluar el desempeño del sistema a través de estadísticas clave.

3.3.1. Descripción general del modelo M/M/2/∞ con dos máquinas y una cola

1. **Entrada de piezas**
   - El modelo cuenta con un componente denominado Entrada1, que representa la llegada de piezas al sistema. Estas llegadas siguen un patrón aleatorio, con un tiempo medio entre cada pieza de 50 segundos. Este elemento inicial marca el comienzo del flujo de piezas en el sistema.
2. **Bifurcación (división del flujo)**
   - Tras ingresar al sistema, las piezas son dirigidas a una bifurcación modelada con un componente de tipo Bernoulli. Este elemento toma decisiones aleatorias para determinar cuál de las dos colas de procesamiento será utilizada por cada pieza. La bifurcación asegura que el flujo de piezas se distribuya de manera equitativa entre las dos líneas de procesamiento.
3. **Colas**
   - El sistema incluye dos colas independientes, denominadas Cola1 y Cola2, donde las piezas esperan su turno para ser procesadas. Estas colas son fundamentales

para el análisis del flujo, y se calculan métricas como la longitud media, que en este modelo alcanza un valor promedio de 15,60 unidades.

4. Máquinas de procesamiento

- Cada cola está conectada a una máquina específica: Maquina1 para Cola1 y Maquina2 para Cola2. Estas máquinas procesan las piezas de forma independiente, con un tiempo promedio de 90 segundos por pieza. Este paso es crucial para transformar las piezas y continuar con el flujo hacia la salida.

5. Salida de piezas

- Una vez que las piezas son procesadas, se envían al componente Salida, que acumula el total de unidades procesadas durante la simulación. En este modelo, se ha registrado un total de 71,784 unidades procesadas, lo que refleja la capacidad y eficiencia del sistema diseñado.

3.3.2. Asunciones modelo M/M/2/∞ con dos máquinas y una cola

1. La distribución de Bernoulli mide el número de llegadas por unidad de tiempo. La distribución del tiempo entre llegadas de un proceso de Poisson es la distribución exponencial negativa.
2. La distribución de Poisson mide los tiempos de servicio de las máquinas 1 y 2. La distribución del tiempo entre llegadas de un proceso de Poisson es la distribución exponencial negativa.
3. Tiempos de servicio de la máquina 1 son aleatorios.
4. Factor de utilización $\rho<1$.
5. El sistema está en un estado estable.
6. Los clientes son atendidos según el principio de primero en entrar, primero en salir (FIFO) en la cola disponible en la máquina 1.
7. La cola tiene una capacidad infinita.
8. Tiempo entre llegadas media = 50 segundos, con una distribución Bernoulli.
9. Tiempo de servicio de las máquinas presenta una media = 90 segundos, con una distribución exponencial.

3.3.3. Configuración en JaamSim del modelo M/M/2/∞ con dos máquinas y una cola

A continuación, se detallan los pasos esenciales para implementar el modelo *"Una cola por máquina (M/M/2/∞)"* en JaamSim. Este modelo incluye configuraciones específicas de cada componente para simular correctamente el sistema.

1. Definir los componentes básicos

Abra JaamSim y agregue los elementos necesarios siguiendo estos pasos:

1. **Entrada de piezas (“EntityGenerator”)**
   - Arrastre el componente *“EntityGenerator”* a la ventana principal de trabajo.
   - Configure sus parámetros en el panel de propiedades:
     - **Entity Type**: defina la pieza como entidad (puede ser un nombre como *Pieza*).
     - **Arrival Time Distribution**: seleccione *Exponential Distribution* y configure la **media en 50 segundos**.
2. **Bifurcación (“Branch”)**
   - Agregue el componente *Entity Generator* a la ventana principal de trabajo.
   - En el panel de propiedades:
     - **Distribution**: seleccione *Bernoulli Distribution*.
     - **Probability**: ajuste el parámetro de probabilidad para dividir el flujo en dos salidas. Por ejemplo, configure una probabilidad del 50% para cada línea.
3. **Colas (“Queue”)**
   - Agregue dos componentes *Queue* al modelo, uno para cada máquina:
     - Asigne nombres claros, como *Cola1* y *Cola2*.
     - No es necesario realizar configuraciones avanzadas, ya que las colas gestionan las entidades de manera automática.
4. **Máquinas (“Server”)**
   - Agregue dos componentes *Server* (uno para cada cola).
   - Configure los parámetros en el panel de propiedades:
     - **Service Time Distribution**: seleccione *Exponential Distribution* y configure una **media de 90 segundos**.
     - Vincule cada máquina a su respectiva cola en el panel de conexiones.
5. **Salida (“EntitySink”)**
   - Agregue un componente *Sink* al final de cada línea.
   - Este componente recopilará las piezas procesadas.
6. **Conectar los componentes**

Use el modo de edición para conectar visualmente los componentes en la ventana principal de trabajo:

- Conecte *Source* a *Entity Generator (Bernoulli)*.
- Desde el *Bernoulli*, dibuje conexiones hacia las dos *Queues*.
- Conecte cada *Queue* a su respectivo *Server*.
- Finalmente, conecte los *Servers* a la *Sink*.

7. Agregar reportes

Para analizar los resultados, añade "Graphics Objects" en forma de texto para proporcionar mayor información:

- Longitud promedio cola 1 (unidades):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 8.

Tabla 8. Configurar un campo para la visualización "Longitud promedio cola1"

Object	Keyword	Value
Text 1	Format	Longitud promedio cola 1 (unidades): %.2f
Text 1	Datasource	[Cola1].QueueLengthAverage

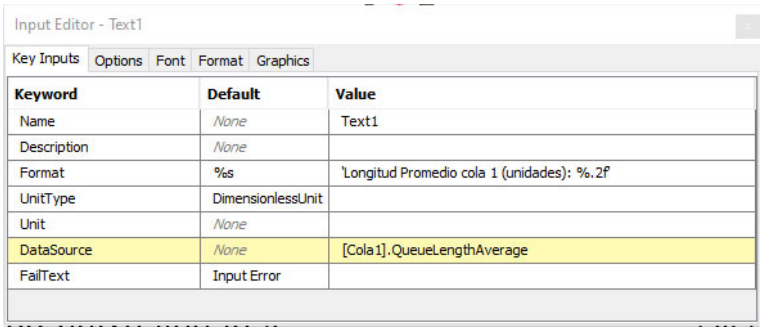


Figura 19. JaamSim Input Editor para Longitud promedio cola1

- Espera promedio cola 1 (h):

  Añade un objeto "Graph" y contemplan los siguientes campos en el apartado "Key Inputs", "X-Axis", "Y-Axis", "Secondary Y-Axis" y "Format" dentro del "Input Editor" según la Tabla 9.

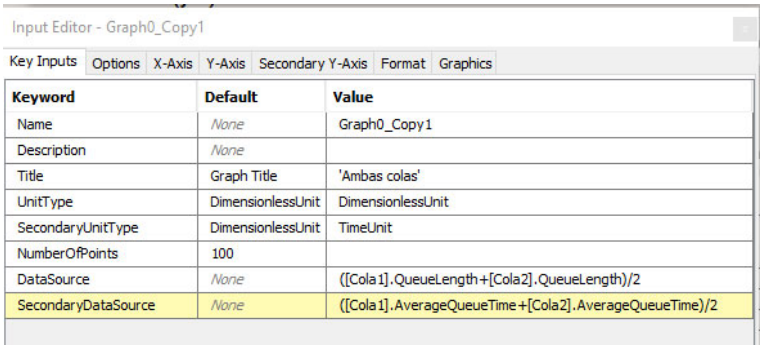
Tabla 9. Configurar un campo para la visualización "Espera promedio cola 1"

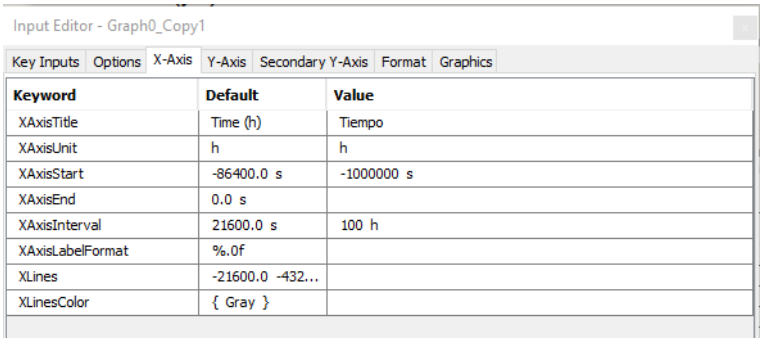
Object	Keyword	Value
Graph0	KeyInputs/Title	Ambas Colas
Graph0	KeyInputs/UnitType	DimensionlessUnit
Graph0	KeyInputs/SecondaryUnitType	TimeUnit
Graph0	KeyInputs/DataSource	([Cola1].QueueLength+[Cola2].QueueLength)/2
Graph0	KeyInputs/SecondaryData-Source	([Cola1].AverageQueueTime+[Cola2].AverageQueueTime)/2
Graph0	X-Axis/XAxisTitle	Tiempo
Graph0	X-Axis/XAxisUnit	h
Graph0	X-Axis/XAxisStart	-1000000 s

(continua)

(continuación)

Graph0	X-Axis/XAxisInterval	100 h
Graph0	Y-Axis/YAxisTitle	Longitud (uds) (rojo)
Graph0	Y-Axis/YAxisEnd	100
Graph0	Y-Axis/YAxisInterval	20
Graph0	Y-Axis/YAxisLabelFormat	%.0f
Graph0	Y-Axis/YLines	20 40 60 80
Graph0	Secondary Y-Axis/ SecondaryYAxisTitle	'Espera media (s) (negro)
Graph0	Secondary Y-Axis/ SecondaryYAxisUnit	s
Graph0	Secondary Y-Axis/ SecondaryYAxisEnd	2000 s
Graph0	Secondary Y-Axis/ SecondaryYAxisInterval	500 s
Graph0	Secondary Y-Axis/ SecondaryYAxisLabelFormat	%.0f





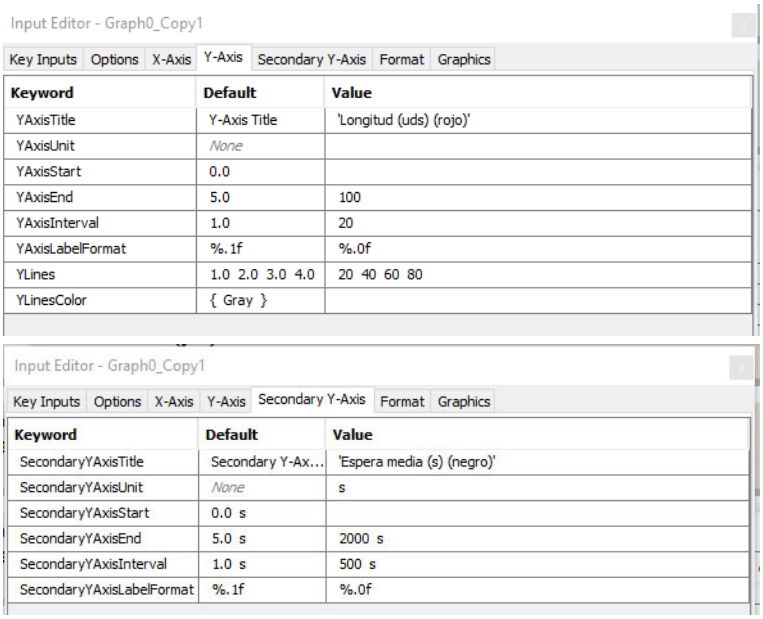


Figura 20. JaamSim Input Editor para gráfica longitud y espera media en cola

Tabla 10. Configuración general modelo M/M/2/∞ con dos máquinas y una cola

Object	Keyword	Value
Entrada	KeyInputs/NextComponent	Transp1
Entrada	KeyInputs/InterArrivalTime	Exp1
Entrada	KeyInputs/PrototypeEntity	Pieza
Bifurcación	KeyInputs/NextComponentList	Transp1 Transp3
Bifurcación	KeyInputs/Choice	Bernoulli
Máquina1	KeyInputs/NextComponent	Transp2
Máquina1	KeyInputs/WaitQueue	Cola1
Máquina1	KeyInputs/ServiceTime	Exp2
Transp1	KeyInputs/NextComponent	Máquina1
Transp1	KeyInputs/TravelTime	500 s
Transp2	KeyInputs/NextComponent	Salida
Transp2	KeyInputs/TravelTime	500 s
Transp3	KeyInputs/NextComponent	Máquina2
Transp3	KeyInputs/TravelTime	500 s
Exp1	KeyInputs/UnitType	TimeUnit
Exp1	KeyInputs/RandomSeed	1
Exp1	KeyInputs/Mean	50 s

(continua)

(continuación)

Exp2	KeyInputs/UnitType	TimeUnit
Exp2	KeyInputs/RandomSeed	2
Exp2	KeyInputs/Mean	90 s
Exp4	KeyInputs/UnitType	TimeUnit
Exp4	KeyInputs/RandomSeed	3
Exp4	KeyInputs/Mean	90 s
Bernouilli	KeyInputs/UnitType	DimensionlessUnit
Bernouilli	KeyInputs/RandomSeed	4
Bernouilli	KeyInputs/ValueList	1 2
Bernouilli	KeyInputs/ProbabilityList	0.5 0.5

Nota: el resultado obtenido en la solución del problema por parte del alumno no tiene por qué coincidir exactamente con el resultado que figura en la tabla, ya que el problema es aleatorio. En su caso, si se aumentara el tiempo durante el que se simula o el número de simulaciones realizadas su resultado en valor promedio y el proporcionado debería ser estadísticamente equivalentes, aunque no fuesen idénticos.

3.3.4. Análisis del modelo M/M/2/∞ con dos máquinas y una cola

El modelo representa un sistema básico que permite analizar aspectos clave como el balance en la distribución del trabajo entre diferentes procesos, donde una asignación aleatoria ayuda a compartir la carga de manera equitativa; la detección de puntos críticos en el flujo, reflejados en tiempos promedio de espera y acumulaciones, que indican posibles ineficiencias que podrían abordarse optimizando los tiempos de operación; y el efecto de la variabilidad, ya que la naturaleza impredecible de las entradas y decisiones genera fluctuaciones en el sistema, evidentes en el comportamiento de las filas o acumulaciones.

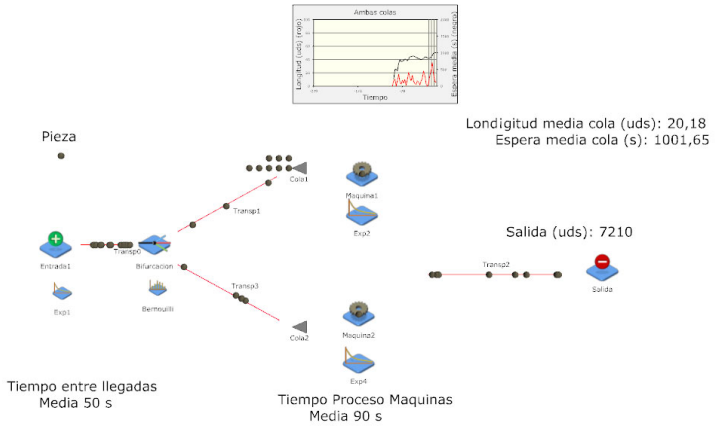


Figura 21. Resultados modelo M/M/2/∞ con dos máquinas y una cola por máquina en JaamSim

3.4. M/M/2/∞ entorno fabricación con dos máquinas y una cola única con tiempos de suministro y servicio variables o exponenciales

El sistema opera siguiendo un flujo genérico que puede aplicarse a múltiples entornos productivos. Primero, las entidades (o elementos) ingresan al sistema desde un punto de entrada, donde son generadas de acuerdo con una distribución específica que define el intervalo de tiempo entre llegadas. A continuación, las entidades se trasladan a través de un mecanismo de transporte hasta una cola, donde se acumulan a la espera de ser procesadas.

Una vez que los recursos disponibles (como máquinas o estaciones de trabajo) están libres, las entidades son asignadas para su procesamiento, el cual se lleva a cabo en un tiempo determinado por las características del sistema o el recurso. Finalmente, las entidades procesadas se trasladan al punto de salida, donde se registra su finalización o son transferidas a etapas posteriores del sistema. Este flujo permite evaluar la eficiencia, la capacidad y los cuellos de botella en el proceso, ayudando en la toma de decisiones para optimizar el rendimiento general. Se analizarán los componentes principales del modelo, las estadísticas generadas y su interpretación Fig. 22.

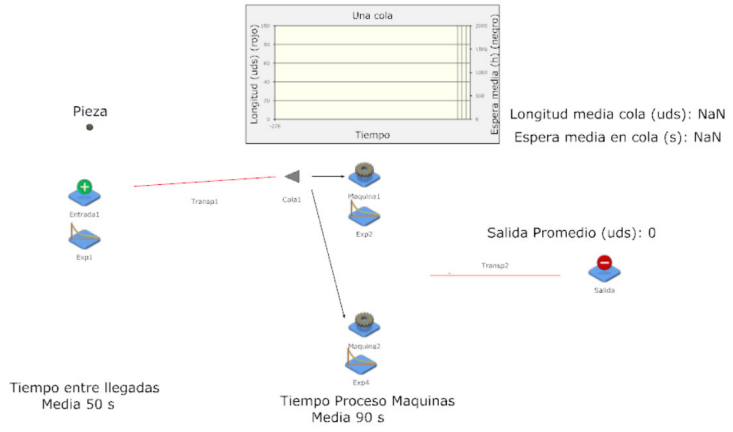


Figura 22. Modelo M/M/2/∞ una cola única

3.4.1. Descripción general del modelo M/M/2/∞ con dos máquinas y una cola única

1. Generación de entidades

El sistema inicia con un proceso de generación de entidades en un punto de entrada ("EntityGenerator"). Estas entidades llegan al sistema siguiendo una distribución aleatoria definida por un tiempo medio entre llegadas.

2. **Transporte de entidades**

   Una vez generadas, las entidades son trasladadas a través de un mecanismo de transporte ("EntityConveyor") hacia una cola ("Queue"). En esta cola, las entidades se acumulan mientras esperan ser atendidas por los recursos del sistema.

3. **Procesamiento de entidades**

   Cuando los recursos del sistema, como máquinas o estaciones de trabajo ("Server"), están disponibles, las entidades son asignadas para su procesamiento. Este proceso tiene una duración que puede seguir una distribución aleatoria definida, con una distribución exponencial específica para cada entidad, lo que refleja las variaciones inherentes al sistema.

4. **Salida del sistema**

   Tras completar el procesamiento, las entidades se trasladan al punto de salida, donde se registran como finalizadas.

3.4.2. Asunciones modelo M/M/2/∞ con dos máquinas y una cola única

1. La distribución de Poisson mide el número de llegadas por unidad de tiempo y los tiempos de servicio de las máquinas 1 y 2. La distribución del tiempo entre llegadas de un proceso de Poisson es la distribución exponencial negativa.
2. Tiempos de servicio de la máquina 1 son aleatorios.
3. Factor de utilización $\rho<1$.
4. El sistema está en un estado estable.
5. Los clientes son atendidos según el principio de primero en entrar, primero en salir (FIFO) en la cola disponible en la máquina 1.
6. La cola tiene una capacidad infinita.
7. Tiempo entre llegadas media = 50 segundos, con una distribución Bernoulli.
8. Tiempo de servicio de las máquinas presenta una media = 90 segundos, con una distribución exponencial.

3.4.3. Configuración en JaamSim del modelo M/M/2/∞ con dos máquinas y una cola única

Este proceso de configuración en JaamSim garantiza que todos los componentes del modelo estén correctamente parametrizados y conectados. Siguiendo estos pasos, es posible implementar un sistema funcional que represente de manera realista un entorno productivo o logístico, facilitando el análisis y la optimización del rendimiento del sistema.

1. **Definir los componentes básicos**

   Abra JaamSim y agregue los elementos necesarios a la ventana principal de trabajo siguiendo estos pasos:

1. **Entrada de piezas ("EntityGenerator")**
   - Arrastre el componente *"EntityGenerator"* a la ventana principal de trabajo.
   - Configure sus parámetros en el panel de propiedades:
     - **Entity Type**: defina la pieza como entidad (puede ser un nombre como *Pieza*).
     - **Arrival Time Distribution**: seleccione *Exponential Distribution* y configure la **media en 50 segundos**.
2. **Cola ("Queue")**
   - Agregue un componente "*Queue"* al modelo para las dos máquinas:
     - No es necesario realizar configuraciones avanzadas, ya que las colas gestionan las entidades de manera automática.
3. **Máquinas ("Server")**
   - Agregue dos componentes *Server* y asigne la misma cola "Queue" a las dos máquinas.
   - Configure los parámetros en el panel de propiedades:
     - Service Time Distribution: seleccione *Exponential Distribution* y configure una media de 90 segundos para cada una de las máquinas.
4. **Salida ("EntitySink")**
   - Agregue un componente *Sink* al final de cada línea.
   - Este componente recopilará las piezas procesadas.
5. **Conectar los componentes**
   - Use el modo de edición para conectar visualmente los componentes:
     - Conecte cada una de las distribuciones estadísticas con la correspondiente entidad.
     - Conecte cada entidad con el siguiente elemento.
     - Conecte cada Queue a las dos máquinas o "Server".
     - Finalmente, conecte las máquinas con la salida "EntitySink" mediante una cinta trasportadora o "EntityConveyor".
6. **Agregar**

Tabla 11. Configuración general modelo M/M/2/∞ con dos máquinas y una cola única

Object	Keyword	Value
Entrada	KeyInputs/NextComponent	Transp0
Entrada	KeyInputs/InterArrivalTime	Exp1
Entrada	KeyInputs/PrototypeEntity	Pieza
Máquina1	KeyInputs/NextComponent	Transp2

(continua)

(continuación)

Máquina1	KeyInputs/WaitQueue	Cola1
Máquina1	KeyInputs/ServiceTime	Exp2
Máquina2	KeyInputs/NextComponent	Transp1
Máquina2	KeyInputs/WaitQueue	Cola1
Máquina2	KeyInputs/ServiceTime	Exp4
Transp1	KeyInputs/NextComponent	Máquina1
Transp1	KeyInputs/TravelTime	500 s
Transp2	KeyInputs/NextComponent	Salida
Transp2	KeyInputs/TravelTime	500 s
Exp1	KeyInputs/UnitType	TimeUnit
Exp1	KeyInputs/RandomSeed	1
Exp1	KeyInputs/Mean	50 s
Exp2	KeyInputs/UnitType	TimeUnit
Exp2	KeyInputs/RandomSeed	3
Exp2	KeyInputs/Mean	90 s
Exp4	KeyInputs/UnitType	TimeUnit
Exp4	KeyInputs/RandomSeed	4
Exp4	KeyInputs/Mean	90 s

3.4.4. Análisis del modelo M/M/2/∞ con dos máquinas y una cola única

El modelo con múltiples recursos y una única cola representa un sistema productivo común, donde las entidades generadas se acumulan en una cola y son atendidas por los recursos disponibles según su disponibilidad. Aunque funcional, este tipo de sistema tiende a presentar cuellos de botella en la cola cuando la tasa de llegada de entidades excede la capacidad de procesamiento de los recursos, lo que se refleja en tiempos de espera elevados y acumulación en la cola.

A pesar de alcanzar un rendimiento promedio aceptable, estos indicadores evidencian que los recursos están trabajando a plena capacidad y que la asignación aleatoria de entidades puede generar desbalances en su utilización. Para optimizar este tipo de sistemas, se pueden considerar estrategias como aumentar la capacidad de los recursos, reducir los tiempos de procesamiento, dividir el sistema en múltiples colas o implementar reglas de asignación más eficientes, mejorando así el flujo de trabajo y minimizando los tiempos de espera en la cola.

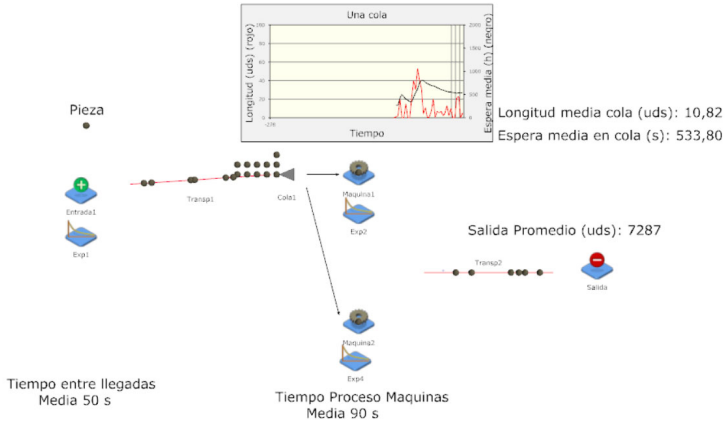


Figura 23. Resultados modelo M/M/2/∞ con dos máquinas y una cola única en JaamSim

3.5. M/M/1/∞ + M/M/1/∞ entorno fabricación con dos máquinas en serie con tiempos de suministro y servicio variables o exponenciales

Este modelo está diseñado para simular un sistema de fabricación donde los productos (o piezas) siguen un modelo de línea de flujo en serie. En este documento se explicará el modelo titulado **"**Dos máquinas en serie **(**M/M/1/∞ + M/M/1/∞)**"**, que simula un sistema de producción básico con dos máquinas operando secuencialmente. El flujo de operación del sistema inicia con la llegada de elementos al punto de entrada, donde se generan según un intervalo de tiempo definido. Los elementos se almacenan temporalmente en una cola mientras esperan ser procesados por la primera unidad de trabajo. Una vez procesados, los elementos son transportados a una segunda cola, donde aguardan su turno para ser atendidos por una segunda unidad de trabajo. Tras completar ambos procesos, los elementos finalizan su recorrido en el sistema al llegar al punto de salida, donde se recopilan datos para evaluar el desempeño del proceso en términos de capacidad, tiempos de espera y balanceo de recursos. Se analizarán los componentes principales del modelo, las estadísticas generadas y su interpretación Fig. 24.

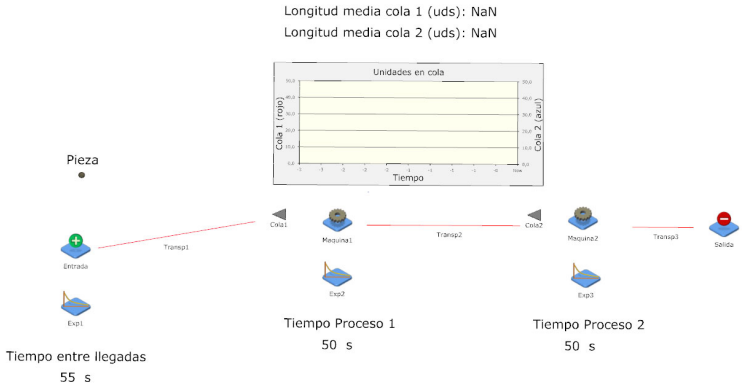


Figura 24. M/M/1/∞ + M/M/1/∞ Entorno fabricación con dos máquinas en serie en JaamSim

3.5.1. Descripción general del modelo M/M/1/∞ + M/M/1/∞ en serie

El modelo titulado **"Dos máquinas en serie"** representa un sistema de producción básico que consta de una secuencia de procesos donde se generan los elementos (o piezas), almacenados temporalmente en colas y procesados por dos máquinas de forma secuencial. A continuación, se detalla cada componente del modelo y su función dentro del flujo operativo:

1. **Entrada de piezas**
   - La entrada es el punto donde se generan los elementos que ingresan al sistema, utilizando una distribución exponencial con un tiempo promedio de 55 segundos entre llegadas. Esto simula un proceso estocástico, generando piezas de forma aleatoria que se procesarán a lo largo del sistema. Este componente marca el inicio del flujo operativo.
2. **Cola 1**
   - Esta es un área de almacenamiento temporal donde las piezas esperan su turno para ser procesadas por la Máquina 1. Aquí se mide la longitud de la cola en términos de unidades acumuladas, que sirve como indicador del rendimiento y balance del sistema. Las piezas permanecen en esta cola hasta que la Máquina 1 esté disponible para procesarlas.
3. **Máquina 1**
   - La primera unidad de procesamiento, que opera con un tiempo promedio de 50 segundos por pieza. Cada pieza es procesada de forma individual y, una vez completada, es transferida al siguiente paso del sistema. Esta máquina simula recursos como estaciones de trabajo o unidades de inspección inicial.

4. **Cola2**
    - Similar a Cola 1, esta área sirve como un buffer donde las piezas esperan ser procesadas por la Máquina 2. La longitud de esta cola también se mide para evaluar el rendimiento del sistema. Este almacenamiento temporal es crítico para mantener la continuidad en caso de que haya un desfase entre los procesos.
5. **Máquina 2**
    - Esta es la segunda unidad de procesamiento, con un tiempo promedio de 50 segundos por pieza. Al igual que Máquina 1, procesa cada elemento individualmente y completa la segunda etapa del trabajo, como operaciones de acabado o inspección final, antes de enviar los productos a la salida.
6. **Salida de piezas**
    - Es el punto final del flujo operativo, donde las piezas procesadas completan su recorrido. Aquí se registran estadísticas clave como el número total de piezas procesadas y el tiempo total en el sistema. Este componente simula la salida de los productos terminados hacia el almacenamiento o despacho.
7. **Transporte**
    - Representa el traslado de las piezas desde la salida de una entidad hasta la entrada de la siguiente entidad. Este componente asegura la continuidad del flujo y simula el movimiento físico de los elementos entre estaciones.

3.5.2. Asunciones modelo M/M/1/∞ + M/M/1/∞ en serie

1. La distribución de Poisson mide el número de llegadas por unidad de tiempo y los tiempos de servicio de las máquinas 1 y 2. La distribución del tiempo entre llegadas de un proceso de Poisson es la distribución exponencial negativa.
2. Tiempos de servicio de la máquina 1 y 2 son aleatorios.
3. Factor de utilización $\rho<1$.
4. El sistema está en un estado estable.
5. Los clientes son atendidos según el principio de primero en entrar, primero en salir (FIFO) en la cola disponible en la máquina 1.
6. La cola tiene una capacidad infinita.
7. Tiempo entre llegadas media = 55 segundos, con una distribución exponencial.
8. Tiempo de servicio de las máquinas presenta una media = 50 segundos, con una distribución exponencial.

3.5.3. Configuración en JaamSim del modelo M/M/1/∞ + M/M/1/∞ en serie

A continuación, se detallan los pasos esenciales para implementar el modelo *"Dos máquinas en serie* M/M/1/∞ + M/M/1/∞" en JaamSim. Este modelo incluye configuraciones específicas de cada componente para simular correctamente el sistema. Abra JaamSim y agregue los elementos necesarios a la ventana principal de trabajo siguiendo estos pasos:

1. Entrada de piezas ("EntityGenerator")

- Arrastre el componente *"EntityGenerator"* a la ventana principal de trabajo.
- Configure sus parámetros en el panel de propiedades:
  - **Entity Type**: defina la pieza como entidad (puede ser un nombre como *Pieza*).
  - **Arrival Time Distribution**: seleccione *Exponential Distribution* y configure la **media en 55 segundos**.

2. Colas ("Queue")

- Agregue dos componentes *Queue* al modelo, uno para cada máquina:
  - Asigne nombres claros, como *Cola1* y *Cola2*.

3. Máquinas ("Server")

- Agregue dos componentes *Server* (uno para cada cola).
- Configure los parámetros en el panel de propiedades:
  - **Service Time Distribution**: seleccione *Exponential Distribution* y configure una **media de 50 segundos**.
  - Vincule cada máquina a su respectiva cola en el panel de conexiones.

4. Salida ("EntitySink")

- Agregue un componente *Sink* al final de cada línea.
- Este componente recopilará las piezas procesadas.

5. Conectar los componentes

Use el modo de edición para conectar visualmente los componentes:

- Conecte cada una de las distribuciones estadísticas con la correspondiente entidad.
- Conecte cada entidad con el siguiente elemento.
- Conecte cada Queue a las dos máquinas o "Server".
- Finalmente, conecte las máquinas con la salida "EntitySink" mediante una cinta trasportadora o "EntityConveyor".

6. Agregar reportes

Para analizar los resultados, añade "Graphics Objects" en forma de texto para proporcionar mayor información:

- Longitud Promedio cola 1 (unidades):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 12.

Tabla 12. Configurar un campo para la visualización "Longitud promedio cola1"

Object	Keyword	Value
Text 1	Format	Longitud Promedio cola 1 (unidades): %.2f
Text 1	Datasource	[Cola1].QueueLengthAverage

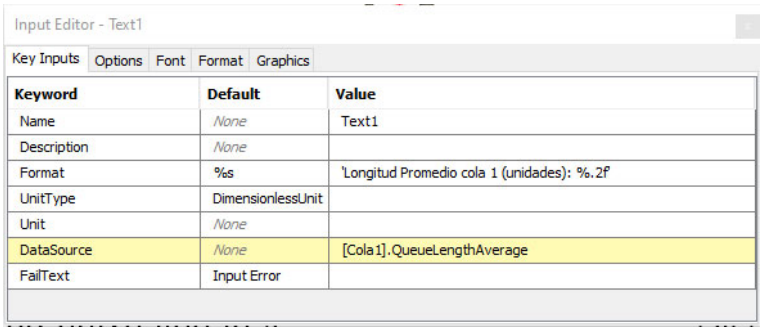


Figura 25. JaamSim Input Editor para Longitud promedio cola1

- Espera promedio cola 1 (h):

Añade un objeto "Graph" y contemplan los siguientes campos en el apartado "Key Inputs", "X-Axis", "Y-Axis", "Secondary Y-Axis" y "Format" dentro del "Input Editor" según la Tabla 13.

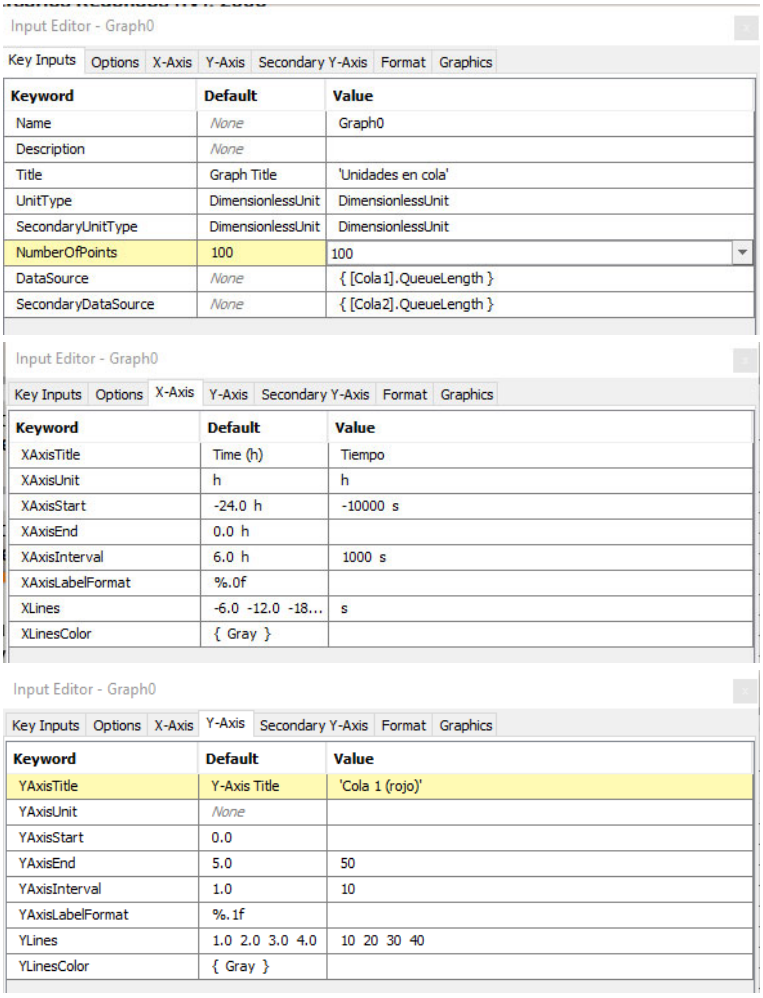
Tabla 13. Configurar un campo para la visualización "Espera promedio cola 1"

Object	Keyword	Value
Graph0	KeyInputs/Title	Unidades en cola
Graph0	KeyInputs/UnitType	DimensionlessUnit
Graph0	KeyInputs/SecondaryUnitType	DimensionlessUnit
Graph0	KeyInputs/NumberOfPoints	100
Graph0	KeyInputs/DataSource	[Cola1].QueueLength
Graph0	KeyInputs/SecondaryDataSource	[Cola2].QueueLength
Graph0	X-Axis/XAxisTitle	Tiempo
Graph0	X-Axis/XAxisUnit	h
Graph0	X-Axis/XAxisStart	-1000000 s
Graph0	X-Axis/XAxisInterval	1000 h
Graph0	X-Axis/XLines	s
Graph0	Y-Axis/YAxisTitle	Cola 1 (rojo)
Graph0	Y-Axis/YAxisEnd	50
Graph0	Y-Axis/YAxisInterval	10
Graph0	Y-Axis/YAxisLabelFormat	%.1f

(continua)

(continuación)

Graph0	Y-Axis/YLines	10 20 30 40
Graph0	Secondary Y-Axis/ SecondaryYAxisTitle	Cola 2 (azul)
Graph0	Secondary Y-Axis/ SecondaryYAxisEnd	50
Graph0	Secondary Y-Axis/ SecondaryYAxisInterval	10
Graph0	Secondary Y-Axis/ SecondaryYAxisLabelFormat	%.1f



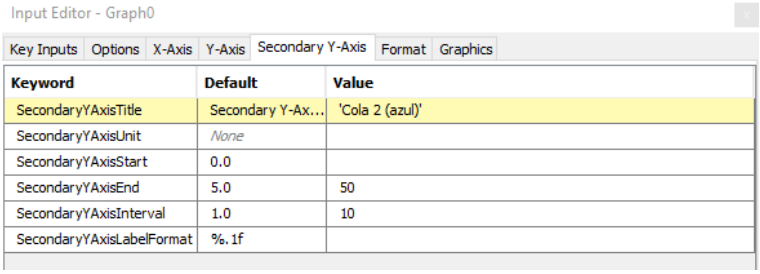


Figura 26. JaamSim Input Editor para gráfica unidades en cola 1-2

Tabla 14. Configuración general modelo M/M/1/∞ + M/M/1/∞ en serie

Object	Keyword	Value
Entrada	KeyInputs/NextComponent	Transp1
Entrada	KeyInputs/InterArrivalTime	Exp1
Entrada	KeyInputs/PrototypeEntity	Pieza
Máquina1	KeyInputs/NextComponent	Transp2
Máquina1	KeyInputs/WaitQueue	Cola1
Máquina1	KeyInputs/ServiceTime	Exp2
Máquina2	KeyInputs/NextComponent	Transp2
Máquina2	KeyInputs/WaitQueue	Cola2
Máquina2	KeyInputs/ServiceTime	Exp3
Transp1	KeyInputs/NextComponent	Máquina1
Transp1	KeyInputs/TravelTime	500 s
Transp2	KeyInputs/NextComponent	Máquina2
Transp2	KeyInputs/TravelTime	500 s
Transp3	KeyInputs/NextComponent	Salida
Transp3	KeyInputs/TravelTime	500 s
Exp1	KeyInputs/UnitType	TimeUnit
Exp1	KeyInputs/RandomSeed	1
Exp1	KeyInputs/Mean	55 s
Exp2	KeyInputs/UnitType	TimeUnit
Exp2	KeyInputs/RandomSeed	2
Exp2	KeyInputs/Mean	50 s
Exp3	KeyInputs/UnitType	TimeUnit
Exp3	KeyInputs/RandomSeed	3
Exp3	KeyInputs/Mean	50 s

3.5.4. Análisis del modelo M/M/1/∞ + M/M/1/∞ en serie

El modelo muestra un flujo relativamente balanceado entre las dos máquinas, con acumulaciones moderadas en las colas que garantizan la continuidad del proceso sin generar interrupciones. Para optimizar el sistema, se podrían realizar ajustes en el tiempo entre llegadas (por ejemplo, incrementándolo ligeramente) para reducir las longitudes promedio de las colas y mejorar la eficiencia general. Este análisis proporciona una base sólida para comprender y mejorar el sistema simulado.

La longitud promedio registrada en Cola 1 es de 9,20 unidades, lo que indica que, en promedio, hay 9 piezas esperando antes de ser procesadas por la Máquina 1, para las primeras 100 horas de simulación. Este valor sugiere una ligera acumulación en esta etapa, posiblemente porque las llegadas de piezas generan una carga mayor que el ritmo de procesamiento. Por otro lado, en Cola 2 la longitud promedio es de 8,22 unidades, mostrando que también se produce acumulación en esta etapa, aunque ligeramente menor que en la primera cola. Esto refleja que el sistema está relativamente balanceado, ya que ambas máquinas procesan las piezas a un ritmo similar, manteniendo la continuidad del flujo con acumulaciones moderadas.

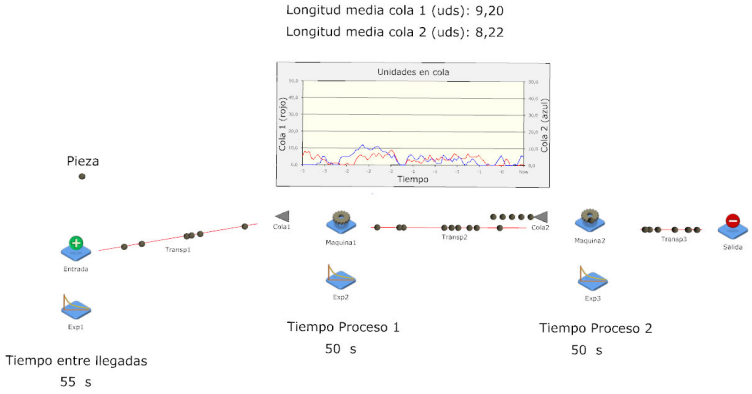


Figura 27. Resultados modelo M/M/1/∞ + M/M/1/∞ en serie en JaamSim

Ambas máquinas tienen un tiempo promedio de procesamiento de 50 segundos, lo que contribuye a un buen balance en el sistema. La similitud en las longitudes promedio de las colas sugiere que no hay un cuello de botella significativo, ya que ninguna de las máquinas está siendo excesivamente cargada en comparación con la otra. Sin embargo, la acumulación moderada en ambas colas indica que el tiempo entre llegadas (55 segundos) está ligeramente desalineado con los tiempos de procesamiento. Esto genera una acumulación sostenida de piezas en ambas colas, aunque no de manera crítica.

El sistema está operando de manera eficiente, procesando un flujo constante de piezas sin interrupciones significativas. La acumulación moderada en las colas puede ser interpretada como una reserva que asegura la continuidad del proceso, aunque también

podría optimizarse ajustando el tiempo entre llegadas para reducir las colas sin generar períodos de inactividad en las máquinas.

3.6. M/M/1/b entorno fabricación con una máquina con cola de longitud finita y con tiempos de suministro y servicio variables o exponenciales (cola con bloqueo)

Este modelo está diseñado para simular un sistema de fabricación donde los productos (o piezas) siguen un modelo de línea de flujo en serie. El modelo M/M/1/b se caracteriza por representar un sistema de colas con una sola máquina (servidor), una única cola de espera con capacidad finita K, y tiempos tanto de llegada como de servicio que siguen una distribución exponencial (lo que implica que los eventos ocurren de manera completamente aleatoria con una tasa promedio constante). Este modelo asume que las llegadas son un proceso de Poisson y que las piezas son procesadas en orden de llegada (FIFO). Si la cola alcanza su capacidad máxima, las nuevas llegadas son bloqueadas y no ingresan al sistema, lo que lo convierte en un modelo adecuado para analizar sistemas con limitaciones físicas o de almacenamiento, como líneas de producción o sistemas logísticos con capacidad restringida.

El comportamiento dinámico del sistema consiste en que las piezas llegan al sistema de forma aleatoria; si la máquina está libre, una pieza pasa directamente al procesamiento, mientras que, si está ocupada, la pieza se almacena en una cola con capacidad limitada. En caso de que la cola esté llena, la pieza es bloqueada y no ingresa al sistema. Una vez que una pieza entra en la máquina, se procesa en un tiempo aleatorio definido por una distribución exponencial y, tras finalizar, abandona el sistema. El modelo permite monitorear en tiempo real aspectos clave como la longitud de la cola, la probabilidad de bloqueo de piezas y el tiempo promedio de permanencia de una pieza en el sistema (WIP), proporcionando un análisis detallado del comportamiento del sistema. Se analizarán los componentes principales del modelo, las estadísticas generadas y su interpretación Fig. 28.

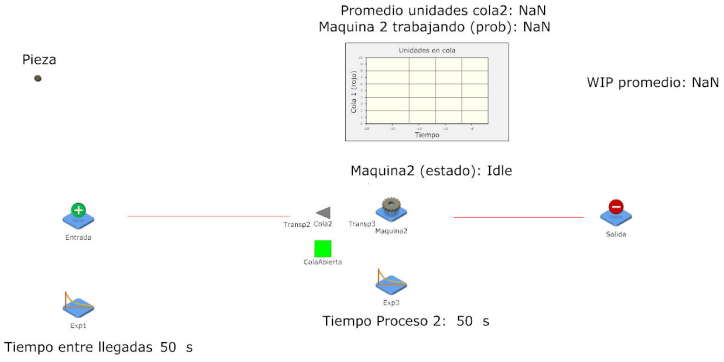


Figura 28. M/M/1/K Entorno fabricación con una máquina con cola de longitud finita en JaamSim

3.6.1. Descripción general del modelo M/M/1/K con cola de longitud finita

El modelo titulado "M/M/1/K" representa un sistema de producción o servicio donde las piezas (o entidades) son generadas de forma aleatoria, almacenadas temporalmente en una cola de capacidad limitada y procesadas por una única máquina. Este modelo es ideal para analizar sistemas con restricciones de capacidad y variabilidad en los tiempos de llegada y servicio. A continuación, se detalla cada componente del modelo y su función dentro del flujo operativo:

1. **Entrada de piezas**

   La entrada es el punto donde se generan las piezas que ingresan al sistema, utilizando una distribución exponencial con un tiempo promedio entre llegadas. Este componente simula un proceso estocástico en el cual las piezas llegan de manera aleatoria, marcando el inicio del flujo operativo.

2. **Cola**

   La cola es un área de almacenamiento temporal con capacidad limitada (K) donde las piezas esperan su turno para ser procesadas por la máquina. Si la cola alcanza su límite, las nuevas piezas que llegan son rechazadas, lo que refleja la capacidad restringida del sistema. La longitud de la cola es una métrica clave para evaluar el rendimiento del modelo y su nivel de utilización.

3. **Máquina**

   La máquina es la única unidad de procesamiento en el sistema y opera con un tiempo aleatorio para cada pieza, definido por una distribución exponencial. Este componente representa recursos como estaciones de trabajo, equipos de inspección o procesos críticos dentro de una línea de producción. Una vez que el procesamiento de una pieza es completado, esta abandona el sistema.

4. **Salida de piezas**

   La salida es el punto final del flujo operativo, donde las piezas procesadas completan su recorrido. Aquí se registran estadísticas clave como el número total de piezas procesadas, el tiempo promedio de permanencia en el sistema (WIP) y la probabilidad de bloqueo, que mide la frecuencia con la que las piezas son rechazadas debido a la cola llena.

3.6.2. Asunciones modelo M/M/1/K con cola de longitud finita

1. La distribución de Poisson mide el número de llegadas por unidad de tiempo y los tiempos de servicio de la máquina 1. La distribución del tiempo entre llegadas de un proceso de Poisson es la distribución exponencial negativa.
2. Tiempos de servicio de la máquina 1 son aleatorios.
3. Factor de utilización $\rho<1$.
4. El sistema está en un estado estable.
5. El sistema está lleno; la tasa de llegadas es $\lambda= 0$.

6. La cantidad de clientes en el sistema no llega a ∞.
7. La cola tiene una capacidad finita de 1 elemento.
8. Las piezas que llegan cuando la cola está llena son rechazados.
9. Tiempo entre llegadas media = 50 segundos, con una distribución exponencial.
10. Tiempo de servicio de las máquinas presenta una media = 50 segundos, con una distribución exponencial.

3.6.3. Configuración en JaamSim del modelo M/M/1/K con cola de longitud finita

A continuación, se detallan los pasos esenciales para implementar el modelo "M/M/1/b con cola de longitud finita" en JaamSim. Este modelo requiere la configuración específica de cada componente para simular correctamente el sistema. Abra JaamSim y siga los pasos detallados a continuación:

1 Entrada de piezas ("EntityGenerator")

- Arrastre el componente **"EntityGenerator"** a la ventana principal de trabajo.
- Configure sus parámetros en el panel de propiedades:
  - **Entity Type**: defina las piezas como entidad (por ejemplo, "Pieza").
  - **Arrival Time Distribution**: seleccione **Exponential Distribution** y configure el tiempo promedio entre llegadas (media) en 50 segundos.
- Vincular el EntityGenerator al ExpressionThreshold:
  - En el panel de propiedades del **EntityGenerator**, localice la sección **OperatingThresholdList**.
  - Vincule este parámetro al componente ExpressionThreshold, que será configurado para limitar las llegadas en función de la cola.

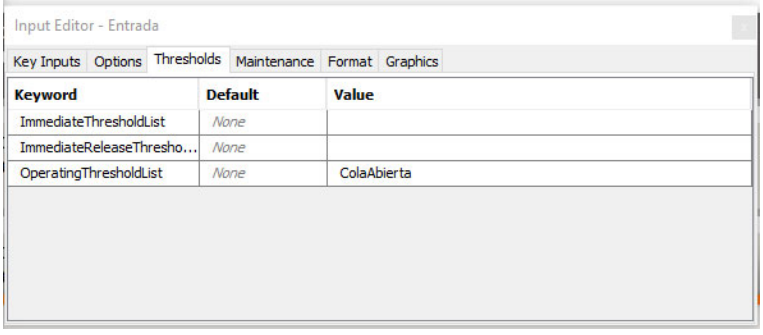


Figura 29. JaamSim Input Editor para la entidad Entrada del process flow

2 Cola ("Queue")

- Agregue un componente **Queue** al modelo para representar la cola de espera.
- Configure los parámetros en el panel de propiedades:
  - **Queue Capacity**: establezca el valor máximo de la cola en KKK, por ejemplo, 5 unidades, para reflejar la longitud finita de la cola.

3 Configuración del control de cola ("ColaAbierta")

El control de flujo hacia la cola se implementa utilizando un componente **ExpressionThreshold**, que evalúa una condición lógica para detener o permitir la entrada de piezas al sistema.

- **Agregar el componente ExpressionThreshold**
  - Arrastre el componente **"ExpressionThreshold"** a la ventana principal de trabajo.
- **Configurar la expresión lógica**
  - En el panel de propiedades del **ExpressionThreshold**, configure el parámetro **ThresholdExpression** con la siguiente expresión: [Cola2].QueueLength < 1
  - Esta expresión asegura que el flujo hacia la cola se detenga cuando su longitud alcance 1. Es decir, solo se permitirá una pieza en la cola en cualquier momento.

Tabla 15. Configurar un objeto "ExpressionThreshold" (ColaAbierta)

Object	Keyword	Value
ColaAbierta	KeyInputs/Name	ColaAbierta
ColaAbierta	KeyInputs/ OpenCondition	[Cola2].QueueLength<1
ColaAbierta	KeyInputs/InitialOpenValue	FALSE
ColaAbierta	KeyInputs/VerifyWatchList	FALSE

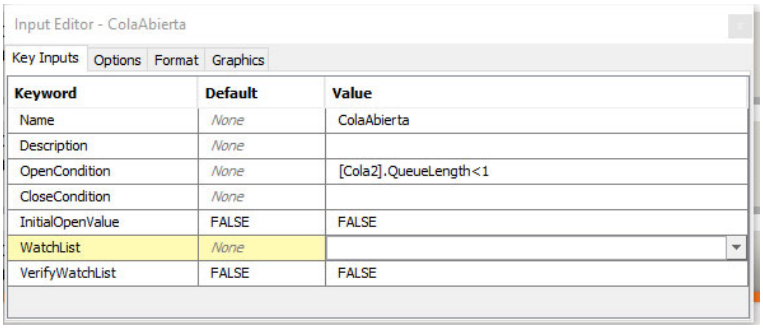


Figura 30. JaamSim Input Editor para ExpressionThreshold" (ColaAbierta)

4 Máquina ("Server")

- Arrastre un componente **Server** a la ventana principal de trabajo para representar la máquina.
- Configure los parámetros en el panel de propiedades:
  - **Service Time Distribution**: seleccione **Exponential Distribution** y configure el tiempo promedio de servicio (media) en 50 segundos.
  - **Input Queue**: vincule el servidor a la cola previamente configurada ("Cola2") en el panel de conexiones.
  - **Server Name**: Asigne un nombre claro, como "Maquina2".

5 Salida ("EntitySink")

- Agregue un componente **EntitySink** al modelo para representar el punto de salida de las piezas procesadas.
- Este componente no requiere configuraciones adicionales más allá de la conexión con la máquina.

6 Conexión de componentes

- Use el modo de edición para conectar visualmente los componentes:
  - Conecte el **EntityGenerator** con la **Queue**.
  - Conecte la **Queue** con el **Server**.
  - Finalmente, conecte el **Server** con el **EntitySink**.

7 Agregar reportes

Para analizar los resultados, añade "Graphics Objects" en forma de texto "Text" para proporcionar mayor información:

- Maquina 2 trabajando (probabilidad):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 16. Configurar un campo para la visualización "Maquina 2 trabajando"

Tabla 16. Configurar un campo para la visualización "Maquina 2 trabajando"

Object	Keyword	Value
Text 3	Format	Maquina 2 trabajando (prob): %.4f
Text 3	UnitType	DimensionlessUnit
Text 3	Datasource	([Maquina2].WorkingTime)/([Maquina2].SimTime)

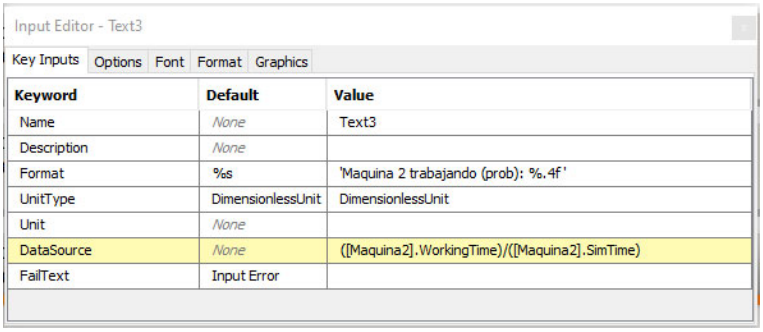


Figura 31. JaamSim Input Editor para Maquina 2 trabajando

- WIP promedio (unidades):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 17.

Tabla 17. Configurar un campo para la visualización "WIP promedio"

Object	Keyword	Value
Text 4	Format	WIP promedio: %.4f
Text 4	UnitType	DimensionlessUnit
Text 4	Datasource	([Maquina2].WorkingTime)/([Maquina2].SimTime)+[Cola2].QueueLengthAverage

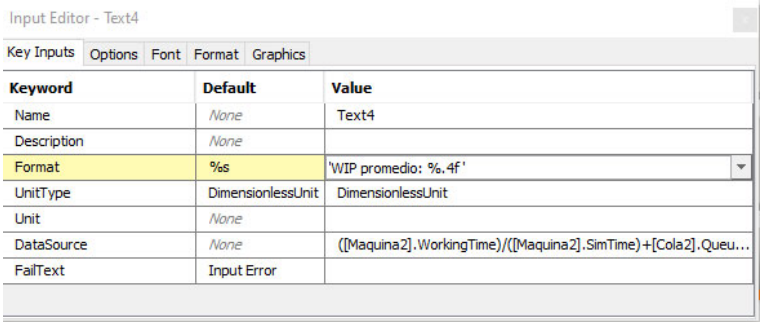


Figura 32. JaamSim Input Editor para WIP promedio

- Máquina 2 (estado):

  Añade un objeto "Texto" y contemplan los siguientes campos en el apartado "Key Inputs" dentro del "Input Editor" según la Tabla 18.

Tabla 18. Configurar un campo para la visualización "Máquina 2 (estado)"

Object	Keyword	Value
Text 5	Format	Maquina2 (estado): %s
Text 5	UnitType	DimensionlessUnit
Text 5	Datasource	[Maquina2].State

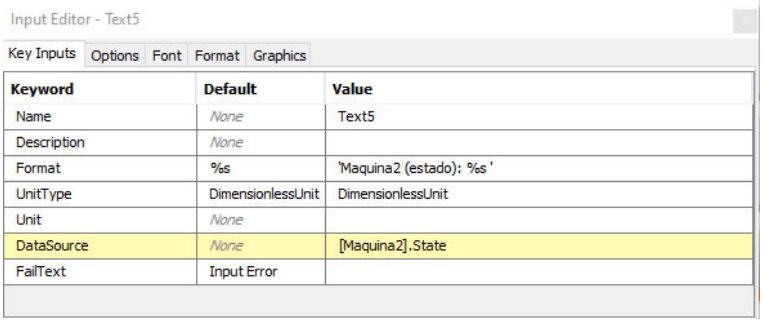


Figura 33. JaamSim Input Editor para Máquina 2 (estado)

8 Monitoreo del sistema

- Agregue gráficos y medidores al modelo para observar las métricas clave:
  - Para la longitud de la cola, utilice el componente **TimeSeriesPlot** vinculado a "Cola2".
  - Para la utilización de la máquina, agregue un componente **Display** que muestre el estado del servidor (Idle/Working).
  - Para estadísticas generales como el WIP o la probabilidad de bloqueo, use componentes adicionales de monitoreo disponibles en JaamSim.

Tabla 19. Configuración general modelo M/M/1/K con cola de longitud finita

Object	Keyword	Value
Entrada	KeyInputs/NextComponent	Transp1
Entrada	KeyInputs/InterArrivalTime	Exp1
Entrada	KeyInputs/PrototypeEntity	Pieza
Entrada	Thresholds	ColaAbierta
Máquina1	KeyInputs/NextComponent	Transp2
Máquina1	KeyInputs/WaitQueue	Cola1
Máquina1	KeyInputs/ServiceTime	Exp2
Transp1	KeyInputs/NextComponent	Máquina1
Transp1	KeyInputs/TravelTime	0 s
Transp2	KeyInputs/NextComponent	Máquina2

(continua)

(continuación)

Transp2	KeyInputs/TravelTime	500 s
Transp3	KeyInputs/NextComponent	Salida
Transp3	KeyInputs/TravelTime	500 s
Exp1	KeyInputs/UnitType	TimeUnit
Exp1	KeyInputs/RandomSeed	1
Exp1	KeyInputs/Mean	50 s
Exp2	KeyInputs/UnitType	TimeUnit
Exp2	KeyInputs/RandomSeed	2
Exp2	KeyInputs/Mean	50 s
ColaAbierta	KeyInputs/OpenCondition	[Cola2].QueueLength<1

3.6.4. Análisis del modelo M/M/1/K con cola de longitud finita

El modelo simulado presenta un sistema bien configurado, caracterizado por un flujo constante de entidades procesadas por el servidor y acumulaciones moderadas en la cola. Esto permite mantener la continuidad del proceso sin interrupciones significativas. Para mejorar el rendimiento del sistema, sería posible ajustar parámetros como el tiempo entre llegadas, con el objetivo de reducir las acumulaciones promedio en la cola y optimizar el nivel de trabajo en progreso (WIP).

Los resultados muestran que la longitud promedio de la cola es baja, lo que indica que, en general, las entidades no permanecen mucho tiempo esperando antes de ser procesadas. El servidor tiene una alta probabilidad de estar ocupado, lo que refleja un uso eficiente del recurso sin incurrir en tiempos muertos innecesarios. Asimismo, el promedio de trabajo en progreso sugiere que el sistema mantiene un nivel de actividad constante, con una proporción equilibrada de entidades en cola y en procesamiento.

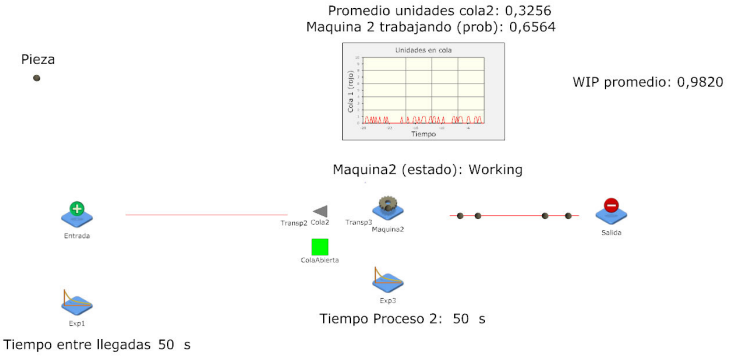


Figura 34. Resultados modelo M/M/1/K con cola de longitud finita en JaamSim

En términos generales, el modelo opera de manera eficiente, asegurando un flujo continuo y un buen nivel de utilización del servidor. Sin embargo, ajustes específicos en los parámetros del sistema podrían reducir aún más las acumulaciones en la cola y mejorar la eficiencia global sin comprometer la continuidad del proceso. Esto demuestra cómo el análisis detallado del sistema puede proporcionar información clave para la toma de decisiones en la optimización operativa.

3.7. M/M/1/∞ + M/M/1/b entorno fabricación dos máquinas y la segunda cola con longitud finita

Este modelo se clasifica como un M/M/1/∞ + M/M/1/b, caracterizado por tiempos de llegada y de servicio distribuidos exponencialmente (M/M), una cola infinita en la primera máquina (∞) y una cola con capacidad limitada (K o b) en la segunda máquina. Este sistema es ideal para estudiar el impacto de las restricciones de capacidad en la etapa final del proceso y su efecto en el flujo operativo.

El modelo funciona de la siguiente manera: las piezas llegan al sistema de forma aleatoria a través del EntityGenerator y se almacenan en la Cola1 si la Máquina 1 está ocupada. Una vez procesadas en la Máquina 1, las piezas se transfieren a la Cola2, que tiene una capacidad limitada a 1 pieza. Si esta cola está llena, las nuevas piezas son bloqueadas hasta que haya espacio disponible, lo cual es controlado mediante una expresión lógica en JaamSim que detiene el flujo hacia la cola al alcanzar su capacidad máxima. Finalmente, las piezas en la Cola2 son procesadas por la Máquina 2, que tiene un tiempo promedio de procesamiento mayor que la primera máquina. Después de ser procesadas, las piezas abandonan el sistema a través del EntitySink, donde se registran estadísticas clave del sistema. Se analizarán los componentes principales del modelo, las estadísticas generadas y su interpretación Fig. 35.

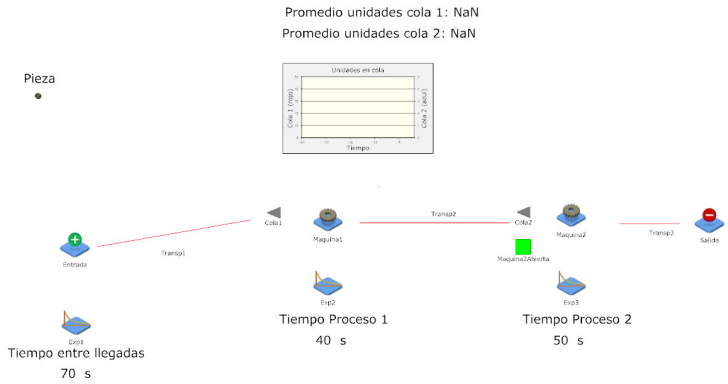


Figura 35. M/M/1/∞ + M/M/1/b entorno fabricación dos máquinas y la segunda cola con longitud finita en JaamSim

Con esta configuración, el modelo **M/M/1/∞ + M/M/1/K** reflejará un sistema con dos máquinas en serie, una cola ilimitada para la primera etapa, y una segunda cola con capacidad limitada a una pieza. El control mediante **ExpressionThreshold** garantizará que el flujo operativo se detenga cuando la segunda cola alcance su capacidad máxima, simulando restricciones físicas en el sistema.

3.7.1. Descripción general del modelo M/M/1/∞ + M/M/1/K con la segunda cola con longitud finita

El modelo titulado "M/M/1/∞ + M/M/1/K" representa un sistema de producción o servicio donde las piezas (o entidades) son generadas de forma aleatoria en un sistema en serie compuesto por dos máquinas, la producción de la primera máquina vendrá limitada por la restricción de capacidad finita de la cola de la segunda cola. Este modelo es ideal para analizar sistemas con restricciones de capacidad y variabilidad en los tiempos de llegada y servicio. A continuación, se detalla cada componente del modelo y su función dentro del flujo operativo:

1. **Entrada de piezas (EntityGenerator)**

   Es el componente encargado de generar las piezas (entidades) que ingresan al sistema. Estas llegan de forma aleatoria, siguiendo una distribución exponencial con un tiempo promedio entre llegadas configurado en 70 segundos. Este componente representa el inicio del proceso productivo, donde las piezas son creadas y enviadas a la primera cola para su posterior procesamiento.

2. **Primera cola (Cola1)**

   La **Cola1** es una cola de capacidad infinita en la que las piezas esperan su turno para ser procesadas por la **Máquina 1**. Este componente asegura que no se pierdan piezas en esta etapa, ya que admite un número ilimitado de entidades. La longitud de esta cola puede variar según la tasa de llegadas y el ritmo de procesamiento de la primera máquina.

3. **Máquina 1 (Server)**

   Esta máquina representa el primer paso del proceso, donde las piezas son procesadas una por una. Este servidor procesa las piezas de una en una con un tiempo aleatorio, definido por una distribución exponencial con una media de 40 segundos. Una vez procesadas, las piezas se transfieren a la **Cola2**, que actúa como un buffer para la segunda etapa.

4. **Segunda Cola (Cola2)**

   La **Cola2** es una cola con capacidad limitada a una sola pieza. Este componente está diseñado para regular el flujo hacia la **Máquina 2**. Si la cola está llena, las piezas procesadas por la **Máquina 1** serán bloqueadas hasta que la **Cola2** tenga espacio disponible. Este control de flujo se implementa mediante una expresión lógica en JaamSim que detiene la entrada de piezas a la cola cuando alcanza su capacidad máxima.

5. **Máquina 2 (Server)**

   Esta máquina representa la segunda etapa del proceso, donde las piezas pasan por un procesamiento adicional. Este servidor opera con un tiempo promedio de procesamiento mayor que la primera máquina, configurado en 50 segundos, y procesa cada pieza de manera individual. Una vez procesadas en esta etapa, las piezas son transferidas al componente de salida.

6. **Salida de piezas (EntitySink)**

   Es el punto final del flujo operativo, donde las piezas procesadas completan su recorrido. Este componente recopila estadísticas clave, como el número total de piezas procesadas y el tiempo promedio que estas pasan en el sistema. La salida marca el cierre del flujo operativo, permitiendo evaluar el rendimiento del sistema en su totalidad.

3.7.2. Asunciones modelo M/M/1/∞ + M/M/1/K con la segunda cola con longitud finita

1. La distribución de Poisson mide el número de llegadas por unidad de tiempo y los tiempos de servicio de las máquinas 1 y 2. La distribución del tiempo entre llegadas de un proceso de Poisson es la distribución exponencial negativa.
2. Tiempos de servicios de las máquinas 1 y 2 son aleatorios.
3. Factor de utilización $\rho<1$.
4. El sistema está en un estado estable.
5. El sistema está lleno, la cola 2 tiene una pieza almacenada; la tasa de llegadas es $\lambda= 0$.
6. La cantidad de clientes en el sistema no llega a ∞.
7. La cola 2 tiene una capacidad finita de 1 elemento.
8. Las piezas que llegan cuando la cola2 está llena son almacenadas en la cola 1.
9. Tiempo entre llegadas media = 70 segundos, con una distribución exponencial.
10. Tiempo de servicio de la máquina 1 presenta una media = 40 segundos, con una distribución exponencial.
11. Tiempo de servicio de la máquina 2 presenta una media = 50 segundos, con una distribución exponencial.

3.7.3. Configuración en JaamSim del modelo M/M/1/∞ + M/M/1/K con la segunda cola con longitud finita

A continuación, se detallan los pasos esenciales para implementar el modelo "M/M/1/∞ + M/M/1/K" en JaamSim, asegurando la correcta configuración de cada componente para reflejar la dinámica de un sistema con dos máquinas en serie y una segunda cola con capacidad limitada. Siga las instrucciones a continuación para configurar el modelo:

1. Entrada de piezas ("EntityGenerator")

- Arrastre el componente **"EntityGenerator"** a la ventana principal de trabajo desde la biblioteca de componentes.

- Configure los parámetros principales en el panel de propiedades:
  - **Entity Type:** defina el tipo de entidad como "Pieza".
  - **Arrival Time Distribution:** seleccione **Exponential Distribution** y configure el tiempo promedio entre llegadas (media) en **70 segundos**.
- Vincule el **EntityGenerator** al **ExpressionThreshold**:
  - En el panel de propiedades del **EntityGenerator**, localice la sección **OperatingThresholdList**.
  - Vincule este parámetro al componente **ExpressionThreshold**, que controlará el flujo de entrada en función de las condiciones de la cola.

2. Primera cola (“Queue1”)

- Agregue un componente **Queue** al modelo para representar la primera cola de espera (**Cola1**).
- Configure sus propiedades:
  - **Queue Capacity:** configure como ∞ para que esta cola tenga capacidad infinita.
- Asigne un nombre claro al componente, como **“Cola1”**, para facilitar su identificación en el flujo operativo.

3. Primera máquina (“Server1”)

- Arrastre un componente **Server** a la ventana principal de trabajo para representar la primera máquina (**Máquina 1**).
- Configure los parámetros principales:
  - **Service Time Distribution:** seleccione **Exponential Distribution** y configure el tiempo promedio de servicio (media) en **40 segundos**.
  - **Input Queue:** vincule la máquina a **Cola1** en el panel de conexiones.
- Asigne un nombre al componente, como **“Máquina 1”**.

4. Segunda cola (“Queue2”)

- Agregue un componente **Queue** al modelo para representar la segunda cola (**Cola2**).
- Configure los parámetros principales:
  - **Queue Capacity:** establezca la capacidad máxima en **1 unidad** para reflejar la limitación física de esta cola.
- Este componente actuará como un buffer limitado entre la primera y segunda máquina.

5. Control de flujo en la segunda cola (ExpressionThreshold)

- Arrastre un componente **ExpressionThreshold** a la ventana principal de trabajo para implementar el control de flujo hacia la segunda cola.

- Configure la condición lógica en el parámetro **ThresholdExpression**: [Cola2].QueueLength < 1
- Esta expresión asegura que el flujo hacia la **Cola2** se detendrá cuando su longitud alcance 1, bloqueando nuevas piezas hasta que haya espacio disponible.

6. Segunda máquina ("Server2")

- Arrastre otro componente **Server** a la ventana principal de trabajo para representar la segunda máquina (**Máquina 2**).
- Configure los parámetros principales:
  - **Service Time Distribution:** seleccione **Exponential Distribution** y configure el tiempo promedio de servicio (media) en **50 segundos**.
  - **Input Queue:** vincule la máquina a **Cola2** en el panel de conexiones.
  - En el panel **OperatingThresholdList**, vincule la máquina al **ExpressionThreshold** configurado anteriormente.
- Asigne un nombre al componente, como **"Máquina 2"**.

7. Salida de Piezas ("EntitySink")

- Agregue un componente **EntitySink** al modelo para representar el punto final del flujo operativo.
- Conecte este componente a la salida de la **Máquina 2**. No se requieren configuraciones adicionales más allá de la conexión.

8. Conexión de Componentes

- Conecte los componentes visualmente en la ventana principal de trabajo para reflejar el flujo operativo:
  - **EntityGenerator → Cola1 → Máquina 1 → Cola2 → Máquina 2 → EntitySink.**

Tabla 20. Configuración general modelo M/M/1/∞ + M/M/1/K con la segunda cola con longitud finita

Object	Keyword	Value
Entrada	KeyInputs/NextComponent	Transp1
Entrada	KeyInputs/InterArrivalTime	Exp1
Entrada	KeyInputs/PrototypeEntity	Pieza
Entrada	Thresholds	ColaAbierta
Máquina1	KeyInputs/NextComponent	Transp2
Máquina1	KeyInputs/WaitQueue	Cola1
Máquina1	KeyInputs/ServiceTime	Exp2
Máquina1	Thresholds	Maquina2Abierta

(continua)

(continuación)

Máquina2	KeyInputs/NextComponent	Transp2
Máquina2	KeyInputs/WaitQueue	Cola2
Máquina2	KeyInputs/ServiceTime	Exp3
Transp1	KeyInputs/NextComponent	Máquina1
Transp1	KeyInputs/TravelTime	500 s
Transp2	KeyInputs/NextComponent	Máquina2
Transp2	KeyInputs/TravelTime	0 s
Transp3	KeyInputs/NextComponent	Salida
Transp3	KeyInputs/TravelTime	500 s
Exp1	KeyInputs/UnitType	TimeUnit
Exp1	KeyInputs/RandomSeed	1
Exp1	KeyInputs/Mean	70 s
Exp2	KeyInputs/UnitType	TimeUnit
Exp2	KeyInputs/RandomSeed	2
Exp2	KeyInputs/Mean	40 s
Exp3	KeyInputs/UnitType	TimeUnit
Exp3	KeyInputs/RandomSeed	3
Exp3	KeyInputs/Mean	50 s
Maquin2Abierta	KeyInputs/OpenCondition	[Cola2].QueueLength<1

3.7.4. Análisis del modelo M/M/1/∞ + M/M/1/K con la segunda cola con longitud finita

Los resultados muestran que la longitud promedio de la Cola 1 es de 6,74 unidades, lo que indica una acumulación relativamente alta en esta etapa debido a la diferencia en las tasas de llegada y procesamiento. Por otro lado, la longitud promedio de la Cola 2 es significativamente más baja, con un valor de 0,57 unidades, lo que refleja que el control de flujo implementado en esta cola con capacidad finita está funcionando adecuadamente para evitar grandes acumulaciones.

La alta utilización de ambas máquinas sugiere un uso eficiente de los recursos. La Máquina 1, con un tiempo de procesamiento promedio menor (40 segundos), alimenta de manera constante la segunda etapa, mientras que la Máquina 2, con un tiempo de procesamiento ligeramente mayor (50 segundos), procesa las piezas sin generar un cuello de botella crítico gracias a la regulación de entrada a la segunda cola. Esto asegura un flujo constante con acumulaciones controladas.

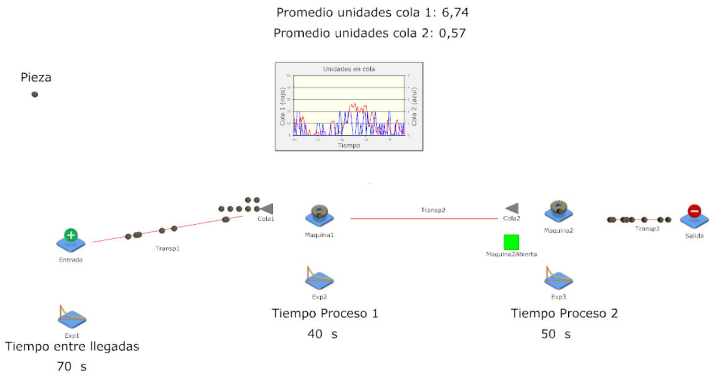


Figura 36. Resultados modelo M/M/1/K con cola de longitud finita en JaamSim

En términos generales, el modelo opera eficientemente, garantizando la continuidad del flujo y un uso balanceado de las máquinas. Sin embargo, ajustes específicos en el tiempo entre llegadas o en los tiempos de procesamiento podrían ayudar a equilibrar aún más las colas, reduciendo la acumulación en la primera etapa y optimizando el tiempo total en el sistema. Este análisis destaca cómo la simulación puede proporcionar información valiosa para la toma de decisiones estratégicas en la optimización de procesos productivos.

Referencias bibliograficas

Abdelmegid, M. A., O'Sullivan, M., González, V. A., Walker, C. G., & Poshdar, M. (2022). A case study on the use of a conceptual modeling framework for construction simulation. Simulation, 98(5), 433–460. https://doi.org/10.1177/00375497211056087

Bautista, J., Corominas, A., Companys, R. (2011). Métodos Cuantitativos de Organización Industrial (Apuntes): Sistemas con esperas: Teoría de Colas y Simulación. Universistat Politècnica de Catalunya. Cátedra Nissan.

García Sabater, José P., & Empresas, D. D. O. De. (2016). Aplicando Teoría de Colas en Dirección de Operaciones. 1–86.

Garcia Sabater, Jose P. (2020) Gestión de los Tiempos de Espera. http://hdl.handle.net/10251/137896

Hashash, Y. M. A., Musgrove, M. I., Harmon, J. A., Ilhan, O., Groholski, D. R., Phillips, C. A., & Park, D. (2020). User Manual. May, 1–169. DEEPSOIL 7, User Manual

King, D. H., Street, H., Harrison, H. S., & Street, H. (n.d.). "JaamSim " Open-Source Simulation Software. https://doi.org/10.5555/2557668.2557669

Kloock-Schreiber, D., Siqueira, R., Gembarski, P. C., & Lachmayer, R. (2020). DISCRETE-EVENT SIMULATION for SPECIFICATION DESIGN of PRODUCTS in PRODUCT-SERVICE SYSTEMS. Proceedings of the Design Society: DESIGN Conference, 1, 255–264. https://doi.org/10.1017/dsd.2020.295

Mora, J. (2011). Instrumentos Estadísticos Avanzados Facultad Ciencias Económicas y Empresariales Departamento de Economía Aplicada Profesor: Santiago de la Fuente Fernández. https://www.estadistica.net/INVESTIGACION/TEORIA-COLAS.pdf

Vieira, A. A. C., Dias, L. M. S., Santos, M. Y., Pereira, G. A. B., & Oliveira, J. A. (2019). A ranking of the most known freeware and open source discrete-event simulation tools. 31st European Modeling and Simulation Symposium, EMSS 2019, c, 200–209. https://doi.org/10.46354/i3m.2019.emss.029

Xanthopoulos, A. S., & Koulouriotis, D. E. (2021). A comparative study of different pull control strategies in multi-product manufacturing systems using discrete event simulation. Advances in Production Engineering And Management, 16(4), 473–484. https://doi.org/10.14743/APEM2021.4.414